# Part-based Graph Convolutional Network for Action Recognition

Kalpit Thakkar
kalpit.thakkar@research.iiit.ac.in

P J Narayanan
pjn@iiit.ac.in

Center for Visual Information
Technology (CVIT), Kohli Center for
Intelligent Systems (KCIS),
IIIT Hyderabad, India

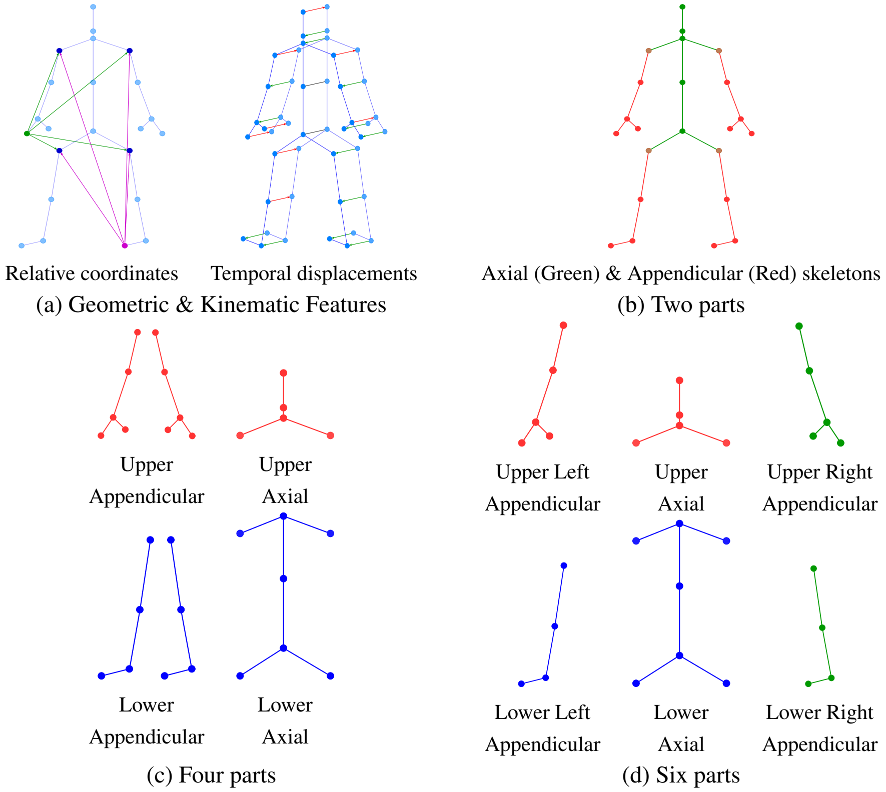
**Abstract**

Human actions comprise of joint motion of articulated body parts or "gestures". Human skeleton is intuitively represented as a sparse graph with joints as nodes and natural connections between them as edges. Graph convolutional networks have been used to recognize actions from skeletal videos. We introduce a part-based graph convolutional network (PB-GCN) for this task, inspired by Deformable Part-based Models (DPMs). We divide the skeleton graph into four subgraphs with joints shared across them and learn a recognition model using a part-based graph convolutional network. We show that such a model improves performance of recognition, compared to a model using entire skeleton graph. Instead of using 3D joint coordinates as node features, we show that using relative coordinates and temporal displacements boosts performance. Our model achieves state-of-the-art performance on two challenging benchmark datasets NTURGB+D and HDM05, for skeletal action recognition.

## 1 Introduction

Recognizing human actions in videos is necessary for understanding them. Video modalities such as RGB, depth and skeleton provide different types of information for understanding human actions. The S-video (or Skeletal modality) provides 3D joint locations, which is a relatively high level information compared to RGB or depth. With the release of several multi-modal datasets [1, 3, 24], action recognition from S-video has gained significant traction recently [12, 18, 19, 26, 36].

Graph convolutions [4, 14, 21] have been used to learn high level features from arbitrary graph structure. State-of-the-art action recognition from S-videos [16, 33] use graph convolutions, wherein the whole skeleton is treated as a single graph. It is, however, natural to think of human skeleton as a combination of multiple body parts. A body-part based representation can learn the importance of each part and their relations across space and time. We present a model using part-based graph convolutional network for recognizing actions from S-videos, using a novel part-based graph convolution scheme. The model attains better performance for recognition than a model entire skeleton as a single graph. Current models for skeletal action recognition [16, 33] use 3D coordinates as features at each vertex. Geometric features such as relative joint coordinates and motion features such as temporal displacements can be more informative for action recognition. Optical flow helps in action recognition from RGB videos [30] and Manhattan line map helps in generating 3D layout from single image



(a) Geometric & Kinematic Features

(b) Two parts

(c) Four parts

(d) Six parts

Figure 1: **(a)** Geometric and Kinematic features, **(b)** Appendicular and axial body parts: two parts, **(c)** Dividing the appendicular and axial skeletons into upper and lower parts: four parts, **(d)** Dividing appendicular upper and lower skeletons into left and right: six parts.

