Spatial Temporal Graph Attention Network for Skeleton-Based Action Recognition

Lianyu Hu\textsuperscript{1}, Shenglan Liu\textsuperscript{2}, and Wei Feng\textsuperscript{1}

\textsuperscript{1} Tianjin University
\textsuperscript{2} Dalian University of Technology

Abstract. It’s common for current methods in skeleton-based action recognition to mainly consider capturing long-term temporal dependencies as skeleton sequences are typically long (>128 frames), which forms a challenging problem for previous approaches. In such conditions, short-term dependencies are few formally considered, which are critical for classifying similar actions. Most current approaches are consisted of interleaving spatial-only modules and temporal-only modules, where direct information flow among joints in adjacent frames are hindered, thus inferior to capture short-term motion and distinguish similar action pairs. To handle this limitation, we propose a general framework, coined as STGAT, to model cross-spacetime information flow. It equips the spatial-only modules with spatial-temporal modeling for regional perception. While STGAT is theoretically effective for spatial-temporal modeling, we propose three simple modules to reduce local spatial-temporal feature redundancy and further release the potential of STGAT, which (1) narrow the scope of self-attention mechanism, (2) dynamically weight joints along temporal dimension, and (3) separate subtle motion from static features, respectively. As a robust feature extractor, STGAT generalizes better upon classifying similar actions than previous methods, witnessed by both qualitative and quantitative results. STGAT achieves state-of-the-art performance on three large-scale datasets: NTU RGB+D 60, NTU RGB+D 120, and Kinetics Skeleton 400. Code is released.

1 Introduction

As a special issue of action recognition, skeleton-based action recognition has attracted much attention recently due to the fast development of pose estimation algorithms \textsuperscript{29} and wearable motion capture equipment. It involves predicting an action label from a sequence of 2D or 3D skeleton representations. Compared to action recognition in videos \textsuperscript{40,39,3}, skeleton data is more compact in 2D \textsuperscript{11,11} or 3D \textsuperscript{23,27} format and suffers less from environmental factors (e.g. camera motion, light transformations and viewpoint changing), which makes it a more robust representation of human action sequences.

It’s usual for current methods to mainly tackle long-term temporal dependencies as skeleton sequences are rather long (>128 frames), which forms a serious obstacle for previous methods to model human dynamics through a whole video.
Fig. 1. (a) Factorized spatial-only and temporal-only modeling over input sequences. (b) Disentangled multi-scale feature aggregation in local spatial-temporal neighborhoods. Edges of the same color indicate equal weights in the predefined spatial-temporal graph. (c) Spatial-temporal feature aggregation with adaptively computed edges. Edges with different colors represent different degrees of connection strength.

However, in such conditions, critical short-term temporal dependencies are few formally considered. In contrast with long-term dependencies which recognize action sequences from an overall perspective, short-term dependencies distinguish similar sequences on small temporal scales with important details. For example, 'Eating' and 'Tooth brushing' have quite similar movements in both upper and lower body where hands are raised next to mouth for a long while, holding food or brushing teeth, which is quite similar in long sequences. To classify them relies on whether there exists subtle movement of the hand to brush horizontally, which is short-term and rather critical. Few methods have realized the importance of short-term motion and proposed dilated temporal convolutions to capture short-term temporal movements. We propose a general framework to perform spatial-temporal modeling, achieving superior performance than them under similar computational budgets as shown in Sec. 4.3.

Most existing methods employ interleaving spatial-only and temporal-only modules to extract spatial features and capture long-term temporal dependencies, respectively. A typical approach is to first use a graph module to extract spatial representations at each time step, and then deploy recurrent or temporal convolution layers to build temporal relationships (Fig 1(a)). Such design are efficient and effective for long-term modeling, but hinders direct information flow in local spatial-temporal neighborhoods. For example, messages have to pass through a spatial module and a temporal module to reach nodes in adjacent frames, which are inevitably weakened in multiple transitions. Especially, large temporal kernels adopted after spatial modules get more information mixed and may cause subtle motion impaired, where short-term temporal dependencies may be ineffectively captured by such factorized modeling.

In this work, we propose STGAT with spatial-temporal modeling to specially capture short-term dependencies. We retain the temporal-only modules for long-term modeling and give spatial-only modules more freedom to perform local spatial-temporal modeling. It builds local spatial-temporal graphs by connecting nodes in local spatial-temporal neighborhoods and dynamically constructing their relationships (Fig 1(c)). Thus messages can directly reach other joints in lo-
STGAT for Skeleton-Based Action Recognition

The connection strength of edges is adaptively determined to better fit local movement patterns. For further releasing the potential of STGAT, we propose several light-weight modules to reduce local feature redundancy. These modules make effect in narrowing the scope of self-attention mechanism, dynamically weighting joints along temporal dimension and separating subtle motion from static features. Equipped with the ability in spatial-temporal modeling, STGAT exhibits great progress on classifying similar action pairs, witnessed by both qualitative and quantitative results. As a result, STGAT achieves state-of-the-art performance on three large-scale datasets: NTU RGB+D 60 [27], NTU RGB+D 120 [23], and Kinetics Skeleton 400 [14]. Code is released at https://github.com/hulianyuyy/STGAT.

2 Related work

2.1 Skeleton-Based Action Recognition

Earlier methods always make use of hand-crafted features [37,38] to depict human dynamic representations. However, they fail to model dependencies across spacetime and lack enough ability in extracting high-dimensional features. Later, convolutional neural network(CNNs) and recurrent neural networks(RNNs) become mainstream by modeling human dynamics as a pseudo-image [15,18,19] or a series of coordinates along time [8,24,21,44,32]. Nevertheless, they overlook the internal relationships between joints which are better captured by graph networks due to their natural advantage over handling non-euclidean data. Graph networks in skeleton-based action recognition fall into two streams: spatial perspective and spectral perspective. ST-GCN [41] belonging to the spatial perspective firstly models human structure as a spatial graph to construct node relationships. Later methods mainly make efforts in enlarging the receptive field [10,20,30,4], combining another stream [30,20,25,29], dividing graphs into structural ones [20,34], design network structure [13,26,33,6,4], combining adaptive learning [30,12,29,45,12,31,1].

