# Human activity Recognition and Authentication System

by

Md. Zubaer ALam
20266033

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
M.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
May 2023

# Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

**Student's Full Name & Signature:**

Md. Zubaer Alam
20266033

# Approval

The thesis/project titled "Human activity Recognition and Authentication System" submitted by

1. Md. Zubaer Alam (20266033)

Of Spring, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of M.Sc. in Computer Science on May 25, 2023.

**Examining Committee:**

Supervisor:
(Member)

Moin Mostakim
Senior Lecturer
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

# Abstract

Automated surveillance motion detection and video data analysis are now crucial jobs for many industries. Understanding human behavior from security video data is crucial, especially in places like banks, hospitals, superstores, and other restricted areas. The two most discussed subjects in the field of computer visions are face detection to identify people and human activity recognition. Over the past 20 years, numerous study projects have been conducted. I'll discuss the Human activity Recognition and Authentication (HARAuth) System initiative in this essay. In this project, I'll suggest an algorithm to identify human activity while also authenticating the individual to determine whether that person is authorized to perform that activity. In this work, I presented a method for classifying and recognizing particular activities based on the pose skeleton of a human. Pose estimation and classification are the first two steps in this procedure. This project uses the OpenPose library for its pose estimation tasks. Additionally, MLPClassifier from the Sklearn library is used to complete the activity classification job. I cropped each person's rectangular area during the pose classification process based on the pose's position in the frame-by-frame video image. Each person's rectangular area is subjected to face recognition in order to verify their identity for the identified action.

**Keywords:** Human Activity Recognition; Facial Recognition; Machine Learning; Prediction; Neural Network; Pose Estimation;

# Acknowledgement

Firstly, all praise to the Great Allah for whom my project have been completed without any major interruption.
Secondly, to my advisor Mr. Moin Mostakim sir for his kind support and advice in our work. He helped us whenever we needed help.
Thirdly, to my parents without their throughout sup-port it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

# Table of Contents

# List of Figures

# Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

$2D$ Two Dimensional

$CNN$ Convolutional Neural Network

$DNN$ Deep Neural Network

$FR$ Facial Recognition

$HAR$ Human Activity Recognition

$HARAuth$ Human Activity Recognition And Authentication

$KNN$ K-Nearest Neighbors

$LSTM$ Long short-term memory

$PAFs$ Part Affinity Field Maps

$ReLU$ Rectified Linear Unit

$RGB$ Red Green Blue

# Chapter 1

# Introduction

## 1.1 Overview of the project

The Human activity Recognition and Authentication System's objective is to anticipate people's particular behaviors and identify individuals by identifying their faces while the activity is being performed. In this project, I developed an action recognition system that uses pose estimation methods to categorize four different kinds of activity and uses face detection to authenticate users. **Figure 1.1** shows the recognized activity in this case "wave" with the name of the person.

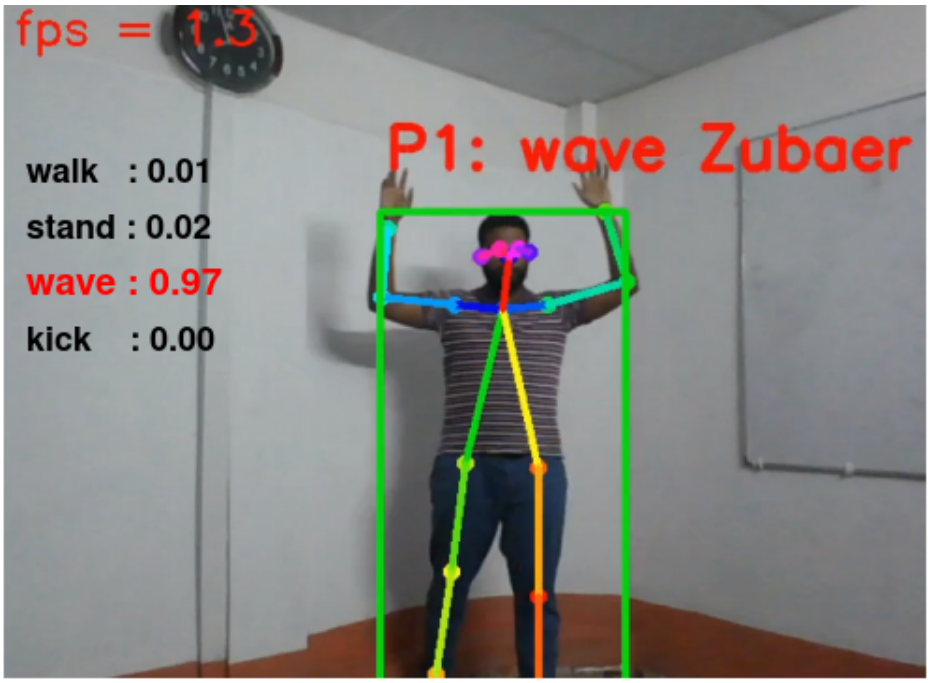


Figure 1.1: Recognition of Human activity and Display of the Person's Name

## 1.2 Applications of this project

This project is both fascinating and practical for real-world use. Some locations needed very restricted people to carry out certain tasks. For instance, only doctors or nurses are permitted to administer injections in hospital rooms, only certain

