SafetyNet

A Threat Detection System using Artificial Neural Networks

21/06/2023

CORPORATE GURUKUL

We, The Team!

VIRAJ PARMAJ
Data Preprocessing
+ Data Preparation

MRIDUL SANGHAVI
Model Building
Model Preprocessing

M. ANANYA RAJU
Model Building
+ Deployment

ARYAMAN DEV
Hyperparameter Tuning
+ Business Logic

HRIDAY GUPTA
Data Collection
+ UI/UX

SNEHA AGARWAL
EDA
+ FrontEnd, Business Logic

CORPORATE
GURUKUL

Index

Project Outline

Problem Statement

To build a model that states whether a person is a threat or not on the basis of pose detection

Stakeholders

Police Department, Court of Law, Security Companies

Approach

Dividing video data into frames and running CNN for a predetermined sequence length, whose output is run through an RNN model (LSTM) to read frame data

Assumption

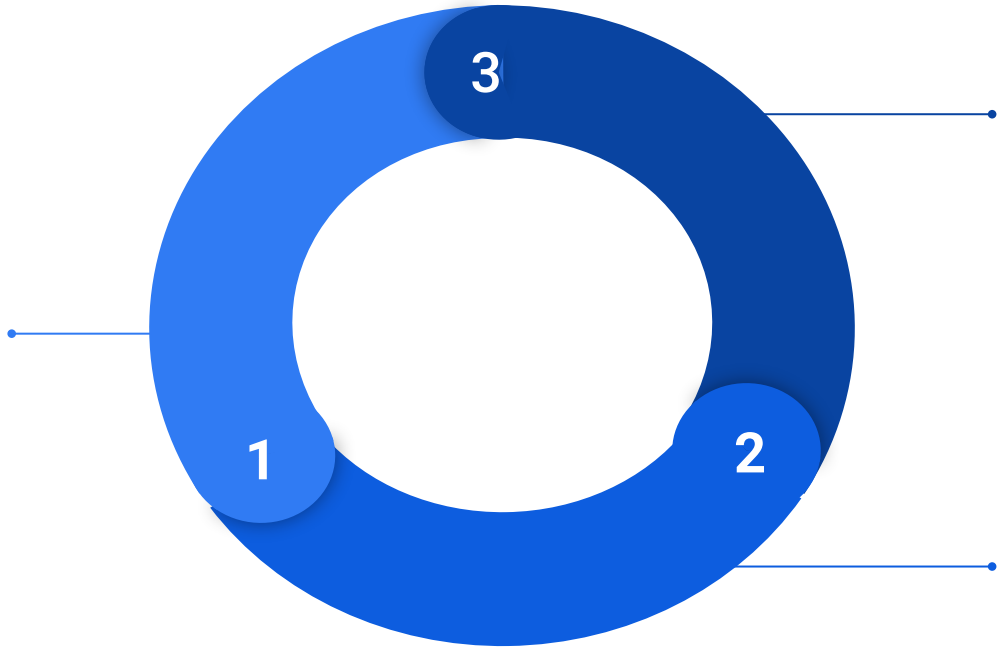
Project will aim to predict degree of violence of an individual based on actions

Motivation

- Help law enforcement officials accurately **assess intentions** of individuals during patrols
- To **prevent accidental loss of lives** of both, police officers and innocent civilians
- Enhancing **safety and security**

ASSESS INTENTIONS

Classifying whether the intentions are threatening or non threatening



PREVENT ACCIDENTAL LOSS OF LIVES

Loss of life due to shortcomings in judgements could be avoided

SAFETY & SECURITY

This idea collectively increases the safety and security of an individual be it the law or a common civilian

CORPORATE GURUKUL

What we are trying to build?

- SafetyNet, an LSTM-based Model, is built with the aim to identify and separate violent actions from non-threatening ones.
- Our goal is to develop a deep learning model that detects suspicious activities through police body-cams and live CCTV footage, promptly alerting authorities.

Abstract

PURPOSE

- To detect the actions and poses of an individual and hence, their violent / non-violent intentions

ARCHITECTURE

- Use of the deep learning model LSTM for detection and classification of human actions

VALUE

- SafetyNet aims to help law enforcement and civilians by preventing accidental loss of lives

Business Use Case

- An estimated 250,000 civilians are injured by law enforcement officers annually, with over 1000 cases leading to death
- Due to racism, both Black/African-Americans and Hispanics/Latinos are twice as likely to experience threat of or use of force during police initiated contact

Source: https://policeviolencereport.org/

With SafetyNet

- **Hold police officials more accountable for their unlawful actions**
- **Prevent unnecessary violent interactions and misunderstanding between the police and civilians**

CORPORATE GURUKUL

Important Use Case

- **SafetyNet devices**

  SafetyNet leverages computer vision and pose estimation techniques to analyze real-time body poses and movements, enabling the detection of weapons and suspicious behavior in individuals.

- **Live CCTV footages**

  Additionally, SafetyNet can be applied for crowd monitoring purposes, providing insights into crowd behavior and facilitating the identification of potential threats or abnormal activities.

Concepts Used

- **Data Pre-processing**
  → Frame Extraction & conversion using OpenCV
  → Videos were standardized to have the same time frame of 3 seconds.

- **Classification Techniques**
  → Binary Classification
  → Categorical Classification

- **Model Building**
  → LRCN (multi-label)
  → ConvLSTM (multi-label)
  → BiLSTM (binary classification)

- **Optimization / HyperParameter Tuning**
  → Varying Epochs and Batch size
  → Changing activation functions and optimizers
  → Switching dropout values

Our Data

- Serre Lab's large **Human Motion DataBase** was used to implement SafetyNet's ConvLSTM and LRCN models
  - HMDB51 is a database of 51 labelled human actions
  - Each action consists of about 100-150 distinct videos

- Additionally,
  Relevant datasets like **UCF50** and **Real Life Violence Dataset** were used to train the BiLSTM model
  - UCF50 has a dataset of 50 labelled human actions with 150 videos in each action
  - Real Life Violence Dataset contains 1000 videos each of real-life violent and non-violent incidents

