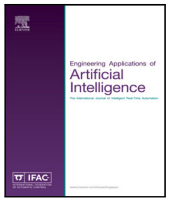
Engineering Applications of Artificial Intelligence 126 (2023) 106855

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Feature reconstruction graph convolutional network for skeleton-based action recognition

Junhao Huang [a], Ziming Wang [a,b], Jian Peng [a,*], Feihu Huang [a,*]

[a] College of Computer Science, Sichuan University, Chengdu, 610065, China
[b] Information Management Department, West China Second Hospital of Sichuan University, Chengdu, 610066, China

ARTICLE INFO

Keywords:
Action recognition
Human skeleton
Graph convolutional network
Feature reconstruction
Partition enhancement

ABSTRACT

Skeleton-based action recognition is an important task in computer vision. Recently, graph convolutional networks (GCNs) have been successfully applied to this task and achieved remarkable results. However, there are still some non-negligible limitations with existing GCN-based methods. First, the artificial predefined skeleton partition lacks the joint modeling for different types of edges. Second, most GCN models use interleaved deployment of spatial-only and temporal-only modules to achieve feature learning, which makes them ineffective in capturing spatiotemporal co-occurrence from action sequences. To tackle the above issues, we propose a novel feature reconstruction graph convolutional network (FR-GCN) for skeleton-based action recognition. The proposed FR-GC combines coarse-grained temporal and spatial features to reconstruct fine-grained spatiotemporal features, realizing simultaneous learning of temporal and spatial representations in a single module and significantly improving the capability of the model for spatiotemporal feature extraction. We also propose a topology partition enhancement module to achieve adaptive complementation among different types of edges. Moreover, an efficient multi-scale dual-domain temporal convolution is used to complete further temporal modeling. Compared with state-of-the-art methods, the proposed FR-GCN achieves competitive results on both NTU RGB+D 60 dataset and NTU RGB+D 120 dataset. Especially under the cross-subject benchmarks of the two datasets, the proposed FR-GCN achieves new state-of-the-art performance.

1. Introduction

Human action recognition is a long-standing problem in neural networks and learning systems, and it plays a fundamental and important role in computer vision. In the past decades, human action recognition has developed rapidly. It has a wide range of application scenarios, such as video understanding, virtual reality, and motion analysis (Sagayam and Hemanth, 2017; Wang et al., 2019; Ma et al., 2017). According to the types of input data, human action recognition methods can be grouped into three categories, which are RGB-based, depth-based, and skeleton-based methods. These methods are complementary to each other. RGB-based and depth-based methods (Rahmani and Bennamoun, 2017; Wang et al., 2015; Tran et al., 2015; Li et al., 2020) provide rich texture information, but they are sensitive to illumination transformation, background clutter, and other interferences (Chen et al., 2017). Compared with RGB-based and depth-based methods, skeleton-based methods have many advantages. (1) Skeleton data is compact data, which makes skeleton-based methods computationally more efficient. (2) Skeleton data is invariant to viewpoint and appearance. In addition, with the availability of cost-effective depth sensors (Zhang, 2012) and the advancement of human pose estimation algorithms (Shotton et al., 2011; Cao et al., 2017), skeleton data can be easily obtained. Based on the above advantages, skeleton-based action recognition has gained popularity and is increasingly receiving attention from researchers (Wang and Wang, 2017; Yan et al., 2018; Shi et al., 2019b; Liu et al., 2020; Chen et al., 2021a; Ding et al., 2022).

Skeleton-based action recognition is challenging due to scale variations, viewpoint changes, velocity fluctuations, etc. To capture crucial information from skeleton data, many efforts have been made in previous works. There are three development stages of skeleton-based action recognition. Initially, skeleton-based action recognition methods focus on designing handcrafted features to describe the motion patterns (Yang and Tian, 2012; Hussein et al., 2013; Vemulapalli et al., 2014; Koniusz et al., 2016). However, these methods are not satisfactory enough in migration and generalization, and require sophisticated feature engineering. To address the above limitations, skeleton-based action recognition methods begin to rely on deep learning: recurrent neural networks (RNNs) and convolutional neural networks (CNNs). RNN-based methods (Wang and Wang, 2017; Liu et al., 2016; Zhang

* Corresponding authors.
E-mail addresses: ddoubest@stu.scu.edu.cn (J. Huang), jianpeng@scu.edu.cn (J. Peng), huangfh@scu.edu.cn (F. Huang).

https://doi.org/10.1016/j.engappai.2023.106855
Received 12 October 2022; Received in revised form 20 March 2023; Accepted 21 July 2023
Available online 24 August 2023

et al., 2017; Li et al., 2018a) describe the skeleton sequence as a series of joint coordinate vectors, and CNN-based methods (Ke et al., 2017; Xie et al., 2018; Li et al., 2018b) construct the skeleton sequence as a pseudo-image. However, it is still difficult for them to fully represent the intrinsic connections between the joints. Recently, graph convolutional networks (GCNs) (Ying et al., 2018; Nguyen and Grishman, 2018; Guo et al., 2019) have extended convolution from images to graphs. Since graphs can effectively represent the complex relationships between skeleton joints, action recognition based on GCNs is gradually becoming the prevailing research. Yan et al. (2018) apply GCNs to skeleton-based action recognition for the first time, and their model (ST-GCN) achieves a milestone success.

Most existing GCN-based methods (Yan et al., 2018; Shi et al., 2019b; Chen et al., 2021a; Ye et al., 2020; Song et al., 2022) employ interleaved deployment of spatial-only and temporal-only modules to achieve the learning of spatiotemporal features. However, this decoupled approach hinders direct information flow across space and time. For example, the action of "kicking ball" can be divided into two phases: "running" and "swing leg". If there is no direct spatiotemporal interaction between "swing leg" and "running", the action may be simply recognized as "swing leg". Suppose the recognition model pays too much attention to the long-time information and ignores the changes in the instantaneous spatial configurations. In that case, the model might recognize the "running" action that occupies a long time in the whole movement as the main action. Therefore, exploiting the co-occurrence relationship between time and space is crucial to improve recognition performance.

To capture spatiotemporal features more efficiently without relying on the interleaved deployment, the methods that explicitly extract spatiotemporal features are expected. Si et al. (2019) integrate GCNs into long short-term memory networks (LSTMs) for learning spatiotemporal features and achieve good results. But their model is computationally inefficient and has the risk of gradient explosion. Liu et al. (2020) extend the joint edges inside frames to outside frames and use sliding time windows of size τ to achieve cross-frame learning. Their proposed 3D-GCN achieves impressive results on several datasets. However, their work also has some non-negligible issues: (1) The receptive field is directly related to the size τ, and the subjectively specified τ may not be valid for all action recognition. (2) Their model has huge parameters and complex computations. In conclusion, the capability and efficiency of existing methods to learn high-level spatiotemporal features need to be further improved.

A reasonable topology partition of the skeleton graph is essential to improve model performance. However, it has not been sufficiently explored in existing researches, and even largely ignored. Most existing works (Shi et al., 2019b; Ye et al., 2020; Li et al., 2019; Plizzari et al., 2021) directly adopt the partition strategy proposed in ST-GCN, which partitions the edges of the initial graph into three groups: root group, centripetal group and centrifugal group. Although the ST-GCN partition strategy is proven to be effective through experiments, it still has the following problems: (1) The root group only considers the effect of the self-loop edges and ignores the co-dependencies between joints. (2) The centripetal group and centrifugal group consider the action recognition as the independent effects of centripetal and centrifugal edges. This pattern leads to inaccurate recognition for the actions that are strongly correlated with the co-effect of the centripetal and centrifugal edges. For example, "rub two hands together" is an action with strongly correlated centripetal and centrifugal edges, in which the left hand does the centripetal motion while the right hand does the centrifugal motion, and vice versa. For such actions, the ST-GCN partition strategy cannot provide effective contribution to feature extraction.

To address the above problems, some efforts are made in our model to achieve high performance for skeleton-based action recognition while minimizing unnecessary training parameters. Firstly, we propose a plug-and-play topology partition enhancement (TPE) module, which aims to enhance topology partitions from the perspective of the original graph decomposition. Our TPE module can enrich the topology information of existing partitions and achieve more reasonable topology extension according to different actions. Taking the partition strategy of ST-GCN as an example, the proposed TPE module can achieve the joint interaction between the self-loop edges and other edges and the adaptive complementation of the centripetal and centrifugal edges without introducing additional edge noise, overcoming the deficiencies of the ST-GCN partition strategy in recognizing complex actions.

Secondly, we propose a feature reconstruction graph convolution (FR-GC), which can efficiently extract high-level spatiotemporal features. Our FR-GC not only considers feature extraction on single-frame spatial domain, but also extracts features on single-node temporal domain. The proposed FR-GC fuses the coarsened temporal features and the coarsened spatial features to refactor out the new spatiotemporal features, so that the refined graph convolution can be performed on the spatiotemporal domain. Based on the feature reconstruction and the feature fusion, our FR-GC can significantly improve the feature extraction capability of graph convolutions in the spatiotemporal domain. The feature extraction capability here refers to the ability to extract features from the original skeleton sequence that are more conducive to action recognition. Generally, the stronger the feature extraction capability, the higher the final recognition accuracy. It is worth noting that our FR-GC mainly improves the capability to transform raw data features into spatiotemporal joint features instead of singular temporal or spatial features, which are more helpful to the classifier for action classification.

Thirdly, we propose a multi-scale dual-domain temporal convolution (MS-DTC), which aims to solve the problem that the existing temporal convolutions are too sophisticated and inefficient. The proposed MS-DTC adopts multi-scale architecture, in which the core component is the dual-domain temporal convolution (DTC). Our DTC is designed according to the "hourglass" structure, in which both the upper and lower layers are depth-wise separable convolutions (Howard et al., 2017). The upper and lower layers capture the temporal context in the low-frequency domain and the high-frequency domain, respectively. And the combination of the two layers can effectively expand the receptive field of the module in the temporal domain. Our DTC outperforms the vanilla temporal convolution (Yan et al., 2018) in both parameters and computation cost, and has more powerful temporal modeling capability.

Combining the TPE module, FR-GC module, and MS-DTC module, we construct a powerful and efficient graph convolutional network named feature reconstruction graph convolutional network (FR-GCN) for skeleton-based action recognition. The overall framework of our FR-GCN is shown in Fig. 1. To verify the superiority of the proposed FR-GCN, we conduct extensive experiments on two large-scale skeleton-based action recognition datasets, NTU RGB+D 60 (Shahroudy et al., 2016) and NTU RGB+D 120 (Liu et al., 2019a). The experimental results show that: (1) Our FR-GCN significantly outperforms other explicit methods for spatiotemporal feature extraction with comparable parameters and computation cost. (2) Our FR-GCN outperforms almost all state-of-the-art methods on two challenging benchmark datasets.

The main contributions of this work are summarized as follows:

- We propose a plug-and-play topology partition enhancement module that dynamically enhances the topology representation for partition strategies without introducing additional edge noise. To the best of our knowledge, this is the first work for the adaptive enhancement of topology partitions from the graph decomposition perspective.
- We propose a feature reconstruction graph convolution (FR-GC), which reconstructs spatiotemporal features and completes the graph convolution in the spatiotemporal domain, leading to flexible and effective spatiotemporal modeling.

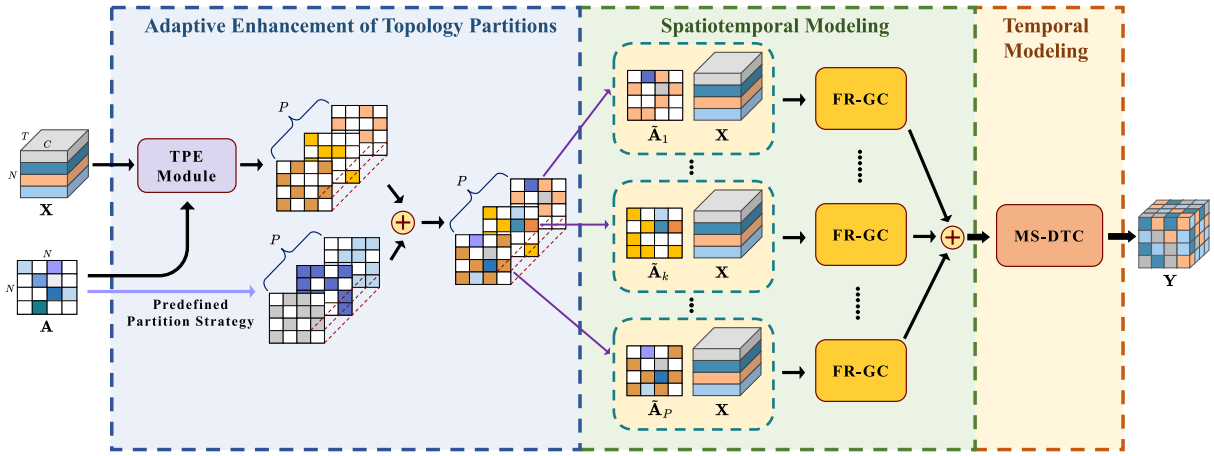


Fig. 1. Overall framework of the proposed feature reconstruction graph convolutional network. The adaptive enhancement of topology partitions integrates the new partitions computed by our TPE module and the predefined partitions to obtain the actual input topologies. The spatiotemporal modeling conducts the refined spatiotemporal graph convolution by our FR-GC. The temporal modeling aims to expand the receptive field of the model in the temporal domain, further capturing the extended temporal context.

- We propose a multi-scale dual-domain temporal convolution (MS-DTC), which efficiently learns temporal features in the low-frequency and high-frequency temporal domains, further expanding the receptive field.
- The extensive experimental results demonstrate the benefits of the partition enhancement and the reconstruction of spatiotemporal features. The proposed FR-GCN achieves remarkable results on two public datasets and outperforms all state-of-the-art methods on the cross-subject benchmarks.

2. Related work

2.1. Graph convolutional networks

Deep learning methods (CNNs and RNNs) can effectively capture the high-level features implied by Euclidean data and have achieved remarkable results in processing regular data such as images, videos, and audios. However, more common and general data in the real world are non-Euclidean data like graphs, such as social networks, molecular networks, and knowledge graphs. In order to extend deep learning to non-Euclidean spaces, there is a growing interest in designing and developing geometric deep learning (Bronstein et al., 2017), and the most prevailing solution is graph convolutional networks (GCNs). GCNs can be categorized as spectral-based GCNs and spatial-based GCNs. The spectral-based GCNs (Bruna et al., 2013; Henaff et al., 2015; Defferrard et al., 2016) use the eigenvalues and eigenvectors of the graph Laplacian matrix to generate the spectral representation of the graph. The spectral-based GCNs rely on the eigen-decomposition of the graph Laplacian matrix, which causes the computation to be heavy. In contrast to the spectral-based GCNs, the spatial-based GCNs (Duvenaud et al., 2015; Niepert et al., 2016; Hamilton et al., 2017; Veličković et al., 2018) define the graph convolution directly on the topology and update the central node features by aggregating its neighbor information. Since the spatial-based GCNs can propagate information directly over the topology, their weights can be easily shared over different structures and are more extensible. The GCN proposed by Kipf and Welling (2017) is widely used in various fields because of its effectiveness and simplicity, and their GCN can be analyzed from both the spectral perspective and the spatial perspective, completing the transition from the spectral domain to the spatial domain. The feature learning in Kipf and Welling (2017) can be summarized as the following two steps: (1) feature transformation of nodes to convert primary features to high-level features and (2) information propagation based on the topology to complete the learning of the central node from its neighbor nodes in a normalized approach. The same feature update rule is used in our work.

