National College of Ireland

Exercise detection and tracking using MediaPipe BlazePose and Spatial-Temporal Graph Convolutional Neural Network

MSc Research Project
Data Analytics

Krishnanunni Raju
Student ID: 20232217

School of Computing
National College of Ireland

Supervisor: Mr. Hicham Rifai

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Krishnanunni Raju
Student ID:	20232217
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Mr. Hicham Rifai
Submission Due Date:	15/08/2022
Project Title:	Exercise detection and tracking using MediaPipe BlazePose and Spatial-Temporal Graph Convolutional Neural Network
Word Count:	7048
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	14th August 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Exercise detection and tracking using MediaPipe BlazePose and Spatial-Temporal Graph Convolutional Neural Network

Krishnanunni Raju
20232217

Abstract

The purpose of this study is to create a deep-learning model based on a pose estimation framework and the Spatial-temporal Graph Convolutional Network that can track and classify exercises performed by humans into different categories. An open-source dataset available online is used to train the model. Despite being a well-liked posture estimation framework, OpenPose struggles while the subject is moving quickly and cannot cope up with cameras with fps greater than 22. This restriction is overcome by using MediaPipe BlazePose. It is used to extract the skeleton information from a moving individual and provides 33 key points of the body. Once a human is identified, the subject is tracked, feature extraction is carried out, and classification is done. The time taken to perform each exercise is recorded. The ST-GCN model implemented in this research was evaluated using accuracy of top-1% and accuracy of top-5%. Four variations of ST-GCN was implemented and evaluated in this research. The best model was obtained when a partitioning strategy of spatial configuration along with learnable edge importance weighting was used for ST-GCN. This model achieved a top-1 accuracy of 41.75% and top-5 accuracy of 89.32%. This solution will to an extend eliminate the need for another person or sensors attached to the body to keep track of the workout.

1 Introduction

1.1 Background

The goal of machine learning and computer vision is to give computers the ability to sense data, understand it, and take action based on previous and present events. Computer vision and machine learning are still in the early stages of development. Just a few examples of computer vision include object detection, pose detection, gesture detection, and other domains. To categorize RGB image sequences from a video into various human postures or activities and estimate human activity, a deep learning technique could be utilized. This study uses a Graph Convolutional Network to categorize human activity. A method called Spatio-temporal graphs is used for feature extraction. The major contribution of this research is to analyze how effectively MediaPipe BlazePose combined with a Spatio-temporal Graph Convolutional Neural Network can track and classify high-speed physical exercises in real-time.

1.2 Motivation

Wearable sensors on the human body could be used to monitor exercises. It is unsuitable to record a variety of workouts, especially when other body parts are involved (Zheng; 2021). There is also the difficulty of connecting a wearable sensor to a specific part of the body and the fact that it can only monitor movement in that one region makes sensor-based methods inefficient. A strategy like this involving sensors has a hefty cost involved as well. Tracking the workout with the use of computer vision and deep learning offers a practical and scalable alternative to the challenge of using sensors.
A person might get a sense of how far they have come in their daily workouts by tracking the exercises. One could make goals for self-improvement by keeping track of this. BlazePose is a modern posture estimation system with the potential to be cutting-edge. It will be used to track people while they engage in an activity and then produce landmarks or key points information. A relatively efficient technique for classifying human activities is the spatial-temporal graph convolutional network. By modifying current ST-GCNs, an advanced activity classification model might be created. Previous approaches had trouble performing in real-time. Combining a highly optimized and upgraded ST-GCN with a lightweight, high-performance posture estimation framework could provide a solution to this problem. The suggested pipeline is efficient and effective in real-time. Depending on the video data provided, the pipeline might be built so that it is scalable for various use cases.

1.3 Research Question

How effectively can the MediaPipe BlazePose and the Spatial-Temporal Graph Convolutional Neural Network track and classify high-speed physical workouts from a real-time video feed?

1.4 Research Objective

The primary goal of this research is to establish a pipeline that uses a posture estimation method and a deep learning model to identify and track a person performing workouts. The objective of this research are as follows:

- Extract key points or landmarks from the video data using an efficient pose estimation framework.
- Construct graphs from the key points provided by pose estimation framework.
- Create an efficient model capable of detecting and tracking exercises in real time with less expense.
- Evaluate the model using test data and examine its real-time performance.

1.5 Outline of the Research Paper

This paper is divided into several sections. Section 2 is devoted to the previous studies conducted in the related field in which the contributions of other researchers using deep learning models and human pose estimation frameworks for tracking and classifying human activities are discussed in depth. Section 3 explains the methodology followed.

Section 4 is Design Specification and explains the architecture of the solution. Section 5 is Implementation and Section 6 is Evaluation. The research findings are summarized, and potential future work is discussed in Section 7 titled Conclusion and Discussion.

2 Related Work

Wearables, instrumented devices, and computer vision have all been used in the past to identify and track exercise. Numerous studies on the tracking and classification of human activities have been carried out over time. The posture estimation algorithm used may function better when depth sensors are used (Zheng; 2021). Several software-based posture estimation frameworks are currently available and might be used to track a person performing any activity. This study attempts to make use of a posture estimation framework that would function effectively in real-time for high-speed activity detection and tracking. Previous research that compared such frameworks are discussed in the first section. After the activity is detected and the landmark data is collected, it must be classified into various exercises. Deep learning or machine learning models might be used to do this. In the part that follows, earlier studies with a similar aim are examined and critically assessed. The conclusions from all the prior research are compiled in the conclusion, and the decisions taken for this research are justified.

2.1 Pose Estimation Methods

BeomJun and SeongKi (2022) compared various pose estimation frameworks. Comparisons were made between the performances of OpenPose, MoveNet Lightning, MoveNet Thunder, and PoseNet. These were all compared using the same pre-classified image data. There were disadvantages and advantages to each framework. In terms of performance, OpenPose was found to be the slowest and MoveNet Lightning to be the fastest. In terms of accuracy, MoveNet lighting was the least accurate, while PoseNet was the most. One might observe that when the model is faster accuracy is lower.

