Visvesvaraya Technological University
Belgaum, Karnataka- 590014

A Project Report On

"HUMAN ACTIVITY RECOGNITION"

Submitted in the partial fulfilment of the requirements for the award of the Degree of

BACHELOR OF ENGINEERING
In
INFORMATION SCIENCE AND ENGINEERING
ACCREDITED BY NBA
Submitted by

Shravani Shirish Urankar (1DS16IS102)
Suresh K (1DS16IS110)
Ranjeet Kumar (1DS16IS078)
Shubham Bhat (1DS16IS103)

Under the Guidance of
MRS. MADHURA J
Asst. Professor, Dept. of ISE

2019-2020

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
DAYANANDA SAGAR COLLEGE OF ENGINEERING
SHAVIGE MALLESHWARA HILLS, KUMARASWAMY LAYOUT, BANGALORE-78

DAYANANDA SAGAR COLLEGE OF ENGINEERING

Shavige Malleshwara Hills, Kumaraswamy Layout Bangalore-560078

Department of Information Science and Engineering

ACCREDITED BY NBA

2019-2020

Certificate

This is to certify that the Project Work entitled **—"HUMAN ACTIVITY RECOGNITION"** is a bonafide work carried out by Shravani Shirish Urankar (1DS16IS102), Suresh K (1DS16IS110), Ranjeet Kumar (1DS16IS078) and Shubham Bhat (1DS16IS103) in partial fulfilment for the 8th semester of Bachelor of Engineering in Information Science & Engineering of the Visvesvaraya Technological University, Belgaum during the year 2019-2020. The Project Report has been approved as it satisfies the academics prescribed for the Bachelor of Engineering degree.

_____	_____	_____
Signature of Guide [Mrs.Madhura J]	Signature of HOD [Dr. K.N. Rama Mohan Babu]	Signature of Principal [Dr. C.P.S Prakash]

Name of the Examiners	Signature with Date
1._____	_____
2._____	_____

ACKNOWLEDGEMENT

It is great pleasure for us to acknowledge the assistance and support of a large number of individuals who have been responsible for the successful completion of this project.

We take this opportunity to express our sincere gratitude to **Dayananda Sagar College of** for having provided us with a great opportunity to pursue our Bachelor Degree in this institution.

In particular we would like to thank **Dr. C. P. S Prakash,** Principal, Dayananda Sagar College of Engineering for his constant encouragement and advice.

Special thanks to **Dr. K.N. Rama Mohan Babu,** HOD, Department of Information Science & Engineering, Dayananda Sagar College of Engineering for his motivation and invaluable support well through the development of this project.

We are highly indebted to our internal guide **Mrs.Madhura J,** Asst. Professor, Department of Information Science & Engineering, Dayananda Sagar College of Engineering for his constant support and guidance. He has been a great source of inspiration throughout the course of this project.

Finally, we gratefully acknowledge the support of our families during the completion of the project.

Suresh K (1DS16IS110)

Shubham Bhat (1DS16IS103)

Ranjeet Kumar (1DS16IS078)

Shravani Shirish Urankar (1DS16IS102)

ABSTRACT

The purpose of this project was for us to develop a model that would take the video input from different sources like security cameras and recorded videos and detect actions that are being performed in them by the people also commonly referred to as HAR or Human Activity Detection.

The project began with the analysis of models and procedures that already exist to perform this activity along with the assessment of their advantages and disadvantages respectively. There were many approaches to build this project that mainly used sensors (tri-axial accelerometer and gyroscope) and various modelling procedures such as CNN's (Convolutional Neural Networks), RNN's (Recurrent Neural Networks), DNN's (Deep Neural Networks) and few Image Processing techniques. The goal is to come up with a model that is able to detect a selection of activities on which the model was trained during the development phase. The designed model helps to overcome the disadvantage of the previously built models that relied on the use of sensors. With the elimination of sensors, we eliminate the cost associated with respect to the sensors and the errors that can be caused due to the use of sensors.

Our team has used the logistic regression/logistic classification to help detect a set of activities. It compares the action performed in videos given as input to the trained model used to detect these actions and if the similarity to the action is found to be greater than the set threshold it is displayed as the output on the top right corner.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Overview:

Video surveillance is being used nowadays in most public places such as schools, colleges, shops, libraries, parks, metro stations, hospitals and on roads to monitor traffic flow. With this increase in footage gathered by the cameras, there is a need to develop a process or method in order to make sense of the activity or action being performed in the videos.

Human activity Recognition abbreviated as HAR is used to make sense of the activities that are being performed by the humans in such videos/camera feeds.

Given the number of cameras, it's not possible for a person or group to monitor them 24*7 with high accuracy while keeping down operating costs. This problem can be solved by using a computer program or software which only has to be developed once and can be implemented and run at large scale to do specific activity detection.

It all begins with the process of designing the model for detection of activities. Firstly a set of activities to detect are determined. This is followed by the process of collection of the data collection for these activities. The collected data is then cleaned and pre-processed while making sure to remove redundant or unnecessary data. Further steps include the extraction of key features from the data to help build the model to detect human activities in real-time. This data is labelled and used to train the model training in the supervised learning process.

The model is then deployed on the camera feeds and the activities performed are determined by the trained model.

1.2 Problem Statement

Security Surveillance is expensive and 24*7 surveillance by humans carries a very high operational cost and humans are also prone to errors during detection. Human activity detection can be done to recognize a set of human activities by training a supervised learning model and displaying the activity being done as a result as per the input activity received from the camera input.

1.3 Objectives

- To prepare a dataset consisting of images for 6 activities namely walking, hand waving, squats, talking on the phone, standing and performing Meditation.
- The dataset is segmented and the relevant data is annotated with labels of the above-mentioned activities.
- Data is pre-processed and redundant and irrelevant data is edited out.
- The development of an optimised model is carried out that uses Logistic Regression as a classifier to segment the dataset images into their respective action labels. This is done following a supervised learning procedure for training the model. The threshold is also set to determine the probability of correctness in the determination of the action being performed.
- Demonstrating the results.

1.4 Motivation

With the increase in video surveillance there is no need to to develop an efficient way to detect what is happening in the videos. With the help of human activity detection models these cameras can be used to provide 24/7 security. The act of detection of the activity of humans has gained attraction nowadays and is a hot field for research and development. The ability of human activity detection is a key prospect of development in fields of human-computer interface, computer vision and in mass video surveillance in public areas such as train stations, ATM machines, schools and colleges and at traffic signals on streets. HAR offers a cost-effective method for surveillance. With the development of a HAR model various sectors will be benefited.

CHAPTER 2

LITERATURE SURVEY

In this section, we will study various approaches used to propose human activity recognition based on designed models.

1) The **Research paper** on "**Human Activity Recognition Using Smartphones** "proposed by **Erhan BÜLBÜL** [1] and **Akram Bayat** gave us knowledge on support vector machines(SVM).

   This paper introduces SVM that makes the best use of hyper-dimensional planes in order to separate examples. Although SVN can be used with and without supervision, it is usually quicker and more successful to use supervised SVN. When supervising SVM with a cubic polynomial kernel used to identify tuples in the dataset, a high success rate of 99.4 percent was achieved.

2) The Research paper on "**Human Activity Recognition Using Deep Learning Networks with Enhanced Channel State information**" proposed by **Zhenguo Shi**[2], **J. Andrew Zhang, Richard Xu**, and **Gengfa Fang**.

