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

Skeleton sequences provide 3D trajectories of human skeleton joints. The spatial temporal information is very important for action recognition. Considering that deep convolutional neural network (CNN) is very powerful for feature learning in images, in this paper, we propose to transform a skeleton sequence into an image-based representation for spatial temporal information learning with CNN. Specifically, for each channel of the 3D coordinates, we represent the sequence into a clip with several gray images, which represent multiple spatial structural information of the joints. Those images are fed to a deep CNN to learn high-level features. The CNN features of all the three clips at the same time-step are concatenated in a feature vector. Each feature vector represents the temporal information of the entire skeleton sequence and one particular spatial relationship of the joints. We then propose a Multi-Task Learning Network (MTLN) to jointly process the feature vectors of all time-steps in parallel for action recognition. Experimental results clearly show the effectiveness of the proposed new representation and feature learning method for 3D action recognition.

1. Introduction

Human representation based on 3D skeleton data encodes the entire human body with joints. It is robust to illumination changes and invariant to camera views [9]. With the prevalence of highly-accurate and affordable devices, action recognition based on 3D skeleton sequence has been attracting increasing attention [32, 28, 4, 24, 36, 16, 14]. In this paper, we focus on skeleton-based action recognition. Given a skeleton sequence, the temporal dynamics of multiple frames and the spatial structural information of the skeleton joints in a single frame provide important cues for action recognition [36].

Most existing works explicitly model the temporal dynamics of skeleton sequences using Hidden Markov Models (HMMs) [31], Conditional Random Fields (CRFs) [26] or Temporal Pyramids (TPs) [28]. To exploit the spatial structure, various features have been investigated, such as histogram of joint positions [32], pairwise relative position [29] and 3D rotation and translation [28]. Recently, recurrent neural networks (RNNs) with Long-Short Term Memory (LSTM) neurons [7, 8] have also been used to model the spatial structure [4, 24, 36] or both the spatial and temporal information of skeleton sequences [16]. All of the mentioned works directly operate on the native 3D coordinates of the joints to extract features and learn models. However, the coordinates of the joints are not always accurate, which generally results in poor features. In addition, it is also difficult to handle the large temporal variations and complex spatial structures using the native coordinates of the noisy skeleton joints.

Considering that CNNs are capable of learning robust features, in this paper, instead of directly operating on the native 3D coordinates of the joints to extract features, we transform each skeleton sequence to three video clips, and then utilize deep networks to learn features from the clips for action recognition.

Specifically, given a skeleton sequence, we select several joints as the reference joints, which are used to generate multiple sets of vectors by separately comparing the reference joints with the others. Three clips corresponding to the 3D coordinates of the vectors are then obtained. Each clip contains multiple frames generated from the different sets of vectors. 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 the different spatial relationships, providing important information of the spatial structure of the skeleton joints.

Unlike the original skeleton sequence, which only contains the coordinates of the discrete joints, the generated clips consist of images. The advantage of the generated clips over the original skeleton sequence is that deep CNN models pre-trained with large-scale ImageNet [22] can be leveraged to extract representations which are invariant and are insensitive to noise. CNNs are known to learn image features that are robust to noise due to the convolution and pooling operators. The learned features are generic and can be transferred to novel tasks from the original tasks [34, 17].

More specifically, each frame of the generated clips is fed to a pre-trained CNN model followed by a temporal pooling layer to extract a CNN feature. Then the three CNN features of the three clips at the same time-step (See Figure 1) are concatenated in a feature vector. Consequently, multiple feature vectors are extracted from all the time-steps. Each feature vector represents one particular spatial relationship between the joints. All the feature vectors of different time-steps represent the different spatial relationships and there exist intrinsic relationships among them. Therefore, 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 all of the feature vectors 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 final loss of the network which is then used to update 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 a skeleton sequence to a new representation, *i.e.*, three clips, to be able to learn CNN features which are more robust to joint noise and temporal variations than the features extracted from the native 3D coordinates (See Section 4.3). **(2)** We concatenate the CNN features of the three clips at the same time-step in a feature vector and propose an MTLN to jointly process the feature vectors of all time-steps for action recognition. The MTLN improves the performance by imposing weight sharing and utilizing intrinsic relationships among multiple feature vectors (See Section 4.3). **(3)** As opposed to other techniques, our method does not require any pre-processing (*e.g.*, normalization, filter smoothing, temporal sampling) of the skeleton data and it still achieves the state-of-the-art performance on three skeleton datasets, including the large scale NTU RGB+D dataset [24].

