A New Representation of Skeleton Sequences for 3D Action Recognition

Qiuhong Ke[1], Mohammed Bennamoun[1], Senjian An[1], Ferdous Sohel[2], Farid Boussaid[1]

[1]The University of Western Australia [2]Murdoch University

qiuhong.ke@research.uwa.edu.au

{mohammed.bennamoun,senjian.an,farid.boussaid}@uwa.edu.au

f.sohel@murdoch.edu.au

Abstract

*This paper presents a new method for 3D action recognition with skeleton sequences (*i.e.*, 3D trajectories of human skeleton joints). The proposed method first transforms each skeleton sequence into three clips each consisting of several frames for spatial temporal feature learning using deep neural networks. Each clip is generated from one channel of the cylindrical coordinates of the skeleton sequence. Each frame of the generated clips represents the temporal information of the entire skeleton sequence, and incorporates one particular spatial relationship between the joints. The entire clips include multiple frames with different spatial relationships, which provide useful spatial structural information of the human skeleton. We propose to use deep convolutional neural networks to learn long-term temporal information of the skeleton sequence from the frames of the generated clips, and then use a Multi-Task Learning Network (MTLN) to jointly process all frames of the generated clips in parallel to incorporate spatial structural information for action recognition. Experimental results clearly show the effectiveness of the proposed new representation and feature learning method for 3D action recognition.*

1. Introduction

3D skeleton data records the trajectories of human skeleton joints and is robust to illumination changes and invariant to camera views [14]. With the prevalence of highly-accurate and affordable devices, action recognition based on 3D skeleton sequence has been attracting increasing attention [49, 42, 6, 37, 54, 26, 22, 46, 19]. In this paper, we focus on skeleton-based 3D action recognition.

To recognize a video action, the temporal information of the sequence needs to be exploited to understand the dynamics of human postures [29, 9, 44, 8, 20]. For skeleton data, the spatial structure of the human skeleton is also an important clue for action recognition [54]. Each skeleton sequence provides only the trajectory of human skeleton joints. The time series of the joints can be used in recurrent neural networks (RNNs) with Long-Short Term Memory (LSTM) neurons [11, 12] to explore the spatial structure and temporal structure of the skeleton sequence for action recognition [6, 41, 54, 37, 26]. Although LSTM networks are designed to explore the long-term temporal dependency problem, it is still difficult for LSTM to memorize the information of the entire sequence with many timesteps [48, 13]. In addition, it is also difficult to construct deep LSTM to extract high-level features [35, 30].

Convolutional neural networks (CNNs) [24] nowadays have achieved great success in image classification [2, 3, 23, 38, 39, 50, 21]. However, for video action recognition, it lacks the capacity to model the long-term temporal dependency of the entire video [45]. In this paper, instead of directly exploring the long-term temporal information from the skeleton sequences, we first represent the skeleton sequences as clips consisting of only a few frames. With the generated clips, the long-term temporal structure of the skeleton sequence can be effectively learned by using deep CNNs to process the frame images of the generated clips. In addition, the spatial structural information of the human skeleton can be exploited from the entire clips.

More specifically, for each skeleton sequence, we generate three clips corresponding to the three channels of the cylindrical coordinates of the skeleton sequence. Each clip consists of four frames, which are generated by computing the relative positions of the joints to four reference joints. Each frame of the clips describes the temporal information of the entire skeleton sequence, and includes one particular spatial relationship between the joints. The entire clips aggregate multiple frames with different spatial relationships, providing important information of the spatial structure of the skeleton joints.

Since the temporal information of a skeleton sequence is incorporated in the frames of the generated clips, the long-term temporal structure of the skeleton sequence can be learned by extracting features from the frames of the generated clips. More specifically, each frame of the generated clips is fed to a deep CNN to extract a CNN feature. Then the three CNN features of the three clips at the same time-step (See Figure 1) are concatenated into one feature vector. Consequently, four feature vectors are extracted from all the time-steps. Each feature vector represents the temporal information of the skeleton sequence and one particular spatial relationship between the joints. The feature vectors of different time-steps represent different spatial relationships with intrinsic relationships among them. This paper proposes to utilize the intrinsic relationships among different feature vectors for action recognition using a Multi-Task Learning Network (MTLN). Multi-task learning aims at improving the generalization performance by jointly training multiple related tasks and utilizing their intrinsic relationships [1]. In the proposed MTLN, the classification of each feature vector is treated as a separate task, and the MTLN jointly learns multiple classifiers each from one feature vector and outputs multiple predictions, each corresponding to one task. All the feature vectors of the same skeleton sequence have the same label as the skeleton sequence. During training, the loss value of each task is individually computed using its own class scores. Then the loss values of all tasks are summed up to define the total loss of the network which is then used to learn the network parameters. During testing, the class scores of all tasks are averaged to form the final prediction of the action class. Multi-task learning simultaneously solves multiple tasks with weight sharing, which can improve the performance of individual tasks [1].