authorities are permitted to manage transactions at bank cash registers, and only certain people are allowed to move equipment in superstores or warehouses. There are numerous instances or events where it is necessary to identify people and keep an eye on their behavior for safety concerns. Since the last two decades, action recognition using videos from webcams or security cameras has become a hot topic in study fields. Activity recognition systems utilizing sensors are very prevalent, but in the real world or in a crowded setting, it is difficult to provide the sensors to every individual and they are not always comfortable to wear. Because of this, image-based action recognition has recently attracted a lot of interest.

## 1.3 Algorithm/Library used for this project

The Human Activity Recognition can be done in a number of ways. One technique uses a single frame to apply a convolution neural network (CNN) [2] [9] and then average each frame's individual probabilities to produce the final probabilities vector. While Late Fusion is a different approach, it is comparable to Single-Frame CNN in that the network's averaging function is integrated. Because of this, the temporal organization of the frames series is also considered. The RGB and temporal dimensions of the video are fused at the beginning before being passed to the model in early fusion, another method for Activity Recognition Systems that is the opposite of late fusion. Human activity can be identified by conducting video classification using CNN along with LSTMs [8] or optical flow. Although I have other techniques, Activity Recognition using Pose Estimation is the most widely used method. Poses reveal a lot of important facts about people. The idea is useful for a number of tasks, including the recognition of human activity, content extraction, and semantic behavior comprehension. I used pose estimation for activity detection as inspiration from a project - Real-time Action Recognition Based on Human Skeleton in Video by Feiyu Chen [1].
The ability to identify a human face from a digital image or video frame against datasets of faces is known as facial recognition. By identifying and measuring facial characteristics from an image, facial recognition systems are used to identify users. A system like this is very helpful in restricted places. In this article, I refer to it as the "HARAuth system" when it is combined with facial recognition and human activity recognition. In this paper, I suggested a HARauth system that utilized pose-based HAR and Face recognition to automate the process of observing human activity and authenticate the individual simultaneously from the live video data. I use the OpenPose [4] citeb4 library to extract human postures from the video's frame-by-frame analysis. Convolution neural networks (CNNs) [7] are used directly by OpenPose to provide the locations of 18 body keypoints in a 2D plane. The activity being done in the video is then identified using MLPClassifier in comparison to our pre-trained model. At the same time, I crop a rectangular section of the human from the video frame using OpenPose to determine the location of the facial keypoints, and then I apply face recognition to that portion of the cropped area using a different pre-trained model of authorized faces from data sets. In the end, I decide if the individual qualifies for the activity.

# Chapter 2

# Training Data

I take sequential images of 4 type of actions consist of (1) wave, (2) stand, (3) walk, (4) kick for training Activity Recognition Model as shown in **Figure 2.1**


Figure 2.1: Samples of dataset of 4 types Activity Data

I organized each action chronologically using video frames from which the skeleton was clearly visible. It is crucial to capture high-quality pictures that clearly show the joints of the human skeleton. Identical folders containing the activity's name contain the data for each action series. I gathered the facial data of five distinct people, including myself, for the facial recognition model. I added their faces to a directory with five other people's identities. **Figure 2.1** shows the sample of faces I used for this project for training facial recognition model to authenticate the person during activity recognition process.


