SKELETON-BASED HUMAN ACTION RECOGNITION USING SPATIO-TEMPORAL ATTENTION GRAPH CONVOLUTIONAL NETWORKS

Lappeenranta-Lahti University of Technology LUT

Master's Program in Computational Engineering, Master's Thesis

2022

Manh Cuong Le

Examiners: Associate Professor Xin Liu
Professor Heikki Kälviäinen

ABSTRACT

Lappeenranta-Lahti University of Technology LUT
School of Engineering Science
Computational Engineering

Manh Cuong Le

SKELETON-BASED HUMAN ACTION RECOGNITION USING SPATIO-TEMPORAL ATTENTION GRAPH CONVOLUTIONAL NETWORKS

Master's thesis
2022
65 pages, 48 figures, 3 tables, 0 appendices
Examiners: Associate Professor Xin Liu and Professor Heikki Kälviäinen

Keywords: computer vision, action recognition, skeleton data, graph convolutional network, attention mechanism

In human action recognition, skeleton-based data is an effective way to represent the performing actions. Many recent studies focus on this line of research and significant results have been achieved. In this study, an attention graph convolutional network model for skeleton-based action recognition is proposed to improve the previous methods. The model consists of two components: spatial and temporal modeling. First, spatial features are captured by combining self-attentions with prior information on human kinetics. Then, time-dependent features across frames can be captured using temporal self-attentions and multi-scale convolutions. By utilizing self-attention mechanisms, a small neural network architecture that can effectively model the skeleton data can be created. The captured information is then globally fused to generate the final representation for the classification of human actions. The proposed method achieved competitive classification results compared to state-of-the-art methods such as MS-G3D and CTR-GCN, on the NTU-RGB+D60 dataset and the NTU-RGB+D120 dataset.

ACKNOWLEDGEMENTS

I would like to thank my supervisors Associate Professor Xin Liu and Professor Heikki Kälviäinen for providing me with great consultation and feedback during my thesis process.

I am also grateful to my family, relatives, and friends that supported me during my studies at Lappeenranta-Lahti University of Technology (LUT).

Lappeenranta, May 25, 2022

Manh Cuong Le

LIST OF ABBREVIATIONS

2s-AGCN	Two-stream Adaptive Graph Convolutional Networks
AS-GCN	Actional-Structural Graph Convolutional Networks
CeN	Context-encoding Network
CNN	Convolutional Neural Network
CTR-GCN	Channel-wise Topology Refinement Graph Convolutional Networks
DCK	Dynamics Compatibility Kernel
FLOPS	Floating point Operations Per Second
FN	false negative
FP	false positive
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GLU	Gated Linear Uint
GNN	Graph Neural Networks
HOD	Histogram of Oriented Displacement
LSTM	Long-Short Term Memory
MLP	Multi-Layer Perceptron
MPNN	Message Passing Neural Networks
MS-G3D	Multi-Scale 3-dimensional Graph Neural Networks
MTLN	Multi-Task Learning Network
NBNN	Naïve Bayes Nearest Neighbor
NN	Neural Networks
PCA	Principal Component Analysis
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
RGB	Red-Green-Blue
RNN	Recurrent Neural Networks
SCK	Sequence Compatibility Kernel
SDK	Software Development Kit
ST-GCN	Spatial-Temporal Graph Convolutional Networks
STA-GCN	Spatio-Temporal Attention Graph Convolutional Networks
SVM	Support Vector Machine
TCN	Temporal Convolutional Networks
TN	true negative
TP	true positive

CONTENTS

1 INTRODUCTION

1.1 Background

Action is a way through which humans can interact with the surrounding environment, express their intentions, and communicate with one another [1]. Understanding of people's behaviors plays a crucial role in human-to-human interaction and interpersonal relations. In practical surveillance systems, one can rely on the human visual system to analyze and understand the purpose of a person. However, relying on human labor in modern surveillance systems is increasingly expensive and ineffective. Therefore, many scientific areas including Computer Vision and Machine Learning are expanding their research focuses on automatic human action recognition systems [2–4].

Similar to the human visual system, computer algorithms are designed to produce an action label after inspecting a human execution. Traditional techniques for action recognition consist of two main tasks: extraction of high-level features from data, and making predictions on the human actions based on the extracted information. Methods for feature extraction can be classified into two main categories: knowledge-driven and data-driven. Due to the recent development of large-scale datasets and machine learning algorithms, data-driven approaches are increasingly providing better results and becoming state-of-the-art in human action recognition [5].

In recent years, computer vision has achieved new milestones thanks to the advancement of machine learning techniques. However, image-based methods suffer from many difficulties such as illumination changes, environmental noises, and camera perspective [6]. One solution to overcome these problems in human action recognition is to utilize skeleton data [7–10]. As shown in Figure 1, a human pose can be encoded to a network of connected joints and provide a high-level action representation [11, 12].

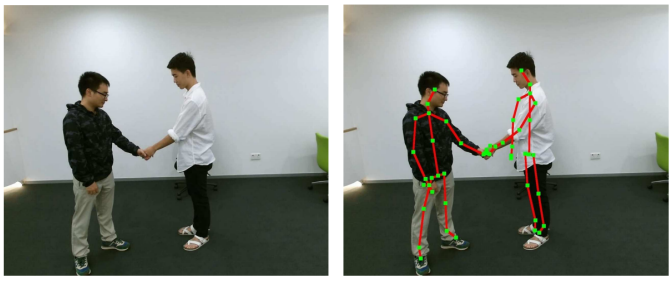


Figure 1. Human action is represented as skeleton graphs. [11]

This thesis focuses on skeleton-based action recognition, which uses a deep learning model to extract features from sequences of skeleton data and make action predictions. The proposed method considers Graph Neural Networks (GNN), combined with attention mechanisms, to model spatial and temporal representations of skeleton data. The encoded feature maps are finally processed by a classifier to predict the action label.

1.2 Objectives and delimitations

The goal of this study is to design and implement a graph-based deep learning model for human action recognition that can automatically extract features and correctly classify the action class from a given sequence of humanoid skeletal data.

The objectives of this research are as follows:

- Make a survey on current skeleton-based human action recognition techniques.
- Collect knowledge and theoretical reasoning of GNN and its variances.
- Propose a GNN-based deep learning model for skeleton-based action recognition.
- Find and prepare available skeleton datasets for testing purpose.
- Train and evaluate the proposed model on the collected datasets.
- Compare the results with other methods using proper metrics.
- Collect findings and make a report on the study.

This thesis is delimited to propose a novel graph-based neural network architecture to make predictions on human skeleton data. The proposed model is expected to produce competitive results with the current state-of-the-art methods.

1.3 Structure of the thesis

The thesis is organized as follows: Chapter 2 provides introductions to skeleton data and existing skeleton-based action recognition methods. Chapter 3 gives an overview of graph neural networks and their related research. Chapter 4 contains the proposal of the novel skeleton-based attention graph convolutional network. Chapter 5 describes the implementation of the proposed model and the experiment results. Discussion and future work are addressed in Chapter 6. Chapter 7 concludes the thesis.

2 SKELETON-BASED ACTION RECOGNITION

2.1 Skeleton data

From a mechanical point of view, human skeleton data includes a set of joints and edges. Joints are the endpoints of human bones, and edges are lines that connect the joints. Figure 2 illustrates the skeleton encoding of human body. The construction of skeleton data can represent the geometrical structures of human action [13,14]. Therefore, skeleton sequences can provide a very informative encoding of the human joints' trajectories for human motion through spatial and temporal dependencies.

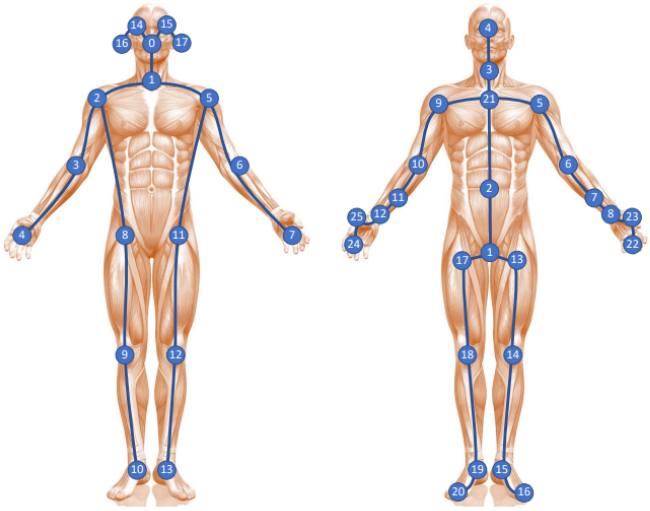
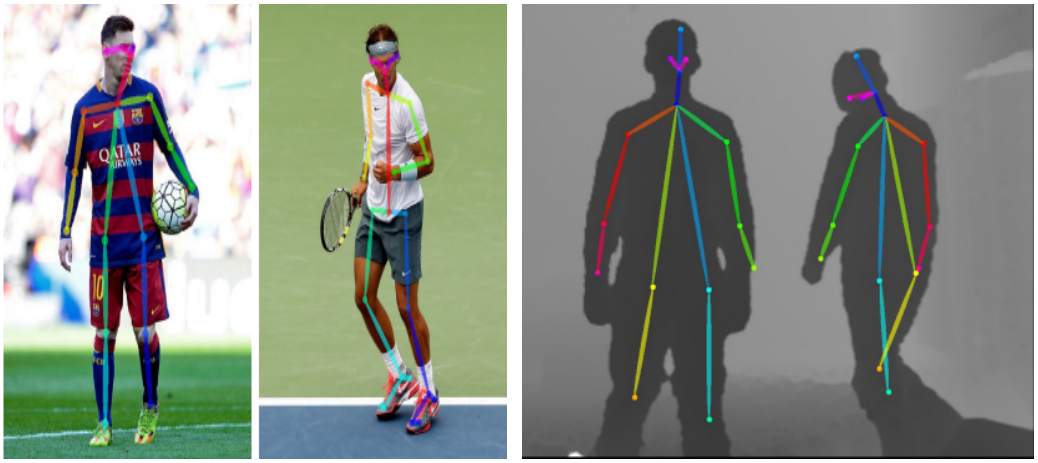


Figure 2. Skeleton encoding examples of human body. [14]

The skeleton data can be acquired using motion capture technologies. Motion capture systems provide very precise annotations of skeleton data. However, they require expensive sensing devices and it is necessary for actors to wear specialized sensors while collecting data. So, motion capture is not a convenient method for collecting data in many human action recognition applications [15].

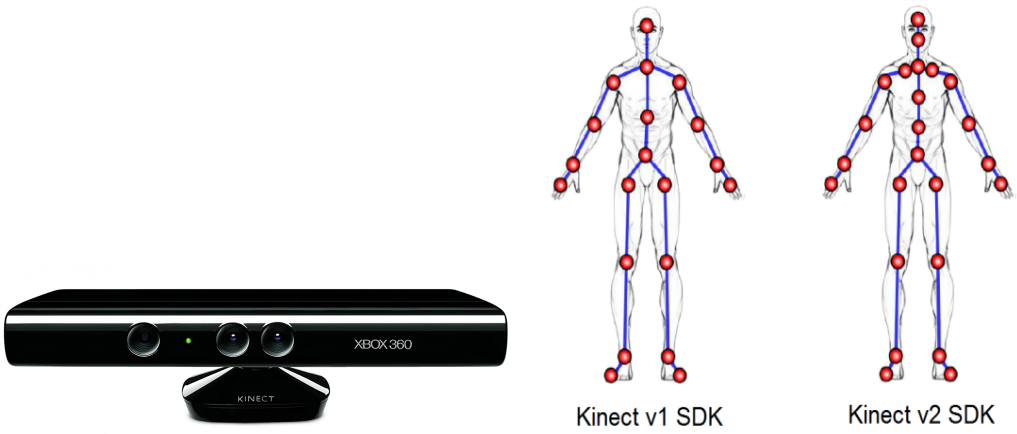
Skeleton data can also be collected by applying pose estimation on Red-Green-Blue (RGB) images or depth maps. Figure 3 shows some examples of skeleton graph generated by pose estimation. Pose estimation from RGB images [16–19] and depth maps [20, 21] are very intensive research topics over the last decade. Despite many difficulties in implementation, real-time state-of-the-art results have been achieved.



(a) (b)

Figure 3. Human pose estimation: (a) Estimation from RGB images [19]; (b) Estimation from depth maps. [21]

Microsoft Kinect is a new modern technology that can automatically extract human joints with real-time processing speed [22]. This device includes two different sensors: an RGB camera and a depth sensor. This combination allows the Kinect device to utilize the advantages of the two estimation methods, and it can produce very accurate estimations of human joints [23,24]. Therefore, current state-of-the-art large-scale skeleton datasets rely on Kinect devices in their data collecting process [11, 12]. Figure 4 presents the Kinect sensing device and the joints generated from Kinect software.



(a) (b)

Figure 4. Microsoft Kinect: (a) Kinect device; (b) Data generated from Kinect Software Development Kit (SDK). [24]

2.2 Action recognition using handcrafted features

The trajectories of skeleton joints can be captured using handcrafted features. These engineered features are intended to create a more distinctive representation of the human movements and spatio-temporal offset from skeleton sequential data. Then, based on the extracted features, classic machine learning classifiers such as Boost, Support Vector Machine (SVM), and probability maps are applied to predict the action.

Yang and Tian [25] introduced a feature based on position differences of joints, namely EigenJoints. The method extracts multiple features such as posture, motion, and offset in each frame, then applies normalization and Principal Component Analysis (PCA) to create the action description. An illustration of the method can be seen in Figure 5. EigenJoints can produce a compact and very discriminative skeleton frame representation. Based on the EigenJoints features, Naïve Bayes Nearest Neighbor (NBNN) classifier was used for multi-class action classification.

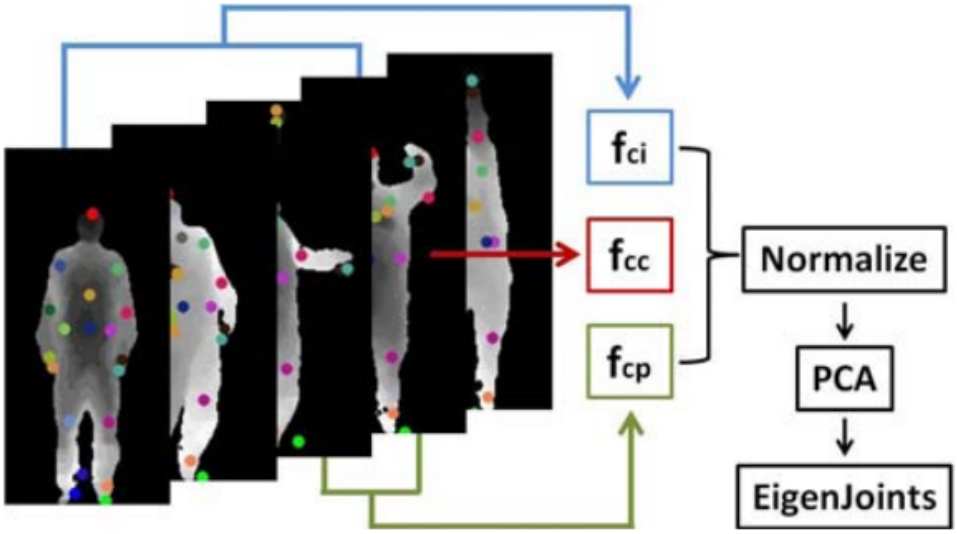


Figure 5. EigenJoints feature extraction. [25]

Gowayyed et al. [26] describe the trajectory of joint points by the Histogram of Oriented Displacement (HOD). The feature is the direction between two consecutive points in the temporal dimension. HOD features are extracted from three projections to form a 3D feature, and they describe how much the joints moved in each range of directions. After creating feature descriptors for every data sequence, a linear SVM algorithm is applied to predict the action label.

Vemulapalli et al. [27] proposed a skeletal representation that can be modeled as curves in the Lie Group SE(3)×...×Se(3). An illustration of the Lie group representation of skeletons can be found in Figure 6. Action curved manifolds from the Lie group are then mapped to their Lie algebra vector space for easier implementation. The authors used a combination of dynamic time warping, Fourier temporal pyramid representation, and linear SVM to perform action classification.

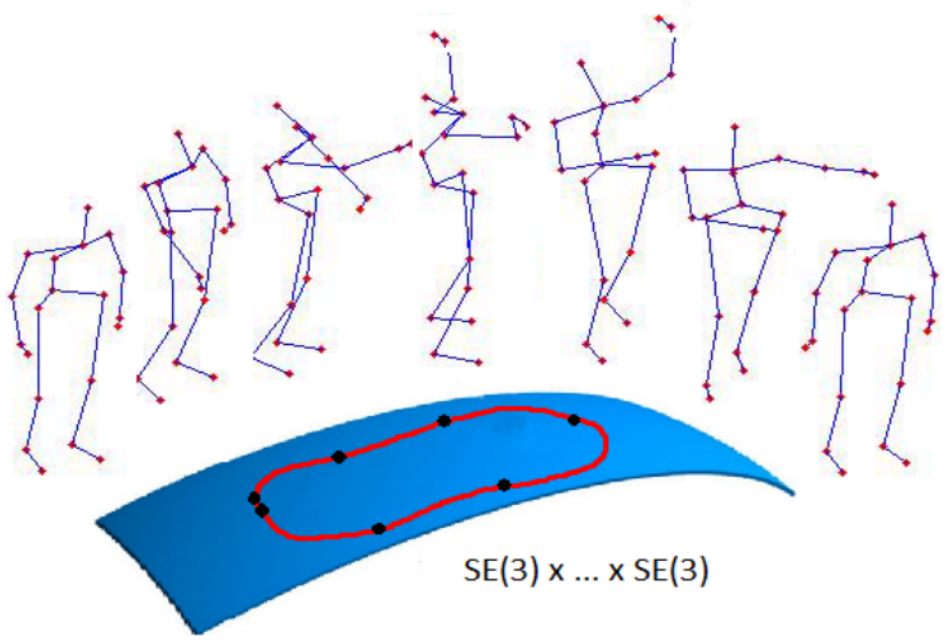


Figure 6. Skeleton representation in the Lie group. [27]

Koniusz et al. [28] used tensor representations to compactly capture higher-order relationships between skeleton joints for action recognition. They introduced two different Radial Basis Function (RBF) kernels, a Sequence Compatibility Kernel (SCK) and a Dynamics Compatibility Kernel (DCK). SCK captures the spatio-temporal compatibility of joints, while DCK models the action dynamics of a sequence. For action recognition, an SVM classifier was trained on linear tensors of the kernels.

