Computer Aided Geometric Design 85 (2021) 101965

Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

SCN: Dilated silhouette convolutional network for video action recognition

Michelle Hua [a,*], Mingqi Gao [b], Zichun Zhong [b]

[a] Cranbrook Schools, Bloomfield Hills, MI, USA
[b] Wayne State University, Detroit, MI, USA

ARTICLE INFO

Article history:
Available online 16 March 2021

Keywords:
Silhouette convolutional network (SCN)
Spatio-temporal representation
Geometric computing
Video action recognition
Deep learning
Artificial intelligence

ABSTRACT

Human action is a spatio-temporal motion sequence where strong inter-dependencies between the spatial geometry and temporal dynamics of motion exist. However, in existing literature for human action recognition from a video, there is a lack of synergy in investigating spatial geometry and temporal dynamics in a joint representation and embedding space. In this paper, we propose a dilated Silhouette Convolutional Network (SCN) for action recognition from a monocular video. We model the spatial geometric information of the moving human subject using silhouette boundary curves extracted from each frame of the motion video. The silhouette curves are stacked to form a 3D curve volume along the time axis and resampled to a 3D point cloud as a unified spatio-temporal representation of the video action. With the dilated silhouette convolution, the SCN is able to learn co-occurrence features from low-level geometric shape boundaries and their temporal dynamics jointly, and construct a unified convolutional embedding space, where the spatial and temporal properties are integrated effectively. The geometry-based SCN significantly improves the discrimination of learned features from the shape motions. Experiment results on the JHMDB, HMDB, and UCF101 datasets demonstrate the effectiveness and superiority of our proposed representation and deep learning method.

1. Introduction

Human action recognition has a wide range of applications in computer graphics, computer vision, and human-computer interaction, e.g., video games, sport analysis, public safety, virtual and augmented reality, etc. Many methods have been proposed to tackle this problem, such as Simonyan and Zisserman (2014); Ji et al. (2012); Carreira and Zisserman (2017). With the availability of low-cost depth cameras and pose estimation technologies (Shotton et al., 2011; Yub Jung et al., 2015), skeleton-based action recognition has provided a viable solution (Du et al., 2015; Song et al., 2017; Yan et al., 2018), which is more robust to variations in camera locations, human appearances, and environmental backgrounds. In addition, the computational cost of skeleton data is much lower due to its conciseness. However, the use of specialized 3D depth cameras limits its broad application and wide penetration to real-world applications for common users.

Therefore, monocular video-based action recognition continues to attract more research attention (Wang et al., 2003; Chaquet et al., 2013; Vishwakarma and Agrawal, 2013; Hassner, 2013), and recently, many deep learning-based methods have been proposed to tackle the problem (Choutas et al., 2018; Yan et al., 2019). To develop an effective deep neural

[*] Corresponding author.
E-mail addresses: mhua23@cranbrook.edu (M. Hua), gaomqgs@gmail.com (M. Gao), zichunzhong@wayne.edu (Z. Zhong).

https://doi.org/10.1016/j.cagd.2021.101965
0167-8396/© 2021 Elsevier B.V. All rights reserved.

network method for video action recognition, two aspects need to be considered: (1) the spatial and temporal representation of video actions; and (2) the corresponding deep neural network to extract distinctive spatio-temporal features for action classification. But, in the existing literature, there remains a lack of synergy in investigating the spatial geometry and temporal dynamics in a joint embedding space.

Nowadays, there are some methods starting to improve along this direction. For instance, PoTion (Choutas et al., 2018) uses human joints to capture spatial information and probability maps to temporally aggregate the pose motion representation, which is coded by the colorization scheme. The Pose-Action 3D Machine (PA3D) (Yan et al., 2019) method consists of three semantic modules, i.e., spatial pose CNN, temporal pose convolution, and action CNN. It can effectively encode multiple pose modalities within a unified 3D framework and, consequently, learn spatio-temporal pose representations for action recognition. Recently, Yan et al. (2018) used a spatio-temporal graph representation of skeleton sequences and employed a graph convolutional network (GCN) for action recognition. However, the information transmission between two correlated joints is not effective if the joints are not directly connected. To alleviate this, Shi et al. (2019) used a directed graph to represent the skeleton and learned adaptive graph structures during training, which allows actional correlated joints to be connected directly. However, these aforementioned methods rely on high-level semantic inputs (i.e., joints and skeletal poses) and the extraction of these high-level semantics itself is error-prone. That is to say, the learning procedure from the original input (pixels) to final semantic analysis (action labels) through the estimated joints/skeletons becomes quite indirect and can be affected by the pose estimation which may not be optimal. On the other side, the recent, very successful, I3D method (Carreira and Zisserman, 2017) takes pixels and optical flows as inputs, which are low-level feature inputs. The pixel-level information and their dynamics is integrated through a two-stream integration. Its action recognition performance on the benchmark datasets shows better results partly because the process of learning directly from the low-level features may be more effective. This inspires us to rethink the use of geometry-based convolution for human action recognition. It may be better to use low-level geometric representations as inputs instead of advanced semantic representations from extracted human joints and skeletons.

Hence, we propose dilated Silhouette Convolutional Network (SCN) for action recognition from a monocular video. More specifically, we model the spatial geometric information of the moving human subject using silhouette boundary curves extracted from each frame of the motion video, which are more robustly extracted than joints and skeletal poses. The silhouette curves are stacked to form a 3D curve volume along the time axis, and resampled to a 3D point cloud as a spatio-temporal representation of the video action. Action is a spatio-temporal motion sequence that involves rich geometric and dynamic properties hidden in the spatial configuration and temporal dynamics. Such a process is able to represent the dynamic and geometric properties with low-level features and integrate them effectively for a better understanding of the spatio-temporal evolutions of video action. Subsequently, a novel SCN framework is proposed to explore the co-occurrence features among silhouettes along time as well as inter-dependencies between the geometric and dynamic properties. Our main contributions can be summarized as follows:

- We present a stacked silhouette point representation to learn co-occurrence features from silhouettes and temporal dynamics jointly, and map them onto a unified convolutional space for the effective integration;
- We introduce a dilated silhouette convolutional network to construct a unified convolutional embedding space, where low-level geometric and dynamic properties can be integrated effectively to explore their inter-dependencies and enhance the discrimination of learned features;
- The explicit geometry-based SCN significantly improves the discrimination of learned features from shape motions and outperforms similar state-of-the-art approaches on the three benchmark datasets of human actions with low computational costs. Combined with I3D, it outperforms all other state-of-the-art methods because its complementarity to pixel-based methods.

