Towards To-a-T Spatio-Temporal Focus for Skeleton-Based Action Recognition

Lipeng Ke¹, Kuan-Chuan Peng², Siwei Lyu¹

¹ University at Buffalo, State University of New York
² Mitsubishi Electric Research Laboratories
lipengke@buffalo.edu, kpeng@merl.com, siweilyu@buffalo.edu

Abstract

Graph Convolutional Networks (GCNs) have been widely used to model the high-order dynamic dependencies for skeleton-based action recognition. Most existing approaches do not explicitly embed the high-order spatio-temporal importance to joints’ spatial connection topology and intensity, and they do not have direct objectives on their attention module to jointly learn when and where to focus on in the action sequence. To address these problems, we propose the To-a-T Spatio-Temporal Focus (STF), a skeleton-based action recognition framework that utilizes the spatio-temporal gradient to focus on relevant spatio-temporal features. We first propose the STF modules with learnable gradient-enforced and instance-dependent adjacency matrices to model the high-order spatio-temporal dynamics. Second, we propose three loss terms defined on the gradient-based spatio-temporal focus to explicitly guide the classifier when and where to look at, distinguish confusing classes, and optimize the stacked STF modules. STF outperforms the state-of-the-art methods on the NTU RGB+D 60, NTU RGB+D 120, and Kinetics Skeleton 400 datasets in all 15 settings over different views, subjects, setups, and input modalities, and STF also shows better accuracy on scarce data and dataset shifting settings.

Introduction

As a fundamental task in computer vision, action recognition (Simonyan and Zisserman 2014; Varol, Laptev, and Schmid 2017; Carreira and Zisserman 2017; Si et al. 2018; Lei et al. 2019; Shi et al. 2020a; Si et al. 2019; Shi et al. 2020a) has a wide range of applications in human-computer interaction (Liu, Liu, and Chen 2017), video surveillance (Ji et al. 2012), and sports analysis (Herath et al. 2017). Existing action recognition methods can be categorized into video-based (Simonyan and Zisserman 2014) and skeleton-based (Si et al. 2018) which take the video and skeleton sequences as inputs, respectively. In recent years, with the improvement in hardware (e.g., MS Kinect) and skeleton extraction algorithms such as (Cao et al. 2017), the skeleton-based action recognition methods have received more attention for their low dimensional representation and robustness to the background changes (Johansson 1973).

Since the action sequence is a time series of human joint locations, we can represent it as a three-dimensional tensor, with the joints layout being the spatial dimension and joints movement in time series as the temporal dimension. To model the temporal information, earlier deep neural network based action recognition methods (e.g., (Liu et al. 2017)) model the temporal movement of joints across frames directly using recurrent neural network with long short-term memory. However, these methods do not explicitly consider the spatial dependencies among different joints (Figure 1(d), middle-row). Subsequently, many methods (Si et al. 2018; Liu et al. 2020a; Peng et al. 2020; Lei et al. 2019; Li et al. 2019b) model the skeleton topology as graphs and use graph convolutional networks (GCNs) to model the spatial connections (topology and intensity) among joints (Figure 1(c), top row). The static topological connections
of the joints are captured by the adjacency matrix in Figure 1(c) top row, and the connection intensity learned from data is shown in Figure 1(c) middle row. Both types of methods (Figure 1(c) top & middle rows) model the fixed physically constrained topology connections that cannot be adapted for varying temporal dependencies in the input sequence. Some recent works (Ye et al. 2020; Shi et al. 2021; Chen et al. 2021b; Zeng et al. 2021) model dynamic connection topology from data, but the dynamic connection topology is generated from a forward pass through the model without any objective directly regularizing the adjacency matrices using spatio-temporal focus.

**Spatio-temporal modeling** Since not all joints and frames are equally important for the recognition (Shi et al. 2021), and only specific joints (spatial) with specific motion (temporal) are critical to distinguish different action classes (Ding, Yang, and Chen 2019), finding these critical joints (spatial) and the motion patterns (temporal) jointly in skeleton sequences is important for action recognition. However, most existing methods (Shi et al. 2020a; Shi et al. 2019; Cho et al. 2020; Xie et al. 2018) simply create the attention modules using trainable parameters, which do not have the objectives to directly enforce the modules to capture the varying spatial and temporal patterns jointly.

