Non-Local Graph Convolutional Networks for Skeleton-Based Action Recognition

Lei Shi
Institute of Automation
Chinese Academy of Sciences
lei.shi@nlpr.ia.ac.cn

Yifan Zhang
Institute of Automation
Chinese Academy of Sciences
yfzhang@nlpr.ia.ac.cn

Jian Cheng
Institute of Automation
Chinese Academy of Sciences
jcheng@nlpr.ia.ac.cn

Hanqing Lu
Institute of Automation
Chinese Academy of Sciences
luhq@nlpr.ia.ac.cn

Abstract

Traditional deep methods for skeleton-based action recognition usually structure the skeleton as a coordinates sequence or a pseudo-image to feed to RNNs or CNNs, which cannot explicitly exploit the natural connectivity among the joints. Recently, graph convolutional networks (GCNs), which generalize CNNs to more generic non-Euclidean structures, obtains remarkable performance for skeleton-based action recognition. However, the topology of the graph is set by hand and fixed over all layers, which may be not optimal for the action recognition task and the hierarchical CNN structures. Besides, the first-order information (the coordinate of joints) is mainly used in former GCNs, while the second-order information (the length and direction of bones) is less exploited. In this work, a novel two-stream nonlocal graph convolutional network is proposed to solve these problems. The topology of the graph in each layer of the model can be either uniformly or individually learned by BP algorithm, which brings more flexibility and generality. Meanwhile, a two-stream framework is proposed to model both of the joints and bones information simultaneously, which further boost the recognition performance. Extensive experiments on two large-scale datasets, NTU-RGB+D and Kinetics, demonstrate the performance of our model exceeds the state-of-the-art by a significant margin.

1. Introduction

Action recognition methods based on skeleton data have been widely studied and drawn considerable attention, due to their strong adaptability to the dynamic circumstance and complicated background. Conventional deep learning based methods manually structure the skeleton to a joint coordinates vector or a pseudo-image, which is then fed into RNNs or CNNs to generate the prediction. However, representing the skeleton data as a vector sequence or a 2D grid cannot fully express the dependency between correlated joints. The skeleton is naturally structured as a graph in a non-Euclidean space with the joints as vertices and the natural connections in human body as edges. The previous methods cannot explicitly exploit the graph structure of skeleton data and are difficult to generalize to skeletons with arbitrary forms. Recently, graph convolutional networks (GCNs), which generalizes convolution from image to graph, has been successfully adopted in many applications. For skeleton-based action recognition, [22] firstly propose to apply GCN to directly model the skeleton data. They construct a spatial graph based on the physical connection of joints in human body and add the temporal edges between corresponding joints in consecutive frames. A distance-based sampling function is proposed for constructing graph convolution, which is then employed as the basic module to build the final spatiotemporal graph convolutional networks (ST-GCN).

However, the skeleton graph employed in ST-GCN, which is heuristically predefined, merely represents the physical structure and is not guaranteed to be suitable for action recognition task. For example, it cannot capture the dependency between two hands as they lie far away in the defined graph. Besides, the structure of CNN is hierarchical where different layers contain different-level semantic information. Nevertheless, the graph applied in ST-GCN is shared among all the layers, which lacks flexibility and is insufficient to model the multi-level semantic informa-

tion contained in all of the layers. Moreover, different samples may need different graphs, which can not be satisfied in the original ST-GCN. Inspired by Non-local neural networks [21], we propose a non-local graph convolutional neural networks to solve above problems. An adaptively learned global graph structure is set as the parameter of the model, which is trained and updated jointly with other parameters. This data-driven method increases the flexibility of the model and brings more generality to adapt to various tasks. Besides, different layers are applied with different graph parameters to better fit the hierarchical structure of CNNs. Moreover, an individual graph will be calculated according to each sample. The overall architecture of the non-local GCN block is shown in Figure 2.

Another notable problem of traditional methods for skeleton based action recognition is that the feature vector attached to each vertex only contain three coordinates of the joints, which can be regarded as the first-order information of the skeleton data. The second-order information, which capture the feature of bones between joints, is neglected. The skeletons are always visualized with both hinged joints and rigid bones. It is nonintuitive to only employ the joints information to classify human activities. To employ the second-order information of the skeletons, the feature of a bone is represented with a six-dimensional vector which contains its length and direction along three dimensions. Moreover, a two-stream framework is proposed to jointly model the first-order and second-order information, which is verified to be a good practice.