   They explored an approach that included the study of recurrent neural networks (RNN) in which node-to-node links shape a directed graph along a timing chain. This type of neural network is usually used in examples including timing series. A model is trained using this approach in order to obtain the temporary dynamic behaviour. It has a segment called a memory segment which mainly processes variable length of input sequence. In order to transform and extract the inherent features from the input data collected from CSI-HAS, Deep Learning networks are used. To further reduce and optimize the feature, a Sparse auto-encoder (SAE) network is utilised. One of the de-merit of using this is that the sensing performance of SAE is susceptible to input quality. To overcome this problem an approach of Recurrent Neural network based on the concept of Long short term memory (LSTM) is used by taking a CSI packet as a raw input. Adding to it, LSTM-RNN based models can also be used to extract features at the later part of the entire process. The classifier here being used is

the Softmax Regression Algorithm. The coefficients of the trained network is used as equipment for the further training process.

3) **Anjana Wijekoon** [3] and **Wenchao Xu** [11] in their research paper "**MEx: Multi-modal Exercises Dataset for Human Activity Recognition**" explored various approaches out of which convolution neural networks (CNN, or ConvNet) is commonly used to process visual images.

They are commonly referred to by move invariant or Space Invariant Artificial Neural Networks, that mainly concentrate on invariance characteristics of translation and common design. To equalize the performance of every secret sheet an approach of Batch Normalisation is applied over the dataset. They define three different segments of the classifier and sensors and each produces an output vector of 100 sizes. DCT-1D input: ACT and ACW. Changes with respect to 1 Dimension and further into the max-pooling hidden layer where the channel count is 1 and the input feature is 180 bits long.Raw-1D: 1-dimensional convolutions and the levels of pooling. ACT and ACW uses a window frame of size 500 (window frames per second) raw data with 3 channels (3 accelerometer axes). The vector is formed by flattening the frame and a timing window is obtained by processing the frames. Single-dimensional display feature-length frame width vector for frame width height/frame height/frame per second for 1 path.–2D: This approach mainly uses two-dimensional convolution and total levels of pooling. 1channel in a timing window along with a 2-dimensional vector is obtained after extracting the information via PM as well as DC.

4) **Akbar Dehghani** [4],**Tristan Glatard** and **Emad Shihab** in their paper "**Subject Cross-Validation in Human Activity Recognition**" proposed a model that uses k-fold Cross-Validation as the primary methodology for testing model output and adjusting hyperparameters.

This means measurements are Random and Identically Distributed, i.e. data points are taken from the same source separately. Nonetheless, samples belonging to the same topic are likely to be interrelated in HAR datasets because of the underlying physical, biological and demographic factors. Additionally, there is often a temporal correlation between a subject's samples due, for example, to fatigue, training, experience. Consequently, k-fold CV could overestimate the performance of the classification by relying on correlations within subjects.

5) **Stefanie Anna Baby** [5] proposed in a paper titled "**Dynamic Vision Sensors for Human Activity Recognition**" proposed a model which includes a dynamic vision sensor, it processes only the foreground objects that being the humans captured the camera while completely ignoring the unnecessary processing of backgrounds thus decreasing processing requirements and greatly enhancing the detection efficiency of the given model.

   The difference in intensity of every pixel solely depends upon the texture edge and hence the output extracted from DVS is very sparse. To obtain higher confidence in Activity Detection, various segments of the DVS data are also represented as motion maps which are further used to describe features of interest although while using the sparse data and segments of movement captured by the DVS. Moreover, motion maps along with the features of interest are combined together to process them thereby increasing the activity recognising accuracy. The event stream from DVS is processed using a timing window to obtain a video segment. Hence by obtaining the mean over the time axis, x-y, x-t, y-t projection is obtained.

6) **Gaming Ding** [6], **Jun Tian**, **Jinsong Wu**, **Qian Zhao**, **Lili Xie** in the paper "**Energy-Efficient Human Activity Recognition Using Wearable Sensors**" used a model that uses a decision tree-based approach to classify different human activities.

   The classification accuracy of a single tree classifier is enhanced by the concept of bagging which helps randomize the selection of the partitioning data nodes in the tree construction. We take a majority vote based on the decision tree provided and assign the vector to the given class. This process is done for each of the trees in the forest. This particular process requires a large collection of labelled data to achieve the required high accuracy. For classification, we make use of accelerative data. The model with its highest accuracy case outperforms the classification capability of SVM and Naive Bayes classifiers.

7) **Artur Jordao** [7], **Antonio Carlos Nazare**, **Jessica Sena**, **William Robson Schwartz** in a paper titled " **Human Activity Recognition Based on Wearable Sensor Data: A Standardization of the State-of-the-Art** ",used a model in which ample amount of

data is collected from certain sensors named tri-axial accelerometer, magnetometer and gyroscope by which human activities are detected and classified into different classifiers based on recognition.

To achieve this one methodology is the way of representing the raw signal to be able to clearly distinguish between human activities exploring at the classification stage. The signals used are signal LTCV and SNLS. They mainly emphasize on rendering the leave which was proposed as leave-one-trial-out. This approach is widely used along with cross-validation. This proposed method also holds higher confidence and accuracy levels for statistical models.

8) **Fabien Barade**l [8], **Christian Wolf**, **Julien Mille**, **Graham W. Taylor** in their paper "**Glimpse Clouds: Human Activity Recognition from Unstructured Feature Point** " proposed the model that uses the method of human activity detection with reference to the feature points captured along with the input data.

We do not need to directly perform pose detection but rather perform two important processes,one to extract the relevant data with respect to the feature points and the other to derive inference based on the values of these points over time intervals. An unstructured set of data is collected as glimpses with the values extracted from trackers and recognizers. The pose that is detected is used in the training of the model while keeping the focus of the model on the structure of the human body based on the feature points. The attentional mechanism is carried out in the feature space which is used in calculation with the global model for the complete processing of the image obtained from the camera feed.

9) **Sumaira Ghazal** [9] and **Umar S. Khan** in their paper titled "**Human Posture Classification Using Skeleton Information**" used an SVM classifier to extract human skeleton information such as the location of joints of the human body shape in skeletal form from the camera input. This data can be used in the estimation of the human pose and can also be used to detect the activity associated with the motion of these data points based on the input obtained by the camera feed.

CHAPTER 3

REQUIREMENTS

3.1 Functional Requirements

- The proposed model should be able to recognise 6 Human Activities namely walking, hand waving, squats, talking on the phone, standing and meditation.

3.2 Non-Functional Requirements

- The model should provide an accuracy of 50% or more.
- The model should work consistently across various platforms.

3.3 Software Requirements

- OS Version: Ubuntu 18.04 LTS
- Coding Language: Python 3.6
- Code Editor: Visual Studio Code

3.4 Hardware Requirements

- Processor: Intel® Core™ i5-6200U cpu@2.30Ghz
- System type: 64-bit Operating system, x64-based processor
- Primary Storage: 8GB RAM
- Secondary Storage: 1TB
- GPU: NVidia GTX 1050
- Webcam

CHAPTER 4

SYSTEM ANALYSIS & DESIGN

4.1 Analysis

System Analysis is used to identify the goals and purpose of the system that we have designed to achieve an optimal solution during implantation. This is done by studying the procedures used and the business requirements. Analysis includes the division of a large problem into multiple smaller more easily solvable problems. During this phase we look at the requirements and structure, mechanism to solve the problem and the system dimensions available to us. The Analysis is the activity of exploration of the system that has been developed. The project development life cycle begins with the analysis phase.

4.2 System Design

In this process we define the system architecture, modules used, interfaces used, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to the development of the product.

The design phase helps to produces the overall design of the software. The goal of this phase is to figure out the different modules can be used for the given system to achieve its goal with the greatest possible accuracy and efficiency. The system design contains details about each of the modules being used along with the way they interact with the other modules and help produce the output. The output of the design process is a description of the software architecture.

4.2.1 System Architecture Diagram

The architecture of a large scale service will have high level of complexity. It can have several micro services deployed running in conjunction with each other in a distributed environment. The comprehensive architecture of a service that involves several different components is called the system architecture.