To better guide the classifier about when and where to look at, and model the learnable dynamic joints connection in spatio-temporal domain, we propose the **To-a-T Spatio-Temporal Focus** method (termed as STF), which uses the joint spatial and temporal information jointly (shown in Figure 1(c) bottom). Specifically, we extract the spatio-temporal focus that strongly influences the recognition in training by projecting the backward gradient to the spatio-temporal domain. Then we propose the STF modules to generate dynamic instance-dependent adjacency matrices, and we use the obtained spatio-temporal focus to regularize the dynamic adjacency matrices, such that the matrices reflect not only the high-order topology connections but also the spatio-temporal importance given the input skeleton sequence. Besides, we also use the gradient-based spatio-temporal focus to encourage the classifier to better emphasize on the critical spatio-temporal inputs and features. To achieve these goals, we introduce three loss terms: (1) the STF exploration loss to enforce the classifier to make prediction over all the critical joints; (2) the STF divergence loss to minimize the similarity of the focus for different classes, and (3) the STF coherence loss to focus on consistent spatio-temporal features across the stacked STF modules. Our proposed STF method outperforms the state-of-the-art skeleton-based action recognition methods in all 15 settings over different views, subjects, setups, and input modalities on the NTU RGB+D 60, NTU RGB+D 120, and Kinetics Skeleton 400 datasets.

Our contributions are summarized below:

- We propose the To-a-T Spatio-Temporal Focus framework (STF) as a flexible framework trained with spatio-temporal gradient for skeleton-based action recognition.
- We design the novel STF module that generates dynamic connection topology and intensity, and propose a loss to incorporate the spatio-temporal focus to regularize the spatio-temporal connection topology and intensity.
- We propose three loss terms defined on the gradient-based spatio-temporal focus to explicitly guide the classifier when and where to look at, distinguish confusing classes, and optimize the stacked STF modules.
- Our proposed STF framework outperforms the SOTA not only in three benchmarks but also in the scarce data and dataset shifting settings.

### Related Works

**Graph neural network for action recognition.** Graph convolutional networks (GCNs) (Zhou et al. 2018) are a type of GNNs that extend convolutional operations to the adjacency matrices of structured data types that can be modeled as graphs. GCNs have been adapted widely to a number of applications including graph classification/embedding/prediction, link prediction, node classification etc. The first work applying GCN to skeleton-based action recognition is ST-GCN (Yan, Xiong, and Lin 2018), which constructs a spatio-temporal graph with the joints as the nodes and skeletal connectivity as the edges to model joint dependencies. However, ST-GCN only considers joints that are directly connected in the skeleton, which limits its representation capacity. Subsequently, multi-scale GCNs (Li et al. 2019b; Liu et al. 2020a) are proposed to capture dependencies among joints that are not neighbors in the skeleton graph. These methods use higher order polynomials of the adjacency matrix to aggregate features from non-adjacent joints. 2s-GCN (Li et al. 2019) further adapts the adjacency matrix to model the learnable dynamic intensity of the joints connection using an embedding function. 2s-GCN also popularizes the use of multi-stream inputs, such as joint, bone, joint motion, bone motion, angular, etc., for skeleton-based action recognition (Lei et al. 2019; Shi et al. 2020a; Shi et al. 2020b; Lei et al. 2019, and only specific joints (spatial) with specific motion (temporal) are critical to distinguish different action classes (Ding, Yang, and Chen 2019), finding these critical joints (spatial) and the motion patterns (temporal) jointly in skeleton sequences is important for action recognition. However, most existing methods (Shi et al. 2020a; Shi et al. 2019; Cho et al. 2020; Xie et al. 2018) simply create the attention modules using trainable parameters, which do not have the objectives to directly enforce the modules to capture the varying spatial and temporal patterns jointly.