2.2 Self-Attention Mechanism with Skeleton Data

One of the main benefits of self-attention mechanism is to adaptively adjust relationships with neighbors according to their responses. A self-attention module flexibly computes the response of a position with others’ in an embedding space and then aggregates their weighted features. Self-attention mechanism has proven its power in many fields [5,22,40,36] and is recently introduced to model relationships [12,29,30,25,45,31] in skeleton-based action recognition. 2s-AGCN adds an adaptively computed graph with the physically predefined graph to flexibly model joint relationships, while MS-G3D [25], CA-GCN [46] and dynamic-GCN [42] inherit it. DSTA-Net [31] and SGN [45] proposes to alternatively employ fully attention-based spatial modeling and temporal modeling for capturing movement patterns. In contrast with previous approaches, our method doesn’t rely on a certain self-attention strategy, but proposes a spatial-temporal modeling framework for capturing long-term and especially short-term dependencies.
3 Learning STGAT

3.1 Preliminaries

**Notations.** Formally, a graph is always represented as \( G = (V, E) \), where \( V = \{v_1, ..., v_N\} \) is a set of \( N \) graph nodes representing joints and \( E \) is a series of graph edges representing bones between joints. An adjacent matrix \( A \) with size \( N \times N \) is adopted to depict bones where \( A_{i,j} \) represents the connection strength between node \( i \) and \( j \), which equals zero if they have no connections. As for input, action sequences have a node features set represented as a feature tensor \( X^{C \times T \times N} \) where each node \( v_n \in N \) has a \( C \) dimensional feature vector over total \( T \) frames.

**Graph Convolutional Networks.** Given the input tensor \( X \) and graph structure \( A \), layer-wise graph convolution at each time step can be implemented as:

\[
X^{out} = \sigma(WX^{in}A^{-\frac{1}{2}}(A + I)A^{-\frac{1}{2}})
\]

where \( X^{out} \) and \( X^{in} \) separately corresponds to the output and input features and \( W \) is a trainable weight matrix. \( A \) is the diagonal degree matrix of \( A \) to employ normalization and \( I \) adds self-loops to keep identity features. At last, \( \sigma(\cdot) \) acts as an activation function for output.

3.2 Spatial-Temporal Graph Attention Network

Following the statement in Sec. \[1] short-term dependencies are critical cues for classifying similar action pairs. We propose a general framework to capture short-term dependencies by performing spatial-temporal modeling, detailed as follows.

**Building Spatial-Temporal Graphs.** To better capture short-term dependencies, in this paper we give spatial-only modules more freedom for short-term aggregation, while maintaining the role of temporal-only modules for long-term modeling. We extend spatial-only modules to build a local robust spatial-temporal feature aggregator which directly exchange messages between nodes in a local spatial-temporal neighborhood. We start by building a spatial-temporal graph consisting of nodes in a local cross-spacetime neighborhood, which have links with not only other nodes in their original spatial graph but also their neighbors in neighboring timestamps. In this way, each node can directly aggregate messages from other local spatial-temporal neighbors.

Let us consider a sliding window of size \( \tau \) and dilation \( d \) at each time step over input sequences, which generates a local action sequence at each time step \( t \) represented as \( X^t = \{x_{t-\tau/2:t+\tau/2} \in R^{C \times \tau \times N} | t \in Z, 0 \leq t < T \} \). Here, \( \tau \) controls the local sequence length along time while \( d \) represents the dilation rate by picking a frame every \( d \) frames. Dilated window allows us to control the interval between sampled frames to fit local movement patterns. Then, for each frame \( x_t \), a unique regionally spatial-temporal adjacent matrix \( \bar{A}_t \) is organized.
by enumerating all possible neighbors in local spatial-temporal neighborhood $X_t^\tau$ as:

$$b_{A_t^\tau} = \left[ b_{A_t^\tau}(1), \ldots, b_{A_t^\tau}(\tau) \right] \in \mathbb{R}^{\tau \times N} \quad (2)$$

where $b_{A_t^\tau}(\tau)$ represents a cross-time graph with size $N \times N$ by establishing relationships between nodes at current timestamp $t$ with their neighbors at $\tau$ neighboring frames in a local spatial-temporal neighborhood. If we take one node of current timestamp $t$ as an example (i.e. one column of $b_{A_t^\tau}$), it’s connected to $\tau \times N$ neighbors consisting of $\tau$ frames with $N$ nodes at each timestamp. By incorporating $A_t^\tau$ into graph operations, the output of STGAT for each frame $x_t$ can be obtained as:

$$[X_{out}]^t = \sigma(W[x_{in}^t] A_{t}^\tau). \quad (3)$$

**Adaptive Computation for Spatial-Temporal Graph.** For well distinguishing similar action pairs, more attention should be paid to aggregate messages from important joints and less to other joints. We incorporate self-attention mechanism to adaptively compute edge weights in local spatial-temporal graphs to achieve this goal.

Following the widely used self-attention formulation, a generic self-attention operation in graph networks at each time step can be defined as:

$$\hat{A}_i = \frac{1}{C(X_{in})} \sum_{v_j} f(X_{in}^i, X_{in}^j) g(X_{in}^j) \quad (4)$$

where $i$ is the index of an output position, $j$ is the index that enumerates all possible neighbors and $\hat{A}_i$ is the computed edge weights of node $v_i$ with others. Here, the pairwise function $f(\cdot)$ computes the affinity of features between node $v_i$ and $v_j$, $g(\cdot)$ offers an embedding feature for node $v_j$, $C(\cdot)$ adds normalization for the result. Given the adjacent matrix $\hat{A}$, a weighted output $X^{out}$ can be computed as:

$$X^{out} = \sigma(WX_{in} \hat{A}). \quad (5)$$

A more stable and flexible choice is employing multi-head attention modules to learn various kinds of edge weights. Specially, $S$ independent self-attention modules are employed to learn different graph structures, as:

$$X^{out} = \sigma\left(\frac{1}{S} \sum_{s=0}^{S} W_s X_{in} \hat{A}_s\right). \quad (6)$$

Here $\hat{A}_s$ and $W_s$ are the computed adjacent matrix and weight matrix separately of $s_{th}$ head.

By incorporating our spatial-temporal graph $\hat{A}_t^\tau$ and our sampled local action sequence $X_t^\tau$ in Eq. 5, we arrive at the final form to compute edge weights and aggregate messages from neighbors in spatial-temporal graphs:

$$X^{out} = \sigma\left(\frac{1}{S} \sum_{s=0}^{S} W_s [X_{in}^t] \hat{A}_{t,s}^\tau\right). \quad (7)$$
where $\tilde{A}_{t,s}^\tau$ represents a unique computed spatial-temporal graph at timestamp $t$ with temporal length $\tau$ of the $s_{th}$ head. $X_{out}$ is the accumulated output of all heads with different spatial-temporal graphs.

Based on above design, our STGAT is a general framework which doesn’t rely on a certain self-attention strategy and can be built with self-attention operators or any attention-based approaches by inflating their spatial-only modules to construct cross-spacetime edges for short-term modeling.