2. Related work

In this section, we review the most related works regarding representation of video action and recent deep neural networks which have been used for video action recognition.

2.1. Representation of video action

In video-based action recognition, human pose is a discriminative cue for human action recognition. There exists a vast literature on action recognition from 3D skeleton data. Most of these approaches need to train a recurrent neural network (RNN) on the coordinates of human joints. However, this kind of methods requires the knowledge of the 3D coordinates of every single joint of the human in each frame. This is not generalizable to videos in the wild, which comprise of occlusions, truncations, and multiple humans in complicated scenarios. First attempts to use 2D poses were based on hand-crafted features (Jhuang et al., 2013; Wang et al., 2013; Nie et al., 2015). For instance, Jhuang et al. (2013) encoded the relative position and motion of joints with respect to the human center and scale. Wang et al. (2013) proposed to group joints on body parts (e.g. left arm) and used a bag-of-words to represent a sequence of poses. Nie et al. (2015) used a similar strategy leveraging a hierarchy of human body parts. However, these representations have several limitations: (a) they require pose tracking across the video, (b) features are hand-crafted, and (c) they are not robust to occlusion and truncation.

There also exists research work that relies on silhouettes of human subjects for action representation. Bobick and Davis (2001) proposed a technique that employs silhouettes to generate motion energy images (MEI), to detect where the movement occurs, and motion history images (MHI), to show how the object moves. Gorelick et al. (2007) accumulated silhouettes into three-dimensional representations and employed Poisson equation to extract features of human actions. More recently, Jahagirdar and Nagmode (2018) proposed the embedded histogram of oriented gradients and principal component analysis to obtain feature descriptors on silhouettes. These methods are also hand-crafted and are not generalizable to large datasets due to limited feature distinctiveness.

Recently, PoTion (Choutas et al., 2018) introduces a new representation that encodes the movement of some semantic keypoints. Specifically, they first ran a human pose estimator and extracted heatmaps for the human joints in each frame. They obtained the PoTion representation by temporally aggregating these probability maps. This is achieved by colorizing each of them, depending on the relative time of the frames in the video clip, and summing them. The network performance has been improved by adding the above-mentioned PoTion streams into a two-stream network. PA3D (Yan et al., 2019) provides a seamless workflow to encode spatio-temporal pose representations for video action recognition. Specifically, PA3D consists of three semantic modules, i.e., spatial pose CNN, temporal pose convolution, and action CNN. First, spatial pose CNN extracts different modalities of pose heatmaps (i.e., joints, part affinity fields, and convolutional features) from wild videos. Second, temporal pose convolution adaptively aggregates spatial pose heatmaps over frames, which generates a spatio-temporal pose representation for each pose modality. Finally, action CNN takes the learned pose representation as input to recognize human actions. Zhang et al. (2019) presented two end-to-end view adaptive neural networks, VA-RNN and VA-CNN, for human action recognition from skeletal data. For each network, a novel view adaptation module learns and determines the most suitable observation viewpoints and transforms the skeletons to those viewpoints for end-to-end recognition with a main classification network. These methods rely on high-level semantic inputs (i.e., joints in these cases) from other work, and the extraction of these high-level semantics is error-prone. The learning procedure from the original input (pixels) to final semantic analysis (action labels) through the estimated joints and skeletons becomes quite indirect and can be affected by the pose estimation, which is used as a learning stepping stone. Different from these methods, we propose to use temporal evolving silhouettes, which are more robustly obtained, and to define them into a spatio-temporal representation of video action, where the geometric and dynamic properties can be jointly learned through convolutions.

2.2. *Deep neural networks for action recognition*

Recently, deep learning methods have shown good performance in video action recognition. Both 2D and 3D CNN-based methods are widely used in video action recognition. 2D CNNs (Simonyan and Zisserman, 2014; Christoph and Pinz, 2016; Wang et al., 2018a) were the first to be used but had limitations due to their inability to learn spatio-temporal representations of complex actions. To address this limitation, 3D CNNs, such as model inflation (e.g., I3D) (Carreira and Zisserman, 2017), spatio-temporal relations (Wang et al., 2018b; Wang and Gupta, 2018), factorization (Xie et al., 2017; Tran et al., 2018), etc., have been applied. RNN-based methods (Du et al., 2015; Donahue et al., 2015; Du et al., 2017) focus on the characteristics of human behavior, introduce the co-occurrence of nodes in behavior into the network, and use it as a constraint of network parameter learning to optimize the performance of human action recognition. Human action is often closely related to the specific set of joints of the skeleton and the interaction of the joints in the set. However, RNN-based methods do not make full use of the optical flow and RGB information, which contain important action information. GCN-based methods (Yan et al., 2018; Shi et al., 2019) for action recognition construct a spatio-temporal graph from a sequence of 3D skeleton points. Skeleton-based data can be obtained from motion capture devices and pose estimation algorithms. Usually, the data is a sequence of frames, each with a set of joint coordinates. Given the serialized coordinates in 2D or 3D form, a spatio-temporal graph can be constructed with joints (as nodes), human body natural connection, and time domain connection (as edges). Among the methods mentioned above, I3D achieves the highest accuracy. However, I3D only works on implicit representation (i.e., pixels) of shape motion (human motion in this paper). It does not take advantage of the explicit geometric shape motion information, and requires a large set of training data. Different from these methods, the interactions of silhouette points in our method are not constrained by connectivities due to our dilated convolution scheme used in the network, whose dynamic and geometric properties can be integrated in a unified convolutional embedding space for the further exploration.

3. Dilated silhouette convolutional network

Our approach for video action recognition is based on a novel silhouette point-based representation and a correspondingly designed dilated silhouette convolutional network. The details of our approach are explained as follows.