<table>
<thead>
<tr>
<th>property / method (group)</th>
<th>( G_1 )</th>
<th>( G_2 )</th>
<th>( G_3 )</th>
<th>STF</th>
</tr>
</thead>
<tbody>
<tr>
<td>adjacency matrix w/ learnable dynamic intensity</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>adjacency matrix w/ learnable dynamic topology</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>adjacency matrix w/ spatio-temporal focus</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>attention (modules) w/ direct supervision</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>conjunction of all the above</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

Table 1: Comparison between STF and recent methods of skeleton-based action recognition. Among all the listed methods, STF models one of the most generalized types of adjacency matrices with explicit backward-knowledge spatio-temporal focus regularization. Method groups: \( G_1 \): (Ye et al. 2020; Shi et al. 2021; Chen et al. 2021b; Zeng et al. 2021), \( G_2 \): (Song et al. 2020b,a; Yang and Yin 2020; Li et al. 2020; Cheng et al. 2020; Heidari, Alex, and ros Iosifidis 2020a,b; Liu et al. 2020a; Yan, Xiong, and Lin 2018; Yang et al. 2020a; Ding, Yang, and Chen 2019; Fan et al. 2020; Yang et al. 2020b; Li et al. 2019b; Qin et al. 2021; Chen et al. 2021a; Plizzari, Cannici, and Matteucci 2020; Shi et al. 2020a,b; Lei et al. 2019), \( G_3 \): (Li, Zhang, and Li 2020; Pan, Chen, and Ortega 2021; Cho et al. 2020; Peng et al. 2020; Si et al. 2018; Liu et al. 2017; Song et al. 2017a; Xie et al. 2018; Huang et al. 2020; Si et al. 2019).
utilizes the spatio-temporal focus acquired by backward gradients to improve the graph modeling in typical GCN and capture critical spatio-temporal features. We first propose the STF modules, which generate learnable and instance-dependent adjacency matrices. Then we use the gradient-based spatio-temporal focus to supervise the STF modules, such that the adjacency matrices capture both the high-order dynamic dependency and encode spatio-temporal importance. Second, we propose the STF exploration loss, STF divergence loss, and STF coherence loss to explicitly enforce the classifier to predict based on all critical joints and frames across the input, to focus on discriminative spatio-temporal features of confusing classes, and to have consistent spatio-temporal attention across the stacked STF modules, respectively. With these proposed losses, STF is better guided when and where to look at, distinguishing confusing classes, and distilling the high-level module’s focus to low-level modules.

Spatio-Temporal Focus (STF) Module

A human skeleton graph is defined as $(V, E)$, where $V = \{v_1, ..., v_N\}$ is the set of $N$ nodes representing the joints, and $E$ is the edge set representing the bones by an adjacency matrix $A \in \mathbb{R}^{N \times N}$. The input skeleton sequence is formulated as a tensor $X \in \mathbb{R}^{C \times T \times N}$, where $C$, $T$, and $N$ are the feature dimension (for 2D/3D input, $C=2/3$), number of frames, and number of joints, respectively.

In the vanilla adjacency matrix (Figure 1(c) top row), $A_{i,j} = 1$ if the joints corresponding to nodes $v_i$ and $v_j$ are connected in the skeleton, and 0 otherwise. $A$ is symmetric since the graph is undirected. We build up our STF module upon (Liu et al. 2020a), which encodes multi-scale adjacency matrices $A_k$. $A_k$ considers connections via fewer than $k$ intermediate joints as spatial connections at the scale of $k$. Since $A_k$ is still constrained by the natural topology of human skeleton, which cannot model the high-order relationships in the spatio-temporal domain, we propose an instance-dependent and learnable dynamic adjacency matrix $G_k$ (Figure 1(c) bottom row) to model the high-order dependency along with $A_k$. The features of the STF modules are computed as:

$$X^{l+1} = \sum_{k=1}^{K} W_k X^l (A_k + G_k)$$

where $K = 3$ denotes the scale size of the spatial dimension, $W_k$ is the weight of the convolution operation, $X^l$ is the feature at the $l$-th layer, and $G_k$ is our proposed instance-dependent adjacency matrix generated from $X^l$.

As shown in Figure 2(d), STF firstly uses a three-layer convolutional module $\beta(\cdot)$ to extract the spatio-temporal embedding $\beta(X^l)$ of dimension $T \times N$ on input $X^l$. Then we apply a CNN with two convolutional layers $\alpha(\cdot)$ to convert the embedding to $G_k = \alpha(\beta(X^l))$ of dimension $N \times N$. Now the embedding $\beta(X^l)$ encodes a data-dependent graph that learns a unique graph for each sample via $\alpha(\cdot)$, but it does not guarantee that $\beta(X^l)$ can represent the high-order topology and importance in the spatio-temporal domain. To address this issue, we propose to use backward gradient to
model spatio-temporal focus to supervise the learning of the STF modules.