From the above discussion, studies that use handcrafted features have proven the ability to obtain strong relationships among skeleton joints, and high-quality results have been achieved. However, Wang et al. [5] have demonstrated that handcrafted features could not generalize well for a wide range of datasets. This problem limits the use of human action recognition algorithms in practice.

2.3 Action recognition using deep learning

In the last few decades, deep learning, a subfield of machine learning, has achieved outstanding results in many computer vision applications [29, 30]. Deep learning is used for extracting high-level features from input data, and therefore, the need for feature engineering can be reduced. Despite the achievement, deep learning methods require a large amount of labeled data to perform well. In recent years, the growth of data storing capability and computing power have motivated the development of deep learning. Nowadays, most state-of-the-art methods in artificial intelligence are based on deep learning.

As a sub-field of computer vision, skeleton-based action recognition also benefits from the advancement of deep learning. Research have been done over the last few years to tackle this problem, and good results have been recorded. Skeleton-based human action recognition using deep learning methods can be divided into three groups depending on their core building block: Recurrent Neural Networks (RNN), Convolutional Neural Network (CNN), and Graph Neural Networks (GNN).

2.3.1 RNN-based methods

The RNN is designed to extract features from sequential data. Since human action analysis often requires sequential modeling in both space and time, current datasets mostly provide a collection of skeleton sequences. Therefore, RNN is a very reasonable choice for the feature extracting process of skeleton-based action recognition methods. Moreover, because RNN usually suffers from gradient problems and long-term modeling, recent studies often utilize some other related modules such as Long-Short Term Memory (LSTM) and Gated Linear Uint (GLU) to overcome those shortages.

Du et al [31] proposed an end-to-end hierarchical RNN model for skeleton-based action recognition. Figure 7 illustrates the pipeline of the method. The skeleton graph at each time step is separated into five parts based on human physical structure. The parts are then fed into multiple layers of bidirectional RNN and LSTM to get a spatial representation. The action is classified by applying a softmax classifier to the accumulated representation of the whole sequence.

Shahroudy et al. [11] proposed Part-aware LSTM to interpret the human action. Instead of using the whole body representation in each LSTM cell, they separate it into five-part groups. In this setup, the model learns the dependencies of each part separately and then

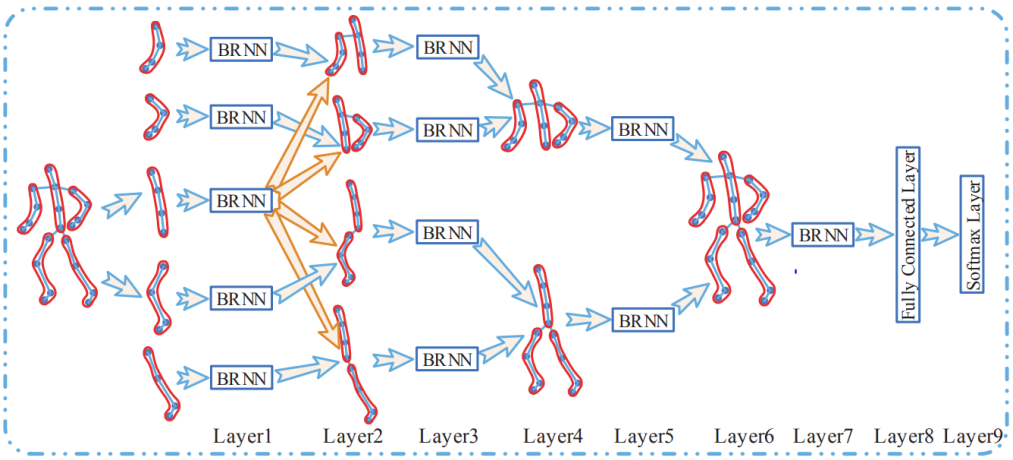


Figure 7. Hierarchical RNN for action recognition. [31]

combines them into a global representation. Action descriptions of the whole sequence are generated by combining the representation at each time step and classified by an SVM.

Zhu et al. [32] approached the skeleton-based action recognition problem by proposing an end-to-end fully connected deep LSTM. They introduced a novel regularization scheme to learn the co-occurrence features from skeleton joints at each time slot. Figure 8 shows the architecture of the fully connected LSTM. Due to the model's huge number of parameters, they implemented a co-occurrence exploration process to ensure effective learning.

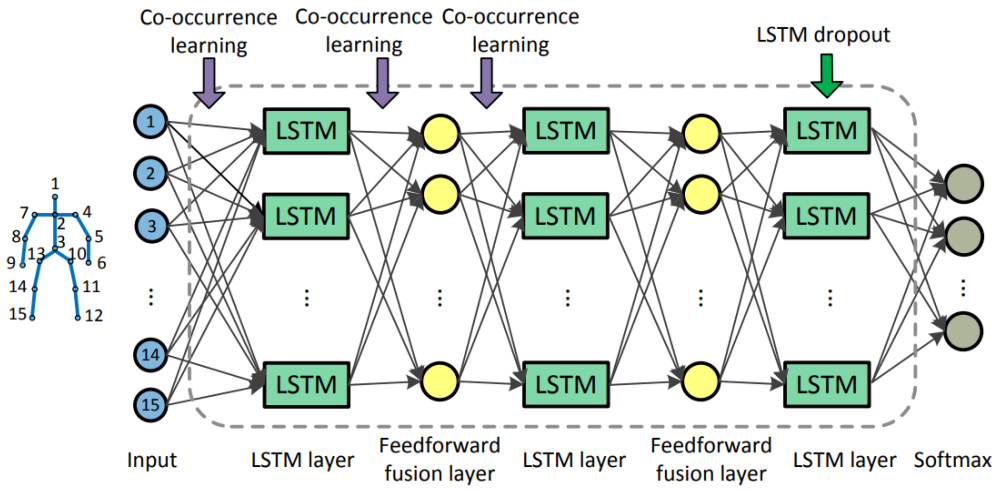


Figure 8. Fully connected LSTM for action recognition. [32]

Wang and Wang [33] introduced the two-stream RNN to model the spatial and temporal representation of skeleton data separately (Figure 9). The model extracts temporal features by applying a stack RNN, while hierarchical RNN is used for spatial dependen-

cies. Each stream of RNN has a Softmax classifier, and the action class is determined by combining the output of two classifiers.

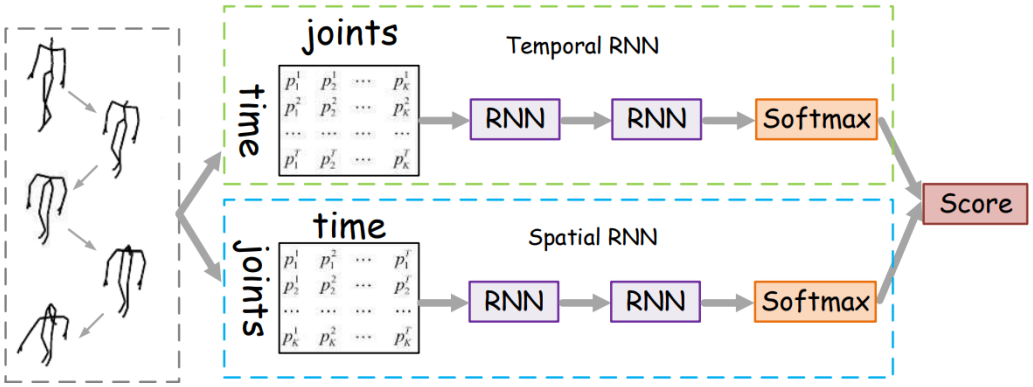


Figure 9. Two-stream RNN model for action recognition. [33]

Liu et al. [34] proposed a tree-structure-based traversal framework, which included spatio-temporal LSTM modeling and a trust gating mechanism, to approach the skeleton-based action recognition. First, they transform the skeleton graphs into chain models and ignore the kinetic interdependence among the joints. Next, spatio-temporal LSTM cells are used to extract high-level features from temporal dependencies and spatial dependencies at the same time. Finally, a softmax classifier is applied to predict the action class.

Although RNN-based methods provide good results, their ability to model spatial information is still weak, compared to other methods such as CNN. Moreover, the sequential processing characteristic of RNN limits them from effectively utilizing the parallel computing architecture of modern GPU, and it makes them slower than CNN models. Therefore, recent researches on skeleton-based action recognition increasingly apply CNN in their solutions.

2.3.2 CNN-based methods

The convolutional neural network is currently the most popular model in deep learning research due to its powerful ability to extract high-level features from data, especially in computer vision. CNN is also applied to skeleton-based human action recognition. However, the non-Euclidean and sequential characteristics of skeleton data are considered a challenge for CNN-based methods.

Li et al. [35] proposed a CNN-based framework for action recognition. The method feeds skeleton coordinates and motion into the CNN model. In this method, skeleton sequences are treated as three-dimensional tensors image data, therefore it shifts the task back to the classical image classification framework [30]. Two-stream CNN is also applied to introduce local temporal dependency into the model. Figure 10 illustrates the architecture of the method. Furthermore, since the ordering of human joints in the input feature maps can directly affect the classification results, a novel skeleton transformer was introduced. The transformer is a simple linear operator that maps the input to a fixed-size tensor. Therefore, the order and location of joints are rearranged in an optimal manner.

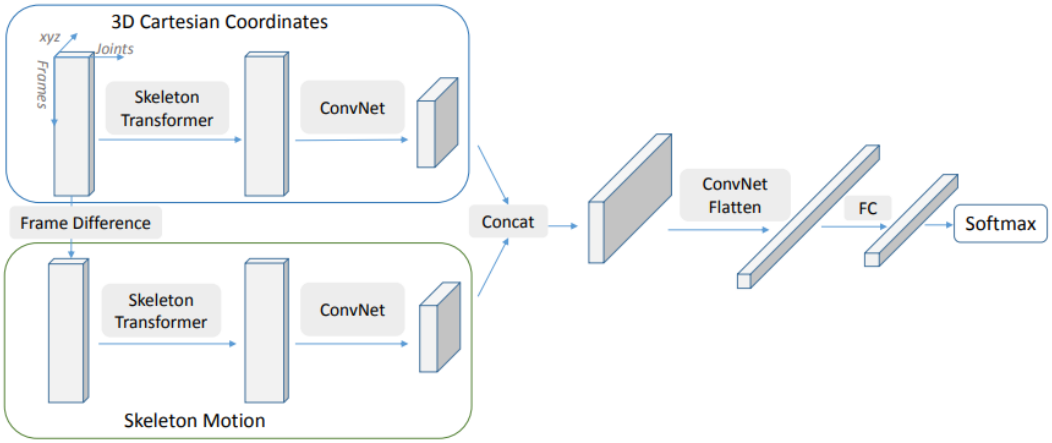


Figure 10. Two-stream CNN model for action recognition. [35]

Kim and Reiter [36] used Temporal Convolutional Networks (TCN) for human action recognition. Their model, Res-TCN, provides a way to learn explicitly the spatio-temporal representation of skeleton data. The input of the model is temporally concatenated frame-wise skeleton features across the entire sequence. Skip connection and 1D convolution filters are applied to learn the spatial and temporal dependencies from the input data. Based on the extracted high-level features, they implemented a global average pooling layer, a fully connected layer, and a softmax classifier to predict the action.

Ke et al. [37] introduced a novel skeletal representation referred to as clips. The method first transforms each skeleton sequence from Cartesian coordinates to cylindrical coordinates. Each generated clip is corresponding to a channel in the cylindrical coordinates. Each frame of the clips encodes the temporal information of the whole sequence. The method utilized the pre-trained VGG-19 CNN model [38] to extract the long-term spatio-temporal features from the clips. Moreover, based on four reference joints (left shoulder, right shoulder, left hip, and right hip), the spatial relationships are incorporated using Multi-Task Learning Network (MTLN).

Many CNN-based action recognition methods emphasized modeling the relationship between human joints in both space and time. However, they are usually assigned using domain knowledge. For example, the input feature maps are constructed by concatenating the human joints in specific orders, and changing the positions affects the outcome. Therefore, the task of developing a method that can implicitly learn the relationship between joints is necessary.

2.3.3 GNN-based methods

Different from RNN-based and CNN-based approaches, GNN provides the ability to model the dependencies of human joints implicitly. By eliminating the manual part assignment, GNN-based models are simpler and could learn the action representation better.

Yan et al. [39] first introduced the concept of GNN into skeleton-based action recognition. The authors proposed Spatial-Temporal Graph Convolutional Networks (ST-GCN) to model a sequence of skeleton graphs. The skeleton sequence includes two types of edges: spatial edges that express connectivity between human joints, and temporal edges that connect the same joints across time steps. As can be seen in Figure 11, the model uses multiple layers of spatial-temporal graph convolution to integrate information along spatial and temporal dimensions simultaneously. For action classification, a global pooling and a Softmax function are applied to the resulting tensor to predict the action.

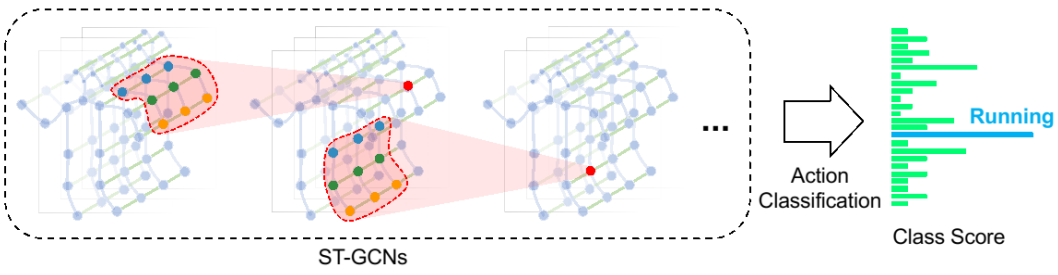


Figure 11. Spatio-temporal Graph Convolutional Network (GCN) for action recognition. [39]

Li et al. [40] proposed Actional-Structural Graph Convolutional Networks (AS-GCN) to capture richer dependencies in the spatial dimension of skeleton data. The method presents a module for capturing action-specific latent dependency between every human joint, namely A-link inference module [41]. The skeleton topology is also extended to represent the higher-order dependencies, referred to as structural links. The two modules

are then combined to produce a high-level spatial feature map. Along with a temporal convolutional neural network, AS-GCN can effectively learn both spatial and temporal features for action recognition. Figure 12 presents the pipeline of AS-GCN, and as can be seen in the figure, a decoding head is added in parallel to the recognition head to help with detailed action patterns capturing in self-supervision.

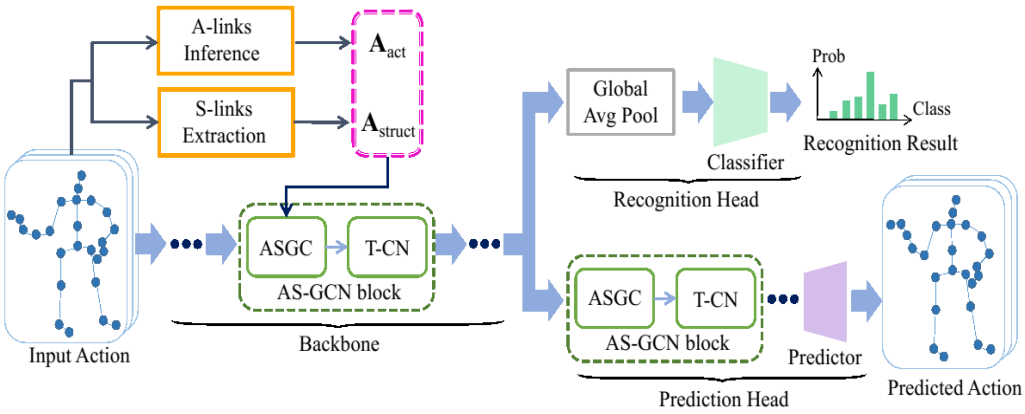


Figure 12. Actional-structural GCN for human action recognition. [40]

Shi et al. [42] reasoned that using predetermined and fixed skeleton graph topology for aggregating information is not optimal for diverse samples. Therefore, they proposed Two-stream Adaptive Graph Convolutional Networks (2s-AGCN) that captures second-order information (bones' lengths and directions) in addition to joints' dependencies on skeleton data. The configuration of the method is shown in Figure 13. Compared to first-order information, second-order relationships are naturally more informative and discriminative for action recognition. Furthermore, the human graph topology is learned adaptively during the training process to increase the flexibility of the model in representing skeleton sequences.

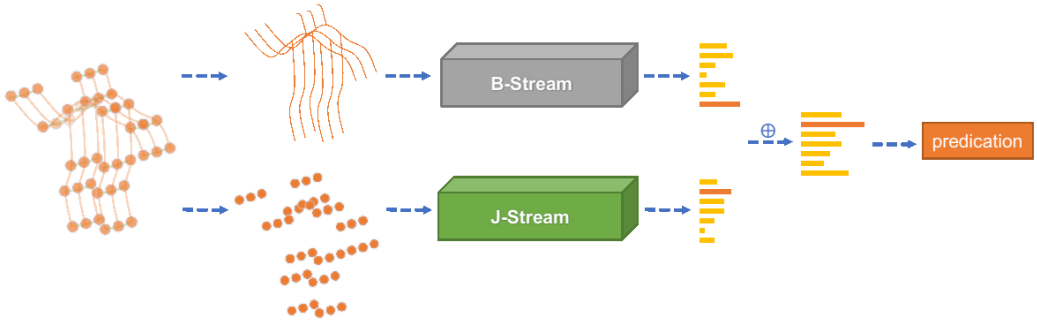


Figure 13. Two-stream adaptive GCN for human action recognition. [42]

Liu et al. [43] proposed a disentangled and unifying GCN framework for action recognition. The disentangling task removes the redundant dependencies between node features from different neighborhoods when aggregating spatial information. In the aggregation process of each human joint, the graph topology is modified to directly obtain information from farther nodes. The unifying task refers to the facilitation of direct information flow across space and time, as can be seen in Figure 14. The combination of the two proposed methods creates a powerful feature extractor with multi-scale receptive fields across both spatial and temporal dimensions, namely Multi-Scale 3-dimensional Graph Neural Networks (MS-G3D).

