A Feature-enhanced shift graph convolutional network and its application in skeleton-based action recognition

MS (Research) Thesis

By

Ananya Roy

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

July 2023

A Feature-enhanced shift graph convolutional network and its application in skeleton-based action recognition

A THESIS

submitted to the

INDIAN INSTITUTE OF TECHNOLOGY INDORE

in fulfillment of the requirements for

the award of the degree

of

Master of Science (Research)

By

Ananya Roy

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

July 2023

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled Skeleton-based Human Action Recognition using Graph Convolutional Network (GCN)in thefulfillment of the requirements for the award of the degree of MASTER OF SCIENCE (RESEARCH)and submitted in the DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING, Indian Institute of Technology Indore, is an authentic record of my own work carried out during the time period from August 2021 to May 2023 under the supervision of Prof. Aruna Tiwari, Professor, Indian Institute of Technology Indore, India, and Dr. Sanjay Singh, Principal Scientist and Group Head, Advanced Information Technologies Group, CSIR – Central Electronics Engineering Research Institute (CSIR-CEERI), Pilani, India.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

Ananya Roy

27/06/2024

Signature of the Student with Date

(Ananya Roy)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Signature of Thesis Supervisor 1

with Date

(Prof. Aruna Tiwari)

Signature of Thesis Supervisor 2

with Date

(Dr. Sanjay Singh)

Ananya Roy has successfully given her MS (Research) Oral Examination held on 27/06/2024

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my heartfelt gratitude to a number of persons who in one or the other way contributed by making this time as learnable, enjoyable, and bearable. At first, I would like to thank my supervisors **Prof. Aruna Tiwari** and **Dr. Sanjay Singh**, who were a constant source of inspiration during my work. Without their constant guidance and research directions, this research work could not be completed. Their continuous support and encouragement has motivated me to remain streamlined in my research work.

I am thankful to the PSPC members, **Prof. Kapil Ahuja** and **Dr. Amod Umarikar** for taking out some valuable time to evaluate my progress throughout my journey in IIT Indore. Their good comments and suggestions helped me to improve my work at various stages. I am also grateful to **Dr. Somnath Dey**, HOD of department of Computer Science and Engineering for his help and support.

My sincere acknowledgement and respect to **Prof. Suhas Joshi** , Director, Indian Institute of Technology Indore for providing me the opportunity to explore my research capabilities at Indian Institute of Technology Indore.

I extend my sincere thanks to the **CSIR-CEERI Pilani** for providing me the necessary infrastructure and technology for my research. I would also like to thank my seniors, **Mr. Rituraj** from IIT Indore and **Mr. Sumeet Saurav** from CEERI-Pilani for sharing their experience and guiding me throughout my research.

I would like to appreciate the fine company of my dearest friends Shibani, Bharath, Debankan, Nisha, Achint and Deepak who have supported me and taken care of me throughout my stay in IIT Indore. I am also grateful to the institute staffs for their unfailing support and assistance.

I would like to express my heartfelt respect to my parents, **Mr. Rajesh Roy** and **Mrs. Rupa Roy** for the love, care and support they have provided to me throughout my life. Finally, I am thankful to all who directly or indirectly contributed, helped and supported me.

Ananya Roy

To my family and friends

Abstract

Human action recognition is the process of automatically identifying and classifying human actions in a video sequence. It involves analyzing the motion and appearance of humans in the video and recognizing the action they are performing. Human actions can be represented using various data modalities, such as RGB videos, skeleton graphs, depth sequences and heat maps. In recent years, skeleton-based action recognition has drawn a lot of attention in the area of computer vision. To extract skeletal data, pose estimation algorithms are used on action videos to track the key joints involved in the action. These joints are connected with edges representing the bones involved in the action, thus forming a graph structure of the human action. Skeleton data is lightweight as compared to video data, and is more robust against changes in appearance, lighting conditions, background clutter and camera viewpoints. Among existing methods, Graph Convolutional Networks (GCNs) have achieved exceptional results as they are highly efficient in feature extraction from non-euclidean or irregular data. However, most existing GCN-based methods are computationally expensive and have inflexible receptive fields, due to which their expressiveness is limited. As a result, focus has shifted towards building lightweight architectures which require fewer parameters. One such method is Shift-GCN [1], which uses shift graph operations that are both lightweight and increase the flexibility of receptive fields in both spatial and temporal dimensions. However, although this method captures non-local and distant spatial relationships in a lightweight and more efficient manner, it does not perform well on fine-grained actions that have subtle differences and require capturing graph connection information.

In this thesis, we propose a Feature-enhanced shift graph convolutional network (FES-GCN) which utilizes graph connectivity knowledge and combines it with the input feature, thus generating a richer and more abundant feature map before performing shift operation. The shift module is lightweight as compared to a regular convolution operation and provides more flexible spatial-temporal receptive fields. We meticulously analyze our proposed method, perform exhaustive ablation studies on

each module and also compare its performance with various state-of-the-art action recognition models. Our proposed model has shown the ability to improve the accuracy of human action recognition task on three benchmark skeleton datasets, while requiring a computational cost less than most state-of-the-art approaches.

Keywords: Skeleton-based Action Recognition, Graph Convolutional Network (GCN), Shift-GCN, FES-GCN, Enhanced Feature Representation.

List of Publications

In Refereed Journals

A. Roy, A. Tiwari, S. Saurav and S. Singh. *Feature-enhanced shift graph convolutional network for skeleton-based human action recognition*, Signal, Image and Video Processing, Springer, Impact factor: 2.553 (Withdrawn since Journal took too long to give decision and was unresponsive to emails.)

A. Roy, A. Tiwari, S. Saurav and S. Singh. *Enhancing skeleton-based action recognition using a knowledge-driven shift graph convolutional network*, Computers and Electrical Engineering, Elsevier, Impact factor: 4.3 (Second revision submitted on June 23, 2024)

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

GCN Graph Convolutional Network

CNN Convolutional Neural Network

RNN Recurrent Neural Network

HAR Human Action Recognition

GFLOPs Giga Floating Point Operations

FES-GCN Feature-enhanced Shift Graph Convolutional Network

ST-GCN Spatial-Temporal Graph Convolutional Network

2s-AGCN Two-stream Adaptive Graph Convolutional Network

MS-AAGCN Multi-stream Adaptive Attention Enhanced Graph Convolutional Network

DGNN Directed Graph Neural Network

STGR Spatio-Temporal Graph Routing

NAS Neural Architecture Search

DAG Directed Acyclic Graph

ReLU Rectified Linear Unit

CIFAR Canadian Institute for Advanced Research

SGCN Spatial Graph Convolutional Network

TGCN Temporal Graph Convolutional Network

GCB Graph Convolutional Block

GAP Global Average Pooling

SGD Stochastic Gradient Descent

GPU Graphics Processing Unit

Chapter 1

Introduction

Human action recognition is a time-series classification task, which aims to understand human behaviour and assign a label to each action. Action recognition has been extensively researched in recent years, as it finds application in many tasks, such as intelligent security, healthcare, video surveillance, virtual reality, self-driving cars, etc. Human actions can be represented using various data modalities, such as RGB videos, skeleton graphs, depth sequences, heat maps, and so on. Among them, video-based and skeleton-based action recognition are the most common methods. However, action recognition using videos [5], [6] is a challenging task, usually due to the high dimension and complexity of the RGB video data, presence of cluttered backgrounds, variations in illumination and so on. In recent years, Skeleton-based action recognition [1],[7],[8],[9],[2],[10],[11],[12] has become one of the mainstream methods due to its adaptability to dynamic circumstances and robustness against complicated backgrounds. Skeleton data is lightweight and contains only the 2D or 3D coordinates of the key joints involved in the action, thus providing highly abstract and compact information that is free from environmental noises (lighting, background clutter, clothing), allowing the model to focus on the robust features of the action.

1.1 Background

Early deep learning-based models [13],[14],[15],[16],[17],[18],[19] treated human joints as a set of independent features and organized them manually into feature vector representations, or pseudo-images, which were then trained using RNNs or CNNs to predict action recognition scores. However, restructuring the skeleton graph into a grid results in loss of inherent structural features and correlations between human joints which reveal information about the human body topology. The skeleton is naturally structured as a non-Euclidean graph, with the joints as vertices, and the natural connections between them (bones) as edges. Recently, Graph Convolutional Networks (GCN) which generalize convolution from images to graphs, have achieved remarkable results by exploring the graph nature of the skeleton data for both spatial and temporal domains. For skeleton-based action recognition tasks, Yan *et al.* [4] first proposed Spatial Temporal Graph Convolutional Network (ST-GCN) to model skeleton data by building a spatial-temporal graph based on the natural connectivity of joints in the human body, and adding temporal edge between the same joints in consecutive action frames. However, they only take into account the neighboring nodes' information during feature aggregation and ignore the distant relationships between non-neighboring nodes. For various actions, distant joints are often related and they might contain useful information for predicting the action. For example, while clapping, the relationship between the joints of the two hands is important, although they are distant. Li *et al.* [20] addressed this problem by proposing a Spatio-Temporal Graph Routing (STGR) model, which learns relationships between distant joints in an adaptive manner. STGR learns the connectivity relationship among distant joints by using subgroup clusters. Peng *et al.* [21] proposed a GCN model based on Neural Architecture Search (GCN-NAS), which replaces the fixed graph with a dynamic graph and increases the receptive field size for every node, using higher order approximation of Chebyshev polynomials. GCN-NAS uses automatic neural searching to reduce manual efforts in designing the optimal network structure. Shi *et al.* [22] replaced the fixed graph with an adaptive graph and proposed a two-stream adaptive GCN (2s-AGCN),

which uses two different types of graph to learn the correlations between the different keypoints in the human body. One is a global graph, which is an adjacency matrix representing the graph structure of the human body, and another is individual graph, which is specific to each convolutional layer and is learnt from the supplied data. The authors also introduced a two-stream strategy in which they exploit the bone information along with the joint information. The same authors improved the model in [7] by introducing spatial, temporal and channel attention modules and using a multi-stream architecture which exploits the joint, bone and their motion information as well. Directed Graph Neural Network (DGNN) [23] uses a Directed Acyclic Graph (DAG) as input instead of an undirected one, which is better at modeling dependencies between joints and bones.

1.2 Motivation

ST-GCN [4] is one of the earliest works on modeling skeleton data using GCN. Several works [7], [8], [9], [10], [12], [20], [22], [23], [24] have followed the design and backbone of ST-GCN, which have developed better modules, increasing recognition accuracy. However, these models have very high computational complexities, which increase further on adding incremental modules and multi-stream fusion strategy. Another drawback is the limited receptive field of convolution operation in most of these architectures. As discussed in Section 1.1, state-of-the-art GCN models either use adaptive modules or dynamic graph structures to enhance the receptive field of each node. However, this further adds to the training time and the computational complexity of the model. For example, DGNN [23] costs more than 100 GFLOPs (Giga Floating Point Operations) for one action sample. Hence, current focus has shifted towards building more lightweight architectures which require fewer parameters. One such architecture is Shift-GCN [1], which replaces the regular 2D convolution operator with a shift convolution operator [3]. This operation has two advantages: firstly, shift convolution is lightweight, and has much less computational complexity as compared to a regular 2D convolution. Secondly, shift convolution makes the receptive field

of each node cover the entire skeleton graph, instead of just the neighboring nodes. Hence, the receptive field of Shift-GCN is more flexible than other state-of-the-art GCN models. However, Shift-GCN does not take into consideration the connection information of the input skeleton due to which it is less effective in distinguishing structurally similar actions having subtle differences. The adjacency matrix captures the structural information of the human body and the connection information of the joints, which allows us to model the spatial and temporal relations between the joints. This information is important for recognizing actions based on how the joints move and interact with each other. Shift-GCN does not utilize the graph connectivity knowledge due to which it does not give optimal performance.

