Skeleton-Based Action Recognition with Shift Graph Convolutional Network

Ke Cheng[1,2], Yifan Zhang[1,2*], Xiangyu He[1,2], Weihan Chen[1,2], Jian Cheng[1,2,3], Hanqing Lu[1,2]
[1]NLPR & AIRIA, Institute of Automation, Chinese Academy of Sciences
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences
[3]CAS Center for Excellence in Brain Science and Intelligence Technology
{chengke2017, chenweihan2018}@ia.ac.cn, {yfzhang, xiangyu.he, jcheng, luhq}@nlpr.ia.ac.cn

Abstract

Action recognition with skeleton data is attracting more attention in computer vision. Recently, graph convolutional networks (GCNs), which model the human body skeletons as spatiotemporal graphs, have obtained remarkable performance. However, the computational complexity of GCN-based methods are pretty heavy, typically over 15 GFLOPs for one action sample. Recent works even reach ∼100 GFLOPs. Another shortcoming is that the receptive fields of both spatial graph and temporal graph are inflexible. Although some works enhance the expressiveness of spatial graph by introducing incremental adaptive modules, their performance is still limited by regular GCN structures. In this paper, we propose a novel shift graph convolutional network (Shift-GCN) to overcome both shortcomings. Instead of using heavy regular graph convolutions, our Shift-GCN is composed of novel shift graph operations and lightweight point-wise convolutions, where the shift graph operations provide flexible receptive fields for both spatial graph and temporal graph. On three datasets for skeleton-based action recognition, the proposed Shift-GCN notably exceeds the state-of-the-art methods with more than $10\times$ *less computational complexity.*

1. Introduction

In the computer vision field, skeleton-based human action recognition has attracted much attention due to its robustness against the dynamic circumstance and complicated background [2, 12, 14, 16, 20, 21, 23–25, 27, 30, 34, 36].

Earlier methods [2, 3, 27] simply employ the joint coordinates to form feature vectors, which rarely explore the relations between body joints. With the development of deep learning, researchers manually structure the skeleton data as a pseudo-image [5, 7, 10, 14] or a sequence of coordinate vectors [2, 16, 19, 35, 36], which is fed into CNNs or RNNs to generate the prediction. Recently, Yan et al. [34] propose ST-GCN to model the skeleton data with graph convolutional networks (GCNs), which contains spatial graph convolution and temporal graph convolution. Many variants of ST-GCN are proposed [9, 12, 20–22, 31], which typically introduce incremental modules to enhance the expressiveness and network capacity.

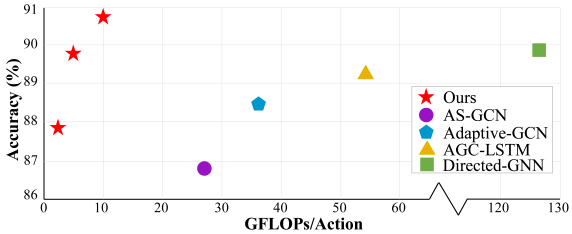


Figure 1. GFLOPs *v.s.* accuracy on NTU RGB+D X-sub task.

However, there are two shortcomings of these GCN-based methods. (1) The computational complexity is too heavy. For example, ST-GCN [34] costs 16.2 GFLOPs[1] for one action sample. Some recent works even reach ∼100 GFLOPs [20] due to introducing incremental modules and multi-stream fusion strategy. (2) The receptive fields of both spatial graph and temporal graph are pre-defined heuristically. Although [20, 21] makes the spatial adjacent matrix learnable, our experiments show that their expressiveness is still limited by the regular spatial GCN structure.

In this paper, we propose shift graph convolutional network (Shift-GCN) to address both shortcomings. Our Shift-GCN is inspired by shift CNNs [4, 32, 37], which use a lightweight shift operation as an alternative of 2D convolution and can adjust receptive fields by simply changing shift distances. The proposed Shift-GCN consists of *spatial shift graph convolution* and *temporal shift graph convolution*.

For spatial skeleton graph, instead of using three GCNs with different adjacent matrices to obtain enough receptive field [9, 12, 20–22, 31], we propose a spatial shift graph operation to shift information from neighbor nodes to the current

[*] Corresponding author.

[1] GFLOPs: Giga FLoating-number OPerations

convolution node. By interleaving the spatial shift graph operations with point-wise convolutions, information is mixed across spatial dimension and channel dimension. Specifically, we propose two kinds of spatial shift graph operation: *local shift graph operation* and *non-local shift graph operation*. For local shift graph operation, the receptive field is specified with the body physical structure. In this case, different nodes have a different number of neighbors, so the local shift graph operation is designed for each node respectively. However, local shift graph operation has two shortcomings: (1) The receptive field is heuristically pre-defined and localized, which is not suitable for modeling the diverse relations between skeletons. (2) Some information is abandoned directly due to the different shift operation for different nodes. To solve both shortcomings, we propose a non-local shift graph operation, which makes the receptive field of each node cover the full skeleton graph and learns the relations between the joints adaptively. Extensive ablation studies demonstrate that our non-local shift graph convolution outperforms the regular spatial graph convolution, even if the adjacent matrices of regular spatial graph convolution are learnable [20, 21].

For temporal skeleton graph, the graph is constructed by connecting consecutive frames on the temporal dimension. Instead of using a regular 1D temporal convolution [9, 12, 20, 21, 34], we propose two kinds of temporal shift graph operations: *naive temporal shift graph operation* and *adaptive temporal shift graph operation*. The receptive field of naive temporal shift graph operation is set manually, which is not optimal for temporal modeling: (1) Different layers may need diverse temporal receptive fields [11, 33, 38]. (2) Different datasets may need different temporal receptive fields [11]. These two problems also exist in regular 1D temporal convolution, whose kernel size is set manually. Our adaptive temporal shift graph operation addresses both problems by adjusting the receptive field adaptively. Extensive ablation studies demonstrate that our adaptive temporal shift graph convolution outperforms the regular temporal convolution with high efficiency.