To verify the superiority of the proposed model, extensive experiments are performed on two large-scale datasets, NTU-RGBD[17] and Kinetics[9], where our model achieves the state-of-the-art performance on both of the datasets. The main contributions of our work include:

- A non-local graph convolutional block is proposed to adaptively learn the graph structure for different layers and samples, which is trained and updated jointly with model parameters in training process and can better suit the action recognition task.

- The second-order information, i.e. the length and direction of bones, is explicitly formulated and combined with the first-order information, i.e. joint coordinates, in a two-stream framework to further improve the recognition performance.

- On two large-scale datasets for skeleton-based action recognition, our model exceed the state-of-the-art by a significant margin.

2. Related work

2.1. Skeleton-based action recognition

Traditional methods design hand-crafted features for skeleton-based action recognition[20, 6]. Vemulapalli et al. [20] encode the skeletons with their rotations and translations in Lie group. Fernando et al. [6] leverage rank pooling method to represent the data with the parameters of ranker. With the development of deep learning, recurrent neural networks become popular due to its advantage for modeling sequence data. [4] apply a hierarchical bi-directional RNN model to identify the skeleton sequence, which divides the skeleton into different parts and sends them to different subnetworks. [19] embed a spatiotemporal attention module in LSTM based model, so that the network can automatically pay attention to the discriminant spatiotemporal region of the skeleton sequence. [23] introduce the mechanism of view transformation in LSTM based model, which automatically translates the skeleton data to a more advantageous angle for action recognition. Recently, CNN has shown the superiority owing to its good parallel ability and easier training process compared with RNN. [10] apply a one-dimensional residual CNN to identify skeleton sequence where the coordinates of joints are directly concatenated. [15] manually design 10 kinds of spatiotemporal images for skeleton encoding, and enhance these images using visual and motion enhancement methods. [14] employ both coordinates and motion information of joints as input, and carefully design a transformer to rearrange the order of joints. [12] employ multi-scale residual networks and various data-augmentation strategies for skeleton-based action recognition. Nevertheless, both the CNNs and RNN are failed to fully represent the structure of the skeleton, as they are embedded in the form of graphs instead of a 2D or 3D grids. [22] employ graph convolution to directly model the raw skeletons, which eliminates the need of hand-crafted part assignment or traversal rules.

2.2. Graph convolutional neural networks

There have been a lot of works for graph convolution, whose principle of constructing GCNs mainly follows two streams: spatial perspective and spectral perspective [1, 16, 8, 11, 5, 3]. Spatial perspective methods directly perform the convolution filters on the graph vertices and their neighbors, which are extracted and normalized based on the manually designed rules [16]. Different with the spatial perspective methods, spectral perspective methods utilize the eigenvalues and eigenvectors of graph Laplace matrices. It performs the graph convolution in frequency domain with the help of graph Fourier transform [18], which does not need to extract locally connected regions from graphs for each convolutional step [1, 3].

2.3. Non-local neural networks

The concept of non-local was first proposed in non-local means, which computes a weighted sum of all pixels in an image. It utilize all of the pixels according to their similarity with the center point. The idea is then successfully employed in many other applications. Recently, Wang et al. [21] propose the non-local neural network and achieve the remarkable performance in action recognition area. It present the non-local operations to capture long-range dependencies with deep neural networks, where each response of output feature map is calculated according to all of the features in input feature map.

3. Methods

In Section 3.1, we briefly introduce the original spatial-temporal graph convolutional network (ST-GCN). In Section 3.2, we present the designed non-local graph convolution block. In Section 3.3, we will describe the methods of utilizing the bone information to further boost the performance.

3.1. Revisit the ST-GCN

In ST-GCN [22], the skeleton graph is constructed with joints as vertices and bones as edges. In adjacent frames, the corresponding joints are connected as time edges. The attribute of each vertex is the coordinate vectors of the joint. The left sketch of Fig. 1 shows an example of the constructed spatial-temporal skeleton graph. Given the graph, multiple layers of spatial-temporal graph convolution operations are applied on the graph to get the high-level feature maps. The global average pooling and SoftMax classifier are then employed to predict the action categories.

