Supplementary of 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

1. Details for computing FLOPs[1].

In the main paper, we show the computational complexity of ST-GCN [8], 2s-AS-GCN [1], 2s-Adaptive-GCN [5], 2s-AGC-LSTM [6], 4s-Directed-GNN [4], and our proposed Shift-GCN. Because the computational complexity was not explicitly discussed in some papers; we estimate them based on their description. Note that the number of body joints and sample frames are different in different dataset; therefore, we discuss the computational complexity on NTU RGB+D [3], NTU-120 RGB+D [2], and Northwestern-UCLA [7] in Section.1.1, Section.1.2, and Section.1.3 respectively.

1.1. NTU RGB+D dataset.

1.1.1 The FLOPs of ST-GCN [8].

ST-GCN [8] is composed of one input block and 9 residual blocks, as shown in Table 1. Each block contains a regular spatial convolution and a regular temporal convolution.

As introduced in our main paper, the regular spatial convolution in a block is a fusion of three graph convolutions, which operate on three different partitions respectively. Every graph convolution contains two matrix multiplications, whose computational complexity is $NCC' + N^2C'$ (For the blocks that $C = C'$, this formula can be simplified to $NC^2 + N^2C$). Every frame requires the same computation process. Therefore, the FLOPs of regular spatial convolution in a block is:

$$S_{FLOPs} = (3 \times (NCC' + N^2C')) \times T \tag{1}$$

The regular temporal convolution in a block is a 1D convolution on the temporal dimension whose kernel size is 9. Every body joint requires the same temporal convolution. Therefore, the FLOPs of a temporal convolution is:

$$T_{FLOPs} = (9 \times TCC') \times N \tag{2}$$

Besides, a fully-connected layer is used to get final scores for 60 classes, whose FLOPs is:

$$FC_{FLOPs} = C' \times classes = 256 \times 60 \tag{3}$$

ST-GCN [8] is composed of 10 spatial convolutions, 10 temporal convolutions, and one fully-connected layer. We compute the FLOPs of every part and get the total FLOPs: 8.1G.

Note that in the NTU RGB+D dataset, there are one or two people in each sample. For samples with only one person, the second person is padded with zeros. The skeleton graphs of these two people are computed respectively. Thus, the total FLOPs for one action sample is 2 × 8.1G = 16.2G, including 4.0G on spatial graph convolution and 12.2G on temporal graph convolution.

Stage	ST-GCN
Block0	$C = 3, C' = 64, T = 300, N = 25$
Block1	$C = 64, C' = 64, T = 300, N = 25$
Block2	$C = 64, C' = 64, T = 300, N = 25$
Block3	$C = 64, C' = 64, T = 300, N = 25$
Block4	$C = 64, C' = 128, T = 150, N = 25$
Block5	$C = 128, C' = 128, T = 150, N = 25$
Block6	$C = 128, C' = 128, T = 150, N = 25$
Block7	$C = 128, C' = 256, T = 75, N = 25$
Block8	$C = 256, C' = 256, T = 75, N = 25$
Block9	$C = 256, C' = 256, T = 75, N = 25$
	global average pooling, FC, softmax

Table 1. The structure of ST-GCN, where C denotes the number of input channels, C' denotes the number of output channels. T denotes the number of temporal frames, N denotes the number of body joints on NTU RGB+D dataset.

*Corresponding author.
[1]FLOPs: FLoating-number OPerations

1.1.2 The FLOPs of Shift-GCN (Ours).

As introduced in our main paper, we use ST-GCN [8] as the backbone structure. We replace the regular spatial convolution with our spatial shift graph convolution and replace the regular temporal convolution with our temporal shift graph convolution.

We propose two kinds of spatial shift graph convolutions. For a local spatial shift graph convolution layer, the FLOPs is $NCC'T$ (i.e., the FLOPs of a point-wise convolution). For a non-local spatial shift graph convolution layer, the FLOPs is $NCC'T + NCT$, where NCT is the FLOPs of the element-wise mask. Compared with the FLOPs of regular spatial graph convolution in Eq.1, both local and non-local spatial shift graph convolution are more than $3\times$ lighter.

We propose two kinds of temporal shift graph convolutions. For a naive temporal shift graph convolution layer, the FLOPs is $NC'C'T$ (i.e., the FLOPs of a point-wise convolution). For an adaptive temporal shift graph convolution layer, the FLOPs is $NC'C'T + 4NC'T$, where $4NC'T$ is the FLOPs of two adaptive temporal shift operations. Compared with the FLOPs of regular temporal graph convolution in Eq.2, both naive and adaptive temporal shift graph convolution are about $9\times$ lighter.