In this thesis, we propose a feature-enhanced shift graph convolutional network which uses both graph connection information and shift modules for an Enhanced Feature Representation and lightweight feature extraction (FES-GCN). We follow the work in the baseline Shift-GCN [1] to increase the flexibility of both spatial and temporal receptive fields. However, we utilize the connection information between skeletal joints and perform shift convolution on a more abundant and enhanced feature map having a larger number of channels. Following ST-GCN [4], we employ a spatial configuration partitioning of the input graph into three subsets. For each subset, we construct its respective adjacency matrix and combine it with the input feature vector. We perform a pointwise convolution on all three subsets and concatenate them into a single intermediate feature vector with more information, which is then passed into the shift modules. The enhanced expressiveness of the input graph, followed by the lightweightedness and flexibility of the shift module result in an architecture which shows a competitive performance on being compared with state-of-the-art methods, while having a lower computational complexity than most existing approaches.

1.3 Objectives

In this thesis, we aim to achieve the following objectives:

(i) To develop a Shift-based GCN with an Enhanced Feature Representation for

skeleton-based human action recognition, that combines the skeletal connectivity information along with the input feature vector to generate a more detailed and enhanced feature map.

(ii) To perform spatial and temporal shift-based convolutions on the enhanced feature map to ensure a lightweight feature extraction and a more flexible spatial-temporal receptive field.

(iii) To explore the effectiveness of the proposed model by performing exhaustive ablation studies on each module and comparing its performance with existing state-of-the-art methods.

1.4 Thesis Contributions

A brief overview of our research contributions is provided below, and more details are available in the later chapters.

Contribution I:

We propose a Feature-enhanced shift graph convolutional network (FES-GCN) and apply it on classifying various skeleton-based human actions. Our model captures fine-grained graph connection information of the skeleton feature, yielding better recognition accuracy, while being computationally cheaper than most state-of-the-art models. Our model is based on the work in Shift-GCN [1]. However, unlike Shift-GCN, we combine the graph structure with the coordinate information to build a more abundant and enhanced feature map, before performing shift operation.

Contribution II: Although Shift-GCN provides a more flexible receptive field, it discards the graph knowledge and only takes into account the coordinate information of the skeleton. Hence we combine both coordinate information and connectivity information of the joints for an enhanced feature representation. The adjacency matrix of the skeletal graph provides information on how the graph vertices are connected with each other. To construct the skeletal graph, we follow ST-GCN [4] and perform a spatial configuration partitioning on the input graph. We divide the neighborhood

of each vertex into three subsets, following which we combine the adjacency matrix of each subset with the input feature vector to build a more enhanced feature map, before it undergoes shift convolution. However, we proceed with a fixed graph structure instead of a dynamic graph due to two reasons: (a) introducing adaptive modules to generate dynamic graphs with respect to each layer result in increased complexity of the model, hence we utilize a simple framework for combining the connectivity information with the feature vector. (b) To enable a global feature sharing across all nodes, spatial-temporal shift modules are a better option as they enable non-local and adaptive feature shifting across channels. However, combining the coordinate and connection information further improves the performance of shift operation, due to enhanced feature representation.

Contribution III:

We have verified the effectiveness of our proposed model by training and testing it on three widely used skeleton datasets, NTU-RGB+D 60 [25], Northwestern-UCLA [26] and Kinetics Skeleton 400 [27]. We have performed exhaustive ablation experiments to verify the necessity and effectiveness of each module. We have also compared the performance of our proposed model with various state-of-the-art models to establish the competitive performance of our proposed model.

1.5 Organization of the Thesis

This thesis is organized into six chapters. A summary of each chapter is provided below:

Chapter 1 (Introduction)

This chapter describes the background knowledge of clustering, the motivation of our work, and the contribution of this thesis.

Chapter 2 (Literature Survey)

This chapter discusses the literature on GCNs, Skeleton-based action recognition and shift convolutions. Here, we also discuss the various graph partitioning strategies, performance measures used in our thesis and finally the datasets used and data pre-

processing.

Chapter 3 (A Feature-Enhanced, Shift-based GCN for Skeleton-based Activity Recognition)

This chapter discusses both the spatial and the temporal units of our proposed model with illustrative diagrams.

Chapter 4 (Block and Network architecture of FES-GCN)

This chapter discusses the overall architecture of one Graph Convolutional Block, the network architecture of one stream, and finally the overall architecture of the full model.

Chapter 5 (Experiments and Results)

In this chapter, we discuss the experiments and ablation studies performed on our model using three datasets. We also compare the performance of our model with other state-of-the-art models to establish the efficiency of our model.

Chapter 6 (Conclusion and Future work)

In this chapter, we conclude the work in our thesis and discuss the future directions of our research.

Chapter 2

Literature Survey

This chapter discusses the various foundational concepts required to know in order to proceed to the architectural details of our proposed model, FES-GCN. Section 2.1 discusses the basics of Graph Convolutional Networks (GCNs). Section 2.2 explores the application of GCNs in Action recognition using Skeleton data. Further, we discuss briefly how Shift Convolutions work in Section 2.3. Section 2.4 discusses the various skeleton graph partitioning strategies. Section 2.5 discusses the various performance measures we have used in our research. Finally, Section 2.6 describes the datasets used in our experiments and also the data preprocessing done for our model.

2.1 Graph Convolutional Networks (GCN)

Deep Learning frameworks such as CNNs and RNNs have proven to be highly effective in handling regularly-structured or Euclidean data, such as images and videos, as shown in Fig. 2.1 A. However, most of the real-world data have irregular or non-Euclidean graph structure as shown in Fig. 2.1 B, such as social networks, protein interaction networks, citation networks, World Wide Web, and so on. For such types of data, it is difficult to generate accurate results using CNN or RNN-based models. The issue of handling non-uniform data has led to the recent advancements in Graph Neural Networks (GNN) [28]. GNNs are categorized into three main types: (a) Recurrent GNN (RecGNN), (b) Convolutional Graph Neural Network (ConvGNN) and (c)

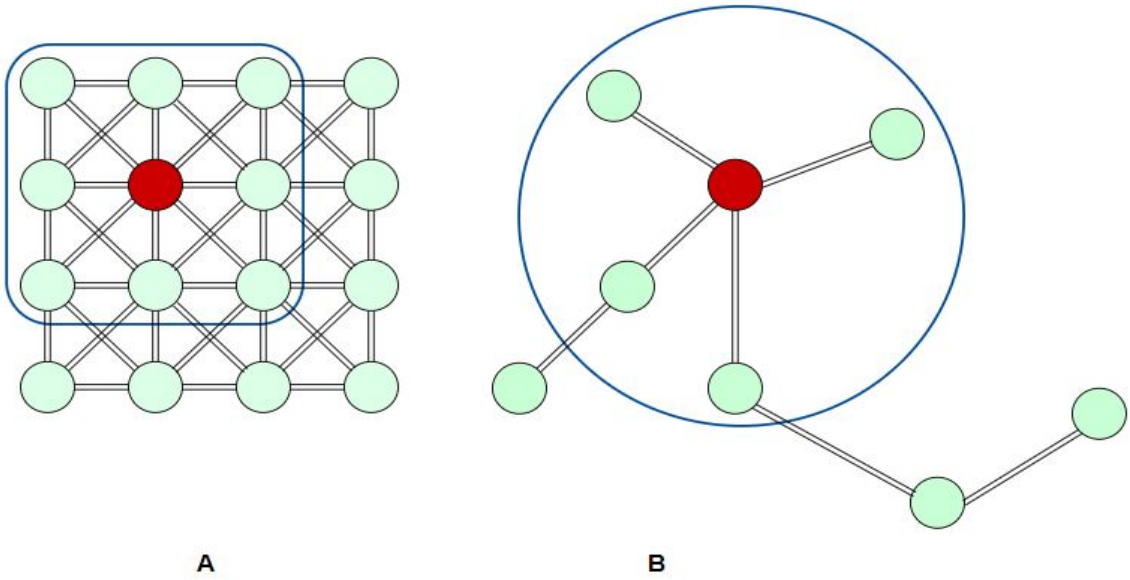

Figure 2.1: A - Euclidean or Regular structured data as found in images and videos. B - Non-Euclidean structured data as found in graphs.
Figure ref. [2].

Graph Autoencoders. The framework which we use for predicting action labels is a Convolutional Graph Neural Network, more popularly known as Graph Convolutional Network (GCN). Graph Convolutional Networks (GCNs) are a class of neural networks that can handle graph-structured data, providing a robust modeling framework for non-Euclidean data and find use in many areas such as social networks, citation networks, recommendation systems, traffic predictions, molecular analysis, and so on. GCNs can effectively capture the nodal information and dependencies between nodes in a graph. GCNs are of two types - Spectral and Spatial. Spectral methods [29],[30],[31] use eigen values of the graph Laplacian matrix to construct filters. These methods use graph Fourier transformation to perform convolution in the frequency domain, which does not require extracting locally connected regions at each convolution step. However, the process of Eigen decomposition is computationally expensive for large graphs. The other method is the Spatial method [2], [4], where convolutional filters or kernels are used directly to process the nodes and their neighboring nodes. The spatial GCN follows an approach similar to traditional CNNs, but it enables the generalization of convolution from image to graph data which is irregularly shaped.

The details of Spectral and Spatial GCNs are provided in the following subsections.

2.1.1 Spectral GCN

Spectral-based GCNs use spectral filters or kernels based on the Laplacian of the graphs to construct network models. Spectral GCN performs convolution in the frequency domain, using a graph Fourier Transformation. This does not require extracting information from locally connected regions at each step. To understand spectral graph convolutions, let us consider a simple, connected and undirected graph $G(V, E)$ having n vertices and m edges. Let A be the adjacency matrix denoting the connection information of the nodes in the graph, D be the diagonal degree matrix where d_i denote the degree of node i of graph G. Finally, let $X \in \mathbb{R}^{n \times d}$ be the input feature vector. The normalized graph Laplacian matrix L is defined as: $L = I_n - D^{-1/2}AD^{-1/2} = U\Lambda U^T$, where Λ denotes the diagonal matrix of eigenvalues and U is the Fourier transform matrix. We compute spectral graph convolution over an input signal $x \in \mathbb{R}^n$ using a filter f_θ, where $\theta \in \mathbb{R}^n$ denotes filter parameters. Then the spectral graph convolutions can be formulated as:

$$Y = f_\theta(L) * x = U f_\theta(\Lambda) U^T * x \tag{2.1}$$

However,the decomposition of graph Laplacian into its eigenvalues is a computationally costly process, requiring $O(n^2)$ time. To avoid eigen decomposition, *Defferrard et al.* [31] follow a K^{th} order approximation of $f_\theta(\Lambda)$ in terms of Chebyshev polynomials $T_r(\Lambda)$ as shown in equation 2.2:

$$f_\theta(\Lambda) \approx \sum_{r=0}^{K-1} \theta_r T_r(\widetilde{\Lambda}) \tag{2.2}$$

This reduces the computational complexity from $O(n^2)$ to $O(K|E|)$. Mainstream GCN models use first order approximation of Chebyshev polynomials to reduce computation time, thus, not involving higher order connections and limiting the feature representation capability of the model.

