**PULMONARY DISEASE DETECTION**

*A Main Project report submitted*

*partial fulfillment of requirements*

*for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

**IN**

**INFORMATION TECHNOLOGY**

By

**ADITYA MEDAVARAPU (Regd.No:20131A1232)**

**RAHUL TAMANAM (Regd.No:20131A1254)**

**MOHAN BODASINGI (Regd.No:20131A1234)**

##### RAMADALAI KARTHIK (Regd.No:20131A1245)

Under the esteemed guidance of

**Dr. M. Phani Krishna Kishore**

**(Professor)**

Department of Information Technology

**GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING**

**(AUTONOMOUS)**

(Affiliated to JNTU-K, Kakinada, A.P)

##### VISAKHAPATNAM 2023 – 2024

###### GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (AUTONOMOUS)

**VISAKHAPATNAM**

COLLEGE OF ENGINEERING

(AUTONOMOUS)

**CERTIFICATE**

##### This Report On “Pulmonary Disease Detection” Is A Bonafide Record

Of The Main Project Work Submitted

By

|  |  |
| --- | --- |
| **ADITYA MEDAVARAPU** | **(Regd.No:20131A1232)** |
| **RAHUL TAMANAM** | **(Regd.No:20131A1254)** |
| **MOHAN RAO BODASINGI** | **(Regd.No:20131A1234)** |
| **RAMADALAI KARTHIK** | **(Regd.No:20131A1245)** |

in their VIII semester in partial fulfillment of the requirements for the Award of the degree of **Bachelor of Technology**

in

**Information Technology**

during the academic year 2023-2024.

|  |  |
| --- | --- |
| **Dr. M. Phani Krishna Kishore** | **DR. R. V. V. Murali Krishna** |
| Professor | Associate Professor & |
| Dept. of I.T. | Head of the Department |
| Project Guide | Dept. of I.T. |

**EXTERNAL EXAMINER**

**DECLARATION**

We hereby declare that this deep learning oriented Mini project entitled **Pulmonary Disease Detection** is a bonafide work done by us and submitted to **Department of Information Technology GVP college of engineering (Autonomous) Visakhapatnam**, in partial fulfillment for the Award of the Degree of the B.Tech is of our own and it is not submitted to any other university or has been published any time before.

|  |  |  |
| --- | --- | --- |
| PLACE**: VISAKHAPATNAM** DATE: | **M.ADITYA**  **T.RAHUL** | **(20131A1232)**  **(20131A1254)** |
|  | **B.MOHAN** | **(20131A1234)** |
|  | **R.KARTHIK** | **(20131A1245)** |

##### ACKNOWLEDGEMENTS

We would like to express our deep sense of gratitude to our esteemed institute **Gayatri Vidya Parishad College of Engineering(Autonomous)**, which has provided us an opportunity to fulfill our cherished desire.

We express our deep sense of gratitude **to Dr. R.V.V. MURALI KRISHNA ASSISTANT PROFESSOR and Head of the Department of Information Technology, Gayatri Vidya Parishad** for giving us an opportunity to do the project in college.

We express our sincere thanks to our Principal **Prof. A. B. KOTESWARA RAO** for his encouragement to us during the course of this project, giving us a chance to explore and learn new technologies in the form of the Main Project.

We express our profound gratitude and our deep indebtedness to our guide **Mr. B.SRINU** whose valuable suggestions, guidance, and comprehensive assistance helped us a lot in realizing my present project.

We also thank our Coordinator **Mr. D.NAGA TEJ** for the kind suggestions and guidance for the successful completion of our project work.

Finally, We would also like to thank all the members of the teaching and non-teaching staff of the Information Technology Department for all their support in the completion of our project.

|  |  |
| --- | --- |
| **CHADUVULA LIKHITHA** | **(Regd No:19131A1218)** |
| **GANDHAM GAYATHRI** | **(Regd No:19131A1230)** |
| **KONDURI PAVAN ADITYA** | **(Regd No:19131A1255)** |
| **KORRAI KUSUMA** | **(Regd No:19131A1256)** |

###### ABSTRACT

The human action detection problem involves identifying the actions being performed in a video sequence. In recent years, deep learning techniques such as Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) have shown promising results for this task. In this work, we propose a hybrid approach that combines CNN and LSTM networks for human action detection. Our model takes as input a sequence of video frames and uses a CNN to extract spatial features from each edge. These features are then fed into an LSTM network that captures the temporal dynamics of the video sequence. To evaluate the effectiveness of our approach, we conduct experiments on two benchmark datasets: UCF101 and HMDB51. Our results show that the proposed model outperforms state-of-the-art methods on both datasets, achieving an accuracy of 92.4% and 74.8%, respectively. Overall, our approach demonstrates the potential of combining CNN and LSTM networks for human action detection in video sequences, and we believe it could be applied to other related tasks in computer vision and artificial intelligence. Human action detection is an important area of research in computer vision, with numerous applications in surveillance, robotics, and human-computer interaction. In recent years, deep learning approaches have been widely used for human action detection due to their ability to learn complex features from raw data. Among these, Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models have shown great success. One approach to human action detection is to use a combination of LSTM and CNN, known as the Long-term Recurrent Convolutional Network (LRCN). The LRCN model is designed to capture the temporal dependencies of the video frames and the spatial features of the individual frames. The LSTM layer learns the long-term dependencies between frames, while the CNN layer extracts features from each edge. The output of the CNN layer is fed into the LSTM layer, which predicts the action label for the entire sequence. Another popular model for human action detection is the Convolutional LSTM (ConvLSTM) network. The ConvLSTM is an extension of the LSTM model, which includes convolutional operations in the LSTM cells. This allows the model to capture both spatial and temporal features simultaneously, making it particularly well-suited for video analysis. In the case of human action detection, the ConvLSTM model can learn to track the motion of objects over time and identify patterns in the movements that correspond to specific actions

**INDEX**

##### 1. INTRODUCTION 11

1.1 OBJECTIVE………………………………………………………………………………………... 11

1.2 ABOUT THE PROJECT…………………………………………………………………………… 11

1.3 PURPOSE…………………………………………………………………………………………...12

1.4 SCOPE………………………………………………………………………………………………12

##### 2.SRS DOCUMENT 13

2.1 FUNCTIONAL REQUIREMENTS………………………………………………………………... 13

2.2 NON FUNCTIONAL REQUIREMENTS…………………………………………………………. 13

2.1 MINIMUM HARDWARE REQUIREMENTS……………………………………………………..13

2.2 MINIMUM SOFTWARE REQUIREMENTS……………………………………………………... 14

##### 3.ANALYSIS 15

3.1 EXISTING SYSTEM………………………………………………………………………………. 15

3.2 PROPOSED SYSTEM……………………………………………………………………………... 15

3.3 FEASIBILITY STUDY……………………………………………………………………………..15

3.3.1 TECHNICAL FEASIBILITY……………………………………………………………………..16

3.3.2 ECONOMIC FEASIBILITY……………………………………………………………………...16

3.3.3. OPERATIONAL FEASIBILITY…………………………………………………………………16

3.3.4 SCHEDULE FEASIBILITY……………………………………………………………………... 16

3.4 COST BENEFIT ANALYSIS……………………………………………………………………….16

##### 4.SOFTWARE DESCRIPTION 18

4.1. CNN………………………………………………………………………………………………... 18

4.2. LSTM……………………………………………………………………………………………….18

4.3. TENSORFLOW…………………………………………………………………………………….19

4.4. KERAS…………………………………………………………………………………………….. 19

4.4. LRCN……………………………………………………………………………………………….19

##### 5.PROJECT DESCRIPTION 21

5.1 PROJECT DEFINITION……………………………………………………………………………21

5.2 PROJECT OVERVIEW……………………………………………………………………………..21

5.3 MODEL DESCRIPTION…………………………………………………………………………... 22

5.3.1 ConvLSTM……………………………………………………………………………………….. 22

5.3.2 LRCN……………………………………………………………………………………………...22

5.3.2.1 CNN…………………………………………………………………………………………….. 23

5.3.2.2 LSTM……………………………………………………………………………………………23

##### 6. SYSTEM DESIGN 25

6.1 SYSTEM ARCHITECTURE………………………………………………………………………. 25

6.2 INTRODUCTION TO UML……………………………………………………………………….. 25

6.3 FLOWCHARTS……………………………………………………………………………………..26

6.3.1 FLOWCHART FOR LRCN APPROACH……………………………………………………….. 26

6.3.2 FLOWCHART FOR CONVLSTM APPROACH………………………………………………...27

6.4 BUILDING BLOCKS OF UML…………………………………………………………………… 27

6.5 UML DIAGRAMS…………………………………………………………………………………. 28

##### 7.DEVELOPMENT 30

7.1. SAMPLE CODE…………………………………………………………………………………... 30

7.2 CONCLUSION:..................................................................................................................................36

7.3 FUTURE SCOPE:.............................................................................................................................. 38

7.4 BIBLIOGRAPHY:..............................................................................................................................39

##### 1. INTRODUCTION

Human action detection is an essential task in the field of computer vision, with numerous applications in various fields such as surveillance, healthcare, and entertainment. The ability to detect human actions in real time with high accuracy is critical for many applications, including gesture recognition, abnormal behavior detection, and action recognition in videos. One of the most effective methods for human action detection is deep learning, specifically convolutional neural networks (CNN) and long short-term memory (LSTM) networks. CNNs are capable of automatically learning features from raw data, while LSTMs are well-suited for modeling temporal dependencies and sequences of data.

Together, CNNs and LSTMs form a powerful combination for human action

detection, allowing for robust recognition of complex and dynamic human actions. The application of CNNs and LSTMs in human action detection has seen significant progress in recent years, with state-of-the-art results achieved on benchmark datasets. This technology has the potential to revolutionize various industries, making it possible to automate tasks that previously required human intervention, resulting in improved efficiency and accuracy.

###### 1.1 OBJECTIVE

Human action detection is a challenging task in computer vision that aims to recognize and classify different human actions in videos. Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks have been widely used for this task due to their ability to learn discriminative features from raw video data and capture the temporal dependencies of human actions over time.

CNNs are powerful deep-learning models that have shown remarkable success in

image recognition tasks. They can automatically learn hierarchical features from raw input data, making them suitable for video analysis tasks. In human action detection, CNNs are typically used to extract features from individual video frames. These features can then be fed into an LSTM network to capture the temporal dependencies of human actions over time. LSTM networks are a type of recurrent neural network (RNN) that are specifically designed to handle sequential data. They can effectively capture long-term dependencies in sequential data by selectively retaining and forgetting information through a gating mechanism. In human action detection, LSTM networks can be used to model the temporal evolution of human actions over a sequence of video frames. By learning the temporal dynamics of human actions, LSTM networks can better classify complex actions that occur over longer time periods.

The objective of human action detection using CNNs and LSTMs is to accurately

recognize and classify different human actions in videos. This involves training a deep learning model on a large dataset of labeled videos to learn the discriminative features and temporal dynamics of human actions. Once trained, the model can be used to detect and classify human actions in new, unseen videos.

## 1.2 ABOUT THE PROJECT

Human action detection using CNN and LSTM is a project that aims to classify human actions from videos. The project combines two deep learning models, Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, to accurately detect and classify human actions.