To extract the spatio-temporal focus to guide the learning, we apply the Grad-CAM (Selvaraju et al. 2017) to the last layer (which contains more abstract features and has a larger receptive field than the previous layers) and extract the spatio-temporal focus $Q$ as follows:

$$ Q = ReLU \left( \sum_c \sum_{t,v} \frac{1}{Z} \frac{\partial \hat{y}}{\partial X_{ctv}} X'_{ctv} \right), \quad (2) $$

where $\hat{y}$ is the probability of the predicted class $y$, $c$, $t$, and $v$ denote the channel, temporal, and spatial dimensions of the intermediate feature map $X'_{ctv}$, respectively, and $Z$ is the normalization factor in the spatio-temporal dimension. $Q$ is normalized to $[0, 1]$ by the min-max normalization. Then we learn the spatio-temporal embedding network $\beta$ using $Q$ in Eq. 2 as supervision via $L_{G_k} = \|Q - \beta(X')\|_2$, which ensures that the spatio-temporal embedding encodes the instance-dependent and high-order spatio-temporal dependencies. As we use the spatio-temporal focus $Q$ as the guidance, the accuracy of $Q$ becomes critical. We propose the following three objectives to further regularize $Q$.

### STF Exploration

Firstly, the STF exploration loss is introduced to ensure the complete and global view of the spatio-temporal focus $Q$. Typical classification losses, such as the cross-entropy loss used in (Krizhevsky, Sutskever, and Hinton 2012), do not enforce the classifier to infer based on all critical skeleton joints in all important frames. Thus there is no constraint that the classifier will look over the entire critical parts in the spatio-temporal space. The classifier which only pays attention to partial critical parts can predict incorrectly when such information is occluded or noisy (Li et al. 2019a).

Therefore, we propose the STF exploration loss $L_e$ to enforce the classifier’s spatio-temporal focus to cover all the critical joints and all the critical frames. Specifically, suppose that the input sequence is classified to class $y$. An ideal classifier’s focus should cover the entire critical parts across the input sequence. If we mask the corresponding parts of the spatio-temporal input sequence, the predicted probability of class $y$ should be as low as possible. We formulate this process as:

$$ L_e = g^y(X - Q \odot X'), \quad (3) $$

where $g^y(\cdot)$ extracts the prediction score of the spatio-temporal focus masked input $(X - Q \odot X')$ at the original prediction class $y$ in Eq. 2, $\odot$ is the element-wise multiplication, and $X'$ is the input skeleton sequence. $L_e$ uses an exclusion strategy by eliminating $Q$’s corresponding spatio-temporal input to guide the network to focus on the critical parts across the input sequence. We observed that the focus $Q$ expands to include more critical joints in the spatio-temporal space than $Q$ without using $L_e$.

### STF Divergence

Secondly, we propose the STF divergence loss to encourage the classifier to focus on different parts when predicting different classes. The skeleton sequences of different classes usually involve either different joints or different temporal movements, or both. For example, the reading and walking classes involve different joints (the upper body and lower body parts). In contrast, the walking and running classes

Figure 2: The illustration of the proposed objectives (all the spatio-temporal focus is projected to the input sequence for visualization): (a) STF exploration loss $L_e$, which masks the input sequence according to the focused parts and minimizes the probability of the originally predicted class of the masked input sequence; (b) STF coherence loss $L_c$ enforces the focus to be coherent across the last two STF modules; (c) STF divergence loss $L_d$ enforces the top two predicted classes to have different focused parts; (d) STF adjacency matrix and its loss function $L_G$, which encourages the consistency between the spatio-temporal focus and STF adjacency matrix such that the adjacency matrix is adaptive, high-order and instance-dependent. The yellow boxes in STF-GCN-1 $\sim$ STF-GCN-6 are not shared (i.e., each STF module has its own STF adjacency matrix).
involve similar parts (legs and arms) but have different temporal patterns (legs/arms’ movement is different). These differences can be represented by the differences of their spatio-temporal focus. Inspired by the finding that overlapped attention of different classes causes visual confusion (Wang et al. 2019), we propose the STF divergence loss \( L_d \) to reduce the overlap of the focus of different classes. This loss discourages the network from covering all the parts across the sequence and focuses more on different classes’ discriminative features. Although the concept of this loss can be applied to all the confusing classes, to simplify the optimization, we mainly focus on separating the most confusing classes (e.g., touching neck vs. touching head). Specifically, given an input sequence, we select the top two predictions as the most confusing classes and enforce the top two classes’ focus to overlap as little as possible.

