deep progressive reinforcement learning (DPRL) method to select the most informative frames of the input sequences and leverage GCN to learn the dependency among joints. Bin et al. [35] propose a spatio-temporal graph routing (STGR) scheme for skeleton-based action recognition, which learns both the spatial connectivity and temporal connectivity. However, the graph constructions in these methods have certain limitations: graphs in [32] are restricted by small partitions; graphs in [33] only model joints bridged by a bone; there is no explicit temporal graph in [34]; the computation complexity of graph learning in [35] is high, and the spatial graph is built over clusters, each of which is assigned a weight and thus may not capture delicate pairwise spatial relationship among joints.

Since the graph construction is crucial to graph convolution in GCNs, we propose a graph regression based GCN (GR-GCN) model to further improve the graph construction of skeleton data for stronger expressive power, providing an alternate view of the action sequence. The problem of learning the underlying graph structure from data (a.k.a., graph regression) is fundamental and helps discover the relation among graph signals. In the context of dynamic skeletons, we provide spatio-temporal modeling of skeletons and pose an optimization problem on the underlying graph Laplacian matrix[1] over consecutive frames. The optimization not only enforces the graph Laplacian to capture the structure of each spatio-temporal frame (i.e., every three consecutive frames), but also impose the sparsity constraint on the graph for compact representation. We then obtain the common structure of the graph Laplacian optimized from multiple observations of spatio-temporal frames by statistical analysis. The resulting graph not only connects each joint to its neighboring joints in the same frame strongly or weakly, but also links with relevant joints in the previous and subsequent frames.

After learning the common optimal graph for spatio-temporal frames in a dynamic skeleton sequence, we feed the optimized graph into the GCN along with the coordinates of the skeleton sequence for feature learning. We deploy high-order and fast Chebyshev approximation of spectral graph convolution [31], which leads to final classification scores. Further, we provide analysis of the variation characterization by the Chebyshev approximation. We analyze that the Chebyshev approximation essentially extracts the variation of the coordinates of joints, which is suitable to learn action features for final classification. As strong edges in the graph reflect strong relationship among physical/non-physical connections and weak edges represent potential relationship among non-physical connections, the proposed network strengthens learning actions which are accomplished by joints that are not bridged by bones (i.e., non-physical connections), such as "drink water" with the interaction between one hand and the head.

In summary, our contributions include the following aspects:

- We propose efficient graph regression to learn the underlying common graph of spatio-temporal frames in a dynamic skeleton sequence, by posing an optimization problem on the graph Laplacian from the constraints of data structure and sparsity.
- We integrate our graph regression with the GCN, and analyze the variation characterization by the Chebyshev approximation of spectral graph convolution, which leads to effective action feature learning.
- We achieve the state-of-the-art performance on the widely used NTU RGB+D, UT-Kinect and SYSU 3D datasets, and validate the effectiveness of the proposed graph regression.

The rest of the paper is organized as follows. Section II reviews previous works on skeleton-based action recognition and GCNs. Next, we introduce some basic concepts in spectral graph theory in Section III. Then, we present the proposed spatio-temporal graph regression and sparsified graph construction in Section IV, and elaborate on the proposed GR-GCN in Section V. Finally, experimental results and conclusions are presented in Section VI and Section VII, respectively.

II. Related Work

A. Skeleton-based Action Recognition

Previous skeleton-based action recognition methods can be divided into 2 classes [32]: hand-crafted feature based methods and deep learning methods.

Hand-crafted feature based methods. Hand-crafted features include covariance matrix for skeleton joint locations over time as a discriminative descriptor [37], modeling human actions as curves in the Lie group [14], and Spatio-Temporal Naive-Bayes Nearest-Neighbor [16], etc. However, these methods either lose information of interactions between specific sets of body parts or depend on complicated hand-crafted features.

Deep learning methods. Recent methods learn features via deep learning due to the notable performance, including RNN [6], [7], [17]–[23] and CNN [21], [24]–[27]. However, these methods typically lose structural information when converting the raw skeleton data into the grid-shaped input of the neural networks. A natural way to address this issue is to represent skeleton data on graphs. Yan et al. [32] and Li et al. [33] are the first to employ GCNs to automatically learn both the spatial and temporal patterns from data. Specifically, Yan et al. [32] construct graph convolution operations on partitions, which however may not capture the relationship among joints in different partitions due to the small receptive field. Li et al. [33] design multi-scale convolutional filters, and simultaneously perform local convolutional filtering on temporal motions and spatial structures. For each frame, an undirected graph is constructed, where only joints bridged by a bone are connected, whereas there is no explicit temporal connectivity. Tang et al. [34] propose a deep progressive reinforcement learning (DPRL) method to select the most informative frames of the input sequences and apply GCN to learn the spatial

[1]In spectral graph theory [36], a graph Laplacian matrix is an algebraic representation of the connectivities and node degrees of the corresponding graph, which will be introduced in Section III.

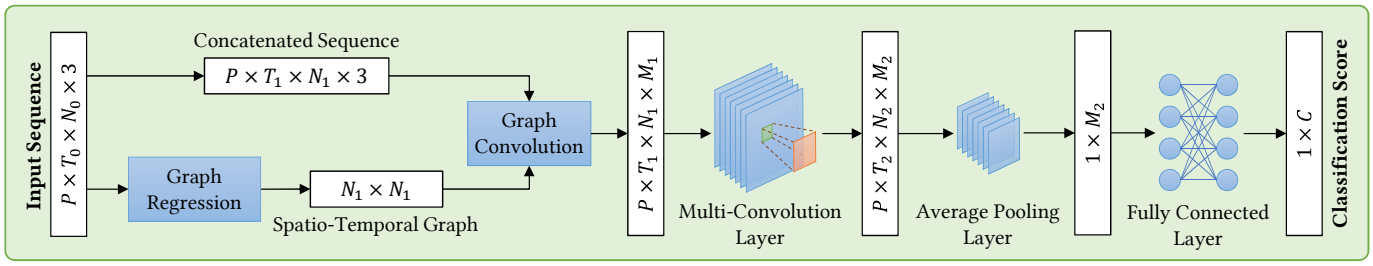

Fig. 2. **The architecture of the proposed GR-GCN for skeleton-based action recognition.** Our proposed network takes a skeleton sequence as the input, which goes through sequence concatenation and sparsified spatio-temporal graph construction before feeding into the network. We then employ graph convolution and standard 2D convolution to the concatenated sequence, followed by feature aggregation via average pooling. Thereafter, a fully-connected layer is utilized to generate the output classification scores for C classes.

dependency between the joints. Edges in the constructed graph reflect both intrinsic dependencies (i.e., physical connection) and extrinsic dependencies (i.e., physical disconnection) by different weights. Nevertheless, there is no explicit graph construction in the temporal domain. Bin et al. [35] propose a spatio-temporal graph routing (STGR) scheme for skeleton-based action recognition, which learns both spatial connectivity and temporal connectivity. Nevertheless, the computation complexity of the spatial and temporal graph learning is high.