The study by Gadhiya and Kalani (2021) compared the performance of the pose estimation frameworks BlazePose, DeepPose, OpenPose, and Hourglass. These models were evaluated using three different datasets. The performance of the frameworks were really intriguing. With a pushup posture, BlazePose and OpenPose were compared, and it was found that BlazePose produced more precise landmarks. However, on the datasets, OpenPose somewhat outperformed BlazePose. The performance of the Hourglass and OpenPose models was quite similar. Because the dataset used was images instead of videos, the speed at which the activity is carried out is not taken into account.

BlazePose is used to extract essential pose points from the video data the camera captured using a combination of computer vision and deep learning techniques. Then, three machine learning techniques are employed to classify this data after filtering and feature extraction on these key points. According to the experimental findings, the accuracy of the counting algorithm is 97.9%, and the action recognition rate achieved by the ANN algorithm is 97.5%. The potential of MediaPipe BlazePose for activity classification is highlighted by this study. However, since images make up the dataset, each posture is classified according to its pose. Although the classification component is not relevant to this investigation, The performance of MediaPipe BlazePose was proven to be good. Min (2022)

Google created MediaPipe BlazePose and later made it open source. The tracker and the detecter are its two constituent parts. Once a human is identified by the detector, the tracker keeps tabs on the individual in succeeding frames. BlazePose is quicker, lighter, and capable of being employed in mobile devices as a result of this architecture. In contrast to BlazePose, which could process films at 140 frames per second, OpenPose could only process videos at 22 frames per second. Mroz et al. (2021) conducted a comparative analysis, and it was discovered that when the person was moving between frames, when the scene had a strong contrasting background, or when there were extra objects in the frames, BlazePose did not perform well. The research, however, places a high priority on real-time performance, and BlazePose excelled in this aspect.

2.2 Activity Classification Methods

Without the help of a personal trainer, a system for tracking user workouts is presented by Nagarkoti et al. (2019). The technology can spot minor limb positioning errors, which in many workouts can be crucial. This is an example of a physician-patient patient monitoring use case in which a physician can acquire the progress update of a patient from the database. A CNN classifier was trained using the COCO dataset to recognize human action. The error was calculated using the highest deviation for any included limb pair, the average deviation for all involved limb pairs, and the sum of all deviations for all involved limb pairs. Specifically, the 25°, 45°, and 60° deviation angles were examined. The error frames that were expected were validated. Findings showed that the most effective method for locating the incorrect limb pair was to use the maximum deviation criterion. The success of the system depends on the DTW algorithm that was used for frame matching. It suggests that perfecting the leveraging of data from the physical activities could lead to noticeably higher body-part recognition accuracy. Even though future research on these use cases is possible, the method had flaws. It only functions with two-dimensional data, which makes it unsuitable for this research.

Fanuel et al. (2021) conducted a review of recent work on several Skeleton-Based Activity Recognition techniques using Graph Convolutional Networks (GCNs). The author concludes that the Spatial-temporal Graph Convolutional Neural Network is one of the best method for measuring skeletal activity. The traditional implementation of a GCN was discussed first, followed by approaches for addressing the constraints of conventional GCNs. A moving skeleton appears as a collection of linked isomorphic graphs in the temporal domain. As a result, each human skeleton in a frame may be transformed into a graph, with the joints serving as the nodes and the natural connections between joints serving as the edges. The nodes would hold positional information, either 2D or 3D coordinates. Additionally, joint nodes in one frame can be connected to their adjacent nodes in another frame via temporal edges. As a result, generalizing movement into this spatial-temporal graph effectively reduced human action recognition to a graph learning challenge. This technique was found to have increased expressiveness and network bandwidth.

According to Peng et al. (2021), ST-GCN and its variants work very well for skeleton-based action recognition. Unfortunately, either setting the skeleton joint correlations or offering a computationally expensive technique to build a dynamic topology for the skeleton hinders the effectiveness and performance. A spatial-temporal global graph network was created to get around the limitations of earlier models. With the global graph approach, the graph sequence is collapsed into Euclidean space, necessitating the

deployment of a multi-scale temporal filter to effectively and dynamically capture the information. The method with much fewer features was used to extract the graph relationships. Although this approach could be researched, the difficulty of training and pre-processing may preclude it from being a workable solution. It could be explored as a future work using the key points provided by MediaPipe Pose.

Alsawadi and Rio (2021) used an ST-GCN model to recognize daily living activities. The experiment includes a comparison of four different partitioning techniques. Uniform labeling, spatial configuration partitioning, full distance split, connection split, and index split were the four partitioning techniques that were investigated. Uniform labeling is the most basic partitioning strategy available. It was found that the model performed best when index split partitioning was used. When the batch size was increased with index split and spatial configuration partitioning, accuracy continually rose. Performance was comparable for partitioning based on spatial configuration and index split. However, given that the study used OpenPose, which only offered 18 nodes compared to 33 nodes for BlazePose, there is a strong likelihood that both of these methodologies would provide the same outcomes as indicated by the author. The author also suggests using spatial configuration partitioning when spatially localized data is being used.

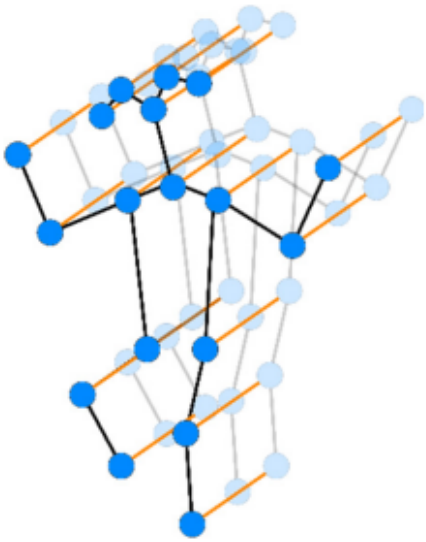


Figure 1: Space-time skeleton diagram, source: Wang et al. (2022)