3.1. *Stacked silhouette point representation*

To recognize human actions involved in monocular RGB videos effectively, we need to exploit both the spatial and temporal information contained in the videos. Therefore, we propose a novel stacked silhouette point representation to express the moving human subject across different frames of a given video. We first extract silhouette boundary curves of the moving human subject for each frame of a given video using a modified Mask R-CNN (He et al., 2017), which only

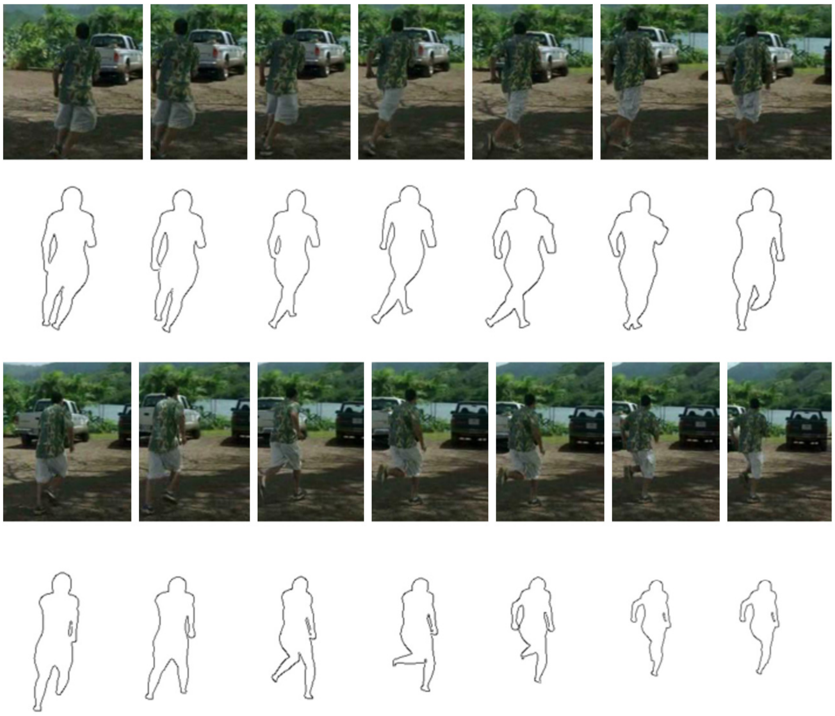

Fig. 1. Illustration of video frames and their corresponding silhouette boundary curves. The silhouette boundary curves of the moving human subject are extracted from each frame using a modified Mask R-CNN (He et al., 2017), which only detects human subjects.

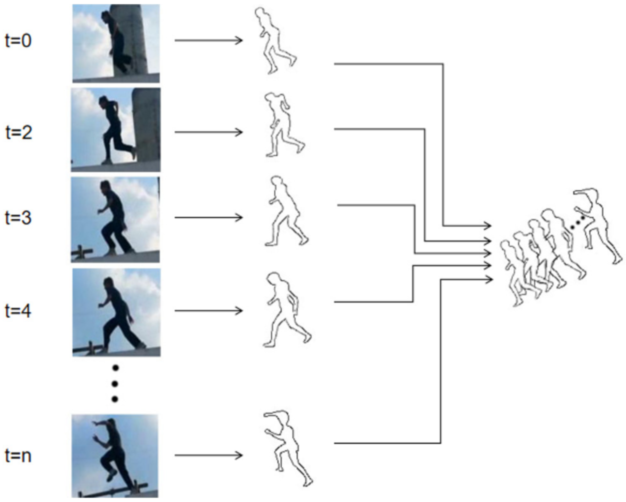

Fig. 2. Stacking silhouette boundaries along the time dimension based on their centers of gravity and resulting in a 3D curve volume. The coordinate of each point is denoted as (x, y, z), where (x, y) represents the point position in the corresponding frame, and z represents the time step of the frame.

detects human subjects. Compared to extracting joints and skeletons through pose estimation, silhouettes are more reliably extracted since they are simple separations of the foreground and background without high-level semantic information. Since Mask R-CNN is only used for human silhouette segmentation in our approach, it is very fast to process videos. In our experiments, we have tested the modified Mask R-CNN, and it takes around 25 ms $\sim$ 50 ms to process a single frame, which is fast enough to meet the needs of real application scenarios. Fig. 1 shows a sequence of extracted silhouette boundary curves of a running person.

After extracting silhouette curves of the moving human subject from each frame, we stack the obtained silhouette curves along the time dimension based on their centers of gravity, which results in a 3D curve volume along the time axis (as illustrated in Fig. 2). In order to tackle the problem of variation in video length and obtain a unified representation of different videos, the 3D curve volume is then resampled to a 3D point cloud with a fixed number of points. The coordinate

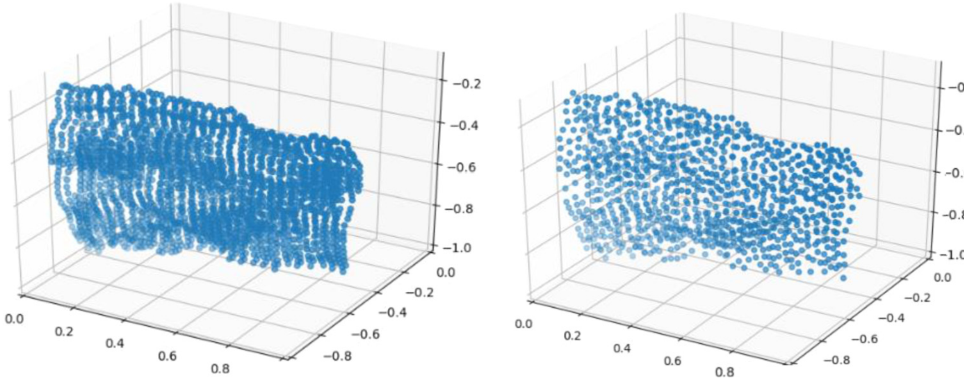

Fig. 3. Resampling a stacked silhouette point representation to a standard sampling rate using the Farthest Point Sampling algorithm (Moenning and Dodgson, 2003).

of each point is denoted as (x, y, z), where (x, y) represents the point position in the corresponding frame, and z represents the time step of the frame. In our method, the Farthest Point Sampling (FPS) algorithm (Moenning and Dodgson, 2003; Qi et al., 2017) is adopted for downsampling. Compared with random sampling, FPS can get better coverage of the entire point set. The implementation of FPS algorithm is described as follows:

Step 1: Given a point set of n points, i.e., $N = \{p_1, p_2, ..., p_n\}$, set the number of centroids to be selected as m;

Step 2: Randomly select a point p_i as the starting point, and add this point to the selected-set B, i.e., $B = \{p_i\}$;

Step 3: Calculate the distances between p_i and each of the remaining $n - 1$ points, find the point p_j with the largest distance and add p_j to set B, i.e., $B = \{p_i, p_j\}$;

Step 4: Calculate the distances of the remaining $n - 2$ points to each point in set B. For each of the remaining $n - 2$ points, select the shortest distance as its distance to the set B. Find the point p_k with the largest distance among $n - 2$ point-to-set distances and add p_k to set B, i.e., $B = \{p_i, p_j, p_k\}$;

Step 5: Repeat the step of selecting points until m points are selected.

Fig. 3 shows an example of stacked silhouette point representation before and after resampling.