B. Graph Convolutional Neural Networks

GCN extends CNN by consuming data defined on irregular grids. The key challenge is to define convolution over graphs, which is difficult due to the unordered data. According to the definitions of graph convolution, most of these methods can be divided into two main categories: spectral-domain methods and nodal-domain methods.

Spectral-domain methods. The convolution over graphs is elegantly defined in the spectral domain, which is the multiplication of the spectral-domain representation of signals. Specifically, the spectral representation is in the graph Fourier transform (GFT) [38] domain, where each signal is projected onto the eigenvectors of the graph Laplacian matrix [38], [39]. The computation complexity, however, is high due to the eigen-decomposition of the graph Laplacian matrix in order to get the eigenvector matrix. Hence, it is improved by [31] through fast localized convolutions, where the Chebyshev expansion is deployed to approximate GFT. Besides, Susnjara et al. introduce the Lancoz method for approximation [40]. Spectral GCN has shown its efficiency in various tasks such as segmentation and classification [30], [41].

Nodal-domain methods. Many techniques are introduced to implement graph convolution directly on each node and its neighbors, i.e., in the nodal domain. Gori et al. introduce recurrent neural networks that operate on graphs in [42]. Duvenaud et al. propose a convolution-like propagation to accumulate local features [29]. Bruna et al. deploy the multi-scale clustering of graphs in convolution to implement multi-scale representation [28]. Furthermore, Niepert et al. define convolution on a sequence of nodes and perform normalization afterwards [43]. Wang et al. propose edge convolution on graphs by incorporating local neighborhood information, which is applied to point cloud segmentation and classification [44]. Nodal-domain methods provide strong localized filters, but it also means it might be difficult to learn the global structure.

The above methods apply convolutional aggregators in the propagation step. Besides, there are other related works based on different aggregators, including attention aggregators [45], which incorporate the attention mechanism [46] into the propagation step, aiming to compute the hidden states of each node by attending over its neighbors; and gate aggregators [47]–[52], which use the gate mechanism like GRU [53] or LSTM [54] in the propagation step to improve the long-term propagation of information across the graph structure.

III. Preliminaries

We consider an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{A}\}$ composed of a vertex set $\mathcal{V}$ of cardinality $|\mathcal{V}| = n$, an edge set $\mathcal{E}$ connecting vertices, and a weighted *adjacency matrix* $\mathbf{A}$. $\mathbf{A}$ is a real symmetric $n \times n$ matrix, where $a_{i,j}$ is the weight assigned to the edge (i, j) connecting vertices i and j. We assume non-negative weights, *i.e.*, $a_{i,j} \geq 0$.

The Laplacian matrix, defined from the adjacency matrix, can be used to uncover many useful properties of a graph. Among different variants of Laplacian matrices, the *combinatorial graph Laplacian* used in [55], [56] is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \tag{1}$$

where $\mathbf{D}$ is the *degree matrix*—a diagonal matrix where $d_{i,i} = \sum_{j=1}^{n} a_{i,j}$. We will optimize $\mathbf{L}$ in the proposed graph regression method in Sec. IV. Further, the symmetric *normalized* Laplacian is defined as $\mathcal{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$, which will be deployed in the GCN so as to avoid numerical instabilities.

Graph signal refers to data that resides on the vertices of a graph, such as social, transportation, sensor, and neuronal networks. In our context, we treat each joint in a skeleton sequence as a vertex in a graph, and define the corresponding graph signal as the coordinates of each joint.

IV. Dynamic Skeleton Modeling

The fundamental of skeleton-based action recognition is to capture the variation of joints both in the spatial and temporal domain, so as to learn motion features for classification.

We propose spatio-temporal graph regression modeling for dynamic skeletons, and come up with the optimization of the underlying graph so as to characterize the variation.

A. Spatio-temporal Graph Regression Modeling of Skeletons

Let $\mathbf{x}_t \in \mathbb{R}^{n\times 3}$ be the coordinate signal in one frame at time t, where n is the number of joints in each skeleton. We define a *spatio-temporal frame* as $\mathbf{x} = [\mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1}]^\top \in \mathbb{R}^{3n\times 3}$, i.e., three consecutive frames are concatenated. We then represent $\mathbf{x}$ on a spatio-temporal graph described by $\mathbf{L}$, which models the correlation among joints.

We formulate the graph regression problem as the optimization of the graph Laplacian $\mathbf{L}$:

$$\begin{aligned} \min_{\mathbf{L}} \quad & \mathrm{tr}(\mathbf{x}^\top \mathbf{L}\mathbf{x}) + \beta\|\mathbf{L}\|_F^2, \\ \text{s.t.} \quad & \mathrm{tr}(\mathbf{L}) = 3n, \\ & \mathbf{L}_{i,j} = \mathbf{L}_{j,i} \le 0, i \ne j, \\ & \mathbf{L}\cdot\mathbf{1} = \mathbf{0}, \end{aligned} \tag{2}$$

where β is a weighting parameter, and $\mathbf{1}$ and $\mathbf{0}$ denote the constant one and zero vectors. In addition, $\mathrm{tr}(\cdot)$ and $\|\cdot\|_F$ denote the trace and Frobenius norm of a matrix, respectively. The first term in the objective function aims to fit the graph structure to the data by minimizing the *total variation* of the input signal (discussed soon), while the second term enforces the sparsity of the underlying graph for compact representation. The constraints ensure that the learned $\mathbf{L}$ satisfies the properties of the desired graph Laplacian: normalized, symmetry, non-negativity of edge weights, and the zero row sum. Next, we discuss the variation characterization by $\mathbf{L}$.