To verify the superiority of our proposed model, namely, the spatiotemporal shift graph convolutional network (Shift-GCN), extensive experiments are performed on three datasets: NTU RGB+D [19], NTU-120 RGB+D [15] and Northwestern-UCLA [29]. We notably exceed the state-of-the-art methods on all three datasets with more than 10× less computational cost. The GFLOPs *v.s.* accuracy diagram of NTU RGB+D is shown in Fig.1.

Our contributions can be summarized as follows: (1) We propose two kinds of spatial shift graph operations for spatial skeleton graph modeling. Our non-local spatial shift graph operation is computationally efficient and achieves strong performance. (2) We propose two kinds of temporal shift graph operations for temporal skeleton graph modeling. Our adaptive temporal shift graph operation can adjust the receptive field adaptively and outperforms regular temporal model with much less computational complexity. (3) On three datasets for skeleton-based action recognition, the proposed Shift-GCN exceeds the state-of-the-art methods with more than 10× less computational cost. Code will be available at https://github.com/kchengiva/Shift-GCN.

2. Preliminaries

In this section, we provide a brief overview of the previous GCN-based skeleton action recognition models and the shift module in CNNs.

2.1. GCN-based skeleton action recognition

Graph convolutional networks (GCNs) have been successfully adopted to model skeleton data [9, 12, 20–22, 31, 34]. In these methods, the skeleton data is represented as a spatiotemporal graph $G = (V, E)$ with N joints and T frames. The skeleton coordinates of a human action can be represented as $\mathbf{X} \in \mathbb{R}^{N \times T \times d}$, where d is the dimension of joint coordinates. GCN-based models contains two parts: *spatial graph convolution* and *temporal graph convolution*.

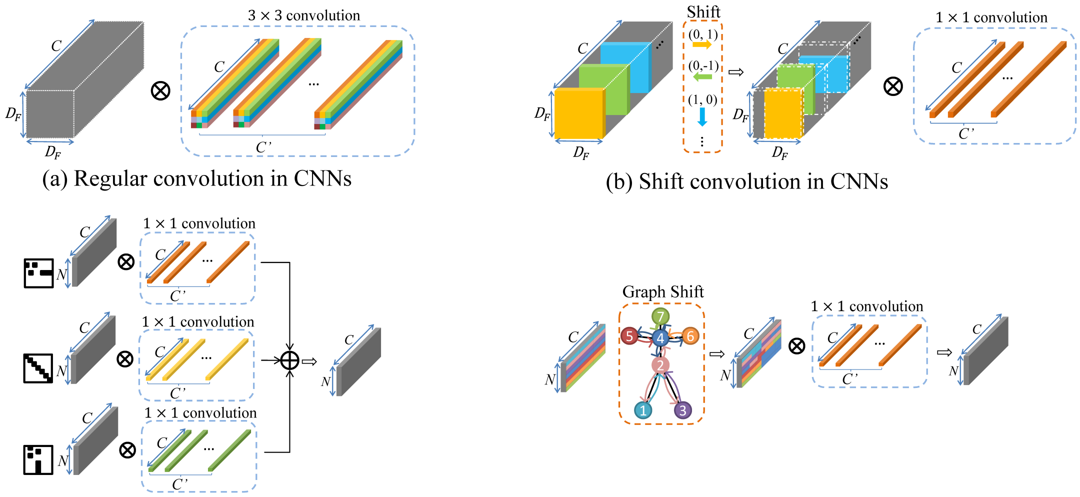
For spatial graph convolution, the neighbor set of joints is defined as an adjacent matrix $\mathbf{A} \in \{0, 1\}^{N \times N}$. To specify the spatial location of graph convolution, the adjacent matrix is typically partitioned into 3 partitions: 1) the centripetal group, which contains neighboring nodes that are closer to the skeleton center; 2) the node itself; 3) otherwise the centrifugal group. For a single frame, let $\mathbf{F} \in \mathbb{R}^{N \times C}$ and $\mathbf{F}' \in \mathbb{R}^{N \times C'}$ denote the input and output feature respectively, where C and C' are the input and output feature dimension. The graph convolution is computed as:

$$\mathbf{F}' = \sum_{p \in \mathcal{P}} \bar{\mathbf{A}}_p \mathbf{F} \mathbf{W}_p, \tag{1}$$

where $\mathcal{P}$ ={root, centripetal, centrifugal} denotes the spatial partitions, $\bar{\mathbf{A}}_p = \mathbf{\Lambda}_p^{-\frac{1}{2}} \mathbf{A}_p \mathbf{\Lambda}_p^{-\frac{1}{2}} \in \mathbb{R}^{N \times N}$ is the normalized adjacent matrix and $\mathbf{\Lambda}_p^{ii} = \sum_j (\mathbf{A}_p^{ij}) + \alpha$. α is set to 0.001 to avoid empty rows. $\mathbf{W}_p \in \mathbb{R}^{1 \times 1 \times C \times C'}$ is the weight of the 1×1 convolution for each partition group.

For temporal dimension, since the temporal graph is constructed by connecting consecutive frames, most GCN-based models [9, 20, 21, 31, 34] use regular 1D convolution on the temporal dimension as the temporal graph convolution. The kernel size is denoted as k_t, typically set to 9.