**Discussion.** We give some in-depth analysis for STGAT as follows: (1) Compared with MS-G3D [25] which builds a spatial-temporal graph with size $\tau N \times \tau N$ for cross-spacetime modeling, ours owns size $\tau N \times N$. Thus, MS-G3D introduces more redundant messages from neighbors and cost $N$ times of computations than ours. More importantly, MS-G3D assigns same edge weights for joints in a local spatial-temporal neighborhood (fig.1(b)), which fails to distinguish beneficial messages and allows too much information to flood into current node without filter. In contrast, we dynamically construct joint relationships and thus is able to aggregate representative features. In this condition, MS-G3D deploys two pathways and proposes multiscale temporal convolutions to capture various short-term motion but still performs worse than ours. 2. STGAT is analogous to 3DConv [35,3] in modeling spatial-temporal dependencies by a weighted aggregation of regional cross-spacetime messages. While the parameters of 3D convolution kernels are structurally learned and fixed for samples, STGAT adaptively computes the edge weights for each sample and is less prone to overfit than 3DConv during training.

### 3.3 Reducing local spatial-temporal redundancy

While STGAT is theoretically effective for spatial-temporal aggregation, we empirically found direct aggregation of messages from spatial-temporal neighbors brings redundant static information, resulted from similar contents in adjacent frames. We propose several simple modules to reduce local spatial-temporal redundancy and further release the potential of STGAT, resulting in a robust feature extractor.

**Separate Learning.** Compared to the original $N \times N$ spatial graph, STGAT builds a $\tau N \times N$ spatial-temporal graph to model local dependencies across spacetime. Obviously, the size of adjacent matrix increases $\tau$ times. That makes it harder for self-attention operators to adaptively construct joint relationships among neighbors with similar contexts. As we found in practice, as node number grows, attention-based modules tend to assign uniform weights around $1/(\tau \times N)$ for each joint, which thus gradually loses the ability to adaptively construct relationships. We propose Separate Learning Strategy to limit node numbers and keep beneficial information lossless to relieve this problem. Inspired by group-wise convolution [16], we propose to divide joints in a local spatial-temporal region into several small groups, where STGAT only weights edges within groups.
Fig. 2. Comparison of our proposed Explicit Motion Modeling strategy with normal $1 \times 1$ implementation. We try to separate short-term motion in local spatial-temporal region from static features for better short-term modeling.

Thus, the scope of self-attention operators is narrowed and the computation costs are proportionally lowered down as well. Features from different groups are concatenated as outputs which won’t bring information loss. Practically, while there exist lots of methods to group joints, we choose to divide them according to their distances with target joint.

**Dynamic Temporal Weighting.** As defined in Eq. 2 and 3, STGAT builds spatial-temporal graphs to model cross-spacetime information. We notice that some keyframes play more important roles than others where considering frames equally may hurt modeling short-term motion. We propose two methods to dynamically weight frames regionally as follows and compare them in the experiments:

1. We add a learnable parameter $T_{DTW}$ with size $S \times \tau$ to weight frames for $S$ heads with temporal duration $\tau$, which is updated through network propagation. Note that $T_{DTW}$ structurally considers the weights of frames in each head. It’s fixed for each sample during inference.
2. Inspired by SENet [11], we design an adaptive temporal weighting module to weight frames by considering the short-term motion in a local spatial-temporal region. We first perform 1-hop graph convolutions for current frame $X_t$ to aggregate local spatial features. Then, a $1 \times 1$ convolution kernel is utilized to enhance features of current frame $X_t$. We subtract features of regional neighbors from those of current frame to obtain motion features, which reflect the movement degree of neighboring frames against current one. Finally, after pooling along channel and temporal dimension, they are passed through a sigmoid function to weight frames in local spatial-temporal region. These weights are computed adaptively in accordance with input for regional weighting. An Illustration can be found in the supplementary material.
Fig. 3. Architecture Overview of STGAT. After Explicit Motion Modeling, input data passes r consecutive STGABs and temporal convolutions, followed by global average pooling, FC and softmax to perform classification. In each STGAB, S heads are concatenated with two 1 x 1 convolutions to transform channels. For simplicity, BN and ReLU are neglected in the figure. Best viewed with color.

Explicit Motion Modeling. Considering neighboring frames always contain similar contents, we infer only one frame can provide enough static information and short-term motion benefits recognition more, for reducing local feature redundancy. We propose Explicit Motion Modeling strategy to separate short-term motion in local spatial-temporal neighborhood from static features. As illustrated in fig.2, compared to original 1 x 1 transformations, we first split the features along temporal dimension. And then, we subtract features of adjacent frames from those of current frame to obtain motion features. In this condition, only static features of current timestamp is reserved and short-term motion is obtained. With appropriate padding, these motion features pass a 1 x 1 convolution kernel for enhancement and are finally concatenated with current frame’s features as output. Compared with normal 1 x 1 convolution, this design introduces no additional parameters but enhances the ability for distinguishing short-term motion.

Discussion. These three modules are lightweight and can be painlessly incorporated into STGAT, with quite little costs. Only Dynamic Temporal Weighting consumes S x T parameters, which are negligible in deep-learning-based networks. They make powerful effect for reducing local redundancy, resulting in superior performance of STGAT, as witnessed in sec. 4.3.
3.4 Model Architecture

**Overall Architecture.** Our method is a general framework which can be flexibly constructed upon self-attention operators or previous attention-based approaches. To demonstrate the effectiveness of STGAT, we choose DSTA-Net [31] as our baseline, which employs fully attention-based spatial-only modules and temporal-only modules alternatively. It’s already among the top-performing methods currently. As illustrated in fig.3, our model is consisted of \( r \) spatial-temporal graph attention blocks (STGABs) followed by a series of global average pooling, fully connected and softmax operations to perform prediction. Each STGAB contains a spatial-temporal graph attention module to capture regional cross-spacetime dependencies and a temporal convolution module to perform long-range temporal information aggregation. For simplicity, BN and ReLU are neglected in the figure.

**Multiple-Stream Fusion.** Inspired by 2s-AGCN [30], beneficial bone information is modeled by another stream. Following our baseline, we also calculate motion for both joint and bone streams to obtain another two motion streams. The softmax scores of input streams are averaged to give the final prediction.

4 Experiments

4.1 Datasets

**NTU RGB+D 60 and NTU RGB+D 120.** NTU RGB+D 60 [27] and NTU RGB+D 120 [23] are both large-scale widely used action recognition datasets which separately contain 56578 action sequences over 60 classes and 113945 action sequences from 120 classes. All samples are performed by 40 or 106 different volunteers modeled by their 3D locations with \( N=25 \). Two experiment setups are recommended by the author: (1) Cross-subject (X-Sub), where samples are split into training and testing groups with respect to subjects. (2) Cross-view (X-View) for NTU RGB+D 60 or Cross-Setup (X-Set) for NTU RGB+D 120, where action sequences are divided by their shooting cameras or camera setups.