The quadratic term $\mathbf{x}^\top\mathbf{L}\mathbf{x}$ in Eq. (2) describes the total variation. This is because $\mathbf{x}^\top\mathbf{L}\mathbf{x}$ can be written as [57]:

$$\mathbf{x}^\top\mathbf{L}\mathbf{x} = \sum_{i\sim j} a_{i,j}(x_i - x_j)^2, \tag{3}$$

where $i \sim j$ denotes two vertices i and j are one-hop neighbors in the graph. Hence, $\mathbf{x}^\top\mathbf{L}\mathbf{x}$ computes the total variation among connected vertices in $\mathbf{x}$. By minimizing this term in Eq. (2), we enforce the edge weight between a pair of vertices with different features to be small, while allowing for a large edge weight between a pair of similar vertices. Thus, the optimized graph is able to characterize the variation in the skeleton data.

The optimization problem in Eq. (2) is convex and thus can be solved optimally, which leads to the learned graph for one given observation of $\mathbf{x}$. In order to acquire a spatio-temporal graph that captures the common structure of skeleton sequences, we propose to solve Eq. (2) over multiple observations of $\mathbf{x}$, and then statistically compute the common structure. For the purpose of succinct representation, we further restrict the connectivities of the common graph spatially and temporally, as discussed in the following.

B. Sparsified Graph Construction

The graph construction includes spatial connectivity and temporal connectivity.

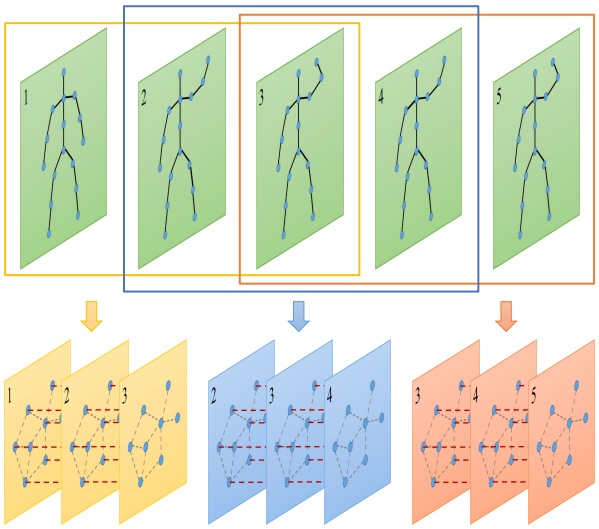

Fig. 3. **Illustration of the learned graph construction.** We learn a common structure of the spatio-temporal graph for spatio-temporal frames (i.e., every three adjacent frames). The yellow, blue, and red groups include three adjacent frames respectively, containing intra-frame connectivities (gray dotted lines) and inter-frame connectivities (red dotted lines). Note that the connectivities are simplified for clear visualization.

Spatial connectivity. Within each frame, we model the human body via a connected graph, based on two types of connectivities in particular: strong connections $\mathcal{E}_s$ and weak connections $\mathcal{E}_w$ for describing different correlations. Strong connections aim to capture strong correlations with large weights to emphasize the variation, including physical connectivity and some physical disconnection among joints, while weak connections are used to represent potential correlations among joints that are not physically connected. As shown in Fig. 1, whereas the "head" joint and "hand" joint are not bridged by a bone, a weak connectivity could be built between them because of the latent relationship during some actions (e.g., "drink water"). In particular, different weights are assigned to strong and weak edges, i.e., edge weights within a frame are set as

$$a_{i,j} = \begin{cases} w_1, & (i,j) \in \mathcal{E}_s \\ w_2, & (i,j) \in \mathcal{E}_w \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

where $w_1 > w_2$.

Temporal connectivity. Unlike previous works where each joint is disconnected in the temporal domain or only connected to its corresponding joints in the adjacent frames in general, we allow connecting each joint in frame $\mathbf{x}_t$ to the neighborhood of its correspondence in the previous frame $\mathbf{x}_{t-1}$ and subsequent frame $\mathbf{x}_{t+1}$, which are referred to as *potential edges*, as shown in Fig. 3. This is to capture the latent variation between one joint in frame $\mathbf{x}_t$ and its neighboring joints in the adjacent frames. The receptive field in the temporal domain is thus enlarged by exploiting more neighboring joints, which contributes to learning the temporal variation. Taking the action "typing on a keyboard" as an example, the left thumb may

have little motion in a short period. However, the left index finger moves relative to the left thumb both spatially and over time, which can be captured by the proposed potential edges. Hence, the final spatio-temporal adjacency matrix of consecutive frames $\{\mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1}\}$ is defined as

$$\mathbf{A}_g = \begin{bmatrix} \mathbf{A}_{t-1,t-1} & \mathbf{A}_{t-1,t} & \mathbf{O} \\ \mathbf{A}_{t,t-1} & \mathbf{A}_{t,t} & \mathbf{A}_{t,t+1} \\ \mathbf{O} & \mathbf{A}_{t+1,t} & \mathbf{A}_{t+1,t+1} \end{bmatrix}, \tag{5}$$

where $\mathbf{O} \in \mathbb{R}^{n\times n}$ is a zero matrix, $\mathbf{A}_{i,i} \in \mathbb{R}^{n\times n}$ is the weighted adjacency matrix of frame i for representing the intra-frame connectivity, while $\mathbf{A}_{i,j} \in \mathbb{R}^{n\times n}(i \neq j)$ is the adjacency matrix between frame i and j for description of the inter-frame connectivity.