Figure 2.2: Recognition of Human activity and Display of the Person's Name

# Chapter 3

# Project Algorithm

The overall workflow of the action recognition algorithm is shown in 3.1

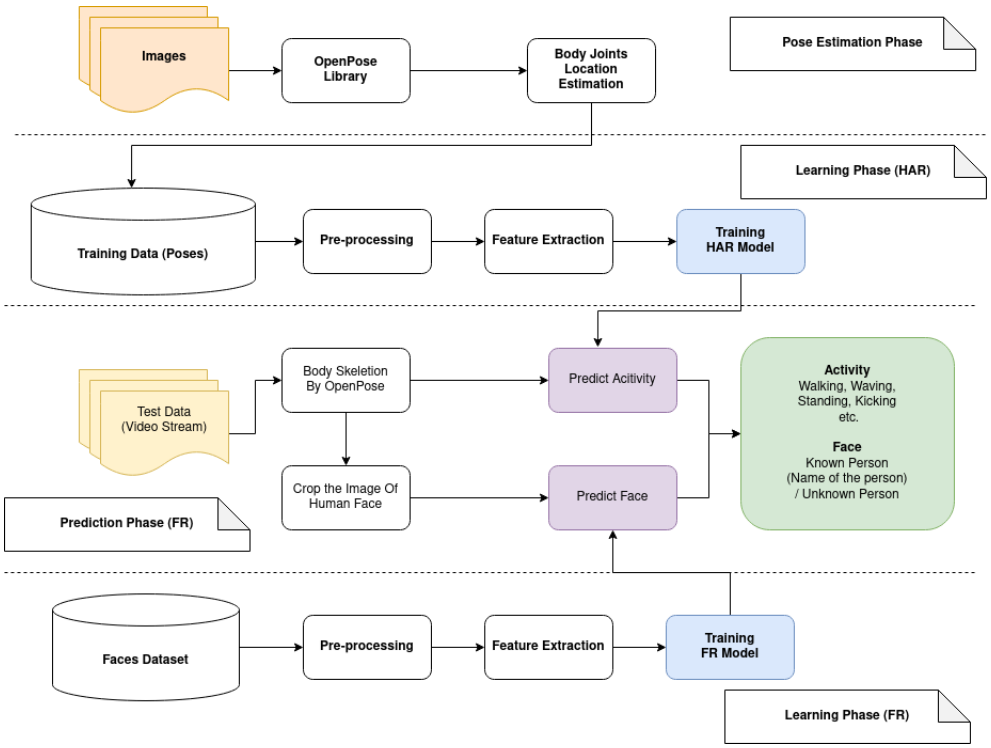


Figure 3.1: Shows the full architecture of the proposed approach

Pose estimation from video frames to identify the human skeleton (joint locations) is the first step in my method for the Human activity Recognition and Authentication System. Next, I use classification to identify the activities using the extracted pose key points as input. To get the final recognition outcome, I used MLPCliassifer from the Sklearn library. (of this window). In order to use facial recognition to authenticate the person, I crop the image at the same time to create a new, smaller-sized image that only includes the person's facial region from the present frame.

## 3.1 Human Pose Estimation

Human pose estimation is the process of identifying and following the major joints of the human body in a picture or video sequence, including the shoulders, elbows, hips,

and knees. This activity is crucial in numerous uses, including surveillance, gaming, sports analysis, and human-computer interaction. In this project, I estimated the human body's pose positions from the input video frame using the OpenPose library.

### 3.1.1 OpenPose

Popular open-source software called OpenPose [5] is used for two-dimensional posture estimation in real-time. In real-time video frames, it can infer the two-dimensional positions of body components. The library, which is built on deep learning algorithms, can determine the poses of one or more individuals in a single frame. Keypoint detection is the primary function of OpenPose, which uses a Convolution neural network (CNN) to forecast the locations of body parts in the video frame or image. A set of heatmaps are created by CNN using an image as input. Each heatmap includes a set of specific keypoints for human body parts at various locations in the image.
OpenPose uses a multi-stage pipeline, which includes the following steps:

1. Preprocessing: Input image is preprocessed to improve and make it fit for the convolution neural network.

2. Keypoint detection: Convolution Neural network is used to predict the heatmaps for each body part.

3. Part affinity fields: The relationships of the different keypoints are detemined by part affinity fields, which represent the likelihood of two keypoints being connected by a limb.

4. Pose estimation: The estimated keypoints and part affinity fields are combined to generate the final pose estimation, which includes the location of all the body parts and the connections between them.

### 3.1.2 Part Affinity Field Maps (PAFs)

Human pose estimation tasks frequently make use of the computer vision and deep learning idea known as Part Affinity Field Maps (PAFs). PAFs are a particular kind of heatmap that are used to show the relationships between important spots in an image.
In short, the OpenPose library is used to identify the human skeleton in a movie or image. Two heatmaps are created by the OpenPose using convolutional neural networks, one for predicting joint locations and the other for connecting the joints to human skeletons. The head, neck, limbs, and legs are comprised of 18 joints in each skeleton. Heatmaps created by OpenPose are shown in Fig. 3.2.

## 3.2 Preprocessing Features

The raw skeleton data are preprocessed before I go for features extraction. The process consist of 2 steps.