2.1.2 Spatial-Temporal GCN

A Spatial-temporal graph convolutional network [1],[7],[8],[22], [12],[10],[4],[23],[24],[32] is designed to handle data that has both spatial and temporal components, such as video or motion capture data. The main aim is to learn hidden patterns and extract features from spatial-temporal graphs, which is crucial in a range of applications, from predicting traffic speeds to anticipating driver maneuvers, recognizing human actions, and many more. By integrating spatial and temporal information, spatial-temporal GCNs can capture both types of dependencies simultaneously in an action. We design a spatial-temporal GCN model to perform accurate classification of human actions.

Similar to conventional CNNs, spatial GCNs aggregate neighborhood information during convolution process for each node. Spectral graph convolutions take a lot of time while operating on larger graphs, which can be reduced by using spatial graph convolutions. Spatial GCNs follow a sequence of feature aggregation, followed by a non-linear activation such as ReLU or Tanh. Another advantage of Spatial GCNs is that they facilitate weight sharing across all layers. Consider an input feature vector X and adjacency matrix A, the output feature at each layer Y is computed as:

$$Y = \sigma(W\widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}X) \tag{2.3}$$

Here, σ is an activation function, W is the weight matrix, $\widetilde{A} = A + I_n$ is adjacency matrix added with self loops and $\widetilde{D}$ is the diagonal degree matrix of $\widetilde{A}$.

For extracting temporal features, most GCN models apply a standard $K_t \times 1$ convolution on the intermediate feature map, since every action time frame has 2 neighbors. Here, K_t refers to the kernel size of temporal convolution, typically set to 9.

2.2 Application of GCNs in Skeleton-based Activity Recognition

Activity recognition using skeleton data differs from RGB video-based recognition as it prioritizes body movements over frame sequences. Also, the method is lightweight and is free from environmental noises (e.g. background clutter, lighting conditions, clothing). Extensive research has been done on Skeleton-based activity recognition in the past few years. Primitive methods used manually-generated features to model the human body, such as Lie group [33] and Rank Pooling [5]. However, these methods can be computationally expensive and can require specialized mathematical knowledge to implement and interpret. Additionally, they are not suitable for all types of action recognition tasks and may require modifications or extensions for different types of data or motion.

With the advent of deep learning models, CNNs and RNNs are widely used as the mainstream methods for various applications. RNN-based methods [13],[14],[15] model the skeleton data into vector representations, CNN-based methods [16], [17], [18], [19] employ hand-crafted transformation rules which represent the skeleton data in the form of pseudo-images. Such methods tend to rearrange the skeleton data from a non-Euclidean into a grid-like structure, due to which it is difficult to preserve the topology structure. This results in a loss of information between joints. Therefore, the current focus has shifted to GCNs since they can operate on non-euclidean data such as graphs. Skeleton data is constructed into graphs and GCNs are used to extract features from connected vertices.

Work in [4] is one of the earliest known works on GCN which directly models the skeletal data into a graph, thus eliminating the requirement of designing handcrafted features and achieving better performance than the previous methods. Shi et al. [22] use a two-stream adaptive model which utilizes second order bone information besides joint information. The same authors improve the model in [7] by introducing attention modules along with the adaptive layer and using four streams namely: joints, bones and their respective motion streams. Another approach is Directed Graph Neural

Network (DGNN) [23], which uses a Directed Acyclic Graph (DAG) to represent the skeletal information, portraying the inherent kinematic dependencies between the joints and their connections in the natural structure of the human body.

However, most of the state-of-the-art GCN models are computationally expensive and memory-intensive. Shift-GCN [1] replaces the computationally expensive 2D convolution operation with lightweight feature shift operations followed by lightweight pointwise convolutions in both spatial and temporal domains, which gives a competitive result against state-of-the-art GCNs, requiring 10 times less computational complexity. However, it discards the graph knowledge and connection information of the input skeleton sequence due to which it does not give optimal performance on fine-grained actions. Hence, we propose to integrate coordinate information and graph connectivity information for a better performance in classifying fine-grained actions.

2.3 ShiftNet

In CNNs, the convolution operation is used to extract features from input images or videos by applying a set of learnable filters over the input pixels. However, convolutional operations can be computationally expensive and memory-intensive and with increase of network depth, the amount of calculation and parameters of the convolution become difficult to control. ShiftNet [3, 34, 35] is a computationally efficient substitute for standard 2D convolutional operations within CNNs. ShiftNet is a deep neural network architecture that uses shift operations instead of standard convolution operations. A shift operation involves adjusting the channels of the input feature vector in different directions. This operation is directly followed by a pointwise convolution which exchanges information across various channels. Such an implementation enables aggregation of neighborhood information in a computationally cheaper way [1]. Wu *et al.* [3] proposed the use of a combination of shift and pointwise convolution instead of a 2D spatial convolution. They replaced the 3x3 convolution of ResNet with shift modules and obtained improved accuracy on CIFAR-10 and CIFAR-100 with 60% fewer parameters.

Consider a spatial convolution whose input is a tensor F of size $F \in R^{D_F \times D_F \times M}$ as shown in Fig. 2.2 **A**. Let D_F be the width and height of the tensor and M be the input channel size. The kernel of a spatial convolution is a tensor K of dimensions $K \in R^{D_K \times D_K \times M \times N}$, where N is the number of filters and D_K is the kernel size. For simplicity, we assume the stride to be 1 and identical spatial dimensions for input and output. For a regular spatial convolution, the number of parameters required is $M \times N \times D_K^2$ and the computational cost in FLOPS (Floating Point Operations Per Second) is $M \times N \times D_K^2 \times D_F^2$. Larger the kernel size D_k, higher the number of parameters and computational cost, as they grow quadratically.

A shift convolution consists of two operations: (a) Adjusting feature channels in different directions, (b) Pointwise convolution operation to enable communication between various feature maps. The computation cost of a shift convolution is $D_F^2 \times M \times N$ FLOPS, much lower than that of a regular convolution operation, since $D_K = 1$. Shift convolution also has a highly flexible receptive field which it can enlarge by simply increasing the shift distance, rather than relying on larger kernels and complex models.

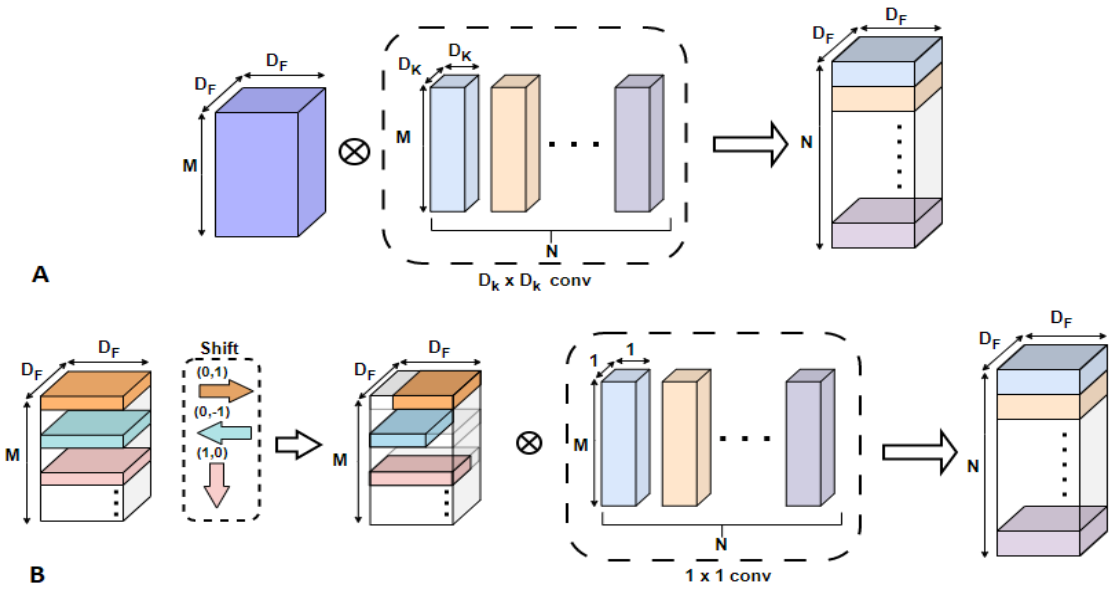

Figure 2.2: A - Regular convolution operation, B - Shift operation followed by point-wise convolution. Here, D_F is height and width of input tensor, M is input channel size, D_K is kernel size, N is output channel size. Figure ref. [3]

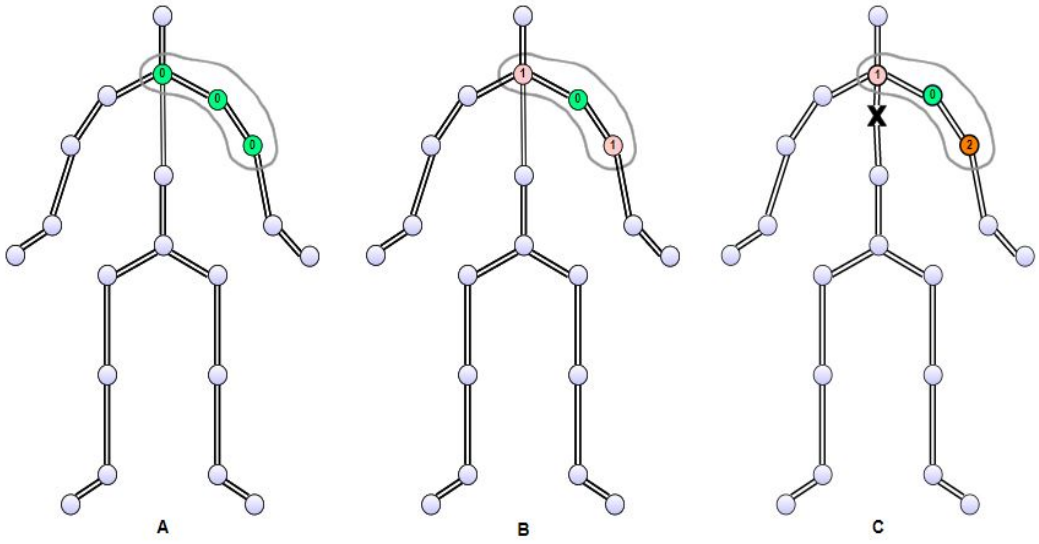

Figure 2.3: Partitioning strategies [4]: A - Uni-labeling, B - Distance Partitioning, C - Spatial Configuration Partitioning. "X" denotes the center of gravity and 0,1,2 denote the subset number.

2.4 Skeleton Graph Partitioning Strategies

Unlike images or videos, convolution operation on graphs is a more challenging task, since the nodes do not have a fixed neighborhood. In case of images and videos, it is easier to define the weight function as the neighborhood of each pixel has a fixed spatial order. In case of graphs, there is no such implicit order. Hence, it becomes necessary to partition the neighboring vertices into a fixed number of subsets K. We require a mapping function $l(v_i)$, which maps the neighbors of a vertex into one of the subsets in the range 0,1,..., $K-1$. The commonly used partition strategies are:

1. **Uni-labeling**: This is the simplest partitioning strategy which creates one subset constituting the whole neighborhood of the vertex, including the vertex v_i itself. For uni-labeling, we have $K = 1$ and $l(v_j) = 0$, $i, j \in |V|$. The drawback of this strategy is that it requires computing the inner product of the weight matrix and the average feature vector of the whole neighborhood, which results in losing of local distinct features.

2. **Distance partitioning**: Another partitioning technique is to divide the neigh-

borhood according to distance $d(v_i, v_j)$ from root vertex v_i. This divides the entire first order neighborhood of a vertex v_i into 2 subsets, hence $K = 2$ and $l(v_j) = 0, 1$, $i, j \in |V|$. For the root vertex itself, $d(v_i, v_j) = 0$, hence it gets partitioned into subset 0. For the other neighboring vertices, $d(v_i, v_j) = 1$, hence they fall into subset 1.

