

# DGCMam: Fusing Distance Graph Convolution and Mamba for Skeleton-based Action Recognition

Juncen Long, Gianluca Bardaro, Simone Mentasti and Matteo Matteucci

Department of Electronics Information and Bioengineering, Politecnico di Milano, Milan, Italy

**Abstract**—Most skeleton-based action recognition algorithms extract spatio-temporal features from skeleton data using Graph Convolutional Networks (GCNs) and Transformers. However, typical GCNs use static learnable parameters to aggregate joint features, which prevents the model from dynamically adjusting aggregation weights based on different motion patterns. Although the Transformer can dynamically aggregate features by the self-attention mechanism, it has high computational complexity when processing long sequences. To address these limitations, we propose DGCMam, a model fusing distance graph convolution and Mamba for skeleton-based action recognition. DGCMam uses the Distance Graph Convolution (DGC) to extract features, where learnable distance parameters are applied to adjust the aggregation weights of joints based on spatial distances. Moreover, DGCMam effectively extracts global spatio-temporal features through the alternating-direction Mamba (AD-Mamba) structure, achieving lower computational complexity than bidirectional Mamba and Transformers for long sequence data. A gated fusion mechanism is also employed to enhance high-level features by selectively adding initial features. Experiments on the NTU RGB+D, NTU RGB+D 120, and NW-UCLA datasets show that DGCMam achieves the state-of-the-art performance with lower inference time.

## I. INTRODUCTION

Skeleton-based action recognition plays an important role in many areas, such as accident detection [1], intelligent surveillance [2], robot navigation [3], and sports analysis [4], because skeleton data is less sensitive to changes in lighting, perspective and background compared to images [5].

Graph Convolutional Networks (GCNs) are widely applied in skeleton-based action recognition algorithms, because the skeleton sequence data can be represented as a spatio-temporal graph, where joints are represented as nodes. However, the typical graph convolution aggregates node features using static learnable weights regardless of the input features. For instance, as shown in Fig. 1, the network adds the same proportion of the right hand's features to the left hand, whether the human is putting palms together or waving a hand, which is unreasonable because the correlation between the left and right hands is greater in putting palms together than in waving a hand.

To dynamically aggregate joint features, the Transformer [6] is applied in some algorithms, where the self-attention mechanism calculates the correlation between two joint nodes in the spatio-temporal graph based on their features. However, the length of the input sequence is the product of the number of frames and the number of joints, combined with the quadratic computational complexity of the

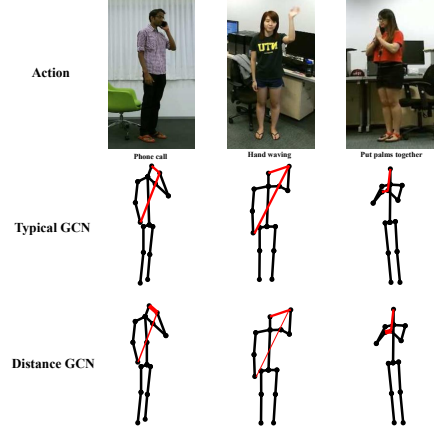


Fig. 1. An example of two types of graph convolution. Black lines represent skeletal structure; the thickness of the red line denotes the magnitude of hand-head aggregation weight and hand-hand aggregation weight.

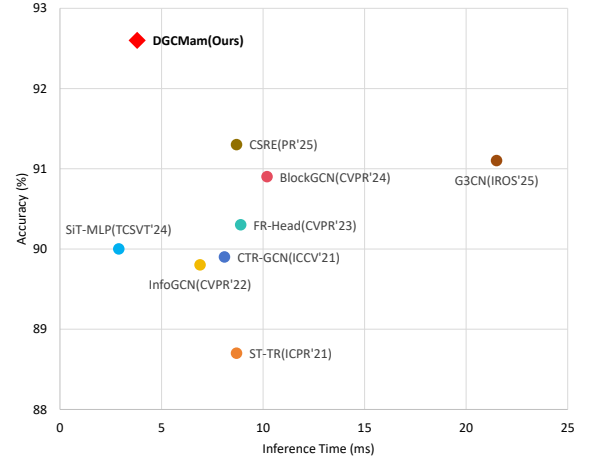


Fig. 2. The top-1 accuracy and inference time of DGCMam and SOTA algorithms for the X-Sub benchmark.

Transformer, resulting in a high computational effort for this global attention mechanism.

To address the weakness of static aggregation weights and high computation effort of global attention, we propose DGCMam, a model fusing distance graph convolution and Mamba [7] for skeleton-based action recognition.

As shown in Fig. 2, by using our proposed DGC (Distance Graph Convolution) operation, AD-Mamba (Alternating-Direction Mamba) structure, and SGF (Semantic-Guided

Fusion), DGCmam achieves the highest recognition accuracy and lower inference time compared to the state-of-the-art (SOTA) algorithms. As shown in Fig. 3, DGC blocks are first employed to extract initial features from the skeleton sequence data. Then, AD-Mamba blocks are used to extract high-level features from the initial features. Finally, the SGF block is applied to fuse the high-level features with the initial features.

For the DGC operation, the learnable distance parameters are applied to adjust the aggregation weights of joints based on their spatial distances, allowing the model to capture the relationship between joints dynamically. As shown in Fig. 1, for the distance graph convolution, the hand-head aggregation weight increases and the hand-hand aggregation weight decreases in the ‘phone call’ action, whereas the hand-hand aggregation weight increases in the ‘put palms together’ action.

For the AD-Mamba structure, Mamba models perform forward and backward scans in different layers, achieving global scanning without increasing the computational effort. In the SGF block, we weight the initial feature based on the high-level feature and employ the residual connection, ensuring that the final feature for classification contains both global semantics and noteworthy details.

The main contributions of this paper are as follows.

- 1) We propose DGCmam, a model fusing distance graph convolution and Mamba for skeleton-based action recognition. DGCmam fuses high-level features and initial features by an adaptive gating mechanism, achieving state-of-the-art performance with low inference time.
- 2) We propose the Distance Graph Convolution (DGC) operation, where learnable distance parameters are used to adjust the aggregation weights based on the spatial distance dynamically. The DGC operation allows the model to focus on more relevant joints.
- 3) We propose an Alternating-Direction Mamba (AD-Mamba) structure, where Mamba models perform forward and backward scans in different layers. AD-Mamba enables the model to capture global correlations with less computational effort.