Exploratory Data Analysis

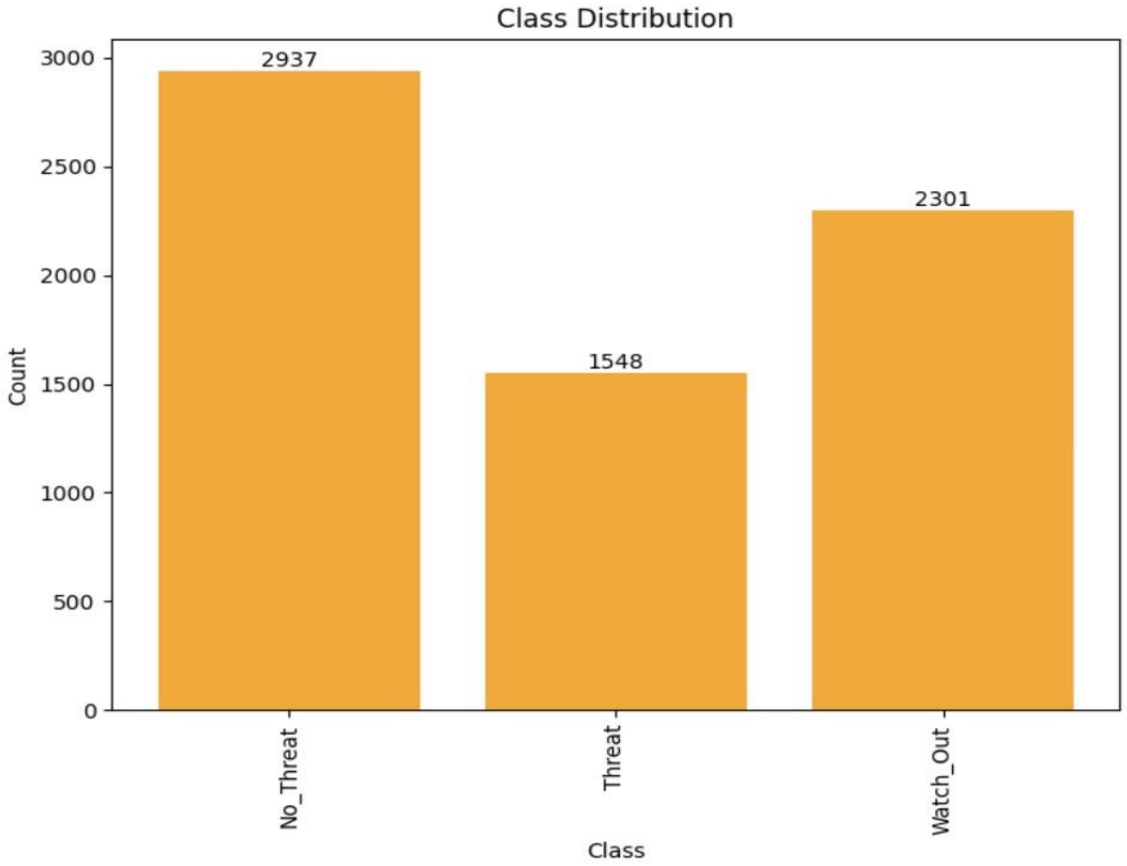
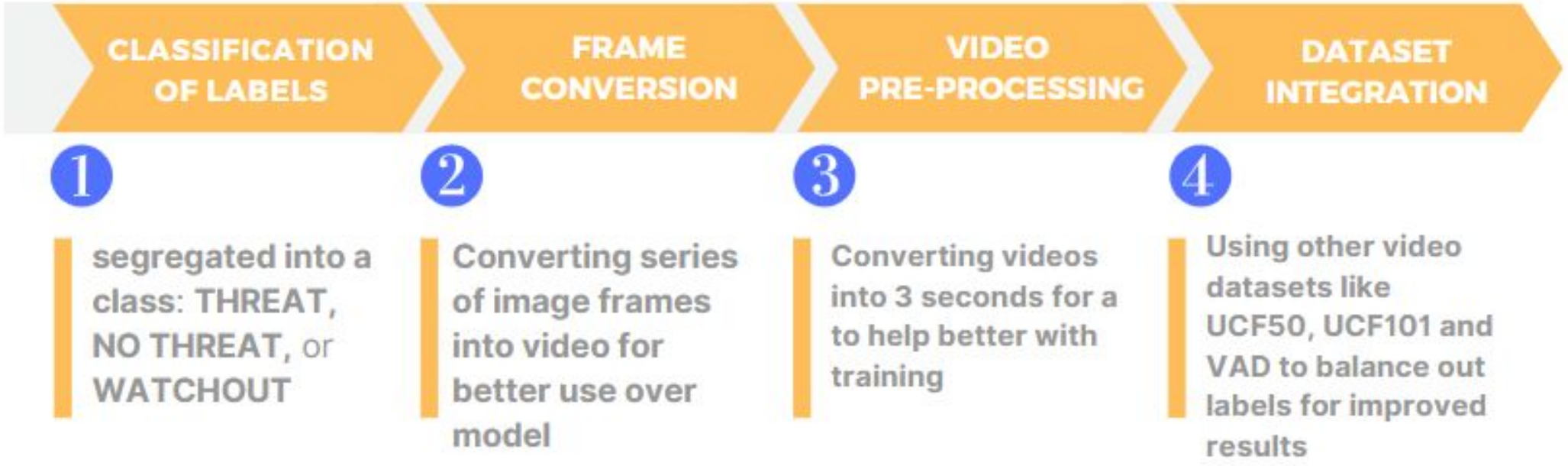


Fig: Class Distribution in Dataset Used

How we Processed Data



CORPORATE GURUKUL

How we Processed Data

1. Every action was **segregated into a class**: **THREAT**, **NO THREAT**, or **WATCHOUT**
2. **Conversion of frames into videos** to fit our model and provide better accuracy 1,21,000 images into 7000 videos

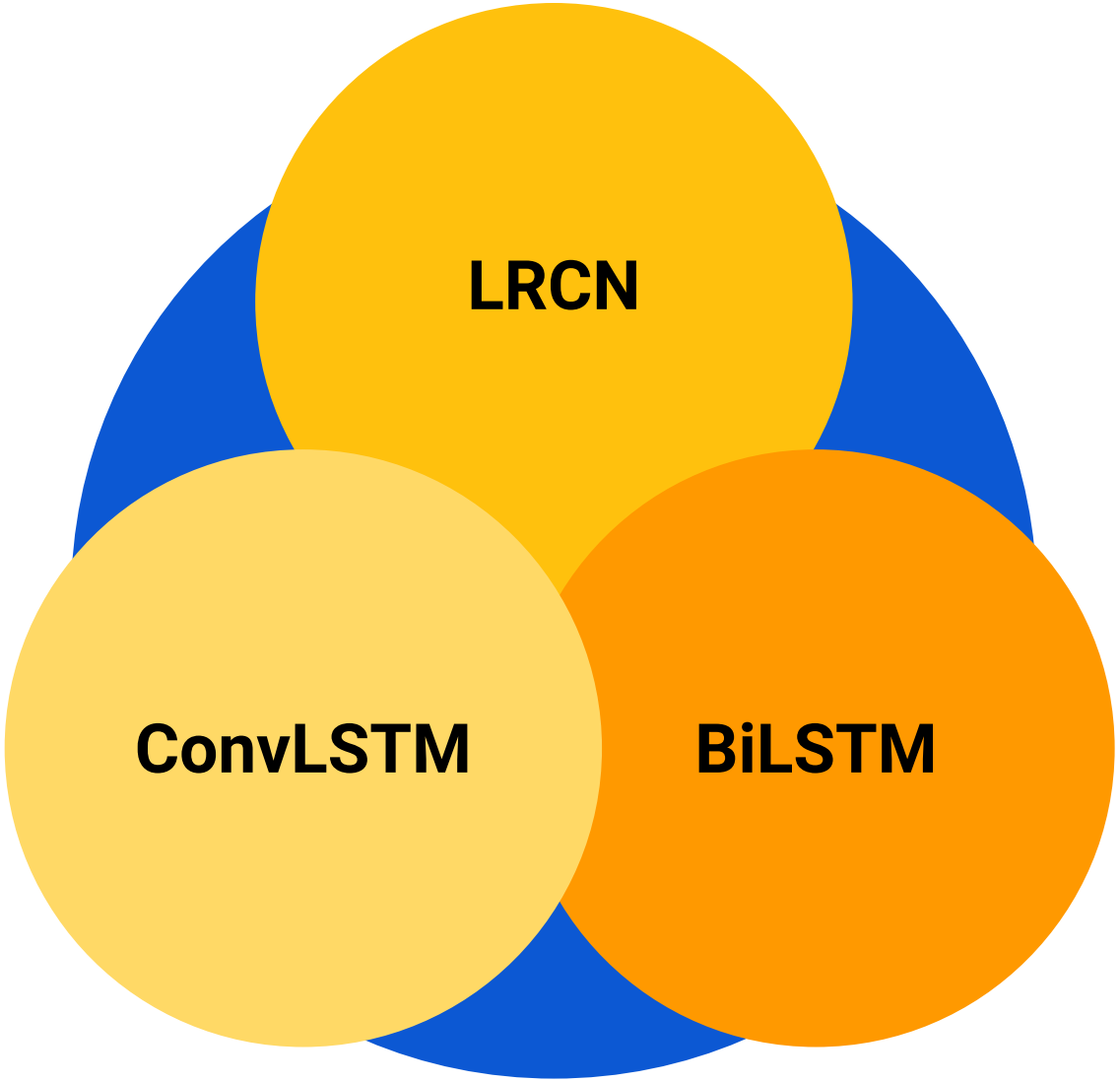
   A Python code using OpenCV was implemented to convert a collection of images into video files. It iterates through subfolders, reads image files, and creates videos with desired durations

   - The code calculates frames per second based on the number of images and desired duration
   - It then writes each image to the video file and saves it in the output directory
   - The process involves file operations, image manipulation, and video encoding/decoding

We integrated other relevant datasets like the UCF50 and VID (Violent Action Dataset) to improve accuracy and model prediction by balancing videos under listed labels