The system architecture design shows us the relationship between the different components being used. They are usually created for gaining a deep understanding of how the different components work with each other in order to achieve the goals that were set to be achieved by the project.

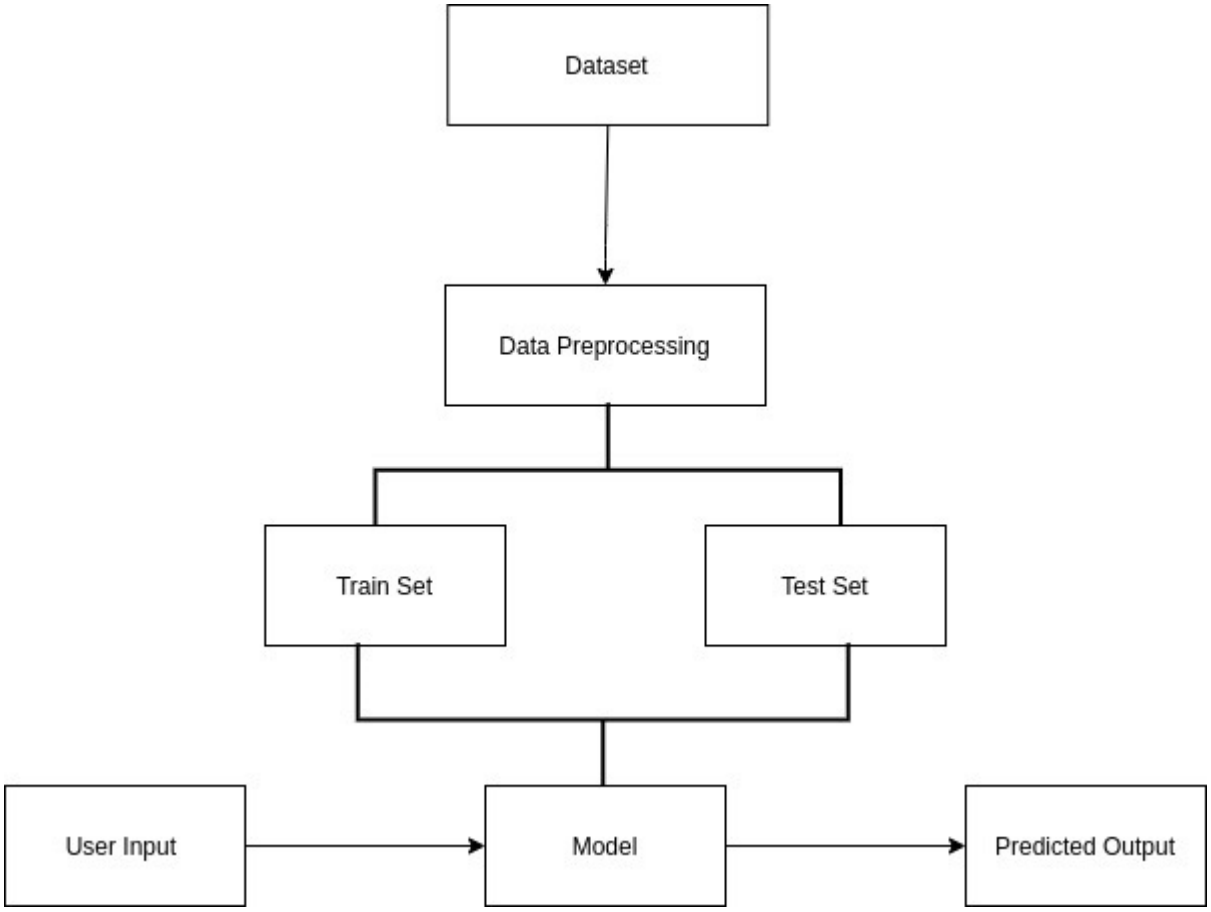

Fig. 4.2.1 System Architecture Design

4.2.1.1 Data Flow Diagram

A data-flow diagram is a way of representing the flow of a data of a process or a system (usually an information system). The data flow diagram also helps us to monitor what data we are feeding to a given component of the program and what output data it generates after processing. A data-flow diagram doesn't have any control flow as there are no decision statements or loops. The data flow diagram is just the graphical representation of the flow of data through the information system. The Data Flow Diagram is very useful in understanding a system and can be efficiently used during analysis.

The data flow diagram helps us to represent the functions and processes that capture, manipulate and store the data in a graphical format. It also shows how the data flows between the various modules of the system.

This visual representation is easy to understand and makes it easy for the user and system designer to communicate with each other using design process. It also helps us to find any lapses in security where the data flow is compromised.

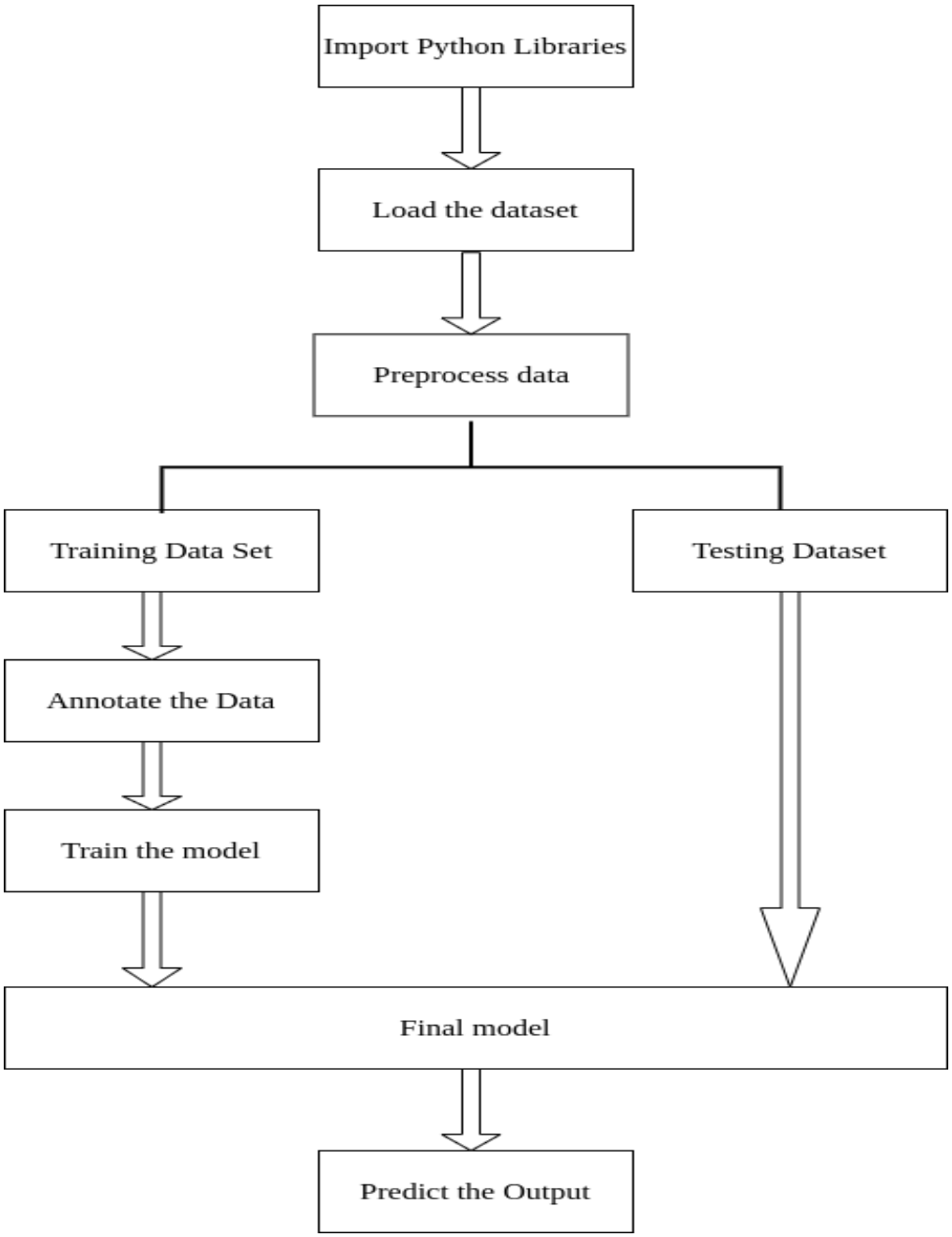

Fig. 4.2.1.1 Data Flow Diagram

4.2.1.2 Flow Chart

A flowchart is diagrammatic representation of an algorithm, workflow or process. There flowchart have certain shape conventions to represent certain operations that are followed during the drawing process. This diagrammatic representation gives us the solution for the given problem. Flowcharts are used in analysing, designing, documenting or managing a process or program in various fields.