## II. RELATED WORKS

### A. Skeleton-based Action Recognition

Some early algorithms use convolutional neural networks (CNNs) or LSTMs [8] for skeleton-based action recognition by adapting skeleton data into image or sequence formats [9], [10]. Subsequently, algorithms using graph neural networks significantly improve recognition accuracy by modeling skeleton data as a spatio-temporal graph. ST-GCN [11] and ST-GCN++ [12] use the graph convolution network to extract features from spatio-temporal graphs constructed based on human joint connections. CTR-GCN [13] introduces a channel-wise topology refinement mechanism to perform channel-wise feature aggregation. Some algorithms then improve accuracy by adapting feature aggregation methods in graph convolution. BlockGCN [14] uses topological

encoding and block graph convolution to efficiently extract features. G3CN [15] applies a Gaussian filter with Gated Recurrent Units (GRUs) [16] to better distinguish ambiguous actions. Some algorithms focus on enhancing the representation of features. FR-Head [17] proposes an auxiliary feature refinement head to discriminate samples, and CSRE [18] applies the attentive features to contrastive learning for better exploiting latent data distributions. The aggregation weights in these algorithms are mostly static, and remain constant regardless of action classes, which limits the adaptability of these algorithms. In contrast, our proposed DGCmam uses dynamic aggregation weights by distance graph convolution to adapt to diverse actions.

Considering the limitations of static weights in graph convolutions, some algorithms use Transformers to dynamically extract features [19]. ST-TR [20] employs spatial and temporal Transformers to extract features from joints and frames. TranSkeleton [21] proposes a Transformer framework to unify the spatial and temporal modeling of skeleton sequences. HyperFormer [22] uses a hypergraph transformer to better model joint interactions. FreqMixFormerV2 [23] proposes a frequency-aware mixed transformer for identifying skeletal movements in the frequency domain. As the self-attention mechanism in Transformers makes a high computational effort, some algorithms employ the Mamba to extract features with linear computational complexity. SkeMamba [24] stacks several dual-path spatio-temporal Mamba blocks with gating mechanisms to extract features dynamically. TSkel-Mamba [25] proposes a hybrid Transformer-Mamba framework with temporal dynamic modeling blocks for modeling complex actions. Both SkeMamba and TSkel-Mamba extract features by the bidirectional Mamba structure, but the AD-Mamba block in DGCmam achieves a similar function with lower computational cost.

### B. Dataset

There are already many datasets used for skeleton-based action recognition [26], [27], [28], [29], [30]. The NTU RGB+D [26] and NTU RGB+D 120 [27] datasets are currently the most widely used datasets for skeleton-based action recognition. In these two datasets, each sample contains up to two humans, whose 3D coordinates of 25 joints are captured by three Kinect V2 cameras. The NTU RGB+D dataset contains 60 action classes and 56,880 video samples, and the NTU RGB+D 120 dataset contains 120 classes and 114,480 samples. In addition, the NW-UCLA dataset [28] is also used by some algorithms to evaluate the cross-view adaptation ability of models. The NW-UCLA dataset contains 1,400 samples across 10 action classes. Each sample contains one pedestrian, whose 3D coordinates of 20 joints are captured by three Kinect V1 cameras.

## III. METHODOLOGY

### A. Model Structure

The model structure of DGCmam is shown in Fig. 3. DGCmam mainly consists of three types of blocks: the DGC block, the AD-Mamba block, and the SGF block.

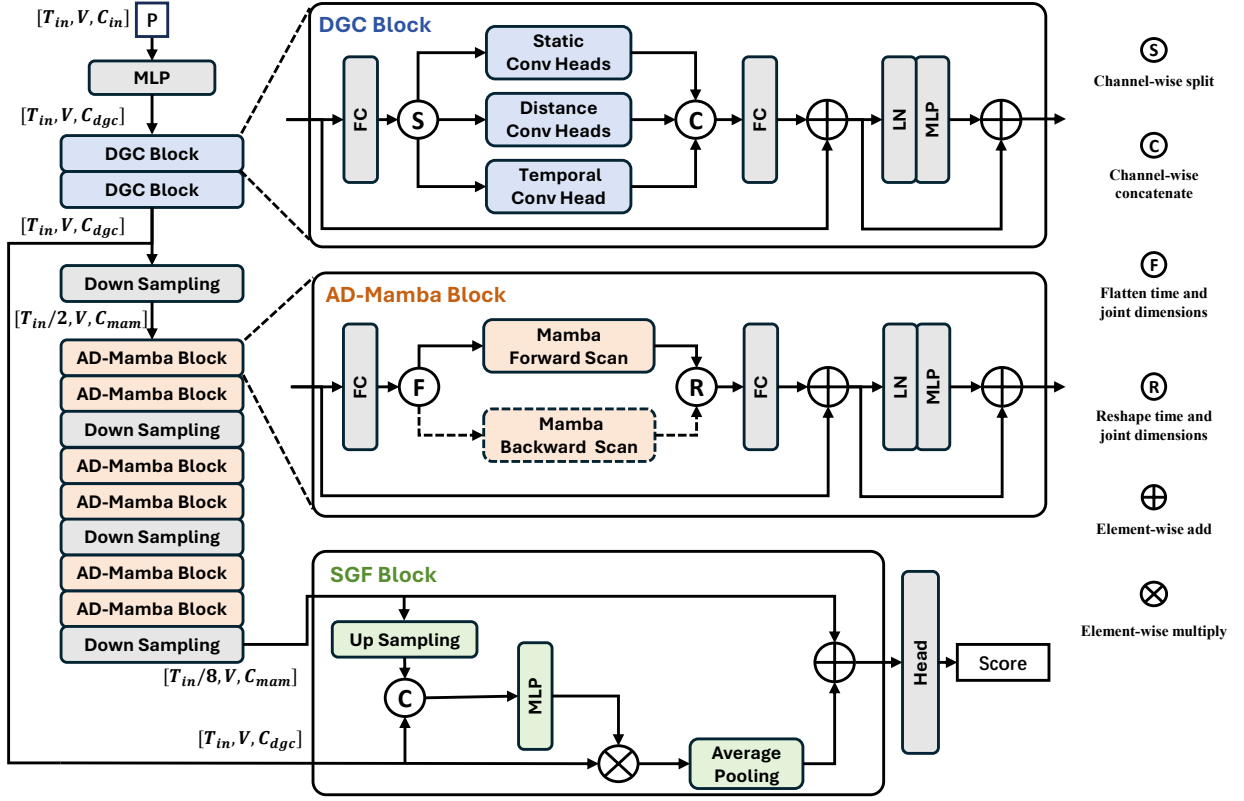


Fig. 3. The structure of the DGCmam. The DGC block dynamically aggregates initial features from joint features through distance graph convolution. The AD-Mamba block captures global high-level features with lower computational cost by alternating-direction scanning. The SGF block performs semantic-guided fusion to fuse semantic information from high-level features with details from initial features.

The input of DGCmam is the skeleton sequence data  $\mathbf{P} \in \mathbb{R}^{T_{in} \times V \times C_{in}}$ , where  $T_{in}$  denotes the number of frames,  $V$  denotes the number of joints in each frame,  $C_{in}$  denotes the input information dimension of each joint. DGCmam only uses joint position information as input, and  $\mathbf{P}^{t,i} \in \mathbb{R}^3$  denotes the 3D position coordinates of joint  $i$  at frame  $t$ .