However, there are two disadvantages to these GCN-based models: (1) The computational cost is too heavy. For example, ST-GCN [34] costs 16.2 GFLOPs for one action sample, including 4.0 GFLOPs on spatial graph convolution and 12.2 GFLOPs on temporal graph convolution. Some recent variants of ST-GCN are even heavy to ∼100 GFLOPs



(a) Regular convolution in CNNs

(b) Shift convolution in CNNs

(c) Regular convolution in Spatial GCNs

(d) Shift convolution in Spatial GCNs

Figure 2. The diagram of regular convolution (a), shift convolution in CNNs (b), and the regular convolution in spatial GCNs (c). Our spatial shift graph convolution is illustrated as (d).

[20]. (2) The receptive field of both spatial graph and temporal graph are pre-defined. Although some works [20, 21] make the adjacent matrix learnable, our experiments show that its expressiveness is still limited by the regular GCN structure.

2.2. Shift CNNs

Let $\mathbf{F} \in \mathbb{R}^{D_F \times D_F \times C}$ denote the input feature, where D_F is the feature map size and C is the channel size. As shown in Fig.2 (a), regular convolution kernel is a tensor $\mathbf{K} \in \mathbb{R}^{D_K \times D_K \times C \times C'}$, where D_K is the kernel size. The FLOPs of regular convolution is $D_K^2 \times D_F^2 \times C \times C'$.

Shift convolution [32] is an efficient alternative to regular convolution in CNNs. As shown in Fig.2 (b), shift convolution is composed of two operations: (1) shift different channels in different directions; (2) apply a point-wise convolution to exchange information across channels. The FLOPs of shift convolution is $D_F^2 \times C \times C'$.

Another advantage of shift convolution is the flexibility of receptive field. The shift convolution can enlarge its receptive field by simply increasing the distance of shift, instead of using larger convolution kernels and increasing computation cost. Let shift value of each channel be denoted as a series of vectors $S_i, i = 1, 2, \cdots, C$, where $S_i = (x_i, y_i)$ denotes the 2D shift vector. The receptive field of shift convolution can be represented as a union set of each shift vectors in the opposite direction:

$$R = \{-S_1\} \cup \{-S_2\} \cup \cdots \cup \{-S_C\} \tag{2}$$

For example, if $x_i \in \{-1, 0, 1\}, y_i \in \{-1, 0, 1\}$, the receptive field is enlarged to 3×3.

3. Shift graph convolutional network

With the above discussion, it motivates us to introduce the lightweight shift operation to the heavy GCN-based action recognition models. In this section, we propose shift graph convolutional network, which contains spatial shift graph convolution and temporal shift graph convolution.

3.1. Spatial shift graph convolution

Introducing the shift operation from CNNs to GCNs is challenging, because graph features are not well-ordered like image feature maps. In this subsection, we first discuss the analogy from CNNs to spatial GCNs. Based on these analyses, we propose the spatial shift graph convolution for spatial skeleton graphs.

The analogy from CNNs to GCNs

A regular convolution kernel in CNNs can be regarded as a fusion of several point-wise convolution kernels, where each kernel operates on a specified location, as shown in Fig.2 (a) in different colors. For example, a 3×3 convolution kernel is a fusion of 9 point-wise convolution kernels, where each point-wise convolution kernel operates on "top-left", "top" , "top-right", ... , "bottom-right" respectively.

Similarly, a regular convolution kernel in spatial GCNs is a fusion of 3 point-wise convolution kernels, and each kernel operates on a specified spatial partition, as shown in Fig.2 (c) in different colors. As introduced in Sec.2.1, the spatial partitions are specified by 3 different adjacent matrices, which denote "centripetal", "root", "centrifugal" respectively.

A shift convolution in CNNs contains a shift operation and a point-wise convolution kernel, where the receptive field is specified by the shift operation, as shown in Fig.2 (b).

Therefore, a shift graph convolution should contain a shift graph operation and a point-wise convolution, as shown in Fig.2 (d). The main idea of the shift graph operation is shifting the features of the neighbor nodes to the current convolution node. Concretely, we propose two kinds of shift graph convolution: local shift graph convolution and non-local shift graph convolution.

Local shift graph convolution

For local shift graph convolution, the receptive field is specified with the physical structure of the human body, which is pre-defined by skeleton datasets. In this setting, the shift graph operation is conducted between neighbor nodes of the body physical graph.

Because the connection between body joints are not well-ordered like CNN features, different nodes have different number of neighbors. Let v denote a node and $B_v = \{B_v^1, B_v^2, \cdots, B_v^n\}$ denote the set of its neighbor nodes, where n denotes the number of neighbor nodes of v. We equally divide the channels of node v into $n+1$ partitions. We let the 1^{st} partition retain the feature of v. The other n partitions are shifted from the $B_v^1, B_v^2, \cdots, B_v^n$ respectively. Let $\mathbf{F} \in \mathbb{R}^{N \times C}$ denote the feature for a single frame and $\widetilde{\mathbf{F}} \in \mathbb{R}^{N \times C}$ denote the corresponding shifted feature. We operate the shift operation on each node of $\mathbf{F}$.

$$\widetilde{\mathbf{F}}_v = \mathbf{F}_{(v,:c)} \| \mathbf{F}_{(B_v^1,c:2c)} \| \mathbf{F}_{(B_v^2,2c:3c)} \| \cdots \| \mathbf{F}_{(B_v^n,nc:)} \quad (3)$$

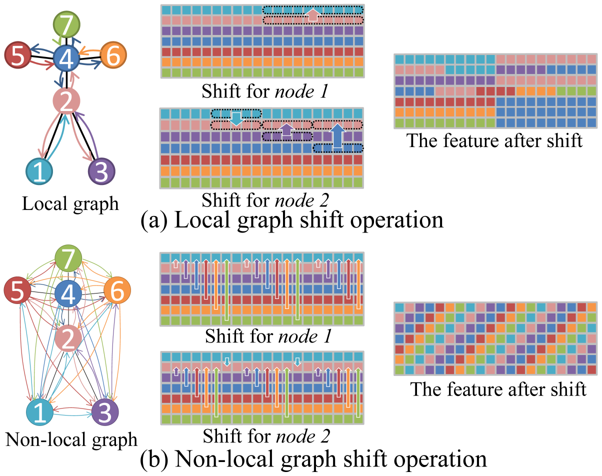
where $c = \lfloor \frac{C}{n+1} \rfloor$, the indexes of $\mathbf{F}$ are in Python notation, and $\|$ represents channel-wise concatenation.