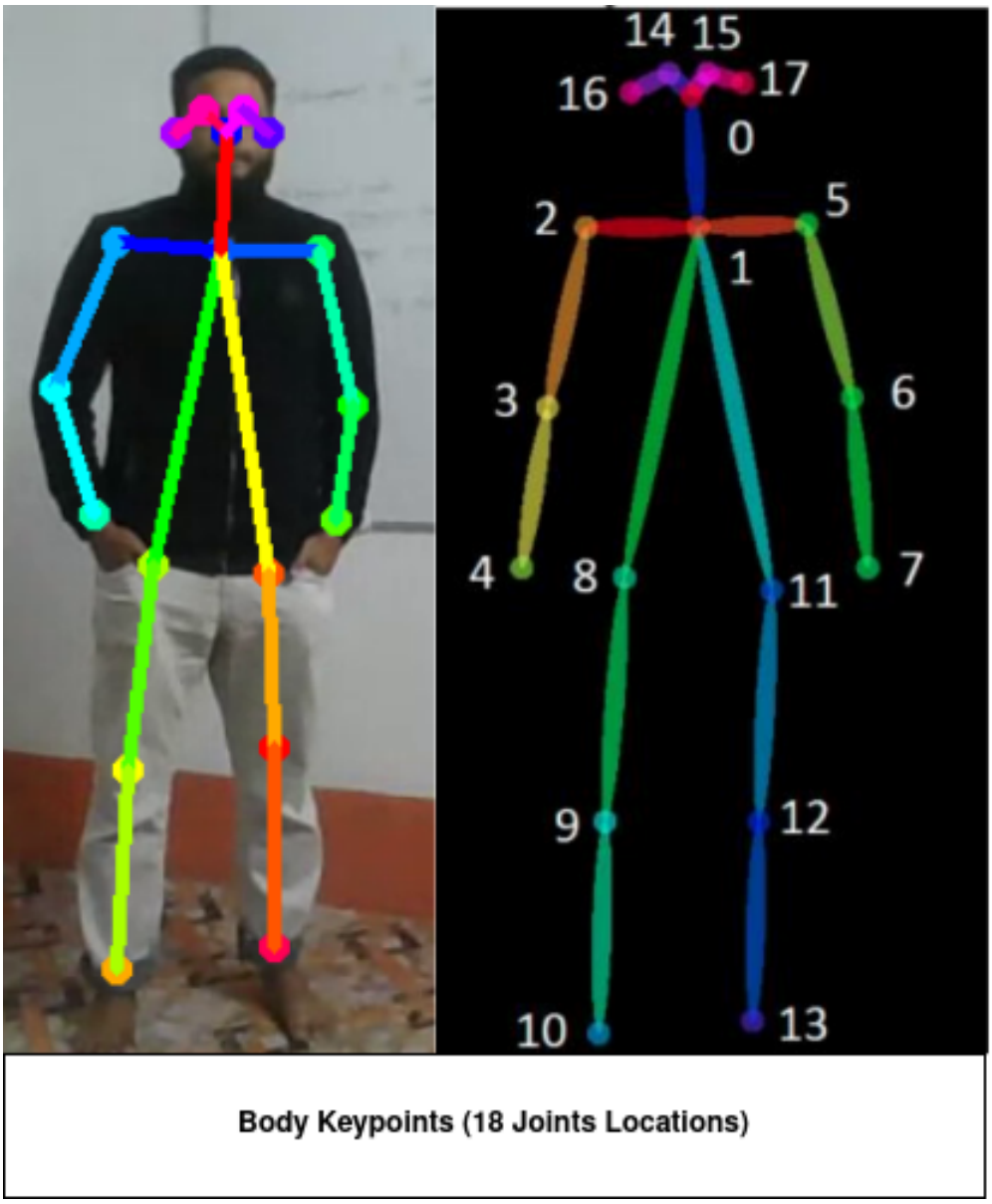


Figure 3.2: Shows pose estimation by OpenPose

1. Scale the joint's position : The skeleton coordinates' various height/width ratios are transformed into the same unit.

2. Fill in the missing ponits: Occasionally, OpenPose is unable to recognize a complete human skeleton from pictures. To keep a complete human skeleton, these missing points of the skeleton must be filled with some values. By these joints' relative position in the prior frame, the missing joints of the person are filled.

## 3.3 Feature Extraction

When the skeleton joints are ideal and finished with the preprocessing features, feature extraction is carried out. To identify features, I used a window size of 0.5 seconds (5 frames). It is based on three variables: the body's velocity, normalized

joint locations, and joint velocities. The feature dimension is then reduced to 80 using PCA. Finally, it is categorized using a Deep Neural Network (DNN) with three 50x50x50-layer levels.

## 3.4 Human Activity Recognition

To identify human activity is one of my project's objectives. Following the estimation of the human pose and feature extraction, the model is taught using the extracted, processed features. The model is created by fitting the features with the help of the MLPClassifier from the Sklearn neural network.

### 3.4.1 MLPClassifier

A multi-layer perceptron (MLP) [3] algorithm for classification jobs is implemented by the Scikit-learn library's MLPClassifier class. One or more layers of fully connected nodes make up an MLP, a type of neural network where each node conducts a weighted sum of its inputs and applies an activation function to the outcome. The numbers of the nodes in the final layer affect the network's output. The MLPClassifier can be used for both binary and multiclass classification problems, and it supports several activation functions, such as logistic sigmoid, hyperbolic tangent, and rectified linear unit (ReLU). It also allows the user to specify the number of hidden layers, the number of nodes in each layer, and the regularization parameter, among other hyperparameters. To use the MLPClassifier, one needs to first instantiate the object and then fit it to the training data using the fit() method. Once the model is trained, it can be used to make predictions on new data using the predict() method. The score() method can be used to evaluate the accuracy of the model on a validation set or test set.