The most recent work on skeleton-based ST-GCN for activity recognition was done by Wang et al. (2022). This research made use of two large scale benchmark datasets. They are the NTU-RGB-D and the Skeleton-Kinetics dataset with each having 114000 video clips with 120 categories and 300000 video clips with 400 categories respectively. To extract a large portion of the non-adjacent joint relational information in the model for powerful discriminative features, an extended skeleton graph topology and extended partitioning approach was introduced. Joints are represented by vertices on an extended skeleton graph. Weighted edges indicate intrinsic and extrinsic interactions between joints that are physically connected and those that are not. In order to account for as many motion dependencies as possible, the input graph for GCN is further divided using an enhanced partitioning approach into a five-category fixed-length tensor. A variation of learnable edge weighting mask is also used in this research. The schematic representation of the space-time skeleton diagram that was used in this research is shown in

Figure 1. Finally, Spatio-Temporal Graph Convolutional Network is used to implement the expanded skeleton graph and partitioning strategy. The skeleton data extraction was done using OpenPose. The model had a top-1% accuracy of 37.1% and top-5% accuracy of 59.7%. On the NTU-RGB-D dataset cross-subject and cross-view metrics were used. Both were really exceptional with values in the 90s. This study needed a significant amount of computational power and had the disadvantage of utilizing 18 landmarks by OpenPose. When compared to earlier versions, the new ones are excellent. Future studies could assess the efficacy of this model on these datasets by utilizing MediaPipe BlazePose with 33 landmarks.

2.3 Conclusion

It could be concluded that OpenPose is the best available framework for extracting skeleton information. Nevertheless, the performance of OpenPose is below par in a real-time scenario where high-speed exercises are considered (BeomJun and SeongKi; 2022). One drawback of OpenPose was that it could only process 22 frames per second as opposed to 140 for BlazePose. Hence, BlazePose is the suggested framework for activity classification (Mroz et al.; 2021). MediaPipe BlazePose and MoveNet were found to work better for faster and lightweight tracking. Due to the advantage that BlazePose has over MoveNet in terms of accuracy, MediaPipe BlazePose is chosen for this research (Gadhiya and Kalani; 2021).

It can be inferred from the results of previous research that LCN or ST-GCN would perform well with skeletal data. The performance of ST-GCN and LCN was not, however, compared. The two effective partitioning strategies were index splitting and spatial configuration. It should be highlighted that the studies advise employing spatial configuration partitioning for skeletal data (Alsawadi and Rio; 2021). In comparison to other studies, the number of nodes used in this research is more substantial. Consequently, spatial configuration partitioning was picked above the alternatives. In contrast to studies that employed deep learning methods, those that used machine learning models did not provide good performance and results (Fanuel et al.; 2021). Consequently, ST-GCN was chosen in this study to classify skeletal activity. Wang et al. (2022) suggests that a learnable edge importance will enhance ST-GCN. The majority of the studies classified activity using data from various postures (Nagarkoti et al.; 2019). In this study, posture data is not required because video data is transformed into a spatial-temporal graph before being used. As a result, the model could be scaled to detect more activities using video data rather than acquiring images of postures which is difficult. Accuracy of top-1% and top-5% are the recommended metrics to be used if cross-subject and cross-view methods could not be used (Wang et al.; 2022). In this research all the videos have different avatars and have different viewing angles. The videos cannot be distinguished into different avatar sets or viewing perspectives set by any available indicators. Hence, cross subject and cross-view methods could not be used.

3 Methodology

The presented study of exercise tracking using MediaPipe BlazePose and Spatial-Temporal Graph Convolutional Neural Network uses a set of steps based on the KDD approach throughout the entire data mining process.

3.1 Data Acquisition

The dataset used for this is an open-source dataset known as InfiniteRep[1]. Given that it is a synthetic dataset, none of the videos contain any human subjects. Animated avatars that resemble humans practicing different exercises are provided as an alternative. One thousand videos of various avatars completing repeated sets of basic workouts are included. The environment, lighting, demographics of the avatars, and movement trajectories all vary significantly. There are 100 videos for each exercise in the dataset, and each video includes five to ten repetitions of that exercise. Each video has a resolution of 224×224 pixels, a frame rate of 24 frames per second, and is in RGB format. Every repetition is carried out somewhat differently, similar to how real humans move, in terms of cadence and kinematic trajectory. No two repeats are carried out in precisely the same manner. Numerous movement variations allow for the training of robust algorithms. Figure 2 shows all 10 exercises performed by different avatars with different backgrounds. All the videos are in similar format. Annotations for the video which is structured in COCO format is available with the dataset. But these are not useful in context of this research due to the lesser number of landmarks provided.

Figure 2: All 10 exercises performed by different avatars with different backgrounds[2]

3.2 Data Pre-processing and Transformation

After the dataset is acquired, many operations are carried out to extract the necessary data and prepare it for use in training. These steps taken are listed below in their order.

The dataset is kept on a local hard drive, and OpenCV is used to read each frame of every video so that the pose estimation framework can recognize the key points of the skeletal data in each frame.

3.2.1 MediaPipe BlazePose

MediaPipe BlazePose is a machine learning system for high-fidelity body posture tracking, inferring 33 3D landmarks, and background segmentation masks on the entire body from

[1]https://toinfinity.ai/infiniterep
[2]https://github.com/toinfinityai/InfiniteRep

RGB video frames[3]. While most modern mobile phones, in python, laptops/desktops, and even the web can run BlazePose, current state-of-the-art techniques generally require strong desktop environments for inference. A two-step detector-tracker ML pipeline is used for the solution. The pipeline begins by locating the subject/pose region of interest(ROI) within the frame using a detector. The tracker then uses the ROI-cropped frame as input to forecast the pose landmarks and segmentation mask within the ROI. Only when necessary, such as for the first frame or when the tracker was unable to detect the presence of a body pose in the previous frame, is the detector called upon. The pipeline simply calculates the ROI for other frames using the pose landmarks from the previous frame. It is used to extract the landmarks or nodes of a subject and improves the real-time performance of the pipeline.

3.2.2 Data Transformation

Each frame of every video is extracted using computer vision and then the pose estimation framework MediaPipe BlazePose is used to extract key points from each frame. The data is then split into training and testing subsets. Consider that there are many videos overall, a batch size of N is chosen, each with T frames. The maximum number of persons that may be detected in each frame is M, with C for the number of coordinates or channels for each node and V for the number of nodes. Finally, the data will be having a dimension of $N \times T \times V \times C \times M$.