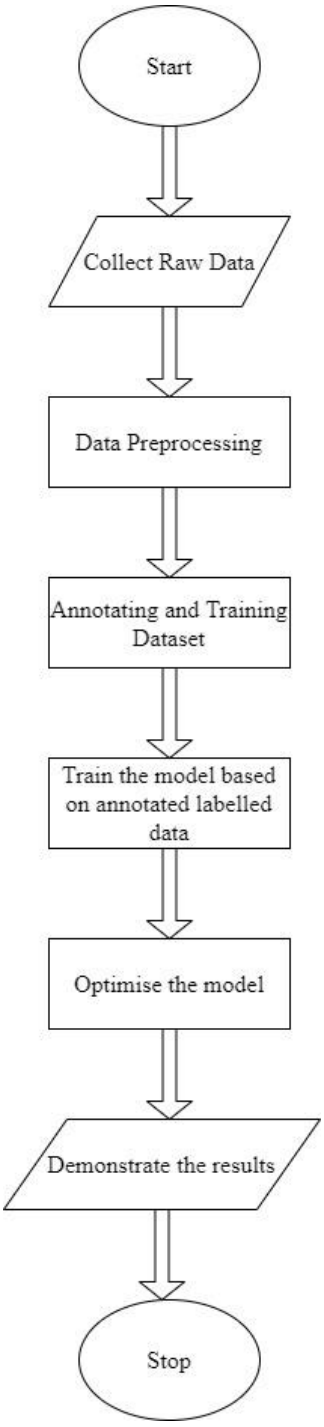

Fig. 4.2.1.2 Flowchart

A flowchart is a graphical representation of steps followed in completing a given task. It originated from computer science as a tool for representing algorithms and programming logic but had extended to use in all other kinds of processes.

Nowadays flowcharts play a major role during the design and assessment process. They help the designers to visualize the complex steps of an algorithm into a simple diagram of the flow chart. The flow chart can be used to illustrate the implementation of the project pictorially.

4.2.1.3 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system. The use case diagram shows the relationship between the user and the different use cases in which the user is involved. The use case diagram is used to capture the dynamic aspects of a system and the functional requirements of a system.

When we are planning to draw a use case diagram, we should have the following items identified.

- Functionalities to be represented as use case
- Actors
- Relationships among the use cases and actors.

Actors can be defined as something that interacts with the system. They can be a human user, some internal applications, or may be some external applications. In this project, user of the system is the actor.

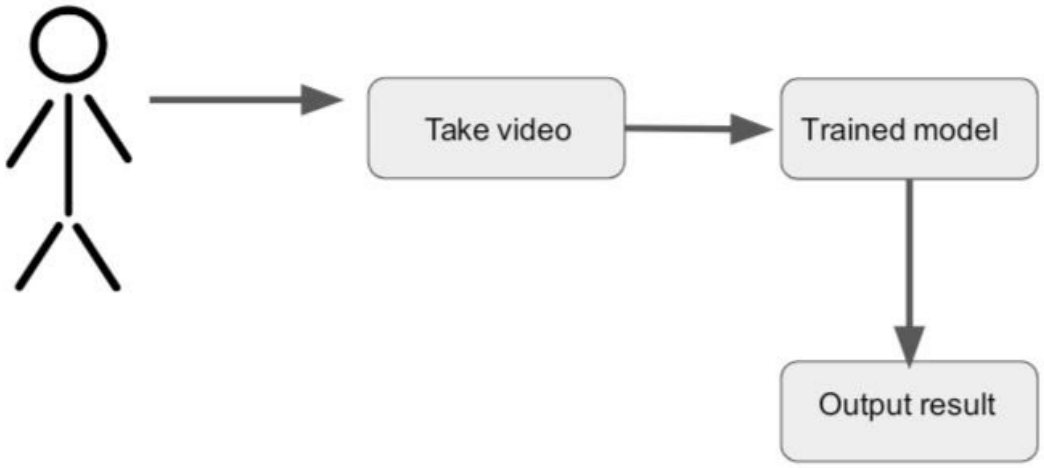

Fig. 4.2.1.3 Use Case Diagram

4.2.1.4 Sequence Diagram

The sequence diagram helps us to depict the various classes and objects and the interactions that occur between them. Based on the given task the implementation may follow different steps and execute the given task. They often exchange messages among each other during this implementation phase to process a given scenario or to achieve a certain functionality.
Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development.

Sequence diagrams show us how and in which order the system uses its objects to complete the given functionality. Sequence diagrams are widely used by developers and can be shown to stakeholders to inform them about the development and functioning of the modules of the programs. They can be used in updating certain modules in the future by understanding the program flow up to and after a given module is processed.

A sequence diagram shows, as parallel vertical lines (*lifelines*) different processes or objects executing concurrently , and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

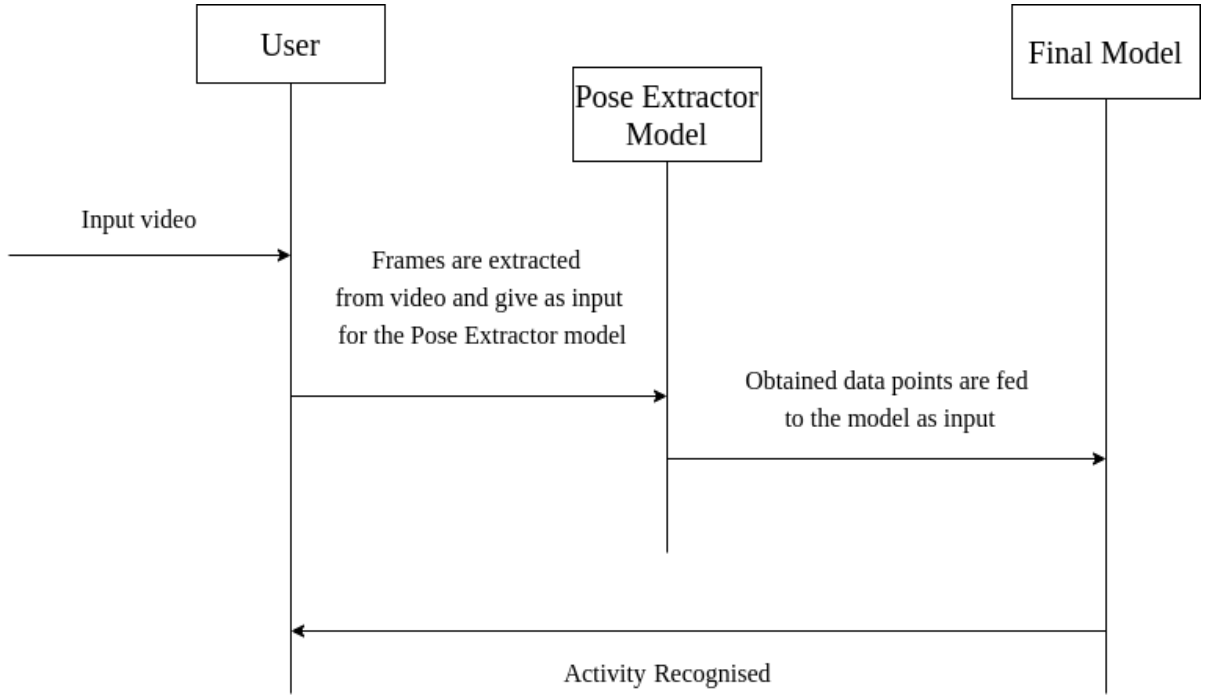

Fig. 4.2.1.4 Sequence Diagram

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

The realization of an application or execution of a plan, idea, model, design, specification, standard, algorithm, or policy is known as implementation. In other words, an implementation is a realization of a technical specification or algorithm as a program, software component or other computer systems through programming and deployment. Many implementations may exist for a given specification or standard.