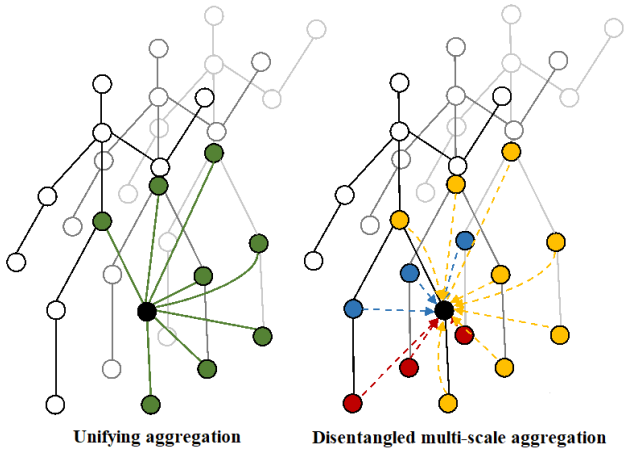


Figure 14. Unifying and disentangled spatial-temporal GCN for action recognition (different colors represent different distance from the node). [43]

Ye et al. [44] introduced a novel CNN named Context-encoding Network (CeN) for learning skeleton topology automatically. CeN modules are lightweight and can be embedded into a graph neural network to create the Dynamic-GCN. By using CeN, when learning the dependency between two joints, contextual features from the rest joints are incorporated in a global manner. Dynamic-GCN achieved better performance with fewer Floating point Operations Per Second (FLOPS) than existing methods.

Chen et al. [45] proposed a Channel-wise Topology Refinement Graph Convolutional Networks (CTR-GCN) for action recognition. The proposed method can dynamically refine a shared prior topology for each channel. Similar to [46], CTR-GCN creates multi-channel attention maps to model the correlation between human joints in each skeleton graph. The generated attention maps are then used as refinements for the actual human action topology. Figure 15 illustrates the pipeline of the model. This approach provides an effective aggregation of joint features in different channels, leading to stronger representation.

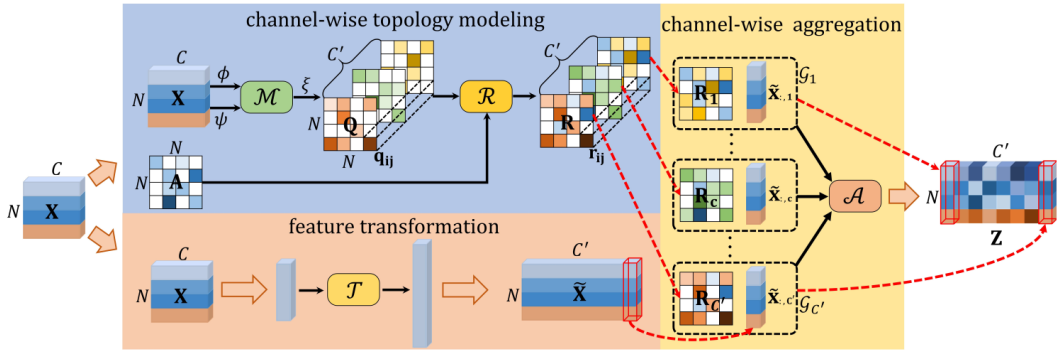


Figure 15. Channel-wise topology refinement GCN for action recognition. [45]

Due to the non-Euclidean characteristic of skeleton data, GNN-based methods have outperformed CNN and RNN in many recent studies on human action recognition. Most research follows the framework introduced by Yan et al. [39], and they focus on improving the spatio-temporal modeling of skeleton data.

In this thesis, an attention-based GCN feature extractor that can model long-range dependencies along spatial and temporal dimensions for skeleton-based human action recognition, namely Spatio-Temporal Attention Graph Convolutional Networks (STA-GCN) is proposed. The model inherits the advantages and eliminates some bottlenecks of the previous works. Detailed information on the model is presented in Chapter 4.

3 DEEP LEARNING FOR GRAPHS

3.1 Deep neural networks

Deep neural network is the quintessential model in deep learning. The main task of a deep neural network is to approximate a function f that maps an input $\mathbf{x}$ to a label $\mathbf{y}$. The mapping is defined as follows:

$$\hat{\mathbf{y}} = f(\mathbf{x}, \theta) \tag{1}$$

where θ is the learned parameters that produce the optimal approximation.

In the simplest form, neural networks are typically represented by stacking multiple different functions together. For example, in Figure 16, an example of a four-layer neural network model is shown. The model includes an input layer $f^{(1)}$, two hidden layers $f^{(2)}$, $f^{(3)}$, and an output layer $f^{(4)}$. The layers are connected in a chain to form the expression as follows:

$$f(\mathbf{x}) = f^{(4)}\big(f^{(3)}\big(f^{(2)}\big(f^{(1)}(\mathbf{x})\big)\big)\big) \tag{2}$$

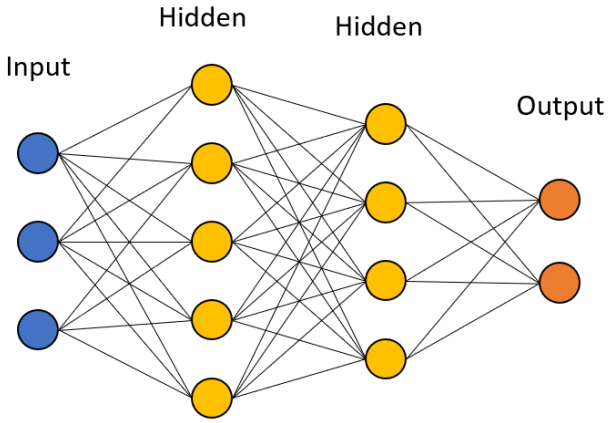


Figure 16. An example of the four-layer neural network model.

Deep learning is a wide research area, and many types of neural networks have been proposed. The models that have achieved significant results recently are Multi-Layer Perceptron (MLP) [47], recurrent neural network (RNN), and convolutional neural network (CNN). The last two methods, however, are defined in structured data such as sequences or digital images. In this thesis, to consider the human natural dynamic in calculations,

the data is constructed in unstructured, non-Euclidean form, referred to as graphs. Therefore, MLP is used as the main learning method skeleton-based action recognition. Each layer of an L-layer MLP indexed by $l \in \{1, ..., L\}$ is defined as follows [48]:

$$f^l(\mathbf{h}) = \sigma(\mathbf{W}^l\mathbf{h} + \mathbf{b}^l) \tag{3}$$

where $\mathbf{W}^l$ is the weight matrix of *l-th* layer, $\mathbf{h}$ is the feature vector of a node, $\mathbf{b}^l$ is the bias vector, and σ is the non-linearity activation function. Based on the recommendation from Goodfellow et al. [49], throughout this study, the Rectified Linear Unit (ReLU) is used as the activation function defined as

$$\sigma(\mathbf{h}) = \mathrm{ReLU}(\mathbf{h}) = \max(0, \mathbf{h}) \tag{4}$$

When a deep neural network is used to produce the output $\hat{\mathbf{y}}$ from input $\mathbf{x}$, the information flows forward in the network. This process is called forward propagation. After that, a scalar cost $\mathbf{J}(\theta)$ is calculated from the produced $\hat{\mathbf{y}}$ and the label $\mathbf{y}$. By applying back propagation, the cost information can flow backwards to the network and the gradient can be calculated. To train the neural networks, stochastic gradient descent (SGD) is used to minimize the cost function $\mathbf{J}$.

3.2 Preliminaries on graphs

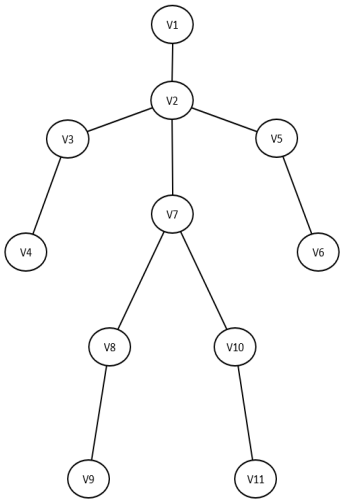
A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is an ordered pair constructed by a set of N vertices (also referred to as nodes) $\mathcal{V} = \{v_1, v_2, ..., v_N\}$ and a set of edges $\mathcal{E}$ between these vertices. An edge going from node $u \in \mathcal{V}$ to node $v \in \mathcal{V}$ is noted as $(u, v) \in \mathcal{E}$, and edges can be directed or undirected. In this thesis, only simple graphs are concerned, where each pair of vertices contains one edge, and all edges are undirected. A human skeleton can be represented by a graph as illustrated in Figure 17a. Each node represents a joint of the human graph, and each edge represents the connection between joints depending on the human kinetic.

A graph can be conveniently formulated by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$. If the graph is undirected, $\mathbf{A}$ is a symmetric matrix. Moreover, when the graph contains directed edges, $\mathbf{A}$ is not necessarily symmetric. In an adjacency matrix, vertices are ordered to index a particular row and column. The connectivity between joints is presented by the value in

each matrix's entries, such that

$$a_{uv} = \begin{cases} 1, & \text{if}(u, v) \in \mathcal{E} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

In practice, each entry can contain binary values or real values between $\{0, 1\}$ to represent the weight of each connection. A simple example of an adjacency matrix for the human skeleton is shown in Figure 17b.



(a)

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11
V1	0	1	0	0	0	0	0	0	0	0	0
V2	1	0	1	0	1	0	1	0	0	0	0
V3	0	1	0	1	0	0	0	0	0	0	0
V4	0	0	1	0	0	0	0	0	0	0	0
V5	0	1	0	0	0	1	0	0	0	0	0
V6	0	0	0	0	1	0	0	0	0	0	0
V7	0	1	0	0	0	0	0	1	0	1	0
V8	0	0	0	0	0	0	1	0	1	0	0
V9	0	0	0	0	0	0	0	1	0	0	0
V10	0	0	0	0	0	0	1	0	0	0	1
V11	0	0	0	0	0	0	0	0	0	1	0

(b)

Figure 17. Human skeleton graph: (a) Geometric graph; (b) Adjacency matrix.

One of the key challenges when applying deep learning to graphs is the variable-sized input. Unlike the fixed dimension of input when working with images, the number of vertices and edges of a graph input may vary within an application, and the deep learning model still needs to perform correctly under those changes.

Another challenge of graph deep learning is the invariance to node permutation. When dealing with graph data, it is desired for an approach that does not depend on the arbitrary ordering of nodes specified in the adjacency matrix $\mathbf{A}$. This is called permutation invariance, and it implies that for any two isomorphism graphs, the outcomes of the deep learning model need to be the same [50].

3.3 Graph neural networks

Graph neural networks (GNNs) are a class of Neural Networks (NN) models suitable for processing graph-structured data. Current well-studied machine learning on graphs includes: node classification; relation prediction; graph classification, regression, and clustering [51]. The method proposed in this thesis falls into the latter category, graph classification. Similar to traditional supervised learning methods, each skeleton graph is represented as a data point with an associated action label, and the goal is to learn the mapping from data points to labels.

The main challenge of learning on graphs is the difficulty in generating graph representations while utilizing structured information. Standard deep learning toolboxes are optimized for either grid-like data (CNN) or sequences (RNN), so when applying them to graphs, prior knowledge and assumption about the data structure are required. To overcome the limitation, the GNN was introduced to define the learning task on graph-structure data.

The feature extracting structure of the GNN is identical to any NN model [52], as can be seen in Figure 18. However, each hidden layer in the GNN acts as an update step rather than a linear transformation. These update layers are where information propagates throughout the graph. Similar to the NN models, a k number of hidden layers are used, and at the k-layer, each node now contains the information from the nodes that are k-step away.

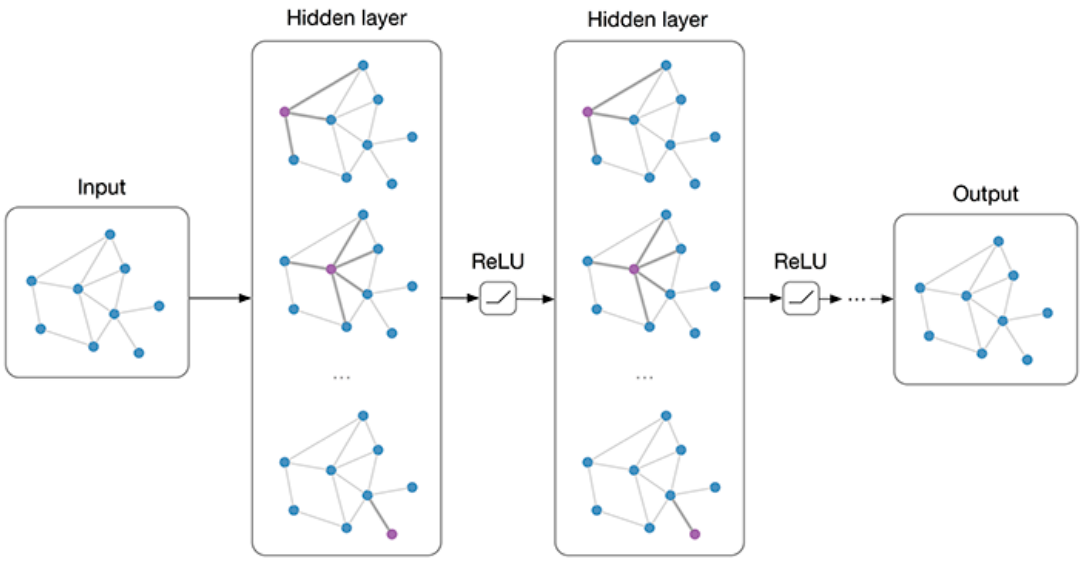


Figure 18. The general idea of GNN. [52]

In many cases, attributes or feature information associated with a graph are provided. Specifically, each joint in the human graph contains a feature vector that includes 3-D locations, 2-D locations, confident scores, and joint orientations. The main goal of the GNN is to process these node features, along with the structure information, to create a distinctive graph representation of the human graph for further processing.

In the perspective of each graph node, the basic GNN framework is analogous to a standard MLP. The only difference is the additional aggregation process before each layer of the network. This idea is illustrated in Figure 19.

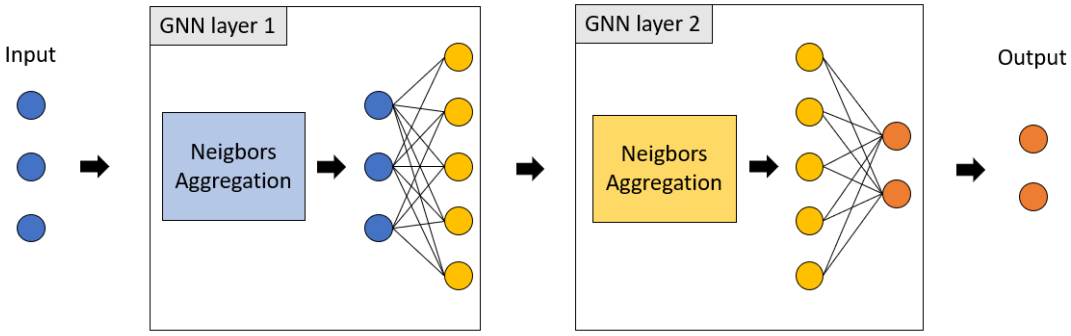


Figure 19. Node representation update in the GNN.

3.3.1 Neural message passing

The basic graph neural network GNN model has multiple variations, but they all utilize a form of neural message passing framework described by Gilmer et al. [53] in their approach. The goal of the method is to encode graph information in the form of node embeddings by iteratively combining neighboring node features.

The message passing technique takes an input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a set of node features $\mathbf{x}_u \in \mathbb{R}^{d \times |\mathcal{V}|}$, to generate node embeddings $\mathbf{h_u}, \forall u \in \mathcal{V}$. In each message-passing layer, a hidden embedding $\mathbf{h}_u$ of node u is updated based on the information sent from a set of neighbors $\mathcal{N}(u)$ as illustrated in Figure 20.

The message passing process can be generally expressed as follows:

$$\begin{aligned} \mathbf{m}_{\mathcal{N}(u)}^{(k)} &= \text{AGGREGATE}^{(k)}(\mathbf{h}_v^{(k)}, \forall v \in \mathcal{N}(u)) \\ \mathbf{h}_u^{(k+1)} &= \text{UPDATE}^{(k)}\left(\mathbf{h}_u^{(k)}, \mathbf{m}_{\mathcal{N}(u)}^{(k)}\right) \end{aligned} \tag{6}$$

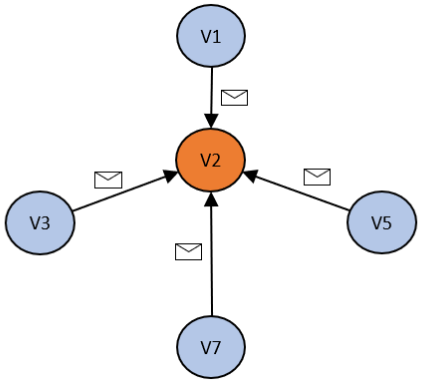


Figure 20. Neural message passing.

where $\mathbf{m}_u^{(l)}$ is the encoded message from neighborhood $\mathcal{N}(u)$ at the update step k. The AGGREGATE functions process all the incoming messages from the node's neighbors by either summing them, taking the average, or concatenating them together. Then, the UPDATE function transforms the aggregated information into a new node feature representation.

The intuition behind the GNN is straightforward: at each layer, every node aggregates information from its local neighborhood, and after k layers, every node embedding contains information about its k-hop neighborhood. The encoded information comes in two forms: structural information, which represents the degree of all the nodes in the k-hop neighborhood; and feature-based information. Similar to the CNNs, the GNN also has the local feature-aggregation behavior. However, instead of operating on spatial-defined regions in an image, the GNN aggregates information from local neighborhoods.

Gilmer et al. [53] proposed the information aggregating function as follows:

$$\begin{aligned} \mathbf{m}_{\mathcal{N}(u)}^{(k)} &= \sum_{v \in \mathcal{N}(u)} {f_e}^{(k)}(\mathbf{h}_v^{(k)}, \mathbf{h}_u^{(k)}, \mathbf{h}_{(u,v)}^{(0)}) \\ \mathbf{h}_u^{(k+1)} &= {f_v}^{(k)}\left(\mathbf{h}_u^{(k)}, \mathbf{m}_{\mathcal{N}(u)}^{(k)}\right) \end{aligned} \tag{7}$$

where each message from node u to a neighbor node v is encoded based on their node features $\mathbf{h}_u^{(k)}, \mathbf{h}_v^{(k)}$, and their edge feature $\mathbf{h}_{(u,v)}^{(0)}$. The message function ${f_e}^{(k)}$ and update function ${f_v}^{(k)}$ are learnable functions (e.g. small MLPs) that have shared weights across the graph.

The proposed method from Gilmer et al. [53] is called Message Passing Neural Networks (MPNN), and is currently the most powerful GNN. However, the model also suffers from

many disadvantages because of its complex structure. First, the MPNN requires the intervention of edge features, which cause memory and representational issues. Second, the model could be over-fitted if the dataset is sparse, similar to a traditional MLP. Therefore, this type of GNN is only applied to small graphs, and to tasks that requires the representation of edge features.