3. **Spatial configuration partitioning**: Yan *et. al* [4] adopt a strategy to divide the neighborhood of a vertex v_i into 3 subsets: (a) The root vertex itself (b) Centripetal subset, consisting of neighboring nodes closer to the center of gravity of skeleton than the root, (c) Centrifugal subset, consisting of neighboring nodes farther from the center of gravity of skeleton than the root. Here, $K = 3$ and $l(v_j) = 0, 1, 2$, $i, j \in |V|$. Following [4], we adopt a spatial configuration partitioning technique to partition the neighborhood of every vertex into 3 subsets.

2.5 Performance measures used

Top-1 Accuracy: Top-1 accuracy is the conventional accuracy used as a performance metric. It measures the percentage of correctly classified samples with respect to the total number of samples used for generating predictions. Top-1 accuracy will consider a prediction correct if and only if the class with the highest prediction probability matches with the truth label.

Example:

True label	Predicted label	Result
Cat	Cat	✓
Dog	Giraffe	✗
Dog	Dog	✓
Monkey	Horse	✗
Horse	Horse	✓

In the above example, the model correctly predicts 3 out of 5 inputs. Hence, the top-1 accuracy is 60%.

Top-5 accuracy: Top-N accuracy considers the top N predicted labels having the highest probabilities. If one of the top N predictions matches with the true label, the prediction is classified as correct. For evaluating the Kinetics-Skeleton [27] dataset, we report both the top-1 and top-5 accuracies.
Example:

True label	Top 5 Predicted labels	Result
Cat	Cat, Dog, Monkey, Horse, Swan	✓
Dog	Giraffe, Dog, Cat, Lion, Horse	✓
Dog	Cat, Giraffe, Swan, Lion, Horse	✗
Monkey	Horse, Monkey, Swan, Cat, Dog	✓
Horse	Cat, Swan, Dog, Giraffe, Horse	✓

In the above example, there are 4 out of 5 inputs for which the true label is present in the top 5 predictions. Hence, the top-5 accuracy is 80%.

FLOPs: FLOPs stands for Floating Point Operations. In deep learning, FLOPs refer to the number of floating point operations required for one sample input in a single forward pass. Higher the FLOPs, slower the model, and hence longer the training time. FLOPs describe how much calculation is required to pass data through the model network.

To compute the number of FLOPs, we assume sliding window convolution and that the non-linearity function requires no computation cost. For a kernel or filter, we have,

$$FLOPs = 2WH(K^2C_{in} + 1)C_{out} \tag{2.4}$$

Where W and H are height and width of input feature map, C_{in} is the number of input channels, C_{out} is the number of output channels and K is the kernel size. For

fully connected layers, FLOPs are computed as:

$$FLOPs = (2I - 1)O \tag{2.5}$$

Where I = number of input neurons, O = number of output neurons.

2.6 Datasets and Data Preprocessing

Subsection 2.6.1 discusses the three widely used datasets used in this thesis throughout our experimentation, and Subsection 2.6.2 discusses the data preprocessing required to implement the multi-stream architecture.

2.6.1 Datasets Used

We have trained and evaluated our model on three benchmark datasets, which are described as follows.

1. **NTU RGB+D 60** [25] is a large-scale dataset of indoor actions that is widely used in the recognition of human actions. It contains 56,880 action sequences which are performed by 40 actors and are classified into 60 categories.Three cameras are used to capture each action at constant height by varying the horizontal angles to -45°, 0° and 45°. Kinect depth sensors are used to locate the 3D joint locations in each action frame. Each video contains not more than 2 subjects, and there are 25 joints for each subject in the action sequence. The joints are denoted by three-dimensional coordinates. The authors of this dataset recommend two benchmarks: 1) Cross-Subject (X-sub), where the dataset is divided into a training set of 40,320 videos and a validation set of 16,560 videos. Here, the training data comes from 20 subjects and validation data comes from the other 20 subjects. The other benchmark is 2) Cross-View (X-view), where the training data contains 37,920 videos captured by cameras 2 and 3, and the validation data contains 18,960 videos captured by camera 1. Following the convention in [25], we report the top-1 accuracy on these two benchmarks.

2. **Northwestern-UCLA** [26] is a dataset which is captured by 3 Kinect cameras simultaneously from multiple viewpoints. It contains 1494 videos categorised into 10 action classes. We follow the convention in [26] and use the videos of the first and second cameras as training data and the third camera as testing data.

3. **Kinetics-400 Skeleton** is a large-scale dataset obtained from the Kinetics-400 [27] dataset, which contains human action video clips taken from YouTube videos. Authors in [4] estimate the 2D coordinates of 18 joints and their prediction confidence using the OpenPose toolbox [36] which is publicly available. They have released this dataset named Kinetics Skeleton, which we use for evaluating our model. Kinetics Skeleton contains 240,436 skeleton sequences for training and 19,796 skeleton sequences for testing over 400 action classes. We follow the evaluation method in [4] while training the model, and report the top 1 and top 5 accuracies obtained on the validation set.

2.6.2 Data Preprocessing

Earlier approaches [4] used only the 2D or 3D joint coordinates as the feature vector. This is known as the first order skeletal information, which lacks the necessary level of detail to accurately capture the dynamic changes in human skeletal structure during certain actions. Hence, we employ a four-stream architecture in accordance with MS-AAGCN [7], where the four data modalities are: joints, bones and their respective motion streams.

In any action, besides the joints, the bones also play a significant role, and thus represent second-order information in skeleton data. The bone data consists of length and direction of bones which typically provide more information regarding the action and enables distinguishing similar actions. For example, in actions like standing up and sitting down, the coordinates of knee joint do not change significantly, while there is a more visible and obvious movement between the connected bones. Hence, adding bone information enables the model to recognize complex action patterns more efficiently. We construct the bone vectors in accordance with the natural connections

between the human body joints. Each bone vector starts from a source joint and ends at a target joint. Given a source vertex p with coordinates (x_p, y_p, z_p) and its neighboring target vertex q with coordinates (x_q, y_q, z_q), the bone vector is computed as an edge from source joint to target joint, $e_{p,q} = (x_q - x_p, y_q - y_p, z_q - z_p)$.

Additionally, there are actions which are similarly performed such as sitting down and standing up, for which we need to model the temporal evolution of consecutive frames. Hence, adding motion information of joints and bones to the feature vector enables the model to distinguish between actions that appear spatially similar. For example, consider a joint vertex $p_t = (x_{p,t}, y_{p,t}, z_{p,t})$ in time frame t and the same joint $p_{t+1} = (x_{p,t+1}, y_{p,t+1}, z_{p,t+1})$ in time frame $t + 1$, the movement between the two joints is denoted as $m_{p,t,t+1} = (x_{p,t+1} - x_{p,t}, y_{p,t+1} - y_{p,t}, z_{p,t+1} - z_{p,t})$. The model takes in four data modalities as inputs, and each modality is processed in a separate stream. In order to ensure a fair comparison, we adopt a strategy of weighted summation as in [7] to fuse the softmax scores of all four modalities, to generate the final prediction score and infer the action label.

Chapter 3

A Feature-Enhanced Shift Graph Convolutional Network for Skeleton-based Action Recognition

In this thesis, we propose a feature-enhanced shift graph convolutional network (FES-GCN) for skeleton-based human action recognition, which performs spatial and temporal shift graph operations on a feature map consisting of both the coordinate information and joint connectivity information. Our model is based on the architecture of Shift GCN [1] for a lightweight feature extraction and increased flexibility of spatial-temporal receptive fields. Although Shift-GCN is lightweight, it does not take into account the graph connectivity information, due to which it gives suboptimal performance on fine-grained actions having subtle details. By combining the coordinate and connectivity information, we construct a more detailed and enhanced feature vector, which is then passed into spatial and temporal shift modules for feature extraction. Fig. 3.1 is an outline showing the modules in a block in FES-GCN. The detailed architecture of each module is discussed in Sections 3.1 and 3.2.

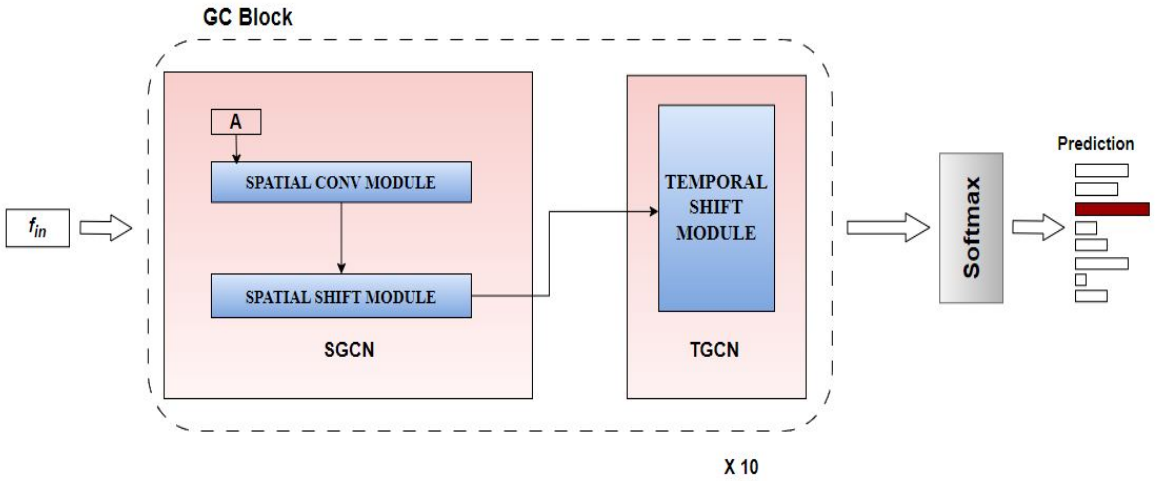

Figure 3.1: An outline of our proposed model FES-GCN. The spatial and temporal GCN units, denoted by SGCN and TGCN are present serially in every block to study the spatio-temporal characteristics of the action sample. f_{in} is the input feature map and A is the graph adjacency matrix.