First,  $\mathbf{P}$  is embedded through a Multi-Layer Perceptron (MLP) layer to create  $\mathbf{E} \in \mathbb{R}^{T_{in} \times V \times C_{dgc}}$ . Then, two stacked DGC blocks are used to aggregate spatial and temporal correlations of joints from  $\mathbf{E}$  to generate  $\mathbf{F}_{dgc} \in \mathbb{R}^{T_{in} \times V \times C_{dgc}}$ . Further,  $\mathbf{F}_{dgc}$  is fed into six stacked AD-Mamba blocks to extract high-level features  $\mathbf{F}_{mam} \in \mathbb{R}^{T_{out} \times V \times C_{mam}}$ . The downsampling convolution is applied between every two blocks. Finally, the features  $\mathbf{F}_{dgc}$  and  $\mathbf{F}_{mam}$  are fused by the SGF block to obtain  $\mathbf{F}_{class}$  for the classification head.

### B. DGC Block

The DGC (Distance Graph Convolution) block is used to extract local spatial features and temporal features dynamically. As shown in Fig. 3, in the DGC block, the input feature  $\mathbf{IC} \in \mathbb{R}^{T \times V \times C}$  first halves the channels by a fully connected (FC) layer, and splits along the channel into the static spatial feature  $\mathbf{XS} \in \mathbb{R}^{T \times V \times \frac{C}{8}}$ , the distance spatial feature  $\mathbf{XD} \in \mathbb{R}^{T \times V \times \frac{C}{8}}$ , and the temporal feature  $\mathbf{XT} \in \mathbb{R}^{T \times V \times \frac{C}{4}}$ . Then, three types of convolution heads are used to extract features from them. Finally, an MLP layer is applied to fuse the extracted features. The details of the three

types of convolution heads and feature fusion in the DGC block are presented as follows.

1) *Static Convolution Head*: We use multiple static convolution heads to capture common relationships and fixed dependencies of joints in all actions.  $\mathbf{XS}$  is divided into  $N$  segments along the channel according to the number of static convolution heads  $N$ . The  $k$ -th feature of joint  $i$  is denoted as  $\mathbf{XS}^{i,k} \in \mathbb{R}^{T \times 1 \times \frac{C}{8N}}$ .

The static convolution head uses a learnable adjacency matrix  $\mathbf{A}_{sch} \in \mathbb{R}^{V \times V}$  as the aggregation weights. For the  $k$ -th static convolution head, the aggregated feature  $\mathbf{YS}^{i,k} \in \mathbb{R}^{T \times 1 \times \frac{C}{8N}}$  for joint  $i$  is calculated as follows:

$$\mathbf{YS}^{i,k} = \sum_{j=1}^V \mathbf{A}_{sch}^{i,j,k} \cdot \mathbf{XS}^{j,k} \quad (1)$$

where  $\mathbf{A}_{sch}^{i,j,k} \in \mathbb{R}$  represents the learnable weights for joint  $j$  to joint  $i$  in the  $k$ -th static convolution head.

2) *Distance Convolution Head*: We use multiple distance convolution heads to capture unique joint relationships from different actions dynamically. The distance convolution head introduces a learnable distance decay function to adjust the aggregation weights between joints in different actions dynamically. A distance mask  $\mathbf{M} \in \mathbb{R}^{V \times V}$  is first calculated based on the distances of joint positions. The mask value  $\mathbf{M}^{i,j}$  for joint  $i$  to joint  $j$  is calculated using a Gaussian

decay function:

$$\mathbf{M}^{i,j} = \exp \left( - \frac{\sum_{t=1}^{T_{in}} \|\mathbf{P}^{t,i} - \mathbf{P}^{t,j}\|_2}{T_{in} \cdot 2\sigma^2} \right) \quad (2)$$

where  $\sigma$  is a learnable parameter for adjusting the decay rate;  $\|\cdot\|_2$  denotes the Euclidean norm. The adjacency matrix  $\mathbf{A}_{dch} \in \mathbb{R}^{V \times V}$  used for the distance convolution head is obtained by combining the distance mask  $\mathbf{M}$  and a learnable adjacency matrix  $\mathbf{A} \in \mathbb{R}^{V \times V}$  through the Hadamard product:

$$\mathbf{A}_{dch} = \mathbf{A} \odot \mathbf{M} \quad (3)$$

For the  $k$ -th distance convolution head, the aggregated feature  $\mathbf{YD}^{i,k}$  for joint  $i$  is calculated as follows:

$$\mathbf{YD}^{i,k} = \sum_{j=1}^V \mathbf{A}_{dch}^{i,j,k} \cdot \mathbf{XD}^{j,k} \quad (4)$$

where  $\mathbf{A}_{dch}^{i,j,k} \in \mathbb{R}$  represents the learnable weights for joint  $j$  to joint  $i$  in the  $k$ -th distance convolution head.

In the distance convolution head, the aggregation weights dynamically adjust depending on the action. For instance, in the action of raising a single hand, even if  $\mathbf{A}^{i,j}$  for both hands is high, the aggregation weight  $\mathbf{A}_{dch}^{i,j}$  becomes smaller due to the smaller  $\mathbf{M}^{i,j}$  caused by the greater distance between both hands.

3) *Temporal Convolution Head*: The temporal convolution head is applied to extract the motion and temporal features for each joint. Specifically, we apply the 1D convolution operation along the temporal axis of the temporal feature  $\mathbf{XT}$ . The aggregated temporal feature  $\mathbf{yt}_i^t$  for joint  $i$  at frame  $t$  is computed as follows:

$$\mathbf{YT}^{t,i} = \sum_{n=-R_{tc}}^{R_{tc}} \mathbf{W}^n \cdot \mathbf{XT}^{(t+n),i} + \mathbf{b} \quad (5)$$

where  $\mathbf{XT}^{t,i} \in \mathbb{R}^{1 \times 1 \times \frac{C}{4}}$  denotes the feature of joint  $i$  at frame  $t$  in  $\mathbf{XT}$ ,  $R_{tc}$  is a scalar related to the kernel size,  $\mathbf{W}^n$  denotes the learnable weight corresponding to the temporal offset  $n$ ,  $\mathbf{b}$  denotes the learnable bias.