3.3.2 The basic graph neural network

Scarselli et al. [54] proposed a simple type of GNN, which can be mathematically presented as follows:

$$\mathbf{h}_u^{(k)} = \sigma\big(\mathbf{W}_{\text{self}}^{(k)}\mathbf{h}_u^{(k-1)} + \mathbf{W}_{\text{neighbors}}^{(k)} \sum_{v\in\mathcal{N}(u)} \mathbf{h}_v^{(k-1)} + \mathbf{b}^{(k)}\big) \tag{8}$$

where $\mathbf{W}_{\text{self}}^{(k)}$, $\mathbf{W}_{\text{neigh}}^{(k)}$ are trainable weight matrices, and σ is a non-linearity function ReLU.

This approach differs from the MPNN in the aggregation process. Instead of calculating a message for each pair of neighboring nodes, the model aggregates the feature of neighbor nodes directly. Then a linear transformation is applied to the aggregated nodes to enable the learning capability. In addition to the information of the neighbors, the feature of node u itself is also taken into account, similar to the CNN.

When considering the graph-level processing, Eq. 8 can be written in matrix form as follows:

$$\mathbf{H}^{(k)} = \sigma\big(\mathbf{A}\mathbf{H}^{(k-1)}\mathbf{W}_{\text{self}}^{(k)} + \mathbf{H}^{(k-1)}\mathbf{W}_{\text{neighbors}}^{(k)} + \mathbf{b}^{(k)}\big) \tag{9}$$

where $\mathbf{H}^{(l)} \in \mathbb{R}^{|V|\times d}$ the matrix representation of the graph at layer l, $\mathbf{A}$ is the adjacency matrix that encode the relationship of the nodes. In practice, neural message passing models are usually simplified by using self-loops mechanism. Therefore neural message passing process is reduced to

$$\mathbf{h}_u^{(k)} = \text{AGGREGATE}(\{\mathbf{h}_v^{(k-1)}, \forall v \in \mathcal{N}(u) \cup \{u\}\}) \tag{10}$$

When using self-loops, the separated update function is eliminated by implicitly integrating it into the aggregation method. This combination of the two processes helps alleviate overfitting problem, but can decrease the representation ability of the GNN [51]. The

graph-level update in matrix form is shown as follows:

$$\mathbf{H}^{(k)} = \sigma\big((\mathbf{A} + \mathbf{I})\mathbf{H}^{(k-1)}\mathbf{W}^{(k)} + \mathbf{b}^{(k)}\big) \tag{11}$$

with $\mathbf{I}$ is the identity matrix, $\mathbf{W}^{(k)}_{\text{self}}$, $\mathbf{W}^{(k)}_{\text{neigh}}$ are now combined into one weight matrix $\mathbf{W}^{(k)}$ at layer k. The invention of Eq. 11 lays a foundation for many other GNN future methods because of its simplicity and scalability.

3.3.3 Graph convolutional networks

The message passing of the traditional GNN takes the form

$$\mathbf{m}^{(k)}_{\mathcal{N}(u)} = \sum_{v \in \mathcal{N}(u)} \mathbf{h}^{(k-1)}_{v} \tag{12}$$

where the message is simply the summation of neighbor nodes. One major problem with this approach is that it can be very unstable and sensitive to node degrees. For instance, when a node u has a much higher number of degrees than a node u' within a graph, the aggregated feature from u is larger than from u'. This problem can cause numerical instabilities during the training process and make the optimization become more difficult. A solution for this problem is to normalize the aggregated message based on the node degrees, illustrated as

$$\mathbf{m}^{(k)}_{\mathcal{N}(u)} = \frac{\sum_{v \in \mathcal{N}(u)} \mathbf{h}^{(k-1)}_{v}}{|\mathcal{N}(u)|} \tag{13}$$

Kipf and Welling proposed the idea of symmetric normalization into the CNN [55], namely GCN, which was motivated by convolution operation in spectral domain. The message passing is defined as follows:

$$\mathbf{m}^{(k)}_{\mathcal{N}(u)} = \sum_{v \in \mathcal{N}(u)} \frac{\mathbf{h}^{(k-1)}_{v}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(u)|}} \tag{14}$$

where the aggregated information is normalized symmetrically in two dimensions. The update message passing function of GCN is now presented as follows:

$$\mathbf{h}^{(k)}_{u} = \sigma\big(\mathbf{W}^{(k)} \sum_{v \in \mathcal{N}(u) \cup \{u\}} \frac{\mathbf{h}^{(k-1)}_{v}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(u)|}}\big) \tag{15}$$

The idea of the GCN is simple, easy to implement, and can overcome the problem of node degree imbalance. The GCN is currently the most influential model in the field of graph deep learning. However, the model also has some issues of its own. The normalization may cause loss of information, and eliminating the learning function within the message passing process reduces the representative power of the model.

3.3.4 Graph attention networks

To overcome the limitation of the GCN, while retaining the characteristic of the MPNN, a popular strategy is to apply the self-attention mechanism [56] into the message passing process.

Veličković et al. [57] proposed Graph Attention Network (GAT) by using attention weights to define the message passing. The algorithm can be visually explained in Figure 21 and mathematically expressed as follows:

$$\mathbf{m}_{\mathcal{N}(u)}^{(k)} = \sum_{v \in \mathcal{N}(u)} a_{u,v}^{(k)} \mathbf{h}_v^{(k-1)} \tag{16}$$

where $a_{u,v}$ denotes the important factor of neighbor $v \in \mathcal{N}(u)$ to the aggregating node u.

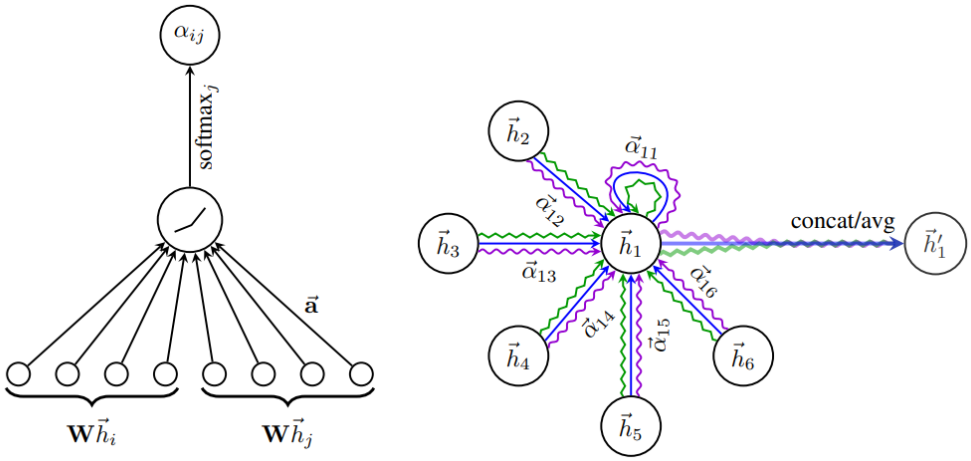


Figure 21. Graph attention message passing. [57]

The attention weights are calculated as

$$a_{u,v} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_\mathbf{u} \oplus \mathbf{W}\mathbf{h}_\mathbf{v}]\right)\right)}{\sum_{v' \in \mathcal{N}(u)} \exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_\mathbf{u} \oplus \mathbf{W}\mathbf{h}_{\mathbf{v}'}]\right)\right)} \tag{17}$$

where $\mathbf{a}$ is a learnable weight vector, $\oplus$ denotes the concatenation operation.

Once the attention weights are obtained, the update function of the GAT now becomes

$$\mathbf{h}_u^{(k)} = \sigma\big(\sum_{v \in \mathcal{N}(u) \cup \{u\}} a_{u,v}^{(k)} \mathbf{W}^{(k)} \mathbf{h}_v^{(k-1)}\big) \tag{18}$$

The GCNs define the coefficients $a_{u,v}$ explicitly, that depends entirely on the node degrees. The GATs overcome this shortage by computing $a_{u,v}$ implicitly, allowing for a boost in learning capability. Even though the GATs are not as general as MPNNs, but due to the simple calculation, they are totally more scalable.

In practice, multi-head attention [46] is usually used to stabilize the process of attention coefficients learning. Instead of producing one attention mask only, N masks (heads) can be created and combined together for better estimation. The multi-head attention calculation at a GAT layer k is defined as

$$\mathbf{h}_u^{(k)} = \bigodot_{n=1}^{N} \sigma\big(\sum_{v \in \mathcal{N}(u) \cup \{u\}} a_{u,v}^{(k,n)} \mathbf{W}^{(k)} \mathbf{h}_v^{(k-1)}\big) \tag{19}$$

In this thesis, GAT, GCN, and multi-head attention are utilized to model human skeleton graph sequence in both spatial and temporal dependencies.

4 SPATIO-TEMPORAL ATTENTION GRAPH CONVOLUTIONAL NETWORKS

In this chapter, details of the proposed model Spatio-Temporal Attention Graph Convolutional Networks (STA-GCN) are discussed. Figure 22 introduces the general pipeline of the model. The pipeline includes three main tasks: spatial modeling, temporal modeling, and action classification. The self-attention mechanism is the core building block of the model. Each part of the pipeline is discussed in the following subsections.

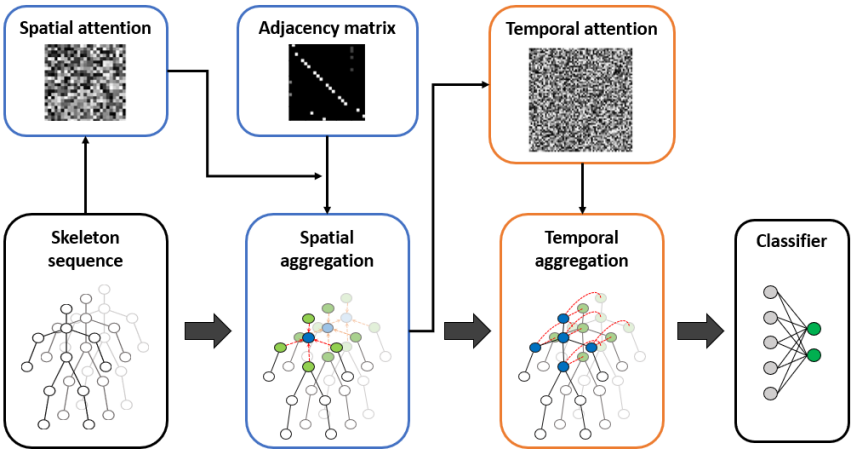


Figure 22. The pipeline of the proposed model (blue frames represent spatial modeling, orange frames represent temporal modeling).

4.1 Spatial modeling

4.1.1 Feature extraction from human kinetics

Given an input sample, the action is encoded by a sequence of skeleton graphs. Each graph represents the instance pose of the subject at a time step. In order to extract the spatial features for classification purpose, multiple layers of GNN is applied. The connection of human joints in any skeleton graph within the sequence is encoded in the adjacency matrix. By utilizing the prior knowledge of joint relationships, the GNN can aggregate the neighborhood information according to the Eq. 11. Figure 23 demonstrates the process of feature aggregating for each skeleton graph. After a GNN layer, each human joint contains the combined spatial features of all connected neighborhoods and itself. The resulting embeddings consist of both structure and feature information.

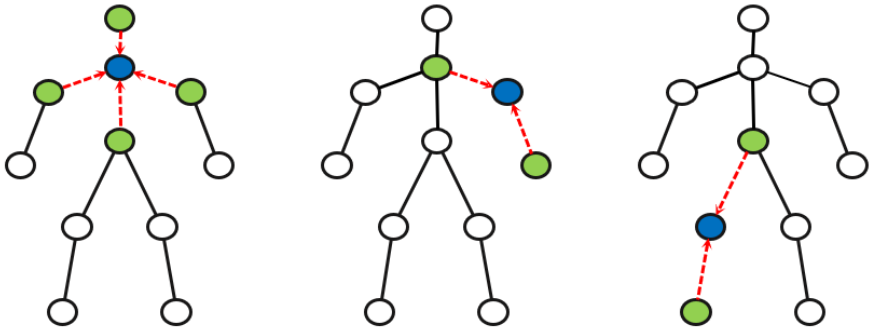


Figure 23. The aggregation processes of some example joints in a GNN layer.

The GNN provides an effective way to extract neighborhood features from the human skeleton graph. However, the simplicity in GNN's calculation raises one problem. The aggregation process may cause numerical instability when dealing with graphs that contain high-degree nodes. Thus, a normalization method is much needed to overcome the problem. The GCN provides a very popular symmetric normalization method for the GNN using Eq. 15. Therefore, the GCN is chosen for the spatial modeling.

There are two important aspects when doing calculations on graph data: structure information and feature information. Structure information refers to the architect or the relationships between nodes. Feature information is related to the actual values within each node's embedding vector. In this study, because the human kinetics is the same for every action sequence in each dataset, the structure information is not very useful for the classification of human actions. For this reason, structure information is considered less important than feature information. Therefore, the usage of the normalization method is reasonable, according to [51].

4.1.2 Extraction of additional joint features

Prior knowledge of human kinetics plays a major role in previous skeleton-based action recognition methods. The skeleton graphs in those methods are predefined and based only on the physical structure of the human body. This may limit the representative capacity of the model. Thus, in order to achieve outstanding performance, hidden information also needs to be considered. Figure 24 shows some examples that hidden connections can collect more discriminative features for classification.

Another disadvantage of relying only on prior human topology is related to the architecture of the GCN. Different layers of the GCN contain different types of semantic spatial

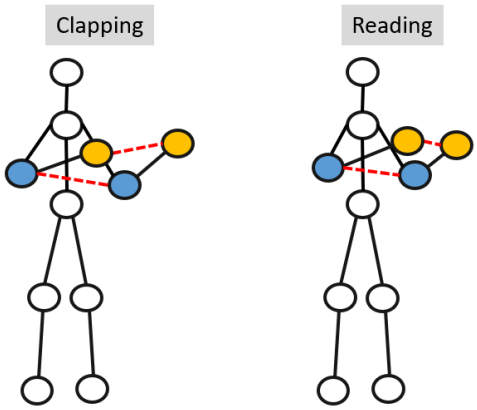


Figure 24. An example of the hidden connections. The relationship between two hands is more beneficial for classification of hard samples.

information. Using a fixed topology for the calculation cannot optimally aggregate useful features. Therefore, similar to [42], every entry of the adjacency matrix is parameterized and learned during training. This simple idea can naturally consider the hidden connections between every body joint, and at each GCN layer. Figure 25 illustrates the adaptively learned adjacency matrix.

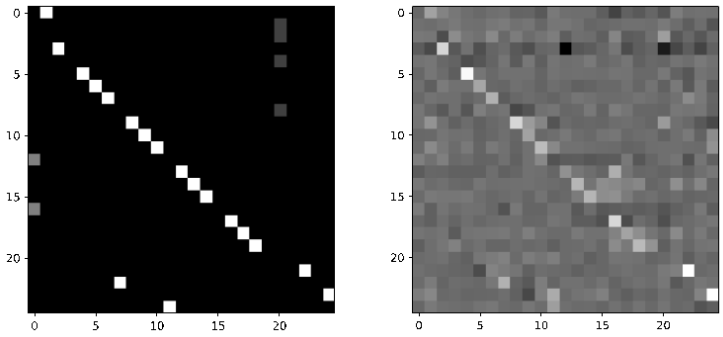


Figure 25. The original adjacency matrix (left) and the adaptively optimized matrix (right). [42]

Furthermore, human actions are mostly differentiated based on the movement of the hands, legs, and head. Thus, the feature information around the limbs' area is much more discriminative than others in action recognition. An illustration is shown in Figure 26, where limbs' information is much more useful to differentiate two actions. The calculation of the GCN is explicitly defined based on the prior knowledge of human kinetics and does not allow the ability to shift attention among the skeleton graph. This problem is one major drawback of previous methods that used only the GCN for spatial modeling. By adopting the proposal from [42] to parameterize the adjacency matrix, the strength of joint connections is learned and optimized during training.

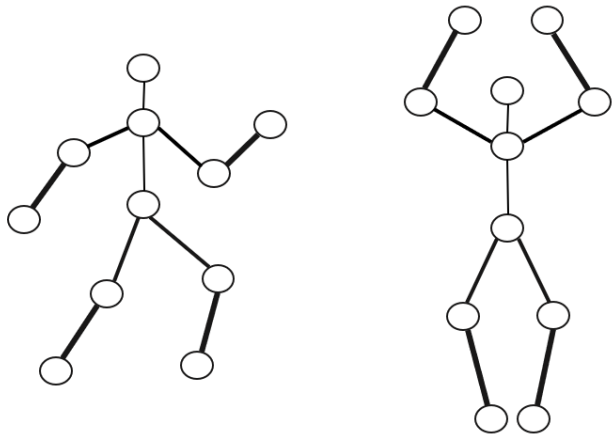


Figure 26. Some part of human kinetics is more informative than others for action recognition.

4.1.3 Spatial self-attention

Using GCN with adaptive mechanisms creates a good starting point for the spatial modeling problem. However, the adaptive adjacency matrix depends only on the data during training. This could limit the expressive power of the feature extracting process. In addition to the adaptive GCN, the calculation from the GAT [57] is also integrated. By using the GAT, each body joint can interact with every other joint to decide its importance. In Figure 27, an example of joint interaction is shown. By combining the two, the benefits of both methods can be utilized. The GCN is good for capturing spatial dependency between the nodes, given the prior knowledge about human kinetics in the adjacency matrix. The GAT is good for capturing hidden correlations between joints that are not connected according to prior human kinetics information.

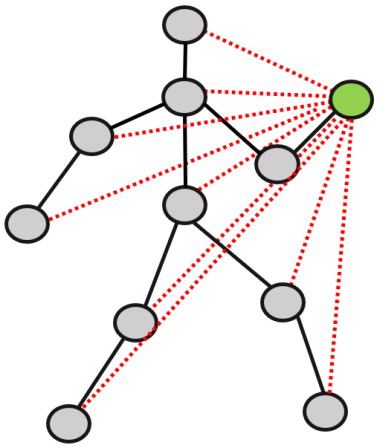


Figure 27. The hidden relationships between one joint and every other one.