Similar to \( L_c \), we define \( L_d \) with the spatio-temporal focus \( Q \) via Eq. 2 as:

\[
L_d = -||Q^{y_i} - Q^{y_j}||_2, \tag{4}
\]

where \( Q^{y_i} \) and \( Q^{y_j} \) are the spatio-temporal focus of the top two prediction classes \( y_i \) and \( y_j \), respectively. We observed that by introducing \( L_d \), the top two predictions’ focus overlaps less.

**STF Coherence**

Thirdly, we propose the STF coherence loss to utilize the high-level GCN module’s focus to assist the low-level GCN module’s learning. In skeleton-based action recognition, stacked GCNs are commonly used network structure (Si et al. 2018; Liu et al. 2020a; Peng et al. 2020; Lei et al. 2019). The modules closer to the final output (high-level modules) capture more abstract information with a larger receptive field. The modules closer to the input (low-level modules) have relatively smaller receptive fields. Thus, the high-level modules’ focus has a more global view of the input skeleton sequence than that from the low-level modules. We can use the focus of a more global view from the high-level module to guide the learning of the low-level modules. Moreover, the stacked GCNs operate on the same input sequence; thus, if there is an optimal focus for these networks to pay attention to, it will be more likely to be the focus of the high-level module. Therefore, we propose the STF coherence loss \( L_c \), which enforces the stacked GCNs to have coherent focus across the network:

\[
L_c = ||Q_i - Q_j||_2, \tag{5}
\]

where \( Q_i \) and \( Q_j \) are the spatio-temporal focus from the STF modules \( i \) and \( j \), respectively. Ideally, the concept of \( L_c \) can be applied to all the GCN modules in addition to the GCN modules \( i \) and \( j \). For the convenience and ease of optimization, we choose the last two GCN modules (STF-GCN-5 and STF-GCN-6 in Figure 2) as the GCN modules \( i \) and \( j \) given that they capture more abstract information of the input sequence.

We qualitatively verify the efficacy of \( L_c \) where we noticed that with \( L_c \), the last two GCN modules’ focus is more coherent than that without \( L_c \), and that enforcing the coherence of the focus from different GCN modules can correct the misclassification from the classifier which does not use \( L_c \).

**Overall STF Loss**

We use our proposed STF modules and objectives together with the cross-entropy loss \( L_{ce} \) for skeleton-based action recognition. Specifically, the overall loss is:

\[
L = L_{ce} + \lambda_e L_e + \lambda_d L_d + \lambda_c L_c + \lambda_G L_G, \tag{6}
\]

where \( \lambda_e, \lambda_d, \lambda_c, \lambda_G \) are the weights of the losses, so that each loss term has comparable absolute range. We separate \( L_c \) from \( L_d, L_e, \) and \( L_G \) during training, and ensemble them during testing because we find that optimizing \( L_c \) and \( L_d \) together makes the training process unstable. We hypothesize that it is because the goals of \( L_e \) and \( L_d \) can be conflicting implicitly – \( L_e \) tends to expand the focus to include all critical joints, but \( L_d \) typically shrinks the focus to reduce the overlap of the focus corresponding to confusing classes.

**Datasets**

We conduct experiments on three benchmark datasets, namely, the NTU RGB+D 60 (Shahroudy et al. 2016), NTU RGB+D 120 (Liu et al. 2019), and Kinetics Skeleton 400 (Kay et al. 2017) datasets (denoted as NTU-60, NTU-120, and Kinetics-400, respectively).