The main contributions of this paper are summarized as follows. **(1)** We propose to transform each skeleton sequence to a new representation, *i.e.*, three clips, to allow global long-term temporal modelling of the skeleton sequence by using deep CNNs to learn hierarchical features from frame images. **(2)** We introduce a MTLN to process all the CNN features of the frames in the generated clips, thus to learn the spatial structure and the temporal information of the skeleton sequence. The MTLN improves the performance by utilizing intrinsic relationships among different frames of the generated clips. Our experimental results demonstrate that MTLN performs better than concatenating or pooling the features of the frames (See Section 4.3). **(3)** The proposed method achieves the state-of-the-art performance on three skeleton datasets, including the large scale NTU RGB+D dataset [37].

2. Related Works

In this section, we cover the relevant literature of skeleton-based action recognition methods using hand-crafted features or using deep learning networks.

Hand-crafted Features In [17], the covariance matrices of the trajectories of the joint positions are computed over hierarchical temporal levels to model the skeleton sequences. In [43], the pairwise relative positions of each joint with other joints are computed to represent each frame of the skeleton sequences, and Fourier Temporal Pyramid (FTP) is used to model the temporal patterns. In [51], the pairwise relative positions of the joints are also used to characterize posture features, motion features, and offset features of the skeleton sequences. Principal Component Analysis (PCA) is then applied to the normalized features to compute EigenJoints as representations. In [49], histograms of 3D joint locations are computed to represent each frame of the skeleton sequences, and HMMs are used to model the temporal dynamics. In [42], the rotations and translations between various body parts are used as representations, and a skeleton sequence is modelled as a curve in the Lie group. The temporal dynamics are modelled with FTP.

Deep Learning Methods In [6], the skeleton joints are divided into five sets corresponding to five body parts. They are fed into five LSTMs for feature fusion and classification. In [54], the skeleton joints are fed to a deep LSTM at each time slot to learn the inherent co-occurrence features of skeleton joints. In [37], the long-term context representations of the body parts are learned with a part-aware LSTM. In [26], both the spatial and temporal information of skeleton sequences are learned with a spatial temporal LSTM. A Trust Gate is also proposed to remove noisy joints. This method achieves the state-of-the-art performance on the NTU RGB+D dataset [37].

3. Proposed Method

An overall architecture of the proposed method is shown in Figure 1. The proposed method starts by generating clips of skeleton sequences. A skeleton sequence of any length is transformed into three clips each consisting of several gray images. The generated clips are then fed to a deep CNN model to extract CNN features which are used in a MTLN for action recognition.

3.1. Clip Generation

Compared to RGB videos which consist of multiple frame images, skeleton sequences only provide the trajectories of the 3D coordinates. This paper proposes to transform the original skeleton sequence to a collection of clips each consisting of several images, thus to allow spatial temporal feature learning using deep neural networks. Intuitively, one could represent the content of each frame of the skeleton sequence as an image to generate a video. However, if the skeleton sequence has many frames, this method will result in a long video of which the temporal dynamics will be difficult to learn. In addition, each frame of the generated

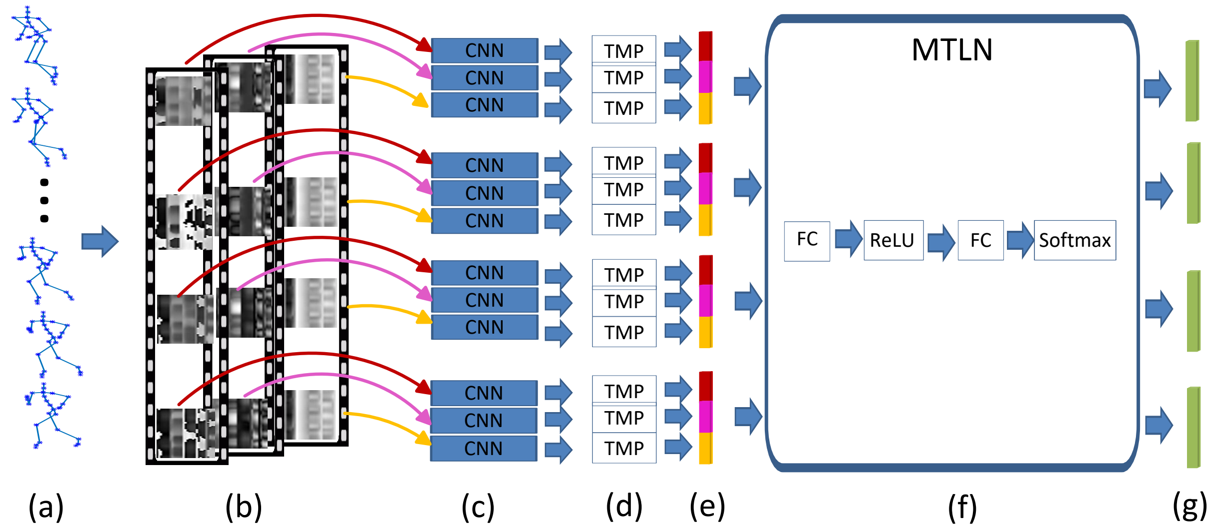

Figure 1. Architecture of the proposed method. Given a skeleton sequence (a), three clips (b) corresponding to the three channels of the cylindrical coordinates are generated. A deep CNN model (c) and a temporal mean pooling (TMP) layer (d) are used to extract a compact representation from each frame of the clips (see Figure 3 for details). The output CNN representations of the three clips at the same time-step are concatenated, resulting four feature vectors (e). Each feature vector represents the temporal information of the skeleton sequence and a particular spatial relationship of the skeleton joints. The proposed MTLN (f) which includes a fully connected (FC) layer, a rectified linear unit (ReLU), another FC layer and a Softmax layer jointly processes the four feature vectors in parallel and outputs four sets of class scores (g), each corresponding to one task of classification using one feature vector. During training, the loss values of the four tasks are summed up to define the loss value of the network used to update the network parameters. For testing, the class scores of the four tasks are averaged to generate the final prediction of the action class.