By combining the GCN and the GAT, the calculation is constructed as follows:

$$\mathbf{h}_u^{(k)} = \sigma\Big(\sum_{v \in \mathcal{N}(u) \cup \{u\}} a_{u,v}^{(k)} \mathbf{W}^{(k)} \mathbf{h}_v^{(k-1)} + \mathbf{W}^{(k)} \sum_{v \in \mathcal{N}(u) \cup \{u\}} \frac{\mathbf{h}_v^{(k-1)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(u)|}} \Big) \tag{20}$$

Eq. 20 can be written in matrix form as follows:

$$\begin{aligned} \mathbf{C} &= \mathbf{M} + \tilde{\mathbf{D}}^{-1/2} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-1/2} \\ \mathbf{h}^k &= \sigma\big(\mathbf{C}\mathbf{x}^k \mathbf{W}\big) \end{aligned} \tag{21}$$

where $\mathbf{M}$ is the attention map calculated from GAT, $\tilde{\mathbf{A}}$ is the original adjacency matrix with self-loop, $\mathbf{D}$ is the degree matrix of skeleton graphs, $\mathbf{C}$ is the combined matrix.

Self-attention is utilized to model spatial dependencies. The algorithm consists of three main elements: query, key, and value. First, the input sequence is first globally pooled along the temporal dimension. The pooled matrix is used to derive the query and key for computing the attention score. Then, the attention map is combined with the adjacency matrix to produce the final tensor, according to Eq. 21. The resulting matrix now contains information based on both human kinetics and hidden interactions. The value element is created from the input using linear transformations. The value is then multiplied with the combined attention matrix to get the final embedding tensor of the skeleton sequence. Broadcasting subtraction is used for calculating the attention score, similar to [45]. Finally, a Softmax function is applied to normalize the coefficients across different nodes. The full pipeline is shown in Figure 28.

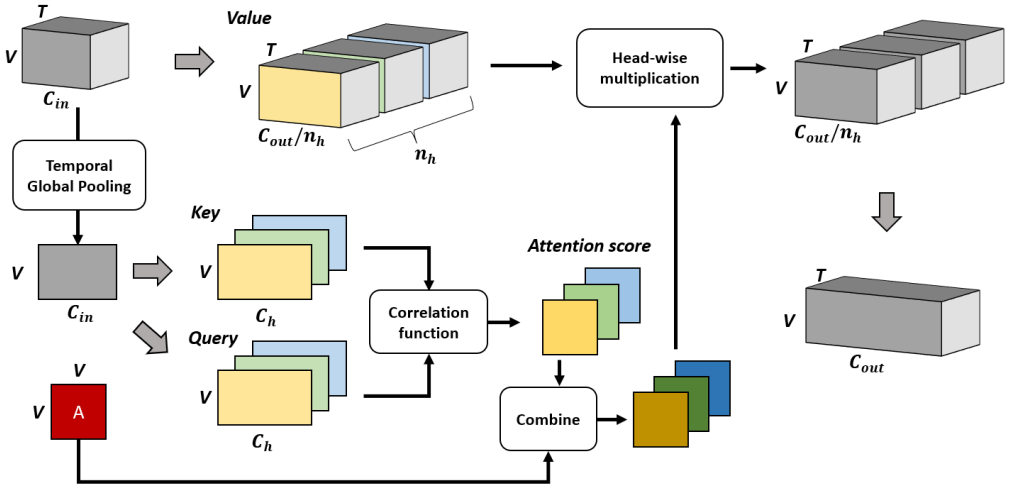


Figure 28. Attention modeling for spatial dependencies.

A multi-headed version of self-attention is integrated to stabilize the calculations. The input sequence is transformed into the output shape using linear transformations. The tensor is then divided into n heads, according to the number of heads in key and query layers. The information aggregation process is illustrated in Figure 29.

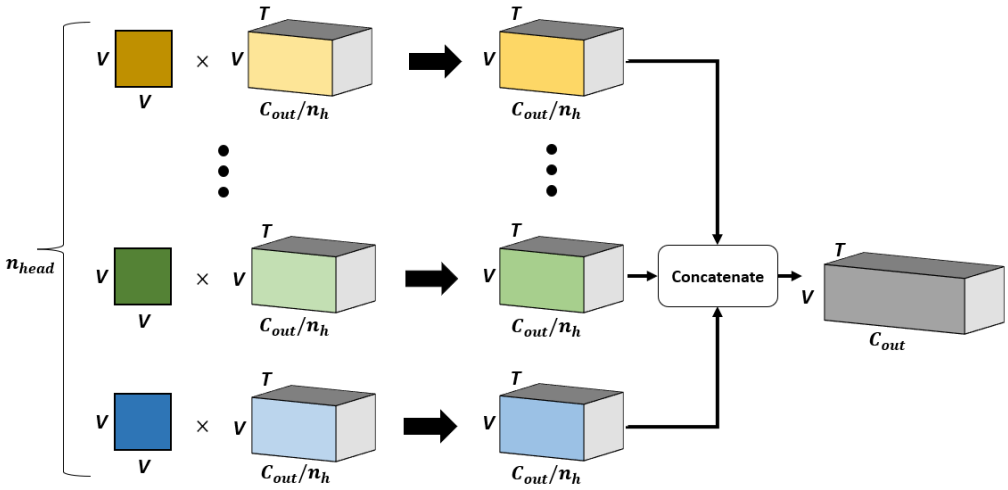


Figure 29. Head-wise information aggregation.

The multiplication between modified attention matrices with temporary sequence tensors at each head happens frame-wise, along the temporal dimension. The resulting tensors are then concatenated to produce the final embedding. At this stage, Global Pooling and the Softmax function can be applied to classify the samples. However, the valuable temporal information is wasted entirely. Therefore, it is necessary to consider methods for modeling temporal dependencies also.

4.2 Temporal modeling

Whenever mentioning sequential modeling, the first thought would be using the RNN or its most popular variance LSTM, due to their powerful capability. However, as previously discussed, their sequential modeling characteristics limit the computational efficiency. To overcome the limitation, the CNN is also recently used to model sequential data. But to accomplish a good balance between feature-extracting capability and computing speed, the CNN model needs to be very deep and complex. The attention mechanism has the potential to solve both problems, thanks to the non-Euclidean characteristic, and is currently the main building block of many state-of-the-art Natural Language Processing models.

Previous studies [43, 45] utilize multi-scale convolutions for extracting temporal features (Figure 30). Though good results have been achieved, the use of convolutions raises one problem. At each convolution layer, only information from local frames is aggregated.

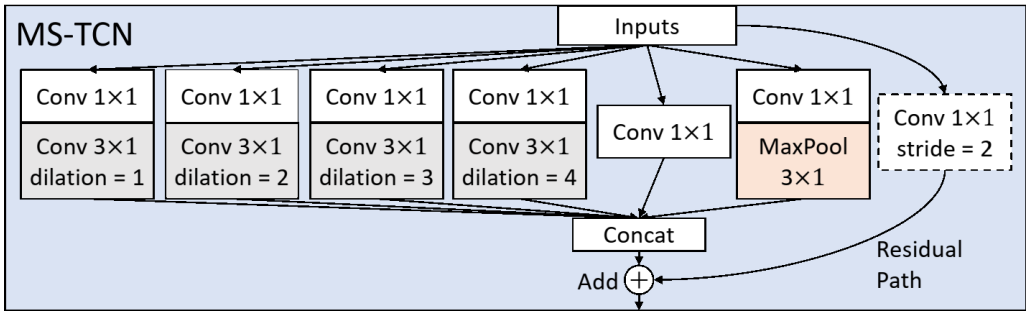
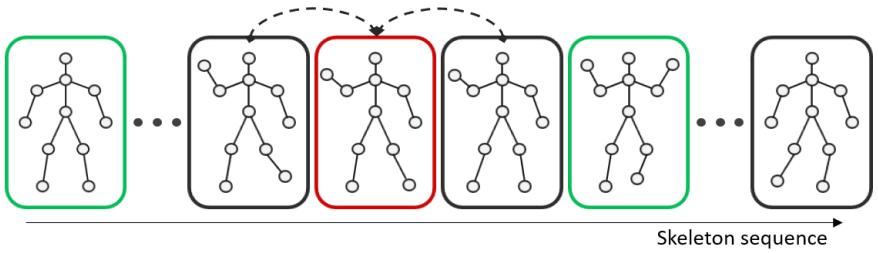
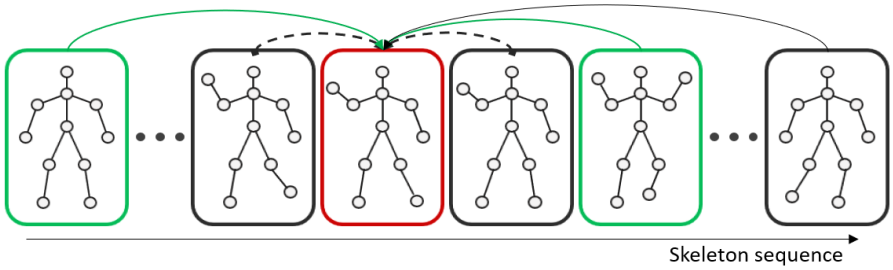


Figure 30. Multi-scale temporal convolution. [43]

Consecutive skeleton frames are similar in their structure. Discriminative features to effectively classify human actions are usually present in farther time steps. Therefore, using only convolutions for extracting local features limit the flow of useful information into calculations. To overcome the limitation, feature extraction needs to consider global information of the whole skeleton sequence. The proposed idea is illustrated in Figure 31.



(a)



(b)

Figure 31. The ability to pinpoint useful information along the sequence of self-attention: (a) Traditional temporal modeling with convolutions; (b) Temporal modeling with added self-attentions. (Red color denotes the frame of calculation and green frames indicate useful information. Farther frames can contain more discriminated features than the local ones.)

Self-attention is applied to temporally model the skeleton sequence sample. The task is essentially the same as the spatial modeling process. The algorithm also consists of query, key, and value. First, the input sequence is globally pooled along the spatial dimension. The pooled matrix is used to derive the query and key for computing the temporal attention score. The output tensor is produced by performing multiplication on the attention score and the value tensors. Figure 32 illustrates the temporal information extraction process.

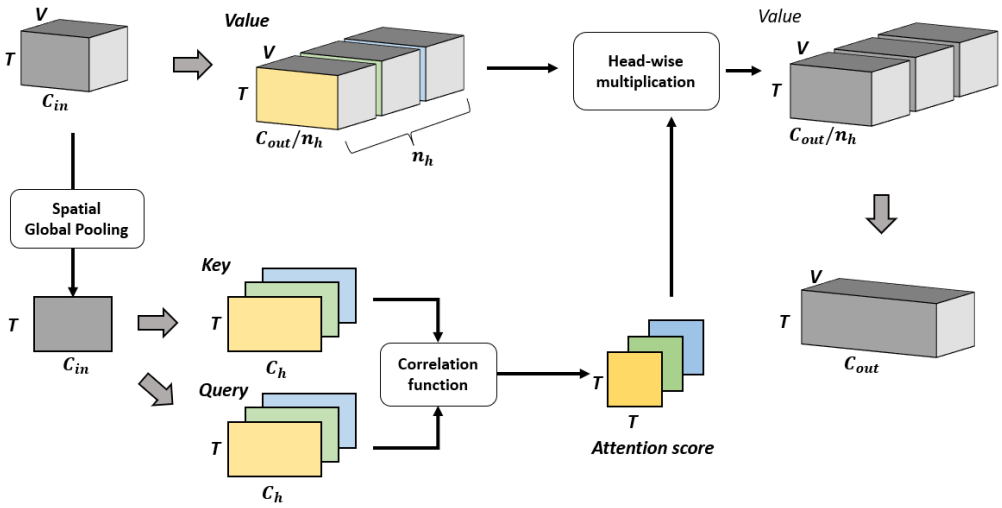


Figure 32. Attention modeling for temporal dependencies.

The multi-headed version of attention is also applied for stabilizing purposes. Because no prior information is given on how every frame in a sequence connects to each other, an obvious assumption is that frames in a sequence are inter-connected, and only the attention map is used for calculating the features.

By using temporal self-attention, long-range dependencies can be effectively extracted within one layer. However, neighborhood features are also very important for classification purposes. Thus, combining self-attentions with convolutions to model the temporal dimension of skeleton sequences is proposed. From the perspective of temporal dimension, skeleton data becomes the standard sequential modeling problem. The multi-scale convolution module from [43, 45] is adopted to extract local features across frames. Dilated convolutions are applied in the temporal convolution module. The main advantage of dilated convolution is the large receptive field. It can capture long-range dependencies while maintaining a small number of parameters. In the proposed model, a self-attention module for capturing long-range features is already used. Therefore, the convolution is changed back to the standard version. By using the standard convolutions, denser local features are effectively captured. The modified module is shown in Figure 33.

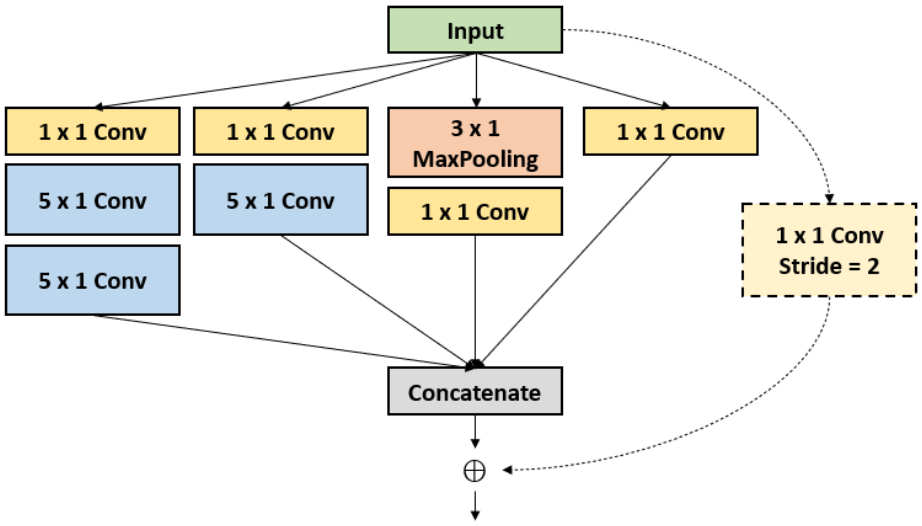


Figure 33. Multi-scale temporal convolution.

4.3 Model architecture

In this section, the method of combining all the presented modules into one feature extraction block STA-GCN is presented. The detailed architecture of the module is illustrated in Figure 34. For a fair comparison, the number of the STA-GCN blocks in the model is kept the same as [39], which is ten blocks of the STA-GCN connected sequentially.

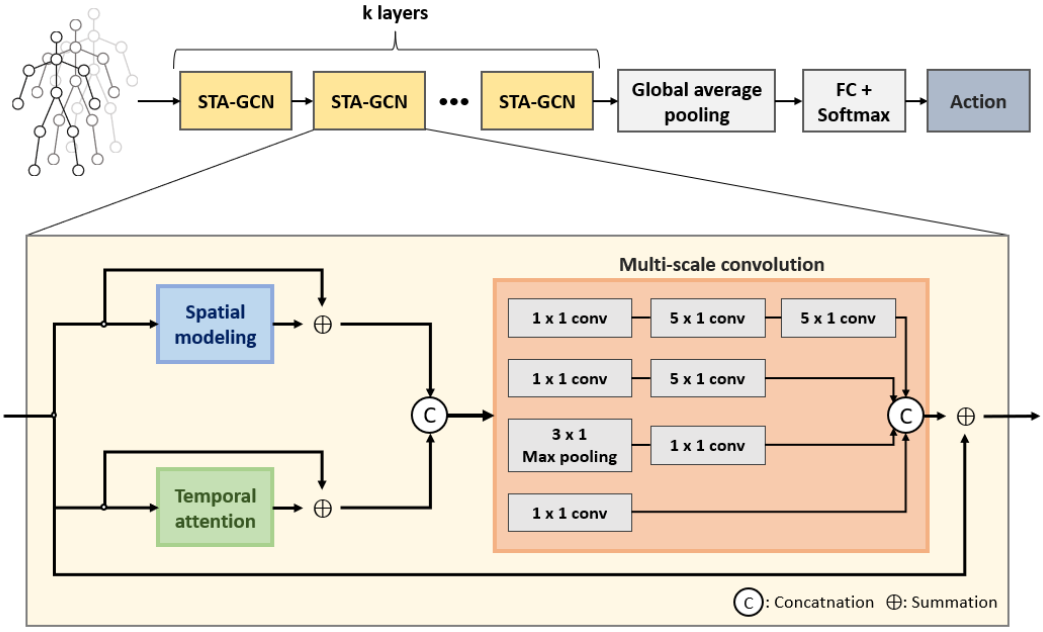


Figure 34. Spatial-temporal graph attention convolution module.

For a given action sample, k blocks of the STA-GCN are first applied to extract representation features. Each STA-GCN block contains three connected modules: spatial attention, temporal attention, and multi-scale convolution. The spatial attention module extracts skeleton-based dependencies using the adaptive GCN and the GAT. Temporal attention pinpoints and collects informative long-range frames into the calculation. Finally, a multi-scale convolution module is used to blend local temporal features to produce the representation of the skeleton sequence.

Because many layers are used for extracting features, residual connections are necessary to maintain the flow of gradients, which helps against gradient vanishing. Also, self-attentions in spatial and temporal modules get benefit from the residual connection as well. Since self-attentions are essentially graph neural networks, they suffer from "deep learning" due to their large-receptive field characteristic. Going too deep with graph neural networks can cause over-smoothing [58], where every node features, after multiple aggregations, start to be the same as each other. This problem can result in bad performance.

5 EXPERIMENTS

In this chapter, the selection of required datasets for skeleton-based action recognition is discussed. The criteria for evaluating the classification results are introduced next. Descriptions of the experiment and the final analysis of experimental results are mentioned in the last two sections.

5.1 Data

In this study, the proposed STA-GCN is tested on two skeleton-based action recognition datasets: NTU-RGB+D60 and NTU-RGB+D120.

NTU-RGB+D60 [11] is a large-scale dataset for action recognition that consists of 56,578 videos. The training samples are collected as skeleton sequences from 60 action classes, 40 distinct subjects, and 3 camera view angles. The data modalities were provided in this dataset: depth-map, 3D joint information, RGB frames, and IR sequences. In this study, only joint information is used for the action recognition task. In this thesis, only 3D joint information is used as the input data for the main classification model. The provided joint information contains 3D locations of 25 body joints according to Figure 35. Prior knowledge of joint connections of human kinetics is very important for the spatial modeling task. The given relationships between body joints in Figure 35 are encoded into an adjacency matrix during calculation.