To illustrate the intuition of local shift graph operation, we use a tiny graph feature of 7 nodes and 20 channels as an instance, shown in Fig.3 (a). We use *node 1* and *node 2* as two examples. For *node 1*, it only has one neighbor node, so its channels are divided into two partitions. The first partition retains the feature of *node 1*, while the second partition is shifted from *node 2*. For another example, *node 2* has three neighbor nodes, so its channels are divided into four partitions. The first partition retains the feature of *node 2*, while the other three partitions are shifted from *node 1, node 3, node 4*.

The feature after shift operation is illustrated in Fig.3 (a). In the shifted feature, every node obtains the information from its receptive field. By combining the local shift graph operation with a point-wise convolution, we get local shift graph convolution.

Non-local shift graph convolution

Local shift graph convolution has two shortcomings: (1) Some information is not utilized. For the example in Fig.3 (a), the last quarter of channels in *node 3* are abandoned directly during the shift operation. This is because different nodes have different numbers of neighbors. (2) Recent researches show that only considering local connections is not optimal for skeleton action recognition [12, 20, 21, 31]. For example, the relationship between two hands is important for recognizing actions such as "clapping" and "reading", but the two hands are far away from each other in the body structure.



(a) Local graph shift operation

(b) Non-local graph shift operation

Figure 3. The process of spatial shift graph operation.

We propose a simple solution to solve both shortcomings: making the receptive field of each node covers the full skeleton graph. We call it non-local shift graph operation.

Non-local shift graph operation is illustrated in Fig.3 (b). Given a spatial skeleton feature map $\mathbf{F} \in \mathbb{R}^{N \times C}$, the shift distance of i^{th} channel is i **mod** N. The shifted-out channels are used to fill the corresponding empty spaces. The shift operation of *node 1* and *node 2* are shown as examples. After the non-local shift, the feature looks like a spiral, which makes every node obtains the information from all other nodes, as shown in Fig.3 (b). By combining the non-local shift graph operation with a point-wise convolution, we get non-local shift graph convolution.

In the non-local shift graph convolution, the connection strength between different nodes is the same. But the importance of human skeletons is different. Hence, we introduce an adaptive non-local shift mechanism. We compute element-wise product between the shifted feature and a learnable mask:

$$\widetilde{\mathbf{F}}_M = \widetilde{\mathbf{F}} \circ Mask = \widetilde{\mathbf{F}} \circ (tanh(\mathbf{M}) + 1) \quad (4)$$

The FLOPs of regular spatial graph convolution is $3 \times (NCC' + N^2C')$. The FLOPs of shift spatial graph convolution is about NCC', which is more than three times lighter. Compared to regular graph convolution which only uses three adjacent matrices to model skeleton relations, our non-local shift operation can model various relations across different skeletons in different channels. Experiments in Sec.4.2.1 show that our non-local shift GCN achieves better performance than regular GCNs, even if the adjacent matrices in regular GCNs are set to be learnable [21].

3.2. Temporal shift graph convolution

After formulating a lightweight spatial shift graph convolution for modeling each skeleton frame, we now design a lightweight temporal shift graph convolution to model the skeleton sequence.

Naive temporal shift graph convolution

The temporal aspect of the graph is constructed by connecting consecutive frames on the temporal dimension. Therefore, the shift operation in CNNs can be extended to the temporal domain directly [13]. We equally divide the channels into $2u + 1$ partitions, and each partition has a temporal shift distance of $-u, -u+1, \cdots, 0, \cdots, u-1, u$ respectively. The shifted-out channels are truncated, and the empty channels are padded with zeros. After the shift operation, each frame obtains information from its neighbor frames. By combining this temporal shift operation with a temporal point-wise convolution, we get naive temporal shift graph convolution.

Typically, the kernel size of regular temporal convolution in GCN-based action recognition is 9 [9, 20, 21, 34]. Compared to regular temporal convolution, naive temporal shift graph convolution costs 9× less computational cost.

Adaptive temporal shift graph convolution

Although naive temporal shift graph convolution is lightweight, the setting of its hyper parameter u is manual. This causes two drawbacks: (1) Recent researches [11, 33, 38] show that different layers need diverse temporal receptive fields in video classification tasks. The exhaustive search over all possible combinations of u is intractable. (2) Different datasets may need different temporal receptive fields [11], which limits the generalization ability of naive temporal shift graph convolution. These two drawbacks also exist in regular temporal convolutions, whose kernel size is set manually.