2.2. Skeleton-based action recognition

Skeleton-based action recognition receives widespread attention due to its efficient and simple representation of the human skeleton. Traditional skeleton-based action recognition methods (Yang and Tian, 2012; Hussein et al., 2013; Vemulapalli et al., 2014; Koniusz et al., 2016) focus on designing handcrafted features based on physical intuition, such as joint angles, joint orientations, positional differences, and kinematic features. Vemulapalli et al. (2014) encode the human skeleton using the rotations and translations in the Lie group, representing the human skeleton as a point in the Lie group, and each human action corresponds to a unique temporal evolution of such a point. Koniusz et al. (2016) utilize the tensors to construct high-level joint representations on the human skeleton. They apply two different radial basis function (RBF) kernels to capture the spatiotemporal compatibility and the dynamics of action sequences, respectively. Finally, they train a support vector machine (SVM) on the feature maps for action classification. However, designing well-performing handcrafted features is time-consuming and requires extensive field experience. More importantly, handcrafted features cannot take into account sufficient factors relevant to recognition. Therefore, the handcrafted methods have sub-optimal recognition performance.

As deep learning comprehensively outperforms traditional machine learning for most tasks in computer vision and natural language processing, data-driven methods are beginning to be used in skeleton-based action recognition. RNNs are effective models for processing sequence data. RNN-based methods (Wang and Wang, 2017; Liu et al., 2016; Zhang et al., 2017; Li et al., 2018a) for skeleton-based action recognition usually concatenate all joint feature vectors of each frame in some order and then extract the frame-level temporal dynamics using RNNs (e.g., LSTMs). Wang and Wang (2017) design a two-stream RNN model to recognize human actions. One of their streams is similar to the common methods, i.e., processing the feature vectors frame by frame, while the other stream interchanges the semantics of "joint" and "frame" in the RNN, and finally fuses the features of the two streams for action recognition. Zhang et al. (2017) propose a view-adaptive RNN model, in which they employ two LSTMs to obtain the optimal rotation parameters and optimal translation parameters for the original skeleton coordinates. Then, they adjust the skeleton sequence to a more consistent observation view, and finally stack multiple layers of LSTMs for action recognition. CNNs are a universal choice for processing regular mesh data, and can learn high-level semantic cues more conveniently than RNNs. CNN-based methods (Ke et al., 2017; Xie et al., 2018; Li et al., 2018b) for skeleton-based action recognition generally encode the skeleton data as pseudo-images and then utilize popular networks

(e.g., ResNet He et al., 2016) to explore the spatial and temporal dynamics. Li et al. (2018b) propose an end-to-end co-occurrence feature learning framework that considers not only "coordinates" as the feature channel, but also "joints" at the same time. Finally, they stack multiple convolution layers to achieve high-level feature extraction.

Due to the inherent limitations, neither RNN-based methods nor CNN-based methods can effectively represent the natural connections between the joints. The topological relationships of human joints naturally form graphs, and GCNs can efficiently process graph-structured data. To better capture the high-level semantics represented by the human skeleton, GCN-based methods (Yan et al., 2018; Shi et al., 2019b; Liu et al., 2020; Chen et al., 2021a; Ye et al., 2020; Song et al., 2022; Si et al., 2019; Cheng et al., 2020b) are gradually becoming the mainstream for skeleton-based action recognition. GCN-based methods can be categorized into implicit spatiotemporal methods and explicit spatiotemporal methods according to the extraction patterns for spatiotemporal features. For **implicit spatiotemporal methods**, these methods commonly employ interleaved deployment of spatial-only and temporal-only modules to achieve the extraction of the spatiotemporal semantics. Yan et al. (2018) predefine graphs based on human body structure and utilize GCN to model spatial configurations for the first time. Shi et al. (2019b) adaptively build dependencies between the joints to supplement the prior topology based on the contextual information. Chen et al. (2021a) propose a multi-scale graph convolution that effectively expands the spatial and temporal receptive fields. Wen et al. (2022) propose a motif-based graph convolution method, which utilizes latent relations among non-physically connected joints to impose high-order locality and combines local and nonlocal temporal blocks to complete action recognition. Chi et al. (2022) design an information bottleneck-based learning goal to guide the model to learn rich and compact potential representations. And an attention-based graph convolution is introduced in the model to capture the contextually relevant intrinsic topology. For **explicit spatiotemporal methods**, these methods consider feature extraction in both temporal and spatial domains together within a single module, mining high-level co-occurrence features of skeleton sequences. The explicit spatiotemporal methods can have large receptive fields in the spatiotemporal domain with only a small number of module stacks. Si et al. (2019) fuse the sequence modeling capability of LSTMs and the spatial modeling capability of GCNs to capture co-occurrence relations. Liu et al. (2020) expand the topological connection to achieve cross-frame communication between the in-frame and out-frame nodes, and complete the extraction of spatiotemporal features using a GCN-only architecture. Cheng et al. (2020b) propose a shift operation to achieve cross swaps of features among different nodes, which allows spatiotemporal features of different nodes to complement each other without stacking the modules. Hu et al. (2022) proposed a generic framework for modeling information flow across space–time, which equips spatial-only modules with spatiotemporal modeling for region awareness. Qiu et al. (2023) proposed a new spatiotemporal segmental attention method, which divides the skeleton sequence into segments, and several consecutive frames contained in each segment are encoded. Then an intra-segment self-attention module is proposed to capture the relationships of different joints in successive frames.

Recently, some emerging skeleton-based recognition methods have also become active in the research community. Some of them try to use advanced techniques from other fields in combination with the human skeleton recognition task, and some propose novel solutions tailored to specific recognition scenarios. We will present these state-of-the-art methods next. Duan et al. (2022) propose a new approach to skeleton-based action recognition by using the 3D heatmap representation of the human skeleton instead of the mainstream GCN architecture. They achieve state-of-the-art results using a novel pose estimation schema on multiple datasets. Ke et al. (2022) fuse spatiotemporal gradients to focus on spatiotemporal features in action sequences, and define three loss terms on the gradient-based spatiotemporal foci to guide the classifier learning. Xu et al. (2022) design topology-aware modules using a pure CNN architecture and enhance topological features using joint-level and coordinate-level information. Wang et al. (2022) use multiform skeletons to improve GCNs, and their model not only learns action representations from a single form of skeleton, but also adaptively imitates useful representations derived from other forms of skeletons. Liu et al. (2022) construct the recognition model based on the multi-head self-attention mechanism. They design graph transformer operators to model the higher-order spatial dependencies between joints and use temporal channel attention to enhance temporal motion correlations. Zhou et al. (2023) design spatial–temporal decoupling and contrastive feature refinement based on contrastive learning architecture to obtain discriminative representations of the skeleton. Peng et al. (2023) design a benchmark for action recognition considering data scarcity from the body occlusion problem, and their model remains valid for actions with partial body occlusion.

3. Method

To make the paper self-contained, we first briefly review the graph construction for the skeleton sequence and the vanilla graph convolution for skeleton-based action recognition in this section. Then, we elaborate our topology partition enhancement module and feature reconstruction graph convolution. Finally, we introduce the architecture of our FR-GCN in detail.

3.1. Preliminaries

3.1.1. Graph construction

The human skeleton sequence can be represented as a set of joints in consecutive T frames, each frame containing 2D or 3D coordinate vectors of N joints. For skeleton-based action recognition, the spatiotemporal graph on the skeleton sequence can be defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, in which $\mathcal{V} = \{v_{t,i} \mid t = 1, \ldots, T,\ i = 1, \ldots, N\}$ contains all joints on the skeleton sequence. $\mathcal{E} = \{\mathcal{E}_S, \mathcal{E}_\mathcal{T}\}$ represents the set of all edges on the spatiotemporal graph, which consists of two parts: intra-frame edges $\mathcal{E}_S$ and inter-frame edges $\mathcal{E}_\mathcal{T}$. $\mathcal{E}_S = \{(v_{t,i}, v_{t,j}) \mid (i,j) \in C\}$ describes the joint connections inside each frame, where C is the set of naturally connected joints and $(v_{t,i}, v_{t,j})$ indicates that there is an undirected edge between $v_{t,i}$ and $v_{t,j}$. Since different frames share the topology, the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ can be used to represent the topology of all frames. The element $a_{i,j} = 1$ represents the directed edge $\langle v_{t,j}, v_{t,i} \rangle \in \mathcal{E}_S$, and conversely, $a_{i,j} = 0$ represents the directed edge $\langle v_{t,j}, v_{t,i} \rangle \notin \mathcal{E}_S$. $\mathcal{E}_\mathcal{T} = \{(v_{t,i}, v_{t+1,i}) \mid i = 1, \ldots, N\}$ describes the temporal connections between consecutive frames of the same joint. $\mathcal{X} = \{\mathbf{x}_{t,i} \in \mathbb{R}^C \mid t = 1, \ldots, T,\ i = 1, \ldots, N\}$ represents the set of all node features on the spatiotemporal graph.

3.1.2. Vanilla graph convolution

Given the graph as defined above, we can construct graph convolution models on the spatial domain to mine the relationships between joints and extract more advanced node representations. The neighbor set of node $v_{t,i}$ can be defined as $\mathcal{N}(v_{t,i}) = \{v_{t,j} \mid d(v_{t,i}, v_{t,j}) \le D\}$, where $d(v_{t,i}, v_{t,j})$ denotes the shortest path length from node $v_{t,i}$ to node $v_{t,j}$. Normally, we set $D = 1$ to denote the first-order neighbors. Based on the above definition, the graph convolution of node $v_{t,i}$ can be formulated as:

$$Y(v_{t,i}) = \sum_{v_{t,j} \in \mathcal{N}(v_{t,i})} \frac{1}{Z_{t,i}(v_{t,j})} X(v_{t,j})\, W(\mathcal{L}_{t,i}(v_{t,j})) \tag{1}$$

where $X(\cdot)$ obtains the features of the nodes, such as $X(v_{t,j}) = \mathbf{x}_{t,j}$. $W(\cdot)$ is the weight assignment function that assigns a weight indexed by label $\mathcal{L}_{t,i}(v_{t,j})$ from P weight parameters. The normalization term $Z_{t,i}(v_{t,j}) = \left|\{v_{t,k} \mid \mathcal{L}_{t,i}(v_{t,k}) = \mathcal{L}_{t,i}(v_{t,j})\}\right|$ represents the number of elements in the corresponding label set of $v_{t,j}$. $Y(v_{t,i})$ is the output of the graph convolution on node $v_{t,i}$.

3.1.3. Topology partition strategy

In Eq. (1), we define the labeling function $\mathcal{L}_{t,i}(\cdot) : v_{t,j} \to \{1, 2, \dots, P\}$, which partitions the neighbor set of $v_{t,i}$ into P subsets, i.e., any $v_{t,j} \in \mathcal{N}(v_{t,i})$ will be assigned a label. From the topological perspective, we can define the equivalent concepts on $\mathcal{E}_S$. For any directed edge $\langle v_{t,i}, v_{t,j} \rangle \in \mathcal{E}_S$, it is uniquely classified as a label in the set $\{1, 2, \dots, P\}$, i.e., there exists a labeling function $\mathcal{L}_{\mathcal{E}}(\cdot) : \langle v_{t,i}, v_{t,j} \rangle \to \{1, 2, \dots, P\}$ on the set of the edges. Given an adjacency matrix $\mathbf{A}$, and assuming that the partition strategy classifies the edges into P labels, the partition strategy can be formulated as:

$$\begin{aligned} &\mathbf{A} = \mathbf{A}_1 + \mathbf{A}_2 + \cdots + \mathbf{A}_P \\ &\text{s.t.} \quad a_{i,j}^{(k)} = \begin{cases} a_{i,j} & \text{if } \mathcal{L}_{\mathcal{E}}\left(\langle v_{t,j}, v_{t,i} \rangle\right) = k \\ 0 & \text{if } \mathcal{L}_{\mathcal{E}}\left(\langle v_{t,j}, v_{t,i} \rangle\right) \neq k \end{cases} \end{aligned} \tag{2}$$

where $k \in \{1, \dots, P\}$. The constraint denotes that any directed edge $\langle v_{t,j}, v_{t,i} \rangle$ can only be classified into a unique label. The most widely used partition strategy is proposed in ST-GCN (Yan et al., 2018). They partition the edge set into three groups: root group, centripetal group, and centrifugal group. The strategy partitions the adjacency matrix $\mathbf{A}$ into three submatrices, which are $\mathbf{A}_{root}$, $\mathbf{A}_{centripetal}$, and $\mathbf{A}_{centrifugal}$. The labeling function corresponding to this partition strategy can be formulated as:

$$\mathcal{L}_{\mathcal{E}}\left(\langle v_{t,i}, v_{t,j} \rangle\right) = \begin{cases} 0 : root & \text{if } d\left(v_{t,i}, v_{t,g}\right) = d\left(v_{t,j}, v_{t,g}\right) \\ 1 : centripetal & \text{if } d\left(v_{t,i}, v_{t,g}\right) > d\left(v_{t,j}, v_{t,g}\right) \\ 2 : centrifugal & \text{if } d\left(v_{t,i}, v_{t,g}\right) < d\left(v_{t,j}, v_{t,g}\right) \end{cases} \tag{3}$$

where node $v_{t,g}$ is the center of gravity in the skeleton graph.

3.1.4. Implementation

Implementing the graph convolution represented by Eq. (1) is not an intuitive operation. Specifically, the feature set $\mathcal{X}$ can be represented as a feature tensor $\mathbf{X} \in \mathbb{R}^{T \times N \times C}$, where C is the number of feature channels. Given the skeleton input $\mathbf{X}$ and the graph $\mathbf{A}$, Eq. (1) can be implemented as the layer-wise GCN update rule at time t:

$$\mathbf{X}_t^{(l+1)} = \sum_{k=1}^{P} \sigma\left(\mathbf{D}_k^{-\frac{1}{2}} \mathbf{A}_k \mathbf{D}_k^{-\frac{1}{2}} \mathbf{X}_t^{(l)} \Theta_k^{(l)}\right) \tag{4}$$

where $\mathbf{X}_t^{(l)} \in \mathbb{R}^{N \times C}$ is the feature of all nodes in the lth layer at time t. $\Theta_k^{(l)} \in \mathbb{R}^{C_l \times C_{l+1}}$ is the learnable weight matrix of the kth partition in the lth layer. $\mathbf{A}_k$ is the kth partition submatrix, and $\mathbf{D}_k$ is the degree matrix of $\mathbf{A}_k$. $\sigma(\cdot)$ is an activation function. $\mathbf{D}_k^{-\frac{1}{2}} \mathbf{A}_k \mathbf{D}_k^{-\frac{1}{2}} \mathbf{X}_t^{(l)}$ can be intuitively interpreted as the approximate aggregation of nodes over their first-order neighborhoods.

3.2. Topology partition enhancement module

Many researches show that an effective topology partition strategy is helpful in improving the performance of skeleton-based action recognition methods. The topology partition strategy proposed in ST-GCN (Yan et al., 2018) becomes the partition baseline by its efficiency and simplicity. Using the partition strategy of ST-GCN, the adjacency matrix $\mathbf{A}$ can be partitioned into three submatrices with different contextual semantics: $\mathbf{A}_{root}$, $\mathbf{A}_{centripetal}$, and $\mathbf{A}_{centrifugal}$. The topologies represented by the three submatrices contain three types of edges: self-loop edges, centripetal edges, and centrifugal edges. Most existing methods generally utilize GCNs to learn patterns on the above partitioned adjacency submatrices separately, and finally aggregate the learned features of the partitions to recognize actions. However, this partition strategy only considers independent feature extraction for different semantics and ignores the joint dependencies among different types of edges, leading to the strategy being unable to adequately represent the co-occurrence relationships among complementary edges.

To address the above problem, we propose a plug-and-play topology partition enhancement (TPE) module that enhances the existing partitions in a differentiable manner according to our adaptive topology partition algorithm. Our TPE module fully takes into account the correlations between the potential edges and the existing partitions, and adaptively complements the edges to their strongly correlated partition submatrices. Moreover, the partition enhancement is unique to different samples and different model layers, which greatly improves the flexibility of the model. Taking the partition strategy of ST-GCN as an example, using our TPE module to enhance the partitions can make the centripetal group contains the adaptive complementary centrifugal edges and self-loop edges, while the centrifugal group contains the adaptive complementary centripetal edges and self-loop edges. These complements are semantically consistent with the intrinsic representation of actions. The enhanced topology partitions can better capture the co-occurrence features between centrifugal and centripetal motions. Next, we mathematically model the partition enhancement process and elaborate our adaptive topology partition algorithm.