3.2. Dilated silhouette convolutional network design

Compared to 2D images, the stacked silhouette point representation is a point cloud which is more flexible in terms of shape and information carried. For traditional 2D convolution on images, convolutional kernels only operate on a small local area each time. In each local area, the relative position of each pixel is always fixed, which makes the application of 2D discrete convolutions very straightforward. For convolution on irregular 3D point clouds, however, the relative position of each point is not fixed and the traditional 2D convolution cannot be applied directly. And since the points in a 3D point cloud are unordered, the convolution operation on the point cloud should be invariant to input point permutations. In this section, we describe our dilated silhouette convolutional network which are able to learn from the stacked silhouette point representations for classification.

3.2.1. Dilated silhouette convolution computation

For each local region in the input stacked silhouette point representation, i.e., point cloud set, the coordinates of all points in the local region need to be converted from global coordinates to local coordinates. To this end, we use the selected centroid point in the local region and then subtract the coordinate of the centroid from the coordinates of all the remaining points to obtain the respective local coordinates. Through the coordinate transformation operation, each local region can be regarded as a neighborhood that composed of an origin point P_0 and many other points. The point cloud convolution will perform on these neighborhoods.

The input of the entire network contains two parts. Besides the local coordinates of the points, each point also has an unshared density coefficient, which is obtained by calculating its point density in its own local region. The main reason for introducing the density coefficient is to solve the problem of uneven sampling across the constructed silhouette point cloud. The convolution operation is essentially a weighted sum operation; thus, the feature of a point is not only determined by itself, but also the neighboring points. If a point is surrounded by too many other points, there will be a relatively large weight of this point in the extracted feature map. Intuitively, denser sampled areas will have larger weights in the extracted feature map, which is not desirable. In order to tackle this problem, we first calculate the density coefficient of each point in an offline manner, which is then used to adjust the input during the training process, thereby avoiding the impact of uneven sampling. Finally, the outputs of the coordinate input module and the density input module are multiplied to extract the feature of the current point.

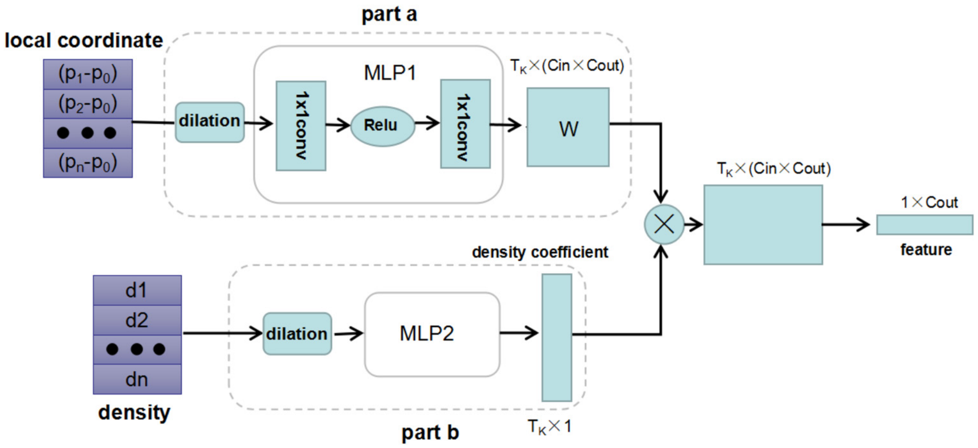

Fig. 4. Dilated silhouette convolution computed in a local neighborhood region, N.

In mathematics, convolution operation is defined as Eq. (1) for functions $f(x)$ and $g(x)$:

$$(f * g)(x) = \iint\limits_{\gamma \in \mathbb{R}^3} f(\gamma) g(x+\gamma) d\gamma. \tag{1}$$

For dilated silhouette convolution in this paper, the continuous convolution operation in a local neighborhood, N, is expressed as follows:

$$Dconv(W, F)_{xyz} = \iiint\limits_{(\delta_x,\delta_y,\delta_z)\in N} W(\delta_x, \delta_y, \delta_z) F(x+\delta_x, y+\delta_y, T(z+\delta_z)) d\delta_x d\delta_y d\delta_z, \tag{2}$$

where $W(x, y, z)$ denotes a trainable weight of point (x, y, z), F is the feature of the point, and $T(\cdot)$ denotes dilated convolution on the z axis in order to obtain a larger temporal receptive field, i.e., we adopt the dilated convolution module on the temporal dimension, the z axis.

Inspired by Wu et al. (2019), we use a density function, $S(\cdot)$, to combat the point sampling density variance. The density function measures the reciprocal of the point density at the local region. The final discrete dilated convolution operation in our network is expressed as follows:

$$Dconv(S, W, F)_{xyz} = \sum_{(\delta_x,\delta_y,\delta_z)\in N} S(\delta_x, \delta_y, \delta_z) W(\delta_x, \delta_y, \delta_z) F(x+\delta_x, y+\delta_y, T(z+\delta_z)). \tag{3}$$

In order to obtain optimal convolution kernel function, W, and density kernel function, S, to compute the dilated point convolutions, we implement it with multilayer perceptron (MLP), similar to Wu et al. (2019). The weights of all the MLP modules are shared in order to be invariant to point permutations. The final network architecture is composed of two branches, one branch takes the local coordinates as input while the other branch takes the density values as input. Each MLP module consists of dilations and two 1×1 convolutional layers. Fig. 4 shows the overall network architecture.

As shown in Fig. 4, the upper branch (part a) takes the local coordinates of points as input and learns a weight W for the current point through a MLP module. The bottom branch (part b) takes the density values as input, which are calculated by kernel density estimation (KDE) beforehand. Since the larger the KDE value, the heavier the weight in the resulted feature map, so we use the reciprocal of KDE value as input in order to decrease the weights of dense regions in the feature map. The output of this branch is a vector of density coefficients S. By multiplying weight coefficients W and density coefficients S, the feature map is adjusted to reduce the influence of uneven sampling. $T(K)$ denotes the dilation of the point set K. C_{in} and C_{out} are the numbers of channels for the input and output features, respectively, while c_{in} and c_{out} are the indices for the c_{in}-th channel for input feature and the c_{out}-th channel for output feature. The final output can be expressed as follows:

$$F_{out} = \sum_{k=1}^{T(K)} \sum_{c_{in}=1}^{C_{in}} S(k) W(k, c_{in}) F(k, c_{in}). \tag{4}$$

