HUMAN ACTIVITY RECOGNITION USING DEEP LEARNING

A PROJECT REPORT

Submitted by

KUMARA VAIBHAV C [REGISTER NO:211418104133]

SHABARI RUBAN M [REGISTER NO:211418104241]

SHALIN P SUNIL [REGISTER NO:211418104242]

in partial fulfillment for the award of

the degreeof

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MAY 2022

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"HUMAN ACTIVITY RECOGNITION USING DEEP LEARNING"** the bonafide work of "**KUMARA VAIBHAV C [211418104133], SHABARI RUBAN M [211418104241], SHALIN P SUNIL [211418104242]",** who carried out the project work under my supervision.

SIGNATURE

Dr. S. MURUGAVALLI, M.E.,Ph.D.,
PROFESSOR
HEAD OF DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI – 6000123.

SIGNATURE

Mr. S.A.K. JAINULABUDEEN
SUPERVISOR
ASSISTANT PROFESSOR

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI – 6000123.

Certified that the abovementioned students were examined in End Semester project viva-voice held on _____.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr. P. CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt. C. VIJAYARAJESWARI**, **Dr. C. SAKTHI KUMAR, M.E.,Ph.D** and **Dr. SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr. K. MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of CSE Department, **Dr. S. MURUGAVALLI, M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank our Project Guide **Mr. S.A.K. JAINULABUDEEN** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

ABSTRACT

Human Activity Recognition using smartphone sensors like accelerometers is one of the hot topics of research. It is one of the time series classification challenges. The proposed project does not include any hardware devices, and it is instead based on machine learning models that have been optimised to obtain the best possible results. LRCN (Long-term Recurrent Convolutional Network) can be used to recognise various activities of humans like standing and walking. The LRCN model is a recurrent neural network that can learn order dependency in sequence prediction challenges. This model is used because it helps with recalling values over long periods of time.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

LRCN - Long-term Recurrent Convolutional Network

LSTM - Long Short Term Memory

CNN - Convolutional Neural Network

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

The traditional behaviour model consists of creating a statistical or analytical model of human behaviour and then fitting a distribution to the model to validate it. The deep learning approach differs significantly from the traditional model. By studying human behaviour, a deep learning system learns to predict outcomes. Deep learning is the study of human behavioural features and its application to prediction. Deep learning could be a game changer in commercial applications. The ability to predict and forecast behaviour, as well as calculate the likelihood of winning a game, are valuable economic advantages.

1.2 AIM OF THE PROJECT

The purpose of this study is to see how effective deep learning-based algorithms are at recognising and analysing different elements of human behaviour. However, the majority of our video surveillance systems are still run in the old-fashioned method, with anomalies and evidence gathered solely through offline videos. It's difficult to create real-time alarms, pop-up notifications, and continuously monitor crisis situations.

As a result, real-time human behaviour identification technologies must be developed to reduce security staff workload and increase productivity.

1.3 PROJECT DOMAIN

1.3.1 DEEP LEARNING

Deep learning is a branch of machine learning that is entirely based on artificial neural networks, designed to resemble the human brain. As a result, deep learning can replicate the human brain. Deep learning is a well-established idea, been around for quite a while and we don't have to explicitly code everything. Because the amount of processing power that existed a decade ago is minute when

compared as of today. As processing power has expanded considerably in the previous 20 years, deep learning and machine learning domains have evolved.

A formal definition of deep learning is as follows: Neurons

Deep learning is a sort of machine learning that teaches itself to represent the world as a layered hierarchy of concepts, with each notion defined by simpler concepts and more abstract representations computed in terms of simpler ones. This gives it a lot of strength and versatility.

Each neuron has thousands of connections to its neighbours. The first question that comes to mind is how to replicate neurons in a computer. As a result, an artificial neural network is constructed which is smade up of nodes or neurons. There are neurons for input and output values, and there may be many neurons interconnected in the hidden layer in between.

1.3.2 SOME DEEP LEARNING ARCHITECTURES

Deep Neural Network - A deep neural network is a complicated neural network (having multiple hidden layers in between input and output layers). Non-linear relationships can be modelled and processed by them.

DBNs (Deep Belief Networks) - are a type of Deep Neural Network. It is referred to as a multi-layered belief system.

The DBN procedure is as follows:

a) Using the Contrastive Divergence algorithm, learn a layer of features from visible units.
b) Learn features of features by treating activations of previously taught features as visible units.

c) Once the learning for the final hidden layer is complete, the entire DBN is trained.

Recurrent (perform the same task for every element of a sequence) Neural Network – This allows for parallel and sequential computation. The human brain is similar to this (large feedback network of connected neurons). They can recall key details about the input they got, allowing them to be more exact.

The deep neural network and a combination of both deep neural network and recurrent neural network are the deep learning classes that will be employed in this project.

The first algorithm tested in this project is a Convolutional Neural Network and it is predominantly used for image detection. But in this project instead of one image, the model tests frames of images combined to form a video clip.

The second algorithm in this project is a Long-term Recurrent Convolutional Network, which constitutes of CNN and LSTM models. This algorithm is regarded to be one of the best to be used for video regression.

1.3.3 IMAGE PROCESSING:

Image Acquisition

This is step one or method of the essential steps of virtual photograph processing. Image acquisition may be as easy as being given a photograph this is already in virtual form. Generally, the photograph acquisition level includes pre-processing, which includes scaling, etc.

CHAPTER 2
LITERATURE SURVEY

TITLE 1: Human Activity Recognition Based On Convolutional Neural Network [1]

AUTHORS: Wenchao Xu; Yuxin Pang; Yanqin Yang; Yanbo Liu

DESCRIPTION: Smartphones are ubiquitous and becoming increasingly sophisticated, with ever-growing sensing powers. Recent years, more and more applications of activity recognition based on sensors are developed for routine behavior monitoring and helping the users form a healthy habit. In this field, finding an efficient method of recognizing the physical activities (e.g., sitting, walking, jogging, etc) becomes the pivotal, core and urgent issue. In this study, a Convolutional Neural Network (CNN) is constructed to identify human activities using the data collected from the three-axis accelerometer integrated in users' smartphones. The daily human activities that are chosen to be recognized include walking, jogging, sitting, standing, upstairs and downstairs.

TITLE 2: Human Activity Recognition Using Convolutional Neural Networks [2]

AUTHORS: Gulustan Dogan; Sinem Sena Ertas; İremnaz Cay

DESCRIPTION: Using smartphone sensors to recognize human activity may be advantageous due to the abundant volume of data that can be obtained. In this paper, a sensor data based deep learning approach is proposed for recognizing human activity. The proposed recognition method uses linear accelerometer (LAcc), gyroscope (Gyr), and magnetometer (Mag) sensors to perceive eight transportation and locomotion activities. The eight activities include: Still, Walk, Run, Bike, Bus, Car, Train, and Subway. In this study, the Sussex-Huawei Locomotion (SHL) Dataset of three participants are used to recognize the

physical activities of the users. Fast Fourier Transform (FFT) spectrograms generated from the three axes of the LAcc, Gyr, and Mag sensor data are used as input data for our proposed Convolutional Neural Network (CNN) model.

TITLE 3: Analysis of Human Activity Recognition using Deep Learning [3]

AUTHORS: Lamiyah Khattar; Chinmay Kapoor; Garima Aggarwal

DESCRIPTION: There will be discussion on mainly two models-2-D Convolutional Neural Network and Long-Short term Memory. In order to maintain the consistency and credibility of the survey, both models are trained using the same dataset containing information collected using wearable sensors which was acquired from a public website. They are compared using their accuracy and confusion matrix to check the true and false positives and later the various aspects and fields, where the two models can separately and together be used in the wider field of Human Activity Recognition using image data have been explained. The experimental results signified that both Convolutional Neural Networks and Long-Short term memory model are equally equipped for different situations, yet Long-Short Term memory model mostly appears to be more consistent than Convolutional Neural Networks.

TITLE 4: A Survey on Human Activity Recognition and Classification [4]

AUTHORS: Abhay Gupta; Kuldeep Gupta; Kshama Gupta; Kapil Gupta

DESCRIPTION: Activity Recognition and Classification is one of the most significant issues in the computer vision field. Identifying and recognizing actions or activities that are performed by a person is a primary key goal of intelligent

video systems. Human activity is used in a variety of application areas, from human-computer interaction to surveillance, security, and health monitoring systems. Despite ongoing efforts in the field, activity recognition is still a difficult task in an unrestricted environment and faces many challenges. In this paper, we are focusing on some recent research papers on various methods of activity recognition. The work includes three popular methods of recognizing activity, namely vision-based (using pose estimation), wearable devices, and smartphone sensors. We will also discuss some pros and cons of the above technologies and take a view on a brief comparison between their accuracy.

TITLE 5: Human Activity Recognition Using Smartphones [5]

AUTHORS: Erhan Bulbul; Aydin Cetin; Ibrahim Alper Dogru

DESCRIPTION: Human Activity Recognition (HAR) is classifying activity of a person using responsive sensors that are affected from human movement. Both users and capabilities(sensors) of smartphones increase and users usually carry their smartphone with them. These facts make HAR more important and popular. This work focuses on recognition of human activity using smartphone sensors using different machine learning classification approaches. Data retrieved from smart phones' accelerometer and gyroscope sensors are classified in order to recognize human activity. Results of the approaches used are compared in terms of efficiency and precision.

TITLE 6: Human Activity Recognition System from Different Poses with CNN [6]

AUTHORS: Md. Atikuzzaman; Tarafder Razibur Rahman; Eashita Wazed; Md. Parvez Hossain; Md. Zahidul Islam

DESCRIPTION: In the principle of Human Activity Recognition, a variety of real-life implementations are available using different types of sensors such as fitness monitoring, day-life monitoring, health monitoring, etc. Especially for the elders, sensor-based applications are not feasible due to many reasons such as carrying a mobile phone or gadgets. In this paper, we focused on CCTV videos and camera images to detect human poses using HAAR Feature-based Classifier and recognize the activities of the human using the Convolutional Neural Network (CNN) Classifier. Our Human Activity Recognition System was trained using our own collected dataset which is composed of 5648 images. The approach accomplished an efficacious detection accuracy of 99.86% and recognition accuracy of 99.82% with approximately 22 frames/second after 20 epochs.