4) *Spatial and Temporal Feature Fusion*: The spatial features  $\mathbf{YS}$ ,  $\mathbf{YD}$  and the temporal feature  $\mathbf{YT}$  are concatenated and fused through the FC layer and MLP layer with the dropout mechanism. The residual connection and the layer normalization are also applied. The output  $\mathbf{OC} \in \mathbb{R}^{T \times V \times C}$  of the DGC block is calculated as follows:

$$\mathbf{YC} = \mathbf{IC} + \phi_{yd}([\mathbf{YS}, \mathbf{YD}, \mathbf{YT}]) \quad (6)$$

$$\mathbf{OC} = \mathbf{YC} + \psi_{od}(\text{LN}(\mathbf{YC})) \quad (7)$$

where  $[\cdot, \cdot]$  denotes concatenating two features along the channel dimension;  $\phi_{yd}$  is an FC layer with input dimensions of  $C/2$  and output dimensions of  $C$ ;  $\psi_{od}$  denotes a three-layer MLP with input and output dimensions of  $C$ , hidden dimensions of  $2C$ , and GELU (Gaussian Error Linear Unit) activation functions;  $\text{LN}(\cdot)$  denotes layer normalization.

### C. AD-Mamba Block

Although the DGC block extracts local features effectively, it has a limited receptive field. Therefore, the AD-Mamba (Alternating-Direction Mamba) block is designed to efficiently capture global spatio-temporal dependencies without doubling the computational effort like the bidirectional Mamba. Specifically, we stack multiple AD-Mamba blocks. For the  $l$ -th block, if  $l$  is odd, the Mamba model scans the feature sequence from beginning to end (forward scan); otherwise, the Mamba model scans the sequence from end to beginning (backward scan).

As shown in Fig. 3, in each AD-Mamba block, the input feature  $\mathbf{IM} \in \mathbb{R}^{T \times V \times C}$  first halves the channels by an FC layer and flattens as  $\mathbf{XM} \in \mathbb{R}^{L \times \frac{C}{2}}$  along the time and joint dimensions, where  $L = T \cdot V$  represents the flattened sequence length. The scanning process in the  $l$ -th AD-Mamba block is shown as follows:

$$\mathbf{YM} = \begin{cases} \text{Mam}^l(\mathbf{XM}) & l \text{ is odd} \\ \text{Re}(\text{Mam}^l(\text{Re}(\mathbf{XM}))) & \text{otherwise} \end{cases} \quad (8)$$

where  $\text{Re}(\cdot)$  denotes the operation of reversing the feature sequence along the length dimension;  $\text{Mam}^l$  is the Mamba model for the  $l$ -th AD-Mamba block with input dimensions of  $C/2$ , state dimensions of  $C/6$ . Similar to the DGC block, the final output  $\mathbf{OM} \in \mathbb{R}^{T \times V \times C}$  of the AD-Mamba block is calculated through the following process:

$$\mathbf{ZM} = \mathbf{IM} + \phi_{ym}(\mathbf{YM}') \quad (9)$$

$$\mathbf{OM} = \mathbf{ZM} + \psi_{om}(\text{LN}(\mathbf{ZM})) \quad (10)$$

where  $\mathbf{YM}' \in \mathbb{R}^{T \times V \times \frac{C}{2}}$  denotes  $\mathbf{YM}$  expanded along the length dimension;  $\phi_{ym}$  is an FC layer with the same structure as  $\phi_{yd}$ ;  $\psi_{om}$  is a three-layer MLP with the same structure as  $\psi_{od}$ .

As each block only performs a single-direction scan, the AD-Mamba block can capture joint relationships globally with the same computational effort as the standard Mamba model.

### D. Down-sampling Convolution

The down-sampling convolution is applied to halve the feature's temporal dimension and adjust its channel dimension. Specifically, assuming the input features  $\mathbf{X} \in \mathbb{R}^{T \times V \times C}$ , the output  $\mathbf{Y} \in \mathbb{R}^{\frac{T}{2} \times V \times C'}$  of the down-sampling convolution is calculated as follows:

$$\mathbf{Y} = \text{BN}(\varphi_{dc}(\mathbf{X})) \quad (11)$$

where  $\varphi_{dc}$  denotes a 2D convolution network with kernel size of  $K_{dc} \times 1$ , stride of  $2 \times 1$ , input dimensions of  $C$ , output dimensions of  $C'$ ;  $\text{BN}(\cdot)$  denotes Batch Normalization.

### E. SGF Block and Classification

As shown in Fig. 3, the SGF (Semantic-Guided Fusion) block is used to introduce the action details from  $\mathbf{F}_{dgc}$  into the high-level semantic features  $\mathbf{F}_{mam}$ . Specifically, the SGF block combines  $\mathbf{F}_{mam}$  and  $\mathbf{F}_{dgc}$  to generate a mask that emphasizes valuable details in  $\mathbf{F}_{dgc}$ , and residually adds these details to  $\mathbf{F}_{mam}$ .

For aligning the temporal dimension, we first upsample  $\mathbf{F}_{mam}$  by the nearest-neighbor interpolation to obtain  $\tilde{\mathbf{F}}_{mam} \in \mathbb{R}^{T_{in} \times V \times C_{dgc}}$ . To determine the importance of the information contained in each frame of  $\mathbf{F}_{dgc}$ , an attention mask  $\mathbf{K} \in \mathbb{R}^{T_{in} \times V \times 1}$  is calculated as follows:

$$\mathbf{K} = \sigma(\psi_{cat}([\mathbf{F}_{dgc}, \tilde{\mathbf{F}}_{mam}])) \quad (12)$$

where  $\psi_{cat}$  denotes a three-layer MLP with input dimensions of  $C_{dgc} + C_{mam}$ , hidden dimensions of  $C_{dgc}/3$ , output dimensions of 1, GELU activation functions and 2D batch normalization;  $\sigma$  represents the sigmoid function.

Then, we multiply the mask  $\mathbf{K}$  with  $\mathbf{F}_{dgc}$  and apply average pooling in the temporal dimension to obtain the importance-weighted feature  $\mathbf{F}_{weight} \in \mathbb{R}^{T_{out} \times V \times C_{dgc}}$ .

$$\mathbf{F}_{weight} = \text{AvgPool}(\mathbf{F}_{dgc} \odot \mathbf{K}) \quad (13)$$

where  $\text{AvgPool}(\cdot)$  denotes the average pooling along the temporal dimension with a pooling size of  $T_{in}/T_{out}$ .

The feature  $\mathbf{F}_{class}$  that combines weighted detail information  $\mathbf{F}_{weight}$  with high-level feature  $\mathbf{F}_{mam}$  is calculated as follows:

$$\mathbf{F}_{class} = \mathbf{F}_{mam} + [\mathbf{F}_{weight}, \mathbf{0}] \quad (14)$$

where  $\mathbf{0} \in \mathbb{R}^{T_{out} \times V \times (C_{mam} - C_{dgc})}$  represents a zero matrix for aligning the channels. The feature vector  $\mathbf{v}_{class} \in \mathbb{R}^{C_{mam}}$  for classification is then obtained through average pooling of  $\mathbf{F}_{class}$ :

$$\mathbf{v}_{class} = \frac{1}{T_{out} \cdot V} \sum_{t=1}^{T_{out}} \sum_{i=1}^V \mathbf{F}_{class}^{t,i} \quad (15)$$

Finally,  $\mathbf{v}_{class}$  is fed into a linear classification head to generate the prediction scores  $\mathbf{s}_{class}$ . The final predicted action is determined by the index with the maximum value in  $\mathbf{s}_{class}$ .