**Kinetics Skeleton 400.** The Kinetics Skeleton 400 Dataset [14] is sampled from the Kinetics-400 dataset by Openpose pose estimation toolkit [2]. It contains 240,436 training and 19,796 testing sequences. Every action sequence contains 18 joints with their 2D locations and 1D confidence score at each time step.

4.2 Training Details

We use 8 STGABs in our experiments with their channels in \{64, 64, 128, 128, 256, 256, 256 and 256\} respectively. We randomly/uniformly sample 150 frames and then randomly/centrally crop 128 frames for training/test splits. The learning rate is 0.1 and divided by 10 at epochs \{60, 90\}, \{60, 90\}, and \{80, 110\} for
NTU RGB+D Datasets, and Kinetics Skeleton 400 with total 120/180 epochs. The temporal convolution following STGAB owns kernel size 7. We combine our Separate Learning Strategy with multi-head mechanism and set one head as a single group. Our model is trained on a two-2080ti machine with batch size 32 and weight decay 0.0005 in PyTorch framework. Note that, no data augmentation technique is adopted for fair comparison.

4.3 Ablation Study

| Model configurations | Top-1 | Model configurations | Top-1 |
|----------------------|-------|----------------------|-------|
| Baseline             | 88.2  | S = 1                | 89.0  |
| STGAT                | 88.0  | S = 4                | 89.7  |
| w/ SL                | 88.6  | S = 8                | **90.2** |
| w/ SL + EMM          | 89.4  |                      | 89.6  |
| w/ SL + EMM + $T_{DTW}$ | **90.2** | $\tau=1$           | 89.0  |
| w/ SL + EMM + ATW    | 90.1  | $\tau=3$, $d=1$    | **90.2** |
| w/ SL + EMM + $T_{DTW}$ + ATW | 90.1 | $\tau=3$, $d=2$ | 89.6  |
| Baseline w/ SL + EMM + $T_{DTW}$ | 89.0 | $\tau=5$, $d=1$ | 89.2  |
| Spatial connections (Baseline) | 88.2 | $\tau=5$, $d=2$ | 89.1  |
| Dense connections within $\tau$ frames | 87.2 | $\tau=7$, $d=1$ | 89.1  |
| Repeating $\tau$ times along time | 89.0 | 2 Paths, $\tau=$\{3,5\}, $d=$\{1,1\} | 89.3  |
| STGAT                | **90.2** | 2 Paths, $\tau=$\{3,5\}, $d=$\{1,2\} | 89.1  |

**Table 1.** Ablations on NTU RGB+D 60 dataset. Top-1 accuracy(%) is reported. We first compare STGAT with our baseline and then equip it with proposed strategies step by step to verify their effectiveness. Here, SL denotes Separate Learning strategy and EMM means Explicit Motion Modeling. $T_{DTW}$ represents the learnable parameter to structurally weight edges while ATW denotes Adaptive Temporal Weighting.

| Model configurations | Top-1 |
|----------------------|-------|
| STGAT                | **90.2** |
| w/ SL                | 88.6  |
| w/ SL + EMM          | 89.4  |
| w/ SL + EMM + $T_{DTW}$ | **90.2** |
| w/ SL + EMM + ATW    | 90.1  |
| w/ SL + EMM + $T_{DTW}$ + ATW | 90.1 |
| Baseline w/ SL + EMM + $T_{DTW}$ | 89.0 |
| Spatial connections (Baseline) | 88.2 |
| Dense connections within $\tau$ frames | 87.2 |
| Repeating $\tau$ times along time | 89.0 |
| STGAT                | **90.2** |

**Table 2.** Ablations on NTU RGB+D 60 dataset. Top-1 accuracy(%) is reported. We compare various STGAT settings, including different values of $S$, $\tau$, $d$. STGAT is configured with the best setting tested previously, i.e. STGAT w/ SL + EMM + $T_{DTW}$.

**Effectiveness of proposed strategies.** We incrementally build up the model with its individual components to verify their effectiveness and list the results in tab. 1. Here, SL denotes Separate Learning strategy and EMM means Explicit Motion Modeling. $T_{DTW}$ represents the learnable parameter to structurally weight edges while ATW denotes Adaptive Temporal Weighting. Compared to our baseline, directly building spatial-temporal graphs with STGAT causes 0.2% performance decrease. This demonstrates local feature redundancy severely hurts modeling short-term motion. We then testify the effectiveness of our proposed strategies for reducing local redundancy. Equipped with SL, STGAT obtains 0.6% performance boost. Through SL, STGAT deals with small groups of joints and is easier to distinguish vital connections. Combined with EMM, STGAT
STGAT for Skeleton-Based Action Recognition

achieves 0.8% accuracy boost, which demonstrates EMM makes effect by capturing subtle motion instead of raw spare features. Next, we test the effect of Dynamic Temporal Weighting strategy. STGAT gains 0.8% performance boost from $T_{DTW}$ which structurally weights frames along temporal dimension. ATW also performs well and brings 0.7% performance boost. These results indicate our strategy corrects the equal weighting problem. We also try to combine $T_{DTW}$ with ATW to perform structural and adaptive weighting together, but don’t witness further promotion. We speculate that they may overemphasize the weights. $T_{DTW}$ is adopted as default due to its high performance and low computation cost. In the following, we make a comparative test to verify the effectiveness of STGAT. We don’t build spatial-temporal graphs but directly equip our baseline with proposed strategies. In this condition, SL and EMM will still make effect. As shown in tab.1, it achieves 89.0% (+0.8%) but performs lower than STGAT (90.2%, +2.0%), which demonstrates STGAT can better capture local spatial-temporal dependencies for its local spatial-temporal modeling. Finally, we examine the effect of spatial-temporal modeling for STGAT. We compare it with spatial connections (baseline), dense connections among all joints in $\tau$ frames, and simply repeating spatial graph $\tau$ times along temporal dimension. Our STGAT outperforms all of them by a large margin ($\geq 1.2\%$) which demonstrates the effectiveness of adaptive computation for edges in a local spatial-temporal neighborhood.