Mathematical modeling. Given the adjacency matrices $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_P$ corresponding to the existing P partitions, the global adjacency matrix $\mathbf{A}$ satisfies $\mathbf{A} = \mathbf{A}_1 + \mathbf{A}_2 + \cdots + \mathbf{A}_P$. For any directed edge $\langle v_i, v_j \rangle$ in the global graph, partition enhancement means adaptively classifying the edge $\langle v_i, v_j \rangle$ as label $k \in \{1, 2, \dots, P\}$ and complementing the edge $\langle v_i, v_j \rangle$ to the existing kth partition $\mathbf{A}_k$, enhancing the topological representation of the kth partition. $\mathbf{A}_k^+$ is the adjacency matrix consisting of all the enhanced edges for the kth partition, and $\tilde{\mathbf{A}}_k = \mathbf{A}_k + \mathbf{A}_k^+$ is defined as the new adjacency matrix after the corresponding enhancement. To obtain the enhancement matrices $\mathbf{A}_1^+, \mathbf{A}_2^+, \dots, \mathbf{A}_P^+$, we define the formula as:

$$\begin{aligned} \mathbf{A} &= \mathbf{A}_1^+ + \cdots + \mathbf{A}_k^+ + \cdots + \mathbf{A}_P^+ \\ &= \mathbf{A} \odot \mathbf{M}_1 + \cdots + \mathbf{A} \odot \mathbf{M}_k \\ &\quad + \cdots + \mathbf{A} \odot \mathbf{M}_P \end{aligned} \tag{5}$$

where $\odot$ is the Hadamard product. $\mathbf{M}_k$ is the mask matrix, and it is a 0–1 matrix. In order to completely and uniquely partition each edge, $\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_P$ needs to satisfy $\mathbf{M}_1 + \mathbf{M}_2 + \cdots + \mathbf{M}_P = \mathbf{J}$, where $\mathbf{J}$ is a matrix filled with ones.

The key to obtain the enhancement matrices $\mathbf{A}_1^+, \mathbf{A}_2^+, \dots, \mathbf{A}_P^+$ is to calculate the mask matrices $\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_P$. Given the input feature tensor $\mathbf{X} \in \mathbb{R}^{T \times N \times C}$, the output $\mathbf{M}^*$ of our adaptive topology partition (ATP) algorithm is formulated as:

$$\mathbf{M}^* = \mathcal{ATP}(\mathbf{X}, P) \tag{6}$$

where $\mathbf{M}^* = Stack\left(\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_P\right) \in \mathbb{R}^{P \times N \times N}$ is the stacking of P mask matrices. Based on the above, our TPE module can be formulated as:

$$\tilde{\mathbf{A}}^* = \mathbf{A}^* + \alpha \cdot (\mathbf{A} \odot \mathcal{ATP}(\mathbf{X}, P)) \tag{7}$$

where α is a trainable scalar that is used to adjust the strength of the partition enhancement. $\mathbf{A}^* = Stack\left(\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_P\right) \in \mathbb{R}^{P \times N \times N}$ is the stacking of the existing partitions. $\mathbf{A}$ is the global topology. $\tilde{\mathbf{A}}^* \in \mathbb{R}^{P \times N \times N}$ is the stacking of the adjacency matrices after enhancement. $\mathbf{A} \odot \mathcal{ATP}(\mathbf{X}, P)$ in Eq. (7) can be interpreted as getting the stacking of the enhancement matrices $\mathbf{A}_1^+, \mathbf{A}_2^+, \dots, \mathbf{A}_P^+$.

In Eq. (5), we calculate the enhancement matrix $\mathbf{A}_k^+$ by determining the mask matrix $\mathbf{M}_k$. From a mathematical analysis standpoint, this hierarchical calculation mode is more advantageous for the gradient optimizer to update the model parameters, which allows for faster convergence of the model. Our reasoning for this is as follows. Actually, obtaining $\mathbf{A}_k^+$ directly through differentiable calculations is a challenging task. Based on the matrix decomposition theory (Wang and Zhang, 2012), it is impossible to directly construct a function or algorithm $Alg(\cdot)$ that completes the optimizable computational process $\mathbf{A}_k^+ = Alg(\mathbf{X}, P)$ while satisfying the constraint of condition $\mathbf{A} = \mathbf{A}_1^+ + \cdots + \mathbf{A}_k^+ + \cdots + \mathbf{A}_P^+$. To ensure that $\mathbf{A}_k^+$ satisfies the constraint, it can only be achieved by constructing a regularization term, which can be formulated as:

$$L_{reg} = \sum_{l}^{L} \alpha_l \cdot \left\| A, Alg_l(\mathbf{X}, P) \right\| \tag{8}$$

where L is the total number of layers in the model. $\|\cdot\|$ is a penalty function. Using this method means that the convergence of the model is limited by the regularization term. In addition to this, it is also challenging to design effective regularization coefficients and penalty functions. Therefore, we first solve for the mask matrix $\mathbf{M}_k$ and then assist in calculating the enhancement matrix $\mathbf{A}_k^+$. In this way, we can get rid of the constraints and achieve the same function more easily.

Adaptive topology partition algorithm. As shown in Eq. (6), our adaptive topology partition (ATP) algorithm aims to obtain the mask matrices $\mathbf{M}^* \in \mathbb{R}^{P\times N\times N}$ through $\mathcal{ATP}(\cdot)$. Specifically, our adaptive topology partition algorithm contains three parts: (1) Semantic feature representation of the partitions completed by hidden feature extraction function $\mathcal{H}(\cdot)$. (2) Pair-wise node feature modeling which includes pair-wise correlation modeling function $\mathcal{M}(\cdot)$ and feature transformation function $\mathcal{T}(\cdot)$. (3) Mask matrix modeling completed by similarity calculation function $S(\cdot)$ and activation function $PeakSoftmax(\cdot)$. Given the above parts, we can refine Eq. (6) as:

$$\mathbf{M}^* = \mathcal{ATP}(\mathbf{X}, P) = \delta\left(S\left(\mathcal{H}([1,\dots,P]), \mathcal{T}(\mathcal{M}(\mathbf{X}))\right)\right) \tag{9}$$

where $\delta(\cdot)$ is a special activation function that can be chosen as $PeakSoftmax(\cdot)$. $PeakSoftmax(\cdot)$ will be described in detail later.

The semantic feature representation aims to assign the semantic information for each partition by $\mathcal{H}(\cdot)$. We utilize an embedding layer to extract partition features, and the formula is as:

$$\mathbf{X}_p = \mathcal{H}([1,\dots,P]) = Embedding([1,\dots,P]) \tag{10}$$

where $\mathbf{X}_p \in \mathbb{R}^{P\times C_{emb}}$ represents the semantic features of the existing P partitions, and the semantic feature of each partition is represented by a C_{emb}-dimensional vector.

The pair-wise node feature modeling aims to construct correlations between the nodes using $\mathcal{M}(\cdot)$. To reduce computation cost, we utilize two independent point-wise convolutions to transform the input feature $\mathbf{X} \in \mathbb{R}^{T\times N\times C}$, and the process can be formulated as:

$$\mathbf{X}_q = Conv2D_{1\times1}(\mathbf{X}) \tag{11}$$

$$\mathbf{X}_k = Conv2D_{1\times1}(\mathbf{X}) \tag{12}$$

where $\mathbf{X}_q \in \mathbb{R}^{T\times N\times \frac{C}{r}}$ and $\mathbf{X}_k \in \mathbb{R}^{T\times N\times \frac{C}{r}}$ are two feature tensors that contain different semantic information, and r is the reduction factor of the feature channels. Given a pair of nodes (i,j), we first obtain the corresponding node features $\mathbf{x}_i \in \mathbb{R}^{T\times \frac{C}{r}}$ and $\mathbf{x}_j \in \mathbb{R}^{T\times \frac{C}{r}}$ from $\mathbf{X}_q$ and $\mathbf{X}_k$, respectively. Then, we utilize a simple channel-wise dot product to compute the feature vectors of the node pair, the computation can be formulated as:

$$\mathbf{x}_{i,j} = [\mathbf{x}_{i(:,1)} \cdot \mathbf{x}_{j(:,1)} \| \mathbf{x}_{i(:,2)} \cdot \mathbf{x}_{j(:,2)} \| \cdots \| \mathbf{x}_{i(:,C/r)} \cdot \mathbf{x}_{j(:,C/r)}] \tag{13}$$

where $\|$ is the concat operation of vectors, and $\mathbf{x}_{i,j} \in \mathbb{R}^{\frac{C}{r}}$ is the feature vector of the node pair (i,j). Based on the above equations, we can define $\mathcal{M}(\cdot)$ as the following form:

$$\mathbf{X}_v = \mathcal{M}(\mathbf{X}) = \frac{\mathbf{X}_q^T \mathbf{X}_k}{\sqrt{N}} \tag{14}$$

where N is the number of nodes, which is used to stabilize the training. $\mathbf{X}_v \in \mathbb{R}^{N\times N\times \frac{C}{r}}$ represents the feature tensor of all node pairs. In order to make $\mathbf{X}_v$ and $\mathbf{X}_p$ compatible in the feature domain, we need the number of feature channels in $\mathbf{X}_v$ to be the same as that of $\mathbf{X}_p$. We transform $\mathbf{X}_v$ using the feature transformation function $\mathcal{T}(\cdot)$. $\mathcal{T}(\cdot)$ is implemented using a simple linear transformation and formulated as:

$$\tilde{\mathbf{X}} = \mathcal{T}(\mathbf{X}_v) = \mathbf{X}_v \mathbf{W} \tag{15}$$

where $\tilde{\mathbf{X}} \in \mathbb{R}^{N\times N\times C_{emb}}$ is the transformed node-pair feature tensor. $\mathbf{W} \in \mathbb{R}^{\frac{C}{r}\times C_{emb}}$ is a trainable weight matrix.

The mask matrix modeling aims to calculate the final mask matrices $\mathbf{M}^* \in \mathbb{R}^{P\times N\times N}$ based on the above two parts. $\mathbf{m}^*_{i,j} = \mathbf{M}^*_{:,i,j} \in \mathbb{R}^P$ is a P-dimensional vector with one-hot form. Assuming that the kth dimension of $\mathbf{m}^*_{i,j}$ is 1 and the remaining positions are 0, the physical meaning of this representation indicates that the edge $\langle v_j, v_i\rangle$ should be enhanced to the kth partition. The goal of $S(\cdot)$ is to calculate the similarity between the feature vectors of the node pairs and the feature vectors of the partitions. $\tilde{\mathbf{x}}_{i,j} = \tilde{\mathbf{X}}_{i,j,:} \in \mathbb{R}^{C_{emb}}$ and $\mathbf{x}_{p(k)} = \mathbf{X}_{p(k,:)} \in \mathbb{R}^{C_{emb}}$ represent the feature vector of the node pair (i,j) and the feature vector of the kth partition, respectively. We choose the vector dot product to calculate the similarity between $\tilde{\mathbf{x}}_{i,j}$ and $\mathbf{x}_{p(k)}$ and formulate it as:

$$\mu = \tilde{\mathbf{x}}_{i,j}^T \cdot \mathbf{x}_{p(k)} \tag{16}$$

where the output μ is a scalar that represents the similarity between $\tilde{\mathbf{x}}_{i,j}$ and $\mathbf{x}_{p(k)}$. By modifying Eq. (16) into matrix form and expanding it, the mask matrix modeling can be formulated as:

$$\mathbf{M}^* = \delta(\mathbf{U}) = \delta\left(S\left(\mathbf{X}_p, \tilde{\mathbf{X}}\right)\right) = \delta\left(\tilde{\mathbf{X}}\mathbf{X}_p^T\right) \tag{17}$$

where $\delta(\cdot)$ is an activation function for converting the similarity matrices $\mathbf{U} \in \mathbb{R}^{P\times N\times N}$ to the mask matrices $\mathbf{M}^* \in \mathbb{R}^{P\times N\times N}$. We propose a new activation function $PeakSoftmax(\cdot)$ to implement $\delta(\cdot)$. We define the similarity vector of the node pair (i,j) as $\mathbf{u}_{i,j} = \mathbf{U}_{:,i,j} \in \mathbb{R}^P$. The function of $PeakSoftmax(\cdot) : \mathbf{u}_{i,j} \to \mathbf{m}^*_{i,j}$ is to assign value 1 to the position of the maximum element in $\mathbf{u}_{i,j}$, and 0 to the rest of the positions. $PeakSoftmax(\cdot)$ can be implemented by two steps:

(1) Obtain the probability distribution of $\mathbf{u}_{i,j}$ over the P partitions by the classical function $Softmax(\cdot)$.
(2) Polarize the probability distribution in a differentiable way to achieve the maximum probability assigned as 1 and the rest assigned as 0.

Based on the functional properties of $PeakSoftmax(\cdot)$, it can also be called as probabilistic polarization function. Given a general input vector $\mathbf{v} \in \mathbb{R}^P$, the probability polarization function $PeakSoftmax(\cdot)$ can be formulated as:

$$PeakSoftmax(\mathbf{v}_i) = \begin{cases} 1 & \text{if } i = argmax(Softmax(\mathbf{v})) \\ 0 & \text{if } i \neq argmax(Softmax(\mathbf{v})) \end{cases} \tag{18}$$

where $argmax(\cdot)$ is used to get the position of the maximum value in a vector. The aim of using $Softmax(\cdot)$ for vector $\mathbf{v}$ is to regularize the scale of the elements in $\mathbf{v}$, which is beneficial for stabilizing the model training. If $Softmax(\cdot)$ is replaced by $Sigmoid(\cdot)$ in Eq. (18), $PeakSoftmax(\cdot)$ converts to $PeakSigmoid(\cdot)$.

The implementation of the activation function in Eq. (18) is not straightforward, and we provide an implementation idea from a differentiable perspective. Given a general input vector $\mathbf{v} \in \mathbb{R}^P$, the steps for concretizing Eq. (18) are as follows. (1) Get $\mathbf{v}_{soft}$ through $Softmax(\mathbf{v})$. (2) Get the index idx_{max} where the maximum value of the element is located using function $argmax(\mathbf{v}_{soft})$. (3) Construct a mask vector $\mathbf{v}_{mask} = [0,\dots,1/\mathbf{v}_{idx_{max}},\dots,0]$. (4) Calculate the final activation vector $\mathbf{v}_{act} = \mathbf{v}_{soft} \cdot \mathbf{v}_{mask}$. As the final step in obtaining the activation vector involves only vector multiplication, it does not hinder the process of gradient propagation. It is certain that there is more than one implementation that can satisfy the requirements of activation function $PeakSoftmax(\cdot)$. It is anticipated that aggregating more dimensional information of the vector can design implementations that are more conducive to gradient descent.

Our ATP algorithm is summarized in Algorithm 1.

3.3. Feature reconstruction graph convolution

Effective extraction of spatiotemporal relationships in skeleton sequences is crucial for the accurate recognition of human actions, but refined spatiotemporal feature extraction is largely neglected in previous works. Most existing works employ interleaved deployment of

Algorithm 1: ATP Algorithm for TPE Module.

Input: Action feature tensor $\mathbf{X} \in \mathbb{R}^{T\times N\times C}$;
Number of partitions P;
Number of embedding channels C_{emb};
Reduction factor r.
Output: Mask matrices $\mathbf{M}^* \in \mathbb{R}^{P\times N\times N}$.