Algorithms Used

1. ConvLSTM
2. LRCN
3. BiLSTM



ConvLSTM

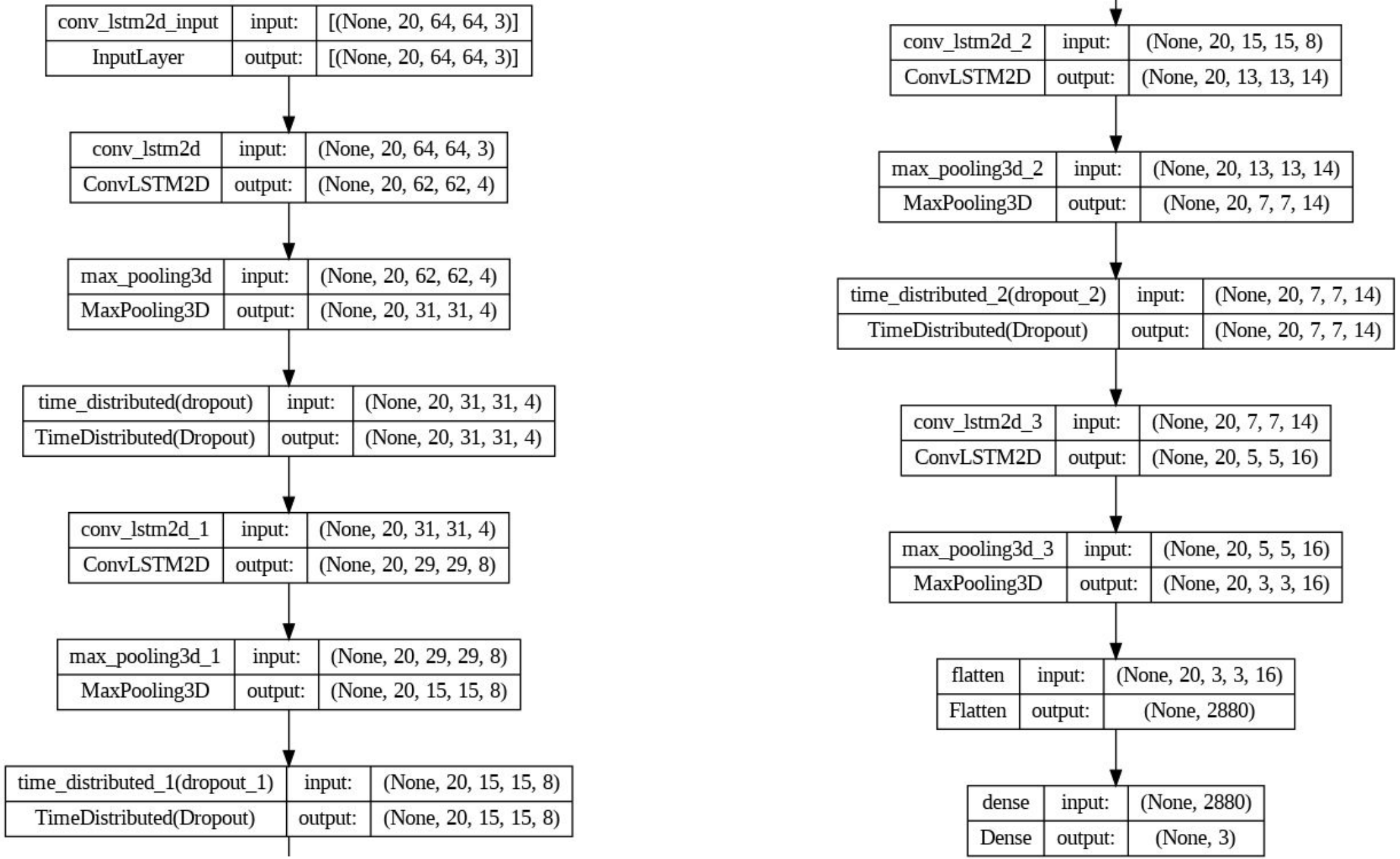
1. **Implementation of ConvLSTM Model:**

    a. **ConvLSTM2D**
    b. **MaxPooling3D Layer**
    c. **Dropout Layer**
    d. **Flattening**
    e. **SoftMax Layer**

The loss function is Categorical Cross Entropy
The model performs a Categorical Classification task
Activation function used in tanh function

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv_lstm2d (ConvLSTM2D)    (None, 20, 62, 62, 4)     1024

 max_pooling3d (MaxPooling3D  (None, 20, 31, 31, 4)    0
 )

 time_distributed (TimeDistr  (None, 20, 31, 31, 4)    0
 ibuted)

 conv_lstm2d_1 (ConvLSTM2D)  (None, 20, 29, 29, 8)     3488

 max_pooling3d_1 (MaxPooling  (None, 20, 15, 15, 8)    0
 3D)

 time_distributed_1 (TimeDis  (None, 20, 15, 15, 8)    0
 tributed)

 conv_lstm2d_2 (ConvLSTM2D)  (None, 20, 13, 13, 14)    11144

 max_pooling3d_2 (MaxPooling  (None, 20, 7, 7, 14)     0
 3D)

 time_distributed_2 (TimeDis  (None, 20, 7, 7, 14)     0
 tributed)

 conv_lstm2d_3 (ConvLSTM2D)  (None, 20, 5, 5, 16)      17344

 max_pooling3d_3 (MaxPooling  (None, 20, 3, 3, 16)     0
 3D)