3.3 Constructing Skeleton Graphs

Each human joint in each frame of a skeleton sequence is represented by 3D coordinates. The skeletal sequences are represented hierarchically using the spatial-temporal graph. If there are N skeleton sequences with T frames each that include both intra-body and inter-frame connections, a spatial -temporal graph is created. An undirected graph G is created for each skeleton sequence. This graph will be a function of nodes V and edges E.

$$G = (V, E) \tag{1}$$

The set of nodes in the sequence V will contain values equal to $T \times N$ as shown below.

$$V = \{\ v_{it} \quad \text{where } i = 1, ..., N\ \&\ t = 1, ..., T\ \} \tag{2}$$

Two steps are taken in the construction of the spatial-temporal graph on the skeleton sequences. To begin, edges are used to link the joints within a frame in accordance with how the human body is connected. Each joint will then be linked to the same joint in the following frame. Two distinct sets of edges are constructed. They are inter-frame edges that link a joint in one frame to the same joint in a subsequent frame and intra-skeleton edges that link joints in one frame with the other joints in that frame. The inter-frame edges E_F could be defined as follows.

$$E_F = \{\ v_{it}v_{i(t+1)} \quad \text{where } i = 1, ..., N\ \&\ t = 1, ..., T\ \} \tag{3}$$

Similarly intra-skeleton edges would be a function of nodes in the same frame. If intra-skeleton edges are called E_S, then E_S for any frame could be defined as follows.

$$E_F = \{\ v_{it}v_{jt} \quad \text{where } (i, j)\, \epsilon \text{ Set of natural human joints } \&\ t = 1, ..., T\ \} \tag{4}$$

[3]https://google.github.io/mediapipe/solutions/pose.html

3.3.1 Partitioning Strategies

The weight coefficients are multiplied in a spatial order around the central pixel location in 2-D convolution. Priority criteria must be established in each neighbor-set for graphs without a predetermined order, in order to map each joint to a label. As a result, network training and the convolution process are both possible. The labeling process utilizes partitioning methods. This study uses the partitioning techniques uniform labeling and spatial configuration partitioning, which are both explained below.

- **Uniform Labeling Partitioning**
  The entire neighbour set itself is the subset, which is the simplest and most straightforward partitioning approach. With this method, feature vector of each neighbouring node will have an inner product with the same weight vector. This method has the obvious disadvantage of computing the inner product of the weight vector and the average feature vector of all nearby nodes in the single frame instance. The local differential characteristics may be lost in this technique, which makes it unsatisfactory for skeletal sequence classification. This is also evaluated, and findings were reported, as it is the simplest partitioning that is feasible. Additionally, it will allow the superiority of the spatial configuration partitioning stated below to be proven with 33 landmarks obtained from MediaPipe BlazePose.

- **Spatial configuration partitioning**
  Body part movements can be generally divided into concentric and eccentric motions. The data will be spatially localized because it is skeletal data. As noted in the earlier studies, spatial configuration partitioning is therefore applied. The neighbor is split into three subsets. They are the centrifugal group, the centripetal group, and the root node. When compared to the root node, the adjacent nodes that are closest to the center of gravity of the skeleton are referred to as the centripetal group. The average coordinate of each joint in the skeleton makes up the centrifugal group. It might be thought of as a center of gravity. This can be mathematically represented as follows:

$$l_{ti}(v_{tj}) = \begin{cases} 0 & \text{if } r_j = r_i \\ 1 & \text{if } r_j < r_i \\ 2 & \text{if } r_j > r_i \end{cases} \tag{5}$$

  Here $l_{ti}(v_{tj})$ represents the mapping of the label for node v_{tj}. The average distance from the centre of gravity to the root node and the i^{th} node is denoted by r_j and r_i, respectively. As can be seen, within a neighbour set, two nodes can share a single label.

3.3.2 Edge Importance Weighting

Although joints tend to move in pairs when people are moving, one joint may be present in several different areas of the body. To describe the dynamics of these parts, however, these appearances need to be given varying degrees of importance. To each layer of the spatial-temporal graph convolution a learnable mask M is added. As this is a learnable mask it is also called learnable edge importance weighting. The weight importance that is learned for each edge of the spatial network will be used by the mask to scale the

contribution of a feature of a node to its nearby nodes. From previous studies, it was discovered that using this mask can enhance the capabilities of ST-GCN even more.

3.4 Modelling

A Spatial-Temporal Graph Convolutional Neural Network is used to predict the exercises performed. The input to the network is of the form $N \times T \times V \times C \times M$. Human action recognition could be efficiently reduced to a graph learning problem by generalising skeletal movement into a spatial-temporal graph. Below is a detailed discussion of the spatial-temporal convolution process and the deep neural network.

3.4.1 ST-GCN Convolution Operation

The number of output channels from spatial convolution and the number of input channels for temporal convolution are the same. A 1×1 kernel is used for spatial convolution to ensure that features from one frame are localized with that frame and do not overlap with features from another frame. One value will be provided for each node by the spatial convolution once it has added up all the data from all the channels. The output of the spatial convolution is subsequently multiplied by an adjacency matrix to create a graphical connection for a specific skeleton. The output of the multiplication is then applied to a layer of temporal convolution. A 9×1 kernel is used for this. This will drag the movement of the node through time. Figure 3 shows the pictorial representation of the convolution operation.