Similar with the edge weights for the spatial connectivity, we define two types of temporal connectivities: the connectivity for corresponding joints, denoted as $\mathcal{E}_c$, and the connectivity between each joint and the neighborhood of its correspondence in the adjacent frames, denoted as $\mathcal{E}_n$. We assign w_1 to edges in $\mathcal{E}_c$, and assign w_2 to edges in $\mathcal{E}_n$, i.e.,

$$a_{i,j} = \begin{cases} w_1, & (i,j) \in \mathcal{E}_c \\ w_2, & (i,j) \in \mathcal{E}_n \\ 0, & \text{otherwise}, \end{cases} \tag{6}$$

where i and j denote vertices in two different frames.

C. Final Graph Modeling

Based on the above restriction of spatial and temporal connectivities, we extract the common structure of the optimized graph Laplacian learned from multiple observations of spatio-temporal frames. Specifically, we first randomly take m spatio-temporal frames from different classes of skeleton sequences, each of which serves as $\mathbf{x}$ in Eq. (2). Then we obtain the optimal spatio-temporal graph Laplacian for each spatio-temporal frame Eq. (2), leading to m optimized graph Laplacian $\{\mathbf{L}^l_{\text{opt}}\}^m_{l=1}$. Next, we derive a common graph Laplacian $\mathbf{L}$ from the statistics of $\{\mathbf{L}^l_{\text{opt}}\}^m_{l=1}$.

V. The Proposed GR-GCN

Having elaborated on the proposed graph regression that provides the underlying common structure of spatio-temporal frames, we now overview the architecture of the proposed GR-GCN. Then we discuss the corresponding graph convolution and feature learning in detail.

A. GR-GCN architecture

As illustrated in Fig. 2, the input is a skeleton-based action sequence organized as a $P\times T_0 \times N_0 \times 3$ tensor, where P is the number of actors in each sequence, T_0 is the number of frames, N_0 is the number of joints in each frame, and 3 means the dimension of x, y, z coordinates. In order to exploit the spatio-temporal dependencies, we firstly concatenate the input sequence in the unit of 3 consecutive frames, e.g., the $\{1,2,3\}^{\text{th}}$ frames are concatenated into the first spatio-temporal frame, and the $\{2,3,4\}^{\text{th}}$ frames into the second one, etc. Thus, the sequence length is changed to T_1, and the number of joints in each frame is N_1 after frame concatenation, where $T_1 = T_0 - 2$ and $N_1 = N_0 \times 3$. We then perform the proposed graph regression, which leads to the learned graph Laplacian of a common spatio-temporal graph. Secondly, we feed a feature matrix containing the coordinates of skeleton joints in the concatenated sequence and the graph Laplacian into the designed graph convolution layer and standard 2D convolution layers for feature extraction. Average pooling is then employed for feature aggregation. Finally, the global feature matrix will go through a fully connected layer followed by a Softmax activation function to output the classification score for C classes. Also, batch normalization is used for all layers before the ReLU activation function.

B. Spatio-Temporal Graph Convolution

Following the definition of graph convolution in [31], we adopt the approximation of spectral convolution by Chebyshev polynomials for efficient implementation:

$$g_\theta * \mathbf{x} \approx \sum_{k=0}^{K-1} \theta_k T_k(\mathcal{L})\mathbf{x}, \tag{7}$$

where $\mathcal{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$ is the symmetric normalized graph Laplacian as defined in Sec. III, which is employed because the domain of Chebyshev polynomials lies in $[-1,1]$. θ_k denotes the k-th Chebyshev coefficient and g_θ denotes a convolution kernel. $T_k(\mathcal{L})$ is the Chebyshev polynomial of order k. It is recurrently calculated by $T_k(\mathcal{L}) = 2\mathcal{L}T_{k-1}(\mathcal{L}) - T_{k-2}(\mathcal{L})$, where $T_0(\mathcal{L}) = 1, T_1(\mathcal{L}) = \mathcal{L}$. When $k > 1$, $\mathcal{L}^k$ essentially describes k-hop connectivities, thus incorporating more neighbors and leading to convolution over a larger receptive field.

We provide analysis of the variation characterization by the above Chebyshev approximation. As discussed in [58], the graph Laplacian matrix $\mathbf{L}$ is essentially a high-pass operator which captures the variation in the underlying signal. For any signal $\mathbf{x}$, it satisfies

$$(\mathbf{L}\mathbf{x})(i) = \sum_{j\in\mathcal{N}_i} a_{i,j}(x_i - x_j), \tag{8}$$

where $(\mathbf{L}\mathbf{x})(i)$ denotes the i-th component of $\mathbf{L}\mathbf{x}$. $\mathcal{N}_i$ is the set of vertices connected to i. This presents that when operating $\mathbf{L}$ on $\mathbf{x}$, for each vertex, it computes the signal difference among the vertex and its one-hop neighbors. In other words, $\mathbf{L}\mathbf{x}$ captures the variation in $\mathbf{x}$. Similarly, $\mathbf{L}^k\mathbf{x}$ captures the variation between each vertex and its k-hop neighbors. Thus, the approximated graph convolution seamlessly enables learning the variation in a skeleton sequence. This also sheds light on why graph convolution works for action recognition.

C. Feature Learning

Having designed the spatio-temporal graph convolution, we define the transfer function as follows:

$$\mathbf{y} = \text{ReLU}(\sum_{k=0}^{K-1} T_k(\mathcal{L})\mathbf{x}\mathbf{W}_k + \mathbf{b}), \tag{9}$$

where $\mathbf{W}_k \in \mathbb{R}^{F_1\times F_2}$ is a matrix of weight parameters θ_k as in Eq. 7, which will be learnt from the network, and F_1,

F_2 are the dimensions of generated features in two connected layers respectively. $\mathbf{b} \in \mathbb{R}^{n \times F_2}$ is the bias, while ReLU is an activation function.

After the graph convolution layer, we employ standard 2D convolution to the output $\mathbf{y}$, followed by feature aggregation via average pooling. Thereafter, a fully-connected layer and a Softmax activation function are adopted to generate the output classification scores. We adopt the categorical cross-entropy loss to train the network. The implementation details of our model will be discussed in Sec. VI-B.

VI. Experiments

We evaluate our proposed GR-GCN on four widely used datasets and compare with state-of-the-art skeleton-based action recognition methods. Experimental details and results are discussed below.