Implementation is one of the most important phases of the Software Development Life Cycle (SDLC). It encompasses all the processes involved in getting new software or hardware operating properly in its environment, including installation, configuration, running, testing, and making necessary changes. Specifically, it involves coding the system using a particular programming language and transferring the design into an actual working system.

5.2 Implementation of Logistic Regression:

In statistics, the logistic model is used to model the probability of a certain class or event existing for binary classification such as pass/fail, win/lose, alive/dead or healthy/sick.

This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc. Each object being detected in the image would be assigned a probability between 0 and 1 and the sum adding to one.

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an

indicator variable, where the two values are labelled "0" and "1". In the logistic model, the log-odds (the logarithm of the odds) for the value labelled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labelled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labelling; the function that converts log-odds to probability is the logistic function, hence the name.

The unit of measurement for the log-odds scale is called a logit, from logistic units, hence the alternative names. The defining characteristic of the logistic model is that increasing one of the independent variables multiplicatively scales the odds of the given outcome at a constant rate, with each independent variable having its own parameter; for a binary dependent variable this generalizes the odds ratio.

In a binary logistic regression model, the dependent variable has two levels (categorical). Outputs with more than two values are modelled by multinomial logistic regression and, if the multiple categories are ordered, by ordinal logistic regression . The logistic regression model itself simply models probability of output in terms of input and does not perform statistical classification (it is not a classifier), though it can be used to make a classifier, for instance by choosing a cut-off value and classifying inputs with probability greater than the cut-off as one class, below the cut-off as the other; this is a common way to make a binary classifier.

The following activities were used to train the model for detection from video feed:

- Walking
- Hand Waving
- Squats
- Talking on phone
- Standing
- Meditation

If we represent the inputs for training the model as a set of pairs (A, B) where A represents the independent variables and Y represents the action that is being performed corresponding to the independent variables, we can train the model for logistic regression. We can say that we have m data samples to train.

In our model we take the output of data points from the trt_pose human pose estimation model as the input to the independent variables of the binary logistic regression model. These data samples are then labelled as their respective activity.

The B matrix will have nx (total number of data points that are extracted from the trt_pose model) rows and will have columns equal to the number of training examples which is m.

The Y matrix will have a single row and m columns each column representing an activity that can be distinguished by this model.

The data is fed to equation t=Ax+B the linear equation of the line and the sigmoid of this equation is calculated. This is done to always keep value in range of 0 and 1.

This data will be fed into the logistic regression model that uses the logistic function for evaluation.

The logistic function is a sigmoid function, which takes any real input t ($t \in R$)

and outputs a value between zero and one; for the logit, this is interpreted as taking input log-odds and having output probability.

The standard logistic function is defined as follows.

$$\sigma(t) = \frac{e^t}{e^t+1} = \frac{1}{1+e^t}$$

A graph of the logistic function on the *t*-interval (−6,6) is shown in Figure 1.

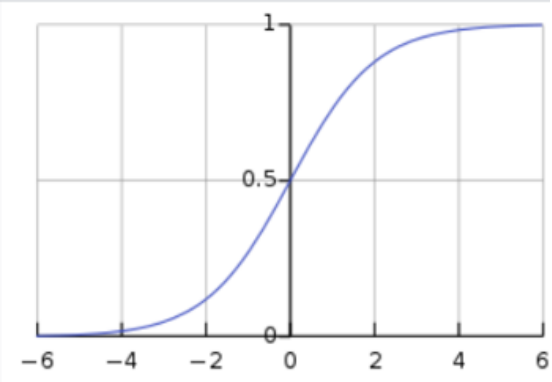

Figure 1. The standard logistic function $\sigma(t)$; note that $\sigma(t) \in (0,1)$ for all t.

Let us assume that *t* is a linear function of a single explanatory variable x (the case where t is a *linear combination* of multiple explanatory variables is treated similarly). We can then express t as follows:

$$t = \beta_0 + \beta_1 x$$

And the general logistic function $p = R \rightarrow (0,1)$ can now be written as

$$p(x) = \sigma(t) = \frac{1}{e^{-(\beta_0 + \beta_1 x)}}$$

R: real numbers

We also define a threshold value. This is the value above with we are certain that a certain activity is being performed and we display that activity as the output of the model.
We have assigned the threshold value to 0.5 or 50 percent confidence in our logistic regression model.

5.3 Overview of System Implementation

5.3.1 System Implementation

System implementation dictates how the information system is built with respect to its physical system design. It ensures that the system remains operational and can be operated at a level that meets the desired quality standard of the projects requirement.

5.3.1.1 Programming Language - Python 3.6

Python was used in the implementation of this project. It is a high level programming language and allows interpreted, interactive object oriented scripting. As it was designed to be human readable it makes us of multiple English words rather than programming keywords. The design philosophy emphasizes code readability. This makes it apt for developing large programs.
Python has a dynamic tape system and an automatic memory management system.It supports many programming paradigms namely the object oriented programming paradigm, functional programming paradigm and the procedural programming paradigm as it has a large set of comprehensive structured libraries.

Python can be run on multiple operating systems. Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications.

5.3.2 Libraries Used

During the development of the project certain pre-trained libraries were employed that make it easier in developing the models. The prominent libraries are mentioned below:

5.3.2.1 ActionAI

ActionAI is a library proposed by python – 3.6. This library is most significantly used to recognize and segregate the human actions by developing Machine Learning algorithms for training a model that identifies different human actions .

5.3.2.2 TensorFlow 2.0

TensorFlow 2.0 is an open-source library that has its significance in development and training of Machine Learning models. It can be implemented on various programming languages. It has a large set of software libraries that handles flow of data across various systems, applications and tasks. One of the biggest implementations of Machine Learning algorithms, i.e.; Neural Networks can be implemented using TensorFlow. We are using 2.0 version of TensorFlow mainly due to additional inbuilt functions available in this version as compared to the previous versions.

TensorFlow has a robust, scalable ecosystem of tools, libraries and community resources that enables researchers to dive in the state of the art in machine learning and the programmers to easily develop and deploy machine-learning driven apps. It mainly concentrates on better usability, proposes high-performance Application programming interfaces, builds a model that is capable enough with changing business needs and user requirements. A situation in which execution time is considered also holds into account TensorFlow's tools and libraries.

Google Brain proposed a second-generation system which is now popular as TensorFlow. It is designed in a way so that it can run on single and multiple applications and is easily portable. The libraries run extremely well on several CPU's and GPU's having optional CUDA and extensions of SYCL which are used for performing basic calculations.

It is compatible with any 64 bit Windows system, MacOS as well as Linux/Unix systems.

With the ease of implementation of TensorFlow, it is used to perform various computations of Machine Learning models and can be operated with a small mobile device to huge performance Supercomputers.