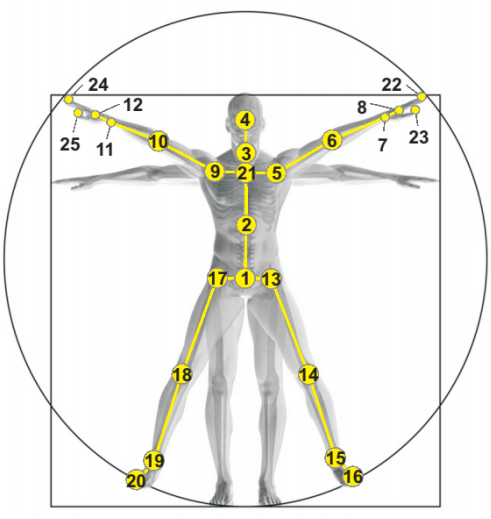


Figure 35. Human major joints were specified by the NTU-RGB+D60 dataset [11].

NTU-RGB+D120 [12] is the extension of the previous dataset. The updated version provides the addition of 57,367 skeleton sequences of 60 extended action classes. In total, the NTU-RGB+D120 consists of 113,945 training samples of 120 action classes, which were performed by 106 human subjects and captured from 32 different camera setups. With an addition of 60 classes and 66 subjects, the NTU-RGB+D120 introduces a much more challenging benchmark for skeleton-based action recognition. Figure 36 presents examples of skeleton-based action samples in the datasets.

There are more than 40 datasets have been created for skeleton-based action recognition in the past, but NTU-RGB+D is currently the largest official database. A detailed comparison between the related datasets and NTU-RGB+D is shown in Table 1. NTU RGB+D datasets outperform every other previous one in all aspects. A large number of samples brings major benefits to any deep learning model. Also, the datasets have a significantly higher number of subjects and camera views. Therefore, NTU RGB+D introduces more difficult testing settings. In particular, the large amount of variations in subjects and camera views makes it possible to have cross-subject and cross-view evaluations.

Table 1. Comparison between the NTU RGB+D and the other main publicly datasets for action recognition [11, 12].

Dataset		Samples	Classes	Subjects	Views	Modalities	Year
MSR-Action3D	[59]	567	20	10	1	D+3DJoints	2010
CAD-60	[60]	60	12	4	-	RGB+D+3DJoints	2011
RGBD-HuDaAct	[61]	1189	13	30	1	RGB+D	2011
MSRDailyActivity3D	[62]	320	16	10	1	RGB+D+3DJoints	2012
Act4^2	[63]	6844	14	24	4	RGB+D	2012
CAD-120	[64]	120	20	4	-	RGB+D+3DJoints	2013
3D ACtion Pairs	[65]	360	12	10	1	RGB+D+3DJoints	2013
Multiview 3D Event	[66]	3815	8	8	3	RGB+D+3DJoints	2013
Online RGB+D Action	[67]	336	7	24	1	RGB+D+3DJoints	2014
Northwestern-UCLA	[68]	1475	10	10	3	RGB+D+3DJoints	2014
UWA3D Multiview	[69]	900	30	10	1	RGB+D+3DJoints	2014
Office Activity	[70]	1180	20	10	3	RGB+D	2014
UTD-MHAD	[71]	861	27	8	1	RGB+D+3DJoints+ID	2015
UWA3D Multiview II	[72]	1075	30	10	5	RGB+D+3DJoints	2015
M^{2}I	[73]	1800	22	22	2	RGB+D+3DJoints	2015
SYSU 3DHOI	[74]	480	12	40	1	RGB+D+3DJoints	2017
NTU RGB+D 60	[11]	56880	60	40	80	RGB+D+IR+3DJoints	2016
NTU RGB+D 120	[12]	114480	120	106	155	RGB+D+IR+3DJoints	2019

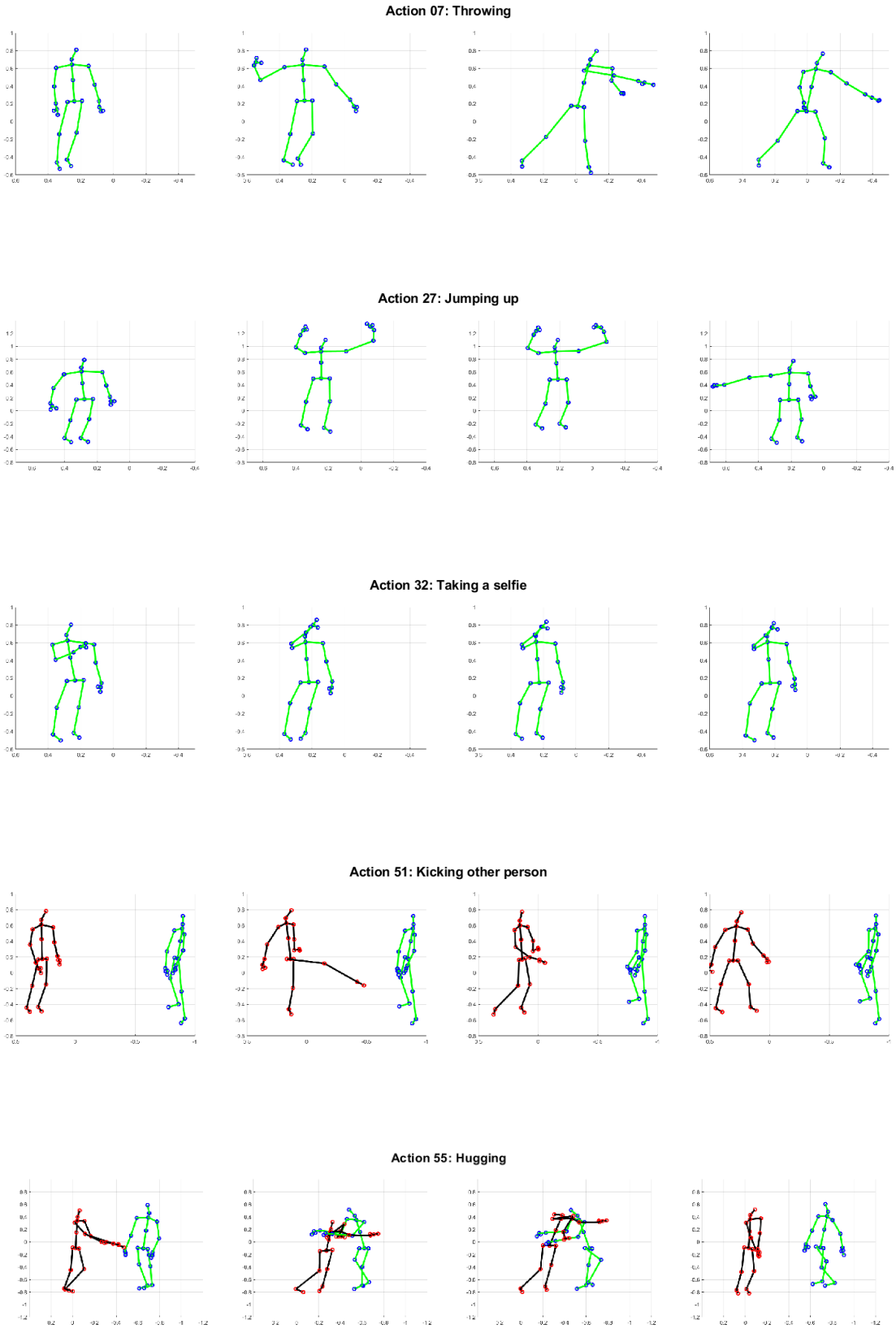


Figure 36. Examples of skeleton-based action samples from the NTU-RGB+D datasets.

5.2 Data pre-processing

The pre-processing of skeleton data includes three main steps: elimination of bad samples, sequence level translation, and sequence resizing. After retrieving the raw skeleton data, noisy and unusable samples need to be first eliminated from the datasets. Figure 37 presents examples of bad sample. Fortunately, the IDs of low-quality samples are provided in the implementation of [11,12], thus making the processing task more convenient.

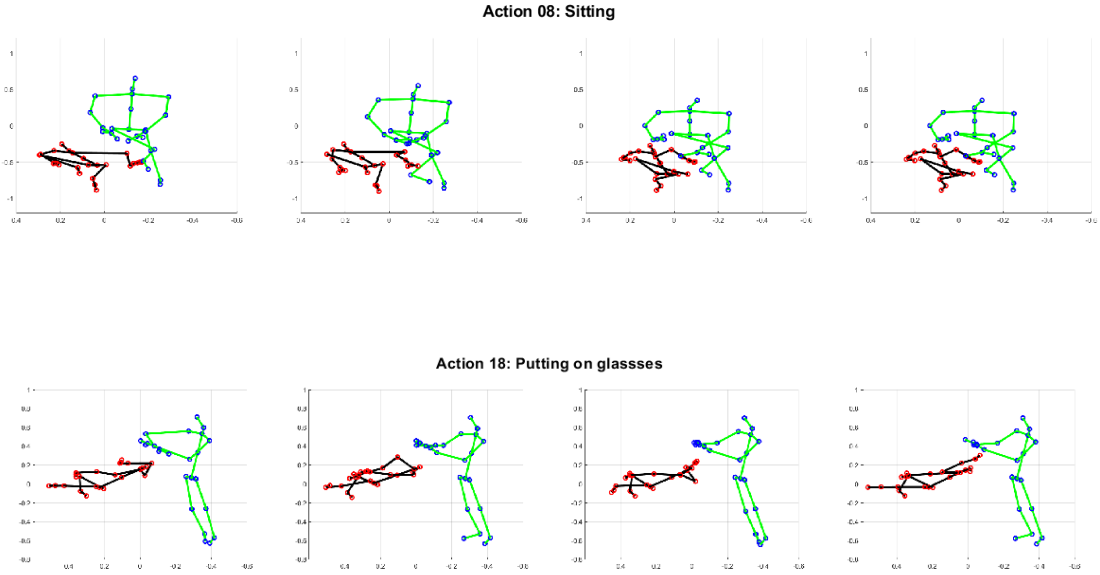


Figure 37. Examples of poor action samples.

The raw 3D skeletons are collected from the camera coordinate system, with the origin at the camera sensor. Similar to [75], sequence level translation is performed based on the first frame, to create an invariance from camera perspectives. In the translation process, the world coordinate at the camera sensor is moved to the center of the first frame.

The last step is padding and resizing skeleton sequences to the same length. Every skeleton sequence was first padded into a length of 300 by repeating the action. Then, a preprocessing from [76] is adapted. In particular, during training, sub-sequences from the original clip are randomly cropped, with a ratio of [0.5 1] from a uniform distribution. During testing, the cropping ratio is 0.95 and the sub-sequence is centered. Due to the variations in action lengths, input sequences are normalized to a fixed size of 64 in this experiment.

5.3 Evaluation criteria

Because skeleton-based action recognition is a standard classification task, typical metrics of deep learning can be used to evaluate the model after training. One way is to use the confusion matrix, which can effectively visualize the performance of classification. The confusion matrix consists of 4 terms: true positive (TP), false negative (FN), false positive (FP), and true negative (TN) for each action class. An TP is an outcome where the model correctly predicts the positive class. Similarly, an TN is a correct outcome when the model predicts a negative class. An FP occurs when the actual label is positive but the model predicts negative. Vice versa, an FN occurs when the actual label is negative but the model predicts positive. The TP, FN, FP, and TN can be used to infer the accuracy as follows:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{22}$$

The accuracy is calculated for every action class of the NTU-RGB+D60 and the NTU-RGB+D120. The overall performance of the proposed methods is the average accuracy over all classes.

Evaluation metrics of NTU-RGB+D60

To standardize the re-evaluation results on these benchmarks, the authors in [11] propose to measure the accuracy in two settings: Cross-subject and Cross-view. In the Cross-subject evaluation, 40 subjects are split evenly into the training and testing sets. The number of samples in the sets is 40,320 and 16,560 separately. In the Cross-view setting, samples collected from cameras 2 and 3 are used for training and samples from camera 1 for testing. The number of samples in the training set and the testing set are 37,920 and 18,960 separately.

Evaluation metrics of NTU-RGB+D120

Similar to [11], the authors in [12] also propose to measure the accuracy in two settings: Cross-subject and Cross-setup. In the Cross-subject evaluation, 106 subjects are split evenly into the training and testing sets, 53 subjects each. 63,026 samples are used for training and the rest 50,919 samples for testing. In the Cross-setup evaluation, 32 collection setups are split based on their ID. 54,468 samples of even-ID setups are used for training and 59,477 samples of odd-ID setups for testing.

5.4 Description of experiments

To evaluate the performance of the proposed spatio-temporal attention graph convolution module, an ablation experiment was conducted. First, with the same training hyper-parameters, a baseline model consisting of only graph convolutional networks and temporal convolutions was tested. Second, the proposed spatial adaptive attention and temporal attention were added. The number of parameters, the accuracy, and the mean accuracy of the last 10 training epochs are recorded. All ablation study experiments were conducted in the cross-subject setting of the NTU-RGB+D60 dataset. The NTU-RGB+D60 dataset was chosen due to its smaller size compared to the NTU-RGB+D120, thus making this process faster to implement. Based on the recorded performance of each training task, the model with the highest accuracy is chosen.

For the final evaluation of the proposed STA-GCN, multiple training with different data modalities are implemented, similar to [42, 44, 45]. In addition to the original data of skeleton joints, bone and motion modalities are also utilized for training. Thus, there are a total of four different experiments in each evaluation setting: joint, bone, joint-motion, and bone-motion. The model with different data modalities is tested on the NTU-RGB+D60 and the NTU-RGB+D120 datasets. At each evaluation metric, the performances of all four modalities including joint, bone, joint-motion, and bone-motion are recorded separately. Moreover, because of the overfitting problem, early stopping is applied to select the best model. The performance of the proposed method is calculated by assembling all the correct predictions of the modalities into one final value. The assembled accuracy is used to compare with other skeleton-based action recognition methods. Particularly, joint-and-bone fusion is used when comparing with [42, 43] and all four modalities are fused together when comparing with [44, 45].

The experiments are conducted on two NVIDIA Tesla V100 GPUs from Finland's CSC server with PyTorch deep learning framework. The model is trained with SGD with momentum 0.9, weight decay 0.0004, batch size 64, and an initial learning rate of 0.1 for 65 epochs. The learning rate is scheduled to decay with a rate of 0.1 at epochs 35 and 55. A warm-up strategy is adopted for the first 5 epochs to stabilize the training process. For two datasets NTU-RGB+D60 and NTU-RGB+D120, the data pre-processing from [45] is used, and all skeleton sequences are resized to 64 frames each.

5.5 Results

5.5.1 Ablation study

In this section, the proposed spatial-temporal attention graph convolution network is tested on the cross-subject evaluation setting on the NTU-RGB+D60 dataset. An architecture similar to ST-GCN [39] is deployed as the baseline. There are three modules that need to be studied in this section: adaptive GCN, additional GAT, and temporal self-attention modeling. Also, the 1D temporal convolution in ST-GCN [39] is changed to the multi-scale temporal convolution module for a fair comparison.

The experimental results are shown in Table 2. In the first experiment, the performance of the original baseline model with normal GCN and multi-scale convolutions is tested. Then, the adaptive characteristic is integrated into the GCN module to observe the improvement. In the third experiment, GAT is fused into the spatial modeling of the baseline. Self-attentions along temporal dependencies are separately considered in the fourth experiment to create the proposed STA-GCN. It can be observed that the accuracy of the classifiers increases gradually as more modules are added. Figure 38 shows the detailed classification results of the baseline model.

Table 2. Accuracy comparison for ablation study.

Methods	**Params.**	**Accuracy** (%)	**Mean last 10 epochs** (%)
Baseline [39]	843868	87.50	87.27
Baseline + adaptive GCN	850118	89.30	89.09
Baseline + adaptive GCN+GAT	1119422	89.88	89.72
STA-GCN	1174560	89.90	89.87

With an overall accuracy of 87.50%, the baseline performs fairly well on the NTU-RGB+D60 dataset. However, the baseline model struggles against many difficult classes, such as: eating snack (action 02), reading (action 11), writing (action 12), taking off shoes (action 17), playing with phone (action 29), sneezing (action 41). These classes differ by small changes in arm movements, thus creating a more challenging problem for skeleton-based action recognition. By integrating the proposed modules into the baseline model, an increase in performance is recorded, as shown in Figure 39.

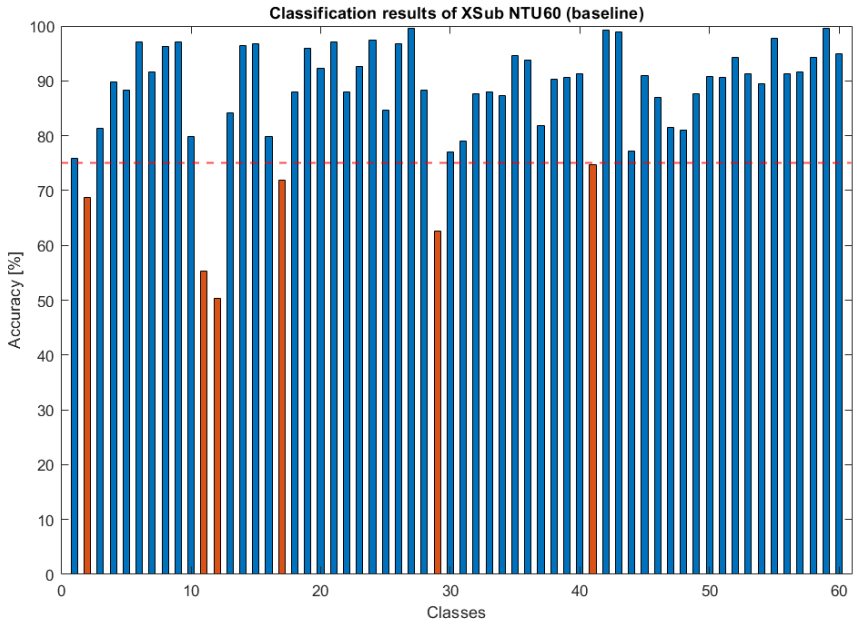


Figure 38. Classification accuracy of the baseline model on the NTU-RGB+D60 XSub setting. The threshold for good accuracy is chosen at 75% (red dash line). Classes considered to have poor performance are marked with red color.