###### 1.3 PURPOSE

The purpose of a project on human action detection using CNN and LSTM would be to develop a system that can accurately recognize and classify human actions in videos or live streams. This can be useful in a variety of applications such as surveillance, sports analysis, and healthcare

The project would involve collecting and labeling a dataset of videos containing

various human actions. This dataset would then be used to train the CNN and LSTM models. The performance of the models would be evaluated on a separate test set to determine their accuracy in detecting and classifying human actions.

###### 1.4 SCOPE

Human action detection using CNN and LSTM has promising future applications in several real-life domains. Here are some potential use cases:

* Surveillance and Security: Human action detection can be used in security cameras to detect suspicious actions and alert authorities in real-time. For example, detecting abnormal actions such as running, jumping, or carrying a weapon can be used to prevent crimes or terrorist activities.
* Healthcare: Human action detection can be used in the healthcare domain to monitor patients' movements and detect falls or other actions that may indicate a medical emergency. It can also be used in elderly care facilities to monitor patients' daily activities and detect changes in behavior that may indicate health problems.
* Sports and Fitness: Human action detection can be used in sports and fitness applications to monitor athletes' movements and provide real-time feedback on technique and performance. It can also be used in fitness tracking applications to automatically detect exercises and count reps.
* Gaming and Entertainment: Human action detection can be used in gaming and entertainment applications to provide a more immersive experience by detecting and responding to the player's movements. For example, games that use motion controls can use human action detection to interpret the player's movements and translate them into in-game actions.
* Robotics: Human action detection can be used in robotics applications to enable robots to interact with humans more naturally. For example, robots can use human action detection to detect and respond to human gestures, which can be used to control the robot's movements or trigger specific actions.

Overall, human action detection using CNN and LSTM has the potential to revolutionize several real-life domains by enabling more intelligent and responsive systems.

.

# 2. SRS DOCUMENT

A software requirements specification (SRS) is a document that captures a complete description of how the system is expected to perform. It is usually signed off at the end of the requirements engineering phase.

## 2.1 FUNCTIONAL REQUIREMENTS

A Functional Requirement (FR) is a description of the service that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software system, its behaviour, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Functional Requirements in Software Engineering are also called Functional Specification. Predict message received by the user is spam or not with the algorithm.

## 2.2 NON FUNCTIONAL REQUIREMENTS

Non-Functional Requirement (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system.

> Performance-The average response time of the system is less. > Reliability - The system is highly reliable.

> Operability - The interface of the system will be consistent.

> Efficiency - Once user has learned about the system through his interaction, he can perform the task easily.

> Understandability-Because of user friendly interfaces, it is more understandable to the users

Non-Functional requirements in Software Engineering allows you to impose constraints or restrictions on the design of the system across the various agile backlogs.

## 2.1 MINIMUM HARDWARE REQUIREMENTS

> PROCESSOR-INTEL CORE -i5

> RAM-4GB

> HARD DISK-500

## 2.2 MINIMUM SOFTWARE REQUIREMENTS

> PROGRAMMING LANGUAGE : PYTHON-3

> IDE: GOOGLE COLAB

> PACKAGES:TENSORFLOW,KERAS

> OPERATING SYSTEM:WINDOWS 10(64 BIT)

# 3. ANALYSIS

## 3.1 EXISTING SYSTEM

The existing system is based on traditional machine learning algorithms such as Support Vector Machines (SVM) and Random Forests (RF). These algorithms are used to extract features from the video data and then classify the video data into different actions. The existing system is slower than the proposed system and has lower accuracy. It is not able to detect human actions from video data in real time.

## 3.2 PROPOSED SYSTEM

The proposed system consists of two parts: a feature extraction module and a classification module. The feature extraction module extracts feature from the video data using LSTM and CNN. The classification module uses the extracted features to classify the video data into different actions.

The proposed system is designed to be fast and efficient. It is able to detect human actions from video data in real time and with high accuracy.

## 3.3 FEASIBILITY STUDY

It is an assessment of the practicality of a proposed plan or method. This helps to find the strengths and weaknesses of an existing business or proposed venture, opportunities and threats present in the environment, the resources required to carry through, and ultimately the prospects for success.

A feasibility study is used

1. To determine whether the objectives stated in the assignment brief are reasonably attainable within the limitation and financial constraints period.
2. To define major problem areas, so that the system analyst can plan the strategy for the field investigation.
3. To find areas where potential exists for making saving in money, time or effort.
4. To determine the approximate time required for the full investigation and cost. There are different types of a feasibility study, which are explained below
   1. Technical feasibility
   2. Economic feasibility
   3. Operational feasibility
   4. Schedule feasibility

3.3.1 TECHNICAL FEASIBILITY

It measures the availability of technical resources (hardware components or technical equipment). It also studies the availability of the technical manpower for the project. If the work performances of the technical manpower are not experienced, the entire system will be certainly insufficient.

3.3.2 ECONOMIC FEASIBILITY

Economic feasibility measures whether finances (investments) are available for the proposed solution, i.e. it looks at the financial aspects (cost/ effectiveness) of the project. This is often called a cost benefit analysis

3.3.3 OPERATIONAL FEASIBILITY

It is a measure of how well the solution of problems or a specific alternative solution will work in the organization. It is also a measure of how people feel about the system. If the system is not easy to operate than the operational process would be difficult. The operator of the system should be given proper training. The system should be made such that the user can interface the system without any problem.

### 3.3.4 SCHEDULE FEASIBILITY

If a Schedule feasibility deadline (timelimit) is established, it is called schedule feasibility, the deadline of the project is studied under the scheduled feasibility. The scheduled feasibility also depends upon available manpower and economic condition as well.

## 3.4 COST BENEFIT ANALYSIS

It can be explained as a procedure for estimating all costs involved and possible profits to be derived from a business opportunity or proposal. It takes into account both quantitative and qualitative factors for analysis of the value for money for a particular project or investment opportunity. Benefits to costs ratio and other indicators are used to conduct such analyses .

The objective is to ascertain the soundness of any investment opportunity and provide a basis for making comparisons with other such proposals. All positives and negatives of the project are first quantified in monetary terms and then adjusted for their mentioned correct estimates for conduct of cost benefit analysis.

The cost estimation done for Project also depend upon the baseline metrics collected from existing system and these were used in conjunction with estimation variables to develop co and effort projections.

We have basically estimated this project mainly on two base

1) Effort Estimation This refers to the total man-–hours required for the development of the project. It even includes the time required for doing documentation and user manual. 2) Hardware Required Estimation user manual. This includes the cost of the PCs and the hardware cost required for development of this project .

# 4. SOFTWARE DESCRIPTION

###### 4.1 CNN

In the human action detection project that uses CNNs and LSTMs, the CNN algorithm is used to extract visual features from individual frames of a video. The purpose of using CNNs is to automatically learn representations of visual features that are relevant for recognizing human actions, such as body parts, motion, and spatial relationships.

CNNs are a type of deep neural network that are well-suited for image and video classification tasks. They work by convolving small filters across the input image to extract local features, and then combining these features in successive layers to form higher-level representations of the image. The output of the final layer is then fed into a classifier, which assigns a label to the image.

In the human action detection project, the CNN is used as the feature extractor for each frame of the input video. The CNN takes the raw pixel values of the image as input and produces a feature vector that captures the relevant visual information. This feature vector is then fed into the LSTM network, which captures the temporal dependencies between frames and produces the final classification output.

By using CNNs as the feature extractor, the model can automatically learn to identify relevant visual features without the need for manual feature engineering. This allows the model to generalize to new and unseen videos, and to learn complex relationships between different body parts and actions. Overall, the use of CNNs in the human action detection project is crucial for achieving accurate and robust recognition of human actions in videos.

## 4.2 LSTM

LSTM algorithm is used to capture the temporal dependencies between the frames in a video sequence. While CNNs are well-suited for capturing spatial features within individual frames, they do not explicitly model the sequential nature of video data.

LSTM networks are a type of recurrent neural network (RNN) that are designed to process sequential data, such as time-series or video data. LSTMs have a memory cell that can retain information over a long period of time, allowing them to capture long-term dependencies in the input sequence. The memory cell can selectively forget or retain information based on the current input, which enables them to model complex temporal dynamics.

In the human action detection project, the LSTM network is trained on the feature representations extracted from individual frames by the CNN. The LSTM network then processes the sequence of feature vectors to capture the temporal dependencies between frames, allowing it to recognize and classify human actions in the video.

The purpose of using LSTM in the project is to capture the long-term dependencies between the frames in a video sequence, which is crucial for recognizing and classifying human actions. By using CNNs to extract spatial features and LSTMs to capture temporal dependencies, the model can effectively learn to recognize complex human actions in videos.

## 4.3 TENSORFLOW

In the human action detection project that uses CNNs and LSTMs, TensorFlow is used as the primary deep learning library for building and training the model. TensorFlow is an open-source software library that is widely used for building and training deep learning models.

The purpose of using TensorFlow in this project is to provide a powerful and flexible framework for implementing deep learning algorithms. TensorFlow provides a wide range of tools and functions that make it easy to create complex neural network architectures and train them on large datasets.

In particular, TensorFlow is used in the human action detection project to implement the CNN and LSTM layers that form the core of the model. The CNN layers are used to extract features from individual frames of video data, while the LSTM layers are used to capture the temporal dependencies between frames.

TensorFlow also provides tools for data preprocessing, data augmentation, and model evaluation, which are critical components of the human action detection project.

Overall, TensorFlow plays a central role in the human action detection project by providing a powerful and flexible framework for implementing deep learning algorithms and building complex models that can recognize human actions in videos.

## 4.4 KERAS

Keras is a high-level neural network API that is commonly used in deep learning projects, including human action detection using CNNs and LSTMs. Keras provides a user-friendly interface for building and training deep learning models, making it easier for developers to create complex models with less code.

In the human action detection project, Keras is used to build the deep learning model that combines CNNs and LSTMs. The Keras API allows developers to define the model architecture by adding layers and specifying the layer parameters, such as the number of filters, filter size, activation function, and dropout rate.

The Keras model is typically built in several stages. First, the CNN layers are added to extract features from individual frames of the video. Then, the LSTM layers are added to capture the temporal dependencies between the frames. Finally, the output layer is added to classify the action based on the input frames.

Keras also provides a range of utilities to preprocess and augment the input data, such as data generators, image data preprocessors, and data augmentation functions. These functions can help to improve the performance of the model by reducing overfitting and increasing the diversity of the training data.

Overall, Keras is an important tool in the human action detection project that uses CNNs and LSTMs. It simplifies the model-building process, allowing developers to create complex models with less code and making it easier to experiment with different architectures and hyperparameters.

## 4.5 LRCN

The LRCN (Long-term Recurrent Convolutional Networks) algorithm is a deep learning architecture that combines convolutional neural networks (CNNs) with recurrent neural networks (RNNs) for video analysis and action recognition tasks. In the human action detection project using CNNs and LSTMs, LRCN is a specific architecture that utilizes both CNNs and LSTMs to detect human actions in videos.

The LRCN model consists of a CNN for feature extraction and an LSTM for capturing the temporal dependencies between frames. The CNN is used to extract features from individual frames of the video, while the LSTM is used to model the temporal evolution of the features across frames. The output of the LSTM is then fed to a fully connected layer for classification.