5.3.2.3 OpenCV

Open Source Computer Vision Library commonly known as OpenCV is a collection of programming functions which is prominently used for Real time Computer Vision. As the name states, it is commonly used for capturing video's or images live.

OpenCV was built by Intel and was putforward by Willow Garage then Itseez. Finally after few moths , it was again undertaken by Intel Coorporation. This Library is inder the BSD licence and can be used across several platforms with no cost. It is an open source software library used for capturing and processing video's. It uses the concepts of Computer Vision to process the captured video.

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. It was developed to deliver a shared platform for Computer Vision applications and to promote the use of machine technology in consumer products. Being a BSD Licensed product, OpenCV is capable of updating the code hence making it easy to fullfill various business requirements.

It includes several interfaces like MatLab, Python, Java and C++. It can be implemented on various Operating systems like Linux, MACOS, Windows, and android. The library widely takes the advantages of SSE and MMX instruction for Real-Time computer vision applications. Many interfaces are built and developed using OpenCV such as OpenCL and CUDA. The latest version of OpenCV includes 10 times as many features and functions that compose or support those algorithms. There also also over 500 algorithms available in this library which related to Machine Learning and Computer Vision Applications. OpenCV works efficiently with STL containers and has a templated interface. It is natively written in C++.

Open Source Computer Vision Library contains over 2,500 tailored algorithms, including a detailed collection of both traditional and state-of-the-art computer vision and machine learning algorithms. OpenCV has more than 47 thousand people in the user community and an estimated number of downloads of more than 18 million. The library is widely used by companies, research groups and government agencies.

OpenCV usually uses computer vision libraries in applications including identifying Real-Time objects, Detection and Recognition of faces and Face Expression Detection, To detect actions performed by humans in live video stream, To Track camera Movements, To Track Moving Objects, in Extracting 3D Models for determining objects, to produce 3D point clouds from Stereo cameras, stitch images together in order to deleiver a high resolution image of the entire

screen. Apart from this few noticiable applications include Processing a Database to classify similar images, removal of red eyed images clicked using a flash, to follow eye movements, recognition of several scenery to establish marker to overlac it with augmented reality, etc.

5.3.2.4 Scikit-Learn

Sklearn or Scikit-Learn is a software library associated with Machine Learning Algorithms implemented using Python Programming Language. It was formerly called scikits.learn. Sklearn is designed in association with Python Scientific and Numeric Libraries Numpy and SciPy. Its features includes Random-Forest, SVM's (Support Vector Machines), Gradient Boosting and many more clustering algorithms. It prominently also includes classification, K-Means algorithms, dB scan, Regression algorithms in its latest versions.

Sklearn provides prominent results in Predictive Data Analysis. As it is a open-source free software library, it is flexible to use in any context and accessible by everyone. Also the code written can be modified and reused in order to suite changing business needs. Sklearn can be implemented in association with many Python Libraries such as Pyplot and Matplotlib for plotting, Array Vectorization using Numpy, Extracting dataframes using Pandas,Scipy and many other libraries.

Sklearn is implemented using Python programming language. It makes use of Numpy library for Array Vectorization, Linear Algebra and High-Performance array manipulation operation. In order to further improve performance of the model, the Core Algorithms of Sklearn is written in Cython. Cython wrapper is used to build SVM's (Support Vector Machines). SVM's are build using LIBSVM, logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

5.3.2.5 Pandas

In computer science, pandas is a Python programming language software library for data processing and analysis. In specific, it includes data structures and procedures for the analysis of numerical tables and time series. It is a free software released under a three-class BSD license. It is majorly utilised for Analysis and Manipulation of Data. The name comes from the word "panel data," an econometric word for data sets that contain findings over various time intervals with the same individuals.

Pandas is majorly used in applications where Data is extracted in the form of frames. Our applications takes input in the form of video feed and individual images needs to be extracted from the video. Hence Pandas was the best choice for this operation. It processes data in various formats that includes csv, excel and several other file formats. The Pandas library is implemented using Python programming language and it allows several data manipulation operations such as Group By, join, merge, melt, concatenation as well as data cleaning features such as filling, replacing or imputing null values.

5.3.2.6 NumPy

Numpy is a software library mainly used for Array Vectorization. It is implemented using Python programming language. It supports large multidimensional arrays and matrices. It also includes many Mathematical functions to operate on single and multidimensional arrays. Numpy was originally written as Numeric and was created by Jim Huguin with support from several other Developers. It is known to be as an ancestor of Numpy. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

NumPy is implemented using Python programming language. In more specific, it is the Cpython reference implementation of Python. The Cpython reference is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. With the use of NumPy, this disadvantage was overcomed by providing Mathematical functions applicaple to both single and multidimentional arrays. Numpy provides operators and functions that operate effectively on Array Structures by modifying few segments of code which mainly included inner-outer loops. Python bindings of the widely used OpenCV computer vision library use NumPy arrays to store and operate data. Since multiple-channel images are essentially represented as three-dimensional arrays, The process of Indexing, slicing or masking with other arrays is a very powerful way to view different image pixels. The NumPy collection as a common data structure in OpenCV for images, derived feature points, filter kernels, and much more simplifies workflow and debugging programs. In our application, the keypoints of features obtained from the pose Extraction model is represented in the form of Multidimentional array having parameters – point coordinate, sliding window size and batch size.

CHAPTER 6

PSEUDO CODE

In software engineering, pseudocode is a casual elevated level portrayal of the working rule of a PC program or other calculation. It utilizes the auxiliary shows of a typical programming language however is expected for human perusing as opposed to machine perusing. Pseudocode normally discards subtleties that are basic for machine comprehension of the calculation, for example, factor affirmations, framework explicit code, and a few subroutines. The programming language is increased with common language depiction subtleties, where advantageous, or with conservative numerical notation.urces.

The motivation behind utilizing pseudocode is that it is simpler for individuals to comprehend than ordinary programming language code, and that it is a proficient and condition autonomous depiction of the key standards of a calculation. It is normally utilized in course readings and logical distributions that are recording different calculations, and furthermore in arranging of PC program advancement, for portraying out the structure of the program before the genuine coding happens.

The principle objective of a pseudo-code is to clarify what precisely each line of a program ought to do, subsequently making the code development stage simpler for the software engineer. It goes about as a scaffold between the program and the calculation or flowchart. Additionally, functions as harsh documentation, so the program of one engineer can be seen effectively when pseudocode is worked out. In businesses, the methodology of documentation is basic. Also, that is the place a pseudo-code demonstrates essential.

Steps:

1.All the necessary python libraries are imported based on the model design requirements.

2.The model is trained and loaded into the memory for evaluation.

3.The video to detect output is loaded.

4. Output displayed on top left corner on the screen of chosen video playback.

6.1 extractor.py

```
import numpy as np
from PIL import Image
import tensorflow as tf
from sklearn.base import BaseEstimator, TransformerMixin

class PoseExtractor(BaseEstimator, TransformerMixin):
  def __init__(self, model_path='./models/pose.tflite'):
    self.model_path = model_path
    self.interpreter = tf.lite.Interpreter(model_path=self.model_path)
    self.interpreter.allocate_tensors()

    self.input_details = self.interpreter.get_input_details()
    self.output_details = self.interpreter.get_output_details()
    _, self.input_dim, _, _ = self.input_details[0]['shape']
    _, self.mp_dim, _, self.ky_pt_num = self.output_details[0]['shape']

  def fit(self, x, y=None):
    return self

  def extract(self, x):
    # x is list of image paths or numpy arrays
    feature_array = []
    file_path = True if isinstance(x[0], str) else False
    for img in x:
      # Read the image from file path or numpy array and resize it for model
      image = Image.open(row) if file_path else Image.fromarray(img)

      image=image.resize((self.input_details[0]['shape'][1],self.input_details[0]['shape'][2]),
      Image.NEAREST)
      image = np.expand_dims(np.asarray(image).astype(self.input_details[0]['dtype'])[:, :,
      :3], axis=0)

      # Get pose data from the image
      self.interpreter.set_tensor(self.input_details[0]['index'], image)
      self.interpreter.invoke()
```

```
        results = self.interpreter.get_tensor(self.output_details[0]['index'])

        # Get feature array  from the results
        result = results.reshape(1, self.mp_dim**2, self.ky_pt_num)
        max = np.argmax(result, axis=1)
        coordinates = list(map(lambda x: divmod(x, self.mp_dim), max))
        feature_vector = np.vstack(coordinates).T.reshape(2 * self.ky_pt_num, 1)
        featute_array.append(feature_vector)

    return np.array(feature_array).squeeze()
```