For samples with only one person, we adopt the same padding strategy as ST-GCN [8] to pad the second person with zeros. The FLOPs of Shift-GCN is 2.5G, including 1.1G on spatial shift graph convolution and 1.4G on temporal shift graph convolution. For 2-stream-Shift-GCN and 4-stream-Shift-GCN, the FLOPs are 5.0G and 10.0G respectively.

1.1.3 The FLOPs of 2s-AS-GCN [1].

2s-AS-GCN [1] contains 2 streams: an actional stream and a structural stream. Both streams use ST-GCN [8] as the backbone, where the temporal convolution kernel size is modified to 7. They introduce an extra decoder network, but the decoder network is not used in inference. Thus we only compute the FLOPs of its backbone: $2 \times$ 13.5G = 27.0G, where "13.5G" is the FLOPs of ST-GCN whose temporal kernel size is 7.

1.1.4 The FLOPs of 2s-Adaptive-GCN [5].

2s-Adaptive-GCN [5] contains 2 streams: a joint stream and a bone stream. Both streams use ST-GCN [8] as the backbone and introduce non-local adaptive spatial graph convolution to enhance the expressiveness. As described in their paper, the FLOPs of non-local adaptive spatial graph convolution is $3 \times (NCC_e + NCC' + N^2C_e + N^2C) \times T$, where C_e is the dimension of embedding space. With the non-local adaptive module, the total FLOPs of 2s-Adaptive-GCN is $2 \times$ 17.9G = 35.8G.

1.1.5 The FLOPs of 2s-AGC-LSTM [6].

The AGC-LSTM model contains three parts: the first linear embedding layer, a standard LSTM layer, and three AGC-LSTM layers.

The first linear layer encodes the 3-dim coordinates of joints into a 256-dim vector as position features. Its FLOPs is $3 \times 256 \times N \times T$, where $N = 25$ and $T = 100$.

After the first linear layer, they use a feature augmentation operation to get 512-dim vectors for every body joint in every frame. These vectors are sent to an LSTM layer whose hidden size is 512. The FLOPs of this LSTM layer is $(4 \times 2C^2) \times N \times T$, where $C = 512$, $N = 25$, $T = 100$, "$4\times$" means that there are 4 matrix multiplications in one LSTM.

After that, they introduce three AGC-LSTM layers. The FLOPs of each AGC-LSTM layer contains three sub-parts: 1) the FLOPs of LSTM, which is $(4 \times 2C^2) \times N \times T$; 2) the FLOPs of graph convolution, which is $3 \times (N^2C + NC^2) \times T \times 2$; 3) the FLOPs of their proposed attention module, which is at least $(4 \times C^2) \times T$. Before each AGC-LSTM layer, a temporal pooling layer is inserted to downsample the temporal dimension. The stride of the pooling layer is not explicitly mentioned in their paper, so we contact the author and confirm that the stride is 2. Therefore, in these three AGC-LSTM layers, T is 50, 25, 13 respectively.

Besides, a fully-connnected layer is used to get final scores for 60 classes, whose FLOPs is $C \times classes = 512 \times 60$.

After computing the above FLOPs, we need to multiply the result by 4, because there can be 2 people in one NTU RGB+D sample and 2s-AGC-LSTM uses 2 streams fusion strategy. The total FLOPs of 2s-AGC-LSTM is 54.4G.

1.1.6 The FLOPs of 4s-Directed-GNN [4].

4s-Directed-GNN [4] fuses 4 streams: "joint stream", "bone stream", "joint motion stream", and "bone motion stream". All 4 streams use ST-GCN [8] as backbone. They divide them into two groups. One group contains the "joint stream" and "bone stream". We call it "spatial group". Another group contains the "joint motion stream" and "bone motion stream". We call it "motion group".

In both groups, they introduce two directed graph modules into every ST-GCN block to exchange information. Each directed graph module contains a fully-connected layer, whose input channel is $3C$ and output channel is C'. Therefore, the FLOPs of one directed graph module is $3CC'NT$, where N is the number of body joints and T is the number of frames.

In their paper, they mention two ST-GCN baselines. One is called 2s-ST-GCN which is the standard two-stream ST-GCN; the other is called 1s-ST-GCN whose number of channels is twice the original number. But they do not explicitly mention which baseline to build their Directed-GNN. We contact the author and confirm that: for spatial convolution, they use two-stream ST-GCN and introduce directed graph modules to exchange information; however, for temporal convolution, they concatenate the two-stream features on the channel dimension and double the number of channels. Notice that the FLOPs is *quadratic* in term of the number of channels.