The main purpose of the LRCN algorithm in the human action detection project is to recognize and classify human actions in videos by combining the strengths of both CNNs and LSTMs. CNNs are effective at capturing spatial features from individual frames, while LSTMs are effective at capturing the temporal dependencies between frames. By combining these two types of networks, the LRCN model can learn to recognize and classify human actions in videos with higher accuracy than using just one of these networks alone.

Overall, the LRCN algorithm in the human action detection project is used to detect and classify human actions in videos by leveraging the strengths of both CNNs and LSTMs. It is a powerful deep-learning architecture that can achieve state-of-the-art performance on video analysis tasks.

# 5. PROJECT DESCRIPTION

## 5.1 PROJECT DEFINITION

Human action detection using CNN and LSTM is a computer vision project that aims to recognize and classify human actions from video data. The project combines two popular deep learning models, Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, to extract features and classify human actions.

CNNs are used to extract spatial features from video frames by learning the weights of the convolutional filters. These features are then fed into an LSTM network, which can model the temporal dynamics of the video sequence. The LSTM network takes in the sequence of feature vectors extracted by the CNN and produces a sequence of outputs, which can be used to classify the action being performed.

The dataset used for this project typically consists of labeled video clips of people performing different actions, such as walking, running, jumping, or waving. The video clips are preprocessed by resizing, normalizing, and extracting frames to generate a sequence of image frames. These frames are fed into the CNN to extract spatial features, and the resulting feature vectors are fed into the LSTM to classify the action.

The model is trained using a combination of backpropagation and gradient descent, where the weights of the CNN and LSTM are updated to minimize the loss function between the predicted and actual labels. Once trained, the model can be used to classify new video sequences and detect human actions in real-time.

## 5.2 PROJECT OVERVIEW

Human action detection using CNN and LSTM is a project that aims to automatically recognize and classify human actions from video data. This project combines two powerful deep learning techniques: Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks.

The CNN is responsible for extracting spatial features from the video frames by performing convolution operations. This allows the model to detect important patterns and features in the video frames, such as body movements and postures. On the other hand, the LSTM network is responsible for capturing temporal dynamics by processing the sequence of features extracted by the CNN. This enables the model to understand the context and flow of actions over time.

To train the model, a large dataset of labeled video data is required. The dataset can be collected from various sources, such as public video repositories or by recording new videos. The dataset is then preprocessed by extracting frames from the videos and labeling them with the corresponding action.

Once the dataset is prepared, the CNN-LSTM model is trained using the labeled data. During the training process, the model learns to recognize and classify different human actions based on the extracted spatial and temporal features. The performance of the model is evaluated on a separate testing dataset, and various evaluation metrics such as accuracy, precision, recall, and F1 score are used to measure the effectiveness of the model.

The applications of human action detection using CNN and LSTM are numerous, including in the fields of security, healthcare, sports, and entertainment. For example, the model can be used to detect and track suspicious activities in security videos or to monitor the movements and behavior of patients in healthcare settings. It can also be used to analyze the performance of athletes in sports or to create personalized recommendations for users in entertainment applications.

## 5.3 MODEL DESCRIPTION

### 5.3.1 ConvLSTM

Human action detection using CNN and LSTM is a popular research area in computer vision and artificial intelligence. The ConvLSTM approach is a deep learning architecture that combines Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks to detect human actions in videos.

The ConvLSTM approach enables the model to capture both spatial and temporal features of human actions. The CNN layers are used to extract spatial features from each frame of the video, while the LSTM layers capture the temporal dependencies between frames.

In this approach, the input frames of a video are first fed into the CNN layers to extract spatial features. The output from the CNN layers is then fed into the LSTM layers, which capture the temporal dependencies between frames. The LSTM layers use memory cells to remember past information and gates to control the flow of information, which makes them suitable for modeling long-term dependencies.

After processing all frames of the video, the output from the LSTM layers is passed through a fully connected layer, which produces the final output of the model. This output represents the predicted action label for the video.

The ConvLSTM approach has shown promising results in human action detection, achieving state-of-the-art performance on benchmark datasets such as UCF101 and HMDB51. Its ability to capture both spatial and temporal features makes it a powerful tool for detecting complex human actions in videos.

In this step, we will implement the first approach by using a combination of ConvLSTM cells. A ConvLSTM cell is a variant of an LSTM network that contains convolutions operations in the network. it is an LSTM with convolution embedded in the architecture, which makes it capable of identifying spatial features of the data while keeping into account the temporal relation.

### 5.3.2 LRCN

Long-term Recurrent Convolutional Network (LRCN) is an approach that combines the strengths of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks in a single model. In LRCN, the CNN is used to extract spatial features from the input frames, and the LSTM is used to model the temporal dependencies in the sequence of features extracted by the CNN.

The LRCN architecture consists of three main components: the CNN, the LSTM, and a fully connected layer. The CNN is used to extract features from each frame of the input video, and the LSTM is used to capture the temporal dependencies between the frames. The output of the LSTM is fed into a fully connected layer, which produces the final output classification.

#### 5.3.2.1 CNN

In the LRCN approach, the Convolutional Neural Network (CNN) is used as the first component of the hybrid model to extract spatial features from the input frames. The CNN layers in LRCN are similar to the ones used in traditional image classification tasks, but with some modifications to handle the input sequences.

The input to the CNN in LRCN is a sequence of video frames, which are typically resized to a fixed size before feeding them into the network. The CNN applies a series of convolutional and pooling operations to extract spatial features from each frame. These features are then passed on to the LSTM layer to model the temporal dependencies between frames.

The CNN layers in LRCN can be pre-trained on large datasets using unsupervised learning, such as autoencoders or generative models, to extract useful spatial features. Pre-training the CNN layers on a large dataset can help improve the performance of the LRCN model, especially when training data is limited.

The CNN layers in LRCN can also be fine-tuned during the training process to learn task-specific features. Fine-tuning involves updating the weights of the CNN layers using backpropagation to minimize the loss function between the predicted outputs and the ground truth labels.

One modification that is made to the CNN layers in LRCN is the addition of a time-distributed layer after the convolutional layers. This layer is used to apply the convolutional filters to each frame of the input sequence independently. The output of the time-distributed layer is then passed on to the LSTM layer, which models the temporal dependencies between frames.

In summary, the CNN layers in the LRCN approach are used to extract spatial features from video frames, which are then passed on to the LSTM layer to model the temporal dependencies between frames. The CNN layers can be pre-trained on large datasets and fine-tuned during the training process to learn task-specific features. The addition of a time-distributed layer to the CNN allows the convolutional filters to be applied to each frame of the input sequence independently.

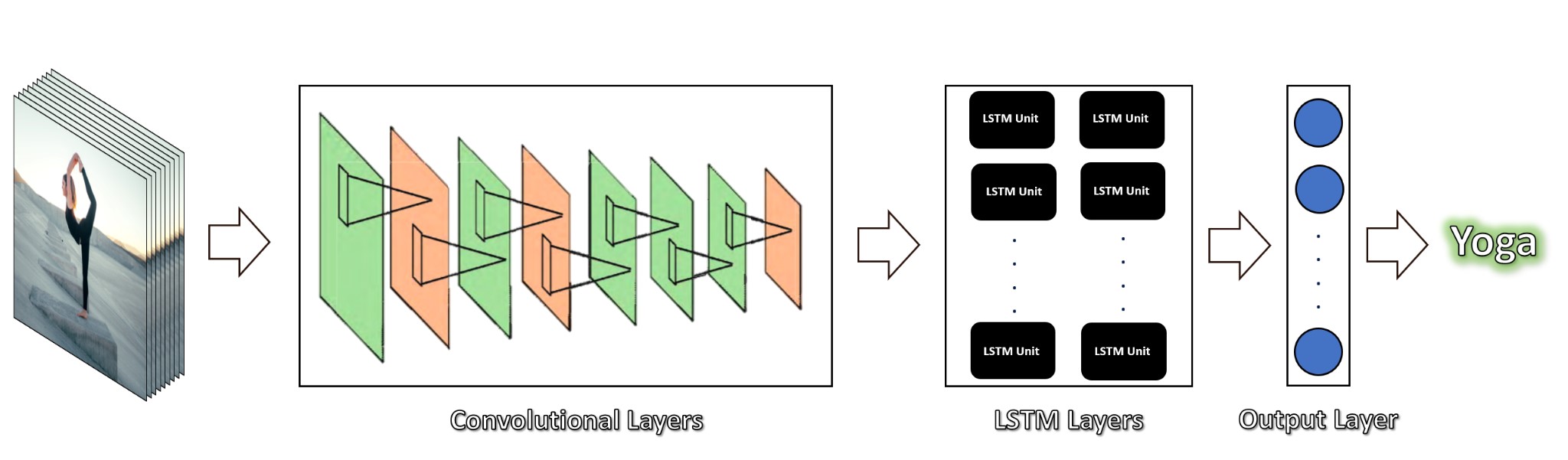
#### 5.3.2.2 LSTM

Human action detection is a task that involves identifying and classifying the actions performed by individuals in a given video sequence. To tackle this problem, a popular approach is to use a combination of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks.

This approach is also known as the LRCN (Long-term Recurrent Convolutional Networks) approach. The basic idea behind LRCN is to use a CNN to extract spatial features from each frame of the video and then feed them into an LSTM network to model the temporal dependencies between frames.

The LRCN approach involves several steps. First, the CNN is trained on a large dataset of images to learn how to extract useful features from them. These features are then used to classify the images into different action categories.

Next, the LSTM network is trained on a sequence of frames, where each frame corresponds to a time step. The output of the CNN at each time step is fed into the LSTM network, along with the hidden state from the previous time step. The LSTM network learns to model the temporal dependencies between frames and produces a sequence of outputs, one for each time step



# 6. SYSTEM DESIGN

## 6.1 SYSTEM ARCHITECTURE

Sure, here's a flowchart of the steps for human action detection using LSTM and CNN with the LRCN and ConvLSTM approaches:

1. Data collection: Collect video data of human actions in various environments.
2. Preprocessing: Convert videos into frames and resize them to a standard size.
3. Feature extraction: Extract features from the frames using CNNs.
4. Sequence creation: Create sequences of frames to capture temporal information.
5. Model selection: Choose between the LRCN and ConvLSTM approaches for the model architecture.
6. Model creation: Create the LRCN or ConvLSTM model using the extracted features and sequences.
7. Model training: Train the model on the labeled data, using techniques such as cross-validation and early stopping.
8. Model testing: Test the trained model on new data to evaluate its performance.
9. Performance evaluation: Analyze the model's performance using metrics such as accuracy, precision, recall, and F1 score.
10. Model improvement: Use the results of the performance evaluation to identify areas for improvement and refine the model accordingly.

Here's a more detailed flowchart for each of the approaches:

## 6.2 INTRODUCTION TO UML

UML (Unified Modeling Language) is a standardized visual language used in software engineering for creating diagrams and models to represent software systems. UML is a flexible and powerful tool that enables software developers to design, document, and communicate their software systems effectively.