2. Related Works

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

Hand-crafted Features In [12], the covariance matrices of the trajectories of the joint positions are computed over hierarchical temporal levels to model the skeleton sequences. In [29], 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 [33], 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 [32], 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 [28], 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 [4], the skeleton joints are divided into five sets corresponding to five body parts. They are fed into five BLSTMs for feature fusion and classification. In [36], the skeleton joints are fed to a deep LSTM at each time slot to learn the inherent co-occurrence features of skeleton joints. In [24], the long-term context representations of the body parts are learned with a part-aware LSTM. In [16], 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 [24].

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 with an arbitrary number of frames is transformed into three clips corresponding to the different channels of the cylindrical coordinates, as shown in Figure 1(b). The generated clips are then fed to a pre-trained CNN model and the proposed MTLN for robust feature learning and action recognition.

3.1. Clip Generation

Given a skeleton sequence, only the trajectories of the 3D Cartesian coordinates of the skeleton joints are pro-

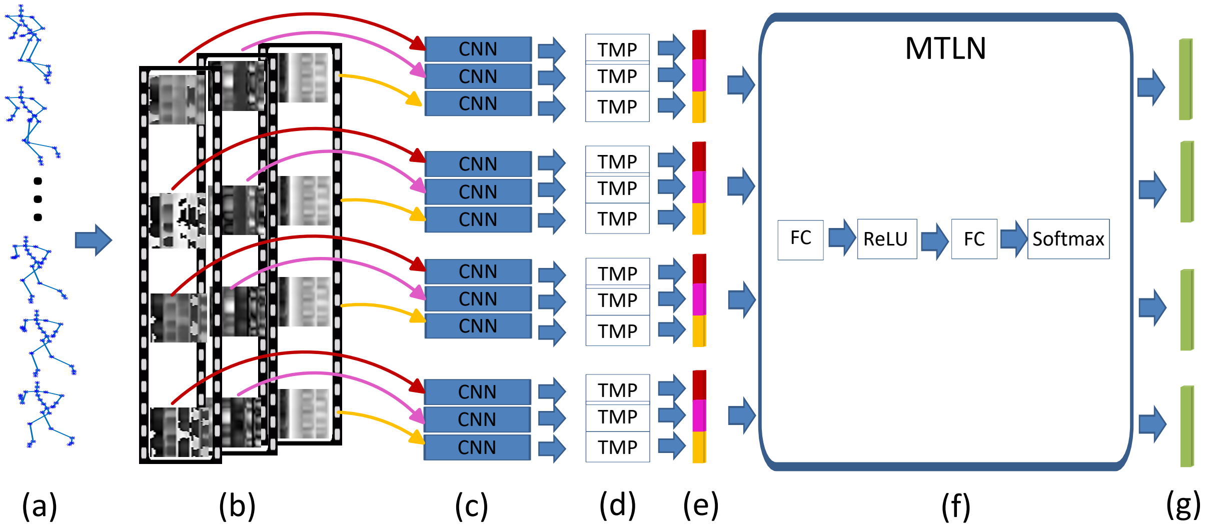

Figure 1. Architecture of the proposed method. Given a skeleton sequence (a), three clips (b) corresponding to the three channels of the cylindrical coordinates are generated. A deep pre-trained CNN model (c) and a temporal mean pooling (TMP) layer (d) are used to extract a compact representation from each frame of the clips (see Figure 2 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.

vided. As mentioned in Section 1, the features extracted from the native 3D format (*i.e.*, coordinates of joints) are sensitive to joint noise and temporal variations. This paper aims to transform the original skeleton sequence to a collection of clips consisting of images, which can be used to learn robust features using deep networks.

To transform a skeleton sequence to a video-based representation, intuitively, one could represent the content of each frame of the skeleton sequence as an image, and then combine all frames in a video. However, 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 video will also be too sparse as the number of the skeleton joints is small.

In this paper, 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 with an arbitrary number of frames, the generated clips contain the same number of frames. The robust and invariant temporal information of the original skeleton sequence could be captured with the powerful CNN representations learned from each frame image.