video will also be very sparse as the number of the skeleton joints is small. To overcome this problem, we propose to represent the temporal dynamics of the skeleton sequence in a frame image, and then use multiple frames to incorporate different spatial relationships between the joints. An advantage of this method is that for any skeleton sequence of any length, the generated clips contain the same number of frames and the long-term temporal information of the original skeleton sequence can be effectively captured with the powerful CNN representations of the frame images in the generated clips.

As shown in Figure 2, for a skeleton sequence, the skeleton joints of each frame are first arranged as a chain by concatenating the joints of each body part. Considering that the relative positions between joints provide more useful information than their absolute locations (*e.g.*, the relative location of the hand to the shoulder in "pushing"), four reference joints, namely, the left shoulder, the right shoulder, the left hip and the right hip, are respectively used to compute relative positions of the other joints, thus to incorporate different spatial relationships between joints and provide useful structural information of the skeleton. These four joints are selected as reference joints due to the fact that they are stable in most actions. They can thus reflect the motions of the other joints. Although the base of the spine is also stable, it is close to the left hip and the right hip. It is therefore discarded to avoid information redundancy. By combing the relative joints of all the frames, four 2D arrays with dimension $(m-1) \times t$ are generated (m is the number of skeleton joints in each frame and t is the number of frames of the skeleton sequence). The relative positions of joints in the 2D arrays are originally described with 3D Cartesian coordinates. Considering that the cylindrical coordinates are more useful to analyse the motions as each human body utilizes pivotal joint movements to perform an action, the 3D Cartesian coordinates are transformed to cylindrical coordinates in the proposed representation of skeleton sequences. The cylindrical coordinates have been used to extract view-invariant motion features for action recognition in [47]. The four 2D arrays corresponding to the same channel of the 3D cylindrical coordinates are transformed to four gray images by scaling the coordinate values between 0 to 255 using a linear transformation. A clip is then constructed with the four gray images. Consequently, three clips are generated from the three channels of the 3D coordinates of the four 2D arrays.

3.2. Clip Learning

Each frame of the generated clips describes the temporal dynamics of all frames of the skeleton sequence and one

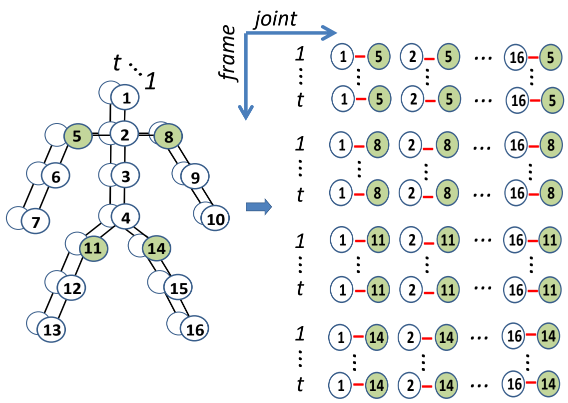

Figure 2. Clip Generation of a skeleton sequence. The skeleton joints of each frame are first arranged as a chain by concatenating the joints of each body part (*i.e.*, 1-2-3-...-16). Four reference joints shown in green (*i.e.*, left shoulder 5, right shoulder 8, left hip 11 and right hip 14) are then respectively used to compute relative positions of the other joints to incorporate different spatial relationships between the joints. Consequently, four 2D arrays are obtained by combining the relative positions of all the frames of the skeleton sequence. The relative position of each joint in the 2D arrays is described with cylindrical coordinates. The four 2D arrays corresponding to the same channel of the coordinates are transformed to four gray images and as a clip. Thus three clips are generated from the three channels of the cylindrical coordinates of the four 2D arrays.

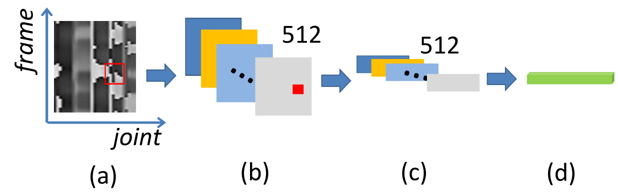

Figure 3. Temporal mean pooling of the CNN feature maps. (a) An input frame of the generated clips, for which the rows correspond to the different frames of the skeleton sequence and the columns correspond to the different vectors generated from the joints. (b) Output feature maps of the conv5_1 layer. The size is $14 \times 14 \times 512$. Each activation (shown in red) of the feature map is a feature correspond to the local region of the original image (shown with a red square). (c) Temporal features of all joints of the skeleton sequence, which are obtained by applying mean pooling to each feature map in the row (temporal) dimension. (d) Output feature, which is achieved by concatenating all the feature maps in (c).

particular spatial relationship between the skeleton joints in one channel of the cylindrical coordinates. Different frames of the generated clip describe different spatial relationships and there exists intrinsic relationships among them. A deep CNN is first leveraged to extract a compact representation from each frame of the generated clips to exploit the long-term temporal information of the skeleton sequence. Then the CNN features of all frames of the generated clips are jointly processed in parallel using multi-task learning, thus to utilize their intrinsic relationships to learn the spatial temporal information for 3D action recognition.