### 3.4.2 Train Dataset and Testing Activity recognition

The five primary steps to create, test, and refine an activity recognition model. Gather skeletons from training pictures first. In order to complete this project, I have collected data on my own standing, waving, and kicking actions and placed them in different folders. For each image in the activity sequences, a unique text file containing skeleton data is used. Second, all of the basic information from each file is combined into a single text file. Thirdly, the text file containing the entire basic information is subjected to a preprocessing of features. The fourth stage involves using MLPClassifier to fit the data to create an activity recognition model. Finally, I launch a video streaming recorder and start recording video data in order to test the recognition algorithm. Each video frame is captured during the main loop, and a pose estimation method is used to determine where the current frame's skeleton joins it. After locating the skeleton's joints, the action is classified using MLPClassifier in comparison to our training model. Mean filtering is used for predicting activity if the score is higher than 0.8 and the prediction scores are between 2 frames.

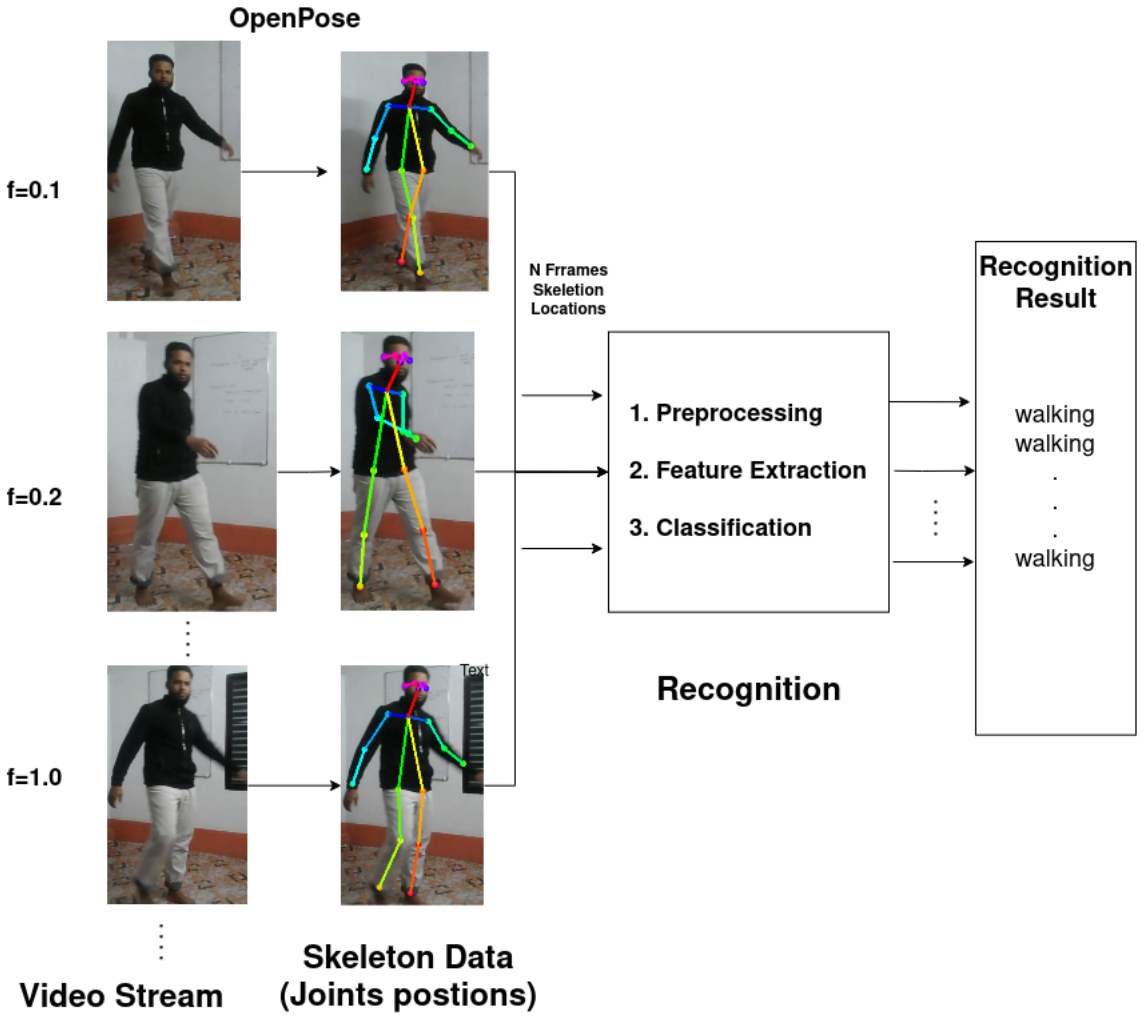


Figure 3.3: Shows Activity Recognition Algorithm

## 3.5 Person Authentication

The last stage of the project is to give the individual permission to participate in the activity recognition process. In order to determine the bounding box of the skeleton, I locate the minx, miny, maxx, and maxy from the positions of joints when I obtain the skeleton on each frame of the video stream. After that, the frame is cropped from the generated picture based on the bounding box. To authenticate the individual, facial recognition was applied to this cropped image.