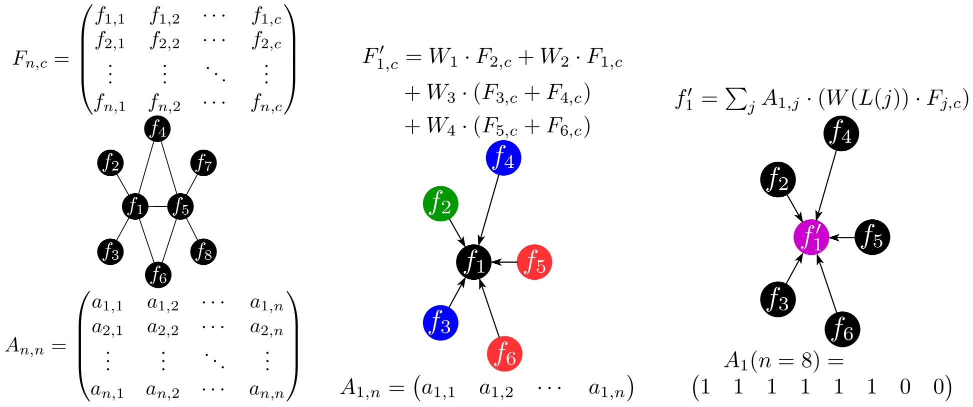
[37]. Geometric feature [36] and kinematic features [34] have been used for skeletal action recognition before. Inspired by these observations, we use a geometric feature that encodes relative joint coordinates and motion feature that encodes temporal displacements at each vertex in our part-based graph convolution model to significant impact.

The major contributions of this paper are: (i) Formulation of a general part-based graph convolutional network (PB-GCN) which can be learned for any graph with well-known properties and its application to recognize actions from S-videos, (ii) Use of geometric and motion features in place of 3D joint locations at each vertex to boost recognition performance, and (iii) Exceeding the state-of-the-art on challenging benchmark datasets NTURGB+D and HDM05. The overview of our representation and signals is shown in Figure 1.

# 2 Related Work

## 2.1 Non graph-based methods

Skeletal action recognition has been approached using techniques such as handcrafted feature encodings, complex LSTM networks, image encodings with pretrained CNNs and non-euclidean methods based on manifolds. Non-deep learning methods worked well initially and proved usefulness of several extracted information from S-videos such as joint angles [22], distances [32] and kinematic features [34]. These methods learn from hand designed


$F_{n,c} = \begin{pmatrix} f_{1,1} & f_{1,2} & \cdots & f_{1,c} \\ f_{2,1} & f_{2,2} & \cdots & f_{2,c} \\ \vdots & \vdots & \ddots & \vdots \\ f_{n,1} & f_{n,2} & \cdots & f_{n,c} \end{pmatrix}$
$f_1$ $f_2$ $f_3$ $f_4$ $f_5$ $f_6$ $f_7$ $f_8$
$A_{n,n} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}$
$F'_{1,c} = W_1 \cdot F_{2,c} + W_2 \cdot F_{1,c} + W_3 \cdot (F_{3,c} + F_{4,c}) + W_4 \cdot (F_{5,c} + F_{6,c})$
$f_4$ $f_2$ $f_1$ $f_5$ $f_3$ $f_6$
$A_{1,n} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \end{pmatrix}$
$f'_1 = \sum_j A_{1,j} \cdot (W(L(j)) \cdot F_{j,c})$
$f_4$ $f_2$ $f'_1$ $f_5$ $f_3$ $f_6$
$A_1(n=8) = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}$

(a) Graph feature and adjacency matrices

(b) Convolution for receptive field of a chosen root vertex $f_1$

(c) Final convolution equation

Figure 2: Equation-based formulation and illustration of a graph convolution

features using shallow models which do not model spatio-temporal properties of actions very well and constrain learning capacity.

On the other hand, LSTM-based methods were used because S-videos can be thought of as time sequences of features. Spatio-temporal LSTMs [18, 19], attention-based LSTM [26] and simple LSTM networks with part-based skeleton representation [7, 27] have been used. These methods either use complex LSTM models which have to be trained very carefully or use part-based representation with a simple LSTM model. We propose a part-based graph convolutional network that has good learning capacity and uses a part-based representation, inheriting the good qualities of both types of aforementioned approaches. Image encodings of skeletons were proposed to facilitate usage of Imagenet pretrained CNNs to extract spatio-temporal features. Ke *et al.* [12] generate images using relative coordinates while Du *et al.* [6] and Li *et al.* [15] proposed a body part-based image encoding. Due to inherent differences in information in such image encodings and RGB images, it is almost impossible to interpret the learned filters. In contrast, our method is intuitive as it uses a graph-based representation for human skeleton.

Manifold learning techniques have been used for skeletal action recognition, where actions are represented as curves on Lie groups [28] and Riemannian manifold [5]. Deep learning on these manifolds is difficult [11] while deep learning on graphs (also a manifold) has developed recently [4, 14]. Our method uses a human skeleton graph and learns a model using part-based graph convolutional network, exploiting the benefits of deep learning on graphs.

## 2.2 Graph-based methods

Representing S-videos as skeleton graph sequences for recognizing actions had not been explored until recently. Li and Leung [17] construct graphs using a statistical variance measure dependent on joint distances and match them for recognition. Recently, Yan *et al.* [33] and Li *et al.* [16] proposed a spatio-temporal graph convolutional network for action recognition from S-videos. Both the methods construct graphs where the human skeleton is treated as a single graph. Our formulation explores a partitioned skeleton graph with a part-based graph convolutional network and we show that it improves recognition performance. Also, we use relative coordinates and temporal displacements as features at each vertex instead of 3D joint coordinates (see Figure 1(a)) which improves action recognition performance.