3.2.1 Temporal Pooling of CNN Feature Maps

To learn the features of the generated clips, a deep CNN is firstly employed to extract a compact representation of each frame of the clips. Since each frame describes the temporal dynamics of the skeleton sequence, the spatial invariant CNN feature of each frame could thus represent the robust temporal information of the skeleton sequence.

Given the generated clips, the CNN feature of each frame is extracted with the pre-trained VGG19 [38] model. The pre-trained CNN model is leveraged as a feature extractor due to the fact that the CNN features extracted by the models pre-trained with ImageNet [34] are very powerful and have been successfully applied in a number of cross-domain applications [5, 10, 33, 15]. In addition, current skeleton datasets are either too small or too noisy to suitably train a deep network. Although the frames of the generated clips are not natural images, they could still be fed to the CNN model pre-trained with ImageNet [34] for feature extraction. The similarity between a natural image and the generated frames is that both of them are matrices with some patterns. The CNN models trained on the large image dataset can be used as a feature extractor to extract representations of the patterns in matrices. The learned representations are generic and can be transferred to novel tasks from the original tasks [52, 27].

The pre-trained VGG19 [38] model contains 5 sets of convolutional layers conv1, conv2, ..., conv5. Each set includes a stack of 2 or 4 convolutional layers with the same kernel size. Totally there are 16 convolutional layers and three fully connected layers in the network. Although deep neural networks are capable of learning powerful and generic features which can be used in other novel domains, the features extracted from the different layers have different transferability. Particularly, the features in earlier layers are more generic, while in later layers, the features are more task-specific, which largely rely on the original classes and dataset. The features of the later layers are thus less suitable than those of the earlier layers to transfer to other domains [52, 27]. Therefore, this paper adopts a compact representation that is derived from the activations of the convolutional layer to exploit the temporal information of a skeleton sequence. The feature maps in the convolutional layer have been successfully applied for action recognition and image retrieval [31, 32]. Specifically, the last 3 convolutional layers and fully connected layers of the network are discarded. Each frame image of the three clips is scaled to 224×224, and is then duplicated three times to formulate a color image, so that it can be fed to the network. The output of the convolutional layer conv5_1 is used as the representation of the input frame, which is a 3D tensor with size $14\times14\times512$, *i.e.*, 512 feature maps with size 14×14.

The rows of the generated frame correspond to different frames of a skeleton sequence. The dynamics of the row features of the generated image therefore represents the temporal evolution of the skeleton sequence. Meanwhile, the activations of each feature map in the conv5_1 layer are the local features corresponding to the local regions in the original input image [31]. The temporal information of the sequence can thus be extracted from the row features of the feature maps. More specifically, the feature maps are processed with temporal mean pooling with kernel size 14×1, *i.e.*, the pooling is applied over the temporal, or row dimension, thus to generate a compact fusion representation from all temporal stages of the skeleton sequence. Let the activation at the i^{th} row and the j^{th} column of the k^{th} feature map be $x^k_{i,j}$. After temporal mean pooling, the output of the k^{th} feature map is given by:

$$\begin{aligned} \mathbf{y^k} &= [y^k_1, \cdots, y^k_j, \cdots, y^k_{14}] \\ y^k_j &= \frac{1}{14}\sum_{i=1}^{14} \max(0, x^k_{i,j}) \end{aligned} \tag{1}$$

The outputs of all feature maps (512) are concatenated to form a 7168D ($14 \times 512 = 7168$) feature vector, which represents the temporal dynamics of the skeleton sequence in one channel of the cylindrical coordinates.

3.2.2 Multi-Task Learning Network (MTLN)

As shown in Figure 1(e), the three 7168D features of the three clips at the same time-step are concatenated to form a feature vector, generating four feature vectors in total. Each feature vector represents the temporal dynamics of the skeleton sequence and includes one particular spatial relationship between the joints in one of three cylindrical coordinates. The four feature vectors have intrinsic relationships between each other. An MTLN is then proposed to jointly process the four feature vectors to utilize their intrinsic relationships for action recognition. The classification of each feature vector is treated as a separate task with the same classification label of the skeleton sequence.