### 3.5.1 Facial Recognition

In order to recognize people, facial recognition technology compares and analyzes patterns in facial features from an image or video. It functions by taking a picture or video of a person's face and then using sophisticated mathematical algorithms to examine and contrast the image's features with a library of recognized faces to identify the subject.

### 3.5.2 KNN Classifier

K-Nearest Neighbors (KNN) is a type of classification algorithm that can be used to classify data based on its nearest neighbors in a feature space

### 3.5.3 Prepare Dataset and Train Model

I gathered some images of people's faces for this project's facial recognition model training in order to authenticate the individual in the activity recognition process. His facing recognition uses a cropped picture from a skeleton's bounding box as its input. KNN classifier was used to learn and identify the face [6].

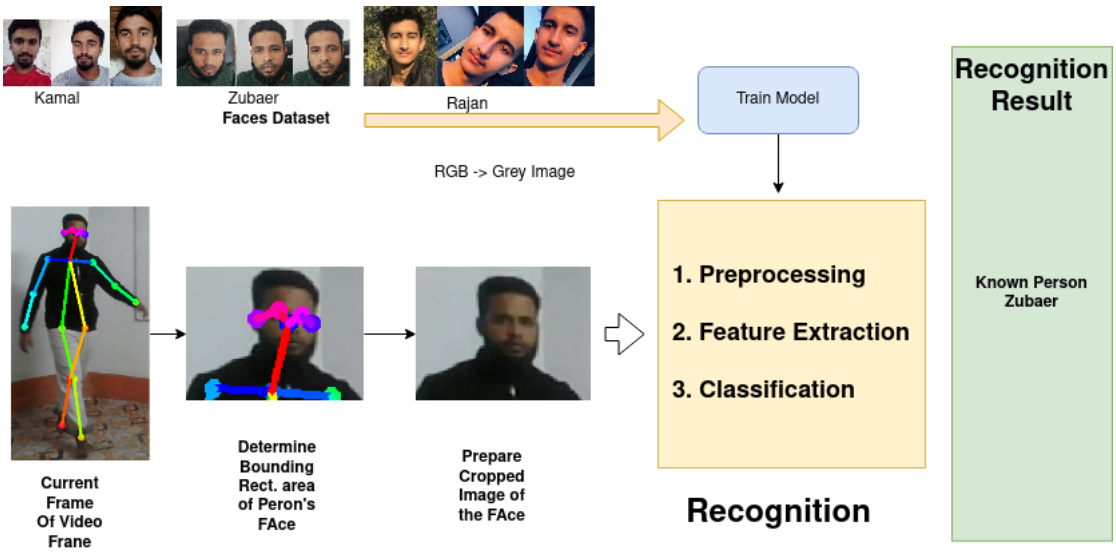


Figure 3.4: Shows Facial Recognition Algorithm

The complete workflow algorithm is:

1. Take each frame from video streams.
2. Use the OpenPose library to extract the joint positions.
3. Locate every person. Two bodies are matched using the Euclidean distance between their joints.
4. Use the relative position of the missing joints in the prior frame to fill in the gaps in a person's body.
5. To attempt to improve the data, add noise to the (x, y) joint positions.
6. To retrieve features, use a window size of 0.5s (5 frames).
7. Highlight the characteristics of joint velocities, normalized joint positions, and body motion.
8. Reduce feature size to 80 using PCA. Classify three levels of 50x50x50 using Deep Neural Network (DNN).
9. Mean filtering between two frames for the prediction values. If the score is higher than, choose the appropriate task for the individual.

10. To create the person's face picture, crop the skeleton's facial portion.
11. Use facial recognition on cropped image to locate the person's identity.

# Chapter 4

# Results Analysis

The output of the project is shown in the **Figure 4.1**
This project only works well for the single person in front of the camera with normal body shapes. Different person with different shapes can make effect on accuracy of the result of the system. Also based on camera quality facial recognition can not be up to the mark. In this case the project will not behave like the expected way. There are several factors that can contribute to poor results.

1. Low-quality input data: Pose estimation depends on accurate detection of key points on the body, which can be difficult if the input data is of low quality. Poor lighting conditions, occlusion, and low camera resolution can all affect the accuracy of pose estimation, leading to incorrect key point detection and inaccurate pose estimates.

2. Differences in human motion: Human motion is highly variable, and it can be difficult to capture all the possible variations in a dataset. This can lead to incomplete coverage of the possible poses and movements, making it more difficult to recognize complex activities.

3. Lack of training data: Activity recognition models require large amounts of labeled training data in order to accurately capture the variability in human motion. If the dataset is small or unrepresentative of the target population, the model may not generalize well to new data.

4. Complexity of activities: Some activities may be very complex and involve multiple stages or components. This can make it difficult to accurately recognize the activity based on pose estimation alone.