A. Datasets and Evaluation Metrics

NTU RGB+D Dataset [17]: This dataset was captured from 40 human subjects by 3 Microsoft Kinect v2 cameras. It consists of 56880 action sequences with 60 classes. Actions 1-49 were performed by one actor, and actions 50-60 were performed by the other two actors. Each body skeleton was recorded with 25 joints. The benchmark evaluations include Cross-Subject (CS) and Cross-View (CV). In the CS evaluation, 40320 samples from 20 subjects were used for training, and the other samples for testing. In the CV evaluation, samples captured from camera 2 and 3 were used for training, while samples from camera 1 were employed for testing.

Florence 3D Dataset [59]: This dataset contains 215 action sequences of 10 actors with 9 classes. Each body skeleton was collected from Kinect, and recorded with 15 joints. We follow the standard experimental settings to perform leave-one-actor-out validation protocol: we use all the sequences from 9 out of 10 actors for training and the remaining one for testing, and repeat this procedure for all the actors. The resulting 10 classification accuracy values are averaged to get the final accuracy.

UT-Kinect Dataset [10]: This dataset was captured using a single stationary Kinect. It consists of 200 sequences with 10 classes, and each skeleton includes 20 joints. The dataset was recorded by three channels: RGB, depth, and skeleton joint locations, whereas we only use the 3D skeleton joint coordinates. We also adopt the leave-one-actor-out validation protocol to evaluate our model on this dataset.

SYSU 3D Dataset [60]: On this dataset, 40 actors were asked to perform 12 different activities. Therefore, there are totally 480 action videos on this dataset. For each video, the corresponding RGB, depth, and skeleton information were captured by a Kinect. We use the skeleton sequences performed by 20 actors for training, and the remaining 20 actors for testing. We employ the 30-fold cross-subject validation and report the mean accuracy on the dataset.

B. Implementation Details

Our proposed model was implemented with the PyTorch[2] framework. The number of actors P is set to be 2, 1, 1, 1 for NTU RGB+D, Florence 3D, UT-Kinect, and SYSU 3D dataset respectively. We learn the edge weight ratio $r = 5$ for the four datasets, i.e., $w_1 = 5$, $w_2 = 1$.

Basic Model: Prior to the graph convolution layer, we set a Batch Normalization layer for the batched input data in order to be less careful about data initialization and speed up the training process [61]. In the graph convolution layer, we set the Chebyshev order K to be 4, and the dimension of the weight matrix $\mathbf{W}_k$ in Eq. 9 to be $3n \times 3n$ (i.e., the same as the spatio-temporal Laplacian matrix $\mathcal{L}$). The Multi-Convolution Layer consists of 2 standard CNN layers. Each convolution layer follows a Batch Normalization layer. We choose ReLU as the activation function after each convolution layer, and assign the dropout rate 0.5.

Deep Stacking: The above convolutional model can be easily extended into a deep architecture. Taking the above model as one basic layer, we stack it into a multi-layer network architecture, in which the output at the previous layer is used as the input of the next layer. Here, we stack it into a 10-layer architecture. In this architecture, we appropriately adjust the kernel size so as to acquire the final output feature of dimension $M_2 = 256$ for each point. With the increase of layers, the receptive field of convolutional kernels become larger, thus enabling abstracting more global information.

Next, we employ three average pooling layers to pool the P, N, and T dimension respectively, followed by a fully connected layer and a Softmax activation function to output the final classification score. The number of neurons depends on the output channel of the last convolution layer of the network. We apply Adam [62] optimizer to train the whole model with the initial learning rate 0.1, and decrease it on the 10^{th} epoch. Note that we did not perform any normalization on the skeleton coordinates during data preprocessing.

C. Data Preprocessing

NTU RGB+D Dataset: Due to some missing skeletons in this dataset, we only use the cleaned data[3] for action recognition [63]. In order to enhance the robustness of model training, we split the sequences into several segments of equal size in a way similar to [33]. Specifically, we split the whole sequence into 32 segments, and pick the $\{1, 2, 3, 4\}^{\text{th}}$ frame respectively from each segment to generate a large amount of training data.

Florence 3D Dataset: Since the sequences in this dataset contain few frames, we design two ways to generate the training set: sampling and interpolation. For longer sequences (i.e., the length of the sequence is greater than 32), we randomly choose 32 frames; for the other sequences, we calculate the mean of two adjacent frames and insert it into the sequence as a new frame, eventually forming a sequence

[2]https://pytorch.org

[3]https://github.com/InwoongLee/TS-LSTM

of 32 frames. For all the sequences, we repeat this operation 3 times to generate the training set.

UT-Kinect Dataset: We also adopt sampling and interpolation methods to generate the training set. Here, we set the length of each training sequence to be 64, and repeat the process twice.

SYSU 3D Dataset: Similar to the NTU RGB+D dataset, we split each sequence into 32 segments, and pick the $\{1, 2, 3, 4, 5\}^{\text{th}}$ frame from each segment to generate the training set. However, this dataset does not provide vertex labels, hence we only adopt the adjacency matrix of physical connections provided by the author as the graph within each frame.

D. Results on NTU RGB+D Dataset

As reported in Tab. I, our model achieves accuracy of 87.5% in CS and 94.3% in CV respectively. Also, as will be discussed in the ablation study, the proposed intra-connections improve the performance by 0.7% in CS and 1.4% in CV over the baseline method (*GR-GCN+Bone*), while the proposed temporal connectivities lead to 3.2% gain in CS and 3.1% gain in CV, thus validating the effectiveness of our method.

Comparison with the State-of-the-arts: We present the comparison with the state-of-the-art methods in Tab. I. We see that our method outperforms all the other state-of-the-art methods. Specifically, compared with the latest state-of-the-art method STGR-GCN [35], our model leads to 0.6% gain in CS and 2.0% gain in CV respectively, which demonstrates the superiority of our method.