## IV. EXPERIMENTS AND ANALYSIS

### A. Datasets

We compared DGCMam with SOTA algorithms on the NTU RGB+D, NTU RGB+D 120, and NW-UCLA datasets. In the NTU RGB+D and NTU RGB+D 120 datasets, we employ the same benchmarks as the majority of algorithms:

- 1) X-Sub: the benchmark across different subjects in the NTU RGB+D dataset.
- 2) X-View: the benchmark with different camera views in the NTU RGB+D dataset.
- 3) X-Sub120: the benchmark across different subjects in the NTU RGB+D 120 dataset.

- 4) X-Set120: the benchmark with different camera setups in the NTU RGB+D 120 dataset.

The dataset splits for each benchmark and NW-UCLA are shown in the corresponding papers. For fair comparisons, all algorithms only use joint positions as input.

### B. Implementation Details

For the NTU RGB+D and NTU RGB+D 120 datasets, each sample contains a maximum of two humans, and each human has 24 joints after removing the spine joint, hence  $V = 48$ . For the UCLA dataset, each sample contains one human with 20 joints, hence  $V = 20$ . The information for each joint is represented as 3D position coordinates, thus  $C_{in} = 3$ .

We use interpolation to transform skeleton sequences with different frame lengths into sequences with 64 frames, so  $T_{in} = 64$ . We use three downsampling blocks, hence  $T_{out} = 8$ . The channel dimensions are set as  $C_{dgc} = 96$ ,  $C_{mam} = 192$ . The convolution kernel size is set as  $R_{tc} = 3$ ,  $K_{dc} = 7$ . The number of static convolution heads and dynamic convolution heads is set as  $N = 4$ .

All the experiments are performed on a single RTX 5090 GPU. The model is trained by the AdamW optimizer [35] with a weight decay of 0.1. The gradient clipping is set to 1. The epoch number is set to 500, and the batch size is set to 128. The warm-up strategy and cosine annealing strategy are employed to adjust the learning rate during training. During the first 25 epochs, the learning rate increases linearly from  $10^{-7}$  to  $10^{-3}$ , and then, the learning rate gradually decreases to  $10^{-5}$ . We perform data augmentation during training, including rotation, rescaling, flipping, temporal cropping, coordinate dropout, and joint dropout of skeleton data [19]. The model is trained using the cross-entropy loss function with label smoothing.

### C. Comparisons with SOTA algorithms

Table I shows the top-1 accuracy of DGCMam and the SOTA algorithms for the three datasets. For the SOTA algorithms, we adopt the accuracy reported by their authors.

By combining distance graph convolution, alternating-direction Mamba scanning, and semantic-guided feature fusion, DGCMam achieves the highest top-1 accuracy among the SOTA algorithms in all the datasets. TSkel-Mamba, which also uses the Mamba model, achieves the second place in the X-Sub and X-Set120 benchmarks, where DGCMam's accuracy surpasses it by 1.2% and 0.6%, respectively. Ske-Mamba achieves the second place in the X-View benchmark with an accuracy of 96.1%, while DGCMam outperforms it by 0.7%. Also, DGCMam outperforms the second-place algorithm FreqMixFormerV2 by 0.4% and 0.8% in the accuracy for X-Sub120 and NW-UCLA, respectively.

Table II shows the model parameter numbers, FLOPs (Floating Point Operations), and inference time of the SOTA algorithms for the X-Sub benchmark. The results are measured using official code. We set the batch size to 1 and run 100 inferences at FP32 precision on the RTX 5090 and Ryzen 7 7700 to measure the inference time of each algorithm.

TABLE I

THE TOP-1 ACCURACY (%) OF THE SOTA ALGORITHMS. **BOLD** AND UNDERLINE MARK THE BEST AND SECOND-BEST RESULTS, RESPECTIVELY. RESULTS ARE CITED FROM ORIGINAL PAPERS UNLESS OTHERWISE NOTED. ‘-’ DENOTES NO AVAILABLE DATA.

| Algorithm            | Source                    | X-Sub                | X-View               | X-Sub120    | X-Set120             | NW-UCLA              |
|----------------------|---------------------------|----------------------|----------------------|-------------|----------------------|----------------------|
| CTR-GCN [13]         | ICCV, 2021                | 89.9                 | 94.8 <sup>[15]</sup> | 84.9        | 86.7 <sup>[15]</sup> | 94.6 <sup>[15]</sup> |
| ST-TR [20]           | ICPR, 2021                | 88.7                 | 95.6                 | 81.9        | 84.1                 | -                    |
| InfoGCN [31]         | CVPR, 2022                | 89.8 <sup>[15]</sup> | 95.2 <sup>[15]</sup> | 85.1        | 86.3                 | 94.0 <sup>[15]</sup> |
| HyperFormer [22]     | arXiv, 2022               | 90.7                 | 95.1                 | 86.6        | 88.0                 | 92.7 <sup>[18]</sup> |
| ST-GCN++ [12]        | ACMMM, 2022               | 89.3                 | 95.6                 | 83.2        | 85.6                 | -                    |
| FR-Head [17]         | CVPR, 2023                | 90.3                 | 95.3                 | 85.5        | 87.3                 | 94.4                 |
| TranSkeleton [21]    | TCSVT, 2023               | 90.1                 | 95.4                 | 84.9        | 86.3                 | -                    |
| BlockGCN [14]        | CVPR, 2024                | 90.9                 | 95.4                 | 86.9        | 88.2                 | 95.5 <sup>[18]</sup> |
| SiT-MLP [32]         | TCSVT, 2024               | 90.0                 | 95.0                 | 83.9        | 85.7                 | 95.7                 |
| STRN [33]            | TIM, 2025                 | 90.4                 | 95.4                 | 85.2        | 87.1                 | -                    |
| DAMD-GCN [34]        | Neurocomputing, 2025      | 89.5                 | 94.8                 | 84.7        | 86.3                 | -                    |
| SkeMamba [24]        | The Visual Computer, 2025 | 90.6                 | 96.1                 | 85.0        | 87.3                 | -                    |
| TSkel-Mamba [25]     | arXiv, 2025               | <u>91.4</u>          | <u>95.8</u>          | 87.4        | <u>88.7</u>          | -                    |
| FreqMixFormerV2 [23] | FG, 2025                  | 90.7                 | 95.4                 | <u>87.6</u> | 88.5                 | <u>96.2</u>          |
| G3CN [15]            | IROS, 2025                | 91.1                 | 95.3                 | 87.4        | 88.6                 | 94.2                 |
| CSRE [18]            | PR, 2025                  | 91.3                 | 95.6                 | 87.2        | <u>88.7</u>          | 95.9                 |
| DGCMam               | Ours                      | <b>92.6</b>          | <b>96.8</b>          | <b>88.0</b> | <b>89.3</b>          | <b>97.0</b>          |

[x] The result is reported by another paper [x].