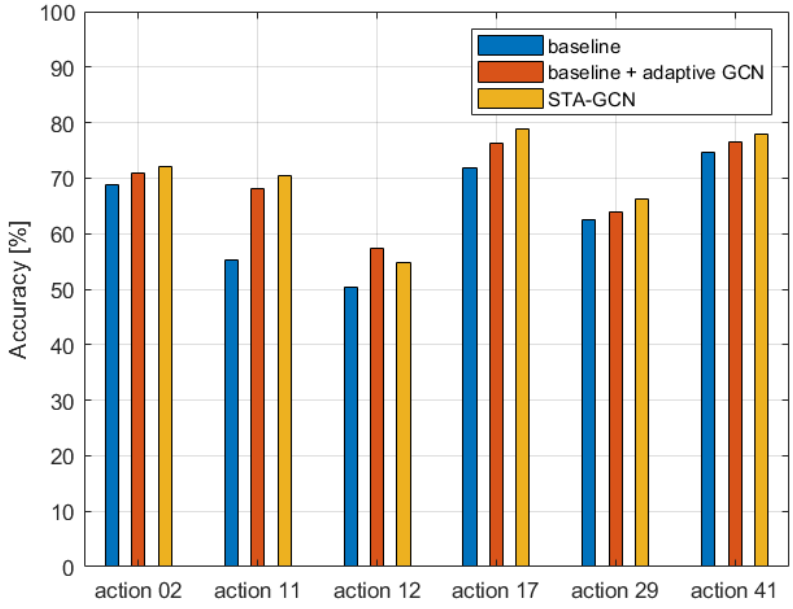


Figure 39. Comparison of performance on difficult classes.

As described in Section 4.2, the self-attention along the time domain has the ability to pinpoint the useful frames for calculation. Figure 40 shows the last 10 epochs of the training process. In the case of the model with only adaptive GCN and GAT in the spatial modeling, the accuracy oscillates between 89.6% to 89.9%, with a mean accuracy of 89.72%. While the performance of the model with the additional temporal self-attention is stable at around 89.9% and an average of 89.87%. Therefore, though the final accuracy of the two models is the same, the one with temporal attention proved to be more consistent and superior. Because of the proven reasons provided in this section, the STA-GCN module is chosen as the main building block of the final skeleton-based action recognition model.

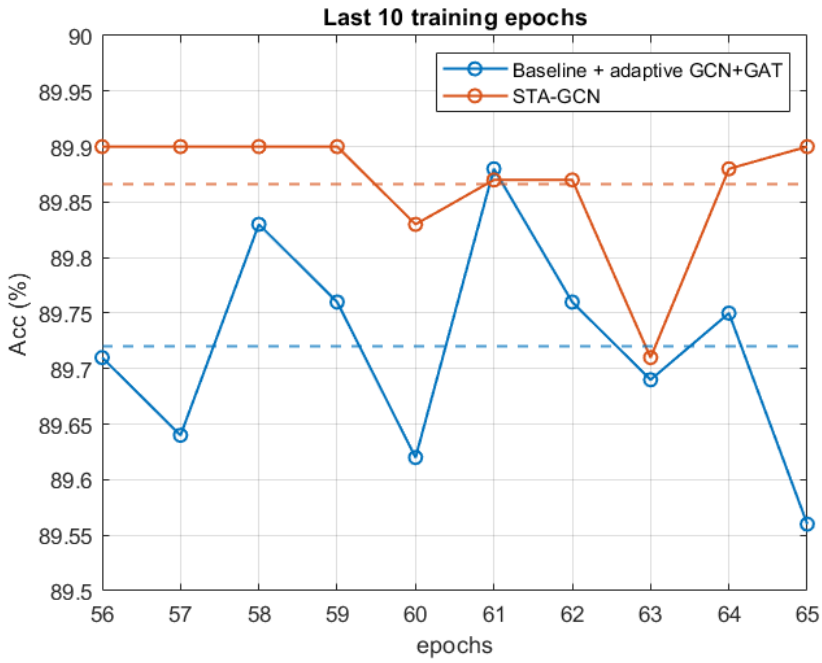


Figure 40. Comparison between two models: baseline + adaptive GCN+GAT and STA-GCN. The dash lines present the mean of the last 10 epochs.

5.5.2 Performance on the NTU-RGB+D60 and the NTU-RGB+D120

As mentioned in Section 5.4, the same multi-stream fusion framework as [42, 45] is adopted, by fusing four different modalities of data: joint, bone, joint motion, and bone motion. The comparisons in classification accuracy of the approach with other graph-based methods is demonstrated in Table 3. In the NTU-RGB+D60, the final STA-GCN model beat earlier methods [27, 34, 36], and some recent GNN-based methods such as [39, 40, 42–44], but still cannot outperform the current state-of-the-art [45] in Xsub and Xview. However, the classification result is equal to [45] in the cross-subject setup.

Table 3. Classification accuracy compared with state-of-the-art methods on the NTU-RGB+D60 and NTU-RGB+D120 datasets.

Methods	NTU-RGB+D60		NTU-RGB+D120	
	Xsub (%)	**Xview** (%)	**Xsub** (%)	**Xset** (%)
Lie Group [27]	50.1	52.8	-	-
Ind-RNN [34]	81.8	88.0	-	-
Temporal CNN [36]	74.3	83.1	-	-
ST-GCN [39]	81.5	88.3	-	-
AS-GCN [40]	86.8	94.2	-	-
2s-AGCN [42]	88.5	95.1	82.9	84.9
MS-G3D [43]	91.5	96.2	86.9	88.4
Dynamic GCN [44]	91.5	96.0	87.3	88.6
CTR-GCN [45]	92.4	96.8	88.9	90.6
STA-GCN (Joint only)	89.9	94.9	84.7	86.3
STA-GCN (Joint-motion)	87.4	93.3	81.4	83.1
STA-GCN (Bone only)	90.3	94.9	86.3	87.8
STA-GCN (Bone-motion)	87.4	91.9	81.2	83.0
STA-GCN (Joint + Bone)	92.0	96.4	88.4	90.0
STA-GCN (Net)	92.4	96.5	88.5	90.4

On the NTU-RGB+D120 dataset, the evaluation quality is similar to the NTU-RGB+D60. The best accuracy on both cross-subject and cross-setting of the model are only lower than [45]. Joint-bone fusion is considered when compare to [42, 43]. STA-GCN outperforms 2s-AGCN [42] (the original proposal of multi-modalities evaluation) by 5.5% in cross-subject and 5.1% in cross-setting. Compared to MS-G3D [43], STA-GCN has a 1.5% and 1.6% increase in performance of cross-subject and cross-setting separately.

A full evaluation of all four modalities (joint, bone, joint-motion, bone-motion) is considered when compared to the Dynamic GCN [44] and the CTR-GCN [45]. The STA-GCN achieves a better performance than Dynamic GCN, 1.1% increase in cross-subject and 1.8% increase in cross-setting. However, the STA-GCN under-performs behind CTR-GCN by 0.4% and 0.2% in cross-subject and cross-setting of the NTU-RGB+D120.

When determining the best performance model of each setting, early stopping was applied. Specifically, because decaying the learning rate causes overfitting, only the epoch that achieved the highest accuracy during training is considered the final model.

Cross-subject (XSub) evaluation on the NTU-RGB+D60 dataset. Figure 41 shows the cross-subject results of STA-GCN model during training on the NTU-RGB+D60. The best models of four modalities are collected at epochs 56, 60, 59, and 58 with the accuracy of 89.9%, 87.4%, 90.3%, and 87.4%, respectively. In Figure 42, the confusion matrix of the best performance model of each modality can be observed.

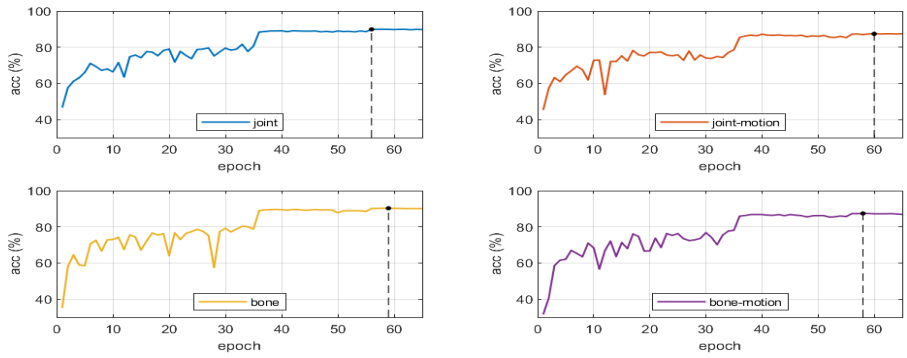


Figure 41. The testing accuracy of STA-GCN on the XSub setting of the NTU-RGB+D60.

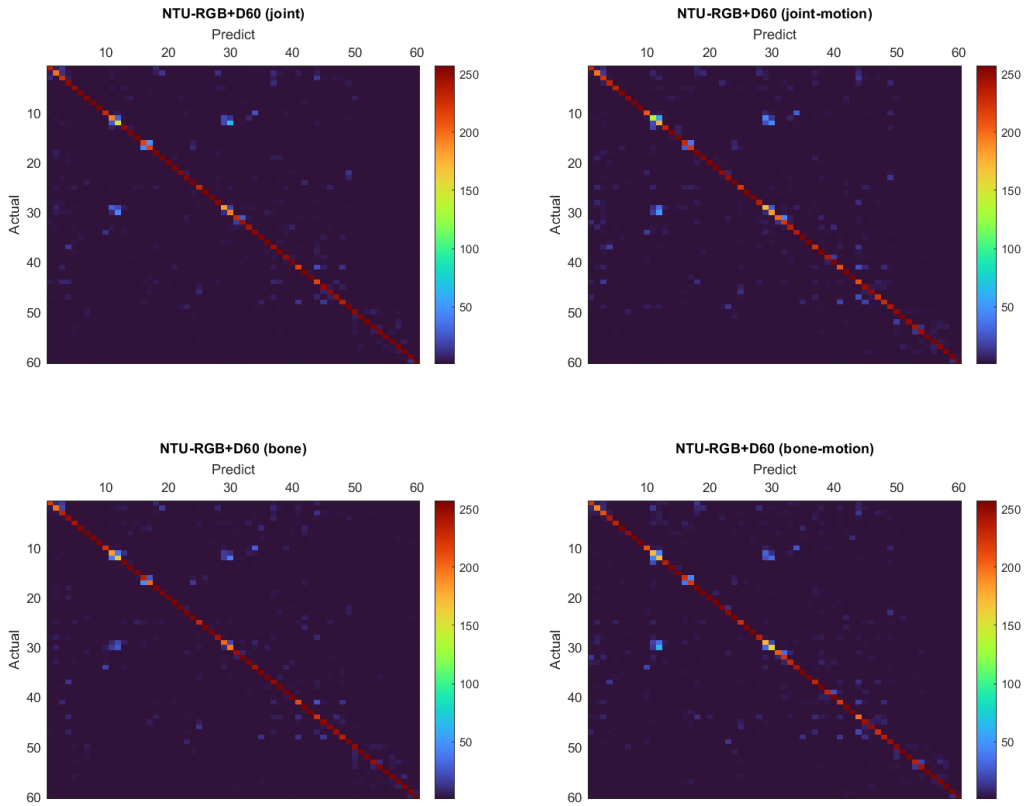


Figure 42. Confusion matrices of the best STA-GCN models on the NTU-RGB+D60 XSub.

Cross-view (XView) evaluation on the NTU-RGB+D60 dataset. Figure 43 shows the cross-view results of STA-GCN model during training on the NTU-RGB+D60. The best models of four modalities are collected at epochs 63, 61, 61, and 58 with the accuracy of 94.9%, 93.3%, 94.9%, and 91.9%, respectively. In Figure 44, the confusion matrix of the best performance model of each modality can be observed.

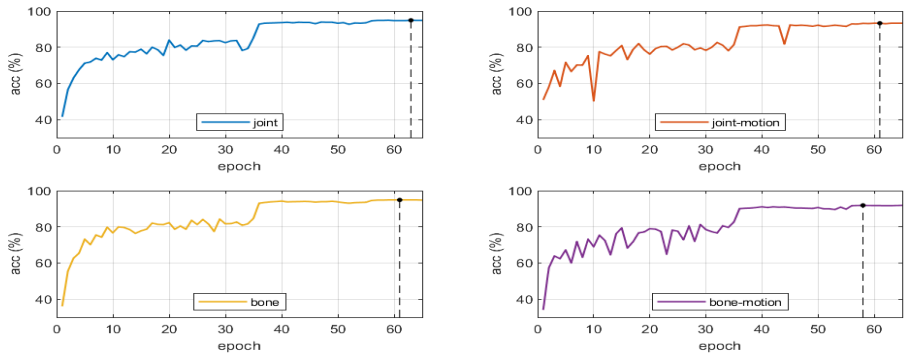


Figure 43. The testing accuracy of STA-GCN on the XView setting of the NTU-RGB+D60.

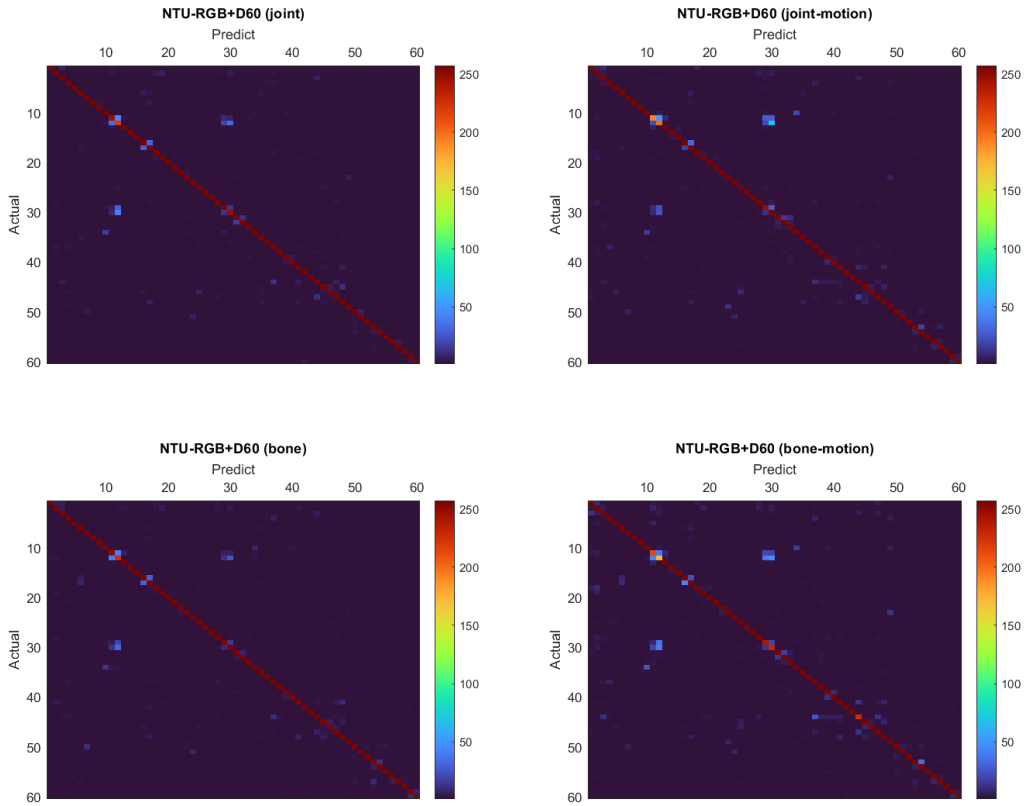


Figure 44. Confusion matrices of the best STA-GCN models on the NTU-RGB+D60 XView.

Cross-subject (XSub) evaluation on NTU-RGB+D120 dataset. Figure 45 shows the cross-subject results of STA-GCN model during training on the NTU-RGB+D120. The best models of four modalities are collected at epochs 58, 60, 58, and 61 with the accuracy of 84.7%, 81.4%, 86.3%, and 81.2%, respectively. In Figure 46, the confusion matrix of the best performance model of each modality can be observed.

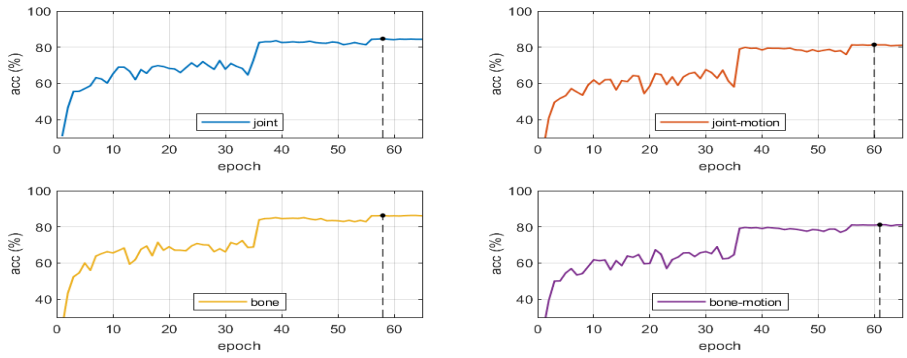


Figure 45. The testing accuracy of STA-GCN on the XSub setting of the NTU-RGB+D120.

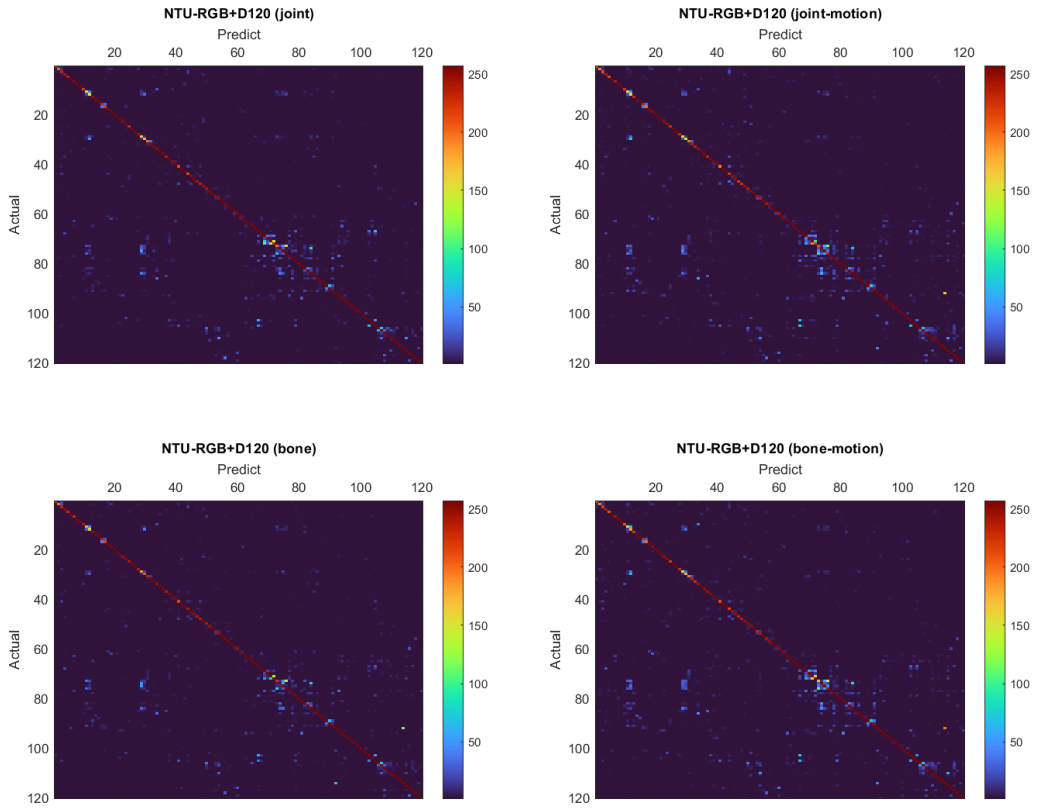


Figure 46. Confusion matrices of the best STA-GCN models on the NTU-RGB+D120 XSub.