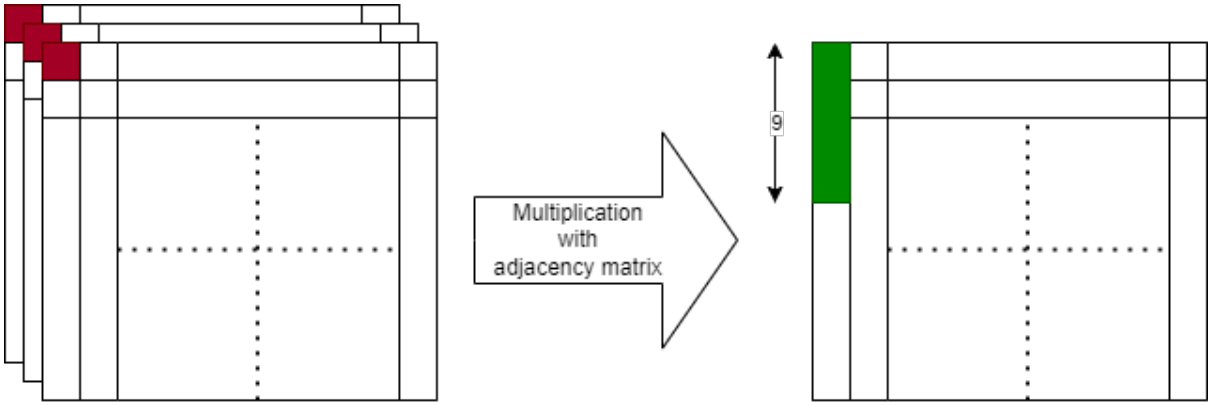


Figure 3: ST-GCN Convolution Operation

3.4.2 Spatial-Temporal Graph Convolutional Neural Network

An adjacency matrix A represents intra-body connections of joints inside a single frame, while an identity matrix I represent self connections. In other words, the adjacency matrix A and the identity matrix I represent the skeleton graph for a single frame. The parameter Γ, also known as the temporal kernel size, determines the temporal range that will be represented in the neighbour graph. In order to implement the graph convolution, a typical 2D convolution $1 \times \Gamma$ is performed, and the resulting tensor is multiplied by the normalized adjacency matrix $\Lambda^{-\frac{1}{2}}(A + I)\Lambda^{-\frac{1}{2}}$ on the second dimension. Here $\Lambda^{ii} = \Sigma_j(A^{ij} + I^{ij})$ and denotes the degree matrix. This is only applicable in case of uniform labeling partitioning. In case of spatial configuration partitioning the adjacency

matrix will be a combination of three matrices. So the value of $(A + I)$ will be equal to $\Sigma_j A_j$, where A_j represents one of the three matrices(Wang et al.; 2022).
When using learnable edge importance weighting a mask M is added to each adjacency matrix. This mask is called a learnable weight matrix. Hence instead of $(A + I)$ it would be $(A + I) \otimes M$ and A_j would be $A_j \otimes M$. Here $\otimes$ is used to represent element wise product of the two matrices.
Given an input feature map F_{in} and weight function W. Let the output value be F_{out}. The network can be realized by the following equation for uniform labeling.

$$F_{out} = \sigma\left(\Lambda^{-\frac{1}{2}} (A + I) \Lambda^{-\frac{1}{2}} F_{in} W\right) \tag{6}$$

If spatial configuration partitioning is used then in Equation 6 the value of $(A + I)$ will be modified as $\sum_j A_j$ and the network could be represented by Equation 7.

$$F_{out} = \sigma\left(\sum_j \Lambda^{-\frac{1}{2}} A_j \Lambda^{-\frac{1}{2}} F_{in} W_j\right) \tag{7}$$

3.5 Evaluation Metrics

The ST-GCN model used for classification is evaluated using two metrics. They are model accuracy, top-1% accuracy, and top-5% accuracy. of the model. Top-1% accuracy and top-5% accuracy are the most recommended metrics among them(Wang et al.; 2022). Previous studies have shown that these metrics are also dependent on the datasets used. Cross-subject and cross-view methods which are another set of recommended metrics could not be used due to the limitations of this dataset.

3.5.1 Top-1 Accuracy

The typical accuracy is top-1 accuracy. This indicates that the model's output value with the highest probability must match the expected result.

3.5.2 Top-5 Accuracy

Top-5 accuracy refers to the ability of the model to produce the five predictions with the highest likelihood of being correct.

4 Design Specification

There are multiple components that make up the overall architecture of the pipeline. Initially video data is made into a folder structure before passing it to the pipeline. An overview of the pipeline is shown in Figure 4 and is described below.

- Pose Estimation using MediaPipe BlazePose:
  The videos are read and converted into frames using OpenCV. Using MediaPipe BlazePose, landmarks or key points of the skeletal data are extracted from each frame.