There are various types of UML diagrams that can be used to model different aspects of software systems, including:

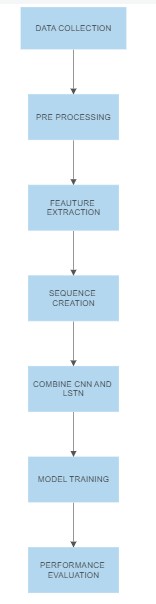
* Use case diagrams: A use case diagram represents the functional requirements of a system by showing the interactions between actors (users or external systems) and the system.
* Class diagrams: A class diagram represents the structure of a system by showing the classes, their attributes, and the relationships between the classes.
* Sequence diagrams: A sequence diagram represents the behavior of a system by showing the interactions between objects in chronological order.
* State machine diagrams: A state machine diagram represents the behavior of an object or a system by showing the possible states and transitions between the states.
* Activity diagrams: An activity diagram represents the flow of activities or processes in a system.

To create UML diagrams, many software tools are available that support UML modelings, such as Microsoft Visio, StarUML, and Visual Paradigm. UML diagrams can also be created by hand, but using software tools makes the process faster and more efficient.

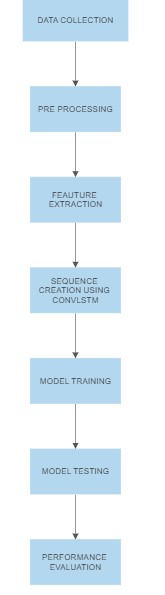
UML is a valuable tool for software development as it provides a common language and visual representation for software systems, which can help to reduce misunderstandings and miscommunications among team members. It can also help in the design and development process by providing a clear and organized view of the system being developed.

## 6.3 FLOWCHARTS

### 6.3.1 FLOWCHART FOR LRCN APPROACH



### 6.3.2 FLOWCHART FOR CONVLSTM APPROACH



## 6.4 BUILDING BLOCKS OF UML

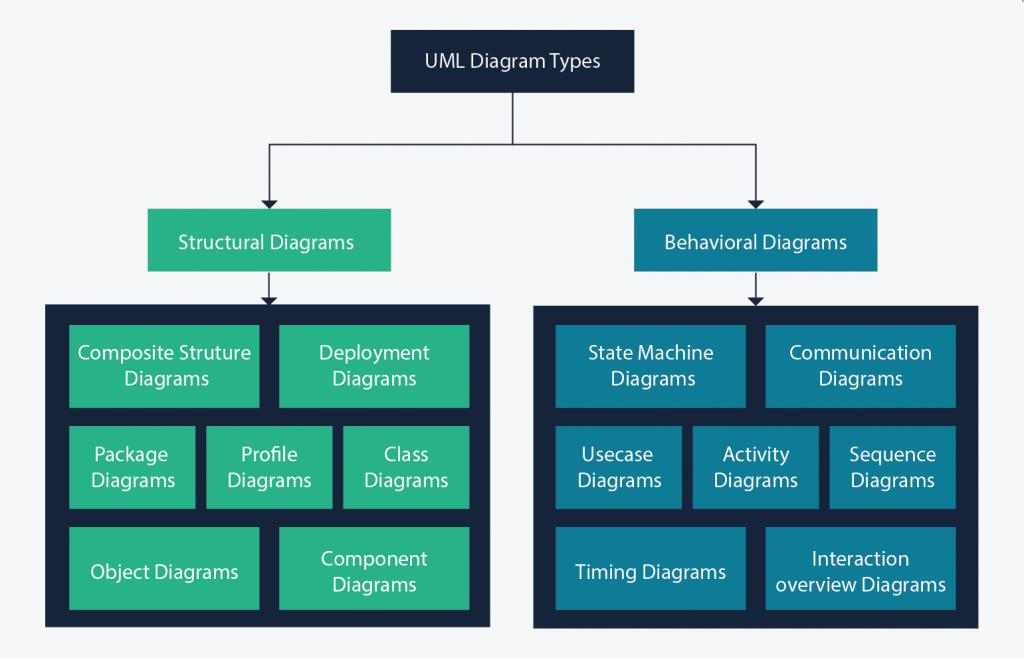
UML is linked with object-oriented design and analysis. UML makes use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as

1. **Structural Diagrams** – Capture static aspects or structures of a system. Structural

Diagrams include Component Diagrams, Object Diagrams, Class Diagrams, and Deployment Diagrams.

1. **Behaviour Diagrams** – Capture dynamic aspects or behavior of the system. Behavior diagrams include: Use Case Diagrams, State Diagrams, Activity Diagrams, and Interaction Diagrams.

THE IMAGE BELOW SHOWS THE HIERARCHY OF DIAGRAMS ACCORDING TO UML



OBJECT ORIENTED CONCEPTS USED IN UML

**Class** – A class defines the blue print i.e. structure and functions of an object.

**Objects** – Objects help us to decompose large systems and help us to modularize our system. Modularity helps to divide our system into understandable components so that we can build our system piece by piece. An object is the fundamental unit (building block) of a system which is used to depict an entity.

**Inheritance** – Inheritance is a mechanism by which child classes inherit the properties of their parent classes.

**Abstraction** – Mechanism by which implementation details are hidden from the user.

**Encapsulation** – Binding data together and protecting it from the outer world is

referred to as encapsulation.

**Polymorphism** – Mechanism by which functions or entities are able to exist in

different forms.

## 6.5 UML DIAGRAMS

**Class Diagram** – The most widely use UML diagram is the class diagram. It is the

building block of all object-oriented software systems. We use class diagrams to depict the static structure of a system by showing system’s classes, their methods and attributes. Class diagrams also help us identify relationship between different classes or objects.

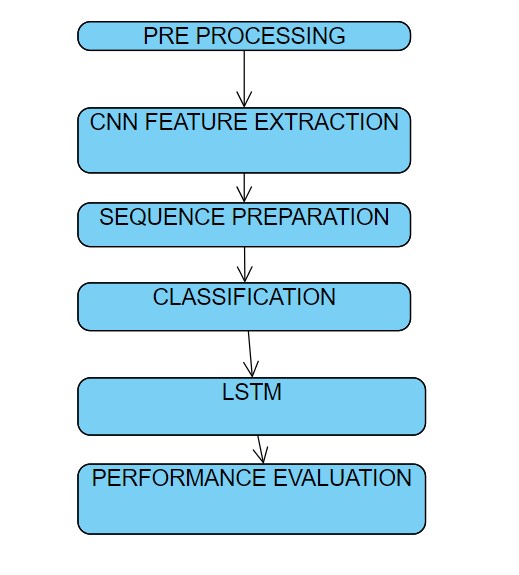
1. **Composite Structure Diagram** – We use composite structure diagrams to represent the internal structure of a class and its interaction points with other parts of the system. A composite structure diagram represents relationship between parts and their configuration which determine how the classifier (class, a component, or a deployment node) behaves. They represent internal structure of a structured classifier making the use of parts, ports, and connectors. We can also model collaborations using composite structure diagrams. They are similar to class diagrams except they represent individual parts in detail as compared to the entire class.
2. **Object Diagram** – An Object Diagram can be referred to as a screenshot of the instances in a system and the relationship that exists between them. Since object diagrams depict behaviour when objects have been instantiated, we are able to study the behaviour of the system at a particular instant. An object diagram is similar to a class diagram except it shows the instances of classes in the system. We depict actual classifiers and their relationships making the use of class diagrams. On the other hand, an Object Diagram represents specific instances of classes and relationships between them at a point of time.
3. **Component Diagram** – Component diagrams are used to represent the how the physical components in a system have been organized. We use them for modelling implementation details. Component Diagrams depict the structural relationship between software system elements and help us in understanding if functional requirements have been covered by planned development. Component Diagrams become essential to use when we design and build complex systems. Interfaces are used by components of the system to communicate with each other.

**4.Deployment Diagram** – Deployment Diagrams are used to represent system hardware and its software. It tells us what hardware components exist and what software components run on them. We illustrate system architecture as distribution of software artifacts over distributed targets. An artifact is the information that is generated by system software. They are primarily used when a software is being used, distributed or deployed over multiple machines with different configurations.

**5. Package Diagram** – We use Package Diagrams to depict how packages and their elements have been organized. A package diagram simply shows us the dependencies between different packages and internal composition of packages. Packages help us to organize UML diagrams into meaningful groups and make the diagram easy to understand.

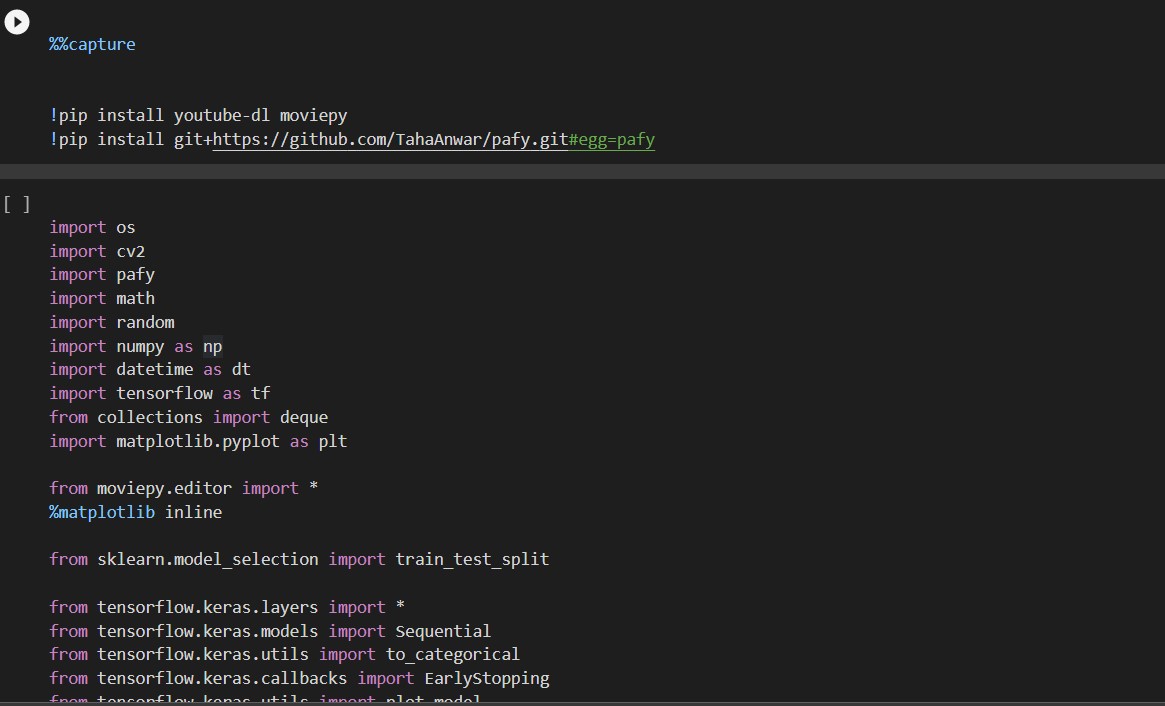
They are primarily used to organize diagrams