# 3 Background

A graph is defined as $\mathcal{G} = (\mathcal{V},\ \mathcal{E})$ where $\mathcal{V}$ is the set of vertices and $\mathcal{E} \subseteq (\mathcal{V} \times \mathcal{V})$ is the set of edges. $\mathbf{A}$ is the graph adjacency matrix having $\mathbf{A}(i,j) = w,\ w \in \mathbb{R} \setminus \{0\}$ if $(v_i, v_j) \in \mathcal{E}$ and $\mathbf{A}(i,j) = 0$ otherwise. $\mathcal{N}_k : v \rightarrow \mathcal{V}$ defines the set of vertices $\mathcal{V}$ in $k$-neighborhood of $v$ which includes neighbors having shortest path length atmost $k$ from vertex $v$. A labeling function $\mathbf{L} : \mathcal{V} \rightarrow \{0, 1, \ldots, \mathcal{L} - 1\}$ assigns a label to each vertex in a vertex set $\mathcal{V}$, where $\mathcal{L}$ is the number of unique labels. The adjacency matrix is normalized using a degree matrix as:

$$\mathcal{D}(i,i) = \sum_j \mathbf{A}(i,j); \quad \mathbf{A}^{\mathbf{norm}} = \mathcal{D}^{-1/2}\mathbf{A}\mathcal{D}^{-1/2} \tag{1}$$

Graph convolutions can be formulated using spectral graph theory [4] or spatial convolution [21] on graphs. We focus on spatial convolutions in this paper as they resemble convolutions on regular grid graphs like RGB images [21]. A graph CNN can then be formed by stacking multiple graph convolution units. Graph convolution (shown in Figure 2) can be defined as [21]:

$$\mathbf{Y}(v_i) = \sum_{v_j \in \mathcal{N}_k(v_i)} \mathbf{W}(\mathbf{L}(v_j))\mathbf{X}(v_j) \tag{2}$$

where, $v_i$ is the root vertex at which the convolution is centered (like center pixel in an image convolution), $\mathbf{W}(\cdot)$ is a filter weight vector of size of $\mathcal{L}$ indexed by the label assigned to neighbor $v_j$ in the $k$-neighborhood $\mathcal{N}_k(v_i)$, $\mathbf{X}(v_j)$ is the input feature at $v_j$ and $\mathbf{Y}(v_i)$ is the convolved output feature at root vertex $v_i$. Equation 2 can be written in terms of adjacency matrix as:

$$\mathbf{Y}(v_i) = \sum_j \mathbf{A}^{\mathbf{norm}}(i,j)\ \mathbf{W}(\mathbf{L}(v_j))\ \mathbf{X}(v_j) \tag{3}$$

$\mathbf{A}^{\mathbf{norm}}(i,j)$ basically defines the neighbors at distance 1 and hence, Equation 2 captures a more general form of convolution by using $k$-order neighborhood $\mathcal{N}_k(v_i)$.

## 3.1 Part-based Graph

Graphs representing real world manifolds can often be thought of as being made up of several parts. For instance, a graph representing a complex molecule consists of several simple structures, such as structure of a protein biomolecule, which can be divided into several polypeptide chains that make up the complex. Similarly, human body can be visualized as connected rigid parts, much like a deformable part-based model [8]. The graph of the skeleton of human body can be divided into parts, where each subgraph represents a part of the human body.

In general, a part-based graph can be constructed as a combination of subgraphs where each subgraph has certain properties that define it. Let us consider that a graph $\mathcal{G}$ has been divided into $n$ partitions. Formally:

$$\mathcal{G} = \bigcup_{p \in \{1,\ldots,n\}} \mathcal{P}_p \mid \mathcal{P}_p = (\mathcal{V}_p, \mathcal{E}_p) \tag{4}$$

$\mathcal{P}_p$ is the partition (or subgraph) $p$ of the graph $\mathcal{G}$. We consider scenarios in which the partitions can share vertices or have edges connecting them. We proceed to explain how the part-based graph convolution is defined for the part-based graph.