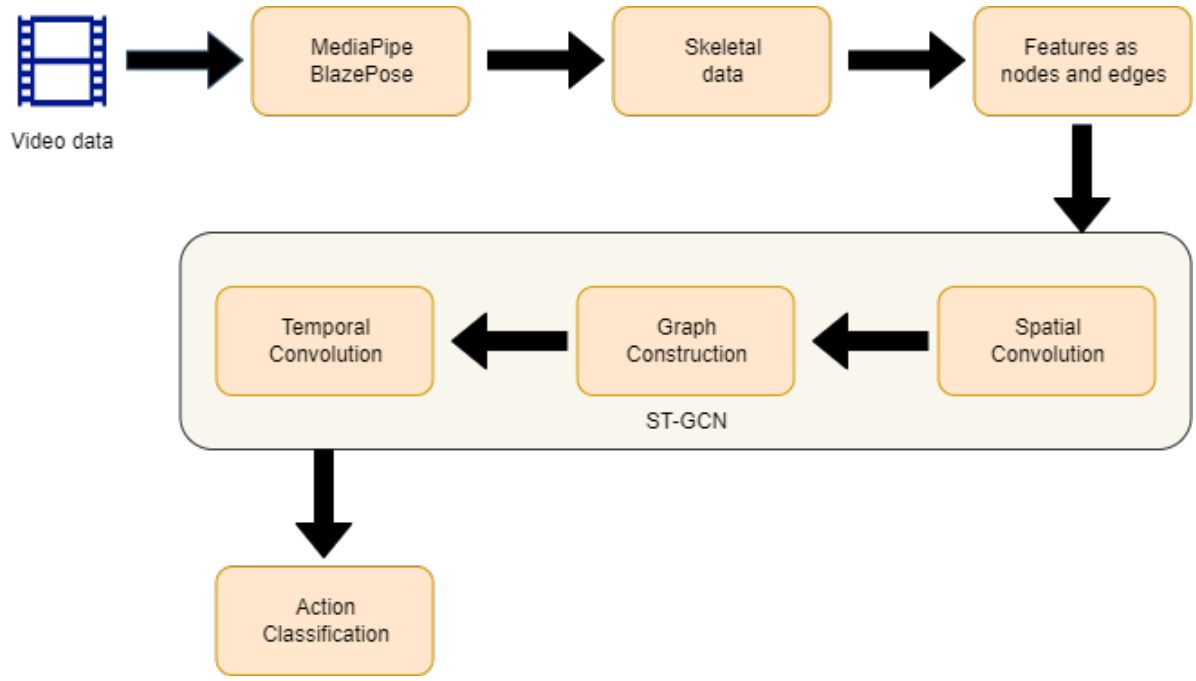


Figure 4: Overview of pipeline

- Skeletal data:
  The landmark data obtained from MediaPipe BlazePose is stored into a .NPY file and labels in a pickle file. This will be then split into training set and testing set.
- Features as nodes and edges:
  The skeletal data is then transformed into two parts, they are edges and nodes. These features are then fed into ST-GCN.
- ST-GCN:
  The data is fed in batches to train the model. The model used in this research is a Spatial-Temporal Graph Convolutional Neural Network. The network uses Spatial-Temporal Convolution operation for feature extraction. The ST-GCN block is realized with the help of the created graph, spatial convolution, and temporal convolution.
- Action Classification:
  The output of the ST-GCN model is then fed into a softmax classifier. It is used to classify the activity at the end.

5 Implementation

The research is done with the help of Python. It is implemented as a pipeline where the data extraction, training and real-time working happens in sequence. Details about the implementation are explained below.

5.1 Frames Extraction and Pose Estimation

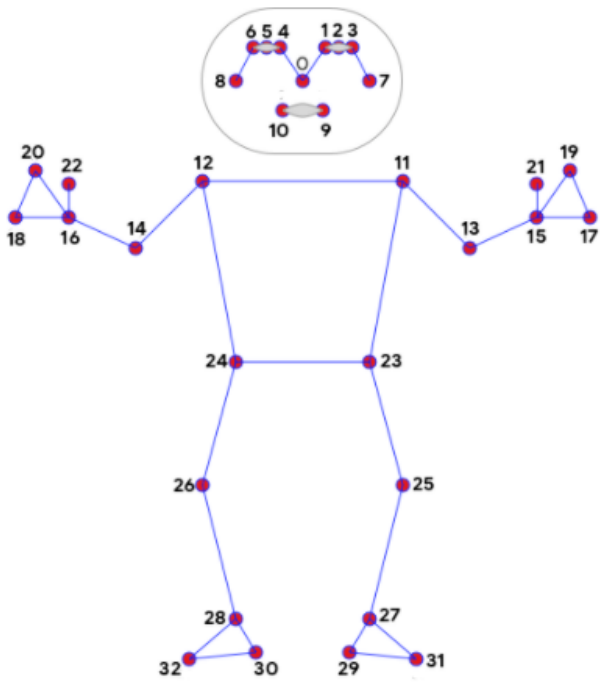
MediaPipe Blazepose API is available as a library for python. The video data is initially converted into multiple frames using OpenCV. The API is ran on each video frame to

generate the landmark values. All videos were trimmed to 200 frames before they were fed to BlazePose because there were several videos with differing frame counts. Since all of the videos had at least 200 frames, that amount was chosen as the baseline for the data. The parameters $min\_detection\_confidence$ and $min\_tracking\_confidence$ range from 0 to 1. These were set to 0.5 because the default values for both parameters are 0.5. The dataset includes 1000 videos that were divided into training and testing sets in an 80 : 20 ratio. The batch size for training was set to 256, hence the value of N is 256. The number of frames in each video is 200 meaning value of T is 200. 33 key points indicates that the value of V is 33 and number of channels C is 3. Only one avatar is present in each video which corresponds to a value of 1 for M. This is represented in Table 1. The data extracted will have a dimension of $N \times T \times V \times C \times M$ which equals $256 \times 200 \times 33 \times 3 \times 1$.

Table 1: Transformed data

Parameter	Representation	Value
Batch Size	N	256
Number of frames per video	T	200
Number of channels	C	3
Number of nodes	V	33
Number of people in a video	M	1

5.2 Graph Construction



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Figure 5: Pose landmarks from MediaPipe BlazePose[4]

The graph construction is done using a Graph Class. The two subsets of edges are defined manually. The first subset is intra-skeleton edges that connect the nodes with other nodes in the same frame. If node 32 is selected it will be linked with nodes 30 and 28. This is done by identifying natural human joints from the landmark information provided by MediaPipe BlazePose shown in Figure 5. The second subset is inter-frame edges. They are formed by connecting the nodes in one frame with the same node in the subsequent frame. It will have a structure of (i, i) meaning node 1 will be connected node 1 in the next frame and that edge will have the structure $(1, 1)$.

The adjacency matrix used for the graph depends on the partitioning strategy used. Two techniques are employed in this study: uniform labeling and spatial configuration partitioning. When uniform labeling is employed the adjacency matrix has a shape of $1 \times 33 \times 33$ and when spatial configuration partitioning is employed the adjacency matrix has a shape of $3 \times 33 \times 33$. This difference in shape is due to the fact that in uniform labeling, the complete neighbor set is the only subset, whereas, in spatial configuration partitioning, the centrifugal group, centripetal group, and root node form the three subsets.

5.3 Classification

The ST-GCN model was implemented using the Python PyTorch library. ST-GCN uses the spatial-temporal graph to carry out a feature extraction from skeletal data. As a result, activity detection is reduced to a graph problem. The following provides a description of the neural network architecture employed in this study. Cross entropy loss is used to calculate the loss in each epoch.