TITLE 7: Human Activity Recognition using Deep Neural Network [7]

AUTHORS: Piyush Mishra; Sourankana Dey; Suvro Shankar Ghosh; Dibyendu Bikash Seal; Saptarsi Goswami

DESCRIPTION: The smartphone has become quite ubiquitous and an indispensable part of our lives in the modern day. It has many sensors which capture several minute details pertaining to our activities. So, it is but inevitable that human desire creeps in to augment and improve one's own actions by studying such behaviour captured through the instrumentalities of the smart-phone. In this context, study of data on human activities captured through

accelerometer and gyroscope get primal significance. In this paper, we have attempted to apply machine learning and deep learning techniques on a publicly available dataset. Initially, classification algorithms like K-Nearest Neighbours and Random Forest are applied. The classification accuracies observed are 90.46% and 92.97% respectively.

TITLE 8: A Survey on Human Activity Recognition using Sensors and Deep Learning Methods [8]

AUTHORS: Khushboo Banjarey; Satya Prakash Sahu; Deepak Kumar Dewangan

DESCRIPTION: The main purpose of intelligent video system is to determine the actions and activities of the individual. This action monitoring system can be used in a variety of settings, which include human-computer interaction, tracking, security, and health monitoring. Detecting activity in an uncircumscribed territory remains a challenging task with multiple challenges, despite ongoing efforts in this area. Throughout this article, some recent research papers that include different approaches are analyzed to detect different human activities. Wearable devices and phone sensors are required for this task. The vision-based approach has become a prevalent HAR technique, according to the researchers.

TITLE 9: A CNN-LSTM Approach to Human Activity Recognition [9]

AUTHORS: Ronald Mutegeki; Dong Seog Han

DESCRIPTION: The ability for a system to use as few resources as possible to recognize a user's activity from raw data is what many researchers are striving

for. In this paper, we propose a holistic deep learning-based activity recognition architecture, a convolutional neural network-long short-term memory network (CNN-LSTM). This CNN-LSTM approach not only improves the predictive accuracy of human activities from raw data but also reduces the complexity of the model while eliminating the need for advanced feature engineering. The CNN-LSTM network is both spatially and temporally deep. Our proposed model achieves a 99% accuracy on the iSPL dataset, an internal dataset, and a 92 % accuracy on the UCI HAR public dataset. We also compared its performance against other approaches. It competes favorably against other deep neural network (DNN) architectures that have been proposed in the past and against machine learning models that rely on manually engineered feature datasets.

TITLE 10: Deep Learning Based Real-time Daily Human Activity Recognition and Its Implementation in a Smartphone [10]

AUTHORS: Tsige Tadesse Alemayoh; Jae Hoon Lee; Shingo Okamoto

DESCRIPTION: An iOS application software was developed to record and stream motion data and to recognize real-time activities. The time series data of an accelerometer and gyroscope motion sensors are structured into 14×60 virtual image. Similarly, their respective amplitudes of 1 dimensional DFT (Discrete Fourier Transformation) are organized into 14×60 image format. The resultant data was given to the designed CNN (Convolutional Neural Network) for classification. Both data structuring methods were analyzed and compared yet the time series data structuring showed a better result and attained an accuracy of 99.5%. Additionally, the model was tested for real-time activity recognition in a computer and smartphone and achieved an excellent result.

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing system has low accuracy and low efficiency in terms of loading time and implementation time. Also, the testing and training are not done with the proper test-train split ratio. In the existing system, human activity is predicted by giving an input image which doesn’t give good accuracy.

Current technology of Action recognition systems is mostly dependent on convolutional networks and it is mostly a slow computing algorithm and requires a large amount of computing power to handle video recognition. The current system has to analyze a whole video before saying what action is performed in those videos. These can’t help in detecting actions in a live video feed.

3.2 PROPOSED SYSTEM

The proposed system has a high level of accuracy. When compared to the existing system, the suggested system's loading and execution times are rapid. The suggested system is highly efficient and scalable, and it may be enhanced for more complex use cases.

With few future enhancements, this proposed system can predict outputs from live video feeds like for example live feeds from a CCTV. As this system predicts action on every frame of a video this system can be used in surveillance programs and supervision programs.

This proposed system makes use of the LRCN algorithm which is a combination of CNN and LSTM. This is a more modern way of handling the problem and it is promised to perform fast with high accuracy and fewer computing resources.

3.3 REQUIREMENT SPECIFICATION

3.3.1 HARDWARE REQUIREMENTS

Processor : Core i3 8th Gen – 3.0 Ghz

Hard disk : 15 GB

RAM : 4 GB (minimum)

Keyboard : 110 keys enhanced

3.3.2 SOFTWARE REQUIREMENTS

Operating system : Windows 10/11

IDE : Anaconda / Google Colab

Language : Python

3.4 PLATFORM SPECIFICATION – ANACONDA

Anaconda is an open-source package manager for Python and R. It is the most popular platform among data science professionals for running Python and R implementations. There are over 300 libraries in data science, so having a robust distribution system for them is a must for any professional in this field. Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms. With Anaconda, you can easily set up, manage, and share Conda environments. Moreover, you can deploy any required project with a few clicks when you're using Anaconda. There are many advantages to using Anaconda and the following are the most prominent ones among them: Anaconda is free and open-source. This means you can use it without spending any money. In the data science sector, Anaconda is an industry staple. It is open-source too, which has made it widely popular. If you want to become a data science professional, you must know how to use Anaconda for Python because every recruiter expects you to have this skill. It is a must-have for data science.

It has more than 1500 Python and R data science packages, so you don't face any compatibility issues while collaborating with others. For example, suppose your colleague sends you a project which requires packages called A and B but you only have package A. Without having package B, you wouldn't be able to run the project. Anaconda mitigates the chances of such errors. You can easily collaborate on projects without worrying about any compatibility issues. It gives you a seamless environment that simplifies deploying projects. You can deploy any project with just a few clicks and commands while managing the rest. Anaconda has a thriving community of data scientists and machine learning professionals who use it regularly. If you encounter an issue, chances are, the community has already answered the same. On the other hand, you can also ask people in the community about the issues you face there, it's a very helpful community ready to help new learners. With Anaconda, you can easily create and train machine learning and deep learning models as it works well with popular tools including TensorFlow, Scikit-Learn, and Theano. You can create visualizations by using Bokeh, Holoviews, Matplotlib, and Datashader while using Anaconda.

How to Use Anaconda for Python

Now that we have discussed all the basics in our Python Anaconda tutorial, let's discuss some fundamental commands you can use to start using this package manager.

Listing All Environments

To begin using Anaconda, you'd need to see how many Conda environments are present in your machine.

conda env list

It will list all the available Conda environments in your machine.

Creating a New Environment

You can create a new Conda environment by going to the required directory and

use this command:

conda create -n <your_environment_name>

You can replace <your_environment_name> with the name of your environment. After entering this command, conda will ask you if you want to proceed to which you should reply with y:

proceed ([y])/n)?

On the other hand, if you want to create an environment with a particular version of Python, you should use the following command:

conda create -n <your_environment_name> python=3.6

Similarly, if you want to create an environment with a particular package, you can use the following command:

conda create -n <your_environment_name>pack_name

Here, you can replace pack_name with the name of the package you want to use. If you have a .yml file, you can use the following command to create a new Conda environment based on that file:

conda env create -n <your_environment_name> -f <file_name>.yml

We have also discussed how you can export an existing Conda environment to a .yml file later in this article.

Activating an Environment

You can activate a Conda environment by using the following command:

conda activate <environment_name>

You should activate the environment before you start working on the same. Also, replace the term <environment_name> with the environment name you want to activate. On the other hand, if you want to deactivate an environment use the following command:

conda deactivate

Installing Packages in an Environment

Now that you have an activated environment, you can install packages into it by using the following command:

conda install <pack_name>

Replace the term <pack_name> with the name of the package you want to install in your Conda environment while using this command.

Updating Packages in an Environment

If you want to update the packages present in a particular Conda environment, you should use the following command:

conda update

The above command will update all the packages present in the environment. However, if you want to update a package to a certain version, you will need to use the following command:

conda install <package_name>=<version>

Exporting an Environment Configuration

Suppose you want to share your project with someone else (colleague, friend, etc.). While you can share the directory on Github, it would have many Python packages, making the transfer process very challenging. Instead of that, you can create an environment configuration .yml file and share it with that person. Now, they can create an environment like your one by using the .yml file.

For exporting the environment to the .yml file, you'll first have to activate the same and run the following command:

conda env export ><file_name>.yml

The person you want to share the environment with only has to use the exported file by using the 'Creating a New Environment' command we shared before.

Removing a Package from an Environment

If you want to uninstall a package from a specific Conda environment, use the following command:

conda remove -n <env_name><package_name>

On the other hand, if you want to uninstall a package from an activated environment, you'd have to use the following command:

conda remove <package_name>

Deleting an Environment

Sometimes, you don't need to add a new environment but remove one. In such cases, you must know how to delete a Conda environment, which you can do so by using the following command:

conda env remove –name <env_name>

The above command would delete the Conda environment right away.

3.4 PLATFORM SPECIFICATION – GOOGLE COLAB

Google Colabs are similar to Jupyter ntoebooks. Because they are hosted by Google Colab, any of our own computer resources is not required to run the notebook. These notebooks can also be distributed so that others can run our code in a standard environment without having them to rely on our local devices. During initialization, some libraries might need to be installed in our environment.

You can go to Google Drive and create a Google Drive account if you don't already have one to build your Google Colab file and get started with Google Colab. Now, at the top left corner of your Google Drive page, select the "New" option, then More Google Colaboratory.

Markdown cells, like Jupyter notebooks, can be used to write text. However,

Colab offers a function that creates a table of contents based on your markdown content and allows you to conceal parts of the code based on their headers in the markdown cells.

You have no choice but to use your computer's CPU if you run Jupyter on your own machine. Because Colab is operated on Google's cloud, you can adjust the runtime to include GPUs and TPUs in addition to CPUs. By navigating to Runtime, you can select a different runtime and alter it.

CHAPTER 4
SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

There are two modules to this project. The first module discusses how the deep learning models are trained, and the second model explains how the actual process of predicting the action in a video is carried out.

Training Module:

The dataset employed in this project is named UCF - 50, and it has 50 action classes, each of which contains clips describing those videos.

The clips are then pre-processed and labeled to serve as training data sets for the two models being used.

Convolutional Neural Network (CNN) and Long Recurrent Convolutional Network (LRCN) are the two models employed here.

The accuracy of these models is tested once they have been trained. The model with the highest accuracy is considered as best performing model.

Activity Prediction Module:

The user provides the video input, which is then converted into several frames of images. After that, each frame is input into the best-performing model, which produces an output video that shows the action performed in each frame of the clip.