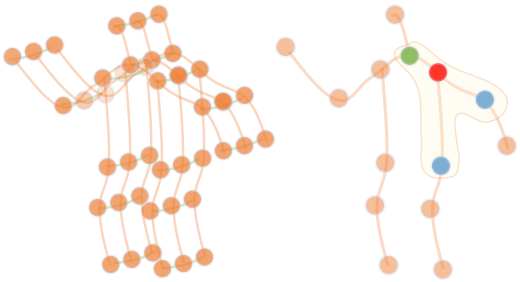

Figure 1. (a).The spatial-temporal graph of skeleton. (b).The partition strategy. Different colors denote different subsets.

For spatial dimension, each graph convolution operation is formulated as:

$$f_{out}(v_i) = \sum_{v_j \in \mathcal{B}(v_i)} \frac{1}{Z_i(v_j)} f_{in}(v_j) \cdot \mathbf{w}(l_i(v_j)) \qquad (1)$$

where f is the feature map and v is the vertex of the graph. $\mathcal{B}$ is the sampling area, which is defined as the 1-distance neighbor of the target vertex. As an example, it is showed as colored points in the right sketch of Fig. 1. $\mathbf{w}$ is the weighting function similar to the original convolution operation, which provides a weight vector corresponding to the given input. Note that the number of weight vectors is fixed while the vertexes in $\mathcal{B}$ is varied. A mapping function l_i is designed to map each vertexes with a unique weight vector. In detail, ST-GCN applies a partition strategy in the frame which divides the neighbors of a vertex into three subsets: 1).the vertex itself; 2).centripetal subset: the neighboring vertexes that are closer to the gravity center of the skeleton; 3).centrifugal subset: the neighboring vertexes that are further to the gravity center. $Z_i(v_j)$ is the number of subsets to normalize the result. The right sketch of Fig. 1 shows this strategy, where different colors represent different subsets and each color is corresponded with the individual learnable weight vector.

To implement the spatial graph convolution, the Eq. 1 is transformed into:

$$\mathbf{f}_{out} = \sum_{i}^{K_v} \mathbf{W}_i (\mathbf{f}_{in} \mathbf{\Lambda}_i^{-\frac{1}{2}} \mathbf{A}_i \mathbf{\Lambda}_i^{-\frac{1}{2}}) \otimes \mathbf{M}_i \qquad (2)$$

Here, $\mathbf{f}$ is the $C_{in} \times T \times N$ feature map where N denotes the number of vertexes, T denotes the temporal length and C_{in} denotes the number of input channels. $\mathbf{A}$ is similar with the $N \times N$ adjacency matrix, whose element A_{ij} indicates whether the vertex v_i is in the subset of vertex v_j. $\mathbf{\Lambda}_j^{ii} = \sum_k (\mathbf{A}_j^{ki}) + \alpha$ is the normalized diagonal matrix. α is set to 0.001 to avoid the empty rows in $\mathbf{A}$. K_v denotes the kernel size of spatial dimension. With partition strategy designed above, K_v is 3. $\mathbf{A_0} = \mathbf{I}$ which denotes the self-connections of vertexes. $\mathbf{A_1}$ denotes the connections of centripetal subset and $\mathbf{A_2}$ denotes the centrifugal subset. $\mathbf{W}_j$ is the $C_{out} \times C_{in} \times 1 \times 1$ weight vector of the 1×1 convolution operation. $\mathbf{M}$ is a $N \times N$ attention map which indicates the importance of each vertex. $\otimes$ denotes the element-wise matrix multiplication, which means it can only effect the vertexes that are connected with current target.

As for temporal dimension, because the number of neighbors for each vertex is fixed as 2 (the correspond joints in former frame and later frame), it is straightforward to perform the graph convolution similar with the classical convolution operation. Concretely, we perform a $K_t \times 1$ convolution on the output feature map calculated above.

3.2. Non-local graph convolutional networks

The spatial-temporal graph convolution for skeleton data described above is calculated based on the given graph $\mathcal{G}$, which is manually designed according to the natural connections of human body. However, it is hard to say this structure is the best choice for action recognition. For example, there is no connection between hand and head in

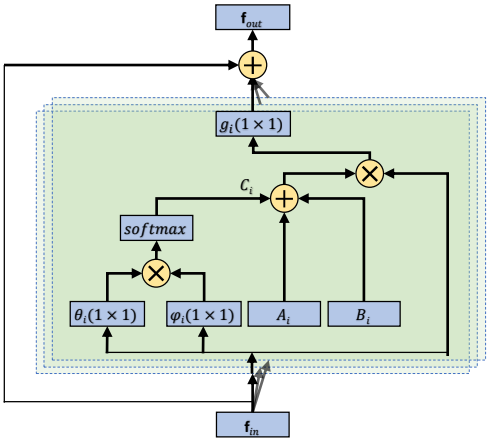

Figure 2. (a).The spatial-temporal graph of skeleton. (b).The partition strategy. Different colors denote different subsets.