Study on model structure. We then test the choices of some hyperparameters in STGAT and give their results in tab. [1] We offer $S$, the number of heads, with four choices in $\{1, 4, 8, 16\}$. We observe that larger $S$ gives higher performance which indicates the effects of multiple built graphs, and the performance reaches top when $S$ equals 8. For other STGAT configurations, like $\tau$ and $d$, we test different combinations of them. $\tau=1$ means no temporal information is incorporated and only spatial modeling is performed. We first observe STGAT outperforms baseline when $\tau=1$ and benefits from larger $\tau$, which demonstrates the effect of local spatial-temporal dependencies. The gain of $\tau$ starts to decrease when it equals 5. We infer large $\tau$ makes similar effect with temporal-only modules and causes redundancy. Considering $d$, we observe accuracy drop when $d$ increases, which shows closer frames are much more relevant in a local spatial-temporal neighborhood. We further construct two STGAT paths to make comparison with MS-G3D [25]. MS-G3D obtains performance boost by incorporating more paths for multiscale temporal information. But STGAT already earns better performance through one path with fewer computations and parameters than MS-G3D for its superior spatial-temporal modeling ability.

Effect on classifying similar actions. We demonstrate the effect of STGAT on classifying similar patterns in tab. [3] We compare STGAT with baseline and another spatial-temporal modeling method, MS-G3D, upon several similar action pairs from NTU RGB+D 60 Dataset. Compared with MS-G3D, STGAT outperforms it in all action pairs with up to 9.2% performance boost. MS-G3D
fails to distinguish short-term motion with equal weights for neighboring joints, which introduces too much redundancy, even with two paths. Compared with baseline, STGAT obtains notable improvements, up to 12.5%, upon similar action pairs. These results verify the effectiveness of our STGAT on classifying similar action pairs owing to its adaptive spatial-temporal modeling. Please refer to the supplementary material for more examples and the full table.

### Table 3.

| Classes               | MS-G3D | Baseline | STGAT |
|-----------------------|--------|----------|-------|
| Average of all classes| 89.4   | 88.2     | 90.2  |
| eat meal/snack        | 67.6   | 70.9     | 74.9  (+7.3, +4.0) |
| brushing teeth        | 87.5   | 82.1     | 89.0  (+1.5, +6.9) |
| reading               | 68.5   | 60.1     | 71.6  (+3.1, +11.5) |
| writing               | 54.4   | 51.1     | 63.6  (+9.2, +12.5) |
| taking a selfie       | 89.1   | 86.6     | 90.5  (+1.4, +3.9) |
| salute                | 89.9   | 88.0     | 93.1  (+3.2, +5.1) |

Comparison with previous methods with short-term modeling. We compare STGAT with MS-GCN [25] which assigns equal weights for local neighbors for spatial-temporal modeling. [25] also proposes MSTCN consisted of temporal convolutions with different dilations for multiscale temporal information. As shown in tab. 4, STGAT outperforms baseline as well as MSGCN by a large margin (+2.0%, +2.4%). When equipped with MSTCN, either baseline or MSGCN obtains satisfactory performance boost (+0.4%, +1.2%), accompanied by notable increase in parameters and GFLOPs. STGAT outperforms MSTCN hybrids by at least 1.2% with fewer parameters and GFLOPs and similar memory usage, which demonstrates the efficacy and efficiency of STGAT for capturing both short-term and long-term dependencies.

### Table 4.

| Methods                  | #Params(M) | GFLOPs | Training Hours | Memory(G) | Top-1(%) |
|--------------------------|------------|--------|----------------|-----------|----------|
| Baseline                 | 2.4        | 10.2   | 6.0            | 16.0      | 88.2     |
| MS-GCN [25]              | 1.4        | 12.4   | 8.6            | 18.8      | 87.8     |
| Baseline+3*MSTCN [25]    | 3.7        | 15.9   | 7.1            | 23.5      | 88.6     |
| MS-GCN+3*MSTCN [25]      | 2.7        | 18.1   | 12.2           | 25.4      | 89.0     |
| STGAT                    | 2.4        | 14.6   | 10.1           | 24.2      | **90.2** |

Visualizations. We visualize the generated edges by STGAT, dense connections among all joints, and spatial connections (Baseline), in fig. 4 to give more analysis on the effect of STGAT. We select the action "hand waving" as an example. The left hand is the most important joint of action "hand waving" which forms the movement patterns of this action. Edges generated by STGAT
all lie in the upper body and especially focuses on the left hand. In contrast, the spatial connections (Baseline) are more sparse and some land on the bottom body which has no movements. Dense connections have no focus and put same attention on all joints.

4.4 Comparison with the state-of-the-art

Table 5, 6 and 7 compare our final model with other state-of-the-art methods on three large-scale datasets NTU RGB+D 60 [27], NTU RGB+D 120 [27] and the Kinetics Skeleton 400 [14]. These methods include CNN-based methods [19], RNN-based methods [24,32], graph networks with predefined graph structure [20,25,28,31,41,7] and graph networks with adaptive adjacent matrix [30,29,31,45,42]. We first observe that most of graph-based methods gain better performance than non-graph methods by better modeling relationships. We then see graph methods with adaptive graph structure [30,29,31,45,42] or multi-scale feature aggregation [25] consistently perform better than previous methods, indicating the necessity to capture long-range dependencies. Our STGAT makes a step further to capture cross-spacetime information and outperforms all existing methods especially when MS-G3D aggregates spatial-temporal features with two paths.

**Fig. 4.** Visualizations for generated edges by STGAT, as well as dense connections among all joints, and spatial connections from adaptively computed spatial graph.
### Table 5.
Comparison of STGAT against other state-of-the-art methods on the NTU RGB+D 60 dataset. Top-1 accuracy(%) is reported.

| Methods          | X-Sub X-View |
|------------------|--------------|
| AS-GCN [20]      | 86.8 94.2    |
| 2s-AGCN [30]     | 88.5 95.1    |
| AGC-LSTM [32]    | 89.2 95.0    |
| GCN-NAS [26]     | 89.4 95.7    |
| DGNN [28]        | 89.9 96.1    |
| MS-AAGCN [29]    | 90.0 96.2    |
| Shift-GCN [7]    | 90.7 96.5    |
| PA-ResGCN [33]   | 90.9 96.0    |
| Dynamic GCN [42] | 91.5 96.0    |
| MS-G3D Net [25]  | 91.5 96.2    |
| 4s DSTA-Net [31] | 91.5 96.4    |
| 4s MST-GCN [4]   | 91.5 96.6    |
| GCsT [1]         | 91.6 96.2    |
| STGAT(Joint)     | 90.2 95.4    |
| STGAT(Bone)      | 90.6 95.8    |
| **2s-STGAT**     | **92.2 96.9**|
| **4s-STGAT**     | **92.8 97.3**|

### Table 6.
Comparison of STGAT against other state-of-the-art methods on the NTU RGB+D 120 dataset. Top-1 accuracy(%) is reported.