Specifically, the time series of each joint of a skeleton sequence can be represented as three 1D feature columns corresponding to the three channels of the 3D Cartesian coordinates (x, y, z). To transform these time series of all joints to an image-based format, a simple way is to concatenate the 1D feature columns of all joints along the row dimension in a sequential order. A 2D array could thus be generated for each channel of the 3D coordinates. The 2D arrays could further be transformed to images by scaling their values. The disadvantage of this method is that it neglects the spatial relationship between the joints, which is a critical cue for action recognition as it describes a particular posture of a human.

To tackle this issue, instead of directly using the coordinates of each joint, this paper selects several joints as reference joints. For each reference joint, a set of vectors can be derived by computing the difference of coordinates between the reference joint and the other joints. Each set of vectors reflects a particular spatial relationships between the joints. In this paper, four joints are selected as the reference joints. The four reference joints are selected from four body parts, namely, the left shoulder, the right shoulder, the left hip and the right hip. The four joints are selected 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. The four joints are respectively compared with other joints to generate four sets of vectors. The four sets of vectors combine different spatial relationships between the joints, providing useful spatial structural information of the skeleton joints.

More specifically, given a frame of a skeleton sequence, let the 3D coordinates of the skeleton joints be:

$$\Omega = \{\mathbf{q}_i \in \mathbb{R}^3 : i = 1, \cdots, m\} \tag{1}$$

where m is the number of the skeleton joints, and $\mathbf{q}_i = [x_i, y_i, z_i]$ represents the 3D coordinate of the i^{th} joint.

Let the reference joint be $\mathbf{q}_0^k = [x_0^k, y_0^k, z_0^k], k = 1, \cdots, 4$, and define:

$$\mathcal{V}_k \triangleq \left\{\mathbf{q} - \mathbf{q}_0^k : \mathbf{q} \in \Omega, k = 1, \cdots, 4\right\}. \tag{2}$$

where $\mathcal{V}_k$ is the set of vectors of the k^{th} reference joint in one frame.

The 3D Cartesian coordinates of each vector in $\mathcal{V}_k$ are further transformed to cylindrical coordinates. The cylindrical coordinates have been used to extract view-invariant motion features for action recognition [30]. Compared to the Cartesian coordinates, the cylindrical coordinates are more useful to analyse the motions as each human body utilizes pivotal joint movements to perform an action. Given a vector (x, y, z), the values are transformed to (θ, ϕ, z) where $\theta = \text{atan2}(y/x)$, $\phi = \sqrt{x^2 + y^2}$. For the k^{th} reference joint, all of the vectors in the set $\mathcal{V}_k$ are arranged in a chain. The three channels of the cylindrical coordinates of all vectors are separately concatenated into three rows, each corresponding to one channel of the cylindrical coordinates of all vectors.

Given a skeleton sequence with t frames, there are t sets of vectors for the k^{th} reference joint. The three rows of the t sets are separately concatenated along the row dimension, resulting three arrays $D_\theta^k, D_\phi^k, D_z^k$ with dimension $\mathbb{R}^{t \times m}$, m is the number of vectors in each frame of the skeleton sequence. Each array can be transformed into a 2D gray image by scaling the values of the array between 0 and 255 using linear transformation. Thus for each channel of the cylindrical coordinates, the four reference joints generate four images, which are then combined in a clip. Consequently, three clips corresponding to the three channels θ, ϕ, z are obtained.

Each frame of the generated clips describes the temporal dynamics of all frames of the skeleton sequence in one channel of the cylindrical coordinates. Specifically, the rows of the frame image correspond to the frames of the skeleton sequence, and the columns correspond to the vectors generated from the joints.

3.2. Clip Learning

The generated clips are different from the natural videos, *i.e.*, there is no temporal order of the frames. For each clip, each frame includes one particular spatial relationship between the skeleton joints in one channel of the cylindrical coordinates. Different frames describe different spatial relationships and there exists intrinsic relationships among them. Therefore, instead of computing optical flow and learning the temporal structure of each clip to provide a video-level prediction, this paper proposes to extract a compact representation from each frame using a deep CNN. The three CNN features of the three clips at the same time-step are concatenated in a feature vector, which represents the temporal information of the skeleton sequence and one particular spatial relationship between the skeleton joints in three channels of the cylindrical coordinates. Then the feature vectors of all time-steps are jointly processed in parallel using multi-task learning, thus to utilize their intrinsic relationships for 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. 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 [25] 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 [22] are very powerful and have been successfully applied in a number of cross-domain applications [3, 6, 21, 10]. 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 [22] 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 [34, 17].