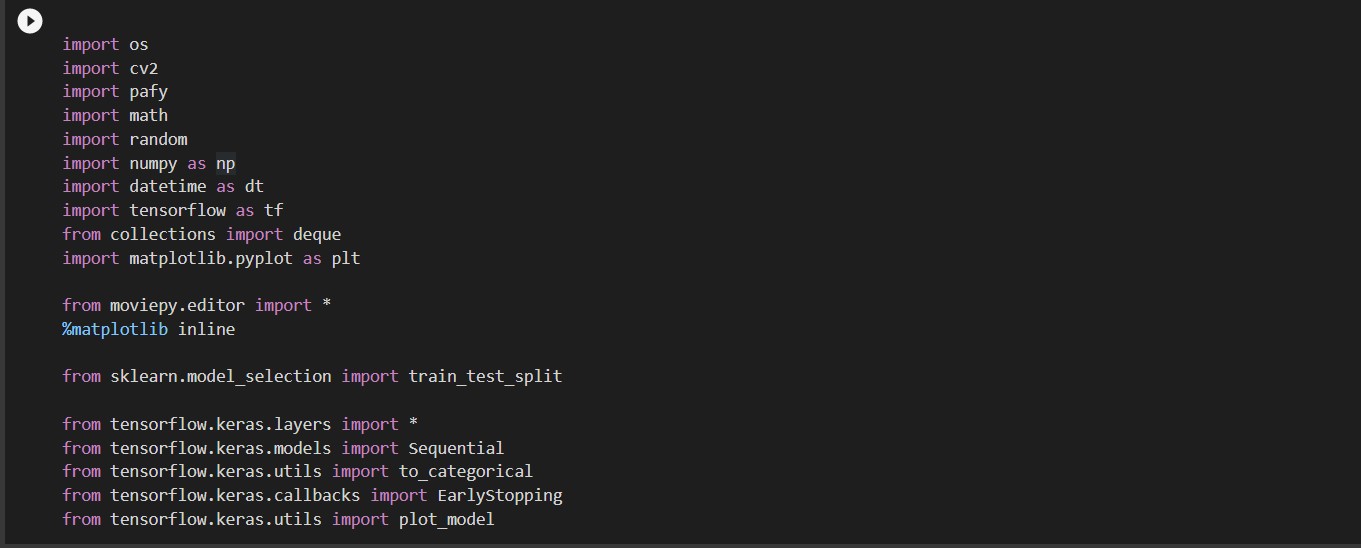
**UML DIAGRAM FOR LRCN/ConvLSTM APPROACH:**