Ablation Study: In order to validate the advantages of the proposed spatio-temporal graph construction in our method, we evaluate various graph construction methods progressively and design the following incomplete models. Model 1 is *GR-GCN (Bone only)*, in which only joints connected with a bone are linked with graph edges. This kind of graph construction is commonly used in existing graph-based skeleton recognition [32]–[34], and thus is the baseline. Model 2 is *GR-GCN (Bone + Intra-connection)* (non-physical), where connectivities are further added to joints that are not physically connected within each frame, including strong and weak edges for capturing latent dependencies. This kind of connectivities are previously exploited in [34]. Model 3 is our complete model with extra temporal connections included. We observe that Model 1 already achieves competitive performance with the state-of-the-art methods, which shows the effectiveness of the proposed GR-GCN. With additional intra-connectivities, Model 2 improves the accuracy by 0.7% in CS and 1.4% in CV over Model 1, validating the benefits of non-physical connections. Further, when the temporal connections are exploited, the complete model achieves 2.5% gain in CS and 1.7% gain in CV over Model 2. We thus conclude that both the proposed non-physical intra-connectivities and the explicit temporal connections make contributions to skeleton-based action recognition, in which the temporal connectivities are more crucial.

TABLE I
COMPARISONS ON THE NTU RGB+D DATASET (%).

Methods	CS	CV	Year
Dynamic Skeletons [60]	60.2	65.2	2015
Part-aware LSTM [17]	62.9	70.3	2016
Geometric Features [20]	70.3	82.4	2017
LSTM-CNN [21]	82.9	91.0	2017
Two-Stream CNN [24]	83.2	89.3	2017
ST-LSTM (Tree)+Trust Gate [23]	69.2	77.7	2018
Deep STGC_K [33]	74.9	86.3	2018
ST-GCN [32]	81.5	88.3	2018
DPRL [34]	83.5	89.8	2018
SR-TSL [64]	84.8	92.4	2018
STGR-GCN [35]	86.9	92.3	2019
GR-GCN (Bone only)	84.3	91.2	
GR-GCN (Bone + Intra-connection)	85.0	92.6	
Complete GR-GCN model	**87.5**	**94.3**	

E. Results on SYSU 3D Dataset

We compare our method with the state-of-the-art skeleton-based action recognition methods on SYSU 3D Dataset, which are presented in Tab. II. Our proposed method outperforms all the other state-of-the-art methods on this dataset, achieving accuracy improvement of 1.0% over the previous best method DPRL [34].

Note that, as vertex labels are not provided by this dataset, we can only build strong physical connections from the given adjacency matrix within each frame while abandoning weak edges. Hence, we provide ablation study with Model 1 in Tab. II. We see that our complete model achieves 2.7% improvement over the baseline method. This validates the benefits of incorporating explicit temporal connectivities across consecutive frames again.

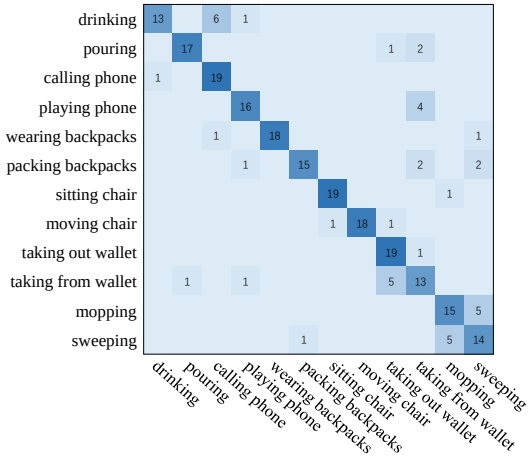

Fig. 4. Confusion matrix of GR-GCN on SYSU 3D dataset.

Also, the confusion matrix of our result is demonstrated in Fig. 4. We see that the matrix is diagonally dominant on all the 12 classes, which validates that our method achieves excellent classification results on this dataset. Besides, we note that our model sometimes confuses the activity of "mopping" with "sweeping", which is mainly due to the highly similar motions in the two actions.

TABLE II
COMPARISONS ON THE SYSU 3D DATASET (%).

Methods	Accuracy	Year
Dynamic Skeletons [60]	75.5	2015
LAFF (SKL) [65]	54.2	2016
ST-LSTM (Tree) [23]	73.4	2018
ST-LSTM (Tree) + Trust Gate [23]	76.5	2018
DPRL [34]	76.9	2018
GR-GCN (Bone only)	75.2	
Complete GR-GCN model	**77.9**	

TABLE III
COMPARISONS ON THE UT-KINECT DATASET (%).

Methods	Accuracy	Year
Lie Group [14]	97.1	2014
LARP+mfPCA [66]	94.9	2015
SPGK [13]	97.4	2016
ST-NBNN [16]	98.0	2017
Bi-LSTM [22]	96.9	2018
ST-LSTM(Tree) + Trust Gate [23]	97.0	2018
DPRL [34]	98.5	2018
GR-GCN (Bone only)	96.9	
GR-GCN (Bone + Intra-connection)	97.4	
Complete GR-GCN model	**98.5**	

F. Results on UT-Kinect Dataset

As listed in Tab. III, our method achieves comparable accuracy of 98.5% to [34], and outperforms all the other methods. Note that the performance difference among all the methods is rather small in general. The reason is that this dataset includes several very similar actions, which are difficult to distinguish without RGB or depth data.

Also, we perform the same ablation study as in Sec. VI-D, as reported in Tab. III. We observe that Model 2 improves the accuracy by 0.5% over Model 1 with additional intra-connectivities. Further, when the temporal connectivities are built, the complete model achieves 1.1% improvement over Model 2, which demonstrates the advantages of the proposed spatio-temporal graph construction.

G. Results on Florence 3D Dataset

We present the performance comparison with the state-of-the-art methods on the Florence 3D dataset in Tab. IV. Our method achieves classification accuracy of 98.5%, outperforming all the other state-of-the-art methods significantly except Deep STGC$_K$ [33]. The reason is that Deep STGC$_K$ benefits from the design philosophy of autoregressive moving average model, which is tailored for time sequences. Due to the few joints in each frame and few frames in the sequence, our model is difficult to capture subtle variation from few joints. Thus we misclassify "drink from a bottle" and "answer phone", "read watch" and "clap", which is difficult to distinguish even with human vision.

Moreover, Tab. IV reports the results of ablation study. We achieve 0.1% improvement from non-physical intra-connections compared with GR-GCN (Bone only), and another 2.8% improvement from temporal connections compared with GR-GCN (Bone + Intra-connection). This validates the effectiveness of the proposed graph construction, in which the temporal connectivities are vital.