3.2.2. Dilated silhouette convolutional network architecture

Our proposed SCN architecture follows a hierarchical structure design consisting of a set of convolution layers. Each convolution layer consists of several operations performed sequentially and produces a subset of input points with newly

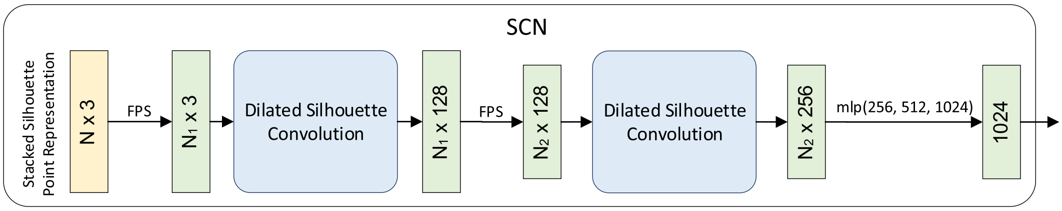
SCN
Stacked Silhouette Point Representation
N x 3
FPS
N₁ x 3
Dilated Silhouette Convolution
N₁ x 128
FPS
N₂ x 128
Dilated Silhouette Convolution
N₂ x 256
mlp(256, 512, 1024)
1024

Fig. 5. SCN feature encoder with the stacked silhouette point representation as input. The downsampling is conducted using Farthest Point Sampling. The output is a feature vector.

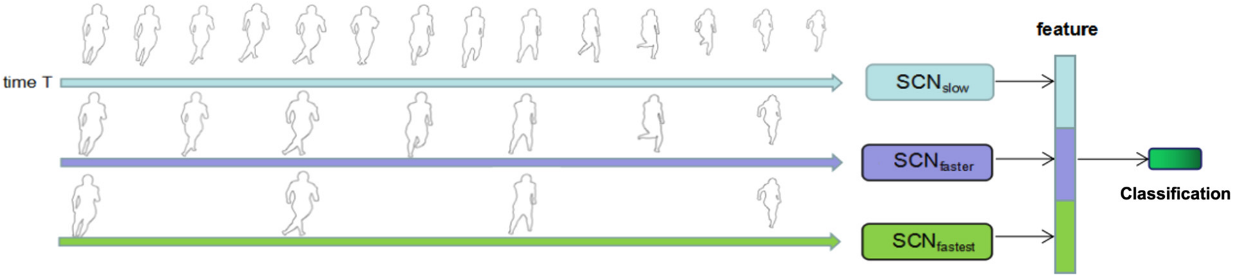

Fig. 6. Slow-to-Fast SCN architecture.

learned features. Firstly, we downsample the silhouette point representation using Farthest Point Sampling algorithm (Moenning and Dodgson, 2003) to extract centroids randomly distributed on the silhouette point cloud of each action. Secondly, K-NN extracts a local neighborhood region for each centroid. Finally, we apply a set of dilated silhouette point convolutions in the local neighborhood regions to produce new feature vectors, which uniquely describe each local region.

Given the 3D stacked silhouette point representation of a video action, our proposed end-to-end SCN network, as shown in Fig. 5, is able to encode a feature for classification of actions. The encoder, i.e., SCN, extracts features from each local neighborhood region independently inside every convolution layer and concatenates them at the end to further feedforward to extract high-level features. In this work, the SCN architecture contains two dilated point convolution layers and each includes both temporal dilations of 1 and 2. We use two dilations per layer because it gives us excellent experimental results for our application. It can be easily extended to more than two dilations per layer, if necessary. Each dilated convolution layer is followed by a batch normalization (BN) and a rectified linear unit (ReLU). Then, the aggregated features are propagated to the next layer. After the two dilated convolution layers, the last layer in the encoder performs convolutions with kernel sizes 1 × 1 followed by BN and ReLU layers. Finally, the aggregated high-level features from the encoder are fed out for classification.

3.2.3. Slow-to-Fast SCN

Note that, the dilated silhouette point convolution only performs dilation operations based on randomly selected centroid points in the original sampling of the stacked silhouette point representation. To further expand the temporal receptive field globally, we resample the stacked silhouette point representation to three temporal scales, $S_{slow} = \{s_0, s_1, s_2, s_3, \cdots, s_n\}$, $S_{faster} = \{s_0, s_2, s_4, \cdots\}$, and $S_{fastest} = \{s_0, s_3, s_6, \cdots\}$, where s_i denotes the silhouette points at Frame t_i. The three sets of silhouette point representations are input to SCNs, respectively, as shown in Fig. 6. The output features from each SCN are concatenated to a single feature vector and fed to the set of fully-connected layers with integrated dropout and ReLU layers to calculate probability of each class. The output size of the classification network is equal to the number of action classes in the dataset.

4. Experiments

We conduct extensive experiments including training and testing on the HMDB, JHMDB, and UCF101 datasets using their standard protocols. It is noted that for comparison experiments, the best results in the tables are shown in bold font. All models in this paper are trained on a single NVIDIA Titan Xp GPU with 12 GB GDDR5X. The source code will be publicly available soon.

4.1. Datasets and settings

The HMDB dataset contains 6766 video clips from 51 classes, such as brushing hair, sitting, drinking, etc. The JHMDB dataset is a subset of HMDB with 928 short videos from 21 classes. A number of frames in the JHMDB dataset are annotated with a puppet model that is fitted to the actor, i.e., this results in an approximative 2D ground-truth pose, and those frames have handcrafted masks of the silhouettes of actions. The UCF101 dataset is much larger than HMDB and JHMDB. It consists of around 13K videos from 101 action classes, including playing a variety of sports and instruments. HMDB and UCF101 have no handcrafted masks.

JHMDB, HMDB, and UCF101 have provided three training/testing splits. Following the standard protocols, we train and test using the provided splits. In a dataset, each video has only a single label, and we test mean classification accuracy (in percentage) of the testing sets, i.e., the ratio of videos in a given class that is correctly classified averaged over all classes in the dataset. For each experimentation on one dataset, we run the experiment three times using the three training/testing splits of the dataset, respectively, and report the average accuracy of the three experiments as the final result of our SCN on the dataset.