6.2 train.py

```
import pandas as pd
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression as classifier
from extractor import PoseExtractor

def actionModel(classifier):
  pipeline = Pipeline([
        ('pose_extractor', PoseExtractor()),
        ('classifier', classifier)])
  return pipeline

def trainModel(csv_path, pipeline):
  df = pd.read_csv(csv_path)
  X = df['image'].values
  y = df['label']
  pipeline = pipeline.fit(X, y)
  return pipeline.get_params()['steps'][1][1]

if __name__ == '__main__':
  import pickle
  import argparse
  import importlib

  parser = argparse.ArgumentParser(description='Train pose classifier')
  parser.add_argument('--config', type=str, default='conf',
            help="name of config.py file inside config/ directory, default: 'conf'")
  args = parser.parse_args()
  config = importlib.import_module('config.' + args.config)

  pipeline = actionModel(classifier)
  model = trainModel(config.csv_path, pipeline)

  # use pickle to save the model to a file
  pickle.dump(model, open(config.classifier_model, 'wb'), protocol=2)
```

6.3 run.py

```
import cv2
import pickle
import argparse
import importlib
from extractor import PoseExtractor
# import numpy as np

parser = argparse.ArgumentParser(description='Run inference on webcam video')
parser.add_argument('--config', type=str, default='config',
                    help="config.py file inside config/ directory, default: 'config.py'")
args = parser.parse_args()
config = importlib.import_module('config.' + args.config)
 # load the model
model = pickle.load(open(config.classifier_model, 'rb'))
pose_extractor = PoseExtractor()
stream = 0
# stream = '/path/to/video/file'
camera = cv2.VideoCapture(stream)
while(camera.isOpened()):
  ret, frame = camera.read()
  if ret == True:
      frame = cv2.flip(frame,1)
      pose_data = pose_extractor.extract([frame])
      activity = model.predict(pose_data.reshape(1, -1))
      cv2.putText(frame,  activity[0],(0,30),cv2.FONT_HERSHEY_SIMPLEX,1,(255,0,  0),2
,cv2.LINE_AA)
      cv2.imshow('Human Activity Recognition', frame)
      print(activity[0])
     if cv2.waitKey(1) & 0xFF == ord('q'):
         break
  else:
      break
 camera.release()
 cv2.destroyAllWindows()
```

CHAPTER 7

TESTING

Software testing is an examination directed to furnish partners with data about the nature of the product item or administration under test. Programming testing can likewise give a target, free perspective on the product to permit the business to acknowledge and comprehend the dangers of programming usage. Test procedures incorporate the way toward executing a program or application with the purpose of discovering programming bugs (mistakes or different deformities) and checking that the product item is fit for use.

As the quantity of potential tests for even straightforward programming segments is for all intents and purposes interminable, all product testing utilizes some methodology to choose tests that are doable for the accessible time and assets.

7.1 Test Cases

Table 7.1.1: Table for testing *walking* from **run.py**

Test case id	Description	Input video	Expected output	Actual output	Result
1	Walking in room	Person walking 1	Walking	Walking	pass
2	Slow walking on street	Person walking 2	Walking	Walking	pass
3	Hand in pocket while walking	Person walking 3	Walking	Walking	pass
4	Wearing skullcap and jacket	Person walking 4	Walking	Walking	pass
5	Wearing a jacket colored	Person walking 5	Walking	Walking	pass

6	Wearing a jacket black	Person walking 6	Walking	phone	Fail
7	Street cam top view of walking	Person walking 7	Walking	Hand waving	Fail
8	Walking in room with a cap	Person walking 8	Walking	Walking	pass

Table 7.1.2: Table for testing *Meditation* from **run.py**

Test case id	Description	Input video	Expected output	Actual output	Result
1	Beside a pool	Person meditating 1	Meditation	Meditation	pass
2	Rotating recording or subject	Person meditating 2	Meditation	Meditation	pass
3	At home with dog in background	Person meditating 3	Meditation	Meditation	pass
4	On stage in robes with microphones	Person meditating 4	Meditation	Meditation	pass
5	On bridge with bent video feed	Person meditating 5	Meditation	phone	Fail
6	At home with windows behind	Person meditating 6	Meditation	Meditation	pass
7	In studio setting	Person meditating 7	Meditation	Meditation	pass

8	In the garden setting	Person meditating 8	Meditation	Meditation	pass

Table 7.1.3: Table for testing *phone* from **run.py**

Test case id	Description	Input video	Expected output	Actual output	Result
1	Talking with skull cap	Person talking 1	phone	phone	pass
2	Talking with phone behind long hair	Person talking 2	phone	Meditation	Fail
3	Talking on Phone in front of bright lights	Person talking 3	phone	phone	Fail
4	Phone with hand motions	Person talking 4	phone	standing	pass
5	Phone in left hand	Person talking 5	phone	phone	pass
6	Phone in right hand	Person talking 6	phone	phone	pass
7	Talking in a class	Person talking 7	phone	phone	pass
8	Office wired chord phone	Person talking 8	phone	phone	pass

Table 7.1.4: Table for testing Standing from **run.py**

Test case id	Description	Input video	Expected output	Actual output	Result
1	Standing at home	Person standing 1	Standing	Standing	Pass
2	Standing with hand moving	Person standing 2	Standing	Standing	Pass
3	Standing with leg moving	Person standing 3	Standing	Standing	Pass
4	Standing with hand movement 2	Person standing 4	Standing	Standing	Pass
5	Standing with talking	Person standing 5	Standing	Standing	Pass
6	Standing with hand movement 3	Person standing 6	Standing	Standing	Pass
7	Standing on skateboard	Person standing 7	Standing	Handwaving	Fail
8	Standing with objects in background	Person standing 8	Standing	Handwaving	Fail

Table 7.1.5: Table for testing squat from **run.py**

Test case id	Description	Input video	Expected output	Actual output	Result
1	Front view	Person squatting 1	squat	Standing	Fail
2	Front view and side view	Person squatting 1	squat	Squat	Pass

3	Irregular squatting	Person squatting 1	squat	Standing	pass
4	Side view with chair object in background	Person squatting 1	squat	Hand waving	Fail
5	Squatting with a trainer in background 1	Person squatting 1	squat	squat	pass
6	Squatting with a trainer in background 2	Person squatting 1	squat	Squat	Pass
7	Regular squatting	Person squatting 1	squat	Squat	Pass
8	Fast squatting	Person squatting 1	squat	Standing	Fail

Table 7.1.6: Table for testing *Hand waving* from **run.py**

Test case id	Description	Input video	Expected output	Actual output	Result
1	Regular hand waving	Person waving hands 1	Hand waving	Hand waving	Pass
2	Hand waving with clap	Person waving hands 2	Hand waving	Hand waving	Pass
3	Hand waving with crossing hands	Person waving hands 3	Hand waving	Hand waving	Pass
4	Zoom and back motion of camera while waving	Person waving hands 4	Hand waving	Hand waving	Pass

5	Slow hand waving	Person waving hands 5	Hand waving	Hand waving	Pass
6	Hand waving with jacket 1	Person waving hands 6	Hand waving	Hand waving	Pass
7	Fast hand waving	Person waving hands 7	Hand waving	Hand waving	Pass
8	Hand waving with jacket 2	Person waving hands 8	Hand waving	Hand waving	Pass

CHAPTER 8

RESULTS

Once the code has been run and the model has been trained, we add the source file of camera feed that needs to be evaluated for human activity.