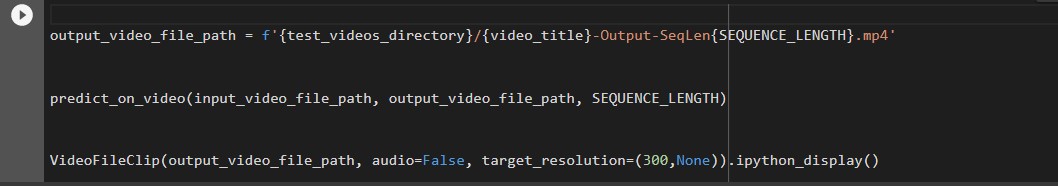
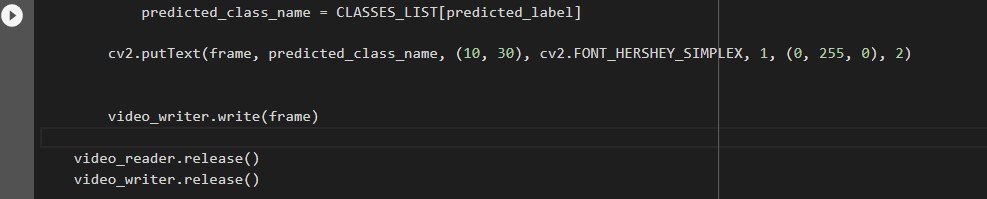
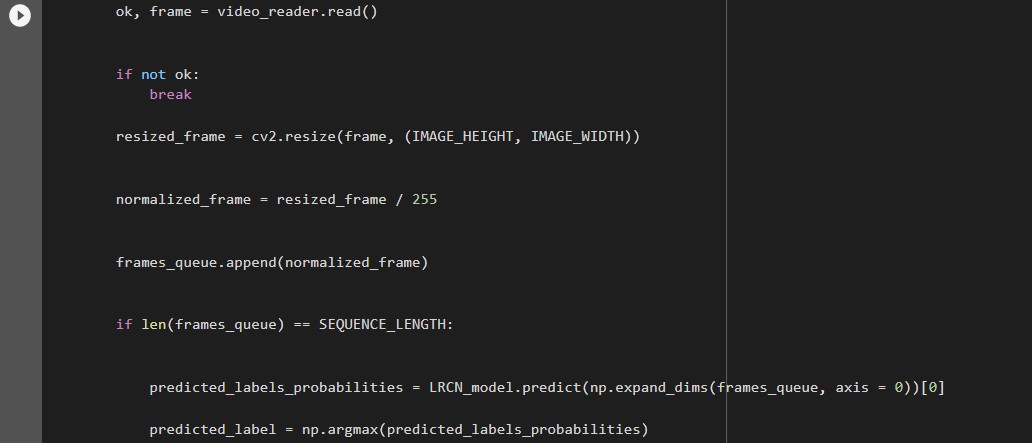
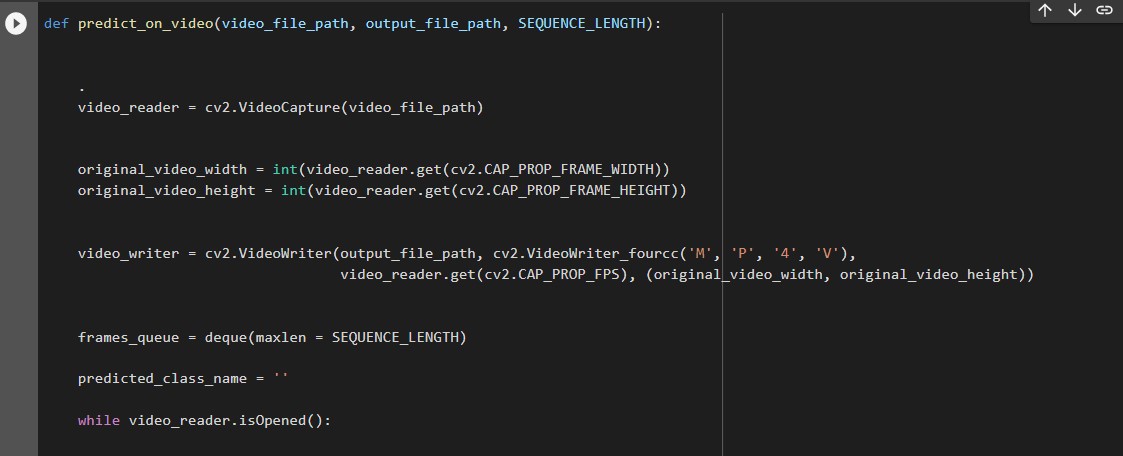
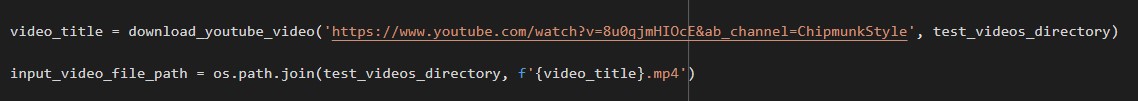
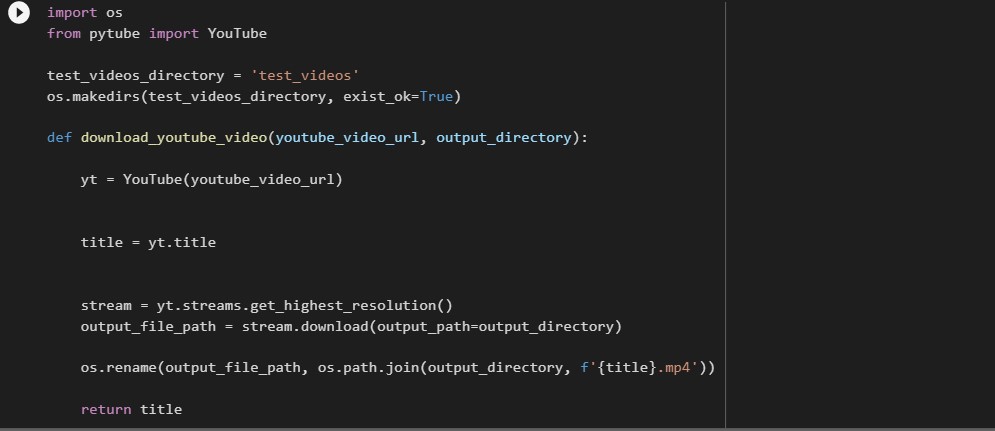
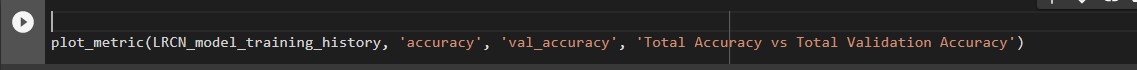
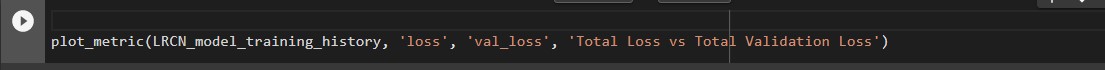
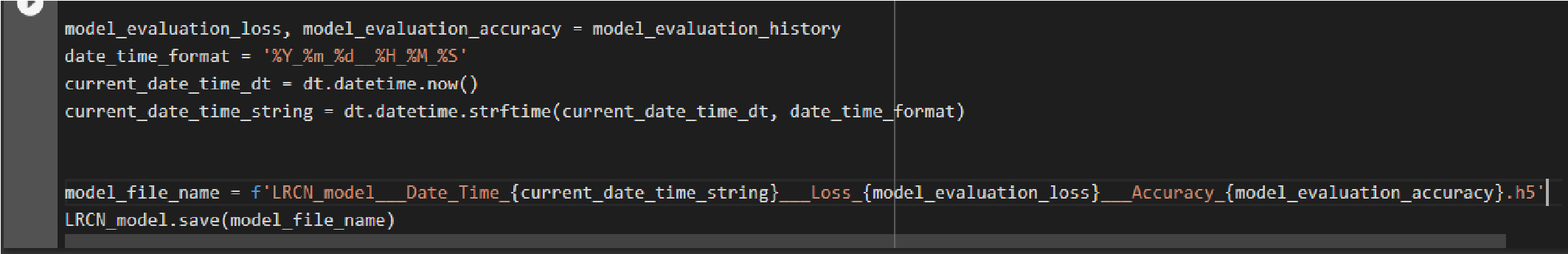
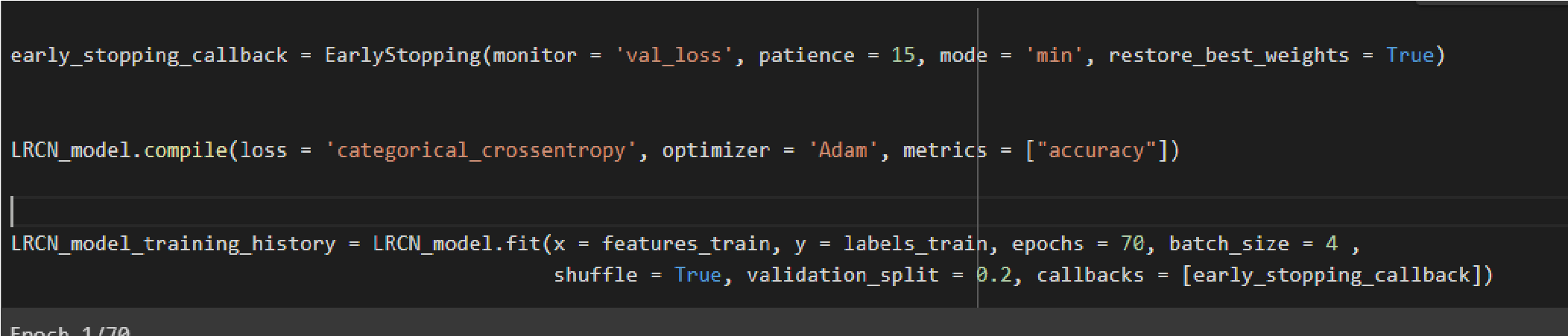
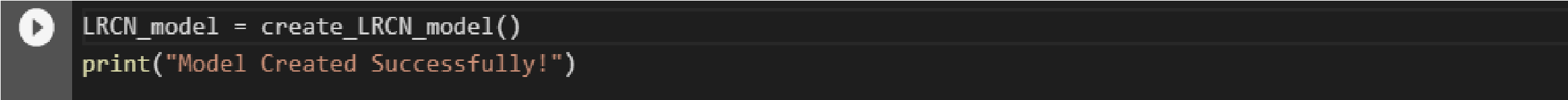
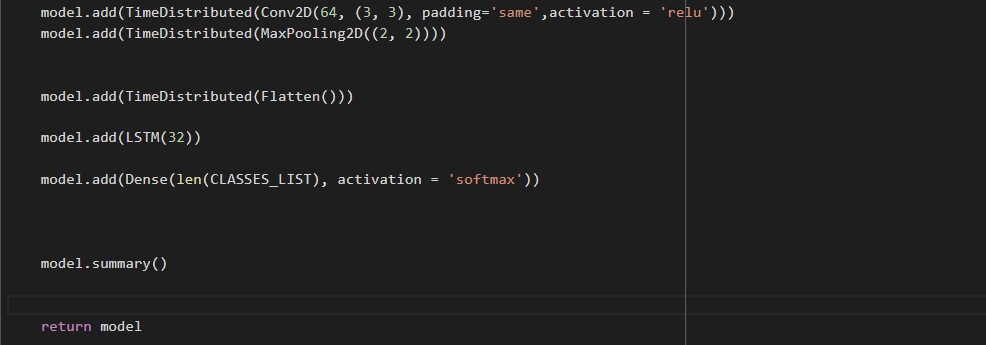