the official provided graph of NTU-RGBD dataset. But for many actions such as wiping face and touching head, the relationship between hand and head may be important. So, the connection relationship should not be constrained in the adjacent joints. Besides, as the information is transferred from lower layers to higher layers, the semantic information contained in different layers is also varied. One shared graph structure can not adapt to this variety. The structure of graph should be updated along the message passing. Moreover, due to the deformation of skeleton for different classes, relations among joints in different samples should also be different.

To solve the problem, inspired by the non-local neural network, we propose the non-local graph convolution which directly focus on all of the joints to decide whether there are connections between pairs of vertexes. The graph structure is learned individually for different layers and samples in the training process in an end-to-end manner. Different with the original non-local block, our non-local graph convolution block contains three parts. The first part is the physical graph (A_i in Fig. 2) same as ST-GCN. The second part is a shared graph (B_i) which is same for different samples. It can represent the common pattern of connections between joints. The third part is a individual graph (C_i) which will learn a unique graph for different samples. It will capture the unique features of each sample. All of the three parts are important, which is verified in the ablation study in Section 4.3.

In detail, according to the Eq. 2, the structure of graph is actually decided by $\mathbf{A}_i$. For shared graph, we add 3 $N \times N$ learnable parameter $\mathbf{B}_i$ to represent the connections between each pair of joints. The value of each element denote not only whether there exist edge between two joints, but also the tightness of the connection or the similarly of the two joints. Instead of directly replace the original $\mathbf{A}_i$ with $\mathbf{B}_i$, we make it as a residual connection which is added to $\mathbf{A}_i$. The value of $\mathbf{B}_i$ is initialed with 0. In this way, it can strengthen the flexibility of model without harming the original performance. Note that it can also play the role of attention mechanism performed by $\mathbf{M}_i$ in Eq. 2 but is more flexible, because the element with 0 will always be 0 no matter what $\mathbf{M}_i$ is. We remove the $\mathbf{M}_i$ to avoid the redundant parameter and found it does not effect the performance.

For individual graph, we apply the embedded Gaussian function to calculate the similarity of two joints:

$$f(v_i, v_j) = e^{\theta(v_i)^T \phi(v_j)} \tag{3}$$

using 1×1 convolution to represent the embedding functions, the individual graph for each input feature map is calculated by:

$$\mathbf{C}_j = softmax(\mathbf{f_{in}}^T \mathbf{W}_{\theta j}^T \mathbf{W}_{\phi j} \mathbf{f_{in}}) \tag{4}$$

where the $softmax$ operation normalizes the result of product to $0-1$. $\mathbf{W}$ is the parameters of the 1×1 convolution and is initialed with 0. Same as the shared graph, it is also employed as a residual connection. Thus the Eq. 2 becomes:

$$\mathbf{f}_{out} = \sum_i^{K_v} \mathbf{W}_i \mathbf{f}_{in} (\mathbf{A}_i + \mathbf{B}_i + \mathbf{C}_i) \tag{5}$$

Fig. 2 shows the structure of the basic non-local graph convolution block. A residual connection, similar with [7], is added for each block to allow it being inserted into any existed models without breaking its initial behavior.

The convolution for temporal dimension is same as ST-GCN. Both the spatial GCN and temporal GCN are followed with the BN layer and Relu layer. One basic block is the combination of one spatial GCN, one temporal GCN and an additional dropout layer with drop rate as 0.5, showed in Fig. 3. Same as the original ST-GCN, a residual connection is added for each block.

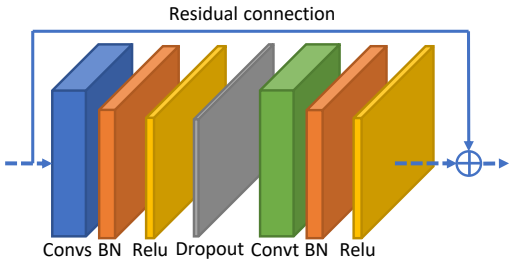

Figure 3. The basic non-local GCN block. Conv_s represent the spatial GCN and conv_t represent the temporal GCN, both are followed with the BN and Relu layers. Besides, a residual connection is added for each block.