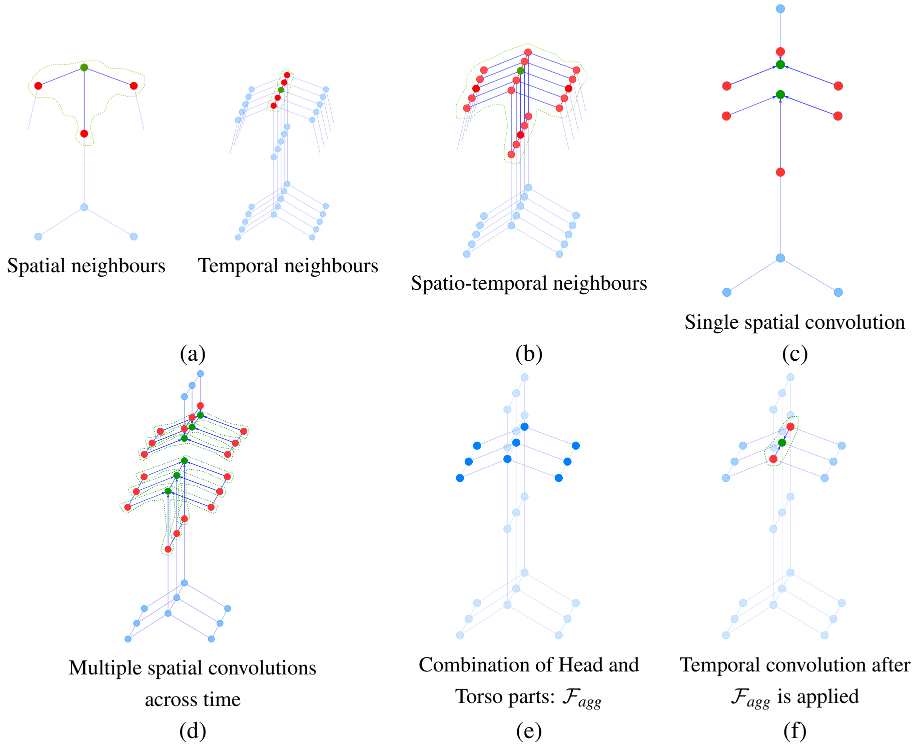

Spatial neighbours
Temporal neighbours
(a)
Spatio-temporal neighbours
(b)
Single spatial convolution
(c)
Multiple spatial convolutions across time
(d)
Combination of Head and Torso parts: $\mathcal{F}_{agg}$
(e)
Temporal convolution after $\mathcal{F}_{agg}$ is applied
(f)

Figure 3: Spatio-temporal neighborhood for root node (in green) and depiction of convolutions in space and time dimensions. Effect of application of $\mathcal{F}_{agg}$ is shown, where the common vertices are in darker shade.

## 3.2 Part-based Graph Convolutions

In essence, graph convolutions over parts are aimed at capturing high-level properties of parts and learn the relations between them. In a Deformable Part-based Model, different parts are identified and relations between them are learned through the deformation of the connections between them. Similarly, graph convolutions over a part identifies the properties of that subgraph and an aggregation across subgraphs learns the relations between them. For a part-based graph, convolutions for each part are performed separately and the results are combined using an aggregation function $\mathcal{F}_{agg}$. Using $\mathcal{F}_{agg}$ over edges across partitions:

$$\mathbf{Y}_p(v_i) = \sum_{v_j \in \mathcal{N}_{kp}(v_i)} \mathbf{W}_p(\mathbf{L}_p(v_j))\mathbf{X}_p(v_j), \;\; p \in \{1, \ldots, n\} \tag{5}$$

$$\mathbf{Y}(v_i) = \mathcal{F}_{agg}(\mathbf{Y}_{p1}(v_i), \mathbf{Y}_{p2}(v_j)) \mid (v_i, v_j) \in \mathcal{E}_{(p1,p2)}, \;\; (p1, p2) \in \{1, \ldots, n\} \times \{1, \ldots, n\} \tag{6}$$

Using $\mathcal{F}_{agg}$ for common vertices across partitions:

$$\mathbf{Y}(v_i) = \mathcal{F}_{agg}(\mathbf{Y}_{p1}(v_i), \mathbf{Y}_{p2}(v_i)) \mid (p1, p2) \in \{1, \ldots, n\} \times \{1, \ldots, n\} \tag{7}$$

The convolution parameters $\mathbf{W}_p$ can be shared across parts or kept separate, while the neighbors of $v_i$ only in that part ($\mathcal{N}_{kp}(v_i)$) are considered. In order to combine the information across parts, the function $\mathcal{F}_{agg}$ combines information at shared vertices (equation 7) or shares information through edges crossing parts (equation 6, $\mathcal{E}_{(p1,p2)}$ contains all edges connecting parts p1 and p2), according to the partition configuration. A sophisticated $\mathcal{F}_{agg}$ can be employed to make the model powerful. Using graph convolutions, part-based graph models can learn rich representations and we demonstrate the strength of this model through application to action recognition from S-videos.

# 4 Spatio-temporal Part-based Graph Convolutions

The S-videos are represented as spatio-temporal graphs. In order to include the temporal dimension, corresponding joints in each part are connected temporally. Figure 3(b) shows the spatio-temporal graph for *torso* over *five* frames. Adapting select-assemble-normalize (PATCHY-SAN) proposed by Niepert *et al.* [21] we present an overview of convolution formulation for our spatio-temporal graph by extending ideas from section 3.2. For in-depth understanding, we refer the reader to [21]. We perform a spatial convolution on each partition following equation 5, combine the convolved partitions using $\mathcal{F}_{agg}$ and perform temporal convolution on the graph obtained by aggregating the partitions. In effect, we spatially convolve each partition independently for each frame, aggregate them at each frame and perform temporal convolution on the temporal dimension of the aggregated graph. For a possible partitioning of human skeleton, this phenomenon is shown in Figure 3(c) for spatial convolution for a vertex common to torso and head, 3(d) for spatial convolutions in different frames, 3(e) for applying $\mathcal{F}_{agg}$ on head + torso and 3(f) for convolution on temporal dimension of the combined graph.