| Methods            | X-Sub X-Set |
|--------------------|--------------|
| SGN [45]           | 79.2 81.5    |
| 2s-AGCN [30]       | 82.9 84.9    |
| Shift-GCN [7]      | 85.9 87.6    |
| 4s DSTA-Net [31]   | 86.6 89.0    |
| MS-G3D Net [25]    | 86.9 88.4    |
| PA-ResGCN [33]     | 87.3 88.3    |
| Dynamic GCN [42]   | 87.3 88.6    |
| 4s MST-GCN [4]     | 87.5 88.8    |
| GCsT [1]           | 87.7 89.3    |
| **4s-STGAT**       | **88.7 90.4**|

### Table 7.
Comparison of STGAT against other state-of-the-art methods on the Kinetics Skeleton 400 dataset. Top-1 and Top-5 accuracies(%) are reported.

| Methods            | Top-1 | Top-5 |
|--------------------|-------|-------|
| ST-GCN [41]        | 30.7  | 52.8  |
| 2s-AGCN [30]       | 36.1  | 58.7  |
| DGNN [28]          | 36.9  | 59.6  |
| GCN-NAS [26]       | 37.1  | 60.1  |
| MS-AAGCN [29]      | 37.8  | 61.0  |
| Dynamic GCN [42]   | 37.9  | 61.3  |
| MS-G3D Net [25]    | 38.0  | 60.9  |
| 4s MST-GCN [4]     | 38.1  | 60.8  |
| **4s-STGAT**       | **39.2** | **62.8** |
References

1. Bai, R., Li, M., Meng, B., Li, F., Ren, J., Jiang, M., Sun, D.: Gcst: Graph convolutional skeleton transformer for action recognition. arXiv preprint arXiv:2109.02860 (2021)
2. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7291–7299 (2017)
3. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6299–6308 (2017)
4. Chen, Z., Li, S., Yang, B., Li, Q., Liu, H.: Multi-scale spatial temporal graph convolutional network for skeleton-based action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 1113–1122 (2021)
5. Cheng, J., Dong, L., Lapata, M.: Long short-term memory-networks for machine reading. arXiv preprint arXiv:1601.06733 (2016)
6. Cheng, K., Zhang, Y., Cao, C., Shi, L., Cheng, J., Lu, H.: Decoupling gcn with dropgraph module for skeleton-based action recognition
7. Cheng, K., Zhang, Y., He, X., Chen, W., Cheng, J., Lu, H.: Skeleton-based action recognition with shift graph convolutional network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 183–192 (2020)
8. Du, Y., Wang, W., Wang, L.: Hierarchical recurrent neural network for skeleton based action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1110–1118 (2015)
9. Fang, H.S., Xie, S., Tai, Y.W., Lu, C.: Rmpe: Regional multi-person pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2334–2343 (2017)
10. Gao, X., Hu, W., Tang, J., Liu, J., Guo, Z.: Optimized skeleton-based action recognition via sparsified graph regression. In: Proceedings of the 27th ACM International Conference on Multimedia. pp. 601–610 (2019)
11. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7132–7141 (2018)
12. Hu, Z., Lee, E.J.: Dual attention-guided multiscale dynamic aggregate graph convolutional networks for skeleton-based human action recognition. Symmetry 12(10), 1589 (2020)
13. Huang, Z., Shen, X., Tian, X., Li, H., Huang, J., Hua, X.S.: Spatio-temporal inception graph convolutional networks for skeleton-based action recognition. In: Proceedings of the 28th ACM International Conference on Multimedia. pp. 2122–2130 (2020)
14. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
15. Ke, Q., Bennamoun, M., An, S., Sohel, F., Boussaid, F.: A new representation of skeleton sequences for 3d action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3288–3297 (2017)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems 25, 1097–1105 (2012)
17. Li, B., Li, X., Zhang, Z., Wu, F.: Spatio-temporal graph routing for skeleton-based action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 8561–8568 (2019)
18. Li, B., Dai, Y., Cheng, X., Chen, H., Lin, Y., He, M.: Skeleton based action recognition using translation-scale invariant image mapping and multi-scale deep cnn. In: 2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). pp. 601–604. IEEE (2017)
19. Li, C., Zhong, Q., Xie, D., Pu, S.: Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation. arXiv preprint arXiv:1804.06555 (2018)
20. Li, M., Chen, S., Chen, X., Zhang, Y., Wang, Y., Tian, Q.: Actional-structural graph convolutional networks for skeleton-based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3595–3603 (2019)
21. Li, S., Li, W., Cook, C., Zhu, C., Gao, Y.: Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5457–5466 (2018)
22. Lin, Z., Feng, M., Santos, C.N.d., Yu, M., Xiang, B., Zhou, B., Bengio, Y.: A structured self-attentive sentence embedding. arXiv preprint arXiv:1703.03130 (2017)
23. Liu, J., Shahroudy, A., Perez, M.L., Wang, G., Duan, L.Y., Chichung, A.K.: Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding. IEEE transactions on pattern analysis and machine intelligence (2019)
24. Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal lstm with trust gates for 3d human action recognition. In: European conference on computer vision. pp. 816–833. Springer (2016)
25. Liu, Z., Zhang, H., Chen, Z., Wang, Z., Ouyang, W.: Disentangling and unifying graph convolutions for skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 143–152 (2020)
26. Peng, W., Hong, X., Chen, H., Zhao, G.: Learning graph convolutional network for skeleton-based human action recognition by neural searching. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 2669–2676 (2020)
27. Shahroudy, A., Liu, J., Ng, T.T., Wang, G.: Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1010–1019 (2016)
28. Shi, L., Zhang, Y., Cheng, J., Lu, H.: Skeleton-based action recognition with directed graph neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7912–7921 (2019)
29. Shi, L., Zhang, Y., Cheng, J., Lu, H.: Skeleton-based action recognition with multi-stream adaptive graph convolutional networks. arXiv preprint arXiv:1912.06971 (2019)
30. Shi, L., Zhang, Y., Cheng, J., Lu, H.: Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12026–12035 (2019)
31. Shi, L., Zhang, Y., Cheng, J., Lu, H.: Decoupled spatial-temporal attention network for skeleton-based action-gesture recognition. In: Proceedings of the Asian Conference on Computer Vision (2020)
32. Si, C., Chen, W., Wang, W., Wang, L., Tan, T.: An attention enhanced graph convolutional lstm network for skeleton-based action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1227–1236 (2019)
33. Song, Y.F., Zhang, Z., Shan, C., Wang, L.: Stronger, faster and more explainable: A graph convolutional baseline for skeleton-based action recognition. In: Proceedings of the 28th ACM International Conference on Multimedia. pp. 1625–1633 (2020)
34. Thakkar, K., Narayanan, P.: Part-based graph convolutional network for action recognition. arXiv preprint arXiv:1809.04983 (2018)
35. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 4489–4497 (2015)
36. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
37. Vemulapalli, R., Arrate, F., Chellappa, R.: 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. pp. 588–595 (2014)
38. Wang, J., Liu, Z., Wu, Y., Yuan, J.: Mining actionlet ensemble for action recognition with depth cameras. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1290–1297. IEEE (2012)
39. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: European conference on computer vision. pp. 20–36. Springer (2016)
40. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7794–7803 (2018)
41. Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. arXiv preprint arXiv:1801.07455 (2018)
42. Ye, F., Pu, S., Zhong, Q., Li, C., Xie, D., Tang, H.: Dynamic gcn: Context-enriched topology learning for skeleton-based action recognition. arXiv preprint arXiv:2007.14690 (2020)
43. Zhang, J., Ye, G., Tu, Z., Qin, Y., Zhang, J., Liu, X., Luo, S.: A spatial attentive and temporal dilated (satd) gcn for skeleton-based action recognition. CAAI Transactions on Intelligence Technology (2020)
44. Zhang, P., Lan, C., Xing, J., Zeng, W., Xue, J., Zheng, N.: View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2117–2126 (2017)
45. Zhang, P., Lan, C., Zeng, W., Xing, J., Xue, J., Zheng, N.: Semantics-guided neural networks for efficient skeleton-based human action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1112–1121 (2020)
46. Zhang, X., Xu, C., Tao, D.: Context aware graph convolution for skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14333–14342 (2020)
A Implementations of Self-attention modules