The pre-trained VGG19 [25] 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,

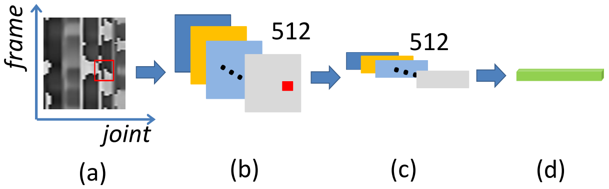

Figure 2. Temporal mean pooling of the CNN feature maps. (a) An input frame of the generated clips, for which the rows correspond to the different frames of the skeleton sequence and the columns correspond to the different vectors generated from the joints. (b) Output feature maps of the conv5_1 layer. The size is 14 × 14 × 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).

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 [34, 17]. 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 [19, 20]. 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.

As mentioned in Section 3.1, 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 [19]. The temporal information of the sequence can thus be extracted from the row features of the feature maps. More specifically, the feature maps are applied 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:

$$\begin{aligned} \mathbf{y^k} &= [y^k_1, \cdots, y^k_j, \cdots, y^k_{14}] \\ y^k_j &= \frac{1}{14}\sum_{i=1}^{14} \max(0, x^k_{i,j}) \end{aligned} \tag{3}$$

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 output 7168D features of the three clips at the same time-step are concatenated, generating four feature vectors. Each feature vector represents the temporal dynamics of the skeleton sequence and includes one particular spatial relationship between the joints in three channels of the 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) [18] 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 update 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 4.

$$\begin{aligned} \ell_k(\mathbf{z_k}, \mathbf{y}) &= \sum_{i=1}^{m} y_i \left(-log \left(\frac{\exp z_{ki}}{\sum\limits_{j=1}^{m} \exp z_{kj}} \right) \right) \\ &= \sum_{i=1}^{m} y_i \left(log \left(\sum_{j=1}^{m} \exp z_{kj} \right) - z_{ki} \right) \end{aligned} \tag{4}$$

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 5:

$$\mathcal{L}(Z, \mathbf{y}) = \sum_{k=1}^{4} \ell_k(\mathbf{z_k}, \mathbf{y}) \quad (5)$$

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 [24], SBU kinect interaction dataset [35] and CMU dataset [2].

The main ideas of the proposed method are a) representing a skeleton sequence as three clips, b) learning the temporal information of the skeleton sequence from each frame of the generated clips with CNN features, and c) utilizing the intrinsic relationships among the different features using MTLN for action recognition. To demonstrate the advantages of the proposed method, experiments were conducted using three different configurations as follows:

Coordinates + FTP In this configuration, the Fourier Temporal Pyramid (FTP) [29] is applied to the 3D coordinates of skeleton sequences to extract temporal features for action recognition.

Frames + CNN In this configuration, the three CNN features of the three clips at the same time-step are concatenated in a feature vector, and only one feature vector of a time-step is 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 4. The average accuracy of the four features is provided. Compared to Coordinates + FTP, this configuration uses CNN features of the generated frames to represent the temporal information of the skeleton sequence for action recognition.

Clips + CNN + MTLN (Proposed Method) In this configuration, the three CNN features of the three clips at the same time-step are concatenated in a feature vector, and the feature vectors of all time-steps are used for action recognition using the proposed MTLN. Compared to Frames + CNN, which can be regarded as a single-task learning method, this configuration uses multi-task learning to utilize the intrinsic relationships among different feature vectors for action recognition.

4.1. Datasets

NTU RGB+D Dataset [24] 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 [35] 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 [35].