The non-local graph convolutional network (NLGCN) is the stack of these basic blocks, showed in Fig. 4. There are totally 9 blocks. The number of output channels for each unit are 64, 64, 64, 128, 128, 128, 256, 256 and 256, respectively. After that, a global average pooling layer is performed and the final output is sent to a SoftMax classifier to get the prediction. More details can be found in the code, which will be released afterwards.

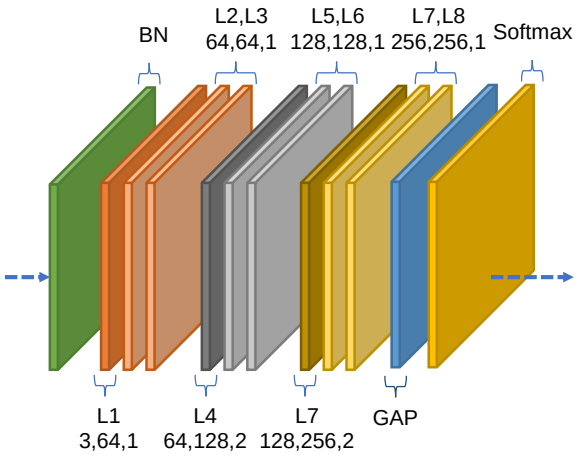

Figure 4. The structure of the NLGCN network. There are totally 9 NLGCN layers. The number of each layer represent the number of input channels, the number of output channels and the stride. GAP represent the global average pooling layer.

3.3. Two-stream networks

Traditional methods employ coordinates of body joints as input, which have three channels along x, y and z axises. It is the first-order information which only relies on the single joint. However, the second-order information, which represents the bones between joints, is also important for recognition but is neglected or not emphasized. In this sense, we propose to explicitly model the second-order information, the bone information, with another stream to boost the action recognition. In particular, the input graph of the new stream, which we call B-stream, has the same topological structure with the graph of original stream (J-stream). Each vertex of B-stream is a vector which represents the length and direction of the bone at the source of the current joint. Since the skeleton data has no ring, each joint can be assigned with a unique bone except for the central joint, which will be filled with 0. The bone vector is calculated by the difference between coordinates of two body joints along the x, y and z axises. The overall architecture of 2s-NLGCN is shown in Fig. 5. The $softmax$ scores of two streams are added to get the fused score and the final prediction. Other fusion methods are left as the future work.

4. Experiments

To have a head-to-head comparison with ST-GCN, our experiments are conducted on the same two large-scale action recognition datasets: NTU-RGB+D [17] and Kinetics [9]. In particular, we first perform exhaustive ablation studies on NTU-RGB+D dataset to examine the contributions of the proposed model components to the recognition performance because it is relatively small compared with Kinetics. Then the final model is evaluated on both NTU-RGB+D and Kinetics to verify the generality and is compared with other state-of-the-art approaches. The joints and connections of two datasets are showed in Fig. 6.

4.1. Datasets

NTU-RGB+D: NTU-RGB+D [17] is currently the largest and most widely used in-door captured action recognition dataset, which contains 56,000 action clips in 60 action classes. The clips are performed by 40 volunteers in different age groups ranging from 10 to 35. Each action is captured by 3 cameras at the same height but from different horizontal angles: $-45°, 0°, 45°$. The dataset provides 3D joint locations of each frame detected by the Kinect depth sensors. There are 25 joints for each subject in the skeleton sequences while each clip has no more than 2 subjects. The original paper [17] of the dataset recommends two benchmarks: 1). Cross-subject (X-Sub): the dataset in this benchmark is divided into training set (40,320 clips) and validation set (16,560 clips), where the actors in two subsets are different. 2).Cross-view (X-View): the training set in this benchmark contains 37,920 clips which are captured by camera 2 and 3, and the validation set contains 18,960 clips which are captured by camera 1. We follow this convention and report the top-1 accuracy on both benchmarks.

Kinetics: Kinetics [9] is a large-scale human action dataset which contains 300,000 videos clips in 400 classes. The video clips are sourced from YouTube videos and have a great variety. It only provides raw video clips without skeleton data. [22] estimate the location of 18 joints on every frame of the clips with the public available OpenPose toolbox [2]. 2 peoples are selected for multi-person clips based on the average joint confidence. We use their released data to evaluate our model. The dataset is divided into training set (240,000 clips) and validation set (20,000 clips). Follow the evaluation method in [9], we train the models on the training set and report the top-1 and top-5 accuracies on the validation set.