4.2. Implementation details and time performance

Recently, Carreira and Zisserman (2017) have highlighted the importance of pre-training for action recognition with the Kinetics dataset. In contrast, our network, which takes the action point cloud as input, can be simply trained from scratch without pre-train. In our experiments, we set the number of sampling points in the stacked silhouette point representation to 4096. Thus, each video action is represented as a point cloud of 4096 points. We have tested different numbers of sampling points used in this representation. The total number of points, 4096, is a good sampling rate considering both accuracy and computational cost. Significantly lower than 4096 points will cause information loss, hence, a lower accuracy. Greater than that does not increase the accuracy. Certainly, this also depends on the action datasets. If SCN is used for fine-grained action recognitions, the sampling rate for the stacked silhouette point representation needs to be higher. The initial learning rate is set to 0.001, and reduced to 0.00075 in the middle and 0.0005 towards the end. The weight decay that we use is 0.0001, the momentum is 0.8, the batch size is 64, and the gamma is 0.1. During training, we drop activations with a probability of 0.3 after each convolutional layer. We optimize the network using the Adam optimization method. The combination of I3D and our SCN is through equal weighted summation of their respective prediction outputs. The simple combination strategy is to prevent interference with each model's capability and their intrinsic complementarity. It can demonstrate how much different models are truly complementary to each other when not learned from each other during training.

Using HMDB training as an example, the typical length of video clips of human actions in the datasets is about 30 seconds. For each video clip of a human action, we evenly extract around 250 frames. We use a modified Mask R-CNN (which detects humans only) to process each frame to obtain a human silhouette with the time cost of 25 ms $\sim$ 50 ms depending on the size of the human in the frame. For training, this process is a preprocessing step and only conducted once for each dataset. Once we have pre-computed our stacked silhouette point representation for every video clip of the dataset, it takes approximatively 3 hours to train our network on the HMDB dataset. As for testing, each stacked silhouette point representation can be classified within 1.5 seconds. With the help of the sparse representation of the point cloud on the shape silhouette (i.e., a 3D curve volume) resulting in a lightweight computation, our video classification training can be done in a few hours on a single GPU (e.g., a single Nvidia Titan Xp in our case) without any pre-training. However, all other state-of-the-art approaches (e.g., using a 3D image volume representation) often require multiple days of training on several GPU cards after a pre-training stage.

4.3. Comparison with state-of-the-art

We compare the performance of our method with a number of state-of-the-art methods on video action recognition using JHMDB, HMDB, and UCF101 datasets. The numerical evaluations are shown in Table 1. It is clearly seen that a combination of our SCN with I3D significantly outperforms all other recent approaches. Overall, we outperform all existing approaches on all datasets, including methods which leverage pose or capture long-term motion. On the JHMDB dataset, we significantly outperform P-CNN (Chéron et al., 2015), which leverages pose estimation to pool CNN features. We obtain a significantly higher accuracy (i.e., 86.3%) than the classification performance reported by action localization approaches. On the HMDB dataset, we obtain 85.1% mean classification accuracy, performing better than I3D by 5.4% and better than other methods by more than 15%. On the UCF101 dataset, we also reach a state-of-the-art accuracy with 98.3%, 0.3% above I3D alone.

Note that, each method uses different modalities and pre-training strategies. Here, we would like to emphasize that our work is within the same category of PoTion (Choutas et al., 2018), PA3D (Yan et al., 2019), and DD-Net (Zhang et al., 2019) (i.e., using explicit 2D frame geometry-based representations). However, our method based on silhouettes outperforms their method based on joints or skeletons. In summary, on smaller datasets (e.g., HMDB) geometry-based (model-driven) SCN performs better than pixel-based (data-driven) I3D, while I3D performs better than SCN on large datasets (e.g., UCF101). When combining implicit pixel-based I3D and explicit geometry-based SCN, the performance becomes preeminent, which indicates that our geometry-based representation and convolution best complements pixel-based representation and convolution.

In Table 1, we also evaluate the performance of different methods combined with I3D. Our method (I3D'+SCN) still outperforms I3D+PoTion (Choutas et al., 2018) and I3D+PA3D (Yan et al., 2019). I3D' denotes our own implementation of I3D. Note that geometry-based representation and convolution complement pixel-based representation and convolution as indicated by PoTion, PA3D, and our method. But when our geometry-based method, SCN, is combined with I3D, it outperforms all the others.

Table 1
Comparison with state-of-the-art methods with mean per-class accuracy on the JHMDB, HMDB, and UCF101 datasets averaged over the three splits. I3D' denotes results which we have reproduced. Our work is within the same category of PoTion, PA3D, DD-Net, etc. (i.e., using explicit 2D frame geometry based representations). However, our method based on silhouettes outperforms their methods based on joints or skeletons. On smaller datasets (e.g., HMDB) geometry-based SCN performs better than pixel-based I3D while I3D performs better than SCN on large datasets (e.g., UCF101) (but I3D requires a pre-train Deng et al. (2009)). Geometry-based representation and convolution complement pixel-based representation and convolution as indicated by PoTion, PA3D, and our method, SCN. When combining implicit pixel-based I3D and our explicit geometry-based SCN, the performance becomes the best, which indicates that our geometry-based representation and convolution best complements pixel-based representation and convolution.

Methods	JHMDB	HMDB	UCF101
2D geometry-based methods			
P-CNN (Chéron et al., 2015)	61.1	–	–
Action Tubes (Gkioxari and Malik, 2015)	62.5	–	–
PoTion (Choutas et al., 2018)	57.0	43.7	65.2
PA3D (Yan et al., 2019)	69.5	55.3	–
DD-Net (Zhang et al., 2019)	77.2	–	–
SCN (Ours)	**77.8**	**82.1**	69.6
Pixel-based methods			
MR Two-Sream R-CNN (Peng and Schmid, 2016)	71.1	–	–
Attention Pooling (Girdhar and Ramanan, 2017)	–	52.2	–
Res3D (Tran et al., 2017)	–	54.9	85.8
Two-Stream (Simonyan and Zisserman, 2014)	–	59.4	88.0
IDT (Wang and Schmid, 2013)	–	61.7	86.4
Dynamic Image Networks (Bilen et al., 2016)	–	65.2	89.1
C3D (3 nets) (Tran et al., 2015)+IDT	–	–	90.4
LatticeLSTM (Sun et al., 2017)	–	66.2	93.5
Two-Stream Fusion (Feichtenhofer et al., 2016)+IDT	–	69.2	93.5
TSN (Wang et al., 2016)	–	69.4	94.2
Spatio-Temporal ResNet (Christoph and Pinz, 2016)+IDT	–	70.3	94.6
I3D (Carreira and Zisserman, 2017)	–	80.7	**98.0**
Spatiotemporal Fusion (Zhou et al., 2020)	–	–	96.5
TEA (Li et al., 2020)	–	73.3	96.9
2D geometry and pixel combined			
Chained (Pose+RGB+Flow) (Zolfaghari et al., 2017)	76.1	69.7	91.1
I3D+PoTion (Choutas et al., 2018)	85.5	80.9	98.2
I3D+PA3D (Yan et al., 2019)	–	82.1	–
I3D'+SCN (Ours)	**86.3**	**85.1**	**98.3**