The architecture of the network is shown in Figure 1(f). It includes two fully connected (FC) layers and a Softmax layer. Between the two FC layers there is a rectified linear unit (ReLU) [28] to introduce an additional non-linearity. Given the four features as inputs, the MTLN generates four frame-level predictions, each corresponding to one task. During training, the class scores of each task are used to compute a loss value. Then the loss values of all tasks are summed up to generate the final loss of the network used to learn the network parameters. During testing, the class scores of all tasks are averaged to form the final prediction of the action class. The loss value of the k^{th} task ($k = 1, \cdots, 4$) is given by Equation 2.

$$\begin{aligned} \ell_k(\mathbf{z_k}, \mathbf{y}) &= \sum_{i=1}^{m} y_i \left(-log \left(\frac{\exp z_{ki}}{\sum_{j=1}^{m} \exp z_{kj}} \right) \right) \\ &= \sum_{i=1}^{m} y_i \left(log \left(\sum_{j=1}^{m} \exp z_{kj} \right) - z_{ki} \right) \end{aligned} \tag{2}$$

where $\mathbf{z_k}$ is the vector fed to the Softmax layer generated from the k^{th} input feature, m is the number of action classes and y_i is the ground-truth label for class i. The final loss value of the network is computed as the sum of the four individual losses, as shown below in Equation 3:

$$\mathcal{L}(Z, \mathbf{y}) = \sum_{k=1}^{4} \ell_k(\mathbf{z_k}, \mathbf{y}) \tag{3}$$

where $Z = [\mathbf{z_1}, \cdots, \mathbf{z_4}]$.

4. Experiments and Analysis

The proposed method is tested on three skeleton action datasets: NTU RGB+D dataset [37], SBU kinect interaction dataset [53] and CMU dataset [4].

The main ideas of the proposed method **Clips + CNN + MTLN** are 1) generating three clips (each clip consists of four frames) from a skeleton sequence, 2) using CNNs to learn global long-term temporal information of the skeleton sequence from each frame of the generated clips, and 3) using MTLN to jointly train the CNN features of the four frames of the clips to incorporate the spatial structural information for action recognition.

We also conducted the following baselines to demonstrate the advantages of the proposed method:

Coordinates + FTP In this baseline, the Fourier Temporal Pyramid (FTP) [43] is applied to the 3D coordinates of the skeleton sequences to extract temporal features for action recognition. This baseline is used to show the benefits of using CNNs for long-term temporal modelling of the skeleton sequences.

Frames + CNN In this baseline, the CNN features of single frames instead of the entire generated clips are used for action recognition. In other words, only one feature vector shown in Figure 1(e) is used to train a neural network for classification. Thus the loss value of the network is given by Equation 2. The average accuracy of the four features is provided. This baseline is used to show the benefits of using the entire generated clips to incorporate the spatial structural information for action recognition.

Clips + CNN + Concatenation In this baseline, the CNN features of all frames of the generated clips are concatenated before performing action recognition. In other words, the four feature vectors shown in Figure 1(e) are concatenated and then fed to a neural network for classification. This baseline is used to show the benefits of using MTLN to process the features of the entire clips in parallel.

Clips + CNN + Pooling In this baseline, max pooling is applied to the CNN features of all frames of the generate clips before performing action recognition. Same as Clips + CNN + Concatenation, this baseline is also used to show the benefits of using MTLN.

4.1. Datasets

NTU RGB+D Dataset [37] To the best of our knowledge, this dataset is so far the largest skeleton-based human action dataset, with more than 56000 sequences and 4 million frames. There are 60 classes of actions performed by 40 distinct subjects, including both one-person daily actions (e.g., clapping, reading, writing) and two-person interactions (*e.g.*, handshaking, hug, pointing). These actions are captured by three cameras, which are placed at different locations and view points. In total, there are 80 views for this dataset. In this dataset, each skeleton has 25 joints. The 3D coordinates of the joints are provided. Due to the large view point, intra-class and sequence length variations, the dataset is very challenging.

SBU Kinect Interaction Dataset [53] This dataset was collected using the Microsoft Kinect sensor. It contains 282 skeleton sequences and 6822 frames. In this dataset, each frame contains two persons performing an interaction. The interactions include approaching, departing, kicking, punching, pushing, hugging, shaking hands and exchanging. There are 15 joints for each skeleton. This dataset is challenging due to the fact that the joint coordinates exhibit low accuracy [53].

CMU Dataset [4] This dataset contains 2235 sequences and about 1 million frames. For each skeleton, the 3D coordinates of 31 joints are provided. The dataset has been categorized into 45 classes [54]. All of the actions are performed by only one person. The dataset is very challenging due to the large sequence length variations and intra-class diversity.

4.2. Implementation Details

For all datasets, the clips are generated with all frames of the original skeleton sequence without any pre-processing such as normalization, temporal down-sampling or noise filtering. The proposed method was implemented using the MatConvNet toolbox [40]. The number of the hidden unit of the first FC layer is set to 512. For the second FC layer (*i.e.*, the output layer), the number of the unit is the same as the number of the action classes in each dataset. The network is trained using the stochastic gradient descent algorithm. The learning rate is set to 0.001 and batch size is set to 100. The training is stopped after 35 epochs. The performance of the proposed method on each dataset is compared with existing methods using the same testing protocol.

4.3. Results

NTU RGB+D Dataset As in [37], the evaluation on this dataset is performed with two standard protocols, *i.e.*, cross-subject evaluation and cross-view evaluation. In cross-subject evaluation, the sequences of 20 subjects are used for training and the data from 20 other subjects are used for testing. In cross-view evaluation, the sequences captured by two cameras are used for training and the rest are used for testing.