5.3.1 Network Architecture

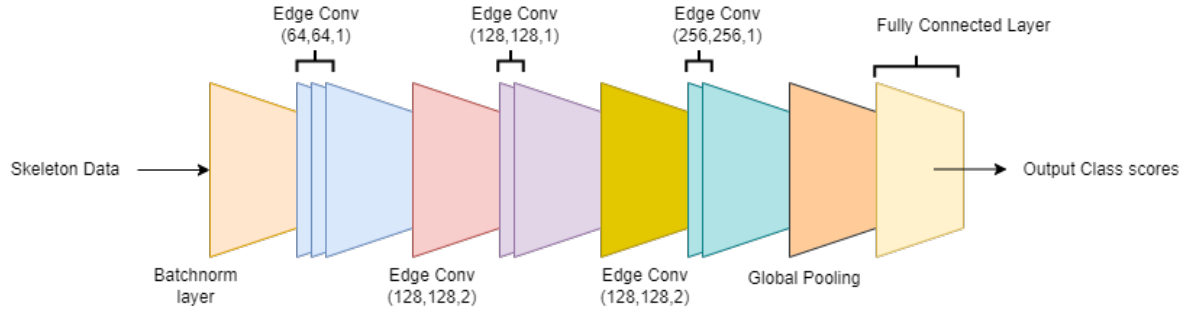


Figure 6: Spatial-Temporal Graph Convolutional Neural Network Architecture

The architecture of the ST-GCN model used is shown in Figure 6. The ST-GCN model consists of nine layers of spatial-temporal graph convolution operators or also called ST-GCN units. Each of the first three layers features 64 input and output channels. In the following three layers, output channels amount to 128. Additionally, the final three layers have 256 output channels. All of these layers have a kernal size of 9, as suggested by the previous studies. The data features are dropped at random with a frequency of 0.5 to prevent overfitting. The hop required to move from one element to the next in the given dimension is called the stride. Stride value is set to 1 for all the layers except

[4]https://google.github.io/mediapipe/images/mobile/pose_tracking_full_body_landmarks.png

the fourth and seventh layer. The stride value is set to 2 for those two layers to act as pooling layers. The resulting tensor was then subjected to a global pooling function to produce a feature vector with 256 dimensions for each sequence. They are then fed into a SoftMax classifier at the end. The models learn using stochastic gradient descent with a learning rate set to 0.01.

5.3.2 Training Process

Once the model is trained with the training data, evaluation is performed using the testing data. Given that the training set consisted of 800 videos and the batch size was 256, there would be 3 batches of data for training during each epoch. In other words, a total of 768 videos were used for training during a given epoch. This occurred as a result of dropping the last incomplete batch. The categorical entropy loss is computed when a forward propagation is finished. Backpropagation is then used to modify the weights. One optimization step is carried out before going on to the following batch of data. The model is used to make predictions for the testing set after training. Results are recorded as the data is propagated in a forward direction to get the prediction. This is then utilized to determine the accuracy of the Top-1% and Top-5%.

5.4 Hyperparameters

The network was trained for 100 epochs. To cut down on training time, the data were separated into batches of 256. To force the dataset loader to discard the final incomplete batch, the *drop_last* argument was set to $True$. Adam optimizer combines the most advantageous features of the RMSProp optimizer and AdaGrad. It was used as optimizer for training the network. Adam was also used by previous research that used ST-GCN. To avoid overfitting, the data features are discarded at random with a frequency of 0.5. How quickly ST-GCN adapts to the problem at hand depends on the learning rate. For training, a learning rate of 0.01 was chosen. Additionally, the previous studies revealed that for ST-GCN, a learning rate of 0.01 was preferable for the best training results. For evaluating the model, testing set was used and the Batch size was set to 64 as it only had around 200 videos. The *drop_last* parameter of the dataset loader was set to False to avoid decreasing testing data.

Table 2: Hyperparameters and their values

Hyperparameter	**Value**
Epoch	100
Optimizer	Adam
Dropout	0.5
Learning rate	0.1
Batch size(Training)	256
Batch size(Testing)	64

5.5 Real-time Tracking and Detection

MediaPipe Pose is used to extract landmarks from each frame of video captured using OpenCV for real-time detection and tracking. The landmark values are retained for K

frames and fed into the ST-GCN model in order to classify the exercise. Here, K is calculated dynamically according to the frame rate of the video feed as shown in Equation 8.

$$K = 2 * \text{Frames per second} \tag{8}$$

This will ensure a minimum of two second window for a person to complete a repetition before passing on the data to ST-GCN for classification and also further enhances the real-time ability of the pipeline. The combination of the partitioning strategy and edge importance weighting used will be determined after evaluating them. The time for which each exercises are performed would be tracked to finally give a report on the exercises performed.

5.5.1 Experiment output

The ST-GCN model was used to perform a real-time experiment and the result are shown in Figure 7. The image on the left is a frame captured by the pipeline when the subject was performing squats, and the image on the right is a frame captured while the subject was performing arm raises.

In this case the camera had a frame rate of 30 and hence the predictions were done using

(a) Detecting squats (b) Detecting arm raise

Figure 7: Real-time working of the solution

60 frames each. The type of exercise and the number of seconds that particular exercise is performed is displayed on the top left corner. The pipeline was successful in identifying and classifying the exercises performed as well as tracking the duration of each one.

6 Evaluation

In this study, four different ST-GCN variants were evaluated. The metrics described earlier were used to evaluate each of these models. It was observed that training duration increased when edge importance weighting was applied. In prior studies, the most recommended metrics were accuracy of top-1% and accuracy of top-5%. Therefore, the models were used to predict the classes of test data, and these metrics were obtained.

6.1 Using Uniform labeling

When uniform labeling was employed as the partitioning technique, the trained model had an accuracy of 78.125% and a mean loss value of 0.56 after 100 epochs. All of the models in the earlier research that used uniform labeling performed poorly when compared to the ones that used other partitioning techniques. This was the case in this research as well. When uniform labeling partitioning was employed, accuracy of top-1% was found to be 28.16%, and accuracy of top-5% accuracy was found to be 97.09% when uniform labeling partitioning was employed.

6.2 Using uniform labeling partitioning method and edge importance weighting

The model has an accuracy of 81.25% after training, indicating that adopting learnable edge importance weighting enhances the model. The mean loss after 100 epochs was found to be 0.445, which is lower than the previous model. On testing data, the model showed a poor performance compared to the previous one. It only had a top-1% accuracy of 18.25% and the lowest top-5% accuracy of all the models with a value of 58.25%. This indicates that the classification done by the model is very poor.

6.3 Using spatial configuration partitioning

The model had an accuracy of 84.76% and a mean loss value of 0.3525 after 100 epochs when spatial configuration partitioning was used in place of uniform labeling. As expected, these results are superior to those obtained using uniform labeling. When the model was used on the test data, the accuracy of top-1% for the model was found to be 14.45% and the accuracy of top-5% was found to be 79.61%. The lowest top-1% accuracy of all the models was with this model.