**NTU RGB+D 60** (Shahroudy et al. 2016) is a large-scale skeleton-based action recognition dataset with over 60 action classes of 40 subjects for indoor scenarios. Each sequence contains one or two persons’ 3D skeletons captured by three Kinect v.2 cameras in three views (termed as views 1, 2, and 3). The recommended two settings are (1) cross-Subject (x-sub), where the dataset is equally split as training and testing sets of 20 subjects each; and (2) cross-View (x-view), where all samples from view 1 are used for testing and the samples from views 2/3 are used for training.

**NTU RGB+D 120** (Liu et al. 2019) extends the NTU-60 dataset with 60 extra action classes, resulting in a total of 120 action classes, for 106 subjects. The recommended two settings are (1) cross-Subject (x-sub), the 106 subjects are split into 53/53 subjects for training/testing; and (2) cross-Setup (x-set), where 16/16 setups are used for training/testing.

**Kinetics Skeleton 400** (Kay et al. 2017) is a skeleton-based action recognition dataset converted from the Kinetics 400 video dataset (Kay et al. 2017) using the OpenPose (Cao et al. 2017) toolbox in 2D keypoints modality.

**Implementation Details and Protocols**

**Data preparation.** We preprocess the input skeleton sequences by subtracting each joint position by the center joint position and normalizing the results by the body height, in the same fashion as (Si et al. 2018). No other data processing or augmentation is used for fair comparisons. All skeleton sequences are padded to \( T=300 \) frames by repeating the sequences. To keep the semantic information of the skeleton sequence in Eq. 3, as done in (Si et al. 2018),
we add a visibility channel alongside the X-Y-Z location channel to indicate the joint’s visibility, where 1/0 means visible/invisible. The initial visibility channel is set to 1. The visibility channel is applied to the NTU-60 and NTU-120 datasets. For the Kinetics-400 dataset, the visibility is embedded with the confidence score channel of joints. Unless otherwise specified, we use the default settings for other parameters.

**Experimental protocol.** Following the experimental protocol of Shi et al. (Lei et al. 2019), we report the classification accuracies under three different input modalities: (i): joint; (B): bone; “×”: results not provided in the reference. The methods requiring more inputs than J+B: “∗”: (Song et al. 2020a) uses 3 streams; “‡”: (Cheng et al. 2020; Li et al. 2020) use J+B+motion+B motion; “⋄”: (Shi et al. 2020a) uses spatial-temporal, spatial, slow-temporal, and fast-temporal streams; “+”: (Song et al. 2020b) uses J+B+velocity.

**Training scheme.** We implement STF using MS-G3D (Liu et al. 2020a) as the backbone. The MS-G3D baseline model is trained using SGD with momentum 0.9, batch size 32, initial learning rate 0.05, and weight decay 0.0005, and the base learning rate is adjusted accordingly for different settings. For NTU-60, NTU-120, and Kinetics-400, the learning rate is decayed at [20, 35, 45], [20, 35, 50], [25, 40, 55] epochs, respectively. After that, we pre-train the STF model from the MS-G3D baseline model, with lower initial learning rates $\{10^{-3}, 5 \times 10^{-4}, 10^{-4}\}$. We empirically set $\lambda_c/\lambda_s/\lambda_{d}\lambda_{c}$ as 0.01/0.1/0.1/0.1, respectively, such that all loss terms have comparable absolute ranges.

**Joint-bone stream fusion.** Moreover, we follow Shi et al. (Lei et al. 2019) to combine the results of joint and bone streams by averaging the prediction probability for each sequence, and report them in Table 2. Specifically, we merge the J and B streams by averaging the prediction probability from both streams to get the J+B results of STF.