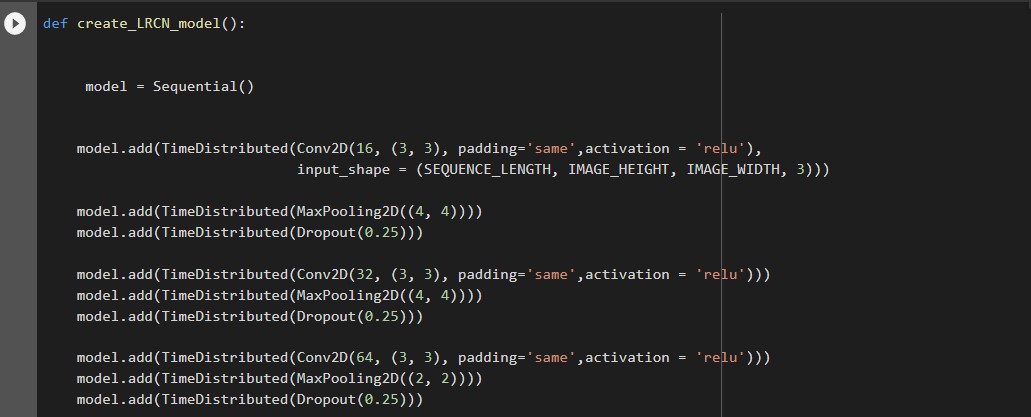
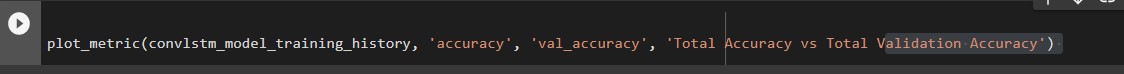
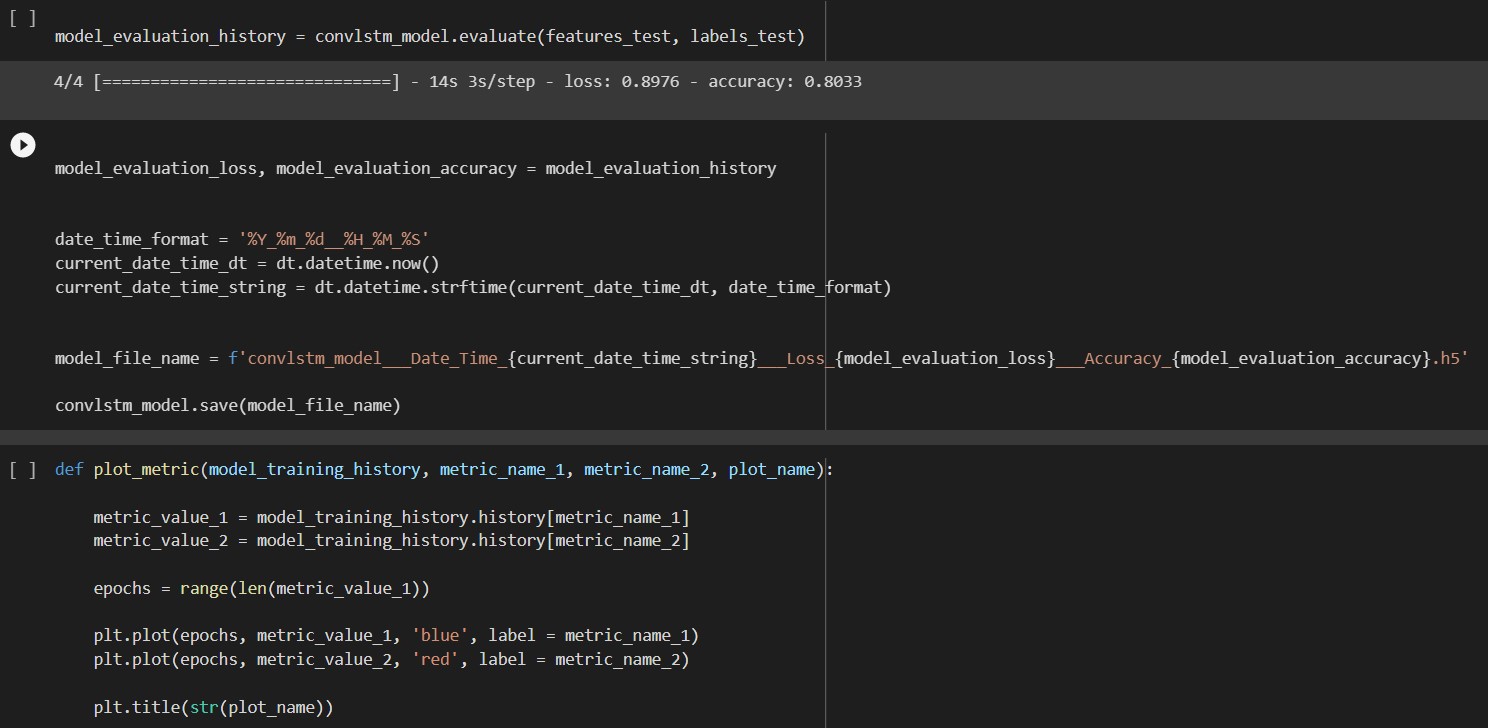
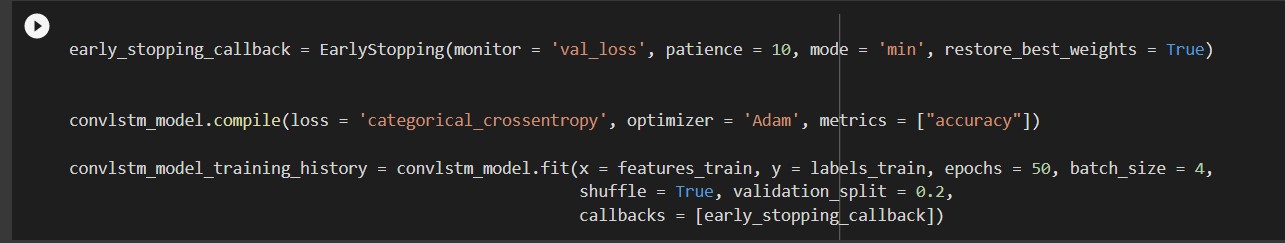
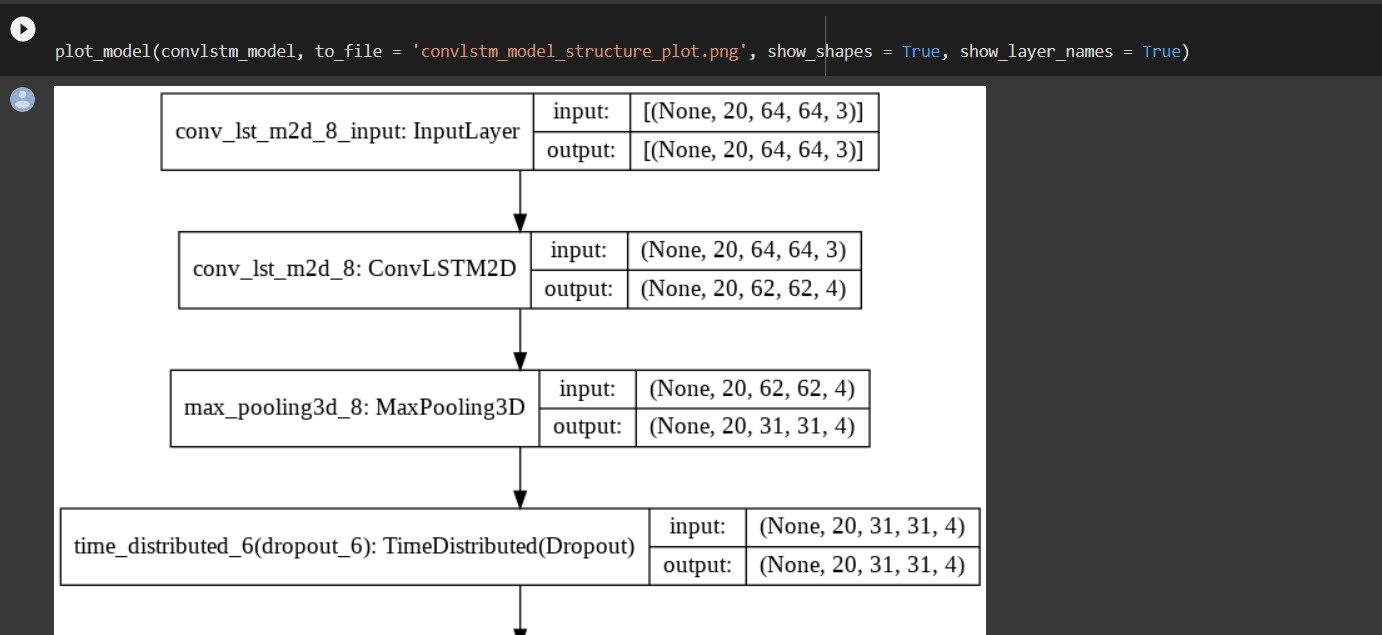
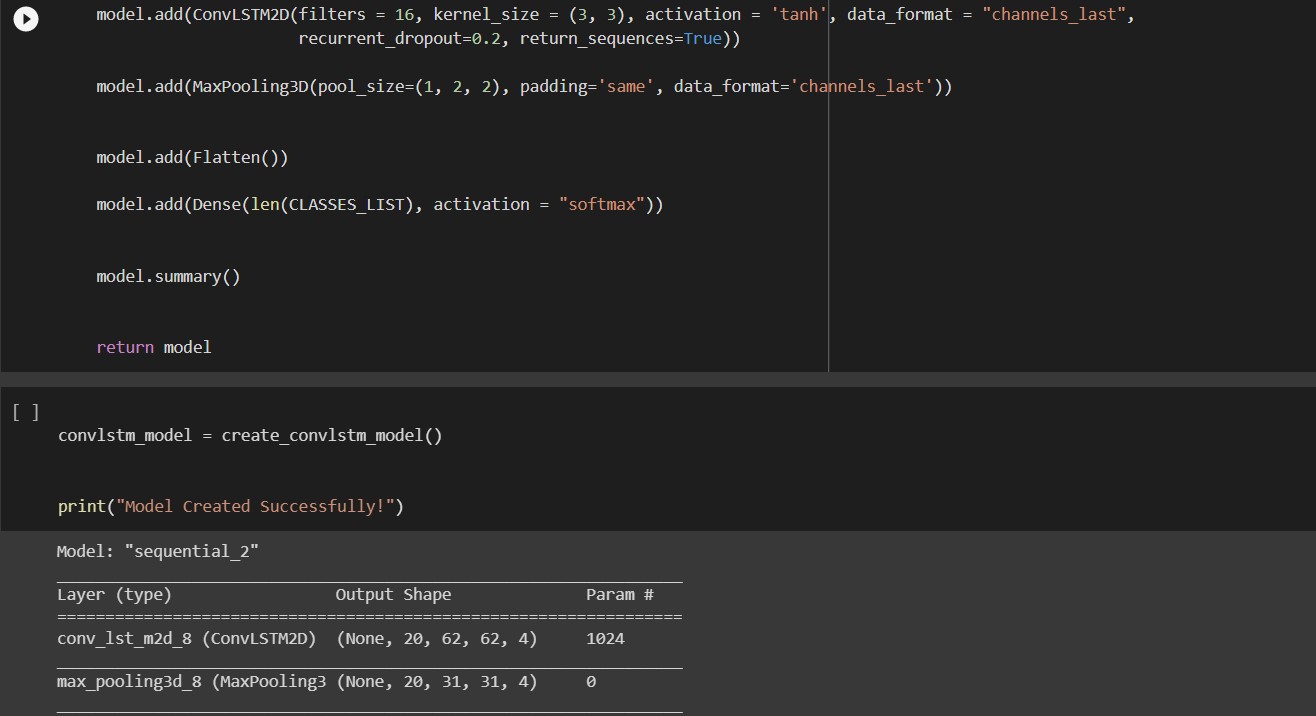
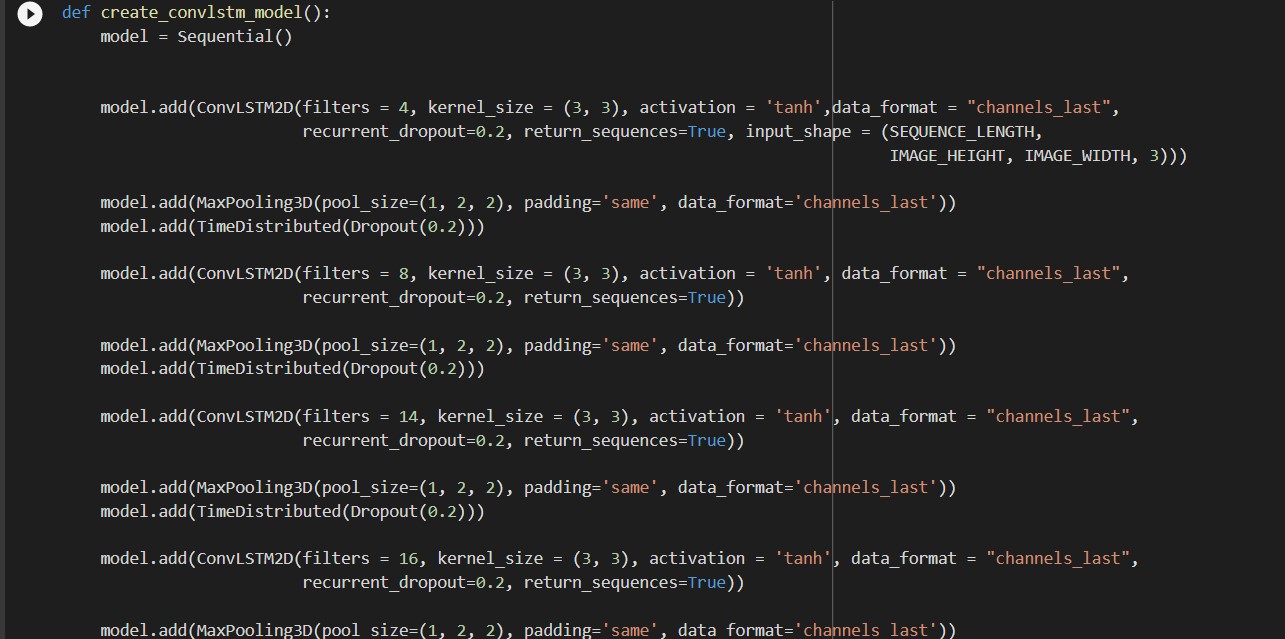
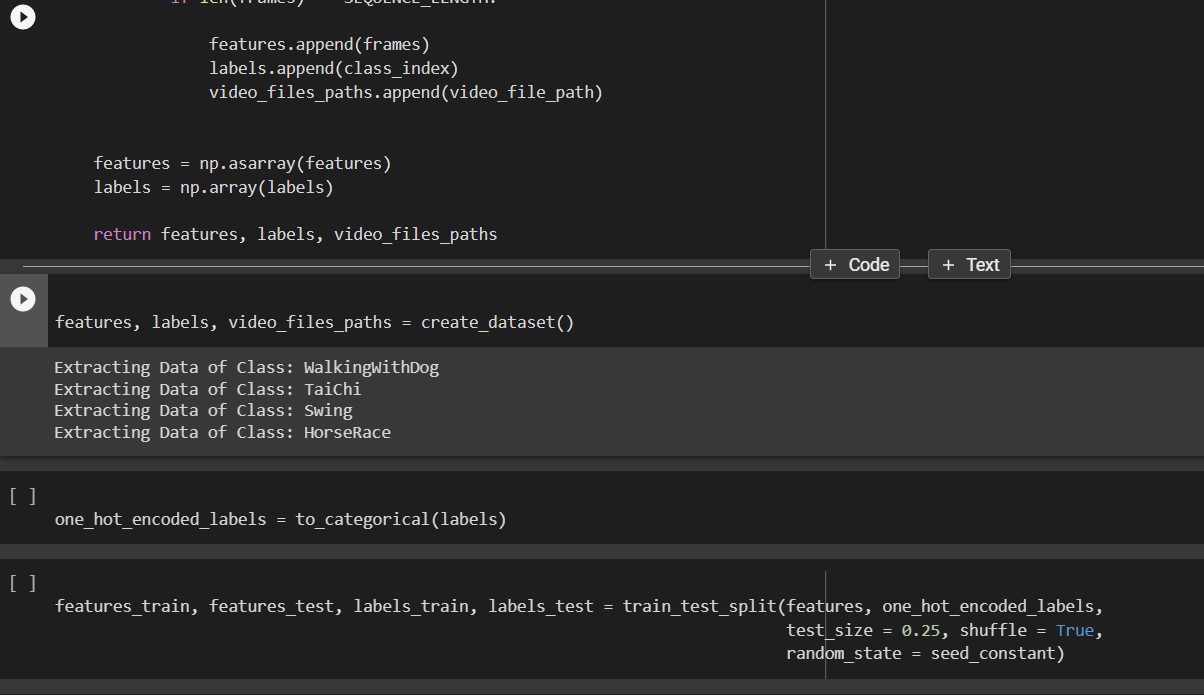
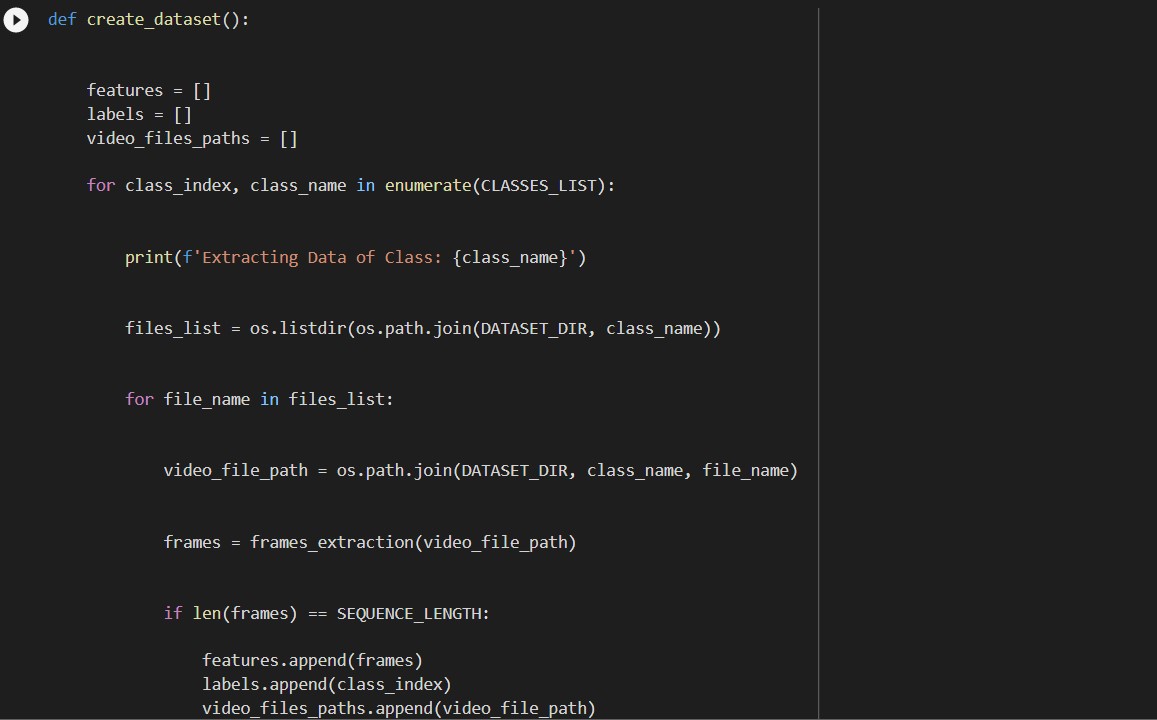
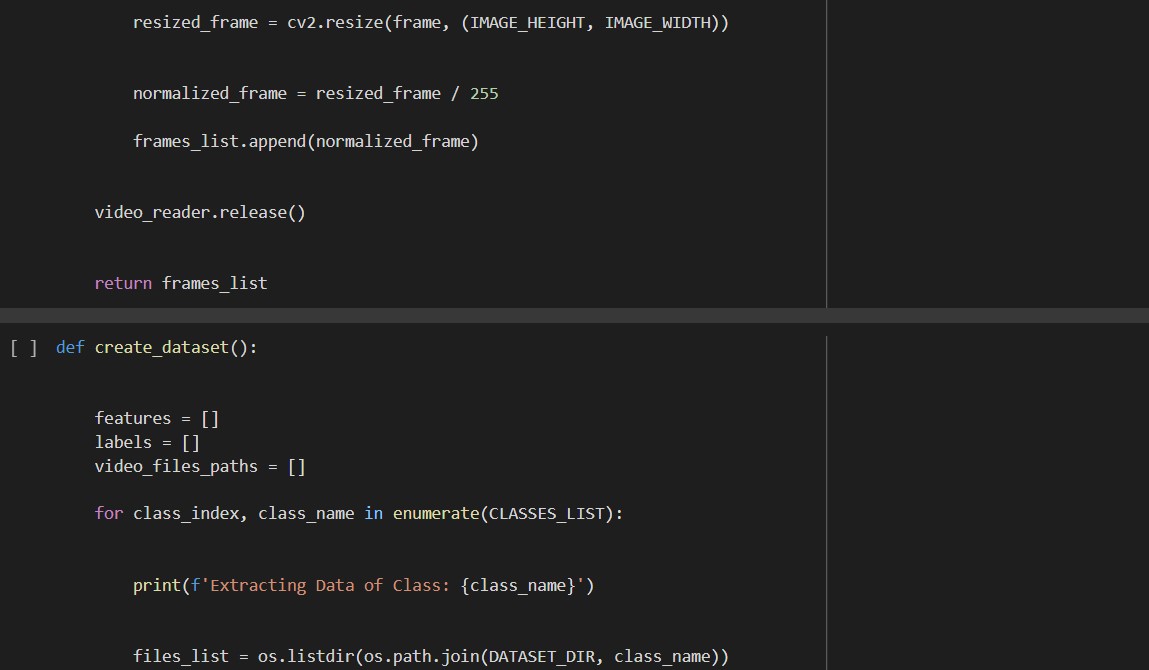
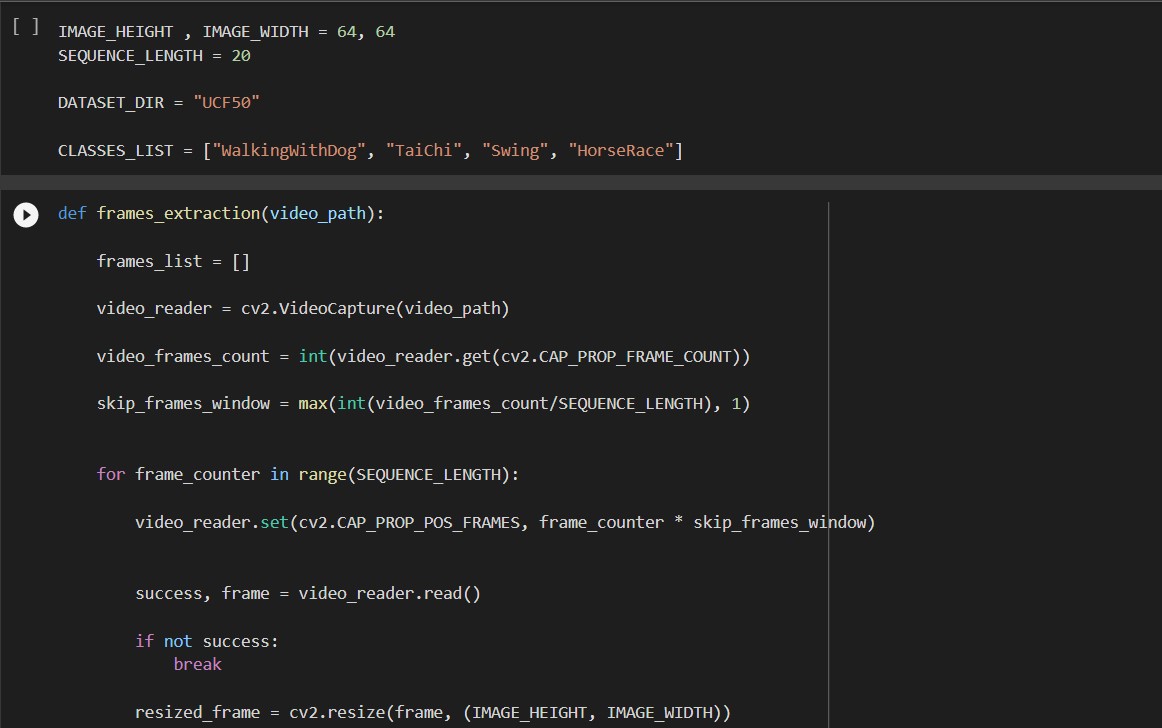
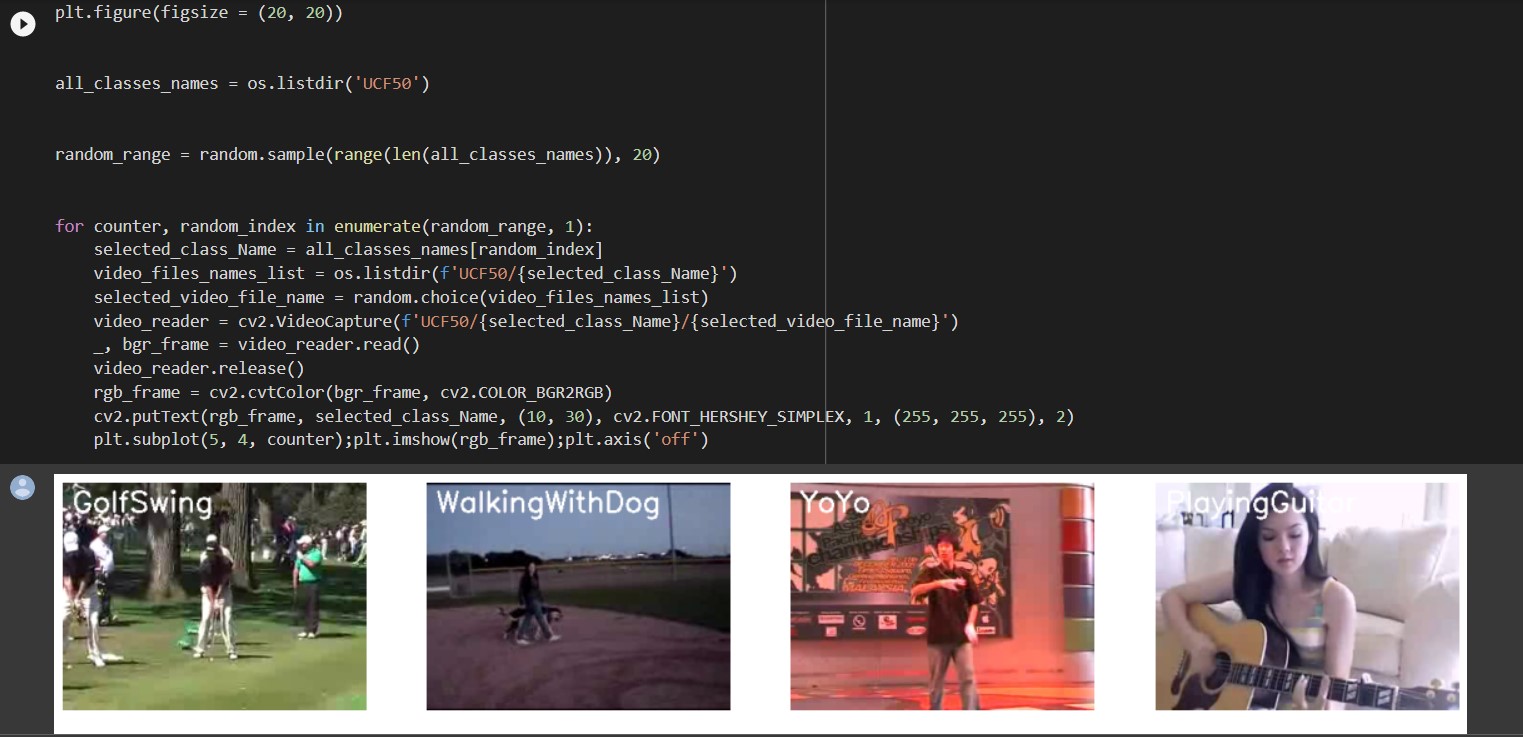