TABLE IV
COMPARISONS ON THE FLORENCE 3D DATASET (%).

Methods	Accuracy	Year
Lie Group [14]	90.9	2014
LARP+mfPCA [66]	89.7	2015
Rolling Rotations [67]	91.4	2016
SPGK [13]	91.6	2016
Transion Forests [68]	94.2	2017
MIMTL [69]	95.3	2017
Bi-LSTM [22]	93.0	2018
Deep STGC$_K$ [33]	**99.1**	2018
GR-GCN (Bone only)	95.5	
GR-GCN (Bone + Intra-connection)	95.6	
Complete GR-GCN model	98.4	

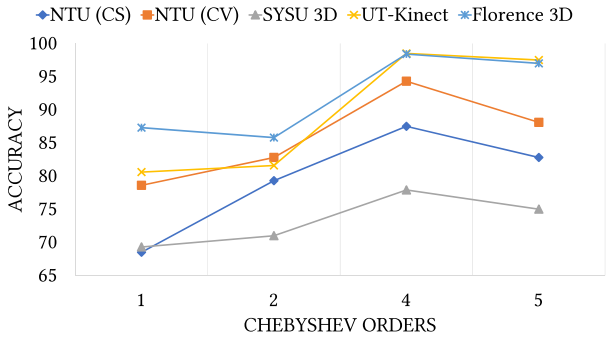

Fig. 5. Classification accuracy on different Chebyshev orders.

H. Analysis on Chebyshev Orders

We explore the effects of different Chebyshev polynomial orders on our complete GR-GCN model, as demonstrated in Fig. 5. When $K = 1$, graph convolution defaults to a fully connected layer according to Eq. (7), thus becoming the baseline with only traditional CNNs. The performance is inferior to those with larger K (corresponding to graph convolution with $(K-1)$-hop neighborhood) in general, thus validating the effectiveness of graph convolution. Further, our model achieves the best performance when $K = 4$ for all the datasets, thus validating the choice of K in the experimental setting. In contrast, the performance with $K = 5$ drops, because a wide range of neighbors will be incorporated, which is unable to capture the local variation well and may lead to overfitting.

VII. CONCLUSION

We propose a graph regression based GCN (GR-GCN) for skeleton-based action recognition, aiming to fully exploit both spatial and temporal dependencies among human joints. As the graph representation is crucial to graph convolution, we propose graph regression to optimize the underlying graph over multiple observations of spatio-temporal frames, and then statistically learn the common sparsified graph representation.

The learned graph not only captures intrinsic physical connections, but also models non-physical spatial connectivities as well as temporal connectivities over consecutive frames so as to represent the latent correlations for better action recognition. We then feed the learned spatio-temporal graph into the GCN with spectral graph convolution approximated by high-order Chebyshev polynomials for feature extraction. Extensive experiments demonstrate the superiority of our method.

REFERENCES

[1] Karen Simonyan and Andrew Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 568–576.

[2] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *IEEE International Conference on Computer Vision (CVPR)*, 2015, pp. 4489–4497.

[3] Limin Wang, Yu Qiao, and Xiaoou Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4305–4314.

[4] 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 (ECCV)*, 2016, pp. 20–36.

[5] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin, "Temporal action detection with structured segment networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, vol. 2.

[6] Yong Du, Wei Wang, and Liang Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1110–1118.

[7] 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 (ECCV)*, 2016, pp. 816–833.

[8] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake, "Real-time human pose recognition in parts from single depth images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1297–1304.

[9] Fei Han, Brian Reily, William Hoff, and Hao Zhang, "Space-time representation of people based on 3d skeletal data: A review," *Computer Vision and Image Understanding (CVIU)*, vol. 158, pp. 85–105, 2017.

[10] Lu Xia, Chia-Chih Chen, and Jake K Aggarwal, "View invariant human action recognition using histograms of 3d joints," in *IEEE Conference on Computer vision and pattern recognition workshops (CVPRW)*, 2012, pp. 20–27.

[11] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan, "Mining actionlet ensemble for action recognition with depth cameras," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1290–1297.

[12] Mohammad Abdelaziz Gowayyed, Marwan Torki, Mohamed Elsayed Hussein, and Motaz El-Saban, "Histogram of oriented displacements (hod): Describing trajectories of human joints for action recognition," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2013, vol. 13, pp. 1351–1357.

[13] Pei Wang, Chunfeng Yuan, Weiming Hu, Bing Li, and Yanning Zhang, "Graph based skeleton motion representation and similarity measurement for action recognition," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 370–385.

[14] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa, "Human action recognition by representing 3d skeletons as points in a lie group," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 588–595.

[15] Chunyu Wang, Yizhou Wang, and Alan L Yuille, "Mining 3d key-pose-motifs for action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2639–2647.

[16] Junwu Weng, Chaoqun Weng, and Junsong Yuan, "Spatio-temporal naive-bayes nearest-neighbor (st-nbnn) for skeleton-based action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4171–4180.

[17] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang, "Ntu rgb+d: A large scale dataset for 3d human activity analysis," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1010–1019.

[18] Wentao Zhu, Cuiling Lan, Junliang Xing, Wenjun Zeng, Yanghao Li, Li Shen, and Xiaohui Xie, "Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2016, vol. 2, p. 6.

[19] 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 Conference on Artificial Intelligence (AAAI)*, 2017, vol. 1, pp. 4263–4270.

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

[21] Chuankun Li, Pichao Wang, Shuang Wang, Yonghong Hou, and Wanqing Li, "Skeleton-based action recognition using lstm and cnn," in *International Conference on Multimedia & Expo Workshops (ICMEW)*, July 2017, pp. 585 – 590.

[22] Amor Ben Tanfous, Hassen Drira, and Boulbaba Ben Amor, "Coding kendall's shape trajectories for 3d action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2840–2849.

[23] Jun Liu, Amir Shahroudy, Dong Xu, Alex C Kot, and Gang Wang, "Skeleton-based action recognition using spatio-temporal lstm network with trust gates," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 40, pp. 3007 – 3021, 2018.