We propose an adaptive temporal shift graph convolution to solve both drawbacks. Given a skeleton sequence feature $\mathbf{F} \in \mathbb{R}^{N\times T\times C}$, every channel has a learnable temporal shift parameter $S_i,\ i = 1, 2, \cdots, C$. We relax the temporal shift parameter from integer constraint to real numbers. The non-integer shift can be computed by linear interpolation:

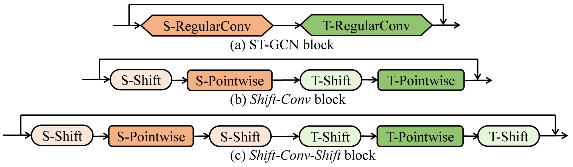
$$\widetilde{\mathbf{F}}_{(v,t,i)} = (1-\lambda)\cdot \mathbf{F}_{(v,\lfloor t+S_i\rfloor,i)} + \lambda\cdot \mathbf{F}_{(v,\lfloor t+S_i\rfloor+1,i)} \quad (5)$$

where $\lambda = S_i - \lfloor S_i \rfloor$. This operation is differentiable and can be trained through backpropagation. By combining this operation with a point-wise convolution, we get adaptive temporal shift convolution. The adaptive temporal shift operation is lightweight, with C extra parameter and $2NCT$ extra FLOPs. Compared with point-wise convolution, this computational cost is ignorable. The effectiveness and efficiency of adaptive temporal shift graph convolution are demonstrated in Sec.4.2.2.

3.3. Spatiotemporal shift GCN

To have an head-to-head comparison with the state-of-the-art methods [9, 12, 20, 21, 31, 34], we use the same backbone (ST-GCN [34]) to construct our spatiotemporal shift GCN. The ST-GCN backbone is composed of one input block and 9 residual blocks, where each block contains a regular spatial convolution and a regular temporal convolution. We replace the regular spatial convolution with our spatial shift operation and a spatial point-wise convolution. We replace the regular temporal convolution with our temporal shift operation and a temporal point-wise convolution.

There are two modes of combining the shift operation with point-wise convolution: *Shift-Conv* and *Shift-Conv-Shift*, as shown in Fig.4. The *Shift-Conv-Shift* mode has a larger receptive field and typically achieves better performance. We verify this phenomenon in the ablation study.



(a) ST-GCN block

(b) *Shift-Conv* block

(c) *Shift-Conv-Shift* block

Figure 4. Two modes of combining the shift operation and point-wise convolution.

4. Experiments

In this section, we first perform exhaustive ablation studies to verify the effectiveness and efficiency of our proposed spatial shift graph operation and temporal shift graph operation. Then, we compare our spatiotemporal shift GCN with the other state-of-the-art approaches on three datasets.

4.1. Datasets and Experiment Settings.

NTU RGB+D. NTU RGB+D [19], containing 56,880 skeleton action sequences, is the most widely used dataset for evaluating skeleton-based action recognition models. The action samples are performed by 40 volunteers and categorized into 60 classes. Each sample containing an action and is guaranteed to have at most 2 subjects, which is captured by three Microsoft Kinect v2 cameras from different views concurrently. The author of this dataset recommends two benchmarks: (1) cross-subject (X-sub) benchmark: training data comes from 20 subjects, and testing data comes from the other 20 subjects. (2) cross-view (X-view) benchmark: training data comes from the camera views 2 and 3, and the testing data comes from the camera view 1.

NTU-120 RGB+D. NTU-120 RGB+D [15] is currently the largest dataset with 3D joints annotations for human action recognition. The dataset contains 114,480 action samples in 120 action classes. Samples are captured by 106 volunteers with three cameras views. This dataset contains 32 setups, and each setup denotes a specific location and background. The author of this dataset recommends two benchmarks: (1) cross-subject (X-sub) benchmark: the 106 subjects are split into training and testing groups. Each group contains 53 subjects. (2) cross-setup (X-setup) benchmark: training data comes from samples with even setup IDs, and testing data comes from samples with odd setup IDs.

Northwestern-UCLA. Northwestern-UCLA dataset [29] is captured by three Kinect cameras. It contains 1494 video clips covering 10 categories. Each action is performed by 10 actors. We adopt the same evaluation protocol in [29]: we use the samples from the first two cameras as training data and the samples from the other camera as testing data.

Experiment Settings. We use SGD with Nesterov momentum (0.9) to train the model for 140 epochs. Learning rate is set to 0.1 and divided by 10 at epoch 60, 80 and 100. For adaptive temporal shift operation, the shift parameters are initialized with uniform distribution between -1 and 1. For NTU RGB+D and NTU-120 RGB+D, the batch size is 64, and we adopt the data pre-processing in [21]. For Northwestern-UCLA, the batch size is 16, and we adopt the data pre-processing in [22]. All experiments in the ablation study use the above setting, including both our proposed method and regular GCN method.

4.2. Ablation Study

4.2.1 Spatial shift graph convolution

In this subsection, we first show that spatial shift graph operation can significantly improve the performance of a point-wise convolution baseline. Then we demonstrate that spatial shift graph convolution outperforms regular spatial GCNs with more than 3× less computational cost.

Improving spatial point-wise baseline. To verify that spatial shift graph operation can effectively enlarge the receptive field, we build a lightweight spatial point-wise baseline by replacing the regular spatial convolution in ST-GCN with a simple point-wise convolution. The only difference between our spatial shift GCN and this point-wise baseline is inserting the spatial shift operation. As shown in Table 1, with our shift graph operation, the spatial point-wise baseline can be significantly improved. Specifically, our non-local shift operation can improve the baseline at 3.6% on NTU RGB+D X-view task.

Model	Shift mode	Top 1
Spatial point-wise	-	90.9
Local shift	Shift+Conv	93.5
	Shift+Conv+Shift	93.9
Non-local shift	Shift+Conv	94.0
	Shift+Conv+Shift	94.2
	Shift+Mask+Conv+Shift	**94.5**

Table 1. Comparisons between the spatial point-wise convolution and our spatial shift graph convolution.