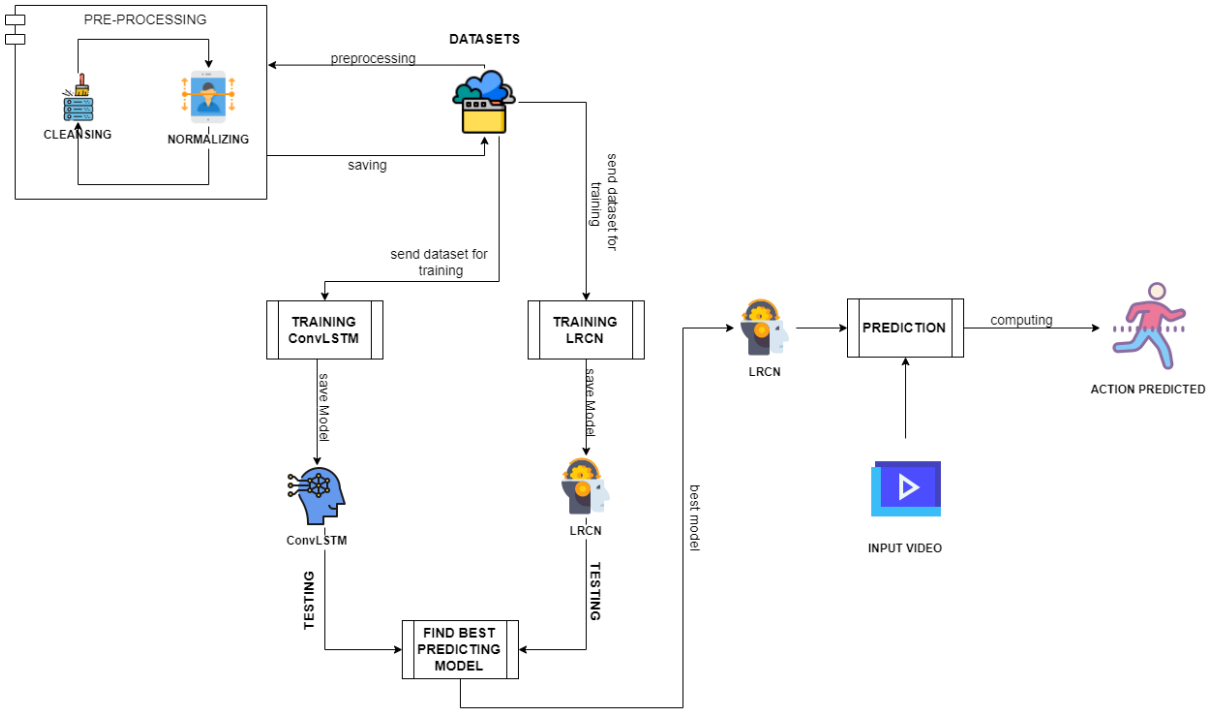


Figure 4.1 – Architecture Diagram

4.2 USE-CASE DIAGRAM

There are three characters in this project: user, algorithm, and dataset. The dataset is in charge of all of the data that will be used in the training of the two models. Model construction is handled by algorithms. Finally, it is the user's responsibility to provide the best-performing model with input.

The data is initially preprocessed. Every video clip from the datasets is shrunk into a common size and supplied into the model for training purposes, making it pre-processed.

Algorithms ensure that the model is built correctly, that it is tested, and that it is retrained in order to improve its accuracy. Finally, the model is evaluated, and the best performing model is selected and utilized to predict activity.

The user provides a video clip as input, and the model predicts the activity that is being performed in the video clip based on the training data. The output precision is guaranteed to be 98 percent accurate.

USE CASE DIAGRAM

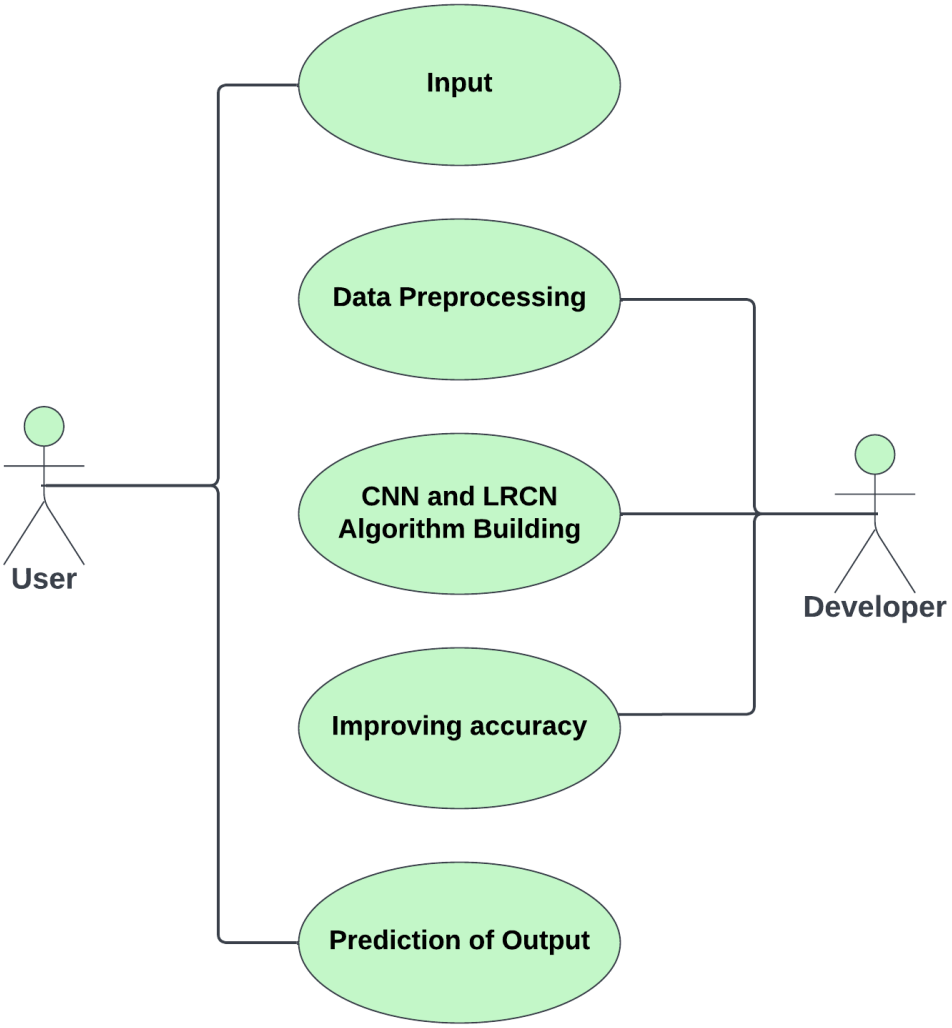


Figure 4.2 – Use-Case Diagram

4.3 ACTIVITY DIAGRAM

Preprocessing of the data is done first. The type of dataset that is dealt with here is video clips. A random frame is selected and labeled from each video clip. These labels help the model identify the actions performed.

In the data cleaning process, video clips with larger frames are reduced in size, and some 30 - 100 random frames from that clips are chosen as the representative frames for their respective video.

Two models are built, one for CNN and one for LRCN. Next, the model is trained with all the processed datasets to improve its accuracy. Finally, the accuracy of those models is checked.

The model with the best accuracy is taken for this project. The user gives the input video of his own and the model predicts the action performed in it with 98% accuracy.

ACTIVITY DIAGRAM

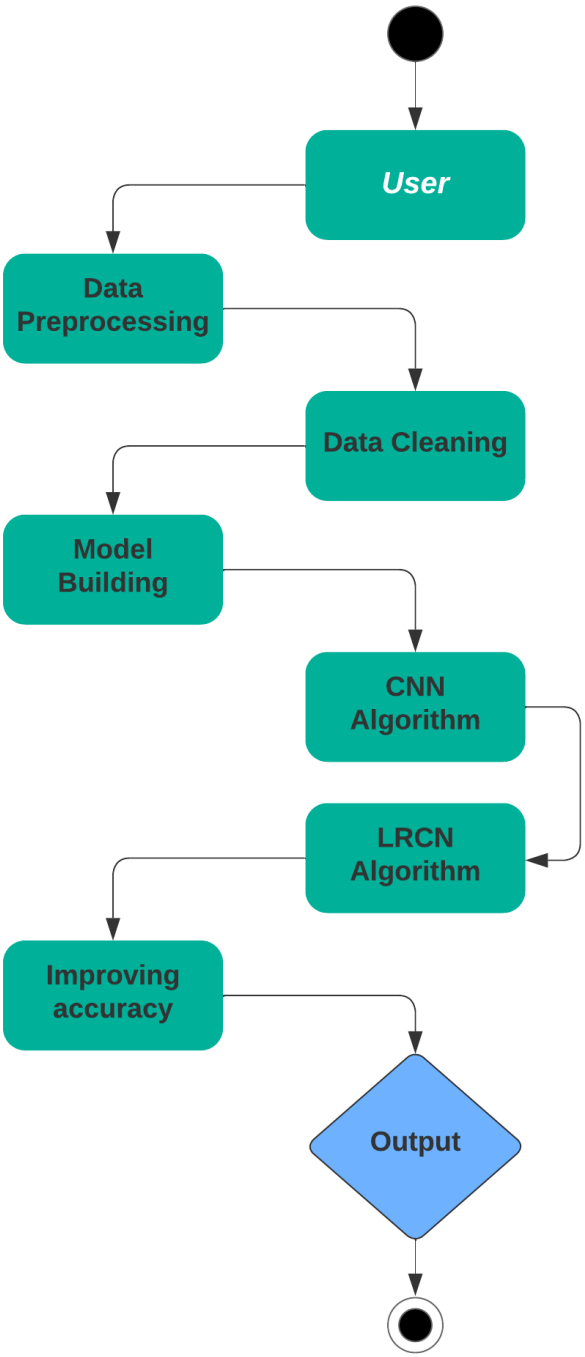


Figure 4.3 – Activity Diagram

4.4 SEQUENCE DIAGRAM

Step 1: The first step in this project is acquiring all the UCF 50 datasets and pre-processing them. Pre-processing in the sense the datasets are modified to suit the training models that are used.

Step 2: This project makes use of two algorithms and for that, both CNN and LRCN algorithms should be built first.

Step 3: The next step in this project is to train the models. With known and available information, the model is trained. Only then a model can predict action in future scenarios. For this purpose, the model is trained with all the processed datasets.