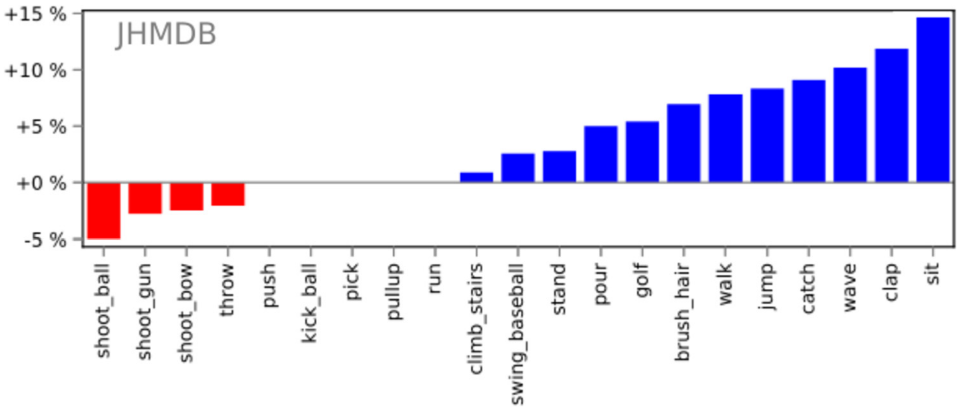

Fig. 7. Visualization of action recognition accuracy difference on each class by I3D' and I3D'+SCN methods on the JHMDB dataset.

In order to further analyze the gains obtained by our SCN network, we provide the detailed statistics of the difference in action recognition accuracy on each class in the results obtained by I3D' and I3D'+SCN on the JHMDB dataset. The specific differences are visualized in Fig. 7. It is worth noting that in JHMDB dataset, since some action classes are quite different and distinguishable from other classes, such as running, kicking a ball, pushing, etc., these classes are perfectly classified, which is impossible to further improve upon using our I3D'+SCN. The categories that have been significantly improved upon are mainly in the classes that have a great correlation between posture and movement patterns, such as sitting, clapping,

Table 2
Model analysis on w/o and w/ Slow-to-Fast architecture.

Datasets	w/o Slow-to-Fast	w/ Slow-to-Fast
JHMDB	76.9	**77.8**
HMDB	81.4	**82.1**
UCF101	68.1	**69.6**

Table 3
Model analysis on w/o and w/ dilation.

Datasets	w/o dilation	w/ dilation
JHMDB	76.4	**77.8**
HMDB	81.0	**82.1**
UCF101	67.5	**69.6**

Table 4
Model analysis on w/o and w/ using manual mask on JHMDB dataset.

Datasets	w/ manual mask	w/o manual mask
JHMDB	77.4	**77.8**

waving, catching, jumping, etc. However, the action classes SCN preformed poorly on are usually very similar in posture and movement, such as throwing, shooting a ball, and shooting a gun. For this type of movements, the appearance of movements from images / videos is more important than the postures (silhouettes); so, our SCN does not perform very well on them.

4.4. Model analysis and ablation study

Based on the characteristics of the action silhouette point cloud and video action recognition, we propose a Slow-to-Fast structure to expand the receptive field of the convolution kernel in order to obtain better action recognition results. The experiments are carried out by w/o and w/ a Slow-to-Fast structure, respectively. In Table 2, it shows that when we use the Slow-to-Fast architecture in our network, the accuracy values of three datasets improve, and the improvement on UCF101 is much higher than those on JHMDB and HMDB. This is due to the characteristics of the datasets. The average video length of the UCF101 dataset is about 60 seconds, while the average video length of the HMDB dataset is only about 30 seconds, meaning that the number of silhouette curve sequences extracted from the UCF101 dataset is doubled those extracted from the HMDB dataset. Due to the longer silhouette curve sequence, the performance of UCF101 dataset is better with the help of the proposed Slow-to-Fast architecture since the receptive field in the neural network is expanded. According to improvements on the three datasets, the slow-to-fast structure which we proposed is effective, and the action recognition performance can be greatly improved by expanding the receptive field. The proposed dilation is used to further expand the receptive field of the convolution without omitting frames from the stacked silhouette point representation inputs. The experiments are carried out by w/o and w/ dilations, respectively. In Table 3, it shows that when we use the dilations in our network, the low-level geometric and dynamic properties can be integrated effectively to explore their inter-dependencies, hence, enhancing the discrimination of learned features to improve the recognition accuracy values in the three datasets.

In the three experiment datasets, UCF101 and HMDB do not have manual mask data, but JHMDB has provided manual mask data. In this case, we conduct a comparison experiment on the JHMDB dataset to analyze the influence of the manually annotated mask data on action recognition performance of our proposed network. In Table 4, it shows that our network performs robustly on with and without manual masks in the JHMDB dataset. The masks are obtained by the Mask R-CNN method (He et al., 2017) when there is no manual mask on JHMDB dataset. It is interesting that our network performance drops slightly by using the manual masks. One possible reason is the action silhouette curves extracted by the deep neural network pays more attention to the low-level geometric features of actions, which is more suitable as the input of our neural network than the manual annotation.

5. Conclusion

In this work, we have presented a dilated Silhouette Convolutional Network (SCN) for human action recognition from a monocular video. It can effectively capture the spatio-temporal evolutions of human silhouette shapes. The spatial geometry and temporal dynamics of the moving human subject are embedded in a 3D silhouette point cloud as a joint spatial-temporal representation of video action. A novel silhouette convolutional network is presented to explore the co-occurrence features among silhouettes along the time dimension as well as inter-dependencies between the geometric and dynamic properties. SCN significantly improves the discrimination of learned features and outperforms similar state-of-the-art approaches on the three benchmark datasets.

The SCN has certain limitations too. Currently, the silhouettes are extracted as boundaries that separate the background and foreground. For certain actions, the extracted silhouettes may not contain enough discriminative information due to self-occlusion. For example, if the action of waving hands is inside the body contour, the current silhouettes extracted by Mask R-CNN cannot capture the action information; therefore, it results in a misclassification. This can be improved by extracting human movement silhouettes including internal parts as well. The potential improvement of SCN in occluded or self-occluded situations will be investigated in our future work.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Bilen, H., Fernando, B., Gavves, E., Vedaldi, A., Gould, S., 2016. Dynamic image networks for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3034–3042.