We first define the spatial and temporal neighborhood of a vertex in spatio-temporal graph and assign labels to the vertices in the neighborhoods, which is required to perform convolutions. For each vertex, we use 1-neighborhood $(k = 1)$ for spatial dimension $(\mathcal{N}_1)$ as the skeleton graph is not very large and a $\tau$-neighborhood $(k = \tau)$ for the temporal dimension $(\mathcal{N}_\tau)$. Figure 3(a) (dashed polygons) shows the spatial & temporal neighborhood for a **root** vertex. The different neighborhood sets for our model are defined as ($\mathbf{d}(v_i, v_j)$ = length of shortest path between $v_i$ and $v_j$):

$$\mathcal{N}_{1p}(v_i) = \{v_j \mid \mathbf{d}(v_i, v_j) \ \leq \ 1, \ v_i, v_j \in \mathcal{V}_p\} \tag{8}$$

$$\mathcal{N}_\tau(v_{it_a}) = \{v_{it_b} \mid \mathbf{d}(v_{it_a}, v_{it_b}) \ \leq \ \left\lfloor \frac{\tau}{2} \right\rfloor\} \tag{9}$$

where, $t_a$ & $t_b$ represent two time instants and $p \in \{1, \dots, n\}$ is the partition index. The set of vertices $\mathcal{V}_p$ differs for each part, with some vertices shared between parts (Figure 1(c)). As temporal convolution is performed on the aggregated spatio-temporal graph, $\mathcal{N}_\tau$ is not part-specific. Figure 3(a) shows the spatial and temporal neighborhoods for a **root** vertex in *torso*. For ordering vertices in the receptive fields (or neighborhoods), we use a single label spatially $(\mathbf{L}_S : \mathcal{V} \to \{0\})$ to weigh vertices in $\mathcal{N}_{1p}$ of each vertex equally and $\tau$ labels temporally $(\mathbf{L}_T : \mathcal{V} \to \{0, \dots, \tau - 1\})$ to weigh vertices across frames in $\mathcal{N}_\tau$ differently. The labeling functions are defined as:

$$\mathbf{L}_S(v_{jt}) = \{0 \mid v_{jt} \in \mathcal{N}_{1p}(v_{it})\} \tag{10}$$

$$\mathbf{L}_T(v_{it_b}) = \{((t_b - t_a) + \left\lfloor \frac{\tau}{2} \right\rfloor) \mid v_{it_b} \in \mathcal{N}_\tau(v_{it_a})\} \tag{11}$$

Using the labeled spatial and temporal receptive fields, we define the spatial and temporal convolutions as (adapted from [14]):

$$\mathbf{Y}_p(v_{it}) = \sum_{v_{jt} \in \mathcal{N}_{1p}(v_{it})} \mathbf{A}_p(i, j)\, \mathbf{Z}_p(v_{jt}) \mid p \in \{1, \dots, 4\} \tag{12}$$

$$\mathbf{Z}_p(v_{jt}) = \mathbf{W}_p(\mathbf{L}_S(v_{jt}))\ \mathbf{X}_p(v_{jt}) \tag{13}$$

$$\mathbf{Y}_S(v_{it}) = \mathcal{F}_{agg}(\{\mathbf{Y}_1(v_{it}), \dots, \mathbf{Y}_n(v_{it})\}) \tag{14}$$

$$\mathbf{Y}_T(v_{it_a}) = \sum_{v_{jt_b} \in \mathcal{N}_\tau(v_{it_a})} \mathbf{W}_T(\mathbf{L}_T(v_{it_b}))\, \mathbf{Y}_S(v_{it_b}) \tag{15}$$

where, $\mathbf{A}_p$ is a normalized adjacency matrix as explained in section 3 for part $p$. $\mathbf{L}_S$ for each part is same but $\mathcal{N}_{1p}$ is part-specific. $\mathbf{W}_p \in \mathbb{R}^{C' \times C \times 1 \times 1}$ is a part-specific channel transform kernel (pointwise operation) and $\mathbf{W}_T \in \mathbb{R}^{C' \times C' \times \tau \times 1}$ is the temporal convolution kernel. $\mathbf{Z}_p$ is the output from applying $\mathbf{W}_p$ on input features $\mathbf{X}_p$ at each vertex. $\mathbf{Y}_S$ is the output obtained after aggregating all partition graphs at one frame and $\mathbf{Y}_T$ is the output after applying temporal convolution on $\mathbf{Y}_S$ output of $\tau$ frames. We use a weighted sum fusion as our $\mathcal{F}_{agg}$:

$$\mathcal{F}_{agg}(\{\mathbf{Y}_1, \ldots, \mathbf{Y}_n\}) = \sum_i \mathbf{W}_{agg}(i)\, \mathbf{Y}_i \tag{16}$$

Human skeleton can be divided into two major components: (1) Axial skeleton and (2) Appendicular skeleton. The body parts included in these two components are shown in Figure 1(b). Human skeleton can be divided into parts based on these components. Different division schemes are shown in Figure 1(b), 1(c) and 1(d) and we use these schemes for experiments to test our PB-GCN.

For the final representation, we divide the human skeleton into *four* parts: **head**, **hands**, **torso** and **legs**, which corresponds to a division scheme where each of the axial and appendicular skeleton are divided into upper and lower components, as illustrated in Figure 1(c). We consider left and right parts of hands and legs together in order to be agnostic to *laterality* [31] (handedness / footedness) of the human when performing an action. To show how being agnostic to laterality is helpful, we divide the upper and lower components of appendicular skeleton into left and right (shown in Figure 1(d)), resulting in six parts and show results on it. To cover all natural connections between joints in skeleton graph, we include an overlap of atleast one joint between two adjacent parts. For example, in Figure 1(c), shoulder joints are common between the head and hands. For the lower appendicular skeleton (viz. legs), we also include the joint at the base of spine to get a good overlap with lower axial skeleton.