5. Camera distance: This can have a significant effect on the performance of facial recognition system. Here are some potential problems that can arise due to camera distance which are Image resolution, Perspective distortion, Occlusion and Lighting conditions. When the camera distance is too high the system will not able to athenticate the person in the frame.

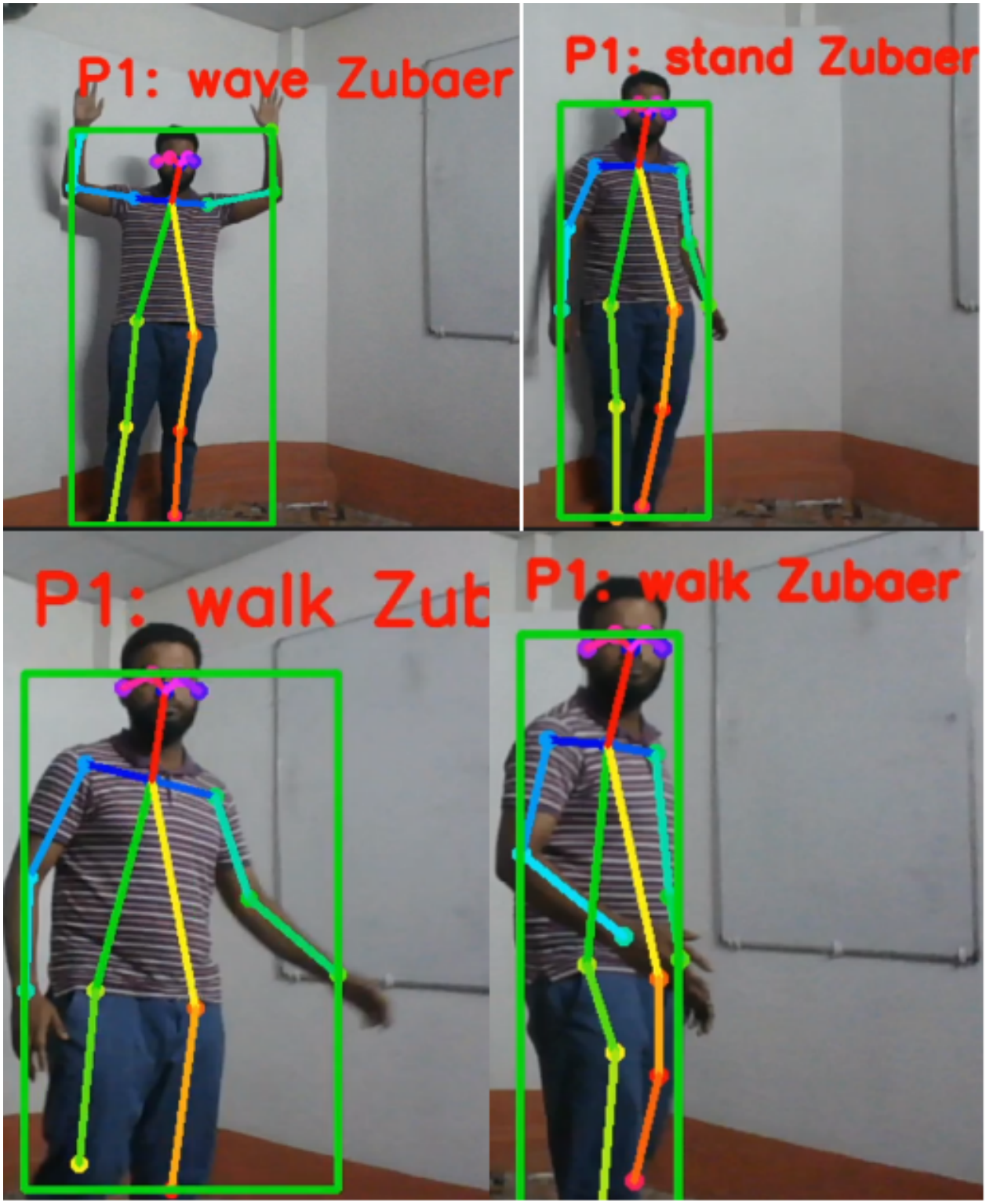


Figure 4.1: Final Output from HARAuth System

# Chapter 5

# Conclusion

In this project, I developed a system for human activity recognition and authentication that can simultaneously use facial recognition to identify the user and recognize four different kinds of actions. The algorithm is built on a real-time framework by averaging the skeleton data from the pose estimation of a 0.5s window for feature extraction and categorization by MLPclssifier and facial identification using KNNClassifier. On the training set, which consisted of few but visually quite distinct single-person activities, the recognition accuracy was acceptable. When multiple people with various body types are present in the video frame, the work fails to function in the same way it is supposed to. A closer distance from the webcam where the entire body and face are visible provides excellent results.

# Bibliography


[1] F. Chen, "Real-time action recognition based on human skeleton in video," *EECS-433 Pattern Recognition Technology*, vol. 2, no. 1, pp. 1–7, 2012, ISSN: 2221-0741. DOI: 2221-0741.