Initialize the index vector of the partitions $\mathbf{v}_{idx} = [1, 2, \dots, P]$;
$\mathbf{X}_p \leftarrow Embedding\left(\mathbf{v}_{idx}\right)$;
$\mathbf{X}_v \leftarrow \frac{Conv2D_{1\times1}(\mathbf{X})^T \cdot Conv2D_{1\times1}(\mathbf{X})}{\sqrt{N}}$;
Initialize $\mathbf{W} \in \mathbb{R}^{\frac{C}{r}\times C_{emb}}$ randomly;
$\tilde{\mathbf{X}} \leftarrow \mathbf{X}_v\mathbf{W}$;
$\mathbf{U} \leftarrow \tilde{\mathbf{X}}\mathbf{X}_p^T$;
Initialize $\mathbf{M}^* \leftarrow []$;
foreach $\mathbf{u}_{i,j} \in \mathbb{R}^P$ *in* $\mathbf{U}$ **do**
  $\mathbf{m}^*_{i,j} \leftarrow PeakSoftmax\left(\mathbf{u}_{i,j}\right)$;
  Add $\mathbf{m}^*_{i,j}$ to $\mathbf{M}^*$;
end
Convert $\mathbf{M}^*$ to $\mathbb{R}^{P\times N\times N}$ by dimensional transformation.

spatial-only and temporal-only modules for spatiotemporal feature extraction, in which spatial-only modules focus only on extracting spatial configurations and temporal-only modules focus only on extracting temporal dynamics. However, the interleaved deployment cannot effectively explore refined spatiotemporal dependencies, resulting in the inability to accurately capture spatiotemporal co-occurrence.

To solve the above problem, we propose a feature reconstruction graph convolution (FR-GC). The illustration of our FR-GC is shown in Fig. 2, which mainly contains two graph convolutions: (1) Spatial graph convolution represented by function $GC_s(\cdot)$, which is quite similar to the GCN represented by Eq. (4) and is mainly used for extracting spatial features. (2) Spatiotemporal graph convolution represented by function $GC_{st}(\cdot)$, which is the core component of FR-GC and is mainly used for extracting refined spatiotemporal features. Specifically, our spatiotemporal graph convolution first extracts the coarsened temporal feature and the coarsened spatial feature from the input tensor by feature transformation, and then combines the two to refactor out the refined spatiotemporal feature tensor that has the same shape as the input tensor. The reconstructed spatiotemporal feature fully integrates temporal and spatial semantics and is a more advanced feature representation. The reconstruction approach can also effectively solve the problem that the spatiotemporal perception for the beginning of the skeleton sequence may gradually weaken with the propagation of the layers. To further refine the spatiotemporal feature, we construct an adaptive topology that conforms to the spatiotemporal relations. Then, we employ a graph convolution on the spatiotemporal topology to complete the final feature learning. Given the input feature $\mathbf{X} \in \mathbb{R}^{T\times N\times C_{in}}$, the output $\mathbf{Z} \in \mathbb{R}^{T\times N\times C_{out}}$ of FR-GC is formulated as:

$$\mathbf{Z} = concat\left(GC_s\left(\mathbf{X}, \tilde{\mathbf{A}}\right), GC_{st}\left(\mathbf{X}\right)\right) \tag{19}$$

where $\tilde{\mathbf{A}} \in \mathbb{R}^{N\times N}$ is the topology indicated by one of the partitions after topology partition enhancement. Next, we introduce the spatial graph convolution and the spatiotemporal graph convolution in detail.

Spatial graph convolution. The spatial graph convolution is indispensable for extracting discriminative spatial features. The vanilla spatial graph convolution is shown in Eq. (4), but the topology in Eq. (4) is static. The static graph is unable to effectively represent the implicit connections between the nodes that do not exist in the actual topology, and also lacks the flexible modeling ability for the different layers of the model. Inspired by Shi et al. (2019b), we modify the basic convolution by adding a self-learning graph. Given the input feature $\mathbf{X} \in \mathbb{R}^{T\times N\times C_{in}}$ and the adjacency matrix $\tilde{\mathbf{A}} \in \mathbb{R}^{N\times N}$, the spatial graph convolution can be formulated as:

$$\mathbf{Z}_s = GC_s\left(\mathbf{X}, \tilde{\mathbf{A}}\right) = \left(\tilde{\mathbf{A}} + \mathbf{B}\right) \cdot \mathbf{X}\mathbf{W} \tag{20}$$

where $\mathbf{W} \in \mathbb{R}^{C_{in}\times\frac{C_{out}}{2}}$ is a learnable weight matrix for high-dimensional feature transformation. $\mathbf{B}$ is a self-learning graph, which each element in it is initialized to 0 for more stable training. There is no restriction on the values of $\mathbf{B}$, which means that the graph is learned entirely based on the training data. With this data-driven approach, the model can better learn the graph that is suitable for action recognition.

Spatiotemporal graph convolution. As shown in Fig. 2, the spatiotemporal graph convolution aims to learn the refined spatiotemporal features, and it is the core component of the whole FR-GC. Our spatiotemporal graph convolution contains two main parts: (1) Spatiotemporal feature reconstruction. This part refactors out the refined spatiotemporal features based on the temporal and spatial semantics of the input data. (2) Spatiotemporal topology modeling. This part adaptively constructs a topology that conforms to the spatiotemporal relationships driven by the reconstructed data. Finally, we employ a graph convolution to accomplish a more high-level spatiotemporal feature aggregation. Given the input feature $\mathbf{X} \in \mathbb{R}^{T\times N\times C_{in}}$, the spatiotemporal graph convolution can be formulated as:

$$\mathbf{Z}_{st} = GC_{st}\left(\mathcal{R}\left(\mathbf{X}\right)\right) = GC_{st}\left(\mathbf{X}_{st}\right) = \mathcal{Q}\left(\mathbf{X}_{st}\right) \cdot \mathbf{X}_{st}\mathbf{W} \tag{21}$$

where $\mathbf{X}_{st} \in \mathbb{R}^{T\times N\times C_{in}}$ is the refined spatiotemporal feature reconstructed by the feature reconstruction function $\mathcal{R}(\cdot)$. $\mathcal{Q}(\cdot)$ is the topology modeling function for constructing the spatiotemporal topology. $\mathbf{W} \in \mathbb{R}^{C_{in}\times\frac{C_{out}}{2}}$ is a learnable weight matrix for high-dimensional feature transformation. In Eq. (21), the reconstructed node features contain not only rich spatial semantics but also global temporal semantics. Therefore, the aggregation of the graph convolution is no longer limited to the spatial domain, but has rich perception in the spatiotemporal domain and can extract more profound spatiotemporal semantics.

The spatiotemporal feature reconstruction aims to obtain the refined spatiotemporal feature $\mathbf{X}_{st}$ by $\mathcal{R}(\cdot)$. $\mathbf{X}_{st}$ will be the actual input of the graph convolution. The reconstructed spatiotemporal features can be updated by back-propagation to better accommodate the spatiotemporal perception for different actions. Given the original input $\mathbf{X} \in \mathbb{R}^{T\times N\times C_{in}}$, we first utilize spatial pooling $AvgPool_s(\cdot)$ and temporal pooling $AvgPool_t(\cdot)$ to obtain the coarsened temporal feature $\mathbf{X}'_\tau \in \mathbb{R}^{T\times C_{in}}$ and the coarsened spatial feature $\mathbf{X}'_s \in \mathbb{R}^{N\times C_{in}}$. To reduce computation cost, we employ two independent multi-layer perceptrons (MLPs) $\phi(\cdot)$ and $\psi(\cdot)$ to accomplish feature transformation on $\mathbf{X}'_\tau$ and $\mathbf{X}'_s$. The pooling procedure and the feature transformation procedure can be formulated as:

$$\mathbf{X}_\tau = \phi\left(AvgPool_s\left(\mathbf{X}\right)\right) = \phi\left(\mathbf{X}'_\tau\right) \tag{22}$$

$$\mathbf{X}_s = \psi\left(AvgPool_t\left(\mathbf{X}\right)\right) = \psi\left(\mathbf{X}'_s\right) \tag{23}$$

where $\mathbf{X}_\tau \in \mathbb{R}^{T\times\frac{C_{in}}{r}}$ and $\mathbf{X}_s \in \mathbb{R}^{N\times\frac{C_{in}}{r}}$ are the coarsened temporal feature and coarsened spatial feature after feature transformation, respectively. r is the reduction factor of the feature channels.

Based on the coarsened temporal feature $\mathbf{X}_\tau$ and coarsened spatial feature $\mathbf{X}_s$, we design two simple and effective spatiotemporal feature reconstruction functions. Given the temporal feature vector $\mathbf{x}_i = \mathbf{X}_{\tau(i,:)} \in \mathbb{R}^{\frac{C_{in}}{r}}$ and the spatial feature vector $\mathbf{x}_j = \mathbf{X}_{s(j,:)} \in \mathbb{R}^{\frac{C_{in}}{r}}$, where $i \in \{1, \dots, T\}$ and $j \in \{1, \dots, N\}$. The first spatiotemporal feature reconstruction function $\mathcal{R}_1(\cdot)$ is formulated as:

$$\mathbf{x}_{i,j} = \mathcal{R}_1\left(\mathbf{x}_i, \mathbf{x}_j\right) = MLP\left(\mathbf{x}_i \| \mathbf{x}_j\right) \tag{24}$$

where $\mathbf{x}_{i,j} = \mathbf{X}_{st(i,j,:)} \in \mathbb{R}^{C_{in}}$ is the spatiotemporal feature vector of the node j at time i. $MLP(\cdot)$ is a multi-layer perceptron for dimensional transformation. $\|$ is the concat operation of vectors. Essentially, $\mathcal{R}_1(\cdot)$ first concatenates the temporal feature vector $\mathbf{x}_i$ and the spatial feature

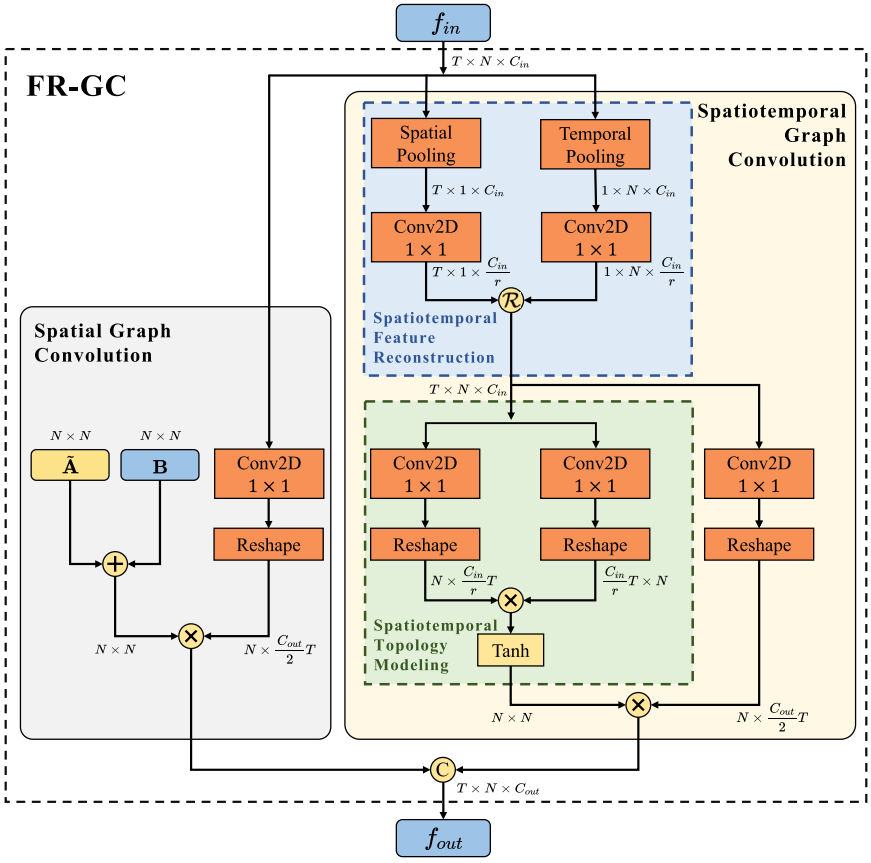


Fig. 2. Illustration of the feature reconstruction graph convolution. FR-GC consists of the spatial graph convolution and the spatiotemporal graph convolution. There are two kinds of graphs as the inputs of the spatial graph convolution, i.e., $\tilde{\mathbf{A}}$ and $\mathbf{B}$. The spatiotemporal graph convolution adaptively constructs a spatiotemporal graph as the input. $\mathcal{R}$ denotes the feature reconstruction function. $\oplus$ denotes the element-wise addition. $\otimes$ denotes the matirx multiplication. C denotes the concat operation.

vector $\mathbf{x}_j$ to construct the weak spatiotemporal feature vector, and then completes a more advanced modeling by $MLP(\cdot)$ to construct the strong spatiotemporal feature. The second spatiotemporal feature reconstruction function $\mathcal{R}_2(\cdot)$ is formulated as:

$$\mathbf{x}_{i,j} = \mathcal{R}_2\left(\mathbf{x}_i, \mathbf{x}_j\right) = MLP\left(\mathbf{x}_i + \mathbf{x}_j\right) \tag{25}$$

where feature vector fusion is accomplished by using a summation operation, which is more efficient than $\mathcal{R}_1(\cdot)$.

We further analyze the feature representation capability of the above two feature reconstruction functions from a mathematical perspective. Without loss of generality, we define $MLP(\cdot)$ to be a classical two-layer network. Given the input vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, the feature reconstruction function in concat mode can be formulated as:

$$\begin{aligned} y_{\text{concat}} &= g\left(W_2^c \cdot g\left(W_1^c \cdot \left[x_1; x_2\right] + b_1\right) + b_2\right) \\ &= g\left(W_2^c \cdot g\left(W_{11}^c \cdot x_1 + W_{12}^c \cdot x_2 + b_1\right) + b_2\right) \end{aligned} \tag{26}$$

Similarly, the function in add mode can be formulated as:

$$\begin{aligned} y_{\text{add}} &= g\left(W_2^a \cdot g\left(W_1^a \cdot \left(x_1 + x_2\right) + b_1\right) + b_2\right) \\ &= g\left(W_2^a \cdot g\left(W_1^a \cdot x_1 + W_1^a \cdot x_2 + b_1\right) + b_2\right) \end{aligned} \tag{27}$$

From Eqs. (26) and (27), if the learnable parameters and activation function satisfy the following condition:

$$W_1^a = W_{11}^c = W_{12}^c \tag{28}$$

$$W_2^a = W_{21}^c = W_{21}^c \tag{29}$$

$$W \cdot g(\alpha + \beta) = W \cdot g(\alpha) + W \cdot g(\beta) \tag{30}$$

then $y_{\text{concat}} \Leftrightarrow y_{\text{add}}$ can be derived. This implies that the representational capability of feature reconstruction function $\mathcal{R}_1(\cdot)$ is greater than that of $\mathcal{R}_2(\cdot)$. This is because all possible outputs of $\mathcal{R}_2(\cdot)$ can be obtained by constructing the learnable parameters of $\mathcal{R}_1(\cdot)$, but not vice versa.

The spatiotemporal topology modeling aims to construct a graph which conforms to the reconstructed feature. For the reconstructed spatiotemporal feature $\mathbf{X}_{st}$, it is not optimal to complete further feature aggregation on the graph constructed by the natural connections of the human body. This is because $\mathbf{X}_{st}$ is not only a representation of each node on the spatial domain, but also a strong spatiotemporal feature that fully incorporates temporal dynamics and spatial configurations. The natural topology cannot simulate the complex spatiotemporal relationships among the nodes in $\mathbf{X}_{st}$. To solve the above problem, we design an effective spatiotemporal topology modeling function $\mathcal{Q}(\cdot)$, which is entirely data-driven and can effectively simulate the complex spatiotemporal topology. Given the input spatiotemporal feature $\mathbf{X}_{st} \in \mathbb{R}^{T \times N \times C_{in}}$, we first execute the feature transformation by two point-wise convolutions and reshape the output features, which is formulated as:

$$\mathbf{X}_{st}^{(1)} = Reshape\left(Conv2D_{1\times1}\left(\mathbf{X}_{st}\right)\right) \tag{31}$$

$$\mathbf{X}_{st}^{(2)} = Reshape\left(Conv2D_{1\times1}\left(\mathbf{X}_{st}\right)\right) \tag{32}$$

where $\mathbf{X}_{st}^{(1)} \in \mathbb{R}^{N \times \frac{C_{in}}{r} T}$ and $\mathbf{X}_{st}^{(2)} \in \mathbb{R}^{N \times \frac{C_{in}}{r} T}$ are the feature vectors of the nodes after transformation and reshaping. r is the reduction factor of the feature channels. Based on $\mathbf{X}_{st}^{(1)}$ and $\mathbf{X}_{st}^{(2)}$, the spatiotemporal topology modeling function $\mathcal{Q}(\cdot)$ can be formulated as:

$$\mathbf{Q} = \mathcal{Q}\left(\mathbf{X}_{st}^{(1)}, \mathbf{X}_{st}^{(2)}\right) = \xi\left(\frac{\mathbf{X}_{st}^{(1)} \mathbf{X}_{st}^{(2)T}}{\sqrt{N}}\right) \tag{33}$$

where $\mathbf{Q} \in \mathbb{R}^{N \times N}$ is the new topology, and $\xi(\cdot)$ is an activation function. We choose $Tanh(\cdot)$ as the activation function because of its ability to explore complex topological relations.