 flatten (Flatten)           (None, 2880)              0

 dense (Dense)               (None, 3)                 8643

=================================================================
Total params: 41,643
Trainable params: 41,643
Non-trainable params: 0
_________________________________________________________________
Model Created
```

CORPORATE GURUKUL


conv_lstm2d_input | input: | [(None, 20, 64, 64, 3)]
InputLayer | output: | [(None, 20, 64, 64, 3)]
conv_lstm2d | input: | (None, 20, 64, 64, 3)
ConvLSTM2D | output: | (None, 20, 62, 62, 4)
max_pooling3d | input: | (None, 20, 62, 62, 4)
MaxPooling3D | output: | (None, 20, 31, 31, 4)
time_distributed(dropout) | input: | (None, 20, 31, 31, 4)
TimeDistributed(Dropout) | output: | (None, 20, 31, 31, 4)
conv_lstm2d_1 | input: | (None, 20, 31, 31, 4)
ConvLSTM2D | output: | (None, 20, 29, 29, 8)
max_pooling3d_1 | input: | (None, 20, 29, 29, 8)
MaxPooling3D | output: | (None, 20, 15, 15, 8)
time_distributed_1(dropout_1) | input: | (None, 20, 15, 15, 8)
TimeDistributed(Dropout) | output: | (None, 20, 15, 15, 8)
conv_lstm2d_2 | input: | (None, 20, 15, 15, 8)
ConvLSTM2D | output: | (None, 20, 13, 13, 14)
max_pooling3d_2 | input: | (None, 20, 13, 13, 14)
MaxPooling3D | output: | (None, 20, 7, 7, 14)
time_distributed_2(dropout_2) | input: | (None, 20, 7, 7, 14)
TimeDistributed(Dropout) | output: | (None, 20, 7, 7, 14)
conv_lstm2d_3 | input: | (None, 20, 7, 7, 14)
ConvLSTM2D | output: | (None, 20, 5, 5, 16)
max_pooling3d_3 | input: | (None, 20, 5, 5, 16)
MaxPooling3D | output: | (None, 20, 3, 3, 16)
flatten | input: | (None, 20, 3, 3, 16)
Flatten | output: | (None, 2880)
dense | input: | (None, 2880)
Dense | output: | (None, 3)

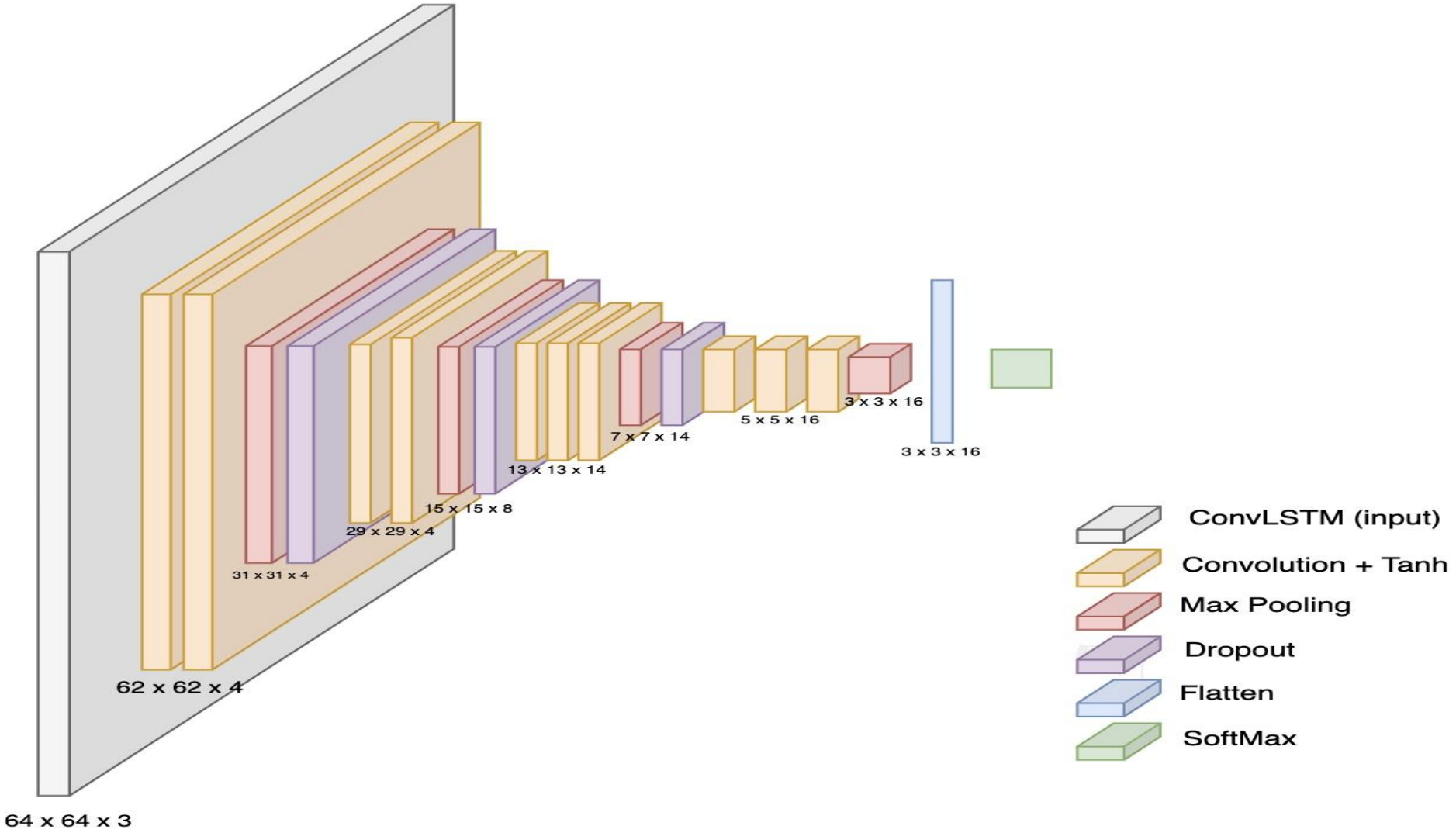


Fig: ConvLSTM Model Architecture Diagram

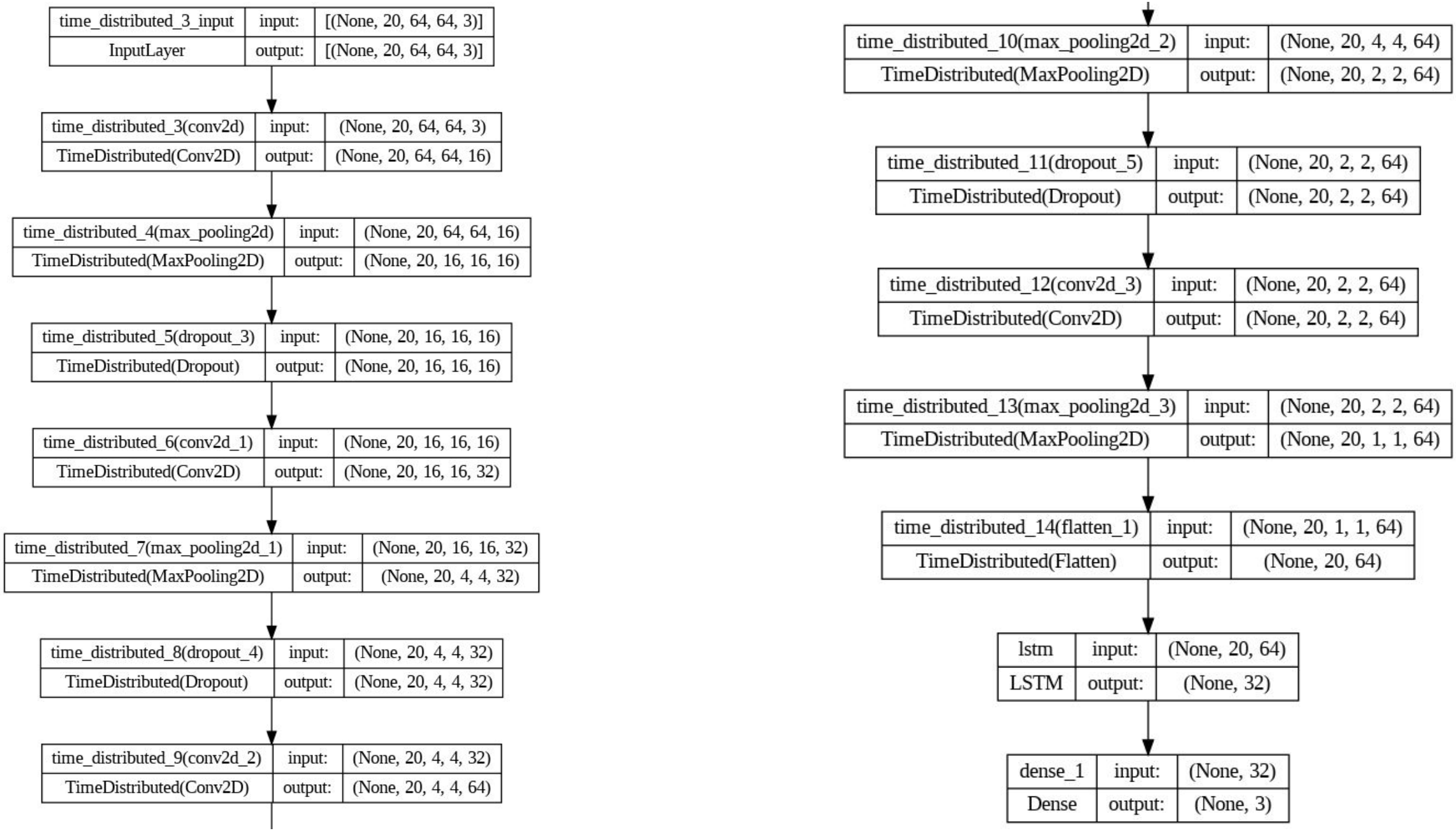
LRCN

2. Implementation of LRCN Model:

    a. Conv2D
    b. MaxPooling2D Layer
    c. Dropout Layer
    d. Flattening
    e. SoftMax Layer

The loss function is Categorical Cross Entropy
The model performs a Categorical Classification task
Activation function used in ReLU

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 time_distributed_68 (TimeDi  (None, 20, 64, 64, 16)   448
 stributed)

 time_distributed_69 (TimeDi  (None, 20, 16, 16, 16)   0
 stributed)

 time_distributed_70 (TimeDi  (None, 20, 16, 16, 16)   0
 stributed)

 time_distributed_71 (TimeDi  (None, 20, 16, 16, 32)   4640
 stributed)

 time_distributed_72 (TimeDi  (None, 20, 4, 4, 32)     0
 stributed)

 time_distributed_73 (TimeDi  (None, 20, 4, 4, 32)     0
 stributed)

 time_distributed_74 (TimeDi  (None, 20, 4, 4, 64)     18496
 stributed)

 time_distributed_75 (TimeDi  (None, 20, 2, 2, 64)     0
 stributed)

 time_distributed_76 (TimeDi  (None, 20, 2, 2, 64)     0
 stributed)

 time_distributed_77 (TimeDi  (None, 20, 2, 2, 64)     36928
 stributed)

 time_distributed_78 (TimeDi  (None, 20, 1, 1, 64)     0
 stributed)

 time_distributed_79 (TimeDi  (None, 20, 1, 1, 64)     0
 stributed)

 time_distributed_80 (TimeDi  (None, 20, 64)           0
 stributed)

 lstm_2 (LSTM)               (None, 32)                12416

 dense_3 (Dense)             (None, 3)                 99

=================================================================
Total params: 73,027
Trainable params: 73,027
Non-trainable params: 0
_________________________________________________________________
Model Created Successfully!
```


time_distributed_3_input | input: | [(None, 20, 64, 64, 3)]
InputLayer | output: | [(None, 20, 64, 64, 3)]
time_distributed_3(conv2d) | input: | (None, 20, 64, 64, 3)
TimeDistributed(Conv2D) | output: | (None, 20, 64, 64, 16)
time_distributed_4(max_pooling2d) | input: | (None, 20, 64, 64, 16)
TimeDistributed(MaxPooling2D) | output: | (None, 20, 16, 16, 16)
time_distributed_5(dropout_3) | input: | (None, 20, 16, 16, 16)
TimeDistributed(Dropout) | output: | (None, 20, 16, 16, 16)
time_distributed_6(conv2d_1) | input: | (None, 20, 16, 16, 16)
TimeDistributed(Conv2D) | output: | (None, 20, 16, 16, 32)
time_distributed_7(max_pooling2d_1) | input: | (None, 20, 16, 16, 32)
TimeDistributed(MaxPooling2D) | output: | (None, 20, 4, 4, 32)
time_distributed_8(dropout_4) | input: | (None, 20, 4, 4, 32)
TimeDistributed(Dropout) | output: | (None, 20, 4, 4, 32)
time_distributed_9(conv2d_2) | input: | (None, 20, 4, 4, 32)
TimeDistributed(Conv2D) | output: | (None, 20, 4, 4, 64)
time_distributed_10(max_pooling2d_2) | input: | (None, 20, 4, 4, 64)
TimeDistributed(MaxPooling2D) | output: | (None, 20, 2, 2, 64)
time_distributed_11(dropout_5) | input: | (None, 20, 2, 2, 64)
TimeDistributed(Dropout) | output: | (None, 20, 2, 2, 64)
time_distributed_12(conv2d_3) | input: | (None, 20, 2, 2, 64)
TimeDistributed(Conv2D) | output: | (None, 20, 2, 2, 64)
time_distributed_13(max_pooling2d_3) | input: | (None, 20, 2, 2, 64)
TimeDistributed(MaxPooling2D) | output: | (None, 20, 1, 1, 64)
time_distributed_14(flatten_1) | input: | (None, 20, 1, 1, 64)
TimeDistributed(Flatten) | output: | (None, 20, 64)
lstm | input: | (None, 20, 64)
LSTM | output: | (None, 32)
dense_1 | input: | (None, 32)
Dense | output: | (None, 3)