TABLE II

THE MODEL PARAMETERS, FLOPS, AND INFERENCE TIME OF THE SOTA ALGORITHMS FOR THE X-SUB BENCHMARK. RESULTS ARE MEASURED USING OFFICIAL CODE UNLESS OTHERWISE NOTED. ‘-’ DENOTES NO AVAILABLE DATA.

| Algorithm       | Parameters | FLOPs | Inference Time |
|-----------------|------------|-------|----------------|
| CTR-GCN         | 1.4M       | 1.8G  | 8.1ms          |
| ST-TR           | 12.1M      | 64.5G | 8.7ms          |
| InfoGCN         | 1.5M       | 1.7G  | 6.9ms          |
| HyperFormer     | 2.6M       | 4.8G  | 8.6ms          |
| ST-GCN++        | 1.4M*      | 2.8G* | -              |
| FR-Head         | 1.4M       | 1.8G  | 8.9ms          |
| TranSkeleton    | 2.2M*      | 2.3G* | -              |
| BlockGCN        | 1.3M       | 1.6G  | 10.2ms         |
| SiT-MLP         | 0.6M       | 0.7G  | 2.9ms          |
| STRN            | 1.4M*      | 1.6G* | -              |
| DAMD-GCN        | 3.8M*      | 8.4G* | -              |
| SkeMamba        | -          | -     | -              |
| TSkel-Mamba     | 2.4M*      | 8.2G* | -              |
| FreqMixFormerV2 | 1.2M*      | -     | -              |
| G3CN            | 7.2M       | 9.3G  | 21.5ms         |
| CSRE            | 2.9M       | 4.8G  | 8.7ms          |
| DGCMam          | 3.3M       | 3.3G  | 3.8ms          |

\* The result is cited from the original paper.

Although the parameter number of DGCMam is relatively high among the SOTA algorithms, the FLOPs and inference time of DGCMam are acceptable compared to the G3CN and CSRE.

#### D. Ablation Study

We study the effect of DGC blocks, AD-Mamba blocks, and SGF blocks on algorithm performance through several base models and fusion methods. The experiment contains the following basic models:

- 1) SGC: All eight blocks are SGC blocks.
- 2) ADGC: All eight blocks are ADGC blocks.

- 3) DGC: All eight blocks are DGC blocks.
- 4) ADMam: All eight blocks are AD-Mamba blocks.
- 5) BDMam: All eight blocks are BD-Mamba blocks.
- 6) SGC/ADMam: First two blocks are SGC blocks, last six blocks are AD-Mamba blocks.
- 7) DGC/BDMam: First two blocks are DGC blocks, last six blocks are BD-Mamba blocks.
- 8) DGC/ADMam: Our proposed base model.

where the SGC block replaces all distance convolution heads in the DGC block with static convolution heads; the ADGC block replaces all static convolution heads in the DGC block with distance convolution heads; the BD-Mamba block replaces the alternating-direction scanning in the AD-Mamba block with bidirectional scanning and fuses features with an FC layer.

Moreover, the experiment contains the following feature fusion methods:

- 1) SGF: Our proposed feature fusion method.
- 2) APF: The average pooling feature fusion method, where  $\mathbf{F}_{class} = \mathbf{F}_{mam} + [\text{AvgPool}(\mathbf{F}_{dgc}), \mathbf{0}]$
- 3) No fusion: Let  $\mathbf{F}_{class} = \mathbf{F}_{mam}$ , represented as ‘-’ in the table.

Table III shows the experiment results of the ablation study. The results of SGC and DGC show that, without significantly increasing the parameters and FLOPs, distance graph convolutions can dynamically aggregate features, thereby achieving better performance than typical graph convolutions. However, the results of ADGC and DGC indicate that without static convolution heads, the common relationships and static dependencies of joints are difficult to capture, leading to limited performance improvement.

The results of ADMam and BDMam indicate that pure Mamba models show similar performance to pure convolution models, probably because of the poor ability to extract local features. The results of DGC/ADMam, DGC,

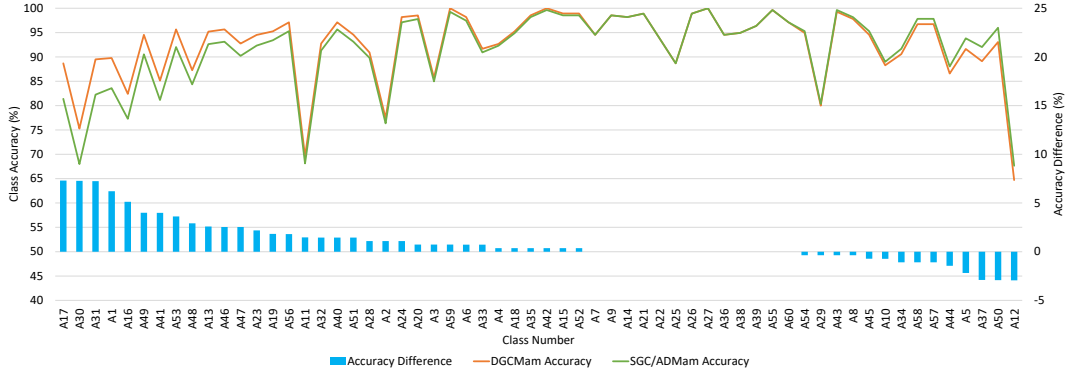


Fig. 4. The recognition accuracy and accuracy difference between DGC/Mam and SGC/ADMam for all classes in the X-Sub benchmark.

TABLE III

THE TOP-1 ACCURACY (%), MODEL PARAMETERS, AND FLOPs OF THE ALGORITHMS FOR THE X-SUB BENCHMARK. ‘-’ DENOTES WITHOUT ANY FUSION METHOD.

| Base Model | Fuse | Top-1 Accuracy | Parameters | FLOPs |
|------------|------|----------------|------------|-------|
| SGC        | -    | 91.5           | 3.01M      | 2.99G |
| ADGC       | -    | 91.6           | 3.01M      | 2.99G |
| DGC        | -    | 91.9           | 3.01M      | 2.99G |
| DGC        | APF  | 92.0           | 3.01M      | 2.99G |
| DGC        | SGF  | 92.0           | 3.02M      | 3.03G |
| BDMam      | -    | 91.5           | 3.86M      | 3.80G |
| ADMam      | -    | 91.4           | 3.29M      | 3.32G |
| ADMam      | APF  | 91.4           | 3.29M      | 3.32G |
| ADMam      | SGF  | 91.6           | 3.30M      | 3.35G |
| SGC/ADMam  | -    | 91.7           | 3.29M      | 3.24G |
| DGC/BDMam  | -    | 92.3           | 3.81M      | 3.60G |
| DGC/ADMam  | -    | 92.3           | 3.29M      | 3.24G |
| DGC/ADMam  | APF  | 92.4           | 3.29M      | 3.24G |
| DGC/ADMam  | SGF  | 92.6           | 3.30M      | 3.27G |

and ADMam indicate that the hybrid model combining graph convolutions with Mamba outperforms both pure graph convolutional models and pure Mamba models. However, using static graph convolutions or bidirectional Mamba in hybrid models does not improve the performance, as shown by the results of SGC/ADMam and DGC/BDMam.