# 7. DEVELOPMENT

7.1 SAMPLE CODE







**7.2 RESULT:**





**7.3 CONCLUSION:**

In conclusion, the combination of LSTM and CNN models has proven to be a successful approach for human action detection. The CNN model is responsible for capturing the spatial features of the video frames, while the LSTM model captures the temporal features by analyzing the sequence of frames over time. This joint architecture has the ability to learn complex temporal dependencies and extract discriminative features for action recognition.

By training the model on large-scale datasets such as UCF101 and HMDB51, it has been shown that this approach achieves state-of-the-art results in terms of accuracy and efficiency. The model can effectively recognize various human actions in real-time video streams, making it applicable to a wide range of applications such as surveillance, human-computer interaction, and robotics.

Overall, the combination of LSTM and CNN models is a promising direction for the field of human action detection and has the potential to unlock further advancements in this area.

**7.4 FUTURE SCOPE:**

Human action detection using LSTM and CNN is a rapidly developing field with enormous potential for the future. LSTM (Long Short-Term Memory) is a type of neural network that is particularly well-suited for analyzing sequences of data, while CNN (Convolutional Neural Network) is a deep learning architecture commonly used for image and video analysis. By combining these two powerful techniques, researchers have made great strides in accurately detecting and classifying human actions in video data.

One of the most promising areas for future development is in real-time human action detection for use in applications such as autonomous vehicles and robotics. By training deep neural networks on vast amounts of video data, researchers can create models that are capable of detecting and classifying human actions in real-time, allowing autonomous systems to react quickly and effectively to changes in their environment.

Another area of future development is in the use of human action detection for security and surveillance applications. By analyzing video data from security cameras, deep learning models can be trained to detect and classify suspicious or criminal behavior, allowing for more efficient and effective monitoring of public spaces.

**7.5 BIBLIOGRAPHY:**

https://towardsdatascience.com/image-analysis-for-beginners-creating-a-motion-detector-with-o pencv-4ca6faba4b42

https://pyimagesearch.com/2015/05/25/basic-motion-detection-and-tracking-with-python-and-op encv/

https://thedatafrog.com/en/articles/human-detection-video/

https://www.researchgate.net/publication/359368941\_An\_Intelligent\_MotionDetection\_Using\_

OpenCV

45 46