4.2. Training details

4.3. Ablation Study

We examine the effectiveness of the proposed components in two-stream non-local graph convolutional network

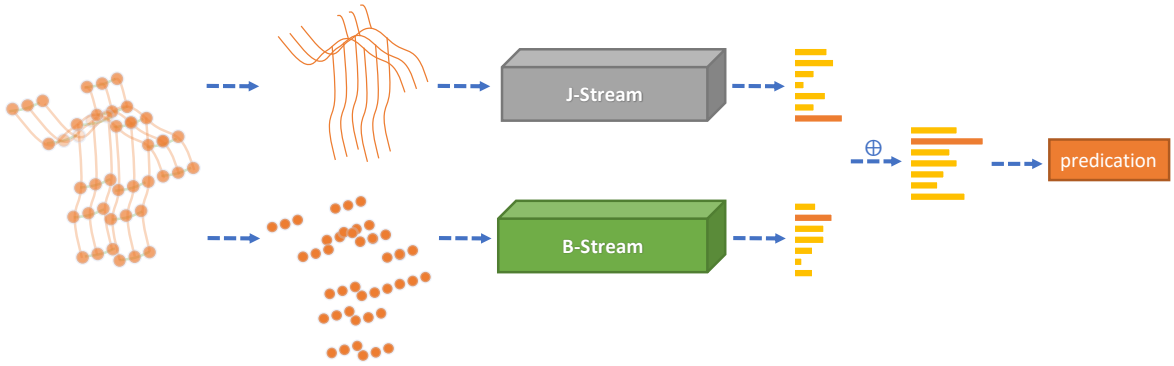

Figure 5. The overall architecture of 2s-NLGCN. The scores of two stream are added to get the final prediction.

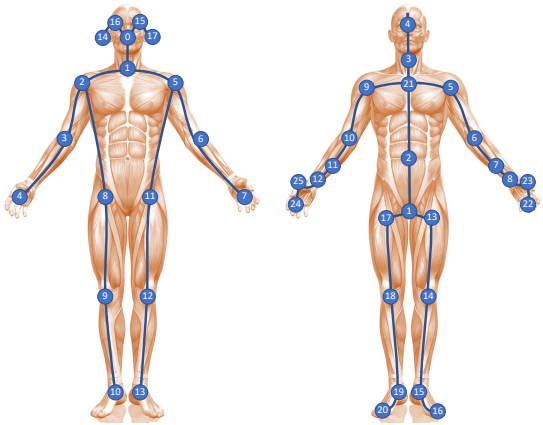

Figure 6. Left is the joint label of Kinetics dataset and right is the joint label of NTU-RGB+D dataset.

Methods	Accuracy (%)
STGCN	92.7
NLGCN wo/A	93.4
NLGCN wo/B	93.3
NLGCN wo/C	93.4
NLGCN	93.7

Table 1. Comparison of the recognition accuracies when adding NLGCN block with or without A, B and C.

(2s-NLGCN) in this section with the X-View benchmark on NTU-RGB+D dataset.

4.3.1 Non-local GCN.

As introduced in Section 3.2, there are 3 parts for in NL-GCN block, i.e. A, B and C. We manually delete one of the part and show their performance in Tab. 4.3.1. It shows that adaptively learning the graph is benefit for action recognition and deleting anyone of the three parts will harm the performance. With all three parts together, the model get the best performance.

4.3.2 Visualization of NLGCN block

Fig.7 shows an example of the learned adjacent matrix by our model for second subset. The color of each element A_{ij} in matrix represents the tightness of the connection between joint i and joint j. The left is the original matrix employed in ST-GCN, which is same for different layers and different samples. The right is the learned matrix of the first layer of one sample. It is obviously different from the original matrix, which is more flexible and not constrained to the physical connections of the human body.