The results from the same base model with different fusion methods indicate that for pure convolution models, both SGF and APF methods only slightly improve model performance. However, our proposed SGF method shows a more notable performance improvement compared to the APF method for the hybrid model.

The recognition accuracy and accuracy difference between DGC/Mam and SGC/ADMam for all classes in the NTU RGB+D dataset are shown in Fig. 4. Among all the 60 classes, there are 33 classes where DGC/Mam achieves a higher accuracy than SGC/ADMam, 13 classes where their accuracy is identical, and 14 classes where DGC/Mam exhibits a lower accuracy than SGC/ADMam. The accuracy of DGC/Mam exceeds that of SGC/ADMam most significantly in action A17 (take off a shoe), by 7.3%. DGC/Mam shows a higher recognition accuracy than SGC/ADMam in over

half of the action classes, reflecting that distance graph convolutions can effectively extract features.

The ablation study shows that distance graph convolution, alternating-direction Mamba, and the semantic-guided feature fusion method effectively improve model performance without significantly increasing the model parameters or computational effort.

#### E. Aggregation Weight Analysis

To further study the effectiveness of the DGC block and the SGF block, we present the feature aggregation weights of the trained DGC/Mam for some actions in the X-Sub benchmark. These actions are illustrated in Fig. 5, where ‘brush teeth’ and ‘phone call’ relate to the hand and head, ‘take off a shoe’ relates to the hand and foot, and ‘put palms together’ and ‘reading’ relate to the left and right hands.

##### 1) Distance Graph Convolution Feature Aggregation:

Table IV presents the Average Aggregation Ratio (AAR) for different types of convolution heads between different joints in some actions. For instance, for the ‘brush teeth’ action, the AAR between the left and right hands of the distance convolution head is calculated as follows:

$$AAR_{dch}^{LH,RH} = \frac{1}{S \cdot 2N} \sum_{k=1}^S \sum_{k=1}^N (|A_{dch}^{LH,RH,k}| + |A_{dch}^{RH,LH,k}|) \quad (16)$$

where  $LH$  and  $RH$  denote the indices for the left and right hands, respectively;  $S$  represents the number of samples with label ‘brush teeth’. The larger the  $AAR$ , the more features the convolution head aggregates from the two joints.

In Table IV, the AARs of the static convolution heads are invariant among different actions, indicating that they aggregate the same proportion of features from the two joints regardless of the action feature. This aggregation method limits the dynamic feature extraction ability of the static convolution head.

Meanwhile, for different joint pairs, the highest AARs of the convolution head belong to different actions. For the pair of left and right hand, the highest AAR occurs in ‘reading’ (0.012), while for the pair of right hand and head, the highest



Fig. 5. Illustration of samples of the five action classes in the X-Sub benchmark.

TABLE IV

THE AVERAGE AGGREGATION RATIOS OF THE DISTANCE CONVOLUTION HEAD (DCH) AND STATIC CONVOLUTION HEAD (SCH) OF DGCMAM FOR SOME ACTIONS IN THE X-SUB BENCHMARK. **BOLD** AND UNDERLINE MARK THE HIGHEST AND SECOND-HIGHEST VALUES OF DCH FOR DIFFERENT ACTIONS, RESPECTIVELY.

| Action             | Right Hand - Left Hand |       | Right Hand - Head |       | Right Hand - Right Foot |       |
|--------------------|------------------------|-------|-------------------|-------|-------------------------|-------|
|                    | DCH                    | SCH   | DCH               | SCH   | DCH                     | SCH   |
| Brush teeth        | 0.005                  | 0.011 | <b>0.038</b>      | 0.039 | 0.001                   | 0.021 |
| Phone call         | 0.006                  | 0.011 | <u>0.034</u>      | 0.039 | 0.003                   | 0.021 |
| Take off a shoe    | 0.010                  | 0.011 | <u>0.026</u>      | 0.039 | <b>0.012</b>            | 0.021 |
| Put palms together | <u>0.011</u>           | 0.011 | 0.019             | 0.039 | <u>0.004</u>            | 0.021 |
| Reading            | <b>0.012</b>           | 0.011 | 0.025             | 0.039 | 0.002                   | 0.021 |

AAR occurs in ‘brush teeth’ (0.038). For the pair of right hand and right foot, apart from ‘take off a shoe’ (0.012), the AARs for the other actions are very low.

When processing actions with significant hand-hand correlation, distance convolution heads exhibit higher hand-hand feature aggregation weights and lower hand-head and hand-foot feature aggregation weights. Moreover, when processing actions with significant hand-head correlation, the hand-head feature aggregation weights in distance convolution heads are enhanced. Results show that the distance convolution head can dynamically focus on and aggregate joint features based on action patterns, thereby achieving flexible feature aggregation.

2) *Semantic-guided Feature Fusion*: We study the adaptability of the SGF block by calculating the Average Gating Ratio (AGR) of different joints for some actions. For action  $l$  and joint  $i$ , AGR is calculated as follows:

$$AGR_l^i = \frac{1}{S_l \cdot T_{in}} \sum_{t=1}^{S_l} \sum_{i=1}^{T_{in}} (K^{t,i}) \quad (17)$$

where  $S_l$  denotes the number of samples labeled as action  $l$ . Table V presents the AGR of different joints in some actions.

In every action in Table V, the hands have larger AGRs than the foot and head, indicating that the SGF block tends to seek details of hand movements from the initial features. This may be because hand movements are more fine-grained. In addition, the proportion of feature fusion is different for actions. For actions ‘brush teeth’ and ‘put palms together’, about 10% of the initial features are fused, whereas for the action ‘take off a shoe’, over 40% of the features are fused. The different AGR values indicate that the SGF block can

TABLE V

THE AVERAGE GATING RATIOS OF THE SGF BLOCK OF DGCMAM FOR SOME ACTIONS IN THE X-SUB BENCHMARK.

| Action             | Left Hand | Right Hand | Head  | Right Foot |
|--------------------|-----------|------------|-------|------------|
| Brush teeth        | 0.096     | 0.107      | 0.090 | 0.068      |
| Phone call         | 0.100     | 0.086      | 0.058 | 0.058      |
| Take off a shoe    | 0.517     | 0.460      | 0.405 | 0.370      |
| Put palms together | 0.122     | 0.099      | 0.090 | 0.069      |
| Reading            | 0.194     | 0.155      | 0.054 | 0.029      |

adaptively extract features from joints guided by high-level semantics in various actions.