**Architecture and Implementation** We represent each subgraph by its adjacency matrix, normalized by corresponding degree matrix $\mathcal{D}$. Our model takes as input a tensor having features for each vertex in the spatio-temporal graph of S-video and outputs a vector of class scores for the video. The architecture of the graph convolutional network is similar to Yan *et al.* [33] and consists of 9 spatio-temporal graph convolution units (each unit with the *four* $\mathbf{W}_p$ kernels, *one* $\mathbf{W}_T$ kernel and a residual) with an initial spatio-temporal head unit, based on a Resnet-like model [9]. First three layers have 64 output channels, next three have 128 and last three have 256. We also use a learnable edge weight mask for learning edge weights in each subgraph [33]. We use the Pytorch framework [23] for our implementation. The code and models are made publicly available: https://github.com/dracarys983/pb-gcn.

## 5 Geometric & Kinematic Signals

Yan *et al.* [33] use the 3D coordinates of each joint directly as the signal at each graph node. Relative coordinates [12, 36] and temporal displacements [34] of joints have been used earlier for action recognition. Derived information like optical flow and Manhattan line map has been found useful on RGB images also [30, 37]. Even a CNN framework can be more effective and efficient if relevant derived information is supplied as input to the network.

We use a signal at each node that combines temporal displacements across time and relative coordinates, with respect to shoulders and hips [12]. This representation provides translation invariance to the representation [29] and improves skeletal action recognition performance significantly. Figure 1(a) illustrates the computation of the two signals for a single

(a) Performance with number of parts

| **#Parts** | **Accuracy** | |
|---|---|---|
| | CS | CV |
| One | 79.4 | 87.9 |
| Two | 80.2 | 88.4 |
| Four | **82.8** | **90.3** |
| Six | 81.4 | 89.1 |

(b) Performance with various signals for best & worst number of parts

| **Signals** | **Accuracy** | | | |
|---|---|---|---|---|
| | **#Parts=1** | | **#Parts=4** | |
| | CS | CV | CS | CV |
| $J_{loc}$ | 79.4 | 87.9 | 82.8 | 90.3 |
| $\mathbf{D}_R$ | 83.6 | 87.7 | 84.6 | 88.4 |
| $\mathbf{D}_T$ | 84.3 | 91.6 | 85.4 | 92.6 |
| $\mathbf{D}_R \| \| \mathbf{D}_T$ | **85.6** | **91.8** | **87.5** | **93.2** |

Table 1: Performance comparison for different number of parts in the skeleton graph and signals at vertices using our PB-GCN, on NTURGB+D [24] (CS: Cross Subject, CV: Cross View). The symbols for signals, $J_{loc}$: Absolute 3D joint locations, $\mathbf{D}_R$: Relative coordinates, $\mathbf{D}_T$: Temporal displacements and $\mathbf{D}_R||\mathbf{D}_T$: Concatenation of $\mathbf{D}_R$ and $\mathbf{D}_T$.

skeleton video frame. We show the effect of relative joint coordinates (geometric signal) and temporal displacements (kinematic signal) individually and the performance improvement obtained by using a combination of these signals for a baseline one-part model as well as our four part-based model in the Table 1(b). The improvement in performance obtained using the geometric and kinematic signals is noteworthy.

# 6 Experimental Setup and Results

We use SGD as the optimizer and run the training for 80 epochs (NTURGB+D) / 120 epochs (HDM05). We set the initial learning rate to 0.1 and all the experiments are run on a cluster with 4 Nvidia GTX 1080Ti GPUs. The batch size is set to 64. Learning rate decay schedule (set to decay by 0.1 at epochs 20, 50 and 70 for NTURGB+D, and at epoch 80 for HDM05) is finalized using a validation set. No augmentation is performed for any of the experiments, consistent with graph-based method [33]. We perform ablation studies on the large-scale NTURGB+D dataset (shown in Table 1) and then compare with state-of-the-art on both HDM05 and NTURGB+D using the best configuration of our model (shown in Table 2).

## 6.1 Datasets

**NTURGB+D** [24] This is currently the largest RGBD dataset for action recognition to the best of our knowledge. It has 56,880 video sequences shot with three Microsoft Kinect v2 cameras from different viewing angles. There are 60 classes among the action sequences and 3D coordinates of 25 joints are provided for each human skeleton tracked. There is a large variation in viewpoint, intra-class subjects and sequence lengths, which makes this dataset challenging. We remove 302 of the captured samples having missing or incomplete skeleton data. The protocol mentioned in Shahroudy *et al.* [24] is followed for comparisons with previous methods.

**HDM05** [20] This dataset was captured by using an optical marker-based Vicon system. It contains 2337 action sequences ranging across 130 motion classes performed by *five* actors. This dataset currently has the largest number of motion classes. The actors are named “bd”, “bk”, “dg”, “mm” and “tr”, and 31 joints are annotated for each skeleton. This dataset is challenging due to intra-class variations induced by multiple realizations of same action and large number of motion classes. We follow the protocol given in [10] which is used by recent deep learning methods.

## 6.2 Discussion

**Part-based graph model:** Our motivation to use a part-based graph model is derived primarily from the fact that human actions are made up of "gestures" which represent motion of a body part. The seminal success of DPMs [8] in detecting humans in images reinforces the motivation further. We discuss the effect of proposed spatio-temporal part-based graph model below.