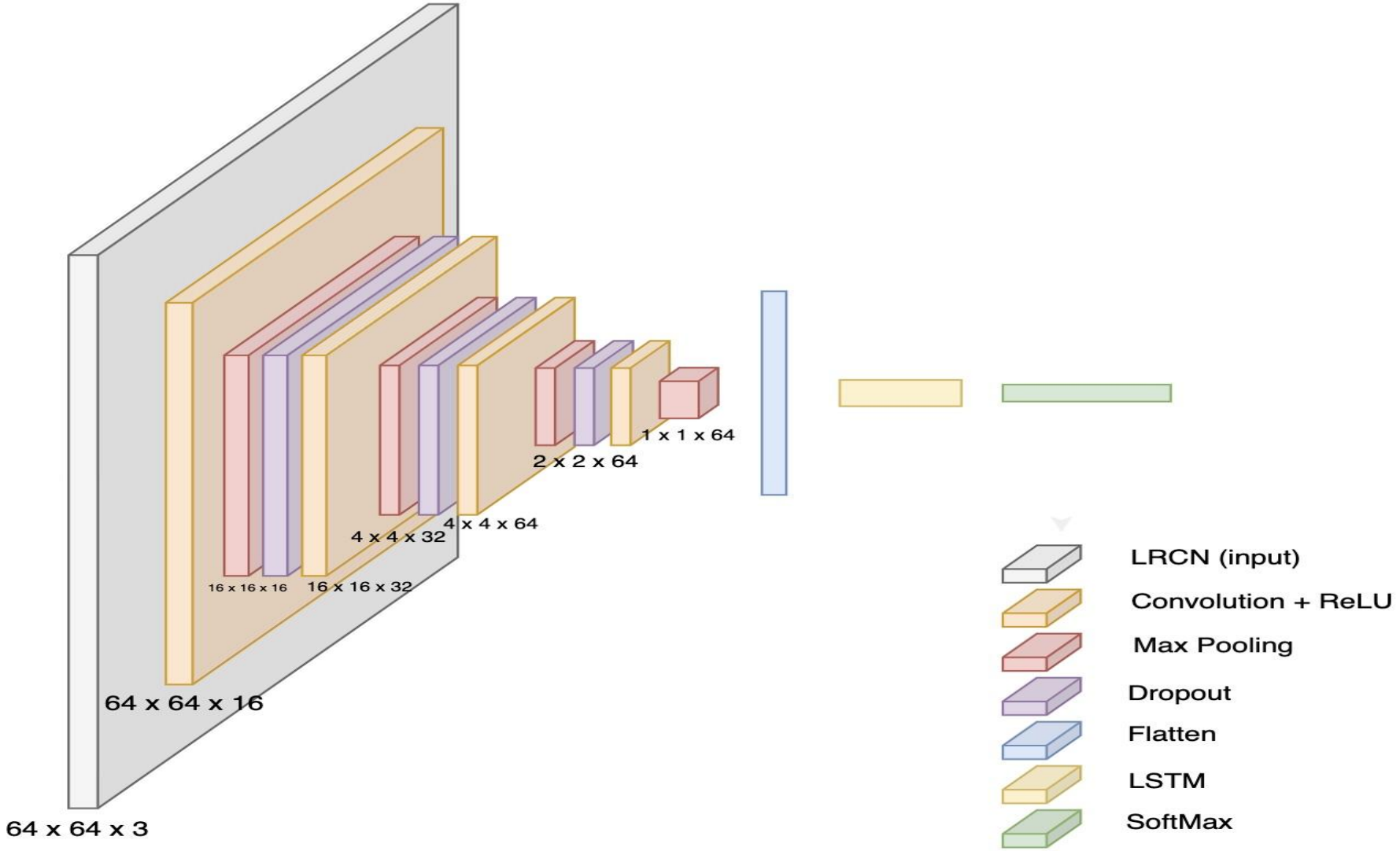


Fig: LRCN Model Architecture Diagram

Bi-LSTM

3. Implementation of Bi-LSTM Model:

   a. Input Layer
   b. MobileNetV2
   c. Dropout Layer
   d. Bi-LSTM layer
   e. Dense Layer
   f. SoftMax Layer

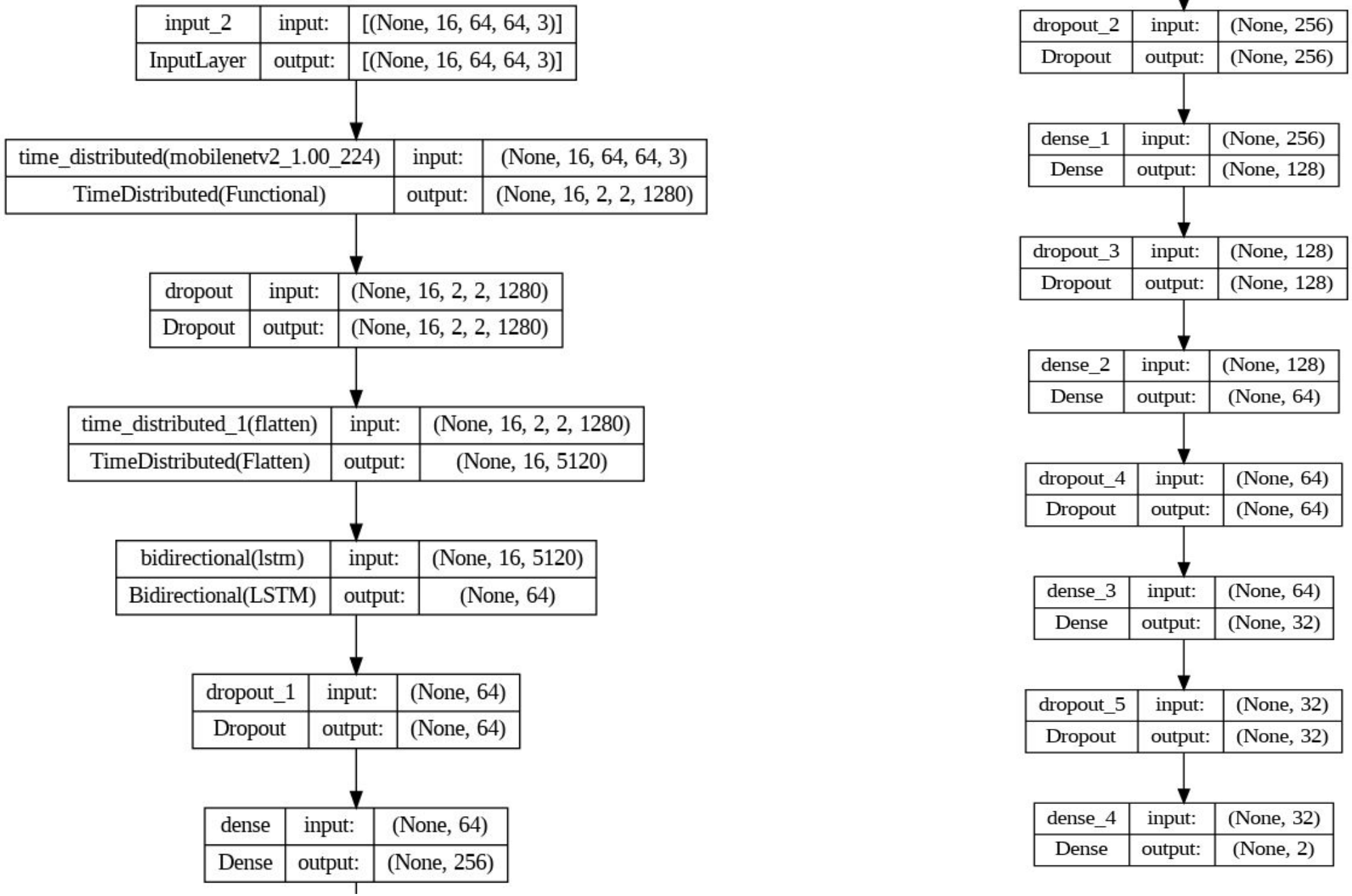
The loss function is Categorical Cross Entropy
The optimizer is SGD
The model performs a binary classification
Activation function used in ReLU