The first three steps are done only for the first execution.

Step 4: The user provides an input video clip which should be short because long videos mean the system should make use of large resources and more time.

Step 5: After analysing and computing the system predicts the final output. for prediction out of the two algorithms, the best performing one is used.

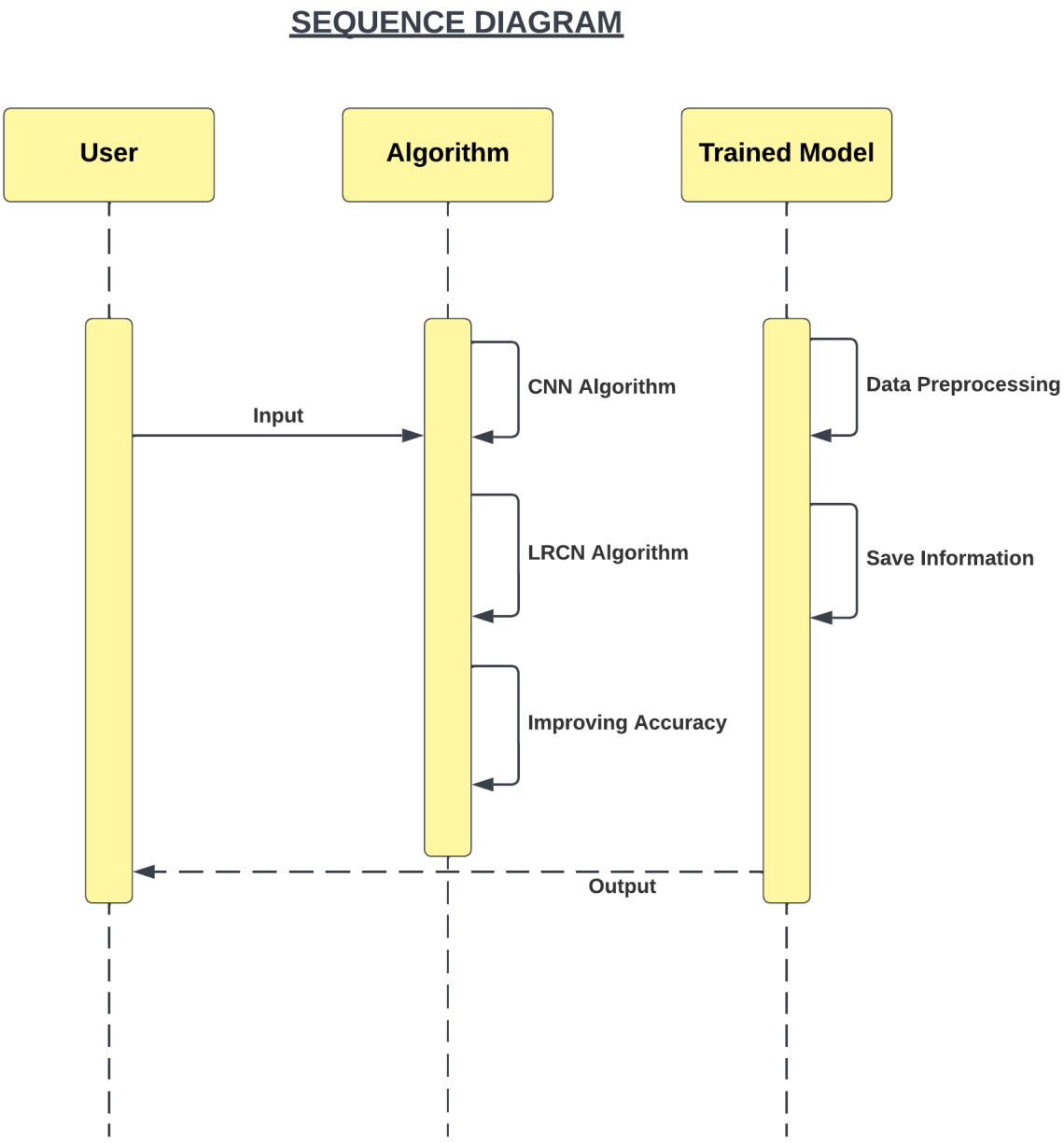


Figure 4.4 – Sequence Diagram

4.5 DATA FLOW DIAGRAM LEVEL 0

A Level 0 Data Flow Diagram is also referred to as a context diagram. It represents the basic overview of the whole system showing the relationship between the entities present. Its purpose is to serve as an at-a-glance view of the system that can be easily understood by the viewer.

The basic skeleton of the proposed system is shown below. We have three main components – User, Human Activity Prediction System, CNN and LRCN Predictor. The user provides the dataset as input to the Human Activity Prediction system, which in turn passes it through the CNN and LRCN models and obtains the output. The output being, the activity performed in the video.

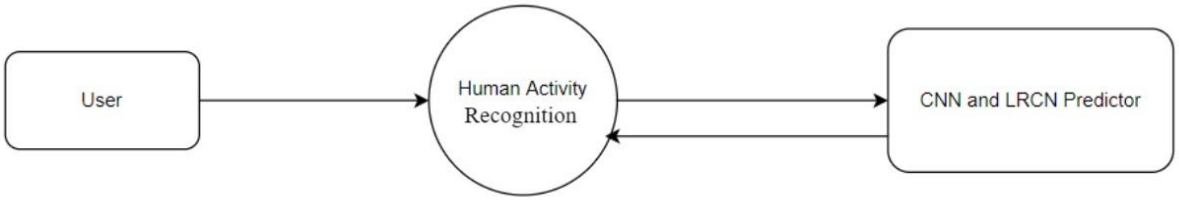


Figure 4.5 – Data Flow Diagram 0

4.6 DATA FLOW DIAGRAM LEVEL 1

The Level 1 diagram presents a more elaborate diagram where the basic components are broken down into detailed processes. The relationship between the entities are also shown in a detailed manner.

The diagram shown below represents the Level 1 DFD diagram. Here, input provided by the user is shown at the start. The input is taken into the data pre-processing phase, every dataset needs to cleaned and filtered in order to get the desired input we require. Next, the process of building the CNN model as well as the LRCN model. After that, the dataset taken as input is fed into both the models and trained. The training is a vital process since it helps in improving the accuracy of the neural network models. Prediction of the output takes place and is sent to the user.

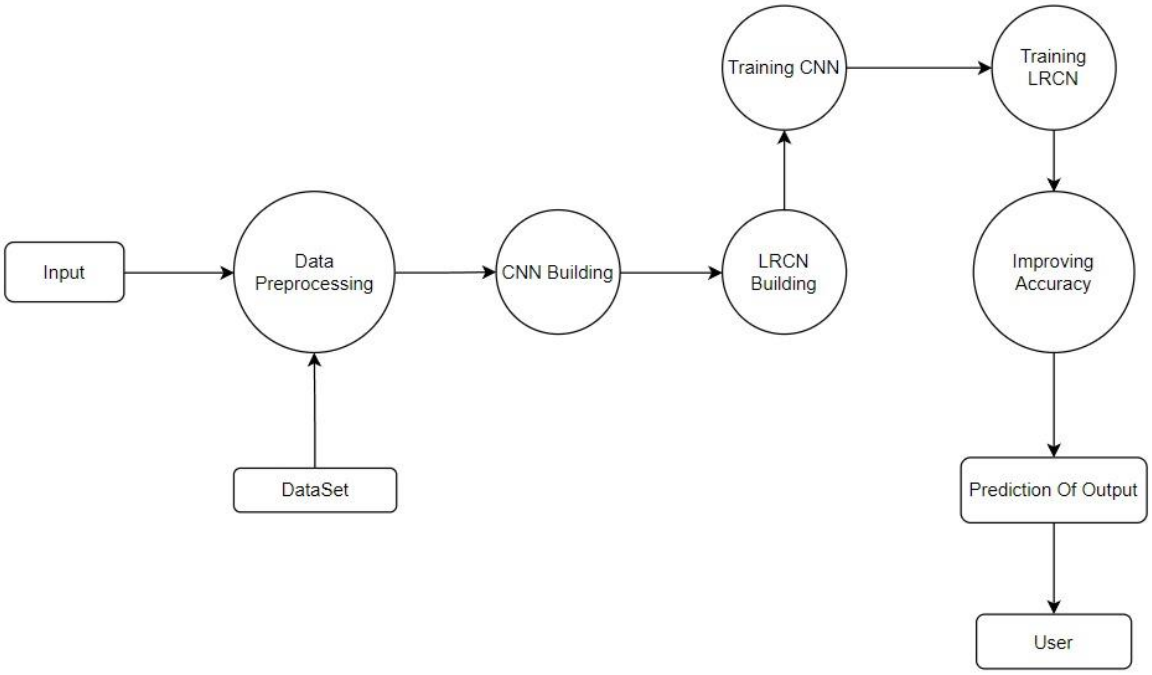


Figure 4.6 – Data Flow Diagram 1

CHAPTER 5
MODULE DESCRIPTION

5.1 DATA PRE-PROCESSING

One of the most significant steps in any deep learning project is data cleaning. The process begins with preprocessing of the given data from a large dataset. The data is cleaned and pre-processed at this stage, where missing and null value records are dropped. In our dataset, we cleaned all the null values and checked whether all the data types are valid. The main purpose of preprocessing is to identify and drop or substitute the missing values in the dataset which occupy a very small part of the whole data, to ensure an accurate result. Data Cleaning such as removing clutter and unnecessary punctuation would be taken off, Feature Engineering would be performed for enhancement. The data leveraged is encoded, scaled sorted if needed, and then transformed into accessible format.