**(a) How many parts to have?** We start with a coarse-grained scheme where entire skeleton is a single part and progress towards finer representations. The different partitions are, *two parts*: dividing skeleton into axial and appendicular skeleton, *four parts*: as explained in section 4 and *six parts*: Assigning left and right in hands and legs. The feature at each vertex in the input is 3D coordinate of the corresponding joint. From Table 1(a), we can see that using two parts improves over one and four improves over two. This shows that partitioning the skeleton graph into subgraphs with useful properties helps. However, dividing upper and lower skeletons into left and right in four part scheme does not improve performance, as per our intuition about *laterality* mentioned in section 4. This experiment suggests that part-based model improves performance over single part and being agnostic to laterality is helpful. Our final model uses the *four* part division of the human skeleton.

**(b) Comparison to graph-based models** From Table 2(a) and Table 1(b), it can be seen that our part-based model performs better than graph based model of Yan *et al.* [33] even when using $J_{loc}$ as the feature at each vertex. The graph construction in [33] uses a spatial partitioning scheme for their final model which divides the skeleton graph egde set into several partitions, while the vertex set has no partitions and contains all the joints. The difference in our model is that we divide the *entire* skeleton into *smaller* parts similar to human body parts and hence we use different edge set and vertex set for each part. Compared to graph based model of Li *et al.* [16], our model performs significantly better on NTURGB+D as well as HDM05. However, it is possible that this is because the number of layers in the network in [16] is much smaller (2 vs 9) compared to our model. Our model outperforms both the previous graph based models proposed for skeleton action recognition on the two datasets.

**Geometric + Kinematic signals:** Providing an explicit cue to a convolutional network, such as optical flow when performing action recognition from RGB videos [25], which is significant for the task at hand helps learn a richer representation by focusing on the cue. This motivates the use of geometric and kinematic features for skeletal action recognition. For the final configuration of our model, we concatenate the geometric and kinematic signals.

**(a) Kinematic: temporal displacements** Temporal displacements provide information about the amount of motion happening between two frames. This information is synonymous to 3D scene flow of a very sparse set of points. We hypothesize that these displacements provide explicit motion information (like optical flow) which makes the model consider displacements as strong features and learn from them. Improvement in performance using this signal can be seen from Table 1(b), for both four-part as well as one-part model across both splits of NTURGB+D.

**(b) Geometric: relative coordinates** These provide translation invariant features as explained in [29] and they have been used effectively to encode skeletons by Ke *et al.* [12] into images. Also, Zhang *et al.* [36] used relative coordinates as a geometric feature which performs much better than 3D joint locations using a simple stacked LSTM network. We

(a) NTURGB+D

| Methods | Accuracy | |
|---|---|---|
| | CS | CV |
| ST Attention [26] | 73.4 | 81.2 |
| GCA-LSTM [19] | 74.4 | 82.8 |
| TCN [13] | 74.3 | 83.1 |
| VA-LSTM [35] | 79.4 | 87.6 |
| CNN + MTLN [12] | 79.6 | 84.8 |
| Deep STGC [16] | 74.9 | 86.3 |
| STGCN [33] | 81.5 | 88.3 |
| PB-GCN | **87.5** | **93.2** |

(b) HDM05

| Methods | Accuracy |
|---|---|
| SPDNet [10] | 61.45 ± 1.12 |
| Lie Group [28] | 70.26 ± 2.89 |
| LieNet [11] | 75.78 ± 2.26 |
| P-LSTM [24] | 73.42 ± 2.05 |
| Deep STGC [16] | 85.29 ± 1.33 |
| STGCN [33] | 82.13 ± 2.39 |
| PB-GCN | **88.17 ± 0.99** |

Table 2: Performance comparison with previous methods on two benchmark datasets. The top group of results correspond to non-graph based methods and the middle corresponds to GCN based methods. PB-GCN is our part-based graph convolutional network. Evaluation protocols used: CS (Cross Subject) and CV (Cross View) for NTURGB+D [24]; 10-fold cross sample validation for HDM05 [10].

can see improvements in performance provided by relative coordinates in Table 1(b) for both global (one part) and four part-based models, which are the worst and best performing models according to Table 1(a).

### 6.3 Comparison to state of the art

**NTURGB+D:** For this dataset, we outperform all previous state-of-the-art methods by a large margin. Even without using the signals introduced in section 5, we outperform the previous methods which can be seen in Table 1(b) ($J_{loc}$ results). We outperform the previous state-of-the-art graph based method of Yan *et al.* [33] (STGCN) which is also the state-of-the-art for skeleton based action recognition to the best of our knowledge, by a margin of ~6% and ~5% for the two protocols.

**HDM05:** This is a ~20x smaller dataset compared to NTURGB+D but contains more than twice the number of classes in NTURGB+D. The length of sequences in this dataset is longer and some of the action classes have only one sequence [2]. Using the protocol of [10] is therefore very challenging, on which we obtain state-of-the-art results using our model. We outperform the previous state-of-the-art Deep STGC [16], which is a network based on spectral graph convolutions for skeleton action recognition by ~3% at the mean accuracy.

## 7 Conclusion

In this paper, we define a partition of skeleton graph on which spatio-temporal convolutions are formalized through a part-based GCN for the task of action recognition. Such a part-based GCN learns the relations between parts and understands the importance of each part in human actions more effectively than a model that considers entire body as a single graph. We also demonstrate the benefit of giving explicit cues to the convolutional model which are significant from the point of view of the task at hand, such as relative coordinates and temporal displacements for skeletal action recognition. As a result, our model achieves state-of-the-art performance on two challenging action recognition datasets. As a future work, we would like to explore the use of part-based graph model for tasks other than action recognition, such as object detection, measuring image similarity, etc.