The variants of spatial shift graph convolution. As shown in Table 1, non-local shift graph operation is more effective than local shift graph operation. This phenomenon indicates that non-local receptive field is important for skeleton-based action recognition. For both local shift and non-local shift model, the *Shift-Conv-Shift* mode is better than the *Shift-Conv* mode. This is because the *Shift-Conv-Shift* mode has a larger receptive field. The performance is further improved by introducing a learnable mask on the shifted feature.

Comparisons to regular spatial GCNs. In Table 2, we compare our spatial shift GCN with three regular spatial GCNs on both effectiveness and efficiency: a) ST-GCN [34], where the adjacent matrices are fixed as a pre-defined human graph, b) Adaptive GCN [21], where the adjacent matrices are learnable, c) Adaptive-Nonlocal GCN [21], where the adjacent matrices are predicted by a non-local attention module. All models in Table 2 use the same temporal model, so that we can focus on evaluating the effectiveness and efficiency of different spatial models.

Model	Spatial FLOPs (G)	Top 1
ST-GCN [34]	4.0	93.4
Adaptive GCN [21]	4.0	93.9
Adaptive-NL GCN [21]	5.7	94.2
ST-GCN (one A)	1.3	92.1
Adaptive GCN (one A)	1.3	92.9
Local shift GCN	**1.1**	93.9
Non-local shift GCN	**1.1**	**94.5**

Table 2. Comparisons between regular spatial GCNs and our spatial shift graph GCN.

As shown in Table 2, our local shift GCN outperforms ST-GCN [34]; our non-local shift GCN outperforms all three regular GCNs. More importantly, our shift graph convolution is much more efficient than regular GCNs. Compared with ST-GCN [34] and adaptive GCN [21], our shift GCN is 3.6× lighter. Compared with adaptive-nonlocal GCN [21] which introduces a non-local attention module, our shift GCN is 5.2× lighter. In Table 2, we also build a lightweight version of regular GCN using only one adjacent matrix, suffixed with "one A", which gets obvious inferior performance. This phenomenon indicates that regular spatial GCNs need multiple adjacent matrices to model the diverse relations between skeletons, leading to high computational cost. Our non-local shift convolution can model the diverse relations across different skeletons and different channels with a lightweight point-wise convolution, which is more effective and efficient.

4.2.2 Temporal shift graph convolution

In this subsection, we fix the spatial model as the regular spatial convolution of ST-GCN [34], and evaluate the effectiveness and efficiency of different temporal models.

Model	Shift mode		Top 1
Temporal point-wise	-		79.2
Regular conv(k_t=3)	-		93.4
Regular conv(k_t=5)	-		93.6
Regular conv(k_t=7)	-		**93.7**
Regular conv(k_t=9)	-		93.4
Regular conv(k_t=11)	-		93.4
Naive shift	Shift+Conv	u=1	93.2
		u=2	93.2
		u=3	93.4
		u=4	93.4
	Shift+Conv+Shift	u=1	93.0
		u=2	93.0
		u=3	**93.6**
		u=4	93.3
Adaptive shift	Shift+Conv		94.0
	Shift+Conv+Shift		**94.2**

Table 3. Comparisons between temporal point-wise convolution, regular temporal convolution, naive temporal shift convolution and adaptive temporal shift convolution. The computation cost of temporal shift convolution is $k_t\times$ less than regular temporal convolution, where k_t is the kernel size of regular temporal convolution.

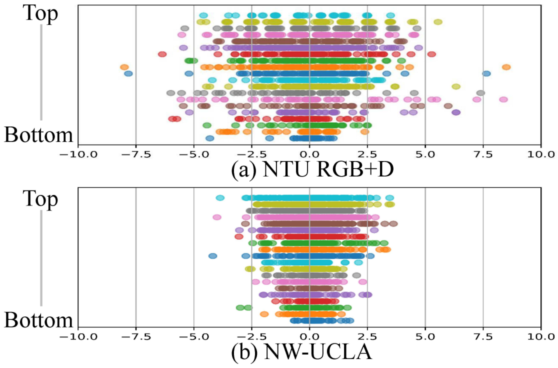
Improving temporal point-wise baseline. By replacing the regular temporal convolution of ST-GCN [34] with a temporal point-wise convolution, we build a temporal point-wise baseline. The only difference between our temporal shift graph convolution and this baseline is inserting our temporal shift operation. As shown in Table 3, with our temporal shift graph operation, the point-wise baseline can be significantly improved. Specifically, our adaptive temporal shift operation can improve the baseline at 15.0% on NTU RGB+D X-view task.

The superiority of adaptive temporal shift. We compare three different temporal models: a) the regular temporal convolution; b) naive temporal shift operation; c) adaptive temporal shift operation. The receptive field of regular temporal convolution and naive temporal shift operation are set manually, while our adaptive temporal shift operation can adjust the receptive field adaptively. In Table 3, we conduct an exhaustive search for the best receptive field of regular temporal convolution and naive temporal shift operation. Our adaptive temporal shift operation does not require the troublesome exhaustive search, and outperforms the best results of the other two methods.

Visualizations of adaptive temporal shift.

We visualize the adaptive temporal shift parameters trained on NTU RGB+D and Northwestern-UCLA respectively. There are 10 temporal blocks in ST-GCN [34], and each block is replaced with our *Shift-Conv-Shift* module, so there are 20 adaptive temporal shift operations in a model. We visualize the learned shift parameters from the bottom layer (input layer) to the top layer (output layer). As shown in Fig.5, the shift parameters of top layers tend to be larger than that of bottom layers, which means the top layers need larger temporal receptive fields while the bottom layers tend to learn spatial relations. Note that in the video classification field, an exhaustive search is conducted in [33] to find which layer should use temporal convolution, and their conclusion is applying temporal convolutions on top layers is more effective. Our adaptive temporal shift operation learns the appropriate temporal receptive field for every layer without heuristic design or manually exhaustive search.