The results are shown in Table 1. It can be seen that the proposed method performs significantly better than others in both cross-subject and cross-view protocols. The accuracy of the proposed method is 79.57% when tested with the cross-subject protocol. Compared to the previous state-of-the-art method (ST-LSTM + Trust Gate [26]), the performance is improved by 10.37%. When tested with the cross-view protocol, the accuracy is improved from 77.7% to 84.83%.

The improved performance of the proposed method is due to the novel clip representation and feature learning method. As shown in Table 1, Frames + CNN achieves an accuracy of about 75.73% and 79.62% for the two testing protocols, respectively. The performances are much better than Coordinates + FTP. Compared to extracting temporal features of skeleton sequences with FTP and native 3D coordinates, using CNN to learn the temporal information of skeleton sequences from the generated frames is more robust to noise and temporal variations due to the convolution and pooling operators, resulting in better performances. From Table 1, it can also be seen that Frames + CNN also performs better than the previous state-of-the-art method. It clearly shows the effectiveness of the CNN features of the proposed clip representation. The performances are improved by learning entire clips with CNN and MTLN (*i.e.*, Clips + CNN + MTLN). The improvements are about 4% and 5% for the two testing protocols, respectively. It can also be seen that the proposed MTLN (*i.e.*, Clips + CNN + MTLN) performs better than feature concatenation (*i.e.*, Clips + CNN + concatenation) and pooling (*i.e.*, Clips + CNN + pooling). Frames + CNN, Clips + CNN + concatenation and Clips + CNN + pooling can be viewed as a single-task method, while using MTLN to process multiple frames of the generated clips in parallel utilizes their intrinsic relationships and incorporates the spatial structural information, which improves the performance of the single-

Table 1. Performance on the NTU RGB+D dataset.

Methods	Accuracy	
	Cross Subject	Cross View
Lie Group [42]	50.1%	52.8%
Skeletal Quads [7]	38.6%	41.4%
Dynamic Skeletons [16]	60.2%	65.2%
Hierarchical RNN [6]	59.1%	64.0%
Deep RNN [37]	59.3%	64.1%
Deep LSTM [37]	60.7%	67.3%
Part-aware LSTM [37]	62.9%	70.3%
ST-LSTM [26]	65.2%	76.1%
ST-LSTM + Trust Gate [26]	69.2%	77.7%
Coordinates + FTP	61.06%	74.64%
Frames + CNN	75.73%	79.62%
Clips + CNN + Concatenation	77.05%	81.11%
Clips + CNN + Pooling	76.37%	80.46%
Clips + CNN + MTLN	**79.57%**	**84.83%**

Table 2. Performance on the SBU kinect interaction dataset.

Methods	Accuracy
Raw Skeleton [53]	49.7%
Joint Feature [18]	86.9%
CHARM [25]	83.9%
Hierarchical RNN [6]	80.35%
Deep LSTM [54]	86.03%
Deep LSTM + Co-occurrence [54]	90.41%
ST-LSTM [26]	88.6%
ST-LSTM + Trust Gate [26]	93.3%
Coordinates + FTP	79.75%
Frames + CNN	90.88%
Clips + CNN + Concatenation	92.86%
Clips + CNN + Pooling	92.26%
Clips + CNN + MTLN	**93.57%**

Table 3. Performance on the CMU dataset.

Methods	Accuracy	
	CMU subset	CMU
Hierarchical RNN [6]	83.13%	75.02%
Deep LSTM [54]	86.00%	79.53%
Deep LSTM + Co-occurrence [54]	88.40%	81.04%
Coordinates + FTP	83.44%	73.61%
Frames + CNN	91.53%	85.36%
Clips + CNN + Concatenation	90.97%	85.76%
Clips + CNN + Pooling	90.66%	85.56%
Clips + CNN+ MTLN	**93.22%**	**88.30%**

task method for action recognition.

SBU Kinect Interaction Dataset As in [53], the evaluation of this dataset is a 5-fold cross validation, with the provided training/testing splits. Each frame of the skeleton sequences contains two separate human skeletons. In this case, the two skeletons are considered as two data samples and the clip generation and feature extraction are conducted separately for the two skeletons. For testing, the prediction of actions is obtained by averaging the classification scores of the two samples.

Considering that the number of samples in this dataset is too small, data augmentation is performed to increase the number of samples. More specifically, each frame image of the generated clips are resized to 250×250, and then random patches with size of 224×224 are cropped from the original image for feature learning using CNN. For this dataset, 20 sub-images are cropped and the total data samples are extended to 11320.

The comparisons of the proposed method with other methods are shown in Table 2. Similar to the NTU RGB+D dataset, CNN features perform better than FTP to learn the temporal information. It can be seen that when using CNN features of individual frames, the accuracy is 90.88%, which is similar to the Deep LSTM + Co-occurrence method [54]. When incorporating the CNN features of the entire clips using concatenation and pooling methods, the performance is improved by about 2%. The performance is improved to 93.57% when learning the entire clips with MTLN. It clearly shows the benefit of using MTLN to learn the CNN features entire clips.