3.1 Graph Construction and Spatial Graph Convolutional Unit

We propose to combine both the coordinate information and the graph connectivity information before performing shift operation, as both types of information are required for a more detailed representation of the body features while performing an action. We construct a spatial-temporal graph which informs how the vertices of the graph are connected with each other. ST-GCN [4] models a graph that is equipped with information extending along spatial-temporal dimensions. Fig. 3.2 A diagrammatically represents the structure of a skeleton graph representing the joints and their natural connections. The corresponding joints across two action frames are linked by temporal edges, forming a temporal graph. A vanilla spatial convolution operation with respect to a vertex v_i is expressed as follows:

$$f_{spatial}(v_i) = \sum_{v_j \in B_i} \frac{1}{Z_{ij}} f_{in}(v_j).w(l_i(v_j)) \tag{3.1}$$

Here f_{in} and $f_{spatial}$ are the input and output feature maps respectively and v_j denotes

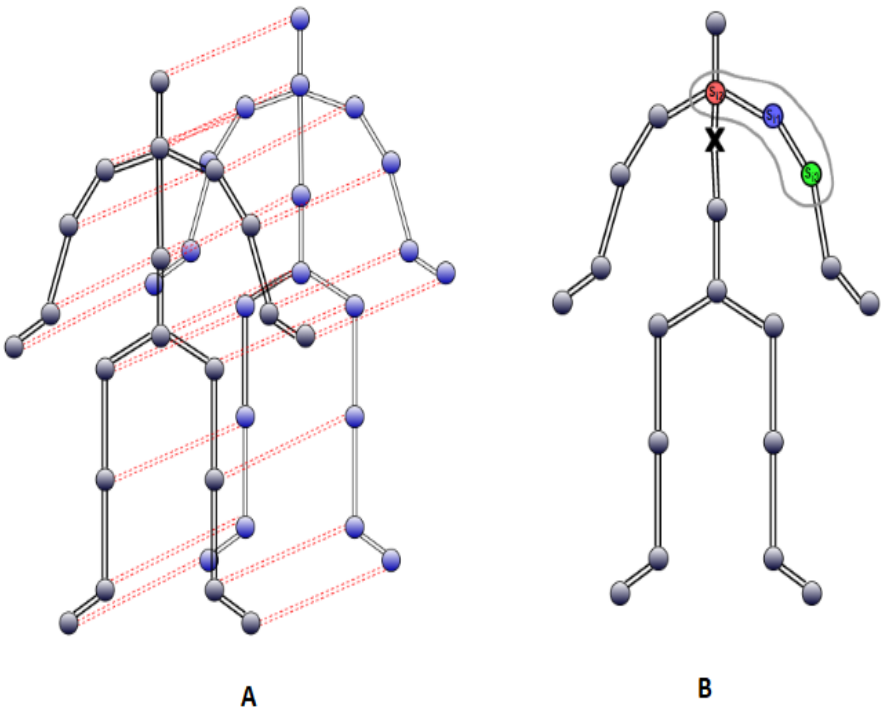

Figure 3.2: A - Spatial-temporal skeleton graph structure showing two frames of action. Black lines denote the connections between joints in one frame, red lines denote the temporal connections between corresponding joints across two consecutive frames. B - Spatial configuration partitioning with respect to the vertex marked in blue. X denotes the center of gravity, the vertex marked in red denotes the centripetal subset, the vertex marked in green denotes the centrifugal subset.

neighborhood of vertex v_i. B_i represents the sampling area of neighborhood for vertex v_i . We set the sampling area as the one-hop neighborhood of target vertex v_i. Weight function w provides a weight matrix for the given input. It is important to note that convolution operation on graph data is trickier than that on images, as every pixel in an input image contains a fixed number of neighbors but each node in a skeleton graph has variable number of neighbors. Hence, we are required to partition the neighborhood set of each vertex into a fixed number of subsets. We have discussed the commonly used partitioning strategies in Section 2.4. For our model, we adopt the spatial configuration partitioning as this strategy leads to a better modeling capability. Fig. 3.2 B illustrates the partitioning technique, where we partition the sampling area B_i into three subsets: S_{i1},S_{i2} and S_{i3}. S_{i1} contains the target vertex v_i itself, S_{i2} is the

centripetal subset containing vertices which are at a lesser distance from the center of gravity than the target vertex, S_{i3} is the centrifugal subset containing vertices at a larger distance from the center of gravity than the target vertex. l_i is a mapping function which maps each neighboring vertex v_j into its corresponding subset. Z_{ij} denotes the cardinality of S_{ik} that balances the contribution of each subset.

The Spatial Graph Convolutional unit has two parts: Spatial Convolution module and Spatial Shift module. The details of each module are discussed in the following subsections.

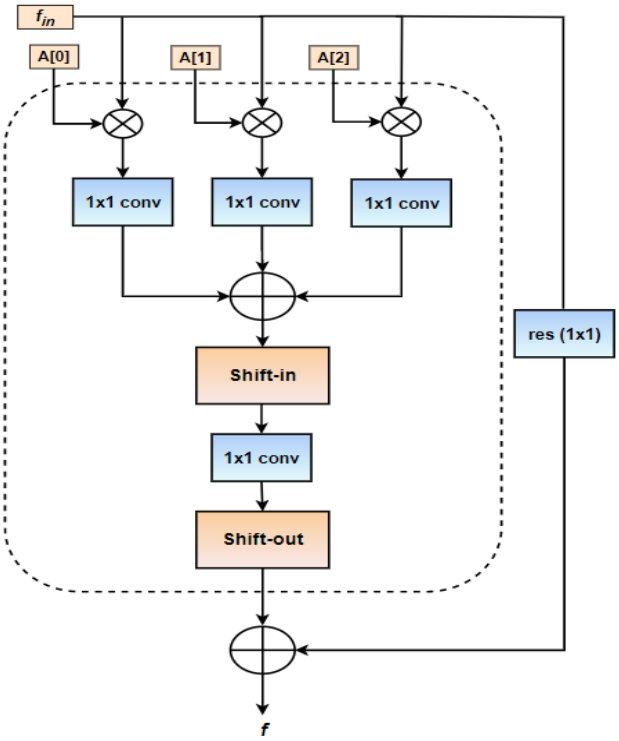

Figure 3.3: Spatial Graph Convolutional unit. f_{in} = input feature map, A = graph adjacency matrix where A[0], A[1] and A[2] correspond to each subset [4], f = intermediate feature map.

3.1.1 Spatial Convolution Module

Unlike Shift-GCN [1], we propose to utilize the graph connectivity knowledge for better feature representation and extraction. We partition the input graph into three subsets in accordance with ST-GCN [4]. For each partition, we mul-

tiply the corresponding adjacency matrix with the input feature map, which is a tensor $f_{in} \in \mathbb{R}^{C_{in} \times T \times N}$, where C_{in} , T and N are the input channel size, number of action frames and number of nodes in the graph respectively. Spatial convolution is implemented by transforming Equation 3.1 into Equation 3.2 as shown below:

$$f_{spatial}(v_i) = \sum_{k}^{K_v} W_k (f_{in} A_k) \tag{3.2}$$

Here, K_v is set as 3 denoting the number of subsets, W_k is the weight matrix associated with each pointwise convolution and A_k is the normalized adjacency matrix for each subset.

3.1.2 Spatial Shift Module

Spatial Shift operation shifts the features of neighboring nodes into the current node. Following the work in Shift-GCN [1] and employ an adaptive non-local shift operation on the intermediate feature map, which makes every node obtain the information of every other node. However, in FES-GCN, the shift module receives a more enhanced feature map, which results from a spatial convolution on a combination of coordinate information and connectivity information. The advantage of the spatial shift operation is, it expands every node's receptive field, thus enabling each node to gather information from all other nodes in the graph, thereby enhancing the model's ability to capture global dependencies. For an input feature $F \in R^{N \times C}$, the shifting distance of the i^{th} feature is calculated as $imodN$. This operation assumes equal importance of connections between every pair of nodes in the graph. However, the strength of the connections vary according to which joints are more involved for a given action. To address this issue, an adaptive learnable mask is incorporated into the shift operation, which allows the network to assign different weights to different pairs of nodes based on their importance in the action recognition task. This helps in better capturing of relevant information from the input skeleton graph. The mask is formulated as $tanh(M) + 1$ where M is a tensor initialized with zeros. We perform two shift operations, Shift-in, which involves upward shifting of feature channels,

and Shift-out, which involves downward shifting of feature channels. The two shift operations are separated by a pointwise convolutional operation.

3.1.3 Architecture of Spatial GCN (SGCN)

The entire architecture of the Spatial unit is illustrated in Fig. 3.3. It is composed of a Spatial Convolution unit, followed by a Shift-in operation, a pointwise convolution and a Shift-out unit. A residual connection is added for preserving gradients during backpropagation. The intermediate feature map is then passed into the Temporal Graph Convolutional Unit, which is described in Section 3.2.

3.2 Temporal Graph Convolutional Unit

The temporal GCN (TGCN) models the temporal evolution of the action and occurs after the SGCN. Earlier works [7],[12], [22], [4], [23] used a simple $K_t \times 1$ convolution over the temporal feature map, where K_t is generally set to 9. The reason of using a classical convolution operation is, the number of neighbors of a vertex in the temporal dimension is always fixed as 2. The operation however, has various drawbacks: (a) Since the receptive field is set manually, it may result in inadequate generalization of model while classifying actions of varying durations. (b) Different layers may require different size of receptive fields. Setting a rigid kernel size limits the models capability to model the temporal aspect of actions. (c) Finally, a large kernel results in a higher computation costs. Hence, we adopt the method of using adaptive temporal shift instead, in accordance with the baseline Shift-GCN [1], as this method adjusts the size of receptive fields adaptively, increasing the flexibility of the model. The details of adaptive temporal shift module are described below in subsection 3.2.1.

3.2.1 Temporal Shift Module

The temporal graph convolutional unit applies the same temporal adaptive shift convolution as described in the baseline [1]. Let the input feature be f and output

feature be f_{out}, where $f \in R^{N \times T \times C}$ and $f_{out} \in R^{N \times T \times C'}$. Each channel is temporally associated with an adaptive shift parameter $P_i, i = 1, 2, ..., C$. In this case, the constraint is relaxed from integers to real numbers The adaptive temporal shift is computed as shown below in Equation (3.3):

$$F'_{(v,i,t)} = (1 - \beta)F_{(v,i,floor(t+P_i))} + \beta F_{(v,i,floor(t+P_i)+1)} \tag{3.3}$$

Here, $\beta = P_i - floor(P_i)$. Since this operation is differentiable, it can be trained using backpropagation. Similar to spatial shift module, here too we perform a temporal Shift-in operation, a pointwise convolution and a temporal Shift-out operation.

Algorithm 3.1 shows the sequence of operations in FES-GCN. In the next chapter, we discuss in details the overall architecture of FES-GCN.

Algorithm 3.1 Algorithm of FES-GCN

Input: Input feature f_{in} of dimensions $C_{in} \times T \times N$, Graph Adjacency Matrix A of dimensions $N \times N$

Output: Output feature f_{out}

$c, t, n = f_{in}.size()$
$y = null$
for i in range K_v **do** ▷ Spatial conv
    $A1 = A[i]$
    $A2 = f_{in}.view(c \times t, n)$
    $z = pointwise\_conv((A2 \times A1).view(c, t, n))$
    $y = z + y$ ▷ Concatenation
end for
$y = ReLU(batch\_norm(y))$
$y = spatial\_shift\_in(y) * (tanh(Feature\_Mask) + 1)$ ▷ Spatial Shift
$y = pointwise\_conv(y)$
$y = spatial\_shift\_out(y)$
$y = ReLU(batch\_norm(y))$
$y = temp\_shift\_in(y)$ ▷ Temporal Shift
$y = ReLU(pointwise\_conv(y))$
$f_{out} = temp\_shift\_out(y)$
$f_{out} = batch\_norm(f_{out})$
return f_{out}

3.3 Summary

In chapter 3, we introduce our proposed model. FES-GCN. We at first discuss the outline of FES-GCN with a diagrammatical representation, followed by the construction of skeletal graph and the spatial graph convolutional unit. The spatial graph convolutional unit consists of the spatial convolution module and the spatial shift module. This is followed by the temporal modeling using the temporal shift module. The final output undergoes a batch processing before being passed into the next block. We finally explain the model workflow using Algorithm 3.1.

Chapter 4

Block and Network Architecture of FES-GCN

Chapter 3 presented an outline of our proposed model and discussed the architecture of the spatial and temporal units present in a block. In this chapter, we discuss in details the network architecture of FES-GCN consisting of a series of such blocks. Section 4.1 discusses the architecture of one block which contains the spatial and temporal modules. Section 4.2 discusses the network architecture of one stream. Section 4.3 finally discusses the architecture of the full model consisting of four streams.