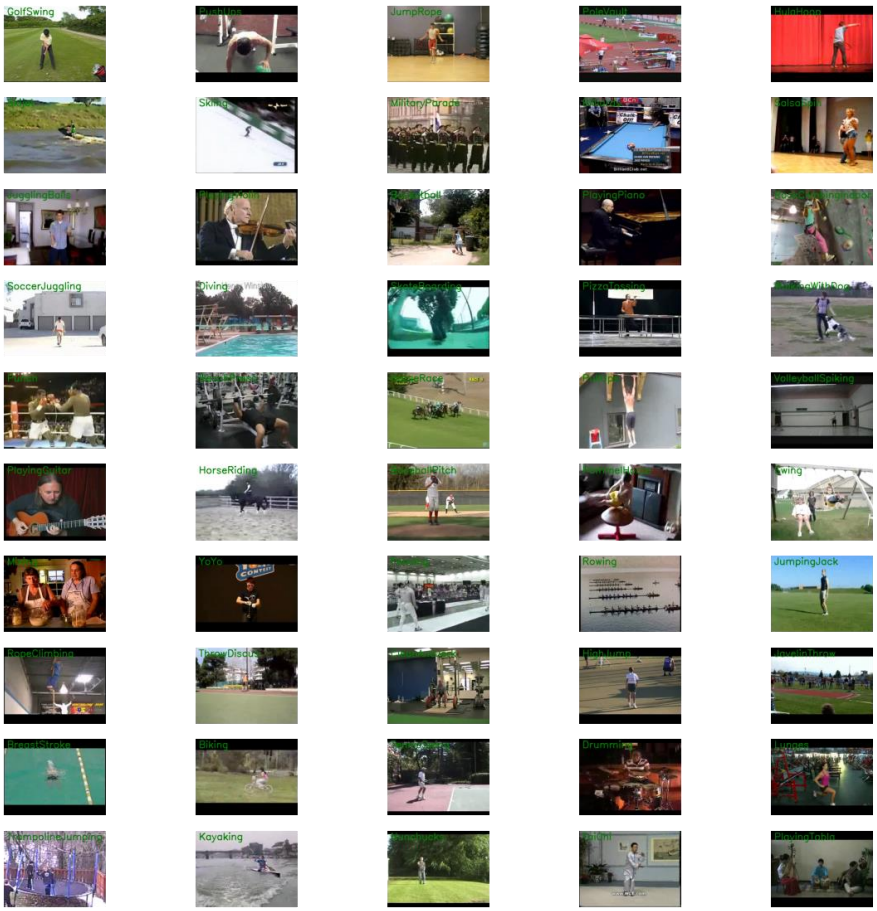


Figure 5.1 – 50 Classes of actions

5.2 CNN AND LRCN IMPLEMENTATION

5.2.1 CNN Model

A combination of ConvLSTM cells is used to implement the first strategy in this stage. A ConvLSTM cell is a type of LSTM cell that includes convolutional operations in the network. It is an LSTM with convolution included into the architecture, allowing it to recognize spatial characteristics of input while taking into consideration temporal relationships.

This method efficiently captures the spatial relationship between individual frames as well as the temporal relationship between distinct frames when it comes to video classification. The ConvLSTM can take in 3-dimensional input (width, height, num of channels) as a result of this convolution structure, whereas a standard LSTM can only take in 1-dimensional input, making it unsuitable for modelling Spatio-temporal data on its own.

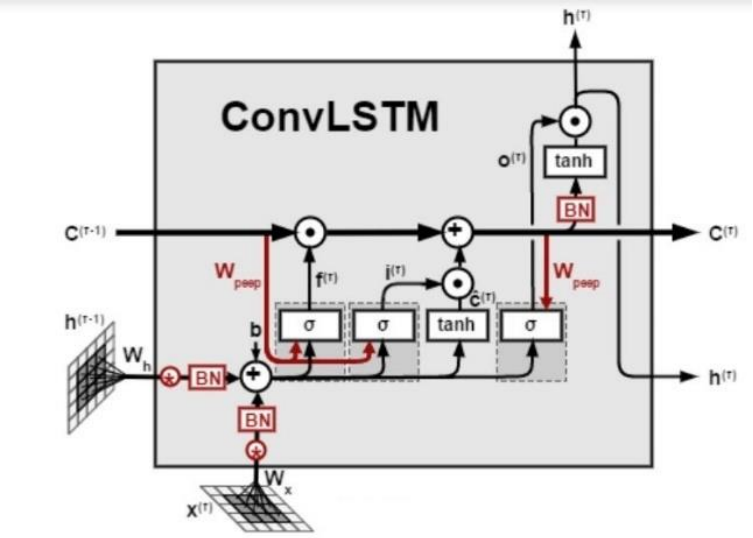
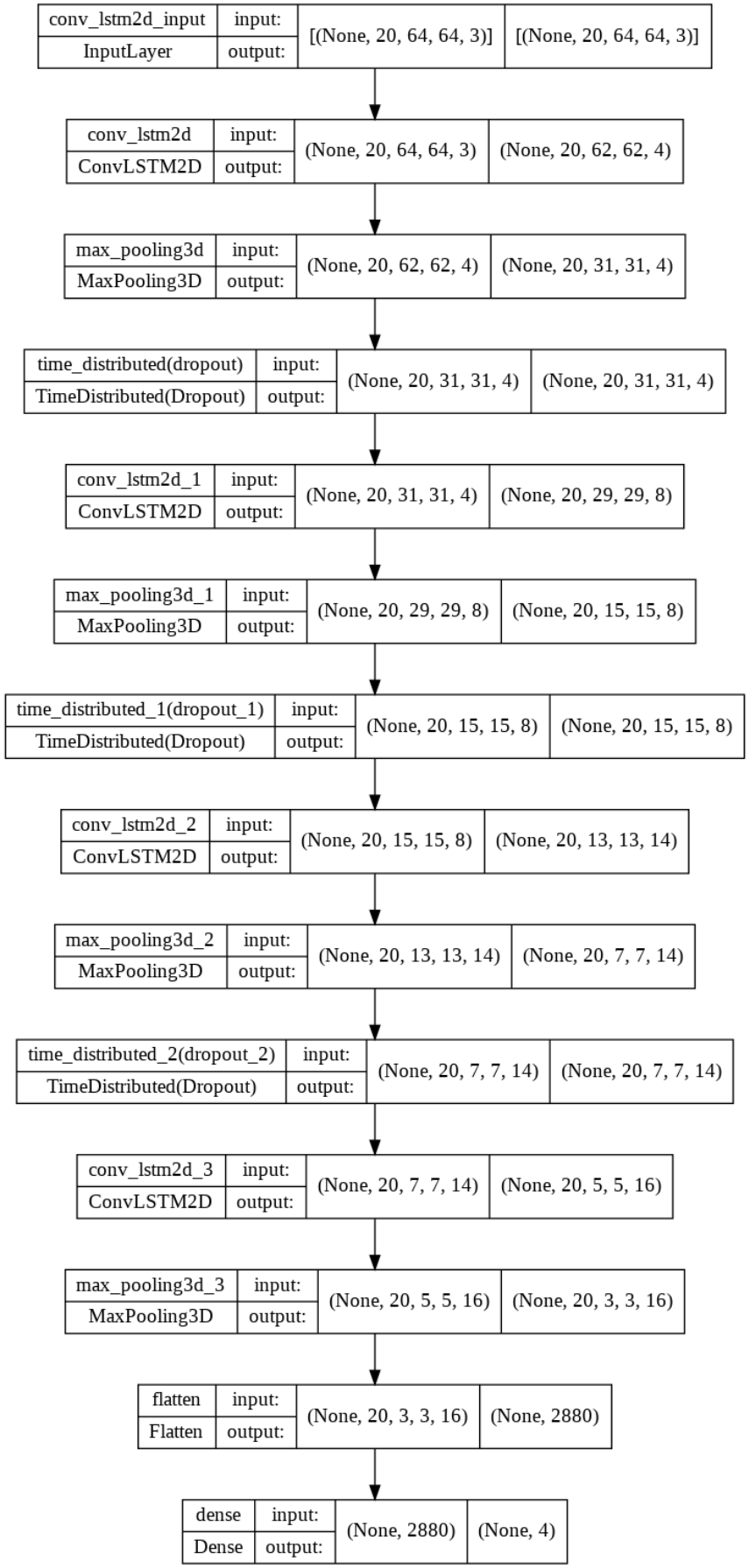


Figure 5.2.1 ConvLSTM

Structure of CNN Model:



Analysis of Trained Model:

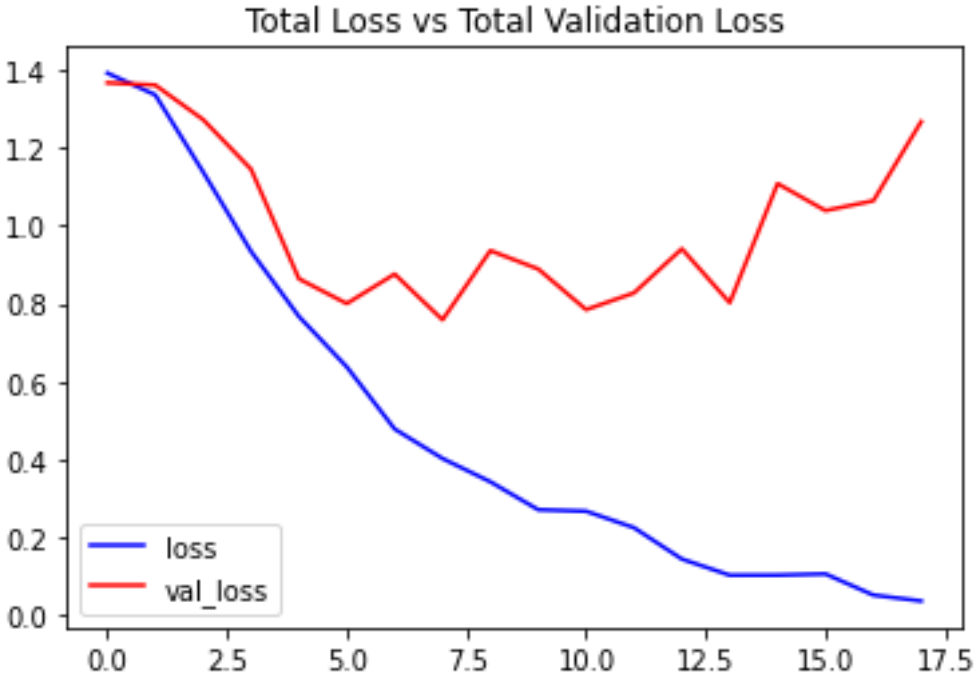


Figure 5.2.2 Total Loss Vs Total Validation Loss Metrics

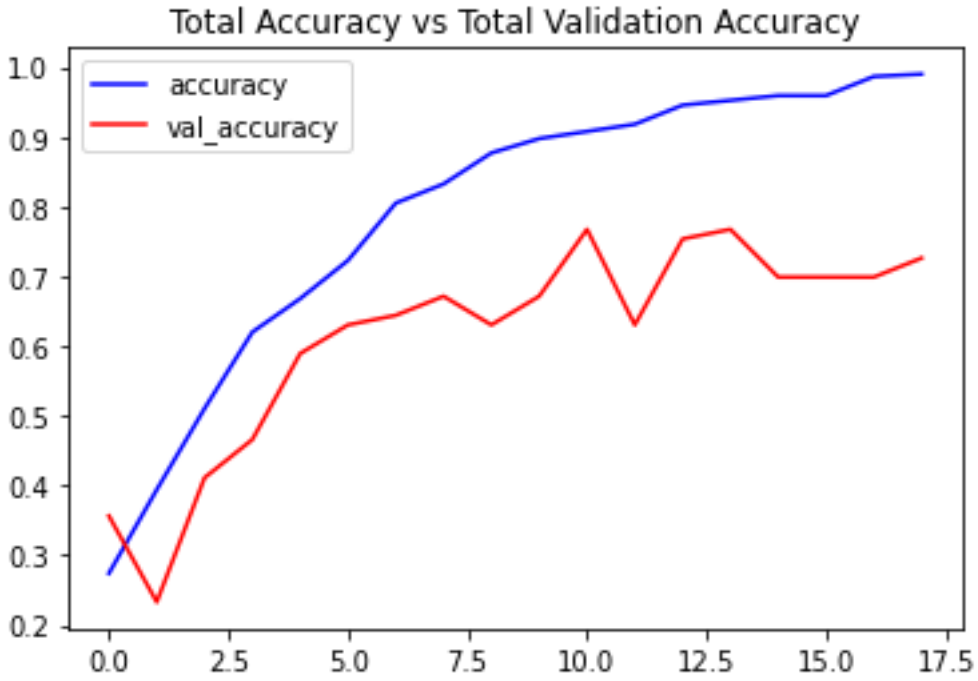


Figure 5.2.3 Total Accuracy Vs Validation Accuracy Metrics

5.2.2 LRCN MODEL

In this stage, the proposed system combines Convolution and LSTM layers in a single model to apply the LRCN Approach. A CNN model and an LSTM model trained separately can be used in a similar way. The CNN model may be used to extract spatial information from video frames, and a pre-trained model that can be fine-tuned for the application can be utilized for this. The CNN features can then be used by the LSTM model to predict the action being performed in the video.

But, in this case, Long-term Recurrent Convolutional Network (LRCN) will be used, which integrates CNN and LSTM layers into a single model. The Convolutional layers are utilized to extract spatial characteristics from the frames, and the retrieved spatial features are given to the LSTM layer(s) for temporal sequence modelling at each time-step. This method allows the network to learn spatiotemporal features in an end-to-end training, resulting in a robust model.

A Time Distributed wrapper layer will also be employed, which allows us to apply the same layer to each frame of the movie separately. As a result, if the layer's input shape was (width, height, num of channels), it may now take input of shape (no of frames, width, height, num of channels), which is highly useful because it allows you to input the entire movie into the model in a single shot.

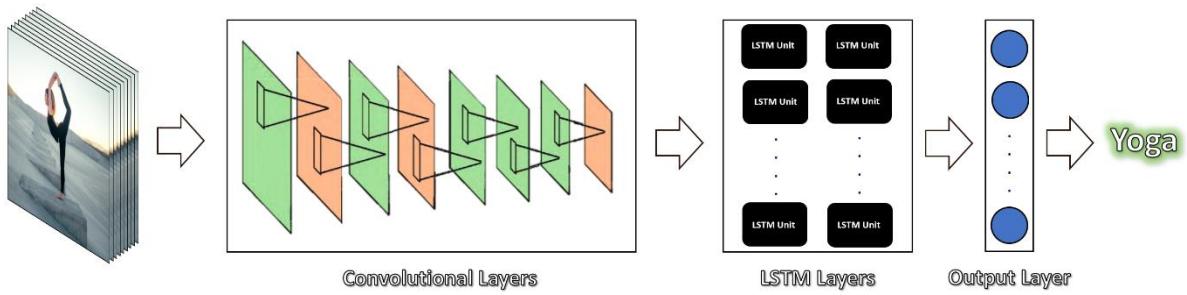
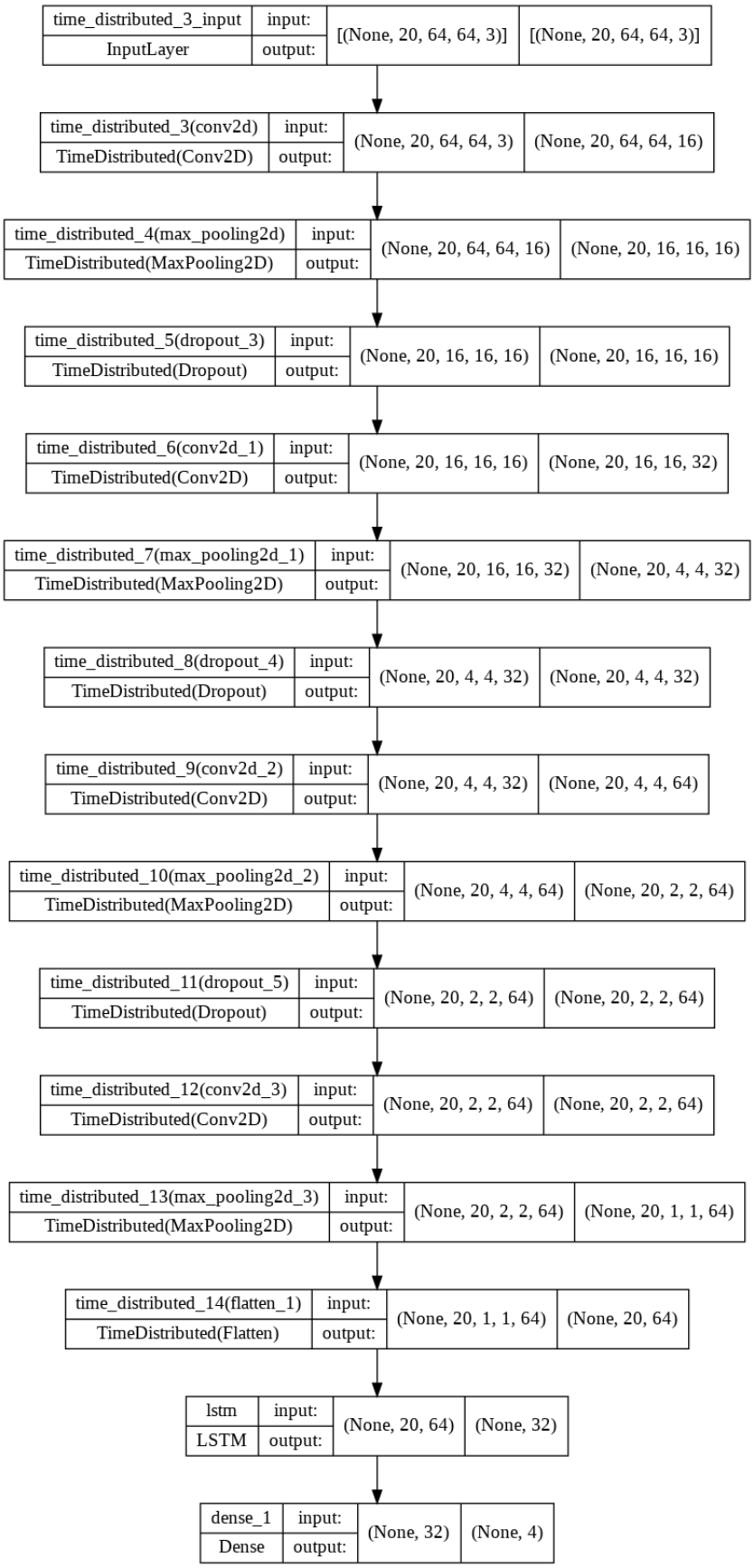


Figure 5.2.4 LRCN Model

Structure of LRCN Model:



Analysis of Trained Model

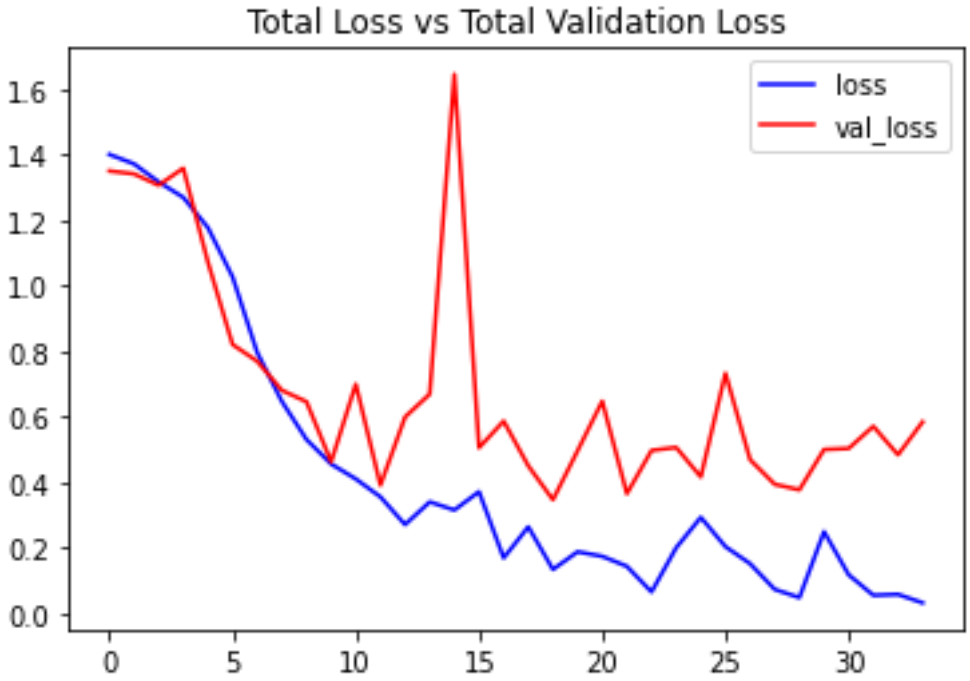


Figure 5.2.5 Total Loss Vs Total Validation Loss Metrics

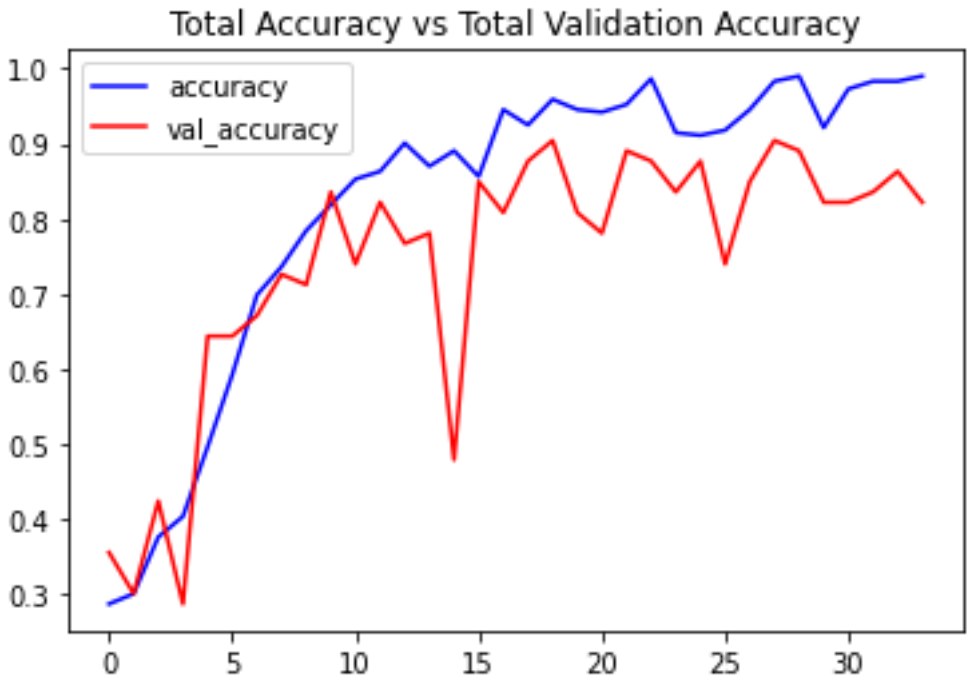


Figure 5.2.6 Total Accuracy Vs Validation Accuracy Metrics

5.3 RESULTS AND FINDINGS:

After carefully inserting the processed dataset of the action classes, the CNN model as well as the LRCN have been trained to some extent. Considering the motive behind this proposed system, which being to portray a comparison between a CNN model and LRCN model, the objective has been achieved at the completion of this project. LRCN is constructed as a combination of Convolutional Neural Networks (CNNs) and Long Short Term Memory (LSTM), which helps to achieve video regression/activity recognition much efficiently when compared to a standard CNN model.

Given both the models have been trained for a significant amount of time using the dataset, the human activity recognition has proved to be effective. In the case of the CNN model, it has achieved an 81.15% accuracy. Whereas, the LRCN model was able to achieve a 93% accuracy.

The amount of processing power has increased all the way over the years. The quantity of data accumulated along the way is beyond imaginable when compared to a few decades ago. The more data used to train the Neural Networks, the more efficiently and accurately the models can detect the human activity.

CHAPTER 6
TESTING

6.1 SYSTEM TESTING

Errors are discovered during testing. It's utilized for quality control. Testing is an important aspect of the creation and maintenance of software. The purpose of testing at this phase is to check that the specification has been accurately and thoroughly included into the design, as well as the design's correctness. For example, any logic flaws in the design must be recognized before coding begins; otherwise, the cost of resolving the flaws will be significantly higher, as reflected. Inspection and a walkthrough are both effective methods for detecting design flaws.

Testing is the process of running a software in order to detect errors. An excellent test case is one that has a high chance of uncovering an error that has yet to be detected. A successful test is one that uncovers a previously unknown flaw. System testing is a stage of implementation focused at ensuring that the system performs as planned before going live. It ensures that the entire set of programs run smoothly. System testing is required for the effective implementation of a new system and consists of numerous key actions and procedures for running a program, string, and system.

Testing is an essential part of the software development process. Testing checks for mistakes and includes the following test cases as part of the overall project testing:

- Static analysis is used to look into the source code's structural qualities.
- Dynamic testing is a technique for investigating the behavior of source code by running it on test data.

Test Case No.	Action	Expected Output	Actual Output	Result (Pass/Fail)
TC_ID_001	Prediction on a Swinging video	Swing	Swing	Pass
TC_ID_002	Prediction on a video of walking a dog	Walking with dog	Walking with dog	Pass
TC_ID_003	Prediction on a Tai Chi video	Tai Chi	Tai Chi	Pass
TC_ID_004	Prediction on a Soccer Juggling video	Soccer juggling	Soccer juggling	Pass
TC_ID_005	Prediction on a Cycling video	Cycling	Swing	Fail
TC_ID_006	Prediction on a Swinging video	Swing	Soccer juggling	Fail
TC_ID_007	Prediction on a Cycling video	Cycling	Cycling	Pass
TC_ID_008	Prediction on a video of Pushups	Pushups	Pushups	Pass
TC_ID_009	Prediction on a Horse Racing video	Horse Racing	Horse Racing	Pass
TC_ID_010	Prediction on a video of Jumping	Jumping	Soccer juggling	Fail

6.2 TESTING TECHNIQUES / TESTING STRATEGIES

Unit Testing

Unit testing is used to ensure that each modular component of the software works properly. The smallest unit of software design (i.e., the module) is the focus of unit testing. Unit testing was mainly reliant on white-box testing approaches.

White-box testing

Glass box testing is another name for this type of testing. By identifying the specific functions that a product is meant to do, tests can be conducted to demonstrate that each function is completely functioning while also looking for flaws in each function. It is a test case design method that derives test cases from the procedural design's control structure. Basis path testing comes under the category of white box testing.

Black box testing

Without knowing how a product works inside, a test can be run to confirm that the internal operation works as expected and that all internal components have been thoroughly exercised. It is primarily concerned with the software's functional requirements.

CHAPTER 7
CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION:

Following extensive testing, comparing results revealed that the model generates exact results with an accuracy of 90.90 percent, which is an important strategy for application in open contexts. Because of its high Frame Per Second, this lightweight model is easy to assess and may be used constantly with the expansion of tiresome computation or picture collapsing.

Since this model can detect activities faster than other models, this system can be used for surveillance and supervision programs. If the GPU resources is improved, this model can be trained to improve its accuracy even more.

FUTURE ENHANCEMENTS:

Human activity recognition system has so many use cases in the current environment like patient monitoring, traffic monitoring, and also in childcare centers to babysit the kids and in other industries where supervision is done. We can also implement real-time action recognition, for example if an individual, tries to assault another individual, and trigger emergency response actions based on the recognized action. This applies to natural disasters, street fights, communal riots, public nuisance, etc.

Actions classes beyond the 50 classes that have been used in this proposed project. The more the data accumulates, the more accurate and precise prediction of the action prediction can be performed. Looking into the future, the collection of data is going to be upside by a huge margin compared to past and present scenarios.

APPENDIX 1:

Code for building CNN model:

```
def create_convlstm_model():
    '''
    This function will construct the required convlstm model.
    Returns:
        model: It is the required constructed convlstm model.
    '''

    # We will use a Sequential model for model construction
    model = Sequential()

    # Define the Model Architecture.
    ########################################################################
########################################################

    model.add(ConvLSTM2D(filters = 4, kernel_size = (3, 3), activation
= 'tanh',data_format = "channels_last",
                         recurrent_dropout=0.2, return_sequences=True,
input_shape = (SEQUENCE_LENGTH,

                IMAGE_HEIGHT, IMAGE_WIDTH, 3)))

    model.add(MaxPooling3D(pool_size=(1, 2, 2), padding='same', data_fo
rmat='channels_last'))
    model.add(TimeDistributed(Dropout(0.2)))

    model.add(ConvLSTM2D(filters = 8, kernel_size = (3, 3), activation
= 'tanh', data_format = "channels_last",
                         recurrent_dropout=0.2, return_sequences=True))

    model.add(MaxPooling3D(pool_size=(1, 2, 2), padding='same', data_fo
rmat='channels_last'))
    model.add(TimeDistributed(Dropout(0.2)))

    model.add(ConvLSTM2D(filters = 14, kernel_size = (3, 3), activation
 = 'tanh', data_format = "channels_last",
                         recurrent_dropout=0.2, return_sequences=True))

    model.add(MaxPooling3D(pool_size=(1, 2, 2), padding='same', data_fo
rmat='channels_last'))
    model.add(TimeDistributed(Dropout(0.2)))
```

```
    model.add(ConvLSTM2D(filters = 16, kernel_size = (3, 3), activation
 = 'tanh', data_format = "channels_last",
                         recurrent_dropout=0.2, return_sequences=True))

    model.add(MaxPooling3D(pool_size=(1, 2, 2), padding='same', data_fo
rmat='channels_last'))
    #model.add(TimeDistributed(Dropout(0.2)))

    model.add(Flatten())

    model.add(Dense(len(CLASSES_LIST), activation = "softmax"))

    ##################################################################
##################################################

    # Display the models summary.
    model.summary()

    # Return the constructed convlstm model.
    return model
```

Code for Building LRCN model:

```
def create_LRCN_model():
    '''
    This function will construct the required LRCN model.
    Returns:
        model: It is the required constructed LRCN model.
    '''

    # We will use a Sequential model for model construction.
    model = Sequential()

    # Define the Model Architecture.
    ########################################################################
########################################################

    model.add(TimeDistributed(Conv2D(16, (3, 3), padding='same',activat
ion = 'relu'),
                              input_shape = (SEQUENCE_LENGTH, IMAGE_HEI
GHT, IMAGE_WIDTH, 3)))

    model.add(TimeDistributed(MaxPooling2D((4, 4))))
    model.add(TimeDistributed(Dropout(0.25)))

    model.add(TimeDistributed(Conv2D(32, (3, 3), padding='same',activat
ion = 'relu')))
    model.add(TimeDistributed(MaxPooling2D((4, 4))))
    model.add(TimeDistributed(Dropout(0.25)))

    model.add(TimeDistributed(Conv2D(64, (3, 3), padding='same',activat
ion = 'relu')))
    model.add(TimeDistributed(MaxPooling2D((2, 2))))
    model.add(TimeDistributed(Dropout(0.25)))

    model.add(TimeDistributed(Conv2D(64, (3, 3), padding='same',activat
ion = 'relu')))
    model.add(TimeDistributed(MaxPooling2D((2, 2))))
    #model.add(TimeDistributed(Dropout(0.25)))

    model.add(TimeDistributed(Flatten()))

    model.add(LSTM(32))

    model.add(Dense(len(CLASSES_LIST), activation = 'softmax'))

    ########################################################################
########################################################
```

```
    # Display the models summary.
    model.summary()

    # Return the constructed LRCN model.
    return model
```