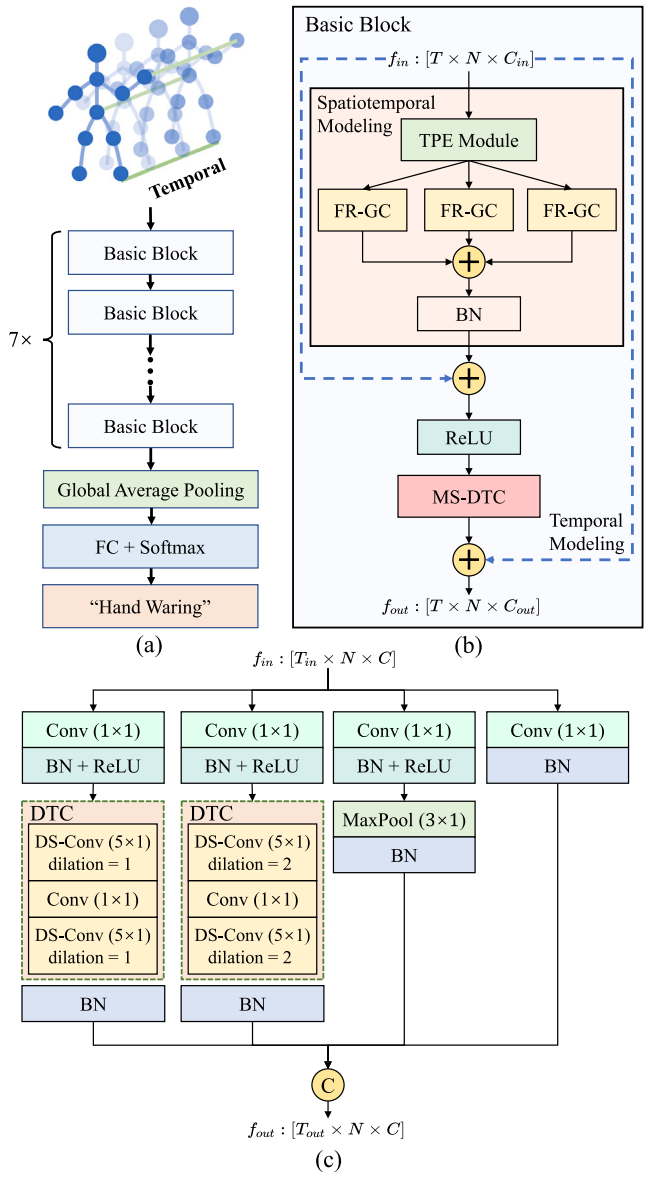


Fig. 3. Overview of the proposed FR-GCN. (a) The architecture of our FR-GCN. (b) The basic block of our FR-GCN, containing the spatiotemporal modeling and the temporal modeling. (c) The multi-scale dual-domain temporal convolution of our FR-GCN. The DTC is the core component of the temporal modeling.

3.4. Model architecture

The overall architecture of the proposed FR-GCN is shown in Fig. 3(a). The input of our FR-GCN is the skeleton action sequence, and the whole network contains seven basic blocks, which are stacked to continuously extract high-level discriminative features. The network ends with a global pooling layer and a softmax classifier for predicting action labels. The number of feature channels for the seven blocks are 64-64-64-128-128-256-256. The temporal dimensions of the 4th and 6th blocks are halved by downsampling with the temporal convolution of step 2. The basic block of our FR-GCN is shown in Fig. 3(b). Each basic block contains two types of modeling. (1) Spatiotemporal modeling first enhances the spatial topology by TPE module, and then utilizes FR-GCs to represent high-level features in the spatiotemporal domain. (2) Temporal modeling further expands the perception of the model in the temporal domain, capturing the extended temporal context. Moreover, residual connections are included in all basic blocks except the first one for more efficient training. We consider the first basic block as the data read-in layer, so it does not contain the residual connection.

Spatiotemporal modeling. In the spatiotemporal modeling, we utilize the TPE module and FR-GCs to extract the refined spatiotemporal features. We adopt the partition strategy in ST-GCN (Yan et al., 2018) to obtain the initial topology partitions. Firstly, our TPE module adaptively enhances the initial topology partitions based on the input feature, improving the topological representation of each partition. Then, we employ three parallel FR-GCs to capture different spatiotemporal semantics in different partitions separately and summarize their outputs as the final output of the spatiotemporal modeling. Our FR-GC contains two components: spatial graph convolution and spatiotemporal graph convolution. The partitions enhanced by the TPE module are used as the input for the spatial graph convolution. The spatial graph convolution is responsible for expanding the receptive field of FR-GC in spatial context, while the spatiotemporal graph convolution is responsible for learning refined spatiotemporal features in spatiotemporal context, and the fusion of the two convolutions can effectively extract discriminative features.

Temporal modeling. Although our FR-GC captures part of the temporal semantics through the feature reconstruction, it is more concerned about the union dependencies between the temporal and spatial semantics. Therefore, its ability to capture the refined temporal context needs to be improved. We propose a multi-scale dual-domain temporal convolution (MS-DTC) to accomplish more refined temporal modeling. The proposed MS-DTC is shown in Fig. 3(c). Our MS-DTC contains four branches, each focusing on the different temporal semantics. To reduce computation cost, each branch in MS-DTC first employs a point-wise convolution to reduce the number of feature channels. The first two branches contain two dual-domain temporal convolutions (DTCs) with different dilations. The third branch contains a max pooling, and the last branch contains only a point-wise convolution.

The DTC is the core of our temporal modeling, which is implemented based on the depth-wise separable convolution (Howard et al., 2017). Our DTC presents an "hourglass" structure, where the upper layer utilizes a depth-wise separable convolution with kernel size 5×1 to learn the coarse-grained temporal context, the middle layer utilizes a point-wise convolution to aggregate the features, and the lower layer further learns the fine-grained context for each temporal channel using a depth-wise separable convolution configured with the same parameters as the upper layer. The upper temporal convolution of our DTC can be seen as learning in the low-frequency temporal domain, and the lower temporal convolution can be seen as learning in the high-frequency temporal domain. The combination of the two can effectively expand the receptive field of the model in the temporal domain.

The proposed DTC outperforms the vanilla temporal convolution (Yan et al., 2018) in both the number of parameters and time complexity. Our analysis is as follows. Without considering dilations and strides, we define the temporal and spatial dimensions of the data as T and N, respectively. The kernel size of the temporal convolution is $K \times 1$. The number of input feature channels and output feature channels are both C. Based on the above definitions, the number of parameters in the DTC is calculated as $2KC + C^2$, where $2KC$ is the number of parameters in the two depth-wise separable convolutions and C^2 is the number of parameters in the point-wise convolution. The number of parameters in the vanilla temporal convolution is calculated as KC^2. Similarly, the time complexity of our DTC can be denoted as $\mathcal{O}\left(2KNTC + NTC^2\right)$, and the time complexity of the vanilla temporal convolution can be denoted as $\mathcal{O}\left(KNTC^2\right)$.

Multi-modality ensemble. The utilization of multi-modal data in skeleton-based action recognition has been demonstrated to be effective in previous works (Shi et al., 2019b; Chen et al., 2021a; Cheng et al., 2020b; Shi et al., 2020). In this work, we generate four modalities for each skeleton action sequence based on the initial input data,

and they are joint, bone, joint motion, and bone motion. Concretely, the joint modality is the initial input data, representing the actual coordinates of the joints. The bone modality is generated by subtracting the coordinates between two adjacent joints, representing the joint-to-joint direction vectors. The joint motion modality and the bone motion modality are both defined by the offsets between adjacent frames in the joint modality and the bone modality, representing the differences in the temporal dimension of the skeleton data. The detailed definition of the four modalities can be found in Shi et al. (2020). The softmax scores of the four streams are summed to obtain the final score for action recognition.

4. Experiments

4.1. Datasets

NTU RGB+D 60. NTU RGB+D 60 (Shahroudy et al., 2016) is a large-scale human action recognition dataset containing 56,578 skeleton action sequences. The actions are performed by 40 different subjects and categorized into 60 different classes. In this dataset, there are 9 classes with the highest number of samples (e.g., "jump up", "salute", "handshaking", etc.), and each of these actions has 948 samples. The class with the fewest number of samples is "hugging another person", which contains 906 samples. The median sample size for classes is 944, and the mean is 942, indicating a relatively balanced distribution. The skeleton graph contains 25 joint nodes, and each skeleton sequence is composed of the 3D joint coordinates. The 3D skeleton data is captured by three Microsoft Kinect v2 cameras from different views. Each skeleton sequence denotes only one action class and is guaranteed to be performed by at most 2 subjects. The authors of this dataset recommend evaluating classification accuracy in two benchmarks. (1) Cross-subject (X-Sub) that divides the subjects into a training group and a test group. The action samples performed by 20 subjects are used for training and the rest for testing. (2) Cross-view (X-View) that uses the action samples collected by camera views 2 and 3 for training and the rest for testing.

NTU RGB+D 120. NTU RGB+D 120 (Liu et al., 2019a) is currently the largest 3D skeleton-based action recognition dataset, and it is an extension of the NTU RGB+D 60 dataset in terms of the number of samples and action classes. This dataset contains 113,945 skeleton action sequences, which are performed by 106 different subjects and categorized into 120 different classes. In this dataset, there are 6 classes with the highest number of samples (e.g., "take off bag", "high-five", "follow other person", etc.), and each of these actions has 960 samples. The class with the fewest number of samples is "hugging another person", which contains 906 samples. The median sample size for classes is 948, and the mean is 949, indicating a relatively balanced distribution. There are 32 different camera setups in the dataset, each denoting a specific location and background. Similarly, the authors of this dataset recommend two evaluation benchmarks. (1) Cross-subject (X-Sub) that uses the action samples performed by 53 subjects for training and the rest for testing. (2) Cross-setup (X-Set) that divides the action samples with even setup IDs for training and odds setup IDs for testing.

4.2. Implementation details

All experiments are conducted on the PyTorch deep learning framework. Stochastic gradient descent (SGD) with Nesterov momentum 0.9 and weight decay 0.0004 is applied as the optimization strategy. The cross-entropy loss with label smoothing (He et al., 2019) is used as the loss function, and the smoothing factor is set to 0.1. Given that almost all mainstream models employ 50–70 training epochs, we set the training epochs to 55 and use a warmup strategy (He et al., 2016) for the first 5 epochs to make the training more stable. The initial learning rate is set to 0.1, and then decays with a factor 0.1 at the 35th epoch and the 45th epoch. We set the batch size to 64, and adopt the data preprocessing in Zhang et al. (2020) to resize each sample to 64 frames. All experiments are performed with the above settings.

Table 1
Comparisons of the model accuracy when adding TPE modules gradually.

Methods	Param.	Acc (%)
Baseline	0.72M	88.4
+2 TPE module	0.72M	88.8$^{\uparrow 0.4}$
+4 TPE module	0.73M	88.9$^{\uparrow 0.5}$
+6 TPE module	0.74M	89.2$^{\uparrow 0.8}$
Full-layer TPE module	0.76M	**89.3**$^{\uparrow \mathbf{0.9}}$

4.3. Ablation study

In this subsection, we comprehensively analyze our proposed TPE module, FR-GC module, and MS-DTC module using only the joint stream on the X-Sub benchmark of the NTU RGB+D 60 dataset. We use ST-GCN (Yan et al., 2018) as the baseline in our ablation study and modify its model architecture to the structure described in Section 3.4 for a fair comparison. Specifically, we add residual connections to the basic block of ST-GCN and replace its temporal convolution with our proposed MS-DTC. The training epochs of the other models in the ablation experiment are all set to 55. Unless otherwise specified, the abbreviations "params". and "acc". used subsequently refer to the number of parameters and accuracy, respectively.

4.3.1. TPE module

Effectiveness of TPE module. To validate the efficacy of our TPE module, we gradually add the TPE modules to the baseline. The experimental results are shown in Table 1. We observe that the accuracy of the model steadily improves with the gradual addition of the TPE modules. Each time we add the TPE modules to the 2 layers of the baseline, the accuracy of the model can basically increase by 0.3% to 0.4%. Notably, the baseline with the 4 layers TPE modules is only 0.1% higher than that with the 2 layers, which indicates that our TPE modules are more effective when added to the lower and upper model layers. When all layers are integrated with the TPE modules, the final accuracy is 0.9% higher than the initial model, which confirms the effectiveness of our TPE module.

Universality of TPE module. To validate the plug-and-play capability of the TPE module, we add our TPE module to several different GCN models. In addition to the baseline and our proposed FR-GCN, we introduce another popular baseline model AGCN (Shi et al., 2019b) to verify the universality of our TPE module. It should be noted that we keep the backbone of the baseline and only replace its graph convolution with that of AGCN for a fair comparison. The experimental results are shown in Table 2. We observe that the performance of all GCN models is improved with the addition of our TPE modules, and our TPE modules only bring additional 0.4M parameters. The main effort of AGCN is to adaptively supplement the edges that do not exist in the original skeleton graph. Our TPE module adaptively partitions the original topology and enhances the new partitions to the predefined partitions, which is essentially a graph supplementation method as well. The experimental results show that our TPE module and AGCN are complementary in supplementing the graph. This is because the supplementation of the TPE module is based on the original graph decomposition, while AGCN focuses more on supplementing edges that do not exist in the original graph. We find that the TPE module can bring certain performance gains to our FR-GC as well, which demonstrates that the TPE module can improve the ability of our FR-GC for spatiotemporal feature extraction.

Table 2
Performance evaluations of different GCN models before and after adding TPE modules.

Methods	TPE module	Param.	Acc (%)
Baseline	✗	0.72M	88.4
	✓	0.76M	89.3$^{\uparrow 0.9}$
AGCN	✗	0.92M	89.1
	✓	0.96M	89.6$^{\uparrow 0.5}$
FR-GCN	✗	0.97M	89.7
	✓	1.01M	90.1$^{\uparrow 0.4}$

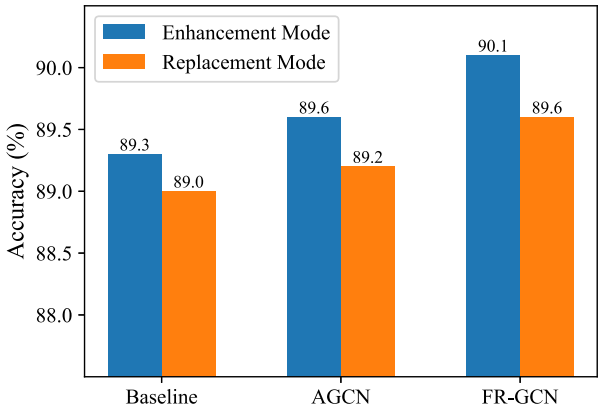


Fig. 4. Comparisons of the model accuracy after changing the enhancement mode to the replacement mode in the TPE module.