[2] G. D. S. S. E. İ. Cay, "Human activity recognition using convolutional neural networks," *IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology*, 2021, (CIBCB).

[3] S. JOUR Chauhan, "Pattern recognition system using mlp neural networks," *PY - 2012/05/01*, SP-990, EP-993.

[4] OpenPose, "Human pose estimation method," [Online]. Available: geeksforgeeks.org/openpose-human-pose-estimation-method.

[5] OpenPose, "Openpose - basiscs," [Online]. Available: mu-perceptual-computing-lab.github.io/openpose.

[6] F. R. with Python, "Dlib, and deep learning," [Online]. Available: dontrepeatyourself.org/post/face-recognition-with-python-dlib-and-deep-learning.

[7] S. Saha, "A comprehensive guide to convolutional neural networks — the eli5 way," [Online]. Available: towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[8] "Understanding lstm networks," [Online]. Available: colah.github.io/posts/2015-08-Understanding-LSTMs.

[9] R. L. V Aruna S Aruna Deepthi, "Human activity recognition using single frame cnn," Part of the Lecture Notes in Electrical Engineering book series, (LNEE,volume 925).

# How to install LaTeX

## Windows OS

### TeXLive package - full version

1. Download the TeXLive ISO (2.2GB) from
   https://www.tug.org/texlive/

2. Download WinCDEmu (if you don't have a virtual drive) from
   http://wincdemu.sysprogs.org/download/

3. To install Windows CD Emulator follow the instructions at
   http://wincdemu.sysprogs.org/tutorials/install/

4. Right click the iso and mount it using the WinCDEmu as shown in
   http://wincdemu.sysprogs.org/tutorials/mount/

5. Open your virtual drive and run setup.pl

or

### Basic MikTeX - TeX distribution

1. Download Basic-MiKTeX(32bit or 64bit) from
   http://miktex.org/download

2. Run the installer

3. To add a new package go to Start ¿¿ All Programs ¿¿ MikTex ¿¿ Maintenance (Admin) and choose Package Manager

4. Select or search for packages to install

### TexStudio - TeX editor

1. Download TexStudio from
   http://texstudio.sourceforge.net/#downloads

2. Run the installer

# Mac OS X

## MacTeX - TEX distribution

1. Download the file from
   https://www.tug.org/mactex/

2. Extract and double click to run the installer. It does the entire configuration, sit back and relax.

## TexStudio - TEX editor

1. Download TexStudio from
   http://texstudio.sourceforge.net/#downloads

2. Extract and Start

# Unix/Linux

## TeXLive - TEX distribution

### Getting the distribution:

1. TexLive can be downloaded from
   http://www.tug.org/texlive/acquire-netinstall.html.

2. TexLive is provided by most operating system you can use (rpm,apt-get or yum) to get TexLive distributions

### Installation

1. Mount the ISO file in the mnt directory

   ```
   mount -t iso9660 -o ro,loop,noauto /your/texlive####.iso /mnt
   ```

2. Install wget on your OS (use rpm, apt-get or yum install)

3. Run the installer script install-tl.

   ```
   cd /your/download/directory
   ./install-tl
   ```

4. Enter command 'i' for installation

5. Post-Installation configuration:
   http://www.tug.org/texlive/doc/texlive-en/texlive-en.html#x1-320003.4.1

6. Set the path for the directory of TexLive binaries in your .bashrc file

**For 32bit OS**

For Bourne-compatible shells such as bash, and using Intel x86 GNU/Linux and a default directory setup as an example, the file to edit might be

```
edit $~/.bashrc file and add following lines
PATH=/usr/local/texlive/2011/bin/i386-linux:$PATH;
export PATH
MANPATH=/usr/local/texlive/2011/texmf/doc/man:$MANPATH;
export MANPATH
INFOPATH=/usr/local/texlive/2011/texmf/doc/info:$INFOPATH;
export INFOPATH
```

**For 64bit OS**

```
edit $~/.bashrc file and add following lines
PATH=/usr/local/texlive/2011/bin/x86_64-linux:$PATH;
export PATH
MANPATH=/usr/local/texlive/2011/texmf/doc/man:$MANPATH;
export MANPATH
INFOPATH=/usr/local/texlive/2011/texmf/doc/info:$INFOPATH;
export INFOPATH
```

**Fedora/RedHat/CentOS:**

```
sudo yum install texlive
sudo yum install psutils
```

**SUSE:**

```
sudo zypper install texlive
```

**Debian/Ubuntu:**

```
sudo apt-get install texlive texlive-latex-extra
sudo apt-get install psutils
```

# Overleaf: GitHub for LaTeX projects

This Project was developed using Overleaf(https://www.overleaf.com/), an online LaTeX editor that allows real-time collaboration and online compiling of projects to PDF format. In comparison to other LaTeX editors, Overleaf is a server-based application, which is accessed through a web browser.