With these analyses, we can compute the FLOPs of Directed-GNN [4]. The FLOPs of both "spatial group" and "motion group" are 63.4G, so the total FLOPs of 4s-Directed-GNN is $2 \times 63.4\text{G} = 126.8\text{G}$.

1.2. NTU-120 RGB+D dataset.

For NTU-120 RGB+D dataset [2], the number of body joints and sample frames are the same with NTU RGB+D dataset. When computing FLOPs, the only difference is the last fully-connected layer, because the number of classes is 120 instead of 60. In this case, the FLOPs of Shift-GCN is still 2.5G, because the FLOPs of the last fully-connected layer is tiny.

1.3. Northwestern-UCLA dataset.

For Northwestern-UCLA dataset [7], the number of body joints is 20. Both 2s-AGC-LSTM [6] and our Shift-GCN samples 50 frames as input. In this case, the FLOPs of 2s-AGC-LSTM is 10.9G; the FLOPs of Shift-GCN is 0.17G. For 2-stream-Shift-GCN and 4-stream-Shift-GCN, the FLOPs are 0.33G and 0.66G respectively.

2. Detailed training settings.

We use SGD with Nesterov momentum (0.9) to train the model for 140 epochs. Initial learning rate is set to 0.1 and is divided by 10 at epoch 60, 80 and 100. For NTU RGB+D and NTU-120 RGB+D, the batch size is 64. The max number of frames in each sample is 300. For samples with less than 300 frames, we pad it to 300 frames by repeating data [8]. For Northwestern-UCLA, the batch size is 16. The number of frames is set to 50 [6]. The weight decay is 1e-4 except the spatial shift graph convolution, whose weight decay is set to 1e-3. This setting is helpful for regularizing spatial shift graph convolution which models diverse relations and tend to be overfitted. For spatiotemporal Shift-GCN, we find the variance of feature is larger, but the variance of the learnable mask is small. So we set the weight decay for learnable mask to 0. The same hyperparameter is used across different layers and different datasets.

For spatial shift graph operation, the learnable mask has NC degree of freedom, which is helpful for modeling diverse shift connections between different nodes. Inspired by this mechanism, we employ a batch norm layer with NC weights and biases with the spatial shift graph convolution. For adaptive temporal shift operation, the shift parameters are randomly initialized with uniform distribution between -1 and 1. To ensure stable learning, we clip the gradient of shift parameters with a bound 0.01. The same hyperparameter is used across different layers and different datasets.

3. Runtime speed.

As shown in the following table, Shift-GCN is not only theoretically efficient but also has notable practical speedup [2]. The graph shift layers only occupy 8% runtime of Shift-GCN, including non-local spatial shift and adaptive temporal shift. All these timing results are measured on 4 TITAN Xp with batch size = 64. As shown in Table 2, Shift-GCN has notable practical speedup than regular GCN methods, even in multi-stream cases [3].

Model Comparison	Theoretical	Practical
1s Shift-GCN *v.s.* 2s AS-GCN	10.8×	5.6×
1s Shift-GCN *v.s.* 2s Adaptive-GCN	14.3×	8.2×
2s Shift-GCN *v.s.* 2s Adaptive-GCN	7.2×	4.0×

Table 2. Theoretical speedup and practical speedup of Shift-GCN.

4. Sensitivity analysis for shift orders.

As mentioned in our main paper, the main idea of the shift graph operation is shifting the features of the neighbor nodes to the current convolution node. Thus, there is a question: if a node has more than one neighbor nodes, we should shift them in what order?

For the non-local shift graph operation and the temporal shift graph operation, the network structure is irrelevant to the shift order. For example, shifting the first channel or shifting the second channel is equivalent to a point-wise convolution.

However, for local shift graph operation, different shift order affects the utilization ratio of feature, because different nodes have different numbers of neighbors. In our experiment, if a node has more than one neighbor node, the shift order is decided by the ordinal number for simplicity, which is pre-defined by skeleton datasets. To evaluate the sensitivity for different shift order, we randomly shuffle the ordinal numbers of body joints and evaluate the performance of local spatial shift GCN for 5 times. As shown in Table 3, the performance does not change obviously in different shift orders.

[2] We compare with 2s-AS-GCN and 2s-Adaptive-GCN because their codes are publicly available.

[3] The runtime of multi-stream cases are measured by serial computing.

Shift order	Top 1
Following ordinal number	93.9
Random Shuffle	93.9
	94.0
	94.0
	93.9
	94.0

Table 3. Sensitivity analysis for different shift orders. The accuracy (%) is evaluated on NTU RGB+D X-view task.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.