Cross-setting (XSet) evaluation on NTU-RGB+D120 dataset. Figure 47 shows the cross-setup results of STA-GCN model during training on the NTU-RGB+D120. The best models of four modalities are collected at epochs 63, 64, 62, and 62 with the accuracy of 86.3%, 83.1%, 87.8%, and 83.0%, respectively. In Figure 48, the confusion matrix of the best performance model of each modality can be observed.

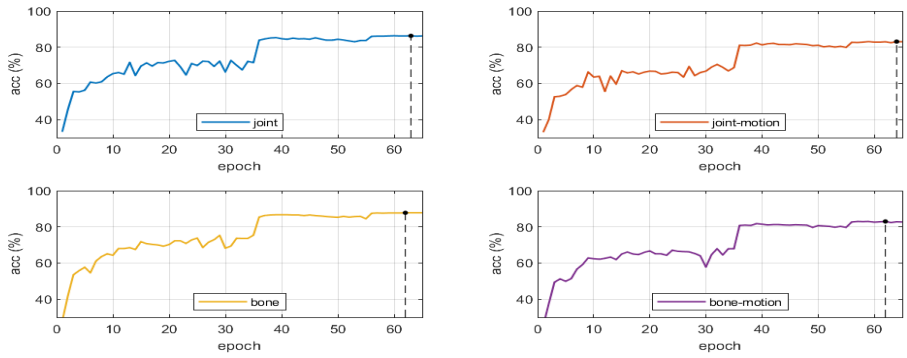


Figure 47. The testing accuracy of STA-GCN on the XSet setting of the NTU-RGB+D120.

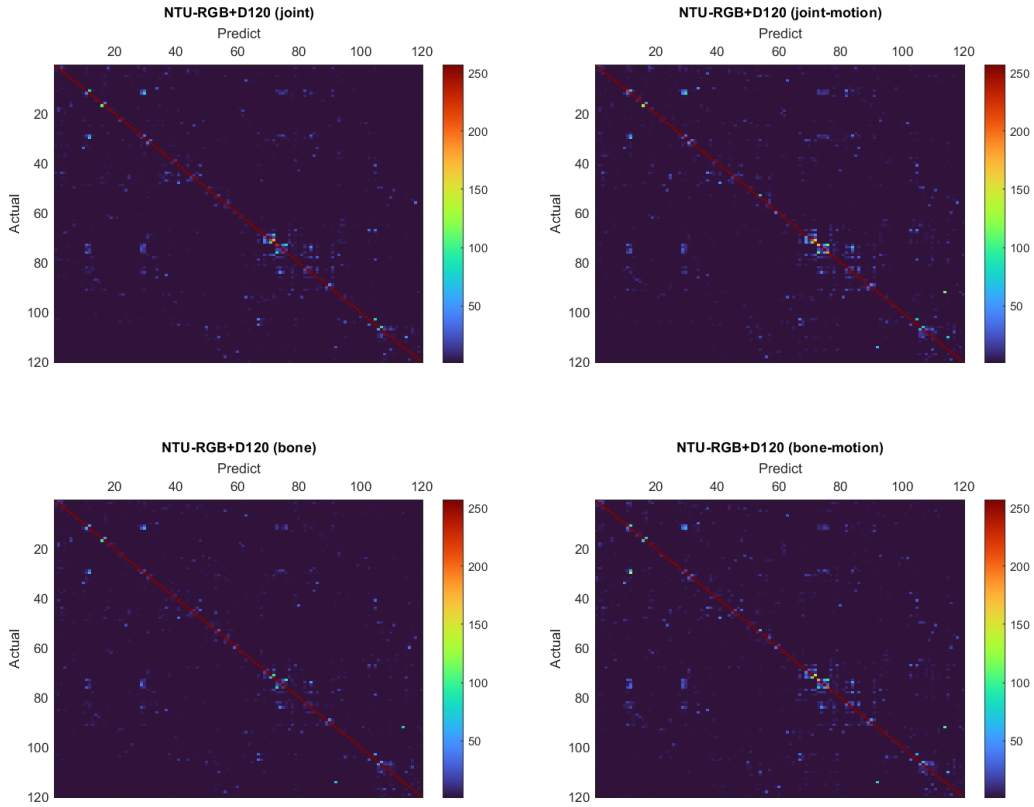


Figure 48. Confusion matrices of the best STA-GCN models on the NTU-RGB+D120 XSet.

6 DISCUSSION

6.1 Current study

Action recognition is the pioneering step toward human activity analysis. Therefore, to further broaden our understanding in this line of research, a method for correctly recognizing human action from digital input data is highly valuable. In this thesis, a novel skeleton-based human action recognition method named Spatio-Temporal Attention Graph Convolutional Networks STA-GCN is introduced. The proposed technique utilizes a combination of graph neural networks and self-attention mechanisms for modeling spatial and temporal dependencies of the skeleton data. One key contribution of this study is the proposal of temporal modeling using self-attentions. In particular, given a skeleton sequence, self-attentions can pinpoint the most beneficial time steps for the main calculations of STA-GCN, thus resulting in a better representation of the input data.

The experiment results demonstrate the high performance of the method on two large-scale skeleton-based datasets: the NTU-RGB+D60 and the NTU-RGB+D120. The final classification results of the model are very competitive with the current state-of-the-arts. The low number of parameters of STA-GCN makes it a very promising approach for a fast and accurate skeleton-based human action recognition solution.

Despite the advantages, the proposed STA-GCN still suffers from difficult samples. Actions that only differ from each other by small movement in the arms can cause confusion for skeleton-based methods in general. Even though STA-GCN is proven to be better at solving this problem compared to earlier methods, the final classification results are still not very high. Solutions for extracting more information from human objects in addition to skeleton data are much needed for addressing this problem.

6.2 Future work

Due to the good accuracy and the low number of parameters, STA-GCN can be implemented in real-time surveillance systems in the future. For instance, the proposed system can be installed at office entrances to analyze people's activities and behaviors. Based on the information, bad intentions or suspicious actions can be prevented in advance. The lightweight characteristic of the system makes it easy to be installed and re-trained. Therefore, this could be a very promising commercial product.

One limitation of the current study of STA-GCN is the ability to differentiate similar-looking actions. In the future, a further in-depth study on the micro gestures of skeleton data can be pursued. The current human topology provided in NTU-RGB+D60 and NTU-RGB+D120 datasets is not focusing on the details of the hands and the fingers. Therefore, another method for extracting skeleton data from small movements can be applied, such as OpenPose. By using the extracted information on the hand movements, an improvement for STA-GCN can be developed and expected to achieve even better performance.

In general, skeleton-based methods are good at recognizing human actions. However, the skeleton is an implicit type of data based on the current understanding of human kinetics. Human interacts with the surrounding environment through the skin, not the bone. Therefore, a more sophisticated method for modeling human structure is highly required. From the understanding of skeleton data, a full 3D representation of the human body can be generated. Then, a line of research for human activities and interactions from a real-world perspective can potentially be explored.

7 CONCLUSION

In this thesis, the capability of graph neural network (GNN) on skeleton-based action recognition was investigated. Existing earlier methods were reviewed and shortly documented. The principle of GNN, and its related method GCN, GAT were carefully studied. Based on the survey, a novel method for skeleton-based action recognition was proposed, named Spatio-Temporal Attention Graph Convolutional Network, or STA-GCN in short. The proposed method was implemented and trained on two large-scale datasets: the NTU-RGB+D 60 and the NTU-RGB+D 120. The evaluation results are very competitive to the current state-of-the-arts, thus proving the strong performance of the STA-GCN. Finally, as a continuation of the research, methods for extracting more detailed information on micro-gestures or full-scale modeling of the human body were suggested for overcoming the limitation of STA-GCN.

REFERENCES

[1] Yu Kong and Yun Fu. Human action recognition and prediction: A survey. https://arxiv.org/abs/1806.11230, 2022.

[2] Xin Liu, Henglin Shi, Haoyu Chen, Zitong Yu, Xiaobai Li, and Guoying Zhao. imigue: An identity-free video dataset for micro-gesture understanding and emotion analysis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10631–10642, 2021.

[3] Zitong Yu, Benjia Zhou, Jun Wan, Pichao Wang, Haoyu Chen, Xin Liu, Stan Z. Li, and Guoying Zhao. Searching multi-rate and multi-modal temporal enhanced networks for gesture recognition. *IEEE Transactions on Image Processing*, 30:5626–5640, 2021.

[4] Haoyu Chen, Xin Liu, Jingang Shi, and Guoying Zhao. Temporal hierarchical dictionary guided decoding for online gesture segmentation and recognition. *IEEE Transactions on Image Processing*, 29:9689–9702, 2020.

[5] Lei Wang, Du Q. Huynh, and Piotr Koniusz. A comparative review of recent kinect-based action recognition algorithms. *IEEE Transactions on Image Processing*, 29:15–28, 2020.

[6] Zehua Sun, Jun Liu, Qiuhong Ke, Hossein Rahmani, Mohammed Bennamoun, and Gang Wang. Human action recognition from various data modalities: A review. https://arxiv.org/abs/2012.11866, 2020.

[7] Xin Liu and Guoying Zhao. 3d skeletal gesture recognition via discriminative coding on time-warping invariant riemannian trajectories. *IEEE Transactions on Multimedia*, 23:1841–1854, 2021.

[8] Xin Liu, Henglin Shi, Xiaopeng Hong, Haoyu Chen, Dacheng Tao, and Guoying Zhao. 3d skeletal gesture recognition via hidden states exploration. *IEEE Transactions on Image Processing*, 29:4583–4597, 2020.

[9] Xin Liu, Henglin Shi, Xiaopeng Hong, Haoyu Chen, Dacheng Tao, and Guoying Zhao. Hidden states exploration for 3d skeleton-based gesture recognition. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1846–1855, 2019.

[10] Xin Liu and Guoying Zhao. 3d skeletal gesture recognition via sparse coding of time-warping invariant riemannian trajectories. In *International Conference on Multimedia Modeling*, pages 678–690, 2019.

[11] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1010–1019, 2016.

[12] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C Kot. Ntu rgb+d 120: A large-scale benchmark for 3d human activity understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 42(10):2684–2701, 2020.

[13] Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 1973.

[14] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with multi-stream adaptive graph convolutional networks. *IEEE Transactions on Image Processing*, 29:9532–9545, 2020.

[15] Liliana Lo Presti and Marco La Cascia. 3d skeleton-based human action classification: A survey. *Pattern Recognition*, 53:130–147, 2016.

[16] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2938–2946, 2015.

[17] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4929–4937, 2016.

[18] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5686–5696, 2019.

[19] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 43(1):172–186, 2021.

[20] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1297–1304, 2011.

[21] Angel Martínez-González, Michael Villamizar, Olivier Canévet, and Jean-Marc Odobez. Real-time convolutional networks for depth-based human pose estimation.

In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 41–47, 2018.

[22] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 19(2):4–10, 2012.

[23] Štěpán Obdržálek, Gregorij Kurillo, Ferda Ofli, Ruzena Bajcsy, Edmund Seto, Holly Jimison, and Michael Pavel. Accuracy and robustness of kinect pose estimation in the context of coaching of elderly population. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1188–1193, 2012.

[24] Fei Han, Brian Reily, William A. Hoff, and Hao Zhang. Space-time representation of people based on 3d skeletal data: A review. *Computer Vision and Image Understanding*, 158:85–105, 2017.

[25] Xiaodong Yang and Ying Li Tian. Eigenjoints-based action recognition using naïve-bayes-nearest-neighbor. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 14–19, 2012.

[26] Mohammad A. Gowayyed, Marwan Torki, Mohamed E. Hussein, and Motaz El-Saban. Histogram of oriented displacements (hod): Describing trajectories of human joints for action recognition. In *International Joint Conference on Artificial Intelligence*, pages 1351–1357, 2013.

[27] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 588–595, 2014.

[28] Piotr Koniusz, Anoop Cherian, and Fatih Porikli. Tensor representations via kernel linearization for action recognition from 3d skeletons. In *European Conference on Computer Vision (ECCV)*, pages 37–53, 2016.

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.

[30] Yann Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[31] Yong Du, Wei Wang, and Wang Liang. Hierarchical recurrent neural network for skeleton based action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1110–1118, 2015.

[32] Wentao Zhu, Cuiling Lan, Junliang Xing, Wenjun Zeng, Yanghao Li, Li Shen, and Xiaohui Xie. Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In *AAAI Conference on Artificial Intelligence*, pages 3697–3703, 2016.

[33] Hongsong Wang and Liang Wang. Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3633–3642, 2017.

[34] Jun Liu, Amir Shahroudy, Dong Xu, Alex C. Kot, and Gang Wang. Skeleton-based action recognition using spatio-temporal lstm network with trust gates. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 40(12):3007–3021, 2018.

[35] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Skeleton-based action recognition with convolutional neural networks. https://arxiv.org/abs/1704.07595, 2017.

[36] Tae Soo Kim and Austin Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1623–1631, 2017.

[37] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Sohel, and Farid Boussaid. A new representation of skeleton sequences for 3d action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4570–4579, 2017.

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[39] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI Conference on Artificial Intelligence*, pages 3482–3489, 2018.

[40] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3590–3598, 2019.

[41] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning (ICML)*, pages 2693–2702, 2018.

[42] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12018–12027, 2019.

[43] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 140–149, 2020.

[44] F. Ye, S. Pu, Q. Zhong, C. Li, D. Xie, and H. Tang. Dynamic gcn: Context-enriched topology learning for skeleton-based action recognition. In *ACM International Conference on Multimedia*, pages 55–63, 2020.

[45] Yuxin Chen, Ziqi Zhang, Chunfeng Yuan, Bing Li, Ying Deng, and Weiming Hu. Channel-wise topology refinement graph convolution for skeleton-based action recognition. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13359–13368, 2021.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *International Conference on Neural Information Processing Systems (NIPS)*, pages 6000–6010, 2017.

[47] C. Van Der Malsburg. Frank rosenblatt: Principles of neurodynamics: Perceptrons and the theory of brain mechanisms. In *Brain Theory*, pages 245–248, 1986.

[48] Thomas Norbert Kipf. *Deep learning with graph-structured representations*. PhD thesis, University of Amsterdam, The Netherlands, 2020.

[49] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT Press, 2015.

[50] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. https://arxiv.org/abs/2104.13478, 2021.

[51] William L. Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159, 2020.

[52] Thomas Norbert Kipf. Graph convolutional networks. http://tkipf.github.io/graph-convolutional-networks/, 2016.

[53] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and Dahl George E. Neural message passing for quantum chemistry. https://arxiv.org/abs/1704.01212, 2017.

[54] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[55] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[56] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*, 2015.

[57] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[58] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations (ICLR)*, 2020.

[59] Wanqing Li, Zhengyou Zhang, and Zicheng Liu. Action recognition based on a bag of 3d points. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops (CVPRW)*, pages 9–14, 2010.

[60] Jaeyong Sung, Colin Ponce, Bart Selman, and Ashutosh Saxena. Human activity detection from rgbd images. In *AAAI Conference on Plan, Activity, and Intent Recognition*, pages 47–55, 2011.

[61] Bingbing Ni, Gang Wang, and Pierre Moulin. Rgbd-hudaact: A color-depth video database for human daily activity recognition. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1147–1153, 2011.

[62] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1290–1297, 2012.

[63] Zhongwei Cheng, Lei Qin, Yituo Ye, Qingming Huang, and Qi Tian. Human daily action analysis with multi-view and color-depth data. In *European Conference on Computer Vision Workshops and Demonstrations*, pages 52–61, 2012.

[64] Hema Swetha Koppula, Rudhir Gupta, and Ashutosh Saxena. Learning human activities and object affordances from rgb-d videos. *The International Journal of Robotics Research*, 32(8):951–970, 2013.

[65] Omar Oreifej and Zicheng Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 716–723, 2013.

[66] Ping Wei, Yibiao Zhao, Nanning Zheng, and Song-Chun Zhu. Modeling 4d human-object interactions for event and object recognition. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3272–3279, 2013.

[67] Gang Yu, Zicheng Liu, and Junsong Yuan. Discriminative orderlet mining for real-time recognition of human-object interaction. In *Asian Conference on Computer Vision (ACCV)*, pages 50–65, 2014.

[68] Jiang Wang, Xiaohan Nie, Yin Xia, Ying Wu, and Song-Chun Zhu. Cross-view action modeling, learning, and recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2649–2656, 2014.

[69] Hossein Rahmani, Arif Mahmood, Du Huynh, and Ajmal Mian. Hopc: Histogram of oriented principal components of 3d pointclouds for action recognition. In *European Conference on Computer Vision (ECCV)*, pages 742–757, 2014.

[70] Keze Wang, Xiaolong Wang, Liang Lin, Meng Wang, and Wangmeng Zuo. 3d human activity recognition with reconfigurable convolutional neural networks. In *ACM International Conference on Multimedia*, pages 97–106, 2014.

[71] Chen Chen, Roozbeh Jafari, and Nasser Kehtarnavaz. Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In *IEEE International Conference on Image Processing (ICIP)*, pages 168–172, 2015.

[72] Hossein Rahmani, Arif Mahmood, Du Huynh, and Ajmal Mian. Histogram of oriented principal components for cross-view action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 38(12):2430–2443, 2016.

[73] Ning Xu, Anan Liu, Weizhi Nie, Yongkang Wong, Fuwu Li, and Yuting Su. Multi-modal & multi-view & interactive benchmark dataset for human action recognition. In *ACM International Conference on Multimedia*, pages 1195–1198, 2015.

[74] Jian-Fang Hu, Wei-Shi Zheng, Jianhuang Lai, and Jianguo Zhang. Jointly learning heterogeneous features for rgb-d activity recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(11):2186–2200, 2017.

[75] Pengfei Zhang, Cuiling Lan, Wenjun Zeng, Junliang Xing, Jianru Xue, and Nanning Zheng. Semantics-guided neural networks for efficient skeleton-based human action recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1112–1121, 2020.

[76] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation. In *International Joint Conference on Artificial Intelligence*, pages 786–792, 2018.