Another superiority of adaptive temporal shift operation is improving the model generalization on different datasets. As shown in Fig.5, the shift parameters trained on NW-UCLA dataset tend to be smaller than that of NTU RGB+D dataset. This is reasonable because the average frame number of action samples in NTU RGB+D (71.4 frames) is about twice larger than that of NW-UCLA (39.4 frames).



(a) NTU RGB+D

(b) NW-UCLA

Figure 5. Visualization of adaptive temporal shift.

4.2.3 Spatiotemporal shift GCN

Both spatial shift graph convolution and temporal shift graph convolution are more effective and efficient than regular graph convolution. We conduct spatiotemporal shift graph convolution and further boost performance and efficiency. As shown in Table 4, spatiotemporal shift GCN outperforms ST-GCN [34] at 1.7% with 6.5× less computation cost.

Spatial model	Temporal model	FLOPs (G)	Top 1
Regular S-GCN	Regular T-GCN	16.2	93.4
Shift S-GCN	Regular T-GCN	13.3	94.5
Regular S-GCN	Shift T-GCN	5.4	94.2
Shift S-GCN	Shift T-GCN	**2.5**	**95.1**

Table 4. The effectiveness and efficiency of spatiotemporal shift graph convolution. The accuracy is on NTU RGB+D X-view task.

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

Many state-of-the-art methods utilize multi-stream fusion strategies. To conduct a fair comparison, we adopt the same multi-stream fusion strategy as [20], which utilizes 4 streams. The first stream uses the original skeleton coordinates as input, called "joint stream"; the second stream uses the differential of spatial coordinates as input, called "bone stream"; the third and fourth stream use the differential on temporal dimension as input, called "joint motion stream" and "bone motion stream" respectively. The *softmax* scores of multiple streams are added to obtain the fused score.

There are three settings of our spatiotemporal shift GCN (Shift-GCN): *1-stream*, which only uses the joint stream; *2-stream*, which uses both joint stream and bone stream; *4-stream*, which uses all 4 streams. To verify the superiority and generality of our approach, the shift GCN is compared with state-of-the-art methods on three datasets: NTU RGB+D dataset [19], Northwestern-UCLA dataset [29], and the recent proposed NTU-120 RGB+D dataset [15], shown in Table 5, Table 6, and Table 7 respectively. We show the computational complexity [2] of the methods that achieve higher than 85% on NTU RGB+D X-sub task.

On NTU RGB+D, 1s-Shift-GCN achieves higher accuracy than 2s-AS-GCN [12] with 10.8× less computational cost; 2s-Shift-GCN is comparable with the current state-of-the-art method 4s-Directed-GNN [20] with 25.4× less computational cost; 4s-Shift-GCN obviously exceeds all state-of-the-art methods with 12.7× less computation than 4s-Directed-GNN [20]. On Northwestern-UCLA dataset, our 2s-Shift-GCN outperforms the current state-of-the-art 2s-AGC-LSTM [22] at 0.9% with 33.0× less computation complexity. On NTU-120 RGB+D dataset, we obviously exceed all previously reported performance.

Methods	X-view	X-sub	FLOPs (G)
Lie Group [26]	52.8	50.1	-
HBRNN [2]	64.0	59.1	-
Deep-LSTM [19]	67.3	60.7	-
VA-LSTM [35]	87.7	79.2	-
TCN [7]	83.1	74.3	-
Synthesized CNN [18]	87.2	80.0	-
3scale ResNet 152 [10]	90.9	84.6	-
ST-GCN [34]	88.3	81.5	-
Motif+VTDB [31]	90.2	84.2	-
2s AS-GCN [12]	94.2	86.8	27.0
2s Adaptive GCN [21]	95.1	88.5	35.8
2s AGC-LSTM [22]	95.0	89.2	54.4
4s Directed-GNN [20]	96.1	89.9	126.8
1s Shift-GCN (ours)	95.1	87.8	**2.5**
2s Shift-GCN (ours)	96.0	89.7	5.0
4s Shift-GCN (ours)	**96.5**	**90.7**	10.0

Table 5. Comparisions of the Top-1 accuracy (%) with the state-of-the-art methods on the NTU RGB+D dataset.

Methods	Top-1	FLOPs (G)
Lie Group [26]	74.2	-
Actionlet ensemble [28]	76.0	-
HBRNN-L [2]	78.5	-
Ensemble TS-LSTM [8]	89.2	-
2s AGC-LSTM [22]	93.3	10.9
1s Shift-GCN (ours)	92.5	**0.2**
2s Shift-GCN (ours)	94.2	0.3
4s Shift-GCN (ours)	**94.6**	0.7

Table 6. Comparisions of the accuracy (%) with the state-of-the-art methods on the Northwesten-UCLA dataset.

Methods	X-sub	X-setup	FLOPs (G)
Part-Aware LSTM [19]	25.5	26.3	-
ST-LSTM [16]	55.7	57.9	-
Multi CNN + RotClips [6]	62.2	61.8	-
SkeMotion [17]	67.7	66.9	-
TSRJI [1]	67.9	62.8	-
1s Shift-GCN (ours)	80.9	83.2	**2.5**
2s Shift-GCN (ours)	85.3	86.6	5.0
4s Shift-GCN (ours)	**85.9**	**87.6**	10.0

Table 7. Comparisions of the Top-1 accuracy (%) with the state-of-the-art methods on the NTU-120 RGB+D dataset.