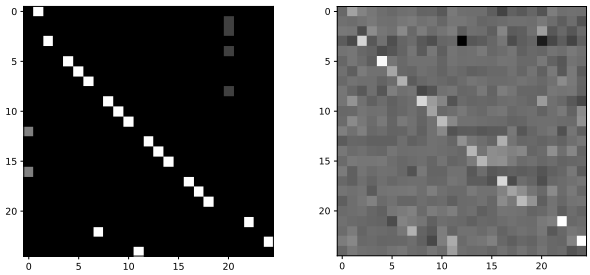

Figure 7. Left is the original adjacent matrix for second subset. Right is the non-local adjacent matrix after adding B_j and C_j in Eq. 5. The color of each element A_{ij} in matrix represents the tightness of the connection between joint i and joint j.

Fig.8 shows an example of edges connected to 25_{th} joint in NTU-RGB+D dataset learned by our model. Each circle represents one joint, whose size indicates the tightness of

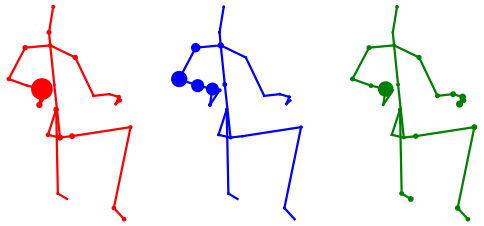

Figure 8. Visualization of the edges for 25_{th} joint of right sketch in Fig.6. The size of the circle represents the tightness of the connection. From left to right is the visualization of different layers (3_{th}, 5_{th} and 7_{th} layer in Fig. 4).

the connection between current joint and the 25_{th} joint. The examples showed from left to right are the visualizations of the second subset of 3_{th}, 5_{th} and 7_{th} layers in Fig. 4, respectively. It shows the connection and their tightness is individual for different layers. It verified our viewpoint that different layers contain different-level semantic information, which should own distinct graphs.

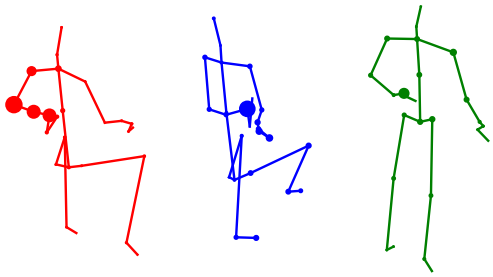

Figure 9. Visualization of different samples.

Fig.9 shows the edges connected to 25_{th} joint in NTU-RGB+D dataset for different samples. The learned parameter (adjacent matrix) is extracted from the second subset of 5_{th} layer in model (Fig. 4). It shows the model learned different connections for different samples, which confirms our point of view and is more intuitive.

4.3.3 Two-stream.

Another important improvement is the utility of second-order information introduced in Section 3.3. Here we compare the performance of using each kind of input data alone, showed as Js-NLGCN and Bs-NLGCN in Tab. 4.3.3, and the performance when combine them as described in Section 3.3, showed as 2s-NLGCN in Tab. 4.3.3. It shows that combining the two kinds of data as input outperforms one stream based methods, which verifies the importance of the second-order information (bones information) for skeleton based action recognition.

Methods	Accuracy (%)
Js-NLGCN	93.7
Bs-NLGCN	93.2
2s-NLGCN	95.1

Table 2. Recognition accuracy with different input modality.

4.4. Comparison with the state-of-the-art

We compare the final model with the state-of-the-art skeleton-based action recognition methods in both NTU-RGB+D dataset and Kinetics dataset.

4.4.1 NTU-RGB+D dataset

In NTU-RGB+D dataset, our model is compared with one hand-craft feature based methods, i.e. Lie Group [20], three LSTM based methods, i.e. HBRNN [4], STA-LSTM [19] and VA-LSTM [23], three CNN based methods, i.e. TCN [10], Synthesized CNN [15], Motion+Trans+CNN [13], 3scale ResNet152 [12] and one graph convolution based method, i.e. ST-GCN [22]. These methods are briefly introduced in Section 2. Table 4.4.1 shows that the performance of deep learning based methods is generally better than hand-craft feature based methods, and CNN based methods are generally better than RNN based methods. Our model outperforms these methods even without using data-augmentation skills, which verifies the superiority of our model for skeleton-based action recognition.

Methods	X-Sub (%)	X-View (%)
Lie Group [20]	50.1	82.8
HBRNN [4]	59.1	64.0
STA-LSTM [19]	73.4	81.2
VA-LSTM [23]	79.2	87.7
Temporal Conv. [10]	74.3	83.1
Synthesized CNN [15]	80.0	87.2
Motion+Trans+CNN	83.2	89.3
3scale ResNet152 [12]	85.0	92.3
ST-GCN [22]	81.5	88.3
2s-NLGCN (ours)	88.5	95.1

Table 3. Compare with the state-of-the-art methods in NTU-RGB+D.