Under the formulation defined in Eq.4 in our manuscript, we give four different flexible instantiations for the pairwise function $f$. Other alternative choices are also applicable and may give better results. We employ Embedded Gaussian function as default for $f$.

**Gaussian.** A natural choice for $f$ is the widely used Gaussian function which can be expressed as:

$$f(x_i, x_j) = e^{x_i^T x_j}. \quad (1)$$

Here $x_i^T x_j$ is the dot-product affinity of two position inputs for simplicity. Other distance measurements are also applicable. Correspondingly, the normalization factor $C(x)$ is set as $\sum_{j} f(x_i, x_j)$.

**Embedded Gaussian.** A simple extension of the Gaussian function for $f$ is to compute it in an embedding space, as:

$$f(x_i, x_j) = e^{\theta(x_i)^T \phi(x_j)} \quad (2)$$

where $\theta(x)$ and $\phi(x)$ are two embeddings. In practice, they are always implemented with $1 \times 1$ convolutions. $C(x)$ is set as $\sum_{j} f(x_i, x_j)$. A reason for the popularity of the Embedded Gaussian function is that it can be easily implemented with the softmax function, which both serves as the activation function and adds normalization. As we can see, for a given position $i$, once we have set $C(x)$ properly, $\frac{1}{C(x)} e^{\theta(x_i)^T \phi(x_j)}$ is exactly the form of the softmax function along dimension $j$.

**Dot Product.** Except for the series of Gaussian functions, a dot-product affinity can also be adopted, as:

$$f(x_i, x_j) = \theta(x_i)^T \phi(x_j). \quad (3)$$

In this case, $C(x)$ is set as the number of possible positions in $x$ for convenience in gradient computation. Other activation functions can be followed.

**Concatenation.** Concatenation function is always implemented as:

$$f(x_i, x_j) = \sigma(W_f[\theta(x_i), \phi(x_j)]) \quad (4)$$
where $[·, ·]$ represents concatenation operation and $W_f$ is a weight vector projecting their concatenated result into a scalar. Besides, $\sigma$ serves as an activation function that is usually ReLU or LeakyReLU in practice.

Under the formulation defined above, a self-attention module can be implemented as shown in fig. 1 in practice. Here, $\theta(·)$ and $\phi(·)$ are instantiated as $1 \times 1$ convolutions and $C_e$ is adopted to reduce computation costs. $g(·) = 1$.

Empirically, $C_e$ is defined as $C_{out}/r$ where $r$ is an adjustable factor to reduce computations.

**B Implementation for Adaptive Temporal Weighting Module**

We give the implementation for the adaptive temporal weighting module in the fig. 2. We first perform 1-hop graph convolutions for current frame $X^t$ to aggregate local spatial features. Then, a $1 \times 1$ convolution kernel is utilized to enhance features of current frame $X^t$. We subtract sampled features of regional neighbors from those of current frame to obtain motion features. Finally, after pooling along channel and temporal dimension, they are passed through a sigmoid function to weight frames in local spatial-temporal region.

**C Class-wise Accuracy Comparison**

Table 1 offers class-wise accuracy comparison for MS-G3D, our baseline and STGAT on the Cross Subject setting of the NTU RGB+D dataset.
Fig. 2. Illustration of our proposed adaptively Dynamic Temporal Weighting strategy. The goal is to weight frames in spatial-temporal graph with an adaptively computed factor with size $\tau \times 1 \times 1$.