Since the joint positions of this dataset are not very accurate [53], existing methods including HBRNN [6] and Co-occurrence LSTM [54] remove the joint noise by smoothing the position of each joint using the Svaitzky-Golay filter [36]. In [26], a Trust Gate is introduced to remove the noisy joints and this improves the accuracy from 88.6% to 93.3%. Our method does not perform any pre-processing to handle the noisy joints, but still performs better than all the others. It clearly shows that the features learned from the generated clips are robust to noise due to the convolution and pooling operators of the deep network.

CMU Dataset As in [54], for this dataset, the evaluation is conducted on both the entire dataset with 2235 sequences, and a selected subset of 664 sequences. The subset includes 8 classes of actions, , *i.e.*, basketball, cartwheel, getup, jump, pickup, run, sit and walk back. For the entire dataset, the testing protocol is 4-fold cross validation, and for the subset, it is evaluated with 3-fold cross validation. The training/tesing splits of the different folds are provided by [54].

Similar to the SBU kinect interaction dataset, data augmentation is also conducted on CMU dataset. For the entire dataset, each frame image is used to generate 5 more images and the total data samples are extended to 11175, and for the subset, the total samples are extended to 13280, which is 20 times of the original number.

The results are shown in Table 3. It can be seen that the performance of the proposed method is much better than previous state-of-the-art methods on both the subset and the entire set. When tested on the subset, the accuracy of the proposed method was about 93.22%, which is about 5% better than the previous method [54]. The performance on the entire dataset is improved from 81.04% to 88.3%.

4.4. Discussions

Three gray clips or one color clip? As shown in Figure 1, the frames of the three generated clips are gray images, each corresponding to only one channel of the cylindrical coordinates. Each frame is duplicated three times to formulate a color image for CNN feature learning. The output CNN features of the three channels are concatenated in a feature vector for action recognition. A simple alternative is to generate a color clip with three channels of the cylindrical coordinates, and then extract a single CNN feature from the color frame for action recognition. When this was tested on CMU dataset, the performance is 84.67%, which is about 4% worse than the proposed method. This is perhaps due to the fact that the relationship of the three generated channels is different from that of the RGB channels of natural color images. The RGB channels are arranged in sequence and there is no matching order between 3D coordinates and RGB channels.

The more frames, the better performance? This paper uses only four reference joints to generate clips, each having four frames. When 6 more joints are selected to generate more frames, *i.e.*, the head, the left hand, the right hand, the left foot, the right foot and the hip, the performance does not improve. When tested on CMU data, the performance is 86.01%, which is about 2% worse than the proposed method. This is due to the fact that the other joints are not as stable as the selected four joints, which can introduce noise.

Cartesian coordinates or cylindrical coordinates? As mentioned in Section 3.1, the 3D Cartesian coordinates of the vectors between the reference joints and the other joints are transformed to cylindrical coordinates to generate clips. We found that when using the original Cartesian coordinates for clip generation and action recognition, the performance drops. When tested on CMU dataset, the accuracy is 86.21%, which is about 2% worse than the proposed method. The cylindrical coordinates are more useful than the Cartesian coordinates to analyse the motions as each human skeleton utilizes pivotal joint movements to perform an action.

Features in different layers As mentioned in Section 3.2.1, the feature maps in conv5_1 layer of the pre-trained CNN model is adopted as the representation of each input image. We found that using the features in the earlier layers decreased the performance. When using the features of the conv4_1 layer, the accuracy on CMU dataset is 84.59%, which is about 4% worse than the proposed method. This is perhaps due to the fact that the features in the earlier layers are not deep enough to capture the salient information of the input image. We also found that using the features in the later layers made the performance worse. When using the features of the fc6 layer, the accuracy on CMU dataset is 83.52%, which is about 5% worse than the proposed method. This is because the features in the later layers are more task-specific, which largely rely on the original classes and dataset. The features of the later layers are thus less suitable than those of the earlier layers to transfer to other domains [52, 27].

5. Conclusion

In this paper, we have proposed to transform a skeleton sequence to three video clips for robust feature learning and action recognition. We proposed to use a pre-trained CNN model followed by a temporal pooling layer to extract a compact representation of each frame. The CNN features of the three clips at the same time-step are concatenated in a single feature vector, which describes the temporal information of the entire skeleton sequence and one particular spatial relationship between the joints. We then propose an MTLN to jointly learn the feature vectors at all the time-steps in parallel, which utilizes their intrinsic relationships and improves the performance for action recognition. We have tested the proposed method on three datasets, including NTU RGB+D dataset, SBU kinect interaction dataset and CMU dataset. Experimental results have shown the effectiveness of the proposed new representation and feature learning method.

6. Acknowledgment

This work was partially supported by Australian Research Council grants DP150100294, DP150104251, and DE120102960. This paper used the NTU RGB+D Action Recognition Dataset made available by the ROSE Lab at the Nanyang Technological University, Singapore.

References

[1] R. Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[2] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[3] D. Ciregan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012.

[4] CMU. CMU graphics lab motion capture database. In *http://mocap.cs.cmu.edu/*. 2013.

[5] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning (ICML)*, pages 647–655, 2014.