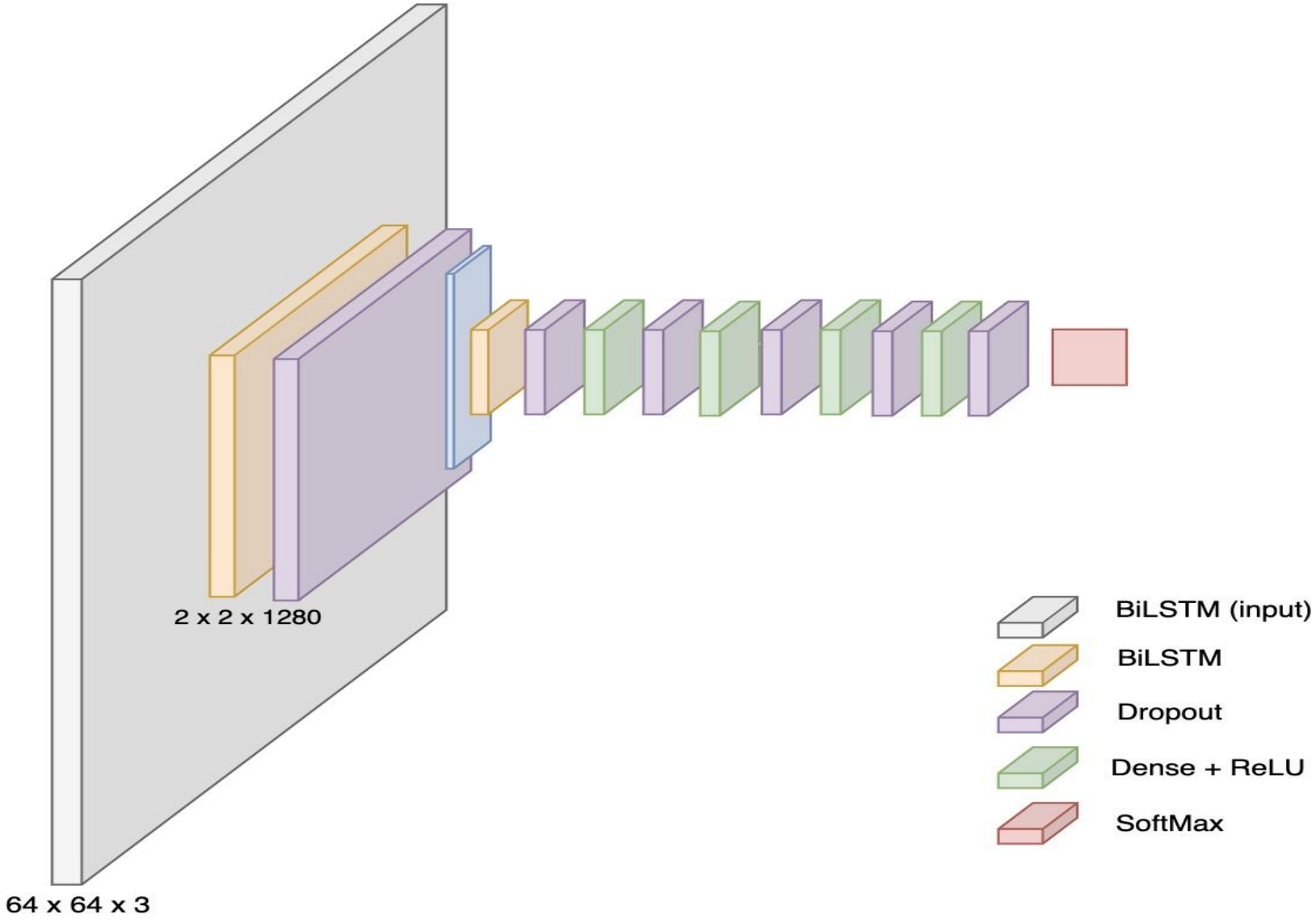
CORPORATE GURUKUL

Model: "sequential_1"

Layer (type)	Output Shape	Param #
time_distributed_2 (TimeDistributed)	(None, 16, 2, 2, 1280)	2257984
dropout_6 (Dropout)	(None, 16, 2, 2, 1280)	0
time_distributed_3 (TimeDistributed)	(None, 16, 5120)	0
bidirectional_1 (Bidirectional)	(None, 64)	1319168
dropout_7 (Dropout)	(None, 64)	0
dense_5 (Dense)	(None, 256)	16640
dropout_8 (Dropout)	(None, 256)	0
dense_6 (Dense)	(None, 128)	32896
dropout_9 (Dropout)	(None, 128)	0
dense_7 (Dense)	(None, 64)	8256
dropout_10 (Dropout)	(None, 64)	0
dense_8 (Dense)	(None, 32)	2080
dropout_11 (Dropout)	(None, 32)	0
dense_9 (Dense)	(None, 2)	66

Total params: 3,637,090
Trainable params: 3,060,642
Non-trainable params: 576,448


input_2 | input: | [(None, 16, 64, 64, 3)]
InputLayer | output: | [(None, 16, 64, 64, 3)]
time_distributed(mobilenetv2_1.00_224) | input: | (None, 16, 64, 64, 3)
TimeDistributed(Functional) | output: | (None, 16, 2, 2, 1280)
dropout | input: | (None, 16, 2, 2, 1280)
Dropout | output: | (None, 16, 2, 2, 1280)
time_distributed_1(flatten) | input: | (None, 16, 2, 2, 1280)
TimeDistributed(Flatten) | output: | (None, 16, 5120)
bidirectional(lstm) | input: | (None, 16, 5120)
Bidirectional(LSTM) | output: | (None, 64)
dropout_1 | input: | (None, 64)
Dropout | output: | (None, 64)
dense | input: | (None, 64)
Dense | output: | (None, 256)
dropout_2 | input: | (None, 256)
Dropout | output: | (None, 256)
dense_1 | input: | (None, 256)
Dense | output: | (None, 128)
dropout_3 | input: | (None, 128)
Dropout | output: | (None, 128)
dense_2 | input: | (None, 128)
Dense | output: | (None, 64)
dropout_4 | input: | (None, 64)
Dropout | output: | (None, 64)
dense_3 | input: | (None, 64)
Dense | output: | (None, 32)
dropout_5 | input: | (None, 32)
Dropout | output: | (None, 32)
dense_4 | input: | (None, 32)
Dense | output: | (None, 2)



CORPORATE GURUKUL

Fig: Bi-LSTM Model Architecture Diagram

Comparison of Models

PARAMETERS	LRCN	CONVLSTM	BILSTM
Epoch	40	30	20
Batch size	8	8	8
Optimizer	Adam	Adam	SGD
Loss Function	Categorical Cross Entropy	Categorical Cross Entropy	Categorical Cross Entropy
Dropout	0.5	0.2	0.25
Activation	Relu	Relu	Relu
Validation Accuracy	0.73	0.64	0.925
Validation Loss	0.85	0.96	0.29

CORPORATE GURUKUL

Validation Split of 0.2 used in every case

LRCN

Activation	Dropout	Epoch	Batch size	Optimizer	Val Accuracy	Val loss
Relu	0.25	40	8	Adam	0.74	1
tanh	0.5	40	8	Adam	0.68	0.85
Relu	0.5	30	8	Adagrad	0.66	0.94
tanh	0.5	30	8	Adagrad	0.61	0.95
Relu	0.25	40	8	SGD	0.61	0.88
tanh	0.25	40	8	SGD	0.618	0.84

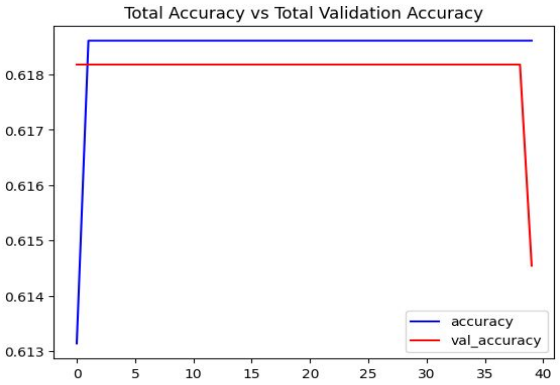
Adam gave the best result for same parameters when compared against adagrad and SGD