4.1 Structure of one Graph Convolutional Block (GCB)

The structure of one Graph Convolutional Block (GCB) is shown in Fig. 4.1-A. Every GCB consists of a spatial unit and a temporal unit in sequence, denoted by SGCN and TGCN respectively. Both the spatial and temporal units are followed by batch normalization and ReLU activation layers respectively. Batch Normalization layer normalizes the input before it enters into the next layer. This helps to stabilize the network during training. For activation, we use ReLU as it does not activate all neurons at the same time. For positive inputs, the output is linear and for negative inputs, the output is zero. Since it activates only a certain number of neurons, ReLU

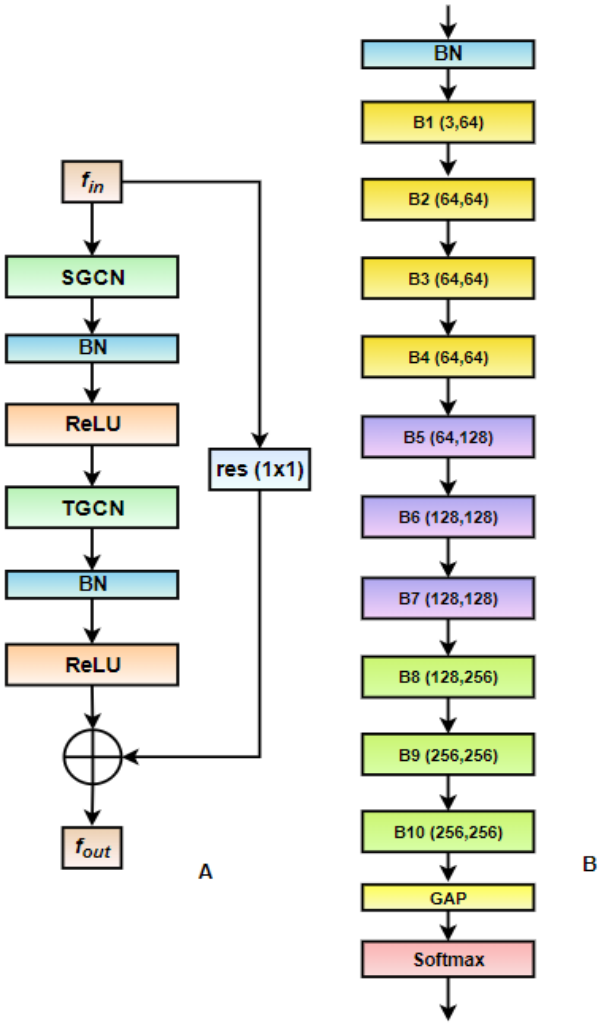

Figure 4.1: A - Basic architecture of one Graph Convolution Block [4] where the spatial and temporal units are denoted as SGCN and TGCN respectively, B - Network architecture of one stream consisting of 10 Blocks, B1 to B10. BN is Batch Normalization and GAP is Global Average Pooling.

is computationally more efficient as compared to other activations such as sigmoid and tanh. Deep neural networks are often prone to vanishing gradient problem [37], hence we have added a residual connection [38] in every block as it eases training in very deep neural networks by enabling the gradient propagation to all layers.

Table 4.1: Table illustrating the number of input channels, output channels and stride in each block

Block	Input channels	Output Channels	Stride
B1	3	64	1
B2	64	64	1
B3	64	64	1
B4	64	64	1
B5	64	128	2
B6	128	128	1
B7	128	128	1
B8	128	256	2
B9	256	256	1
B10	256	256	1

4.2 Network Architecture of one Stream

FES-GCN follows a four-stream structure as discussed in Section 2. Here, we discuss the network architecture of a single stream as illustrated in Fig. 4.1 B, which follows the backbone structure of ST-GCN [4]. It constitutes a linear arrangement of 10 basic Graph Convolutional Blocks, labelled B1 to B10. Table 4.1 displays the number of input channels, output channels and stride of each block. Temporal dimension is halved at blocks B5 and B8 using strided temporal convolution. After block B10, a global average pooling layer (GAP) is included to diminish the spatial dimensions of the feature. A softmax classifier is present at the end to predict the scores for each action label. All four streams follow the identical network architecture.

4.3 Full Architecture of FES-GCN

The overall architecture of FES-GCN is a multi-stream framework, consisting of four streams. MS-AAGCN [7] introduced a four-stream architecture which utilizes four data modalities: joints, bones and their respective motion data. FES-GCN follows a four-stream architecture as well and the preprocessing of input data into four modalities is discussed in Section 2.6.2. The full pipeline of the model is illustrated

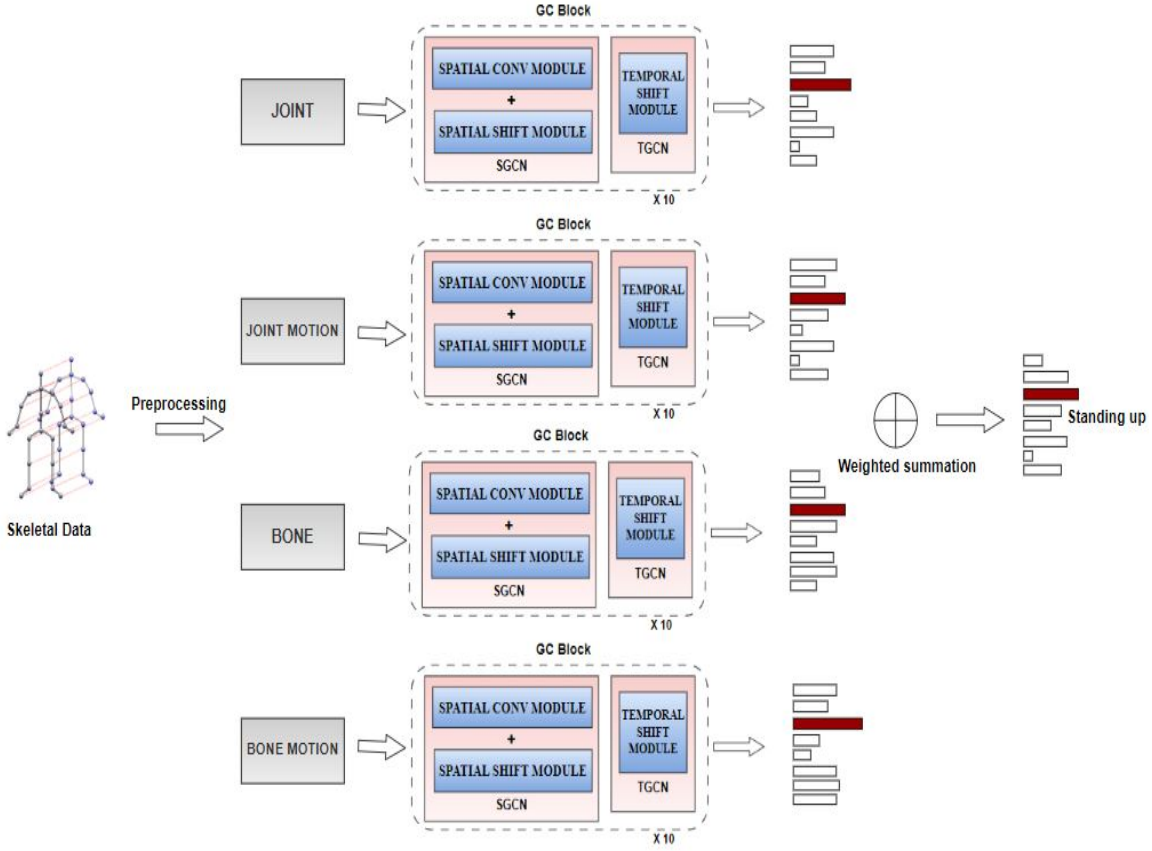

Figure 4.2: Outline of FES-GCN. The input data is processed into four types of data modalities namely, joint data, bone data and their respective motion data. The ultimate prediction is generated by combining each stream's action score by weighted summation

in Fig. 4.2. Each data modality is passed as input into the network, whose structure is identical for every stream. The ultimate prediction is generated by combining each stream's action score by weighted summation. The multi-stream fusion strategy has been adopted in various architectures [1], [7], [8], [10] hence, we too adopt the same strategy to ensure a fair comparison. The overall computational complexity of the model is the sum of GFLOPs required for each stream.

4.4 Summary

To summarize Chapter 4, we have discussed the block and network architecture of FES-GCN. One Graph Convolutional Block (GCB) consists of a spatial and a temporal unit in sequence, denoted by SGCN and TGCN respectively. This is followed by a batch normalization and ReLU layers. The network architecture of a single stream consists of 10 GCBs in sequence, with the number of output channels as follows: 64, 64, 64, 64, 128, 128, 128, 256, 256, 256. Finally, we have explained with a diagrammatic representation the full architecture of FES-GCN consisting of four parallel streams of the four data modalities. The final action score is a result of the weighted summation of each stream's individual action score. Chapter 5 dives into the experimentations and ablation studies conducted on FES-GCN and the comparitive results in details.

Chapter 5

Experiments and Results

This chapter presents the experimentations we have conducted to assess our model's performance, and the results observed. We have used three commonly used datasets to assess the performance of our proposed model and compare it with other methods. The datasets used are NTU RGB+D 60 [25], Northwestern UCLA [26] and Kinetics-400 Skeleton [27] which have been discussed extensively in Chapter 2 Section 2.6. We have further examined our proposed model's effectiveness through ablation studies conducted on each module.

5.1 Experimental Settings

In this section, we discuss the baseline models on which the architecture of FES-GCN is based. We also reveal the implementation details and requirements for carrying out the experiments.

5.1.1 Baseline Models

For the network architecture of the model, we have chosen ST-GCN [4] as the backbone, since the baseline and other state-of-the-art methods too follow the same network architecture. This ensures a fair comparison of our model's accuracy with other methods. The architecture of ST-GCN follows a sequence of 10 spatio-temporal graph convolutional blocks. Each block contains a regular spatial convolution opera-

tion and a regular 9×1 temporal convolution operation. However, ST-GCN follows a fixed graph structure with fixed receptive fields. In order to enhance the flexibility of the model, we have used Shift-GCN [1] as the baseline. Since the performance of Shift-GCN is limited by complete discard of graph knowledge, we have added a spatial convolution module where we have combined the coordinate features and the connection information for a more detailed and enhanced feature map. This ensures a more accurate classification of fine-grained actions having subtle differences in body postures.

5.1.2 Implementation Details

We have used PyTorch framework to conduct all our experiments and three NVIDIA Tesla V100 SXM2 GPUs of capacity 32 GB each for training and evaluating our model. Number of epochs for training is 140. The learning rate is initially taken as 0.1 and gets divided by 10 during the 60^{th}, 80^{th} and 100^{th} epochs. The optimizer used is SGD (Stochastic Gradient Descent) and Nesterov momentum is taken as 0.9. For NTU RGB+D, we have used a batch of 64 action sequences, 16 for NW-UCLA and 256 for Kinetics-Skeleton.

5.2 Ablation Studies

We have assessed the effectiveness of each module in our proposed model through a set of ablation studies.

1. **Spatial Convolutional Module**: As stated in Section 3.1.1, we utilize the graph connectivity information along with the coordinate information by performing a spatial pointwise convolution on the product of feature vector with each subset, and finally concatenating the resultant tensors before performing shift operation. Combining the skeletal connectivity information along with the input feature enables the model to capture the intricate structural details of the skeleton and the connection information among joints, which results in an

enhanced feature representation and improvement in accuracy. We test the effectiveness of utilizing graph connection information on NTU-RGB+D 60 and Northwestern-UCLA, and report the results in Table 5.1. On both the datasets, we observe an improvement in accuracy when we combine the baseline with a spatial convolutional unit.