Specifically, accuracy decrease is marked in red while accuracy increase is marked with green. It can first be observed that MS-G3D and baseline have achieved great performance on most of the classes with 89.4% and 88.2% average, respectively. Our final model consistently outperforms MS-G3D and baseline in 38/60 and 46/60 classes. Especially, on several similar action pairs like 'eat meal/snack' against 'brushing teeth', 'reading' against 'writing' and 'taking a selfie' against 'salute', STGAT makes great improvements (+7.3%/+1.5%, +3.1%/+9.2%, +1.4%/+3.2%) over MS-G3D and (+4.0%/+6.9%, +11.5%/+12.5%, +3.9%/+5.1%) over baseline due to the special care of short-term dependencies.
| Classes                  | #ID | MS-G3D Baseline | STGAT  |
|-------------------------|-----|----------------|--------|
| Average of all classes  | -   | 89.4           | 88.2   | 90.2  |
| drink water             | 0   | 85.8           | 85.4   | 87.6  (+1.8, +2.2) |
| eat meal/snack          | 1   | 67.6           | 70.9   | 74.9  (+7.3, +4.0) |
| brushing teeth          | 2   | 87.5           | 82.1   | 89.0  (+1.5, +6.9) |
| brushing hair           | 3   | 90.8           | 88.3   | 92.7  (+1.9, +4.4) |
| drop                    | 4   | 89.8           | 91.3   | 93.5  (+3.7, +2.2) |
| pickup                  | 5   | 91.6           | 92.4   | 92.7  (+1.1, +0.3) |
| throw                   | 6   | 94.2           | 92.7   | 88.4  (-5.8, -4.3) |
| sitting down            | 7   | 94.5           | 92.7   | 97.4  (+2.9, +4.7) |
| standing up             | 8   | 98.2           | 97.1   | 97.4  (-0.8, +0.3) |
| clapping                | 9   | 82.8           | 78.4   | 81   (-1.8, +2.6) |
| reading                 | 10  | 68.5           | 60.1   | 71.6  (+3.1, +13.1) |
| writing                 | 11  | 54.4           | 51.1   | 63.6  (+9.2, +12.5) |
| tear up paper           | 12  | 90.4           | 91.1   | 86.3  (-4.1, -4.8) |
| wear jacket             | 13  | 98.9           | 96.4   | 98.5  (-0.4, +2.1) |
| take off jacket         | 14  | 96.7           | 96.4   | 97.1  (+0.4, +0.7) |
| wear a shoe             | 15  | 87.2           | 85.0   | 84.2  (-3.0, -0.8) |
| take off a shoe         | 16  | 74.5           | 82.8   | 78.5  (+4.0, -4.3) |
| wear on glasses         | 17  | 91.6           | 91.9   | 89.4  (-2.2, -2.5) |
| take off glasses        | 18  | 91.6           | 93.8   | 90.5  (-1.1, -3.3) |
| put on a hat/cap        | 19  | 95.2           | 94.1   | 94.1  (-1.1, +0.0) |
| take off a hat/cap      | 20  | 94.9           | 94.1   | 97.1  (+2.2, +5.0) |
| cheer up                | 21  | 95.3           | 91.6   | 92.0  (-2.3, +0.4) |
| hand waving             | 22  | 92.3           | 89.4   | 93.0  (+0.7, +3.6) |
| kicking something       | 23  | 96.4           | 95.3   | 97.1  (+0.7, +1.8) |
| reach into pocket       | 24  | 83.9           | 81.8   | 80.3  (-3.6, -1.5) |
| hopping (one foot jumping) | 25     | 97.5           | 96.4   | 98.9  (+1.4, +2.5) |
| jump up                 | 26  | 99.3           | 97.8   | 100  (+0.7, +2.2) |
| make a phone call/answer phone | 27     | 86.2           | 82.5   | 83.6  (-2.6, +1.1) |
| playing with phone/tablet | 28     | 72.4           | 68.4   | 82.5  (+10.1, +14.1) |
| typing on a keyboard    | 29  | 67.3           | 75.6   | 69.8  (+2.5, -5.8) |
| pointing to something with finger | 30     | 81.2           | 75.0   | 80.8  (-0.4, +5.8) |
| taking a selfie         | 31  | 89.1           | 86.6   | 90.5  (+1.4, +3.9) |
| check time (from watch) | 32  | 88.4           | 84.4   | 91.7  (+3.3, +7.3) |
| rub two hands together  | 33  | 86.6           | 88.8   | 88.4  (+1.8, -0.4) |
| nod head/bow            | 34  | 94.6           | 93.1   | 96.0  (+1.4, +2.9) |
| shake head              | 35  | 94.5           | 93.8   | 93.8  (-0.7, +0.0) |
| wipe face               | 36  | 87.3           | 81.5   | 88.4  (+0.9, +6.9) |
| salute                  | 37  | 89.9           | 88.0   | 93.1  (+3.2, +5.1) |
| put the palms together  | 38  | 92.4           | 92.8   | 93.1  (+0.7, +0.3) |
| cross hands in front (say stop) | 39     | 94.9           | 89.5   | 94.6  (-0.3, +5.1) |
| sneeze/cough            | 40  | 75.7           | 73.2   | 75.0  (-0.7, +1.8) |
| staggering              | 41  | 99.6           | 96.4   | 97.8  (-1.8, +1.4) |
| falling                 | 42  | 97.8           | 95.6   | 97.8  (+0.0, +2.2) |
| Action                                                               | Class | Accuracy | Jain et al. | Lianyu et al. |
|---------------------------------------------------------------------|-------|----------|-------------|---------------|
| touch head (headache)                                              | 43    | 83.8     | 81.5        | 74.3 (-9.5, -7.2) |
| touch chest                                                         | 44    | 90.2     | 90.2        | 91.3 (+1.1, +1.1) |
| touch back (backache)                                              | 45    | 94.9     | 93.1        | 90.2 (-4.7, -2.9) |
| touch neck (neckache)                                               | 46    | 86.2     | 80.4        | 86.5 (+0.3, +6.1) |
| nausea or vomiting condition                                       | 47    | 84.0     | 84.7        | 85.1 (+1.1, +0.4) |
| use a fan /feeling warm                                            | 48    | 91.6     | 86.5        | 90.9 (-0.7, +4.4) |
| punching/slapping other person                                     | 49    | 89.8     | 90.9        | 91.2 (+1.4, +0.3) |
| kicking other person                                                | 50    | 92.0     | 92.0        | 94.9 (+2.9, +2.9) |
| pushing other person                                                | 51    | 92.1     | 94.2        | 97.1 (+5.0, +2.9) |
| pat on back of other person                                         | 52    | 92.4     | 80.1        | 96.7 (+4.3, +16.6) |
| point finger at the other person                                    | 53    | 87.7     | 86.2        | 92.0 (+4.3, +5.8) |
| hugging other person                                                | 54    | 98.5     | 96.4        | 98.5 (+0.0, +2.1) |
| giving something to other person                                   | 55    | 93.1     | 91.7        | 93.8 (+0.7, +2.1) |
| touch other person’s pocket                                         | 56    | 92.0     | 89.8        | 93.8 (+1.8, +4.0) |
| handshaking                                                         | 57    | 97.1     | 95.3        | 96.4 (-0.7, +1.1) |
| walking towards each other                                          | 58    | 97.8     | 98.5        | 97.8 (+0.0, -0.7) |
| walking apart from each other                                        | 59    | 96.0     | 95.7        | 94.6 (-1.4, -1.1) |

Table 1: Class-wise Accuracy Comparison for MS-G3D [25], our baseline (DSTA-Net [31]) and STGAT on the Cross Subject setting of the NTU RGB+D 60 dataset, using the joint only. Values in the parentheses are accuracy differences compared to MS-G3D or baseline. Accuracy decrease is marked in red while accuracy increase is marked with green. STGAT outperforms MS-G3D and baseline in 38/60 and 46/60 action classes, respectively. Especially, on several similar action pairs like ‘eat meal/snack’ against ‘brushing teeth’, ‘reading’ against ‘writing’ and ‘taking a selfie’ against ‘salute’, STGAT makes great improvements.