## V. CONCLUSION

In this paper, we propose DGCMam, a model fusing distance graph convolution and Mamba for skeleton-based action recognition. To fuse joint features dynamically according to motion patterns, we propose distance graph convolution, which combines a learnable decay function to calculate fusion weights according to the spatial distance. In addition, we propose an alternating-direction Mamba structure to capture joint features globally with lower computational effort. A semantically guided feature fusion module is also applied to fuse the action details from initial features with the semantic information from high-level features by an adaptive gating mechanism. Experiments on the NTU RGB+D, NTU RGB+D 120, and NW-UCLA datasets show that DGCMam achieves the highest top-1 accuracy among the SOTA algorithms with a lower inference time.

## REFERENCES

- [1] N. Noor and I. K. Park. A lightweight skeleton-based 3d-cnn for real-time fall detection and action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2179–2188, 2023.
- [2] S. Naimi, W. Bouachir, G.-A. Bilodeau, and B. Mishra. Sstar: Skeleton-based spatio-temporal action recognition for intelligent video surveillance and suicide prevention in metro stations. In *Proceedings of the Winter Conference on Applications of Computer Vision*, pages 813–823, 2025.
- [3] J. Long and M. Matteucci. Variable velocity dynamic window approach with learning-based warning policy for environments with non-cooperators. In *International Conference on Intelligent Computing*, pages 113–124. Springer, 2025.
- [4] M. Deyzel and R. P. Theart. One-shot skeleton-based action recognition on strength and conditioning exercises. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5169–5178, 2023.
- [5] C. Wang and J. Yan. A comprehensive survey of rgb-based and skeleton-based human action recognition. *IEEE Access*, 11:53880–53898, 2023.
- [6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [7] A. Gu and T. Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.
- [8] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [9] Y. Du, Y. Fu, and L. Wang. Skeleton based action recognition with convolutional neural network. In *2015 3rd IAPR Asian conference on pattern recognition (ACPR)*, pages 579–583. IEEE, 2015.
- [10] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1110–1118, 2015.
- [11] S. Yan, Y. Xiong, and D. Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.
- [12] H. Duan, J. Wang, K. Chen, and D. Lin. Pyskl: Towards good practices for skeleton action recognition. In *Proceedings of the 30th ACM international conference on multimedia*, pages 7351–7354, 2022.
- [13] Y. Chen, Z. Zhang, C. Yuan, B. Li, Y. Deng, and W. Hu. Channel-wise topology refinement graph convolution for skeleton-based action recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13359–13368, 2021.
- [14] Y. Zhou, X. Yan, Z.-Q. Cheng, Y. Yan, Q. Dai, and X.-S. Hua. Blockgcn: Redefine topology awareness for skeleton-based action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2049–2058, 2024.
- [15] H. Ren, Z. Luo, H. Fan, X. Yuan, G. Wang, and L. Zhang. G3cn: Gaussian topology refinement gated graph convolutional network for skeleton-based action recognition. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9661–9668. IEEE, 2025.
- [16] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [17] H. Zhou, Q. Liu, and Y. Wang. Learning discriminative representations for skeleton based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10608–10617, 2023.
- [18] S. Zhang, J. Yin, and Y. Dang. A generically contrastive spatiotemporal representation enhancement for 3d skeleton action recognition. *Pattern Recognition*, 164:111521, 2025.
- [19] J. Do and M. Kim. Skateformer: skeletal-temporal transformer for human action recognition. In *European Conference on Computer Vision*, pages 401–420. Springer, 2024.
- [20] C. Plizzari, M. Cannici, and M. Matteucci. Spatial temporal transformer network for skeleton-based action recognition. In *International conference on pattern recognition*, pages 694–701. Springer, 2021.
- [21] H. Liu, Y. Liu, Y. Chen, C. Yuan, B. Li, and W. Hu. Transkeleton: Hierarchical spatial-temporal transformer for skeleton-based action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(8):4137–4148, 2023.
- [22] Y. Zhou, Z.-Q. Cheng, C. Li, Y. Fang, Y. Geng, X. Xie, and M. Keuper. Hypergraph transformer for skeleton-based action recognition. *arXiv preprint arXiv:2211.09590*, 2022.
- [23] W. Wu, P. Wang, C. Chen, and A. Lu. Freqmixformerv2: Lightweight frequency-aware mixed transformer for human skeleton action recognition. In *2025 IEEE 19th International Conference on Automatic Face and Gesture Recognition (FG)*, pages 1–5. IEEE, 2025.
- [24] J. Zhao, J. Dai, F. Zhou, J. Pan, and H. Xu. Dual-path spatio-temporal mamba for skeleton-based action recognition. *The Visual Computer*, pages 1–13, 2025.
- [25] Y. Liu, J. Liu, H. Zhang, D. Xu, H. Rahmani, M. Bennamoun, and Q. Ke. Tskel-mamba: Temporal dynamic modeling via state space model for human skeleton-based action recognition. *arXiv preprint arXiv:2512.11503*, 2025.
- [26] A. Shahroudy, J. Liu, T.-T. Ng, and G. 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.
- [27] J. Liu, A. Shahroudy, M. Perez, G. Wang, L.-Y. Duan, and A. C. Kot. Ntu rgb+d 120: A large-scale benchmark for 3d human activity understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2684–2701, 2019.
- [28] J. Wang, X. Nie, Y. Xia, Y. Wu, and S.-C. Zhu. Cross-view action modeling, learning and recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2649–2656, 2014.
- [29] T. Li, J. Liu, W. Zhang, Y. Ni, W. Wang, and Z. Li. Uav-human: A large benchmark for human behavior understanding with unmanned aerial vehicles. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16266–16275, 2021.
- [30] H. Quan, Y. Hu, and A. Bonarini. Polimi-itw-s: A large-scale dataset for human activity recognition in the wild. *Data in brief*, 43:108420, 2022.
- [31] H.-g. Chi, M. H. Ha, S. Chi, S. W. Lee, Q. Huang, and K. Ramani. Infogcn: Representation learning for human skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20186–20196, 2022.
- [32] S. Zhang, J. Yin, Y. Dang, and J. Fu. Sit-mlp: A simple mlp with point-wise topology feature learning for skeleton-based action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 34(9):8122–8134, 2024.
- [33] R. Wang, J. Jin, Z. Chen, C. Wu, X.-J. Wu, and N. Sebe. Structural topology refinement network for skeleton-based action recognition. *IEEE Transactions on Instrumentation and Measurement*, 2025.
- [34] J. Li, P. Ye, Y. Xia, Y. Wang, and Y. Cao. Disentangled adaptive multi-dimensional dynamic graph convolutional network for skeleton-based action recognition. *Neurocomputing*, page 131693, 2025.
- [35] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.