Table 5.1: Accuracies reported on NTU-60 and NW UCLA datasets for one stream input, with and without spatial convolution and graph connection information

Without spatial convolution (1-stream)		**With spatial convolution (1-stream)**	
Dataset	**Accuracy (%)**	**Dataset**	**Accuracy(%)**
NTU-60 X-Sub	87.8	NTU-60 X-Sub	88.1
NTU-60 X-View	95.1	NTU-60 X-View	95.3
NW-UCLA	92.5	NW-UCLA	93.2

2. **Shift Module**: Cheng *et al.* [1] have introduced the concept of feature shifting to learn the spatial relationship between distant nodes in a lightweight and flexible manner. We incorporate spatial and temporal Shift modules for the same reason. However, we perform the spatial shift operation on a larger number of features which are enhanced using spatial convolutional module. Table 5.2 shows a comparison between the accuracies reported with and without spatial shift unit, using NTU-RGB+D 60 and Northwestern-UCLA. Since the shift operation increases the adaptability of each node's receptive field, FES-GCN reports a higher accuracy on the benchmark datasets in presence of shift module as compared to the model with only spatial convolutional unit. In absence of spatial shift, the accuracy reduces to 86.2% and 93.8% on NTU-RGB+D 60 cross-subject and cross-view benchmarks respectively, compared to 88.1% and 95.3% on the same using spatial shift. Similarly, the accuracy reduces to 89.6% for Northwestern-UCLA dataset in absence of spatial shift module.

3. **Multi-stream architecture**: We have tested our proposed model on four

Table 5.2: Accuracies reported on NTU-60 and NW UCLA datasets for one stream input, with and without spatial shift unit

Without spatial shift (1-stream)		**With spatial shift (1-stream)**	
Dataset	**Accuracy (%)**	**Dataset**	**Accuracy (%)**
NTU-60 X-Sub	86.2	NTU-60 X-Sub	88.1
NTU-60 X-View	93.8	NTU-60 X-View	95.3
NW-UCLA	89.6	NW-UCLA	93.2

Table 5.3: Accuracy and GFLOPs obtained for 1-stream, 2-stream and 4-stream architectures respectively on the three benchmark datasets

Data Modality	**NTU 60 X-Sub**	**NTU 60 X-View**		**NW-UCLA**		**Kinetics Skeleton**		
	Top 1 (%)	**Top 1 (%)**	**GFLOPs**	**Top 1 (%)**	**GFLOPs**	**Top 1 (%)**	**Top 5 (%)**	**GFLOPs**
Joint	88.1	95.3	6.0	93.2	0.4	36.0	59.1	4.3
Bone	88.3	94.9	6.0	92.5	0.4	35.1	57.9	4.3
Joint-M	86.0	94.0	6.0	92.9	0.4	32.2	55.9	4.3
Bone-M	86.7	93.8	6.0	90.0	0.4	31.9	54.2	4.3
2-s (J+B)	89.2	96.0	12.0	95.2	0.8	37.7	60.6	8.6
4-s	91.2	96.7	24.0	95.7	1.6	38.6	61.6	17.2

data modalities the results are reported on all three datasets for the individual streams, joint+bone stream and joint+bone+ their motion streams in Table 5.3. Combining the data modalities brings considerable improvement in performance. The multi-stream results on all datasets outperform the single-stream results. However, the increased number of streams also contributes to the complexity of the model, which we plan to address in our future work by reducing it. The joint and bone motion modalities typically exhibit lower performance compared to the joint and bone modalities. However, combining them into a multi-stream architecture still brings improvement in performance. We follow

the procedure in MS-AAGCN [7] and generate the final score as a weighted summation of individual action scores.

5.3 Comparison with Baseline

We evaluate our model's performance on individual action classes and compare it with the baseline Shift-GCN on NTU-60 cross-subject and Northwestern-UCLA datasets.

Since NTU has too many action classes, we present the comparative results for only the first 10 action classes in Fig 5.1. Except for two actions which are "Brushing teeth" and "Throw", our proposed architecture generates better predictions for the first 10 actions.

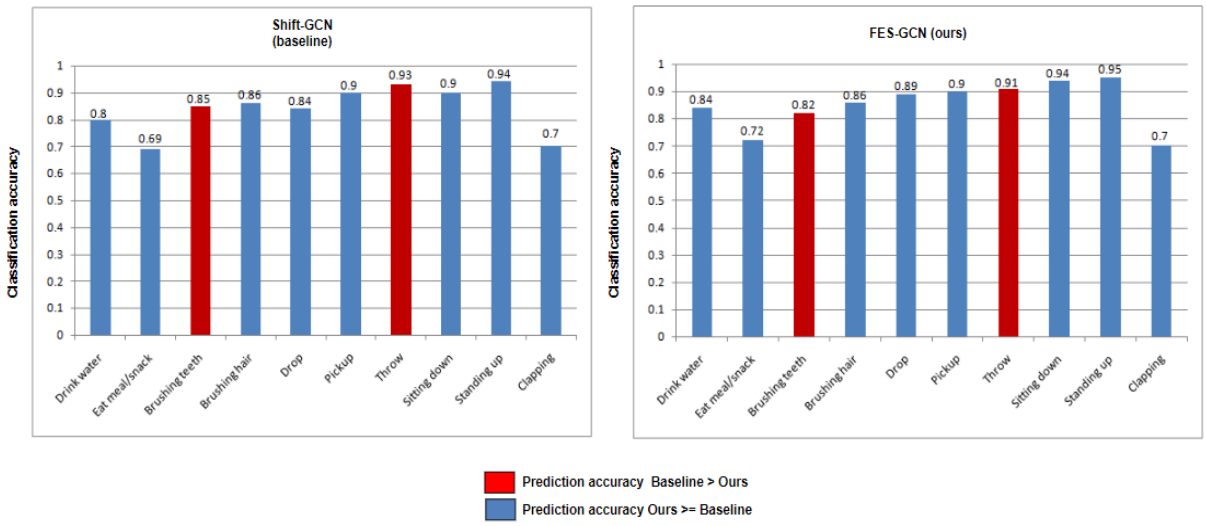

Figure 5.1: Classwise accuracies of first 10 action classes of NTU using baseline model and our proposed model

Fig. 5.2 A and B display the confusion matrices on the joint stream using Shift-GCN and FES-GCN respectively for Northwestern-UCLA dataset. From the confusion matrices, we can observe that except for two classes, which are "Pick up with one hand" and "Stand up", our proposed architecture gives an better prediction accuracy for majority individual action classes. This shows that using skeletal connectivity knowledge enables the model to capture subtle differences in body postures, which

results in a better recognition accuracy of fine-grained actions as compared to the baseline model.

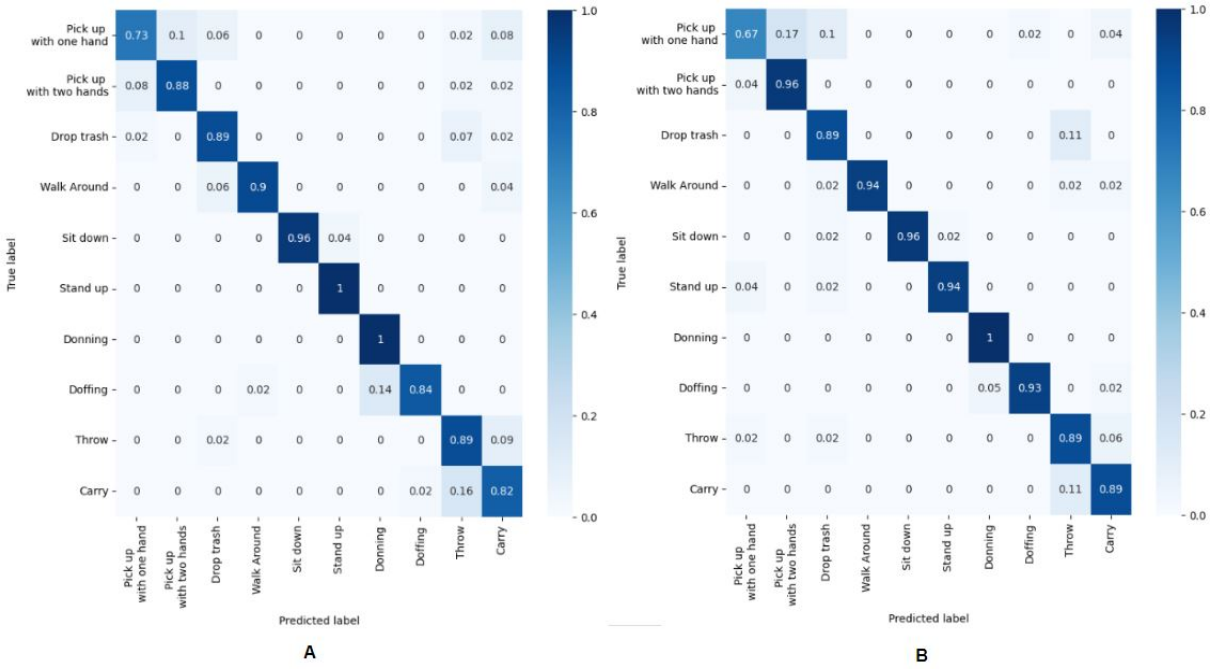

Figure 5.2: Confusion matrices generated by A - Baseline Shift-GCN, B - Proposed model FES-GCN

5.4 Comparison with State-of-the-art Models

We assess the performance of FES-GCN by conducting experiments on the NTU 60, Northwestern-UCLA, and Kinetics-Skeleton datasets, and comparing our model's performance with other state-of-the-art approaches. We show the comparitive results in 5.4,5.5 and 5.6 respectively. The computational results indicate that FES-GCN outperforms the majority of state-of-the-art methods, while requiring lesser GFLOPs than most of the existing approaches.

From Table 5.4 and Table 5.5, we observe that FES-GCN is behind CTR-GCN [8] in terms of accuracy. CTR-GCN learns the topology of each channel in a refined manner, thus showing a powerful correlation capability. However, the operation requires a higher computation cost of 32 GFLOPs on NTU 60 and 3.8 GFLOPs on NW-UCLA

Table 5.4: Comparison with State-of-the-art methods on the Northwestern UCLA skeleton dataset

Method	Year	Accuracy (%)	GFLOPs
Lie Group [33]	2014	74.2	-
Actionlet Ensemble [39]	2013	76.0	-
HBRNN [13]	2015	78.5	-
Ensemble TS-LSTM [40]	2017	89.2	-
Shift-GCN (4s) [1]	2020	94.6	0.7
CTR-GCN (4s) [8]	2021	96.5	3.8
GSTLN (4s) [41]	2023	94.8	-
Ours (FES-GCN)	2023		
1-stream		**93.2**	**0.4**
2-stream		**95.2**	**0.8**
4-stream		**95.7**	**1.6**

while our model requires 24 GFLOPs and 1.6 GFLOPs for the same.

Table 5.6 shows the comparison between accuracies obtained by FES-GCN and other state-of-the-art methods using the Kinetics-Skeleton dataset. Note that the Kinetics dataset, unlike NTU and NW-UCLA, has not been made under controlled laboratory settings and viewpoints. The dataset has been collected from around 400,000 YouTube videos and has 400 action classes, which makes classification task on this dataset quite challenging. Hence, we report both the Top-1 and the Top-5 accuracies on this dataset. On Kinetics-skeleton dataset, our model gives a competitive performance by reporting Top-1 and Top-5 accuracies of 38.6% and 61.6% respectively. It is significantly ahead of the backbone ST-GCN and is slightly behind 2M-STGCN [10] in top-1 accuracy.