CMU Dataset [2] 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 [36]. 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 [27]. 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 [24], 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 [16]), the performance is improved by 10.37%. When tested with the cross-view protocol, the accuracy is improved from 77.7%

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

Methods	Accuracy	
	Cross Subject	Cross View
Lie Group [28]	50.1%	52.8%
Skeletal Quads [5]	38.6%	41.4%
Dynamic Skeletons [11]	60.2%	65.2%
Hierarchical RNN [4]	59.1%	64.0%
Deep RNN [24]	59.3%	64.1%
Deep LSTM [24]	60.7%	67.3%
Part-aware LSTM [24]	62.9%	70.3%
ST-LSTM [16]	65.2%	76.1%
ST-LSTM + Trust Gate [16]	69.2%	77.7%
Coordinates + FTP	61.06%	74.64%
Frames + CNN	75.73%	79.62%
Clips + CNN + MTLN (Proposed)	**79.57%**	**84.83%**

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. The improvements are about 4% and 5% for the two testing protocols, respectively. Frames + CNN 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, which improves the performance of the single-task method for action recognition.

SBU Kinect Interaction Dataset As in [35], 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.

Table 2. Performance on the SBU kinect interaction dataset.

Methods	Accuracy
Raw Skeleton [35]	49.7%
Joint Feature [13]	86.9%
CHARM [15]	83.9%
Hierarchical RNN [4]	80.35%
Deep LSTM [36]	86.03%
Deep LSTM + Co-occurrence [36]	90.41%
ST-LSTM [16]	88.6%
ST-LSTM + Trust Gate [16]	93.3%
Coordinates + FTP	79.75%
Frames + CNN	90.88%
Clips + CNN + MTLN (Proposed)	**93.57%**

Table 3. Performance on the CMU dataset.

Methods	Accuracy	
	CMU subset	CMU
Hierarchical RNN [4]	83.13%	75.02%
Deep LSTM [36]	86.00%	79.53%
Deep LSTM + Co-occurrence [36]	88.40%	81.04%
Coordinates + FTP	83.44%	73.61%
Frames + CNN	91.53%	85.36%
Clips + CNN+ MTLN (Proposed)	**93.22%**	**88.30%**

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 [36]. The performance is improved to 93.57% when learning the entire clips with MTLN.

Since the joint positions of this dataset are not very accurate [35], existing methods including HBRNN [4] and Co-occurrence LSTM [36] remove the joint noise by smoothing the position of each joint using the Svaitzky-Golay filter [23]. In [16], 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 [36], 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 [36].

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 [36]. 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 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 [34, 17].

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 timesteps 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.

References

[1] R. Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998. 2

[2] CMU. CMU graphics lab motion capture database. In *http://mocap.cs.cmu.edu/*. 2013. 6

[3] 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. 4

[4] 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. 1, 2, 7

[5] 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. 7

[6] 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. 4

[7] A. Graves. Neural networks. In *Supervised Sequence Labelling with Recurrent Neural Networks*, pages 15–35. Springer, 2012. 1

[8] 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. 1

[9] 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. 1

[10] 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. 4

[11] 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. 7

[12] 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. 2

[13] 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. 7

[14] P. Koniusz, A. Cherian, and F. Porikli. Tensor representations via kernel linearization for action recognition from 3d skeletons. *arXiv preprint arXiv:1604.00239*, 2016. 1

[15] 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. 7

[16] 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. 1, 2, 6, 7

[17] M. Long and J. Wang. Learning transferable features with deep adaptation networks. *CoRR, abs/1502.02791*, 1:2, 2015. 2, 4, 5, 8

[18] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 807–814, 2010. 5

[19] X. Peng and C. Schmid. Encoding feature maps of cnns for action recognition. 2015. 5

[20] 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. 5

[21] 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. 4

[22] 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. 2, 4

[23] A. Savitzky and M. J. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964. 7

[24] 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. 1, 2, 6, 7

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 4

[26] C. Sminchisescu, A. Kanaujia, and D. Metaxas. Conditional models for contextual human motion recognition. *Computer Vision and Image Understanding*, 104(2):210–220, 2006. 1

[27] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. In *ACM International Conference on Multimedia*, pages 689–692, 2015. 6

[28] 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. 1, 2, 7

[29] 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. 1, 2, 6

[30] 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. 4

[31] D. Wu and L. Shao. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 724–731, 2014. 1

[32] 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. 1, 2

[33] 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. 2

[34] 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. 2, 4, 5, 8

[35] 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. 6, 7

[36] 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. 1, 2, 6, 7, 8