Necessity of enhancement mode. To better explain the working mechanism of our TPE module, we answer a pivotal question: why does the TPE module adaptively partition the original topology and then add the new partitions to the predefined partitions in the enhancement mode, instead of directly replacing the predefined partitions? To demonstrate the necessity of the enhancement mode, we replace the predefined partitions in baseline with the new partitions generated by the TPE module instead of using the enhancement mode in Eq. (7). We also conduct the same experiments in AGCN and FR-GCN with the same settings, and the experimental results are shown in Fig. 4. We observe that the performance of all models degrades to some extent after changing the enhancement mode to the replacement mode. The performance of the baseline and AGCN in the replacement mode decreases by 0.3% and 0.4%, respectively, but their accuracy is still higher than the case without the TPE module. The accuracy of our FR-GCN decreases by 0.5%, and it suffers the worst performance degradation. The experimental results show that the new partitions generated by our TPE module are more appropriate to complement the predefined partitions with the enhancement mode rather than the replacement mode, which validates the necessity of the enhancement mode.

Our analysis of the experimental results is as follows. All models have performance degradation after adopting the replacement mode, probably because direct replacement of predefined partitions causes performance perturbation during the model training. It is also difficult for the TPE module to guarantee a reasonable partition strategy for each action sample. Therefore, the enhancement mode is more stable than the replacement mode.

Configuration exploration of TPE module. To determine the optimal performance of the TPE module, we explore different configurations of the TPE module, including the number of embedding channels C_{emb}, the reduction factor of the feature channels r, and the choice of the activation functions δ. The experimental results are shown in Table 3. We observe that our TPE module brings performance improvement to the baseline in all configurations, which demonstrates that our TPE module is sufficiently robust. (1) Comparing models A, B, C, and D, we find that different activation functions have different performance improvements. Compared with the common $Softmax(\cdot)$ and $Sigmoid(\cdot)$, our proposed probabilistic polarization functions $PeakSoftmax(\cdot)$ and $PeakSigmoid(\cdot)$ perform better. This is because common probabilistic activation functions cannot accomplish the unique partition of the edges. The common probabilistic activation functions partition the directed edges in a normalized manner. The performance difference between $PeakSoftmax(\cdot)$ and $PeakSigmoid(\cdot)$ is not obvious in the experiment, but the performance improvement of $PeakSigmoid(\cdot)$ is a little more. This may be related to that $PeakSoftmax(\cdot)$ is more prone to output the exclusive partition, while $PeakSigmoid(\cdot)$ is more likely to jump out of the partition misconception during the training. (2) Comparing models E and F, we find that C_{emb} has little effect on the TPE module, and more channel numbers bring only slightly better result. (3) Comparing models G and H, we find that model G achieves the best result but brings additional parameters. Model H has a significant performance degradation compared with model G because there are too few channels in pair-wise node feature modeling, which is insufficient to effectively represent the high-level semantics of the node pairs.

Table 3
Comparisons of the model accuracy under different configurations of TPE module. The **best** and second best values are highlighted in bold and underline, respectively.

Methods	C_{emb}	r	δ	Param.	Acc (%)
Baseline	–	–	–	0.72M	88.4
A	128	8	PeakSoftmax	0.76M	89.1$^{\uparrow 0.7}$
B	128	8	PeakSigmoid	0.76M	89.3$^{\uparrow 0.9}$
C	128	8	Softmax	0.76M	88.8$^{\uparrow 0.4}$
D	128	8	Sigmoid	0.76M	88.9$^{\uparrow 0.5}$
E	64	8	PeakSigmoid	0.75M	89.1$^{\uparrow 0.7}$
F	256	8	PeakSigmoid	0.78M	89.3$^{\uparrow 0.9}$
G	128	4	PeakSigmoid	0.80M	**89.4**$^{\uparrow \mathbf{1.0}}$
H	128	16	PeakSigmoid	0.74M	88.7$^{\uparrow 0.3}$

4.3.2. FR-gc module

Effectiveness of FR-GC. To demonstrate the capability of our proposed FR-GC in capturing discriminative spatiotemporal features, we gradually replace the GCs with our FR-GCs in the baseline. The experimental results are shown in Table 4. We observe that the performance of the model steadily improves as the GCs in the baseline are gradually replaced with our FR-GCs. When the first two layers of the model are replaced with our FR-GCs, the parameters of the model only increase by 0.01M, while the recognition accuracy of the model improves by 0.7%, which validates the effectiveness of our FR-GC. It can also be observed from Table 4 that with the gradual addition of FR-GC, the parameters of the model are increased, but the classification accuracy of the model is also significantly improved, which indicates that the increase of model parameters can enhance the feature extraction capability of the model to a certain extent. Moreover, the gain of increasing the model parameters is obvious at the initial phase, but with the stacking of FR-GC layers, the rise of model accuracy tends to saturate, which demonstrates that the model parameters and feature extraction capability are not linearly related, but form a relationship similar to a "logarithmic curve". In addition, we also verify the effect of the two topology-related components by removing the self-learning graph **B** and the spatiotemporal graph **Q** from FR-GC. FR-GCN w/o B removes the self-learning graph from Eq. (20) and uses only the enhanced partition as the topological input. We observe that its performance decreases by 0.7% compared with the FR-GCN, suggesting that the self-learning graph is complementary to the enhanced partitions. The synergy of the self-learning graph and the enhanced partitions can better improve the performance of the model. FR-GCN w/o Q removes the spatiotemporal topology modeling. This makes the spatiotemporal graph convolution and the spatial graph convolution in the FR-GC share the physical topology. We find that removing the adaptive spatiotemporal topology degrades the performance of the model by 0.8%, demonstrating the importance of the spatiotemporal topology modeling.

Table 4
Comparisons of the model accuracy when adding FR-GCs gradually and removing B or Q from FR-GCN.

Methods	Param.	Acc (%)
Baseline	0.72M	88.4
+2 FR-GC	0.73M	89.1$^{\uparrow 0.7}$
+4 FR-GC	0.75M	89.4$^{\uparrow 1.0}$
+6 FR-GC	0.82M	89.6$^{\uparrow 1.2}$
FR-GCN w/o B	1.00M	89.4$^{\uparrow 1.0}$
FR-GCN w/o Q	0.93M	89.3$^{\uparrow 0.9}$
FR-GCN	1.01M	**90.1**$^{\uparrow 1.7}$

Table 5
Performance evaluations of different GCN models before and after removing the temporal modeling modules.

Methods	Temporal modeling	Param.	Acc (%)
Baseline	✓	0.72M	88.4
	✗	0.51M	74.3$^{\downarrow 14.1}$
AGCN	✓	0.92M	89.1
	✗	0.71M	79.5$^{\downarrow 9.6}$
Shift-GCN	✓	0.50M	88.4
	✗	0.29M	70.4$^{\downarrow 18.0}$
MS-G3D	✓	3.62M	89.4
	✗	3.41M	86.2$^{\downarrow 3.2}$
FR-GCN	✓	1.01M	90.1
	✗	0.80M	83.5$^{\downarrow 6.6}$

Spatiotemporal robustness of FR-GC. To demonstrate the powerful spatiotemporal modeling capability of our FR-GC, we remove the temporal modeling modules from several GCN models and only keep the GCs. Besides the baseline and AGCN, whose graph convolutions are spatial-only, we further introduce the spatiotemporal models: Shift-GCN (Cheng et al., 2020b) and MS-G3D (Liu et al., 2020). The experimental results are shown in Table 5. We observe that both the baseline and AGCN have significant performance degradation after removing the temporal modeling modules because their GCs only contain spatial modeling capability. MS-G3D and our FR-GCN have less performance degradation because their GCs have spatiotemporal modeling capability to capture features in both temporal and spatial domains. Compared with our FR-GC, MS-G3D performs much better with the removal of the temporal modules. However, the parameters of MS-G3D after removing the temporal modules are 4 times higher than that of our FR-GCN, which brings a heavy storage burden. There are still sliding windows in the temporal dimension inside the G3D modules, which makes MS-G3D computationally inefficient. In contrast, our FR-GC relies on the spatiotemporal reconstruction when extracting spatiotemporal features, which is substantially better than the G3D in terms of both space occupation and time efficiency. Notably, the performance of Shift-GCN decreases by 18.0% after removing the temporal modeling modules, which is the model with the most severe performance degradation. We analyze two possible reasons: (1) Shift-GCN has too few parameters to fit large datasets after removing the temporal modeling modules. (2) Shift-GCN achieves spatiotemporal interactions by introducing the shift operations, but the learning of the temporal features still relies on the temporal modeling modules to some extent.

Configuration exploration of FR-GC. To evaluate how the configurations of FR-GC affect the model performance, we explore the recognition accuracy of FR-GC under different configurations, including the feature reconstruction function $\mathcal{R}$, the reduction factor of the feature channels r, and the activation function ξ used in constructing the spatiotemporal topology. The experimental results are shown in Table 6. We observe that our FR-GC achieves at least 1.0% higher recognition accuracy than the baseline in all configurations, demonstrating the excellent robustness of our model. (1) Comparing models A and B, we find

Table 6
Comparisons of the model accuracy under different FR-GC configurations. The **best** and second best values are highlighted in bold and underline, respectively.

Methods	$\mathcal{R}$	r	ξ	Param.	Acc (%)
Baseline	–	–	–	0.72M	88.4
A	$\mathcal{R}_1$	8	Tanh	1.01M	<u>90.1</u>$^{\uparrow 1.7}$
B	$\mathcal{R}_2$	8	Tanh	0.97M	89.9$^{\uparrow 1.5}$
C	$\mathcal{R}_1$	4	Tanh	1.27M	**90.2**$^{\uparrow 1.8}$
D	$\mathcal{R}_1$	16	Tanh	0.89M	89.8$^{\uparrow 1.4}$
E	$\mathcal{R}_1$	8	Sigmoid	1.01M	89.7$^{\uparrow 1.3}$
F	$\mathcal{R}_1$	8	Softmax	1.01M	89.4$^{\uparrow 1.0}$

Table 7
Comparisons of the model accuracy when removing different components from MS-DTC.

Methods	Param.	Acc (%)
Baseline	0.72M	**88.4**
w/o MS-DTC	1.40M	88.0$^{\downarrow 0.4}$
w/o DTC	0.80M	87.9$^{\downarrow 0.5}$
w/o DTC (upper)	0.71M	88.0$^{\downarrow 0.4}$
w/o DTC (lower)	0.71M	88.2$^{\downarrow 0.2}$

that our FR-GC achieves good performance whether we use $\mathcal{R}_1$ or $\mathcal{R}_2$. This shows that our feature reconstruction is a general idea, and it can be expected that more advanced feature reconstruction functions will lead to better recognition performance. (2) Comparing models C and D, we find that our model is insensitive to the reduction factor r. The recognition accuracy of model C is 0.1% higher than that of model A but brings additional 0.26M parameters, so we still choose model A as the main model. (3) Comparing models E and F, we find that the different choice of the activation function ξ has a significant influence on the final model performance. Notably, model F shows a significant performance degradation compared with model A after $Softmax(\cdot)$ is chosen as the activation function, probably because the forced sparsification of $Softmax(\cdot)$ cannot effectively fit the relationship between the spatiotemporal nodes. This also supports that $Tanh(\cdot)$ is the most appropriate activation function to represent the spatiotemporal relationship between nodes.

4.3.3. MS-DTC module

Effectiveness of MS-DTC. To validate the effectiveness of our MS-DTC on temporal modeling, we evaluate the recognition performance by removing the components of our MS-DTC. The experimental results are shown in Table 7. The baseline w/o MS-DTC indicates that the MS-DTC in the baseline is removed and replaced with the vanilla temporal convolution (Yan et al., 2018). The baseline w/o DTC indicates that the multi-scale architecture is retained, but our DTC is replaced with the vanilla temporal convolution. We observe that both the baseline w/o MS-DTC and the baseline w/o DTC show performance degradation, in which the parameters of the baseline w/o MS-DTC are twice that of the baseline due to the removal of the multi-scale architecture. This shows that our MS-DTC can effectively capture temporal context even with a small number of parameters. Notably, the performance degradation of the baseline w/o DTC (upper) is more obvious compared with the baseline w/o DTC (lower), which denotes that the temporal features learned in the low-frequency temporal domain are more important for action recognition.

Influence of kernel size. The receptive field in the temporal domain is one of the most important factors affecting the performance of our MS-DTC. The kernel size largely determines the capability of MS-DTC to capture temporal context. To evaluate the effect of the kernel size on performance, we explore several different sets of the kernel sizes. The experimental results are shown in Table 8. We observe that model A can still perform well with the small kernel because the dual-domain

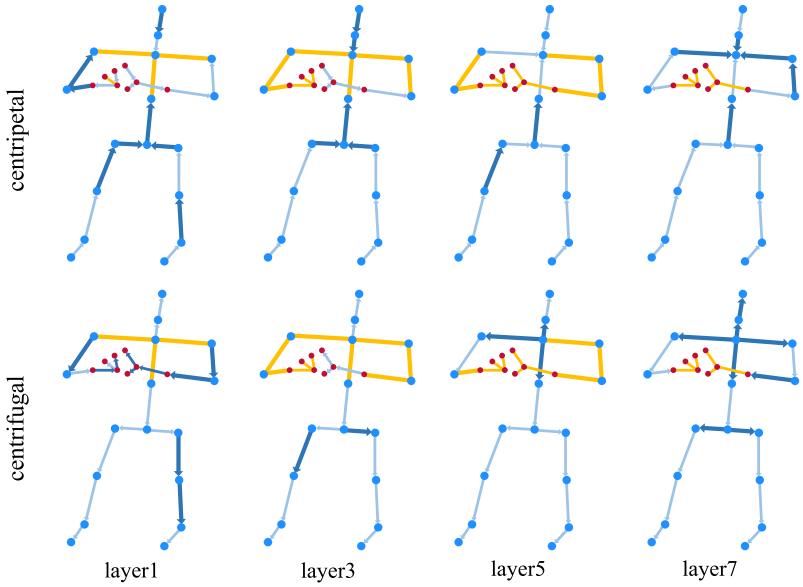


Fig. 5. Visualization of enhanced topology partitions for "rub two hands together" action. The dark blue edge indicates that the TPE module enhances a similar type of edge on it. The orange edge without arrow in the figure is a bi-directional edge, indicating that the TPE module supplements a complementary type of edge on it.

Table 8
Comparisons of model accuracy under different MS-DTC kernel sizes.

Methods	Kernel size	Param.	Acc (%)
Baseline	5 × 1	0.72M	88.4
A	3 × 1	0.71M	88.3
B	7 × 1	0.73M	88.4
C	9 × 1	0.73M	88.1

learning mechanism of our DTC effectively expands the receptive field. Comparing models B and C, our MS-DTC does not bring performance gains with the larger kernels. Especially for model C, there is even a performance penalty with the too large kernel. This may be an over-smoothing phenomenon in the temporal domain.

4.4. Visualization and discussion

4.4.1. Visualization of enhanced topology partitions

We mention in Section 1 that the baseline is unable to accurately recognize the "rub two hands together" action, which is strongly related to the co-effect of centripetal and centrifugal edges. Next, we will analyze this action in the experiment and visualize the enhanced skeleton graph to explain the function of our TPE module.

Experimenting on the X-Sub benchmark of the NTU RGB+D 60 dataset, the accuracy of the baseline in recognizing "rub two hands together" is 82.9%, while our FR-GCN achieves a higher accuracy of 91.3%. This indicates that our TPE module indeed effectively complements the co-occurrence relationship between the centripetal and centrifugal edges. To see the enhanced topology learned by our TPE module more clearly, we draw the enhanced edges relevant to the centripetal and centrifugal skeleton graphs, as shown in Fig. 5. The dark blue edge in the figure indicates that the TPE module enhances a similar type of edge on it, leading to an improved raw topology semantic. The orange edge without arrow in the figure is a bi-directional edge, indicating that the TPE module supplements a complementary type of edge on it. This allows models to learn both centripetal and centrifugal moves in a single topology, adaptively matching the features of different actions. To accurately recognize the action "rub two hands together", the model needs to focus on the anterior-posterior coordinated motion of the hands, capturing the co-occurrence features of centrifugal and centripetal moves. As shown in Fig. 5, our TPE modules have different enhanced semantics in different layers of the model. In the lower layers of the model, the TPE module primarily adds original-type edges that facilitate the convergence of information towards the hands. In the middle layers, additional complementary edges emerge in the upper body, enabling full circulation of information between the arms. In the upper layers, complementary edges mainly converge at the hands, allowing the model to prioritize the complementary motion of the hands. This confirms the effectiveness and necessity of the proposed TPE module.