Bobick, A., Davis, J., 2001. The recognition of human movement using temporal templates. IEEE Trans. Pattern Anal. Mach. Intell. 23, 257–267.

Carreira, J., Zisserman, A., 2017. Quo vadis, action recognition? A new model and the kinetics dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6299–6308.

Chaquet, J.M., Carmona, E.J., Fernández-Caballero, A., 2013. A survey of video datasets for human action and activity recognition. Comput. Vis. Image Underst. 117, 633–659.

Chéron, G., Laptev, I., Schmid, C., 2015. P-CNN: pose-based CNN features for action recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3218–3226.

Choutas, V., Weinzaepfel, P., Revaud, J., Schmid, C., 2018. PoTion: pose motion representation for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7024–7033.

Christoph, R., Pinz, F.A., 2016. Spatiotemporal residual networks for video action recognition. In: Advances in Neural Information Processing Systems, pp. 3468–3476.

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. ImageNet: a large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T., 2015. Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2625–2634.

Du, W., Wang, Y., Qiao, Y., 2017. RPAN: an end-to-end recurrent pose-attention network for action recognition in videos. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3725–3734.

Du, Y., Wang, W., Wang, L., 2015. Hierarchical recurrent neural network for skeleton based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1110–1118.

Feichtenhofer, C., Pinz, A., Zisserman, A., 2016. Convolutional two-stream network fusion for video action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1933–1941.

Girdhar, R., Ramanan, D., 2017. Attentional pooling for action recognition. In: Advances in Neural Information Processing Systems, pp. 34–45.

Gkioxari, G., Malik, J., 2015. Finding action tubes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 759–768.

Gorelick, L., Blank, M., Shechtman, E., Irani, M., Basri, R., 2007. Actions as space-time shapes. IEEE Trans. Pattern Anal. Mach. Intell. 29, 2247–2253.

Hassner, T., 2013. A critical review of action recognition benchmarks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 245–250.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969.

Jahagirdar, A., Nagmode, M., 2018. Silhouette-based human action recognition by embedding HOG and PCA features. In: Bhalla, S., Bhateja, V., Chandavale, A., Hiwale, A., Satapathy, S. (Eds.), Intelligent Computing and Information and Communication. Springer, pp. 363–371.

Jhuang, H., Gall, J., Zuffi, S., Schmid, C., Black, M.J., 2013. Towards understanding action recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3192–3199.

Ji, S., Xu, W., Yang, M., Yu, K., 2012. 3D convolutional neural networks for human action recognition. IEEE Trans. Pattern Anal. Mach. Intell. 35, 221–231.

Li, Y., Ji, B., Shi, X., Zhang, J., Kang, B., Wang, L., 2020. TEA: temporal excitation and aggregation for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 906–915.

Moenning, C., Dodgson, N.A., 2003. Fast marching farthest point sampling. Technical report. University of Cambridge, Computer Laboratory.

Nie, B.X., Xiong, C., Zhu, S.C., 2015. Joint action recognition and pose estimation from video. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1293–1301.

Peng, X., Schmid, C., 2016. Multi-region two-stream R-CNN for action detection. In: Proceedings of the European Conference on Computer Vision. Springer, pp. 744–759.

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017. PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems, pp. 5099–5108.

Shi, L., Zhang, Y., Cheng, J., Lu, H., 2019. Skeleton-based action recognition with directed graph neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7912–7921.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A., 2011. Real-time human pose recognition in parts from single depth images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1297–1304.

Simonyan, K., Zisserman, A., 2014. Two-stream convolutional networks for action recognition in videos. In: Advances in Neural Information Processing Systems, pp. 568–576.

Song, S., Lan, C., Xing, J., Zeng, W., Liu, J., 2017. An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4263–4270.

Sun, L., Jia, K., Chen, K., Yeung, D.Y., Shi, B.E., Savarese, S., 2017. Lattice long short-term memory for human action recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2147–2156.

Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M., 2015. Learning spatiotemporal features with 3D convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4489–4497.

Tran, D., Ray, J., Shou, Z., Chang, S.F., Paluri, M., 2017. ConvNet architecture search for spatiotemporal feature learning. arXiv preprint. arXiv:1708.05038.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M., 2018. A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6450–6459.

Vishwakarma, S., Agrawal, A., 2013. A survey on activity recognition and behavior understanding in video surveillance. Vis. Comput. 29, 983–1009.

Wang, C., Wang, Y., Yuille, A.L., 2013. An approach to pose-based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 915–922.

Wang, H., Schmid, C., 2013. Action recognition with improved trajectories. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3551–3558.

Wang, J., Cherian, A., Porikli, F., Gould, S., 2018a. Video representation learning using discriminative pooling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1149–1158.

Wang, L., Hu, W., Tan, T., 2003. Recent developments in human motion analysis. Pattern Recognit. 36, 585–601.

Wang, L., Li, W., Li, W., Van Gool, L., 2018b. Appearance-and-relation networks for video classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1430–1439.

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L., 2016. Temporal segment networks: towards good practices for deep action recognition. In: Proceedings of the European Conference on Computer Vision. Springer, pp. 20–36.

Wang, X., Gupta, A., 2018. Videos as space-time region graphs. In: Proceedings of the European Conference on Computer Vision, pp. 399–417.

Wu, W., Qi, Z., Fuxin, L., 2019. PointConv: deep convolutional networks on 3D point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9621–9630.

Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K., 2017. Rethinking spatiotemporal feature learning for video understanding. arXiv preprint. arXiv:1712.04851.

Yan, A., Wang, Y., Li, Z., Qiao, Y., 2019. PA3D: pose-action 3D machine for video recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7922–7931.

Yan, S., Xiong, Y., Lin, D., 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 7444–7452.

Yub Jung, H., Lee, S., Seok Heo, Y., Dong Yun, I., 2015. Random tree walk toward instantaneous 3D human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2467–2474.

Zhang, P., Lan, C., Xing, J., Zeng, W., Xue, J., Zheng, N., 2019. View adaptive neural networks for high performance skeleton-based human action recognition. IEEE Trans. Pattern Anal. Mach. Intell. 41, 1963–1978.

Zhou, Y., Sun, X., Luo, C., Zha, Z.J., Zeng, W., 2020. Spatiotemporal fusion in 3D CNNs: a probabilistic view. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9826–9835.

Zolfaghari, M., Oliveira, G.L., Sedaghat, N., Brox, T., 2017. Chained multi-stream networks exploiting pose, motion, and appearance for action classification and detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2904–2913.