4.4.2 Kinetics dataset

In Kinetics, our model is compared with one hand-crafted features based method, i.e. "Feature Encoding" [6], one LSTM base methods, i.e. Deep LSTM [17], one CNN based methods, i.e. TCN [10] and one graph convolution based method, i.e. ST-GCN [22]. These methods are briefly introduced in Section 2. The top-1 and top-5 recognition accu-

racies are reported in Table 4.4.2, where our model outperforms the other methods with a large margin.

Methods	Top-1 (%)	Top-5 (%)
Feature Enc. [6]	14.9	25.8
Deep LSTM [17]	16.4	35.3
TCN [10]	20.3	40.0
ST-GCN [22]	30.7	52.8
Js-NLGCN (ours)	35.1	57.1
Bs-NLGCN (ours)	33.3	55.7
2s-NLGCN (ours)	36.1	58.7

Table 4. Compare with the state-of-the-art methods in Kinetics.

4.5. Conclusion

In this work, we propose a non-local graph convolution block for skeleton based action recognition, which can overcome the weakness of manual design of graph in ST-GCN. Furthermore, we found that not only the joint but also the bone information is important for action recognition. So we represent the bone information with bone vector and propose a two-stream network to separately model the two input. The final model is evaluated on two large-scale action recognition datasets, NTU-RGB+D and Kinetics, and achieves the state-of-the-art performance.

References

[1] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral Networks and Locally Connected Networks on Graphs. In *ICLR*, 2014.

[2] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017.

[3] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3844–3852. Curran Associates, Inc., 2016.

[4] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1110–1118, 2015.

[5] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2224–2232. Curran Associates, Inc., 2015.

[6] B. Fernando, S. Gavves, O. Mogrovejo, J. Antonio, A. Ghodrati, and T. Tuytelaars. Modeling video evolution for action recognition. In *Proceedings CVPR 2015*, pages 5378–5387, 2015.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. bibtex: He_2016_CVPR.

[8] M. Henaff, J. Bruna, and Y. LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[9] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, and others. The Kinetics Human Action Video Dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[10] T. S. Kim and A. 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, 2017. bibtex: kim2017interpretable bibtex[organization=IEEE].

[11] T. N. Kipf and M. Welling. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907 [cs, stat]*, Sept. 2016. arXiv: 1609.02907.

[12] B. Li, Y. Dai, X. Cheng, H. Chen, Y. Lin, and M. He. Skeleton based action recognition using translation-scale invariant image mapping and multi-scale deep CNN. In *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on*, pages 601–604. IEEE, 2017.

[13] C. Li, Q. Zhong, D. Xie, and S. Pu. Skeleton-based action recognition with convolutional neural networks. In *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on*, pages 597–600. IEEE, 2017.

[14] H. Liu, J. Tu, and M. Liu. Two-Stream 3d Convolutional Neural Network for Skeleton-Based Action Recognition. *arXiv:1705.08106 [cs]*, May 2017. arXiv: 1705.08106.

[15] M. Liu, H. Liu, and C. Chen. Enhanced skeleton visualization for view invariant human action recognition. *Pattern Recognition*, 68:346–362, 2017. bibtex: liu2017enhanced bibtex[publisher=Elsevier].

[16] M. Niepert, M. Ahmed, and K. Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023, 2016.

[17] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang. NTU RGB+D: A Large Scale Dataset for 3d Human Activity Analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. bibtex: Shahroudy_2016_CVPR.

[18] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013. arXiv: 1211.0053.

[19] S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu. An End-to-End Spatio-Temporal Attention Model for Human Action Recognition from Skeleton Data. In *AAAI*, volume 1, page 7, 2017.

[20] R. Vemulapalli, F. Arrate, and R. 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.

[21] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local Neural Networks. *arXiv:1711.07971 [cs]*, Nov. 2017. arXiv: 1711.07971.

[22] S. Yan, Y. Xiong, and D. Lin. Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition. In *AAAI*, 2018. bibtex: stgcn2018aaai.

[23] P. Zhang, C. Lan, J. Xing, W. Zeng, J. Xue, and N. Zheng. View Adaptive Recurrent Neural Networks for High Performance Human Action Recognition From Skeleton Data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2126, 2017. bibtex: zhang2017view.