### Experimental Results

**Baselines.** We summarize the experimental results in Table 2, where we compare STF with the SOTA skeleton-based action recognition methods. Of all the methods in Table 2, MS-G3D (Liu et al. 2020a) is one of the most competitive methods, so we build STF on top of it. However, when we run the publicized code (Liu et al. 2020b) of MS-G3D, we found that its accuracy is lower than that reported in the paper (Liu et al. 2020a). Since we implement STF based on the publicized code of MS-G3D, we use the accuracy obtained from running the publicized code of MS-G3D for fair comparisons. We also include the performance reported in the original paper of MS-G3D (Liu et al. 2020a) in Table 2, where we mark the best performance in **bold**.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NTU RGB+D 60</th>
<th>NTU RGB+D 120</th>
<th>Kinetics-400</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SR-TSL</strong> (Si et al. 2018)</td>
<td>84.80</td>
<td>-</td>
<td>92.40</td>
</tr>
<tr>
<td><strong>2s-AGCN</strong> (Lei et al. 2019)</td>
<td>-</td>
<td>88.50</td>
<td>93.70</td>
</tr>
<tr>
<td><strong>TS-SAN</strong> (Cho et al. 2020)</td>
<td>87.20</td>
<td>-</td>
<td>92.70</td>
</tr>
<tr>
<td><strong>GCN-NAS</strong> (Peng et al. 2020)</td>
<td>-</td>
<td>89.40</td>
<td>94.60</td>
</tr>
<tr>
<td><strong>MS-TGN</strong> (Li, Zhang, and Li 2020)</td>
<td>86.60</td>
<td>87.50</td>
<td>89.50</td>
</tr>
<tr>
<td><strong>MS-AAGCN</strong> (Shi et al. 2020b)</td>
<td>88.00</td>
<td>88.40</td>
<td>89.40</td>
</tr>
<tr>
<td><strong>3s RA-GCN</strong> (Song et al. 2020a)</td>
<td>-</td>
<td>87.30</td>
<td>x</td>
</tr>
<tr>
<td><strong>DC-GCN+ADG</strong> (Cheng et al. 2020)</td>
<td>-</td>
<td>90.80</td>
<td>-</td>
</tr>
<tr>
<td><strong>DSTA-Net</strong> (Shi et al. 2020a)</td>
<td>-</td>
<td>91.50</td>
<td>-</td>
</tr>
<tr>
<td><strong>STIGCN</strong> (Huang et al. 2020)</td>
<td>90.10</td>
<td>-</td>
<td>96.10</td>
</tr>
<tr>
<td><strong>PA-ResGCN-B19</strong> (Song et al. 2020b)</td>
<td>-</td>
<td>90.90</td>
<td>-</td>
</tr>
<tr>
<td><strong>Dynamic-GCN</strong> (Ye et al. 2020)</td>
<td>-</td>
<td>91.50</td>
<td>-</td>
</tr>
<tr>
<td><strong>MS TE-GCN</strong> (Li et al. 2020)</td>
<td>87.40</td>
<td>88.50</td>
<td>90.80</td>
</tr>
<tr>
<td><strong>AdaSGN</strong> (Shi et al. 2021)</td>
<td>-</td>
<td>89.10</td>
<td>-</td>
</tr>
<tr>
<td><strong>Yang’s</strong> (Siyouan et al. 2021)</td>
<td>88.00</td>
<td>-</td>
<td>94.90</td>
</tr>
<tr>
<td><strong>MS-G3D (paper)</strong> (Liu et al. 2020a)</td>
<td>89.40</td>
<td>90.10</td>
<td>91.50</td>
</tr>
<tr>
<td><strong>MS-G3D (code)</strong> (Liu et al. 2020a)</td>
<td>87.77</td>
<td>89.59</td>
<td>90.67</td>
</tr>
<tr>
<td><strong>MS-TG</strong> (Wang et al. 2020)</td>
<td>-</td>
<td>91.60</td>
<td>-</td>
</tr>
</tbody>
</table>

**Table 2:** Performance comparison of skeleton-based action recognition in top-1 accuracy (%). **∗:** For MS-G3D (Liu et al. 2020a), the publicized code (Liu et al. 2020b) we use as our baseline has lower accuracy than what was reported in their paper. Annotations: J: joint; B: bone; “×”: results not provided in the reference.
dependencies in STF. This is likely due to the explicit consideration of the high-order

Table 3: Comparison of accuracy gain over the baseline.

<table>
<thead>
<tr>
<th>settings</th>
<th>Kinetics-J</th>
<th>Kinetics-B</th>
<th>Kinetics-J+B</th>
</tr>
</thead>
<tbody>
<tr>
<td>∆(MS-G3D, 2×-AGCN)</td>
<td>0.64% (1.0x)</td>
<td>1.47% (1.0x)</td>
<td>1.13% (1.0x)</td>
</tr>
<tr>
<td>∆(STF, MS-G3D)</td>
<td>2.46% (3.8x)</td>
<td>2.79% (1.9x)</td>
<td>2.64% (2.3x)</td>
</tr>
</tbody>
</table>

Figure 3: The class-wise accuracy difference (%) between STF in Table 2) and the baseline MS-G3D (Liu et al. 2020a) (MS-G3D (code) in Table 2) for the NTU-60 dataset under the x-sub setting with the joint input modality.