6.4 Using spatial configuration partitioning and edge importance weighting

When spatial configuration partitioning was used as the partitioning method and edge importance weighting was turned on, the best model out of all four was produced. This model outperformed all others with an 89% accuracy rate and a mean loss value of 0.2712, which is the lowest after 100 epochs. This might also indicate that it is overfitting. This model has the highest accuracy of top-1% with 41.75% and the accuracy of top-5% was 89.32%. The accuracy of top-1% suggests that the model might not be overfitting as mentioned earlier.

6.5 Discussion

It should be emphasized that the best model was produced using learnable edge importance weighting together with the partitioning approach of spatial configuration. A comparison of all the models is given in Table 3. From the results it could be noted that learnable edge importance mask significantly improved the model created using spatial configuration partitioning. In case of uniform labeling there was an increase in training accuracy when the mask was used, but the validation metrics show that it decreased the performance of the model.

Table 3: Comparison of models created

Partitioning	**Edge Importance**	**Training Acc**	**Top-1 Acc**	**Top-5 Acc**
Uni-labeling	False	78.125%	28.16%	97.09%
Uni-labeling	True	81.25%	18.45%	58.25%
Spatial	False	84.76%	14.45%	79.61%
Spatial	True	89%	41.75%	89.32%

When the model from this work was compared to the model from earlier studies, it exhibited superior accuracy in the top 1% and top 5%. Since they were trained using a separate dataset that contained a substantial number of video clips with numerous categories, this comparison is not completely meaningful. It is crucial to highlight that the model has been greatly enhanced by using 33 landmarks supplied by MediaPipe BlazePose. This study demonstrates that this dataset can be effectively used for activity classification. It has been found that even with the use of sophisticated partitioning strategies, the top-1 accuracy was 37% and the top-5 accuracy was 59% when a huge benchmark dataset with each class having close to 1000 videos was used in earlier research (Wang et al.; 2022).

7 Conclusion and Future Work

The purpose of this study is to assess the efficacy of a spatial-temporal graph convolutional neural network and MediaPipe BlazePose in tracking and recognizing exercises performed by a person. The best ST-GCN model was produced using spatial configuration partitioning and learnable edge significance weighting. The model had a Top-1 accuracy of 41.75% and a Top-5 accuracy of 89.32%.

This study used a dataset with 1000 videos divided into 10 exercise groups. There were 100 videos in each exercise category, all with different backgrounds and avatars. Using MediaPipe BlazePose, the skeleton data was retrieved from each frame of each video after the frames from all the videos had been extracted using OpenCV. After that, the data was transformed into a matrix with five dimensions and used for training. This was then divided into a training and testing set. The training set had 80% of the data and the testing set had 20% of the data. With the use of landmark information provided by MediaPipe BlazePose, a graph that is a function of nodes and edges was created. There were two partitioning techniques employed for the graphs: uniform labeling and spatial configuration partitioning. Models were evaluated with and without a learnable edge weighting mask. In total there were four models that were created by using a combination of partitioning techniques and use of learnable edge importance weighting. When

the learnable edge importance weighting and spatial configuration partitioning strategy were employed, the model performed well. Learnable edge importance weighting mask enhanced the network significantly. This model was used to track and detect the exercise performed in real-time. This model had the highest top-1 accuracy of 41.75% and the second highest Top-5 accuracy with a value of 89.32%. Even though model had highest top-5 accuracy when uniform labeling without the learnable edge importance weighting mask, it had comparably low top-1 accuracy of 28.16%.

The pipeline employed in this study might be packaged, put to use on mobile devices, and used as a virtual trainer. The algorithm must be improved to recognize a greater variety of typically performed actions because it has only been evaluated for a limited subset of possible activities. Advanced partitioning algorithms could be used to increase the classification accuracy of the ST-GCN model. A benchmark dataset could be used to train the model and provide a relevant comparison with the prior research. A lot of computation power would need to be used for training in this case. The number of repetitions and the number of calories burnt by a person might both be tracked using an algorithm that could be added to the pipeline. It would become a very viable option as a virtual personal trainer if such a component was introduced to the pipeline.

References

Alsawadi, M. S. and Rio, M. (2021). Skeleton-split framework using spatial temporal graph convolutional networks for action recognition, *2021 4th International Conference on Bio-Engineering for Smart Technologies (BioSMART)*, pp. 1–5.

BeomJun, J. and SeongKi, K. (2022). Comparative analysis of openpose, posenet, and movenet models for pose estimation in mobile devices., *Traitement du Signal* **39**(1): 119 – 124.

Fanuel, M., Yuan, X., Nam Kim, H., Qingge, L. and Roy, K. (2021). A survey on skeleton-based activity recognition using graph convolutional networks (gcn), *2021 12th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pp. 177–182.

Gadhiya, R. and Kalani, N. (2021). Analysis of deep learning based pose estimation techniques for locating landmarks on human body parts, *2021 International Conference on Circuits, Controls and Communications (CCUBE)*, pp. 1–4.

Min, Z. (2022). Human body pose intelligent estimation based on blazepose, *2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*, pp. 150–153.

Mroz, S., Baddour, N., McGuirk, C., Juneau, P., Tu, A., Cheung, K. and Lemaire, E. (2021). Comparing the quality of human pose estimation with blazepose or openpose, *2021 4th International Conference on Bio-Engineering for Smart Technologies (BioSMART)*, pp. 1–4.

Nagarkoti, A., Teotia, R., Mahale, A. K. and Das, P. K. (2019). Realtime indoor workout analysis using machine learning & computer vision, *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 1440–1443.

Peng, W., Shi, J., Varanka, T. and Zhao, G. (2021). Rethinking the st-gcns for 3d skeleton-based human action recognition, *Neurocomputing* **454**: 45–53.
URL: *https://www.sciencedirect.com/science/article/pii/S0925231221007153*

Wang, Q., Zhang, K. and Asghar, M. A. (2022). Skeleton-based st-gcn for human action recognition with extended skeleton graph and partitioning strategy, *IEEE Access* **10**: 41403–41410.

Zheng, H. (2021). A wireless human pose detection method based on digital twin and inertial sensor, *2021 3rd International Academic Exchange Conference on Science and Technology Innovation (IAECST)*, pp. 24–28.