5. Conclusion

In this work, we propose a novel shift graph convolutional network (Shift-GCN) for skeleton-based action recognition, which is composed of spatial shift graph convolution and temporal shift graph convolution. Our non-local spatial shift graph convolution obviously outperforms regular graph convolution with much less computation cost. Our adaptive temporal shift graph convolution can adjust the receptive field adaptively and enjoy high efficiency. On three datasets for skeleton-based action recognition, the proposed Shift-GCN notably exceeds the current state-of-the-art methods with more than 10× less computation cost.

Acknowledgement: This work was supported by the State Grid Corporation Science and Technology Project(No.5200-201916261A-0-0-00).

[2] The computational complexity was not explicitly discussed in some papers; we estimate them based on their description. Details are provided in supplement material.

References

[1] Carlos Caetano, François Brémond, and William Robson Schwartz. Skeleton image representation for 3d action recognition based on tree structure and reference joints. *arXiv preprint arXiv:1909.05704*, 2019.

[2] Yong Du, Wei Wang, and Liang 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.

[3] Basura Fernando, Efstratios Gavves, Jose M Oramas, Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5378–5387, 2015.

[4] Yunho Jeon and Junmo Kim. Constructing fast network through deconstruction of convolution. In *Advances in Neural Information Processing Systems*, pages 5951–5961, 2018.

[5] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Sohel, and Farid Boussaid. A new representation of skeleton sequences for 3d action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3288–3297, 2017.

[6] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Sohel, and Farid Boussaid. Learning clip representations for skeleton-based 3d action recognition. *IEEE Transactions on Image Processing*, 27(6):2842–2855, 2018.

[7] Tae Soo Kim and Austin Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In *2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW)*, pages 1623–1631. IEEE, 2017.

[8] Inwoong Lee, Doyoung Kim, Seoungyoon Kang, and Sanghoon Lee. Ensemble deep learning for skeleton-based action recognition using temporal sliding lstm networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1012–1020, 2017.

[9] Bin Li, Xi Li, Zhongfei Zhang, and Fei Wu. Spatio-temporal graph routing for skeleton-based action recognition. 2019.

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

[11] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Collaborative spatiotemporal feature learning for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7872–7881, 2019.

[12] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[13] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7083–7093, 2019.

[14] Hong Liu, Juanhui Tu, and Mengyuan Liu. Two-stream 3d convolutional neural network for skeleton-based action recognition. *arXiv preprint arXiv:1705.08106*, 2017.

[15] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C. Kot. NTU RGB+D 120: A large-scale benchmark for 3d human activity understanding. *CoRR*, abs/1905.04757, 2019.

[16] 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.

[17] Jun Liu, Gang Wang, Ping Hu, Ling-Yu Duan, and Alex C Kot. Global context-aware attention lstm networks for 3d action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1647–1656, 2017.

[18] Mengyuan Liu, Hong Liu, and Chen Chen. Enhanced skeleton visualization for view invariant human action recognition. *Pattern Recognition*, 68:346–362, 2017.

[19] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1010–1019, 2016.

[20] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with directed graph neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7912–7921, 2019.

[21] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[22] Chenyang Si, Wentao Chen, Wei Wang, Liang Wang, and Tieniu Tan. An attention enhanced graph convolutional lstm network for skeleton-based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[23] Chenyang Si, Ya Jing, Wei Wang, Liang Wang, and Tieniu Tan. Skeleton-based action recognition with spatial reasoning and temporal stack learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 103–118, 2018.

[24] 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 *Thirty-first AAAI conference on artificial intelligence*, 2017.

[25] Yansong Tang, Yi Tian, Jiwen Lu, Peiyang Li, and Jie Zhou. Deep progressive reinforcement learning for skeleton-based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5323–5332, 2018.

[26] Vivek Veeriah, Naifan Zhuang, and Guo-Jun Qi. Differential recurrent neural networks for action recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 4041–4049, 2015.

[27] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3d skeletons as

points in a lie group. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 588–595, 2014.

[28] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Learning actionlet ensemble for 3d human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):914–927, 2013.

[29] Jiang Wang, Xiaohan Nie, Yin Xia, Ying Wu, and Song-Chun Zhu. Cross-view action modeling, learning and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2649–2656, 2014.

[30] Wei Wang, Jinjin Zhang, Chenyang Si, and Liang Wang. Pose-based two-stream relational networks for action recognition in videos. *arXiv preprint arXiv:1805.08484*, 2018.

[31] Yu-Hui Wen, Lin Gao, Hongbo Fu, Fang-Lue Zhang, and Shihong Xia. Graph cnns with motif and variable temporal block for skeleton-based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8989–8996, 2019.

[32] Bichen Wu, Alvin Wan, Xiangyu Yue, Peter Jin, Sicheng Zhao, Noah Golmant, Amir Gholaminejad, Joseph Gonzalez, and Kurt Keutzer. Shift: A zero flop, zero parameter alternative to spatial convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9127–9135, 2018.

[33] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.

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

[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. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2117–2126, 2017.

[36] Wu Zheng, Lin Li, Zhaoxiang Zhang, Yan Huang, and Liang Wang. Skeleton-based relational modeling for action recognition. *arXiv preprint arXiv:1805.02556*, 2018.

[37] Huasong Zhong, Xianggen Liu, Yihui He, Yuchun Ma, and Kris Kitani. Shift-based primitives for efficient convolutional neural networks. *arXiv preprint arXiv:1809.08458*, 2018.

[38] Mohammadreza Zolfaghari, Kamaljeet Singh, and Thomas Brox. Eco: Efficient convolutional network for online video understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 695–712, 2018.