[24] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu, "Skeleton-based action recognition with convolutional neural networks," in *IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, 2017, pp. 597–600.

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

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

[27] Mengyuan Liu, Hong Liu, and Chen Chen, "Enhanced skeleton visualization for view invariant human action recognition," *Pattern Recognition (PR)*, vol. 68, pp. 346–362, 2017.

[28] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun, "Spectral networks and locally connected networks on graphs," *Computer Science*, 2013.

[29] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 2224–2232.

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

[31] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 3844–3852.

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

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

[34] Yansong Tang, Yi Tian, Jiwen Lu, Peiyang Li, and Jie Zhou, "Deep progressive reinforcement learning for skeleton-based action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5323–5332.

[35] Bin Li, Xi Li, Zhongfei Zhang, and Fei Wu, "Spatio-temporal graph routing for skeleton-based action recognition," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[36] Fan RK Chung, "Spectral graph theory," in *Conference Board of the Mathematical Sciences*. 1997, number 92, American Mathematical Society.

[37] Mohamed E Hussein, Marwan Torki, Mohammad Abdelaziz Gowayyed, and Motaz El-Saban, "Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations.," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2013, vol. 13, pp. 2466–2472.

[38] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis (ACHA)*, vol. 30, no. 2, pp. 129–150, 2011.

[39] Mikael Henaff, Joan Bruna, and Yann LeCun, "Deep convolutional networks on graph-structured data," *arXiv preprint arXiv:1506.05163*, 2015.

[40] Ana Susnjara, Nathanael Perraudin, Daniel Kressner, and Pierre Vandergheynst, "Accelerated filtering on graphs using lanczos method," *arXiv preprint arXiv:1509.04537*, 2015.

[41] Gusi Te, Wei Hu, Zongming Guo, and Amin Zheng, "Rgcnn: Regularized graph cnn for point cloud segmentation," in *ACM Multimedia Conference (MM)*, October 2018.

[42] Marco Gori, Gabriele Monfardini, and Franco Scarselli, "A new model for learning in graph domains," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2005, vol. 2, pp. 729–734.

[43] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov, "Learning convolutional neural networks for graphs," in *International Conference on Machine Learning (ICML)*, 2016, pp. 2014–2023.

[44] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon, "Dynamic graph cnn for learning on point clouds," *arXiv preprint arXiv:1801.07829*, 2018.

[45] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio, "Graph attention networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 5998–6008.

[47] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel, "Gated graph sequence neural networks," in *International Conference on Learning Representations (ICLR)*, 2016.

[48] Kai Sheng Tai, Richard Socher, and Christopher D Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *International Joint Conference on Natural Language Processing (IJCNLP)*, 2015.

[49] Victoria Zayats and Mari Ostendorf, "Conversation modeling on reddit using a graph-structured lstm," *Transactions of the Association of Computational Linguistics (TACL)*, vol. 6, pp. 121–132, 2018.

[50] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih, "Cross-sentence n-ary relation extraction with graph lstms," *Transactions of the Association for Computational Linguistics (TACL)*, vol. 5, pp. 101–115, 2017.

[51] Yue Zhang, Qi Liu, and Linfeng Song, "Sentence-state lstm for text representation," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[52] Xiaodan Liang, Xiaohui Shen, Jiashi Feng, Liang Lin, and Shuicheng Yan, "Semantic object parsing with graph lstm," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 125–143.

[53] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[54] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[55] Godwin Shen, W-S Kim, Sunil K Narang, Antonio Ortega, Jaejoon Lee, and Hocheon Wey, "Edge-adaptive transforms for efficient depth map coding," in *Picture Coding Symposium (PCS)*, 2010, pp. 566–569.

[56] Wei Hu, Gene Cheung, Antonio Ortega, and Oscar C Au, "Multiresolution graph fourier transform for compression of piecewise smooth images," *IEEE Transactions on Image Processing (TIP)*, vol. 24, no. 1, pp. 419–433, 2015.

[57] D. A. Spielman, "Lecture 2 of spectral graph theory and its applications," September 2004.

[58] David I Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, pp. 83–98, 2013.

[59] Lorenzo Seidenari, Vincenzo Varano, Stefano Berretti, Alberto Bimbo, and Pietro Pala, "Recognizing actions from depth cameras as weakly aligned multi-part bag-of-poses," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2013, pp. 479–485.

[60] Jian-Fang Hu, Wei-Shi Zheng, Jianhuang Lai, and Jianguo Zhang, "Jointly learning heterogeneous features for rgb-d activity recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5344–5352.

[61] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on International Conference on Machine Learning (ICML)*, July 2015, vol. 37, pp. 448 – 456.

[62] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *Computer Science*, 2014.

[63] Inwoong Lee, Doyoung Kim, Seoungyoon Kang, and Sanghoon Lee, "Ensemble deep learning for skeleton-based action recognition using temporal sliding lstm networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1012–1020.

[64] Chenyang Si, Ya Jing, Wei Wang, Liang Wang, and Tieniu Tan, "Skeleton-based action recognition with spatial reasoning and temporal stack learning," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 106–121.

[65] Jian-Fang Hu, Wei-Shi Zheng, Lianyang Ma, Gang Wang, and Jianhuang Lai, "Real-time rgb-d activity prediction by soft regression," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 280–296.

[66] Rushil Anirudh, Pavan Turaga, Jingyong Su, and Anuj Srivastava, "Elastic functional coding of human actions: From vector-fields to latent variables," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3147–3155.

[67] Raviteja Vemulapalli and Rama Chellapa, "Rolling rotations for recognizing human actions from 3d skeletal data," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4471–4479.

[68] Guillermo Garcia-Hernando and Tae-Kyun Kim, "Transition forests: Learning discriminative temporal transitions for action recognition and detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 432–440.

[69] Yanhua Yang, Cheng Deng, Shangqian Gao, Wei Liu, Dapeng Tao, and Xinbo Gao, "Discriminative multi-instance multitask learning for 3d action recognition," *IEEE Transactions on Multimedia (TMM)*, vol. 19, no. 3, pp. 519–529, 2017.