In summary, this chapter discusses the effectiveness of each module of FES-GCN through ablation studies and presents the experimental results and comparisons with other methods using three benchmark datasets. The next chapter summarizes and concludes our thesis.

Table 5.5: Comparison with State-of-the-art methods on the NTU RGB+D 60 skeleton dataset

Method	Year	X-sub Top-1 (%)	X-view Top-1 (%)	GFLOPs
Lie Group [33]	2014	50.1	52.1	-
HBRNN [13]	2015	59.1	64.0	-
Deep LSTM [25]	2016	60.7	67.3	-
Synthesized CNN[17]	2017	80.0	87.2	-
ST-GCN (1s) [4]	2018	81.5	88.3	-
DPRL [11]	2018	83.5	89.8	-
AS-GCN (2s) [24]	2019	86.8	94.2	27.0
AGCN (2s) [22]	2019	88.5	95.1	35.8
DGNN (4s) [23]	2019	89.9	96.1	126.8
AAGCN (4s) [7]	2020	90.0	96.2	74.8
Shift-GCN (4s) [1]	2020	90.7	96.5	10.0
CTR-GCN (4s) [8]	2021	92.4	96.8	32.0
GAT (1s) [9]	2022	89.0	95.2	-
AWD-GCN (2s) [32]	2023	89.9	96.1	50.12
2M-STGCN [10]	2023	90.8	96.2	-
Ours (FES-GCN)	2023			
1-stream		**88.1**	**95.4**	**6.0**
2-stream		**89.2**	**96.0**	**12.0**
4-stream		**91.2**	**96.7**	**24.0**

Table 5.6: Comparison with State-of-the-art methods on the Kinetics skeleton dataset

Method	Year	Top-1 (%)	Top-5 (%)
Deep LSTM [25]	2016	16.4	35.3
ST-GCN (1s) [4]	2018	30.7	52.8
AGCN (2s) [22]	2019	36.1	58.7
DGNN (2s) [23]	2019	36.9	59.6
AAGCN (4s) [7]	2020	37.8	61.0
GAT (1s) [9]	2022	35.9	58.9
AWD-GCN (2s) [32]	2023	37.2	61.0
2M-STGCN [10]	2023	39.0	61.6
Ours (FES-GCN)	2023		
1-stream		**36.0**	**59.1**
2-stream		**37.7**	**60.6**
4-stream		**38.6**	**61.6**

Chapter 6

Conclusion and Future Work

Human action recognition is one of the most actively researched areas of deep learning, which aims to understand human behaviour and assign a label to each action. Among the various data modalities used to portray human actions, skeleton data is widely used due to its lightweight nature and robustness against changes in lighting and background. In this thesis, we have proposed a Feature-enhanced shift graph convolutional network (FES-GCN) for skeleton-based human action recognition, which combines skeletal connectivity information with the coordinate information, generating a more enhanced and abundant feature map. The baseline Shift GCN provides a more flexible and lightweight GCN framework as compared to various state-of-the-art GCN models. However, it only utilizes the coordinate information and discards the connection information between the key points involved in the action. This results in a moderate performance while classifying similar actions with subtle differences that require extracting fine-grained features. Hence, our research aims at modifying the baseline architecture so that it utilizes both the graph information and coordinate information to generate a more enhanced feature map. Performing shift operation on a more detailed feature map results in better recognition of fine-grained motion features and a more accurate classification of actions. We utilize a multi-stream architecture as in any action not only joint, but bone and the motion information too play a key role in identifying underlying patterns in the action sample. The network architecture is the same for all the streams. We summarize the network architecture of FES-GCN

through the following points:

1. The network architecture consists of 10 blocks, followed by a Global Average Pooling layer and a softmax layer to predict action score.

2. Each block consists of a spatial unit (SGCN) and a temporal unit (TGCN) connected serially.

3. The spatial unit (SGCN) consists of a spatial graph convolution followed by a spatial shift operation. Spatial graph convolutional unit combines and aggregates the coordinate features and the connection information between the graph coordinates. Spatial shift operation is non-local by nature, which means that it aggregates the features of both neighboring and non-neighboring nodes into each node through shift operation.

4. The temporal unit (TGCN) consists of an adaptive temporal shift operation which diversifies the temporal receptive field of each action frame in an adaptive manner, and provides more flexibility than a conventional 9×1 fixed kernel generally used in GCN architectures [7, 12, 4, 22, 23].

We have trained and evaluated our proposed model on three benchmark datasets: NTU RGB+D 60 [25], Northwestern-UCLA [26] and Kinetics-Skeleton [27]. The details of each dataset have been discussed in Section 2.6.1. The following are the results obtained on each dataset for a four-stream architecture:

1. On NTU 60, FES-GCN reports a cross-subject accuracy of 91.2% and a cross-view accuracy of 96.7% and requires 24.0 GFLOPs computation cost.

2. On Northwestern-UCLA, FES-GCN reports an accuracy of 95.7% while requiring 1.6 GFLOPs computation cost.

3. On Kinetics dataset, our model reports a top-1 accuracy of 38.6% and a top-5 accuracy of 61.6%.

Based on the comparative results in tables 5.4, 5.5 and 5.6, we can conclude that FES-GCN gives a competitive performance and reports high accuracy values while requiring lesser GFLOPs than most approaches.

To summarize, this thesis is a study on skeleton-based human action recognition using a feature-enhanced shift graph convolutional network (FES-GCN). This thesis has investigated an approach to enhance the performance of a lightweight GCN framework Shift-GCN, such that it utilizes graph knowledge along with coordinate information for improved accuracy in classifying diverse actions. Our model delivers a competitive accuracy as compared to various state-of-the-art methods, while requiring lesser GFLOPs then most of the approaches. Some approaches [8], [10] are slightly ahead in terms of accuracy, however these methods involve higher computation costs and have relatively more complex frameworks. FES-GCN delivers a comparatively lighter framework that can efficiently extract fine-grained features and correctly classify actions with similar poses and body movements.

In the future, we intend to come up with a lighter framework for generating the enhanced feature map so that the overall complexity reduces significantly. Specifically, we aim to come up with a more lightweight approach to combine the coordinate information with the graph connectivity information, while maintaining similar levels of performance.

Bibliography

[1] Ke Cheng, Yifan Zhang, Xiangyu He, Weihan Chen, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with shift graph convolutional network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 183–192, 2020.

[2] Tasweer Ahmad, Lianwen Jin, Xin Zhang, Songxuan Lai, Guozhi Tang, and Luojun Lin. Graph convolutional neural network for human action recognition: a comprehensive survey. *IEEE Transactions on Artificial Intelligence*, 2(2):128–145, 2021.

[3] Bichen Wu, Alvin Wan, Xiangyu Yue, Peter Jin, Sicheng Zhao, Noah Golmant, Amir Gholaminejad, Joseph Gonzalez, and Kurt Keutzer. Shift: A zero flop, zero parameter alternative to spatial convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9127–9135, 2018.

[4] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[5] Basura Fernando, Efstratios Gavves, Jose M Oramas, Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5378–5387, 2015.

[6] Preksha Pareek and Ankit Thakkar. A survey on video-based human action recognition: recent updates, datasets, challenges, and applications. *Artificial Intelligence Review*, 54:2259–2322, 2021.

[7] 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.

[8] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13359–13368, 2021.

[9] Jiaxu Zhang, Wei Xie, Chao Wang, Ruide Tu, and Zhigang Tu. Graph-aware transformer for skeleton-based action recognition. *The Visual Computer*, pages 1–12, 2022.

[10] Haiping Zhang, Xu Liu, Dongjin Yu, Liming Guan, Dongjing Wang, Conghao Ma, and Zepeng Hu. Skeleton-based action recognition with multi-stream, multi-scale dilated spatial-temporal graph convolution network. *Applied Intelligence*, pages 1–15, 2023.

[11] Yansong Tang, Yi Tian, Jiwen Lu, Peiyang Li, and Jie Zhou. Deep progressive reinforcement learning for skeleton-based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5323–5332, 2018.

[12] Michael Lao BanTeng and Zhiyong Wu. Channel-wise dense connection graph convolutional network for skeleton-based action recognition. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 3799–3806. IEEE, 2021.

[13] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1110–1118, 2015.

[14] Sijie Song, Cuiling Lan, Junliang Xing, Wenjun Zeng, and Jiaying Liu. An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[15] Pengfei Zhang, Cuiling Lan, Junliang Xing, Wenjun Zeng, Jianru Xue, and Nanning Zheng. View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In *Proceedings of the IEEE international conference on computer vision*, pages 2117–2126, 2017.

[16] Tae Soo Kim and Austin Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In *2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW)*, pages 1623–1631. IEEE, 2017.

[17] Mengyuan Liu, Hong Liu, and Chen Chen. Enhanced skeleton visualization for view invariant human action recognition. *Pattern Recognition*, 68:346–362, 2017.

[18] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Skeleton-based action recognition with convolutional neural networks. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 597–600. IEEE, 2017.

[19] Congqi Cao, Cuiling Lan, Yifan Zhang, Wenjun Zeng, Hanqing Lu, and Yanning Zhang. Skeleton-based action recognition with gated convolutional neural networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(11):3247–3257, 2018.

[20] Bin Li, Xi Li, Zhongfei Zhang, and Fei Wu. Spatio-temporal graph routing for skeleton-based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8561–8568, 2019.

[21] Wei Peng, Xiaopeng Hong, Haoyu Chen, and Guoying Zhao. Learning graph convolutional network for skeleton-based human action recognition by neural searching. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 2669–2676, 2020.

[22] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12026–12035, 2019.

[23] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with directed graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7912–7921, 2019.

[24] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3595–3603, 2019.

[25] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1010–1019, 2016.

[26] Jiang Wang, Xiaohan Nie, Yin Xia, Ying Wu, and Song-Chun Zhu. Cross-view action modeling, learning and recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2649–2656, 2014.

[27] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[28] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[29] Chensheng Li, Xiaowei Qin, Xiaodong Xu, Dujia Yang, and Guo Wei. Scalable graph convolutional networks with fast localized spectral filter for directed graphs. *IEEE Access*, 8:105634–105644, 2020.

[30] Hao Zhu and Piotr Koniusz. Simple spectral graph convolution. In *International conference on learning representations*, 2021.

[31] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

[32] Kai Hu, Junlan Jin, Chaowen Shen, Min Xia, and Liguo Weng. Attentional weighting strategy-based dynamic gcn for skeleton-based action recognition. *Multimedia Systems*, pages 1–14, 2023.

[33] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 588–595, 2014.

[34] Andrew Brown, Pascal Mettes, and Marcel Worring. 4-connected shift residual networks. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 1990–1997, 2019.

[35] Yihui He, Xianggen Liu, Huasong Zhong, and Yuchun Ma. Addressnet: Shift-based primitives for efficient convolutional neural networks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1213–1222, Jan 2019.

[36] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017.

[37] Sunitha Basodi, Chunyan Ji, Haiping Zhang, and Yi Pan. Gradient amplification: An efficient way to train deep neural networks. *Big Data Mining and Analytics*, 3(3):196–207, 2020.

[38] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[39] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Learning actionlet ensemble for 3d human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):914–927, 2013.

[40] Inwoong Lee, Doyoung Kim, Seoungyoon Kang, and Sanghoon Lee. Ensemble deep learning for skeleton-based action recognition using temporal sliding lstm networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1012–1020, 2017.

[41] Meng Dai, Zhonghua Sun, Tianyi Wang, Jinchao Feng, and Kebin Jia. Global spatio-temporal synergistic topology learning for skeleton-based action recognition. *Pattern Recognition*, 140:109540, 2023.