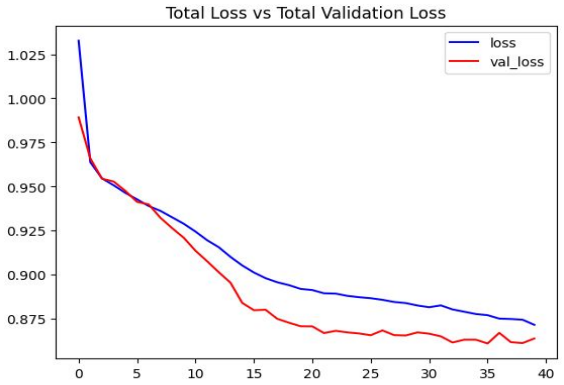
ConvLSTM

Activation	Dropout	Epoch	Batch size	Optimizer	Val Accuracy	Val loss
Relu	0.5	40	4	Adam	0.67	1
tanh	0.75	40	16	Adam	0.62	1
Relu	0.2	40	4	Adagrad	0.61	0.86
tanh	0.2	30	8	Adagrad	0.62	0.91
Relu	0.5	30	8	SGD	0.65	0.82
tanh	0.5	30	8	SGD	0.61	0.85

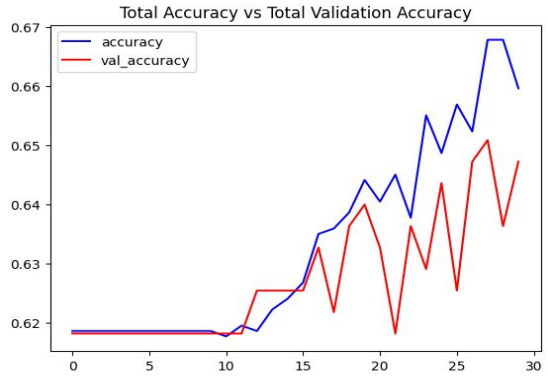
Adam gave best result on higher epoch and lower batch size using relu activation

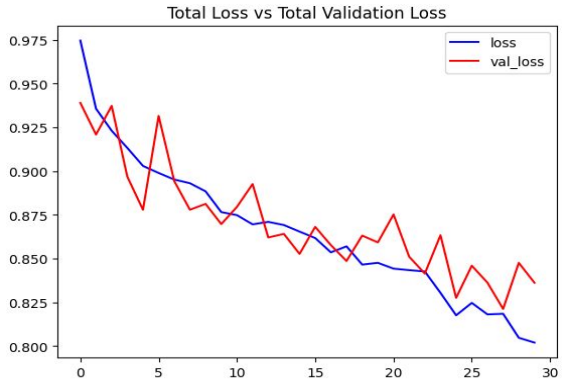
CONVLSTM APPROACH



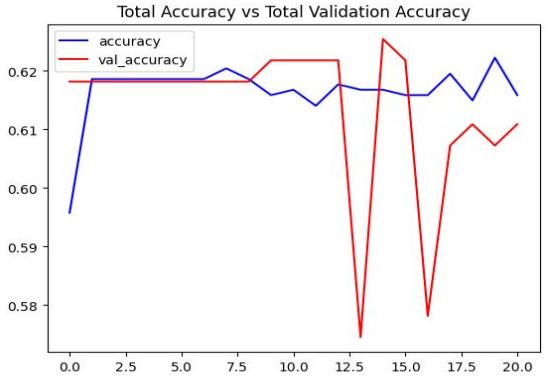


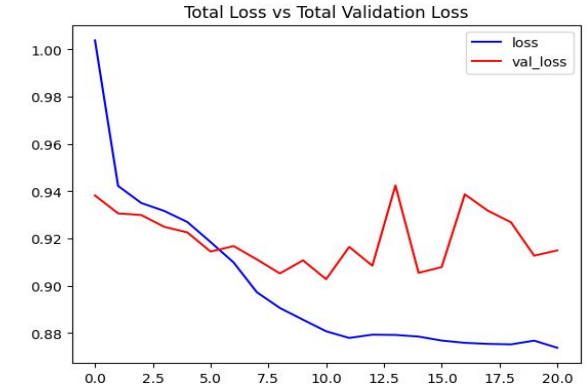
ReLU Adagrad (constant learning rate)



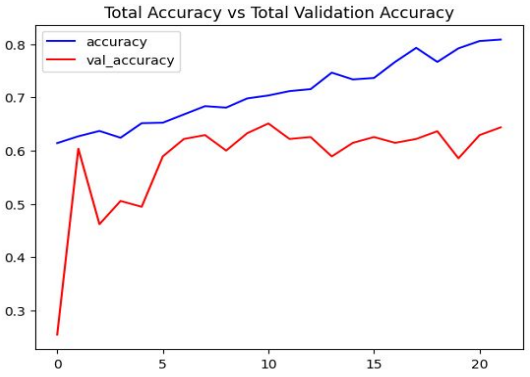


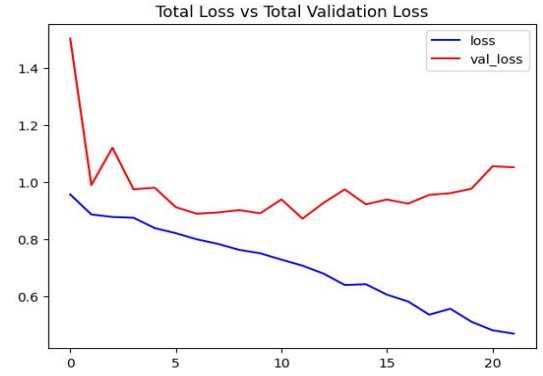
ReLU SGD (somewhat balanced but not affecting output)



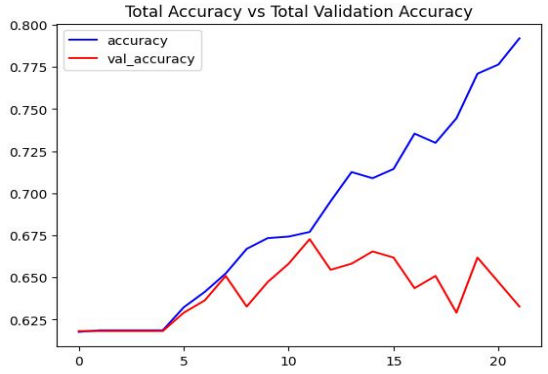


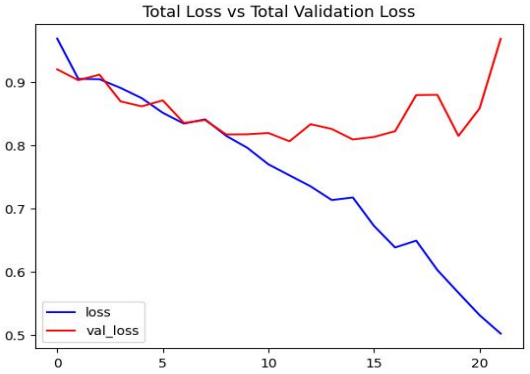
Tanh Adagrad (constant learning rate)