[6] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1110–1118, 2015.
[7] G. Evangelidis, G. Singh, and R. Horaud. Skeletal quads: Human action recognition using joint quadruples. In *International Conference on Pattern Recognition (ICPR)*, pages 4513–4518, 2014.
[8] B. Fernando, E. Gavves, J. M. Oramas, A. Ghodrati, and T. Tuytelaars. Modeling video evolution for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5378–5387, 2015.
[9] A. Gaidon, Z. Harchaoui, and C. Schmid. Temporal localization of actions with actoms. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2782–2795, 2013.
[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.
[11] A. Graves. Neural networks. In *Supervised Sequence Labelling with Recurrent Neural Networks*, pages 15–35. Springer, 2012.
[12] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649. IEEE, 2013.
[13] J. Gu, G. Wang, and T. Chen. Recurrent highway networks with language cnn for image captioning. *arXiv preprint arXiv:1612.07086*, 2016.
[14] F. Han, B. Reily, W. Hoff, and H. Zhang. space-time representation of people based on 3d skeletal data: a review. *arXiv preprint arXiv:1601.01006*, 2016.
[15] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3279–3286, 2015.
[16] J.-F. Hu, W.-S. Zheng, J. Lai, and J. Zhang. Jointly learning heterogeneous features for RGB-D activity recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5344–5352, 2015.
[17] M. E. Hussein, M. Torki, M. A. Gowayyed, and M. El-Saban. Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In *IJCAI*, volume 13, pages 2466–2472, 2013.
[18] Y. Ji, G. Ye, and H. Cheng. Interactive body part contrast mining for human interaction recognition. In *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–6. IEEE, 2014.
[19] Q. Ke, S. An, M. Bennamoun, F. Sohel, and F. Boussaid. Skeletonnet: Mining deep part features for 3d action recognition. *IEEE Signal Processing Letters*, 2017.
[20] Q. Ke, M. Bennamoun, S. An, F. Boussaid, and F. Sohel. Human interaction prediction using deep temporal features. In *European Conference on Computer Vision Workshops*, pages 403–414. Springer, 2016.
[21] Q. Ke and Y. Li. Is rotation a nuisance in shape recognition? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4146–4153, 2014.
[22] P. Koniusz, A. Cherian, and F. Porikli. Tensor representations via kernel linearization for action recognition from 3d skeletons. *arXiv preprint arXiv:1604.00239*, 2016.
[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
[24] Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.
[25] W. Li, L. Wen, M. Choo Chuah, and S. Lyu. Category-blind human action recognition: A practical recognition system. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4444–4452, 2015.
[26] J. Liu, A. Shahroudy, D. Xu, and G. Wang. Spatio-temporal LSTM with trust gates for 3D human action recognition. In *European Conference on Computer Vision (ECCV)*, pages 816–833. Springer, 2016.
[27] M. Long and J. Wang. Learning transferable features with deep adaptation networks. *CoRR, abs/1502.02791*, 1:2, 2015.
[28] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 807–814, 2010.
[29] J. C. Niebles, C.-W. Chen, and L. Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *European conference on computer vision*, pages 392–405. Springer, 2010.
[30] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio. How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026*, 2013.
[31] X. Peng and C. Schmid. Encoding feature maps of cnns for action recognition. 2015.
[32] F. Radenović, G. Tolias, and O. Chum. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. *arXiv preprint arXiv:1604.02426*, 2016.
[33] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 806–813, 2014.
[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.
[35] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4580–4584. IEEE, 2015.
[36] A. Savitzky and M. J. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.
[37] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang. NTU RGB+D: A large scale dataset for 3D human activity analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[38] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[39] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[40] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. In *ACM International Conference on Multimedia*, pages 689–692, 2015.

[41] V. Veeriah, N. Zhuang, and G.-J. Qi. Differential recurrent neural networks for action recognition. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4041–4049, 2015.

[42] R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 588–595, 2014.

[43] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1290–1297, 2012.

[44] L. Wang, Y. Qiao, and X. Tang. Latent hierarchical model of temporal structure for complex activity classification. *IEEE Transactions on Image Processing*, 23(2):810–822, 2014.

[45] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016.

[46] P. Wang, Z. Li, Y. Hou, and W. Li. Action recognition based on joint trajectory maps using convolutional neural networks. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 102–106. ACM, 2016.

[47] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Computer vision and image understanding*, 104(2):249–257, 2006.

[48] J. Weston, S. Chopra, and A. Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[49] L. Xia, C.-C. Chen, and J. Aggarwal. View invariant human action recognition using histograms of 3D joints. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 20–27, 2012.

[50] Y. Xiong, K. Zhu, D. Lin, and X. Tang. Recognize complex events from static images by fusing deep channels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1600–1609, 2015.

[51] X. Yang and Y. L. Tian. Eigenjoints-based action recognition using naive-bayes-nearest-neighbor. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 14–19, 2012.

[52] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[53] K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras. Two-person interaction detection using body-pose features and multiple instance learning. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 28–35, 2012.

[54] W. Zhu, C. Lan, J. Xing, W. Zeng, Y. Li, L. Shen, and X. Xie. Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2016.