4.4.2. Failure case discussion

We further explore the specific recognition results of the baseline and our FR-GCN on the X-Sub benchmark of the NTU RGB+D 60 dataset. We first draw the confusion matrix for the baseline, where the selected actions are those with low recognition accuracy. The confusion matrix of the baseline is shown in Fig. 6 (top). We observe that the baseline does not perform well in recognizing the actions such as "reading", "writing", "playing with phone", and "typing on a keyboard". There is a 13% probability that the baseline will recognize "reading" as "writing". For the "writing" action, 17% of the samples are misclassified as "reading", while another 17% are misclassified as "typing on a keyboard". In conclusion, the baseline is prone to misclassification when recognizing hard actions.

Then, we select the same hard actions mentioned above as analysis objects for our FR-GCN. The confusion matrix of our FR-GCN is shown in Fig. 6 (bottom). We observe that the recognition accuracy of our FR-GCN outperforms the baseline on almost all actions, and our FR-GCN improves the recognition accuracy by an average of 4.9% on these hard actions. In particular, our FR-GCN outperforms the baseline by 9% and 10% in recognizing "reading" and "typing on a keyboard". Moreover, the proposed FR-GCN brings an 8% performance improvement in recognizing "taking a selfie" and effectively suppresses misclassifying it as "pointing to something with finger".

Although our FR-GCN can accurately recognize most of the actions, there are still some actions that are difficult to recognize. Compared with the baseline, our FR-GCN achieves certain performance improvement in recognizing "writing", "playing with phone", and "typing on a keyboard", but the recognition accuracy is still unsatisfactory. Especially for the "writing" action, our FR-GCN still has a high probability to misclassify it as "reading" and "typing on a keyboard".

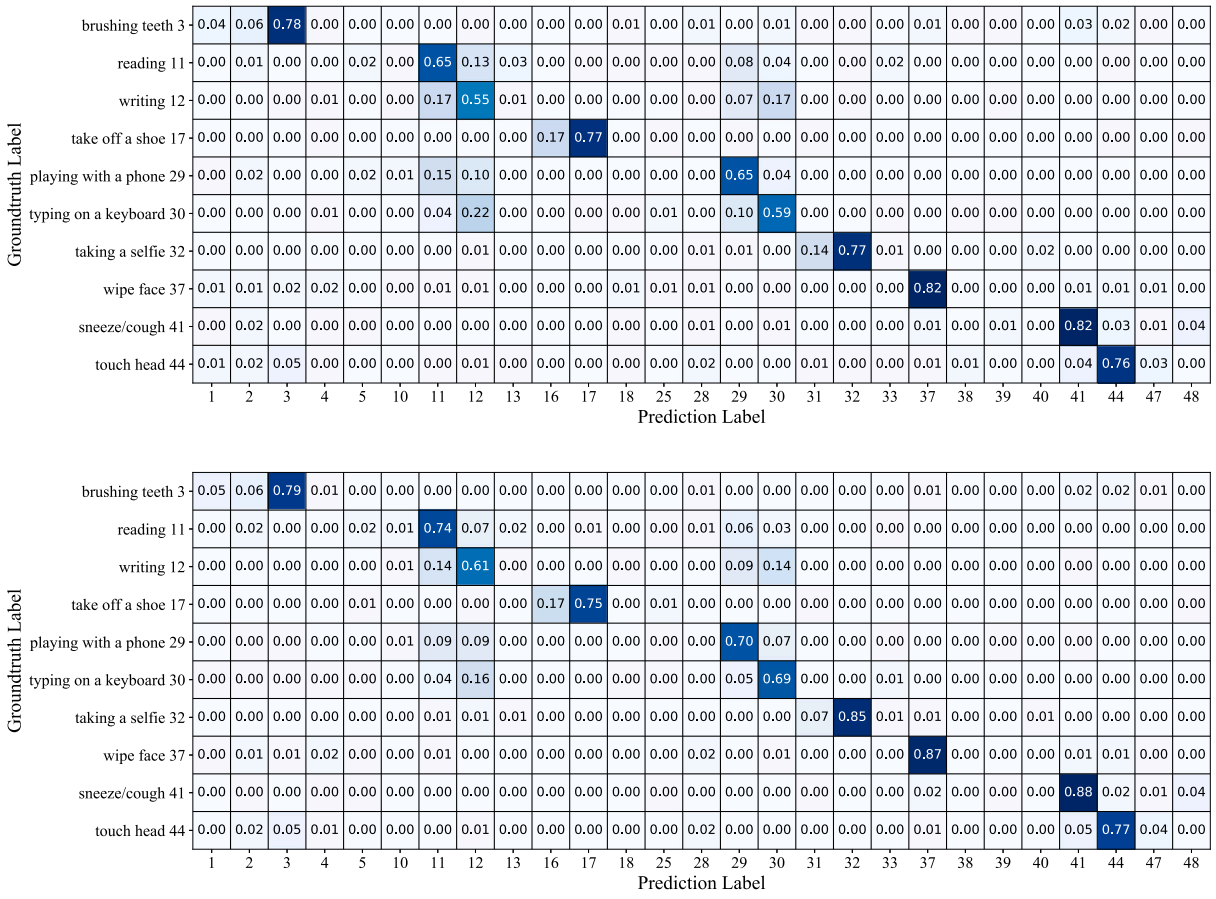


Fig. 6. Confusion matrices of the baseline (top) and our FR-GCN (bottom) on the X-Sub benchmark of the NTU RGB+D 60 dataset. It compares the accuracy of the models in recognizing hard actions.

We select six typical actions to analyze in detail the reasons for the recognition failure. These actions are "brushing teeth", "reading", "taking off a shoe", "taking a selfie", "wipe face" and "sneeze/cough", and snapshots of the action samples are shown in Fig. 7. We can observe that the actions of "brushing teeth", "reading" and "wiping face" involve minimal movements, and the motion changes are mainly concentrated on the hands. The action of "reading" is particularly challenging to recognize, mainly due to its limited variation, which is simply "turning pages". Additionally, there is an action labeled as "writing" in the dataset that bears a striking resemblance to "reading", adding further difficulty to the recognition process. The action of "taking off a shoe" involves large movements, but the recognition effect of the model is still not ideal. This is primarily due to the complex deformation of the human body that occurs during this action, as well as the fact that it is opposite in time sequence to the "wear a shoe" action. In summary, the failure of the model to recognize hard actions is mainly caused by two reasons: (1) the capability of our model to capture subtle spatial and temporal changes needs to be further improved and (2) these hard actions are mainly performed by the slight shaking of two hands, but the NTU datasets lack comprehensive descriptions of the finger joints. Therefore, it is still challenging to recognize such subtle actions.

4.5. Comparisons with state-of-the-art methods

We compare our full model with state-of-the-art methods on the NTU RGB+D 60 dataset and the NTU RGB+D 120 dataset. We first compare non-graph methods (Wang and Wang, 2017; Li et al., 2018a,b; Zhang et al., 2019) and graph-based methods (Yan et al., 2018; Shi et al., 2019b; Liu et al., 2020; Chen et al., 2021a; Ye et al., 2020; Song et al., 2022; Si et al., 2019; Cheng et al., 2020b; Shi et al., 2020; Zhang et al., 2020; Shi et al., 2019a; Peng et al., 2020; Cheng et al., 2020a; Song et al., 2020; Chen et al., 2021c,b) on the NTU RGB+D 60 dataset, and the results are shown in Table 9. Then, we compare non-graph methods (Liu et al., 2016; Ke et al., 2018; Liu et al., 2019b) and graph-based methods (Yan et al., 2018; Shi et al., 2019b; Liu et al., 2020; Chen et al., 2021a; Ye et al., 2020; Song et al., 2022; Cheng et al., 2020b; Zhang et al., 2020; Cheng et al., 2020a; Song et al., 2020; Chen et al., 2021c,b) on the NTU RGB+D 120 dataset, and the results are shown in Table 10.

From Table 9, our FR-GCN outperforms all state-of-the-art models under nearly both the X-Sub and X-View benchmarks of the NTU RGB+D 60 dataset. Especially under the X-Sub benchmark, our FR-GCN achieves a recognition accuracy of 92.7%, outperforming all other state-of-the-art methods. From Table 10, our FR-GCN still shows good performance. The proposed FR-GCN achieves a recognition accuracy of 89.1% under the X-Sub benchmark of the NTU RGB+D 120 dataset, which outperforms all other methods. And our full model also achieves competitive performance on the X-Set benchmark.

Notably, compared with the three explicit spatiotemporal methods (AGC-LSTM Si et al., 2019, Shift-GCN Cheng et al., 2020b, MS-G3D Liu et al., 2020) mentioned in Section 2.2, our FR-GCN significantly outperforms them under all the evaluation benchmarks of the NTU RGB+D 60 dataset and the NTU RGB+D 120 dataset.

To further evaluate our FR-GCN, we select four models (ST-GCN Yan et al., 2018, Shift-GCN Cheng et al., 2020b, MS-G3D Liu et al., 2020, and CTR-GCN Chen et al., 2021b) based on the comparison results in Tables 9 and 10 for a more comprehensive comparative analysis with our full model. Our reasons for selecting these 4 models are as follows. (1) ST-GCN, as the most classical baseline, can well reflect

Fig. 7. Snapshots of hard actions from the X-Sub benchmark of the NTU RGB+D 60 dataset. The selected actions are "brushing teeth", "reading", "taking off a shoe", "taking a selfie", "wipe face" and "sneeze/cough". (Zoom in for best view).

the benchmark performance on the datasets. (2) Shift-GCN and MS-G3D are both the explicit spatiotemporal methods, which belong to the same class as our FR-GCN. (3) CTR-GCN has similar performance to our FR-GCN, and we need to further analyze the advantages of the proposed FR-GCN. It should be noted that this part of the comparison experiments are conducted on two benchmarks of the NTU RGB+D 60 dataset. To reflect the true performance of the models and ensure the fairness of the experiments, we employ the uniform data preprocessing method for all models in this part of the comparison experiments.

We conduct comparison experiments on the X-Sub benchmark first, and the results are shown in Table 11. To reassess all models, we introduce three new metrics: macro-precision, macro-recall, and macro-F1 score (Opitz and Burst, 2019). These metrics are multi-classification versions of precision, recall, and F1-score. The introduction of these three new metrics can better represent the overall performance of the models. Additionally, we analyze the time and space complexity of the models by calculating their FLOPs (Floating Point Operations) and parameter number. It should be noted that the FLOPs here only indicate the time complexity of the model to process one sample.

From Table 11, we can observe that our FR-GCN achieves the best performance while maintaining low time complexity and space complexity. Compared with CTR-GCN, its FLOPs are 42.2% higher than our FR-GCN, and the parameter number is also 40.6% higher than our FR-GCN, but our FR-GCN achieves better results in the comparison experiments. Especially in terms of macro-precision, our full model is 0.6% higher than CTR-GCN, proving that our model is more stable and superior in discriminating a single class. MS-G3D and Shift-GCN both construct modules for explicit extraction of spatiotemporal features, and the design idea is similar to ours. However, these two models represent two extremes that cannot balance performance and complexity at the same time. MS-G3D pursues the most extreme explicit extraction capability of spatiotemporal features. The result in Table 5 shows that MS-G3D can still perform well even if the temporal modules in the full model are completely removed. But among all the comparison models, MS-G3D has the highest FLOPs with the most parameters. Its FLOPs are 316.7% higher than our FR-GCN, and also the number of parameters is 215.8% more than our FR-GCN. In small computing devices, such time and space complexities are intolerable. Shift-GCN pursues the ultimate low complexity with only 1.03G FLOPs and 0.69M parameters, but its performance is also the worst, underperforming the baseline (ST-GCN) in all three metrics. To sum up, our FR-GCN has the best overall performance. The feature reconstruction mechanism of our FR-GCN can ensure that the model has strong ability to explicitly extract spatiotemporal features. Meanwhile, relying on the lightweight design of MS-DTC and TPE modules, the model can excellently complete the recognition task with lower FLOPs and fewer parameters.

We also conduct comparison experiments on the X-View benchmark, and the results are shown in Table 12. From the experimental results, it can be observed that our FR-GCN can still maintain good recognition performance with low time complexity and space complexity. It is particularly noteworthy that our FR-GCN outperforms the CTR-GCN on three new metrics, demonstrating the greater robustness of our model.

5. Conclusion

In this work, we combine the proposed TPE module, FR-GC module, and MS-DTC module to construct a novel feature reconstruction graph convolutional network (FR-GCN) for skeleton-based action recognition. The proposed TPE module re-partitions the original skeleton graph in a

Table 9
Classification accuracy comparisons with state-of-the-art methods on the NTU RGB+D 60 dataset. The **best** and second best values are highlighted in bold and underline, respectively.

Methods	Publisher	NTU RGB+D 60	
		X-Sub (%)	X-View (%)
Two-stream RNN (Wang and Wang, 2017)	CVPR17	71.3	79.5
Ind-RNN (Li et al., 2018a)	CVPR18	81.8	88.0
HCN (Li et al., 2018b)	IJCAI18	86.5	91.1
VA-NN (Zhang et al., 2019)	TPAMI19	89.4	95.0
ST-GCN (Yan et al., 2018)	AAAI18	81.5	88.3
2s-AGCN (Shi et al., 2019b)	CVPR19	88.5	95.1
AGC-LSTM (Si et al., 2019)	CVPR19	89.2	95.0
DGNN (Shi et al., 2019a)	CVPR19	89.9	96.1
NAS-GCN (Peng et al., 2020)	AAAI20	89.4	95.7
SGN (Zhang et al., 2020)	CVPR20	89.0	94.5
Shift-GCN (Cheng et al., 2020b)	CVPR20	90.7	96.5
MS-G3D (Liu et al., 2020)	CVPR20	91.5	96.2
DC-GCN+ADG (Cheng et al., 2020a)	ECCV20	90.8	96.6
PA-ResGCN-B19 (Song et al., 2020)	ACMMM20	90.9	96.0
Dynamic GCN (Ye et al., 2020)	ACMMM20	91.5	96.0
MS-AAGCN (Shi et al., 2020)	TIP20	90.0	96.2
MST-GCN (Chen et al., 2021a)	AAAI21	91.5	96.6
DualHead-Net (Chen et al., 2021c)	ACMMM21	92.0	96.6
CTR-GCN (Chen et al., 2021b)	ICCV21	92.4	**96.8**
EfficientGCN-B4 (Song et al., 2022)	TPAMI22	91.7	95.7
FR-GCN (Joint Only)	–	90.1	95.1
FR-GCN (Bone Only)	–	90.4	94.6
FR-GCN (Joint+Bone)	–	92.0	96.3
FR-GCN	–	**92.7**	96.7

Table 10
Classification accuracy comparisons with state-of-the-art methods on the NTU RGB+D 120 dataset. The **best** and second best values are highlighted in bold and underline, respectively.

Methods	Publisher	NTU RGB+D 120	
		X-Sub (%)	X-Set (%)
ST-LSTM (Liu et al., 2016)	ECCV16	55.7	57.9
RotClips+MTCNN (Ke et al., 2018)	TIP18	62.2	61.8
FSNet (Liu et al., 2019b)	TPAMI19	59.9	62.4
ST-GCN (Yan et al., 2018)	AAAI18	70.7	73.2
2s-AGCN (Shi et al., 2019b)	CVPR19	82.9	84.9
SGN (Zhang et al., 2020)	CVPR20	79.2	81.5
Shift-GCN (Cheng et al., 2020b)	CVPR20	85.9	87.6
MS-G3D (Liu et al., 2020)	CVPR20	86.9	88.4
DC-GCN+ADG (Cheng et al., 2020a)	ECCV20	86.5	88.1
PA-ResGCN-B19 (Song et al., 2020)	ACMMM20	87.3	88.3
Dynamic GCN (Ye et al., 2020)	ACMMM20	87.3	88.6
MST-GCN (Chen et al., 2021a)	AAAI21	87.5	88.8
DualHead-Net (Chen et al., 2021c)	ACMMM21	88.2	89.3
CTR-GCN (Chen et al., 2021b)	ICCV21	88.9	**90.6**
EfficientGCN-B4 (Song et al., 2022)	TPAMI22	88.3	89.1
FR-GCN (Joint Only)	–	84.8	86.5
FR-GCN (Bone Only)	–	86.1	87.2
FR-GCN (Joint+Bone)	–	88.8	89.7
FR-GCN	–	**89.1**	90.2

Table 11
Comparisons with state-of-the-art methods on the X-Sub benchmark of the NTU RGB+D 60 dataset in FLOPs ($\times 10^9$), parameter number ($\times 10^6$), macro-precision (%), macro-recall (%) and macro-F1 score (%). The **best** and second best values are highlighted in bold and underline, respectively.