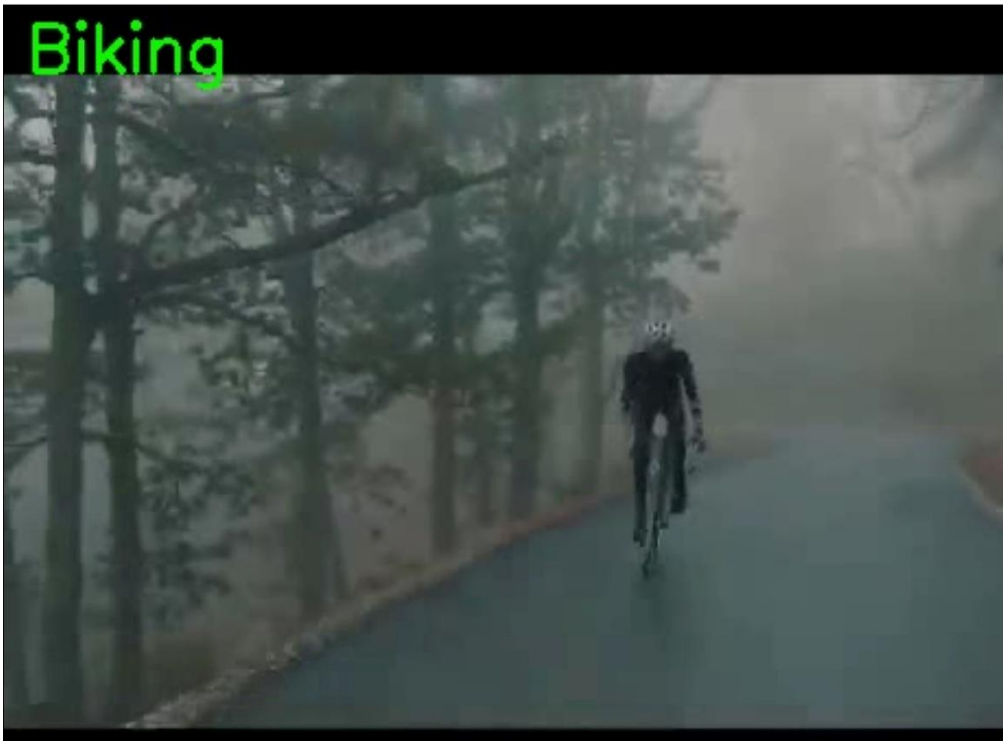
APPENDIX 2:

Action Predicted - Walking with dog

Action Predicted - Horse race detection

Action Predicted - Swing

Action Predicted – Tai Chi



Action Predicted – Biking

Action Predicted – Swing

This is based on another format of output, where it analyzes the whole video with numerous frames. And then, identifies the human action on the whole and gives the output as “Action detected – Swinging”.

REFERENCES

[1] Wenchao Xu; Yuxin Pang; Yanqin Yang; Yanbo Liu,” Human Activity Recognition Based On Convolutional Neural Network”, 2018 24th International Conference on Pattern Recognition (ICPR)

[2] Gulustan Dogan; Sinem Sena Ertas; İremnaz Cay,” Human Activity Recognition Using Convolutional Neural Networks”,2021 IEEE CIBCB

[3] Lamiyah Khattar; Chinmay Kapoor; Garima Aggarwal,” Analysis of Human Activity Recognition using Deep Learning”,Confluence

[4] Abhay Gupta; Kuldeep Gupta; Kshama Gupta; Kapil Gupta, “A Survey on Human Activity Recognition and Classification”, 2020 IEEE ICCSP

[5] Erhan Bulbul; Aydin Cetin; Ibrahim Alper Dogru,” Human Activity Recognition Using Smartphones”, 2018 IEEE ISMSIT

[6] Md. Atikuzzaman; Tarafder Razibur Rahman; Eashita Wazed; Md. Parvez Hossain; Md. Zahidul Islam, “Human Activity Recognition System from Different Poses with CNN”, 2020 IEEE STI

[7] Piyush Mishra; Sourankana Dey; Suvro Shankar Ghosh; Dibyendu Bikash Seal; Saptarsi Goswami, “Human Activity Recognition using Deep Neural Network”, 2020 IEEE WorldS4

[8] Khushboo Banjarey; Satya Prakash Sahu; Deepak Kumar Dewangan, “A Survey on Human Activity Recognition using Sensors and Deep Learning Methods”, 2021 IEEE ICCMC

[9] Ronald Mutegeki; Dong Seog Han, “A CNN-LSTM Approach to Human Activity Recognition”, 2020 IEEE ICAIIC

[10] Tsige Tadesse Alemayoh; Jae Hoon Lee; Shingo Okamoto, “Deep Learning Based Real-time Daily Human Activity Recognition and Its Implementation in a Smartphone”, 2019 IEEE UR

[11] Shicun Chen, Yong Zhang, Baocai Yin, Boyue Wang, 2021, “TRFH: towards real-time face detection and head pose estimation”, Springer

[12] Chaoqun Hong; Jun Yu; Jian Zhang; Xiongnan Jin; Kyong-Ho Lee, 2019, “Multimodal Face-Pose Estimation With Multitask Manifold Deep Learning”, IEEE Transactions on Industrial Informatics (Volume: 15, Issue: 7, July 2019)

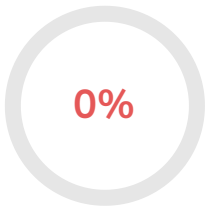
[13] Dweepna Garg; Parth Goel; Sharnil Pandya; Amit Ganatra; Ketan Kotecha, 2018, “A Deep Learning Approach for Face Detection using YOLO”, 2018 IEEE Punecon

[14] Nataniel Ruiz; Eunji Chong; James M. Rehg, 2018, “Fine-Grained Head Pose Estimation Without Keypoints”, IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)

[15] Hao Wu; Ke Zhang; Guohui Tian,2018, “Simultaneous Face Detection and Pose Estimation Using Convolutional Neural Network Cascade”, IEEE Access (Volume: 6)

Plagiarism Scan Report

Report Generated on: May 23,2022



Plagiarised

Unique

Total Words:	0
Total Characters:	
Plagiarized Sentences:	0
Unique Sentences:	48 (100%)

Content Checked for Plagiarism

ABSTRACT:
Human Activity Recognition using smartphone sensors like accelerometers is one of the hot topics of research. It is one of the time series classification challenges. The proposed project does not include any hardware devices, and it is instead based on machine learning models that have been optimised to obtain the best possible results. LRCN (Long-term Recurrent Convolutional Network) can be used to recognise various activities of humans like standing and walking. The LRCN model is a recurrent neural network that can learn order dependency in sequence prediction challenges. This model is used because it helps with recalling values over long periods of time.

INTRODUCTION:
The traditional behaviour model consists of creating a statistical or analytical model of human behaviour and then fitting a distribution to the model to validate it. The deep learning approach differs significantly from the traditional model. By studying human behaviour, a deep learning system learns to predict outcomes. Deep learning is the study of human behavioural features and its application to prediction. Deep learning could be a game changer in commercial applications. The ability to predict and forecast behaviour, as well as calculate the likelihood of winning a game, are valuable economic advantages.

AIM OF THE PROJECT:
The purpose of this study is to see how effective deep learning-based algorithms are at recognising and analysing different elements of human behaviour. However, the majority of our video surveillance systems are still run in the old-fashioned method, with anomalies and evidence gathered solely through offline videos. It's difficult to create real-time alarms, pop-up notifications, and continuously monitor crisis situations.
As a result, real-time human behaviour identification technologies must be developed to reduce security staff workload and increase productivity.

EXISTING SYSTEM:
The existing system has low accuracy and low efficiency in terms of loading time and implementation time. Also, the testing and training are not done with the proper test-train split ratio. In the existing system, human activity is predicted by giving an input image which doesn’t give good accuracy.

Current technology of Action recognition systems is mostly dependent on convolutional networks and it is mostly a slow computing algorithm and requires a large amount of computing power to handle video recognition. The current system has to analyze a whole video before saying what action is performed in those videos. These can’t help in detecting actions in a live video feed.

PROPOSED SYSTEM:
The proposed system has a high level of accuracy. When compared to the existing system, the suggested system's loading and execution times are rapid. The suggested system is highly efficient and scalable, and it may be enhanced for more complex use cases.

With few future enhancements, this proposed system can predict outputs from live video feeds like for

example live feeds from a CCTV. As this system predicts action on every frame of a video this system can be used in surveillance programs and supervision programs.

This proposed system makes use of the LRCN algorithm which is a combination of CNN and LSTM. This is a more modern way of handling the problem and it is promised to perform fast with high accuracy and fewer computing resources.

RESULTS AND FINDINGS:
After carefully inserting the processed dataset of the action classes, the CNN model as well as the LRCN have been trained to some extent. Considering the motive behind this proposed system, which being to portray a comparison between a CNN model and LRCN model, the objective has been achieved at the completion of this project. LRCN is constructed as a combination of Convolutional Neural Networks (CNNs) and Long Short Term Memory (LSTM), which helps to achieve video regression/activity recognition much efficiently when compared to a standard CNN model.

Given both the models have been trained for a significant amount of time using the dataset, the human activity recognition has proved to be effective. In the case of the CNN model, it has achieved an 81.15% accuracy. Whereas, the LRCN model was able to achieve a 93% accuracy.

The amount of processing power has increased all the way over the years. The quantity of data accumulated along the way is beyond imaginable when compared to a few decades ago. The more data used to train the Neural Networks, the more efficiently and accurately the models can detect the human activity.

CONCLUSION:
Following extensive testing, comparing results revealed that the model generates exact results with an accuracy of 90.90 percent, which is an important strategy for application in open contexts. Because of its high Frame Per Second, this lightweight model is easy to assess and may be used constantly with the expansion of tiresome computation or picture collapsing.
Since this model can detect activities faster than other models, this system can be used for surveillance and supervision programs. If the GPU resources is improved, this model can be trained to improve its accuracy even more.

FUTURE ENHANCEMENTS:
Human activity recognition system has so many use cases in the current environment like patient monitoring, traffic monitoring, and also in childcare centers to babysit the kids and in other industries where supervision is done. We can also implement real-time action recognition, for example if an individual, tries to assault another individual, and trigger emergency response actions based on the recognized action. This applies to natural disasters, street fights, communal riots, public nuisance, etc. Actions classes beyond the 50 classes that have been used in this proposed project. The more the data accumulates, the more accurate and precise prediction of the action prediction can be performed. Looking into the future, the collection of data is going to be upside by a huge margin compared to past and present scenarios.

No Plagiarism Found