Improvement on scarce data. To simulate the real-world scenarios where only small amount of training data are available, we train on 10/20/25% of randomly sampled training data from NTU-60 under the x-sub setting with the joint input modality, and evaluate the accuracy on the entire testing set of NTU-60. Table 4 shows that STF consistently outperforms the baseline in all settings. We also verify this on the randomly sampled 25% Kinetics-400, the STF (joint 25.56, bone 26.30) outperforms MS-G3D (joint 24.69, bone 25.22), regardless of the input modality.

Efficacy under data shift. To test the efficacy of the features learned from our proposed STF across different datasets, we test STF model on the dataset different from the training set. Specifically, we freeze the model trained on the NTU-60 dataset under the x-sub setting with joint input modality, and replace the output fully connected layer with a new fully connected layer of 120 output nodes. The new fully connected layer is fine-tuned using the training set of the NTU-120 under the x-sub setting with joint input modality. After fine-tuning, we evaluate its performance on the NTU-120 testing set. Our experimental result shows that STF outperforms the baseline MS-G3D by 1.04% (83.38% vs. 82.34%). This result supports that the features learned from STF has better generalization ability than MS-G3D.

Improvement on confusing classes. In addition, we also show a break-down evaluation of the class-wise accuracy difference (%) for the joint input modality between STF in Table 2) and the baseline MS-G3D (Liu et al. 2020a) (MS-G3D (code) in Table 2) for the x-sub setting on NTU-60. Figure 3 shows that STF outperforms MS-G3D for most classes. The highest performance gain occurs among the more challenging classes with more multiple joint correlations (e.g., touch neck +7.61% vs. touch head +4.71%, and take off a shoe +7.30% vs. wear a shoe +3.66%). This is likely due to the explicit consideration of the high-order dependencies in STF.

Table 4: The accuracy (%) using joint input modality on the NTU-60 dataset under the x-sub setting. We use p% of the randomly sampled training data from the NTU-60 dataset.

<table>
<thead>
<tr>
<th>method \ p</th>
<th>10</th>
<th>20</th>
<th>25</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS-G3D</td>
<td>72.01</td>
<td>79.11</td>
<td>81.91</td>
<td>88.77</td>
</tr>
<tr>
<td>STF</td>
<td>72.73 (10.72)</td>
<td>80.77 (11.66)</td>
<td>84.27 (12.36)</td>
<td>91.34 (12.57)</td>
</tr>
</tbody>
</table>

Table 5: Ablation study of top-1 accuracy (%) using joint only modality on the NTU-60 dataset under the x-sub setting. ∆ shows the accuracy improvement over the baseline, MS-G3D (Liu et al. 2020a).

Ablation Study

Our learning objectives are composed of several different loss terms, so it is important to know the contribution of these losses to accuracy. To this end, we perform an ablation study of the contribution of different loss terms using the joint input modality on the NTU-60 dataset under the x-sub setting in Table 5. We report the accuracy and ∆, the accuracy improvement over MS-G3D (Liu et al. 2020a) by using different combinations of loss terms as the learning objectives. STF results in ∆ = 2.57% improvement with joint only modality on the NTU-60 x-sub setting.

Conclusion

We propose the To-a-T Spatio-Temporal Focus (STF) method for skeleton-based action recognition. First, we propose the STF modules to generate flexible adjacency matrices, and use the spatio-temporal focus to guide the learning of STF modules, such that the adjacency matrices capture the high-order dependency and spatio-temporal importance. To capture critical spatio-temporal features, we propose the STF exploration, STF divergence, and STF coherence losses to encourage the spatio-temporal focus which supports the classifier’s prediction to include all critical spatio-temporal features, to distinguish different classes on spatio-temporal focus, and to make the spatio-temporal focus across the stacked GCNs consistent, respectively. STF outperforms the SOTA methods on the NTU RGB+D 60, NTU RGB+D 120, and Kinetics Skeleton 400 datasets.

We also plan to explore multi-stream input to train all our proposed objectives jointly in the future.

References


1137