Methods	FLOPs. (G)	Param. (M)	X-Sub Benchmark		
			Macro-precision (%)	Macro-recall (%)	Micro-F1 score (%)
ST-GCN[a] (Yan et al., 2018)	6.96	3.09	91.1	90.8	90.9
Shift-GCN[a] (Cheng et al., 2020b)	**1.03**	**0.69**	90.7	90.5	90.5
MS-G3D[a] (Liu et al., 2020)	10.46	3.19	91.9	91.8	91.8
CTR-GCN[a] (Chen et al., 2021b)	3.57	1.45	92.4	92.3	92.3
FR-GCN	2.51	1.01	**93.0**	**92.7**	**92.8**

[a]These models are implemented based on the released codes.

data-driven manner and dynamically enhances the new partitions into the predefined ones, which is the first attempt to enhance the topology partitions from the graph decomposition perspective. It can enhance the topological representation of each partition and improve the flexibility

Table 12
Comparisons with state-of-the-art methods on the X-View benchmark of the NTU RGB+D 60 dataset in FLOPs ($\times 10^9$), parameter number ($\times 10^6$), macro-precision (%), macro-recall (%) and macro-F1 score (%). The **best** and second best values are highlighted in bold and underline, respectively.

Methods	FLOPs. (G)	Param. (M)	X-View Benchmark		
			Macro-precision (%)	Macro-recall (%)	Micro-F1 score (%)
ST-GCN[a] (Yan et al., 2018)	6.96	3.09	95.1	95.0	95.0
Shift-GCN[a] (Cheng et al., 2020b)	**1.03**	**0.69**	95.9	95.7	95.8
MS-G3D[a] (Liu et al., 2020)	10.46	3.19	96.0	96.0	96.0
CTR-GCN[a] (Chen et al., 2021b)	3.57	1.45	96.5	96.4	96.4
FR-GCN	2.51	1.01	**96.8**	**96.7**	**96.7**

[a]These models are implemented based on the released codes.

of the model in handling different actions. Unlike other GCs, the proposed FR-GC will reconstruct the original input features before feature extraction, and the reconstructed features have stronger perception in the spatiotemporal domain. The reconstruction approach enables the model to simultaneously extract spatial configurations and global temporal contexts in a single module, showing stronger capability for spatiotemporal feature extraction. Moreover, the proposed MS-DTC can further enhance the temporal perception of the model, capturing the extended temporal context. On two challenging large-scale datasets (NTU RGB+D 60 and NTU RGB+D 120), the proposed FR-GCN achieves remarkable results. Especially under the cross-subject benchmarks of the two datasets, the proposed FR-GCN achieves new state-of-the-art performance. This work suggests that feature reconstruction is a general idea, and more powerful feature reconstruction approaches may further improve the capability of the model to extract spatiotemporal features. Adaptive partition of the topology is an interesting topic, and we will explore more powerful topology partition methods in the future, which may make the feature extraction more reasonable.

Intelligent factory production has become the development trend of the future manufacturing industry. Skeleton-based action recognition is a promising technology for various industrial applications, offering tangible benefits to the industry. Our model offers excellent performance with low computational complexity and lightweight storage usage. These advantages make the model very suitable for deployment on the computing side of industrial equipment, allowing for a variety of intelligent detection and identification capabilities. For instance, in the context of human–computer interaction and intelligent operation, skeleton-based action recognition technology can enable robots to collaborate with employees in completing production operations by identifying gestures, action operations, and other relevant information. This can significantly enhance both the efficiency and quality of production. When it comes to safety supervision, skeleton-based action recognition technology can analyze the movement trajectory and action status of employees to predict potential safety hazards and avoid blundering safety accidents. In the field of intelligent transportation, skeleton-based action recognition technology can enable the monitoring and evaluation of driving behavior by analyzing the motion state of the driver. For example, it can detect signs of driver fatigue and identify instances of dangerous driving behavior, thus ensuring safe driving. In summary, skeleton-based action recognition technology offers practical benefits for various industrial scenarios. It can promote the organic integration of industry and intelligence, enhance the efficiency of human-machine cooperation, and increase productivity levels. As an emerging field with broad application prospects and significant market demand, this technology holds enormous potential for the future.

CRediT authorship contribution statement

Junhao Huang: Conceptualization, Methodology, Software, Writing, Formal analysis, Visualization. **Ziming Wang:** Validation, Investigation, Data curation. **Jian Peng:** Supervision, Project administration, Funding acquisition. **Feihu Huang:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the Key R&D Program of Sichuan Province, China (2022YFG0034, 22ZDZX0021) and the Cooperative Program of Sichuan University and Yibin (2020CDYB-30).

References

Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P., 2017. Geometric deep learning: going beyond euclidean data. IEEE Signal Process. Mag. 34 (4), 18–42.

Bruna, J., Zaremba, W., Szlam, A., LeCun, Y., 2013. Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203.

Cao, Z., Simon, T., Wei, S.-E., Sheikh, Y., 2017. 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.

Chen, C., Jafari, R., Kehtarnavaz, N., 2017. A survey of depth and inertial sensor fusion for human action recognition. Multimedia Tools Appl. 76 (3), 4405–4425.

Chen, Z., Li, S., Yang, B., Li, Q., Liu, H., 2021a. 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.

Chen, Y., Zhang, Z., Yuan, C., Li, B., Deng, Y., Hu, W., 2021b. Channel-wise topology refinement graph convolution for skeleton-based action recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13359–13368.

Chen, T., Zhou, D., Wang, J., Wang, S., Guan, Y., He, X., Ding, E., 2021c. Learning multi-granular spatio-temporal graph network for skeleton-based action recognition. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 4334–4342.

Cheng, K., Zhang, Y., Cao, C., Shi, L., Cheng, J., Lu, H., 2020a. Decoupling gcn with dropgraph module for skeleton-based action recognition. In: European Conference on Computer Vision. Springer, pp. 536–553.

Cheng, K., Zhang, Y., He, X., Chen, W., Cheng, J., Lu, H., 2020b. 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.

Chi, H.-g., Ha, M.H., Chi, S., Lee, S.W., Huang, Q., Ramani, K., 2022. InfoGCN: Representation learning for human skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20186–20196.

Defferrard, M., Bresson, X., Vandergheynst, P., 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems, Vol. 29.

Ding, C., Wen, S., Ding, W., Liu, K., Belyaev, E., 2022. Temporal segment graph convolutional networks for skeleton-based action recognition. Eng. Appl. Artif. Intell. 110, 104675.

Duan, H., Zhao, Y., Chen, K., Lin, D., Dai, B., 2022. Revisiting skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2969–2978.

Duvenaud, D.K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P., 2015. Convolutional networks on graphs for learning molecular fingerprints. In: Advances in Neural Information Processing Systems, Vol. 28.

Guo, S., Lin, Y., Feng, N., Song, C., Wan, H., 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33. pp. 922–929.

Hamilton, W., Ying, Z., Leskovec, J., 2017. Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, Vol. 30.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

He, T., Zhang, Z., Zhang, H., Zhang, Z., Xie, J., Li, M., 2019. Bag of tricks for image classification with convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 558–567.

Henaff, M., Bruna, J., LeCun, Y., 2015. Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

Hu, L., Liu, S., Feng, W., 2022. Spatial temporal graph attention network for skeleton-based action recognition. arXiv preprint arXiv:2208.08599.

Hussein, M.E., Torki, M., Gowayyed, M.A., El-Saban, M., 2013. Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In: Twenty-Third International Joint Conference on Artificial Intelligence.

Ke, Q., Bennamoun, M., An, S., Sohel, F., Boussaid, F., 2017. 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.

Ke, Q., Bennamoun, M., An, S., Sohel, F., Boussaid, F., 2018. Learning clip representations for skeleton-based 3d action recognition. IEEE Trans. Image Process. 27 (6), 2842–2855.

Ke, L., Peng, K.-C., Lyu, S., 2022. Towards to-at spatio-temporal focus for skeleton-based action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36. pp. 1131–1139.

Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations.

Koniusz, P., Cherian, A., Porikli, F., 2016. Tensor representations via kernel linearization for action recognition from 3d skeletons. In: European Conference on Computer Vision. Springer, pp. 37–53.

Li, M., Chen, S., Chen, X., Zhang, Y., Wang, Y., Tian, Q., 2019. Actional-structural graph convolutional networks for skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3595–3603.

Li, Y., Ji, B., Shi, X., Zhang, J., Kang, B., Wang, L., 2020. Tea: Temporal excitation and aggregation for action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 909–918.

Li, S., Li, W., Cook, C., Zhu, C., Gao, Y., 2018a. 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.

Li, C., Zhong, Q., Xie, D., Pu, S., 2018b. Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. pp. 786–792.

Liu, J., Shahroudy, A., Perez, M., Wang, G., Duan, L.-Y., Kot, A.C., 2019a. Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding. IEEE Trans. Pattern Anal. Mach. Intell. 42 (10), 2684–2701.

Liu, J., Shahroudy, A., Wang, G., Duan, L.-Y., Kot, A.C., 2019b. Skeleton-based online action prediction using scale selection network. IEEE Trans. Pattern Anal. Mach. Intell. 42 (6), 1453–1467.

Liu, J., Shahroudy, A., Xu, D., Wang, G., 2016. Spatio-temporal lstm with trust gates for 3d human action recognition. In: European Conference on Computer Vision. Springer, pp. 816–833.

Liu, Z., Zhang, H., Chen, Z., Wang, Z., Ouyang, W., 2020. 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.

Liu, Y., Zhang, H., Xu, D., He, K., 2022. Graph transformer network with temporal kernel attention for skeleton-based action recognition. Knowl.-Based Syst. 240, 108146.

Ma, Q., Shen, L., Chen, E., Tian, S., Wang, J., Cottrell, G.W., 2017. Walking walking walking: Action recognition from action echoes. In: IJCAI. pp. 2457–2463.

Nguyen, T., Grishman, R., 2018. Graph convolutional networks with argument-aware pooling for event detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32.

Niepert, M., Ahmed, M., Kutzkov, K., 2016. Learning convolutional neural networks for graphs. In: International Conference on Machine Learning. PMLR, pp. 2014–2023.

Opitz, J., Burst, S., 2019. Macro f1 and macro f1. arXiv preprint arXiv:1911.03347.

Peng, W., Hong, X., Chen, H., Zhao, G., 2020. 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.

Peng, K., Roitberg, A., Yang, K., Zhang, J., Stiefelhagen, R., 2023. Delving deep into one-shot skeleton-based action recognition with diverse occlusions. IEEE Trans. Multimed..

Plizzari, C., Cannici, M., Matteucci, M., 2021. Skeleton-based action recognition via spatial and temporal transformer networks. Comput. Vis. Image Underst. 208, 103219.

Qiu, H., Hou, B., Ren, B., Zhang, X., 2023. Spatio-temporal segments attention for skeleton-based action recognition. Neurocomputing 518, 30–38.

Rahmani, H., Bennamoun, M., 2017. Learning action recognition model from depth and skeleton videos. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5832–5841.

Sagayam, K.M., Hemanth, D.J., 2017. Hand posture and gesture recognition techniques for virtual reality applications: a survey. Virtual Real. 21 (2), 91–107.

Shahroudy, A., Liu, J., Ng, T.-T., Wang, G., 2016. 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.

Shi, L., Zhang, Y., Cheng, J., Lu, H., 2019a. Skeleton-based action recognition with directed graph neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7912–7921.

Shi, L., Zhang, Y., Cheng, J., Lu, H., 2019b. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12026–12035.

Shi, L., Zhang, Y., Cheng, J., Lu, H., 2020. Skeleton-based action recognition with multi-stream adaptive graph convolutional networks. IEEE Trans. Image Process. 29, 9532–9545.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A., 2011. Real-time human pose recognition in parts from single depth images. In: CVPR 2011. IEEE, pp. 1297–1304.

Si, C., Chen, W., Wang, W., Wang, L., Tan, T., 2019. An attention enhanced graph convolutional lstm network for skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1227–1236.

Song, Y.-F., Zhang, Z., Shan, C., Wang, L., 2020. 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.

Song, Y.-F., Zhang, Z., Shan, C., Wang, L., 2022. Constructing stronger and faster baselines for skeleton-based action recognition. IEEE Trans. Pattern Anal. Mach. Intell..

Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M., 2015. Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4489–4497.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2018. Graph attention networks. In: International Conference on Learning Representations.

Vemulapalli, R., Arrate, F., Chellappa, R., 2014. 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.

Wang, X., Dai, Y., Gao, L., Song, J., 2022. Skeleton-based action recognition via adaptive cross-form learning. In: Proceedings of the 30th ACM International Conference on Multimedia. pp. 1670–1678.

Wang, L., Huynh, D.Q., Koniusz, P., 2019. A comparative review of recent kinect-based action recognition algorithms. IEEE Trans. Image Process. 29, 15–28.

Wang, P., Li, W., Gao, Z., Zhang, J., Tang, C., Ogunbona, P.O., 2015. Action recognition from depth maps using deep convolutional neural networks. IEEE Trans. Hum.-Mach. Syst. 46 (4), 498–509.

Wang, H., Wang, L., 2017. Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 499–508.

Wang, Y.-X., Zhang, Y.-J., 2012. Nonnegative matrix factorization: A comprehensive review. IEEE Trans. Knowl. Data Eng. 25 (6), 1336–1353.

Wen, Y.-H., Gao, L., Fu, H., Zhang, F.-L., Xia, S., Liu, Y.-J., 2022. Motif-GCNs with local and non-local temporal blocks for skeleton-based action recognition. IEEE Trans. Pattern Anal. Mach. Intell..

Xie, C., Li, C., Zhang, B., Chen, C., Han, J., Liu, J., 2018. Memory attention networks for skeleton-based action recognition. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. pp. 1639–1645.

Xu, K., Ye, F., Zhong, Q., Xie, D., 2022. Topology-aware convolutional neural network for efficient skeleton-based action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36. pp. 2866–2874.

Yan, S., Xiong, Y., Lin, D., 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence.

Yang, X., Tian, Y.L., 2012. Eigenjoints-based action recognition using naive-bayes-nearest-neighbor. In: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. IEEE, pp. 14–19.

Ye, F., Pu, S., Zhong, Q., Li, C., Xie, D., Tang, H., 2020. Dynamic gcn: Context-enriched topology learning for skeleton-based action recognition. In: Proceedings of the 28th ACM International Conference on Multimedia. pp. 55–63.

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W.L., Leskovec, J., 2018. Graph convolutional neural networks for web-scale recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 974–983.

Zhang, Z., 2012. Microsoft kinect sensor and its effect. IEEE Multimed. 19 (2), 4–10.

Zhang, P., Lan, C., Xing, J., Zeng, W., Xue, J., Zheng, N., 2017. 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.

Zhang, P., Lan, C., Xing, J., Zeng, W., Xue, J., Zheng, N., 2019. View adaptive neural networks for high performance skeleton-based human action recognition. IEEE Trans. Pattern Anal. Mach. Intell. 41 (8), 1963–1978.

Zhang, P., Lan, C., Zeng, W., Xing, J., Xue, J., Zheng, N., 2020. 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.

Zhou, H., Liu, Q., Wang, Y., 2023. Learning discriminative representations for skeleton based action recognition. arXiv preprint arXiv:2303.03729.