# References

[1] C. Chen, R. Jafari, and N. Kehtarnavaz. Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In *IEEE International Conference on Image Processing (ICIP)*, 2015.

[2] Kyunghyun Cho and Xi Chen. Classifying and visualizing motion capture sequences using deep neural networks. *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, 2:122–130, 2014.

[3] Liu Chunhui, Hu Yueyu, Li Yanghao, Song Sijie, and Liu Jiaying. Pku-mmd: A large scale benchmark for continuous multi-modal human action understanding. *ACM Multimedia workshop*, 2017.

[4] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

[5] Maxime Devanne, Hazem Wannous, Stefano Berretti, Pietro Pala, Mohamed Daoudi, and Alberto Del Bimbo. 3-d human action recognition by shape analysis of motion trajectories on riemannian manifold. *IEEE transactions on cybernetics*, 45(7):1340–1352, 2015.

[6] Y. Du, Y. Fu, and L. Wang. Skeleton based action recognition with convolutional neural network. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 579–583, 2015.

[7] Yong Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1110–1118, 2015.

[8] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *Int. J. Comput. Vision*, 61(1):55–79, 2005.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Zhiwu Huang and Luc J Van Gool. A riemannian network for spd matrix learning. In *AAAI*, volume 2, page 6, 2017.

[11] Zhiwu Huang, Chengde Wan, Thomas Probst, and Luc Van Gool. Deep learning on lie groups for skeleton-based action recognition. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6099–6108. IEEE computer Society, 2017.

[12] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Sohel, and Farid Boussaid. A new representation of skeleton sequences for 3d action recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4570–4579. IEEE, 2017.

[13] Tae Soo Kim and Austin Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1623–1631. IEEE, 2017.

[14] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[15] Bo Li, Mingyi He, Yuchao Dai, Xuelian Cheng, and Yucheng Chen. 3d skeleton based action recognition by video-domain translation-scale invariant mapping and multi-scale dilated cnn. *Multimedia Tools and Applications*, pages 1–21, 2018.

[16] Chaolong Li, Zhen Cui, Wenming Zheng, Chunyan Xu, and Jian Yang. Spatio-temporal graph convolution for skeleton based action recognition. *AAAI Conference on Artificial Intelligence*, 2018.

[17] Meng Li and Howard Leung. Graph-based approach for 3d human skeletal action recognition. *Pattern Recognition Letters*, 87:195–202, 2017.

[18] Jun Liu, Amir Shahroudy, Dong Xu, and Gang Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *European Conference on Computer Vision*, pages 816–833. Springer, 2016.

[19] Jun Liu, Gang Wang, Ping Hu, Ling-Yu Duan, and Alex C Kot. Global context-aware attention lstm networks for 3d action recognition. In *CVPR*, 2017.

[20] Meinard MÃijller, Tido RÃűder, Michael Clausen, Bernhard Eberhardt, BjÃűrn KrÃijger, and Andreas Weber. Documentation mocap database hdm05, 2007.

[21] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023, 2016.

[22] Ferda Ofli, Rizwan Chaudhry, Gregorij Kurillo, René Vidal, and Ruzena Bajcsy. Sequence of the most informative joints (smij): A new representation for human skeletal action recognition. *Journal of Visual Communication and Image Representation*, 25 (1):24–38, 2014.

[23] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[24] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[25] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 568–576. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5353-two-stream-convolutional-networks-for-action-recognition-pdf.

[26] Sijie Song, Cuiling Lan, Junliang Xing, Wenjun Zeng, and Jiaying Liu. An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In *AAAI*, volume 1, page 7, 2017.

[27] Lingling Tao and René Vidal. Moving poselets: A discriminative and interpretable skeletal motion representation for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 61–69, 2015.

[28] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 588–595, 2014.

[29] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *CVPR 2018-IEEE Conference on Computer Vision & Pattern Recognition*, 2018.

[30] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016.

[31] Wikipedia. Definition of laterality. `https://en.wikipedia.org/wiki/Laterality`, 2015.

[32] Lu Xia, Chia-Chih Chen, and Jake K Aggarwal. View invariant human action recognition using histograms of 3d joints. In *Computer vision and pattern recognition workshops (CVPRW), 2012 IEEE computer society conference on*, pages 20–27. IEEE, 2012.

[33] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. *AAAI Conference on Artificial Intelligence*, 2018.

[34] Mihai Zanfir, Marius Leordeanu, and Cristian Sminchisescu. The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2752–2759, 2013.

[35] Pengfei Zhang, Cuiling Lan, Junliang Xing, Wenjun Zeng, Jianru Xue, and Nanning Zheng. View adaptive recurrent neural networks for high performance human action recognition from skeleton data. *arXiv, no. Mar*, 2017.

[36] Songyang Zhang, Xiaoming Liu, and Jun Xiao. On geometric features for skeleton-based action recognition using multilayer lstm networks. In *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pages 148–157. IEEE, 2017.

[37] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. Layoutnet: Reconstructing the 3d room layout from a single rgb image. *arXiv preprint arXiv:1803.08999*, 2018.