Processing of the videos is done at fixed intervals continuously for the entire video and the trained model determines which action is being performed in the video.

The output of human activity is displayed over the playback of the video in real time on the top left corner of the screen. It keeps on changing based on the activity being performed at that instance of the video being played.

8.1 Results for *walking* activity

Fig.8.1 Walking

8.2 Results for *Meditation* activity

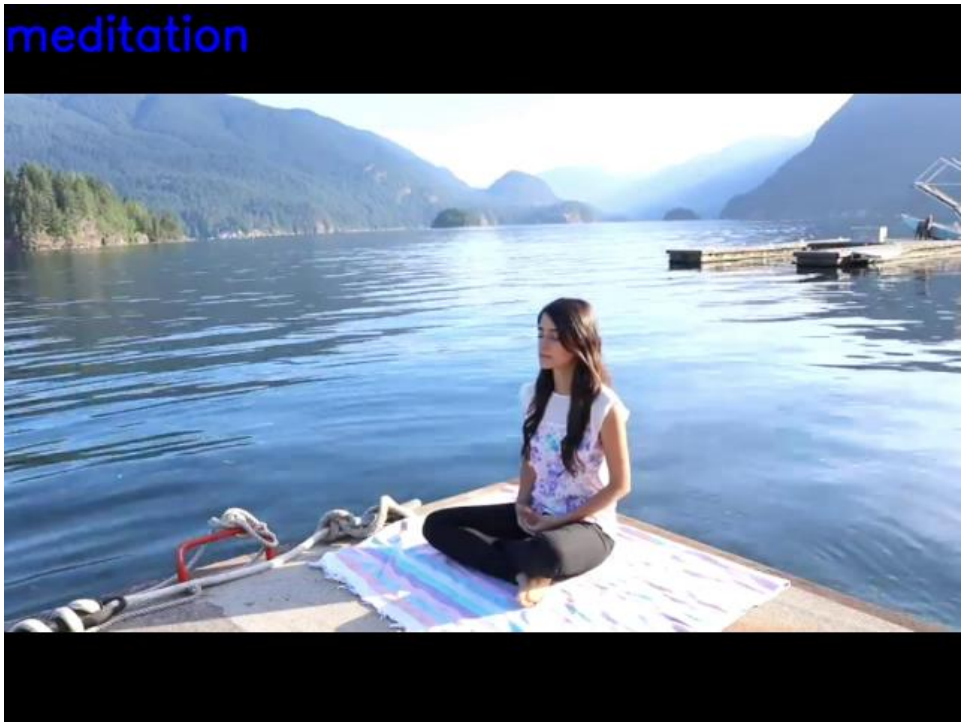

Fig.8.2 Meditation

8.3 Results for *phone* activity

Fig. 8.3 Phone

8.4 Results for *Standing* activity

Fig 8.4 Standing

8.5 Results for *Squat* activity

Fig. 8.5 Squat

8.6 Results for *Hand waving* activity

Fig. 8.6 Hand Waving

CHAPTER 9

CONCLUSION AND FUTURE SCOPE

Human Activity recognition systems are a large field of research and development, currently with a focus on advanced machine learning algorithms, innovations in the field of hardware architecture, and on decreasing the costs of monitoring while increasing safety. We learnt a lot about this developing field of study while researching and developing this project.

During the development of this project to detect human activities from video feeds we came across multiple approaches to help detect a subset of activities. We had limited system hardware resources like GPU for training models and we had only a conservative amount of data for the given set of activities. In future we can do this implementation with greater processing resources and with greater amount of data to increase the accuracy further than what has already obtained.

We made it a point to select activities for which the models had not been trained prior to our implementation and we can propose to add multiple new activities for detection to our currently developed model.

As we all know that the accuracy of the models can never be 100% so the results that are generated will always have a possibility of providing improper output as is the case with all machine learning classifications so if we can increase the data to improve the overall efficiency of detection of human activity we can do so in future models where a huge data set is used to train the developed models.

Other future enhancements can include the use of IOT based smart devices that can perform pre-programmed activities based on actions performed and provide automated solutions to simple problems.

Chapter 10

REFERENCES

Reference Papers:

[1] Erhan BÜLBÜL, Aydın ÇETİN and İbrahim Alper DOĞRU "Human Activity Recognition Using Smartphones"Gazi University Ankara, TURKEY IEEE 2018.

[2] Zhenguo Shi, J. Andrew Zhang, Richard Xu, and Gengfa Fang "Human Activity Recognition Using Deep Learning Networks with Enhanced Channel State information" IEEE 2018.

[3] Anjana Wijekoon, Nirmalie Wiratunga, Kay Cooper "MEx: Multi-modal Exercises Dataset for Human Activity Recognition" arXiv:1908.08992v1 [cs.CV] 13 Aug 2019.

[4] Akbar Dehghani, Tristan Glatard and Emad Shihab "Subject Cross-Validation in Human Activity Recognition" Conference 17, July 2017, Washington, DC, USA.

[5] Stefanie Anna Baby, Bimal Vinod, Chaitanya Chinni, Kaushik Mitra "Dynamic Vision Sensors for Human Activity Recognition" arXiv:1803.04667v1 [cs.CV] 13 Mar 2018.

[6] Gaming Ding*, Jun Tian*, Jinsong Wu**, Qian Zhao*, Lili Xie "Energy-Efficient Human Activity Recognition Using Wearable Sensors" 2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW): IoT-Health 2018: IRACON Workshop on IoT Enabling Technologies in Healthcare.

[7] Artur Jordao, Antonio Carlos Nazare, Jessica Sena, William Robson Schwartz " Human Activity Recognition Based on Wearable Sensor Data: A Standardization of the State-of-the-Art" arXiv:1806.05226v3 [cs.CV] 1 Feb 2019.

[8] Fabien Baradel, Christian Wolf, Julien Mille, Graham W. Taylor ”Glimpse Clouds: Human Activity Recognition from Unstructured Feature Point ” arXiv:1802.07898v4 [cs.CV] 21 Aug 2018.

[9] Sumaira Ghazal, Umar S. Khan “Human Posture Classification Using Skeleton Information”.2018 International Conference on Computing, Mathematics and Engineering Technologies – iCoMET 2018.

[10] Akram Bayat∗ , Marc Pomplun, Duc A. Tran “A Study on Human Activity Recognition Using Accelerometer Data from Smartphones” The 11th International Conference on Mobile Systems and Pervasive Computing (MobiSPC-2014).

[11] Wenchao Xu, Yuxin Pang, Yanqin Yang and Yanbo Liu “Human Activity Recognition Based On Convolutional Neural Network“ sensors." IEEE International Conference on Consumer Electronics-Asia IEEE, 2018.

[12] Shravani Shirish Urankar, Suresh k , Shubham Bhat, Ranjeet Kumar, Madhura J, Dr. Kavitha A “Human Activity Recognition in Video Surveillance– A Survey” International Journal for Research in Applied Science & Engineering Technology (IJRASET) Volume 8 Issue II Feb 2020

Reference Websites

The https://teachablemachine.withgoogle.com/ websites helped us understand many concepts related to Human activity Recognition that helped us to choose appropriate algorithms for the work.

Data has been collected from the following websites:

- https://www.google.com/imghp?hl=en
- https://www.youtube.com/
- https://www.kaggle.com/datasets