Tanh Adam with 0.75 dropout

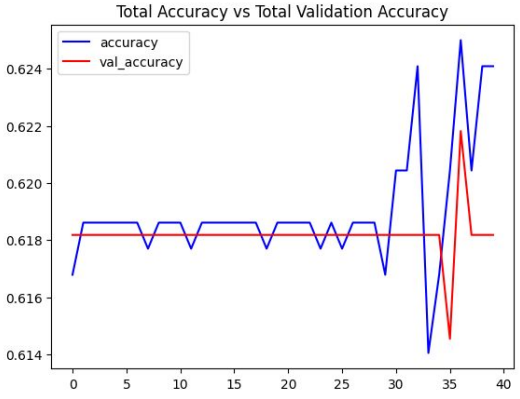


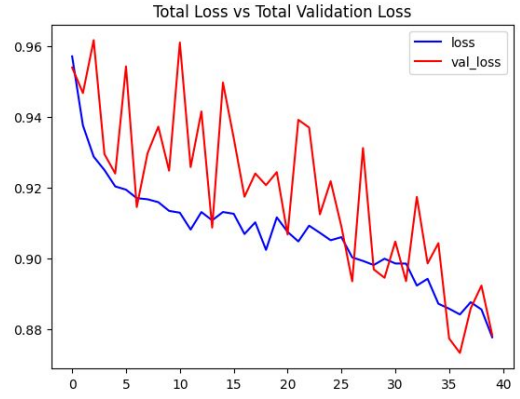


ReLU Adam best case

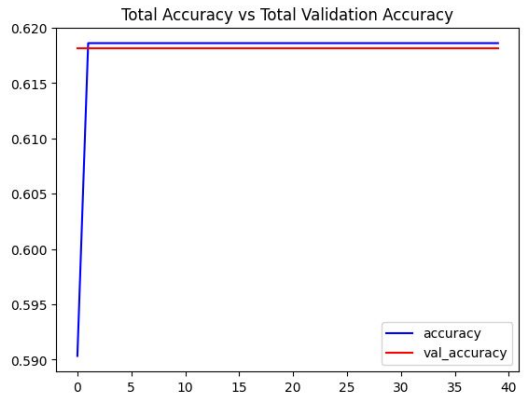
CORPORATE GURUKUL

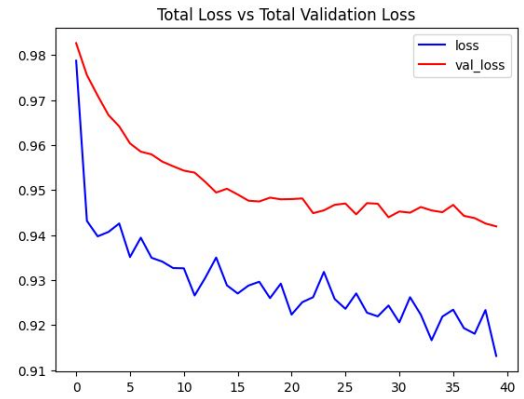
LRCN APPROACH



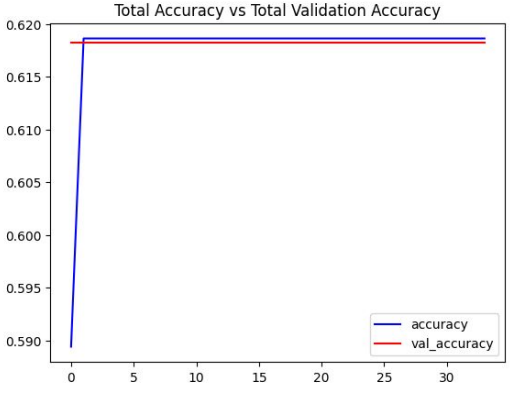


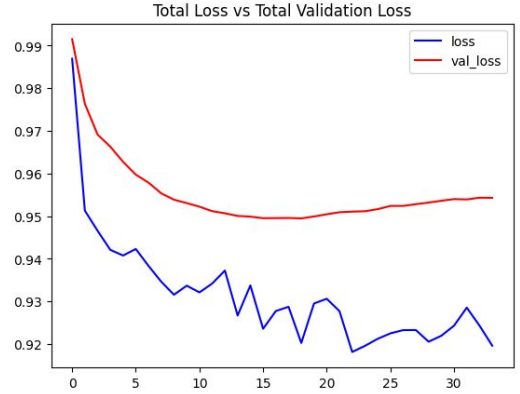
ReLU SGD (somewhat balanced but not affecting output)



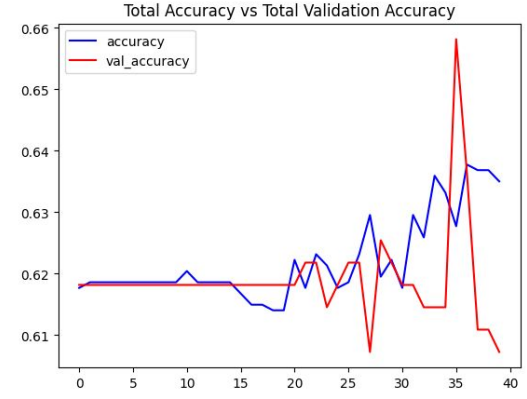


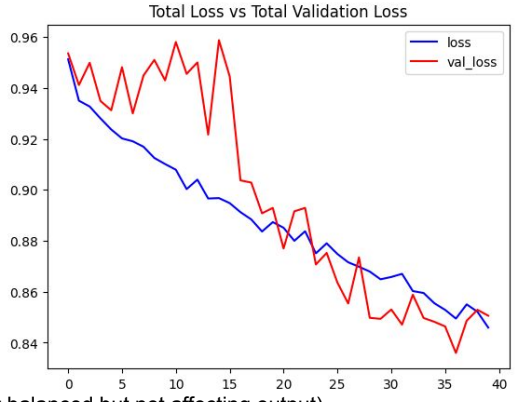
Tanh Adagrad (constant learning rate)



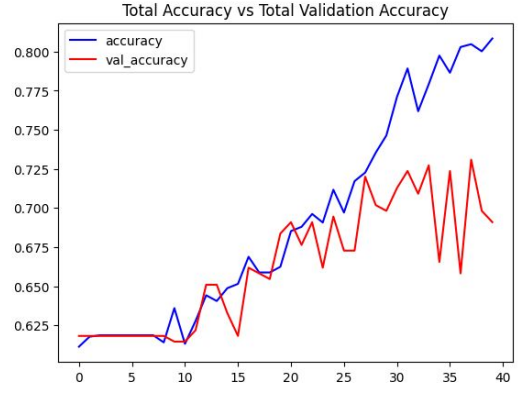


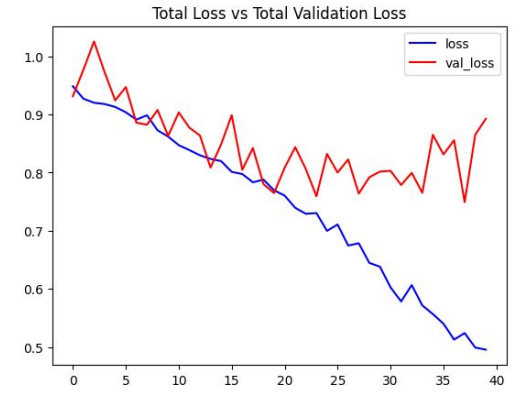
ReLU Adagrad (constant learning rate)





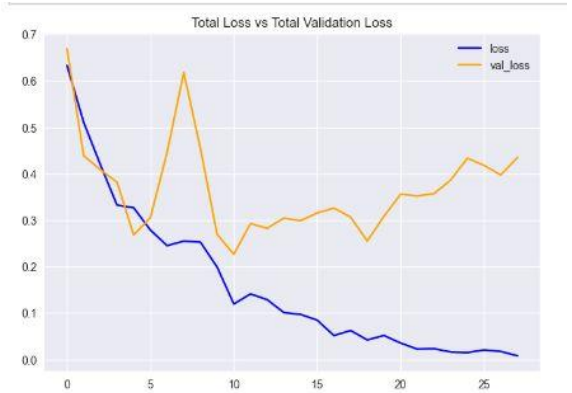
Tanh SGD (somewhat balanced but not affecting output)





ReLU Adam best case

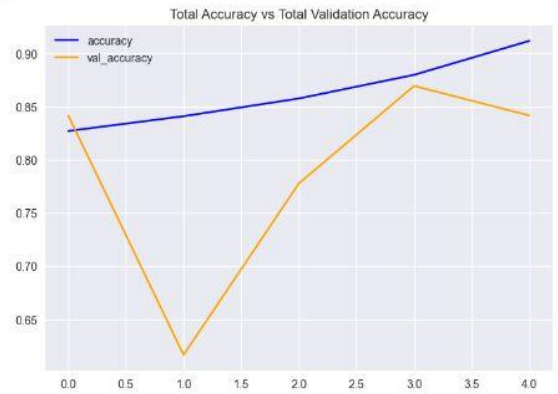
CORPORATE GURUKUL



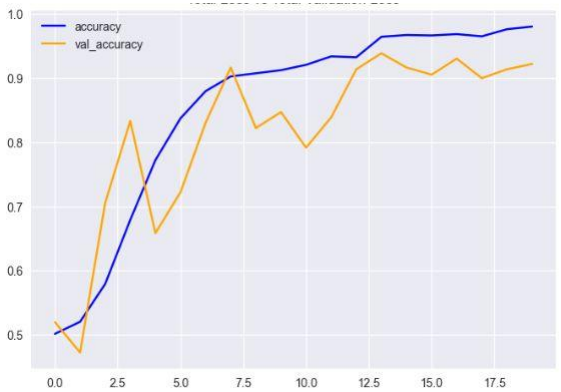

```
]: plot_metric(MobBiLSTM_model_history, 'accuracy', 'val_accuracy', 'Total Accuracy vs Total
```

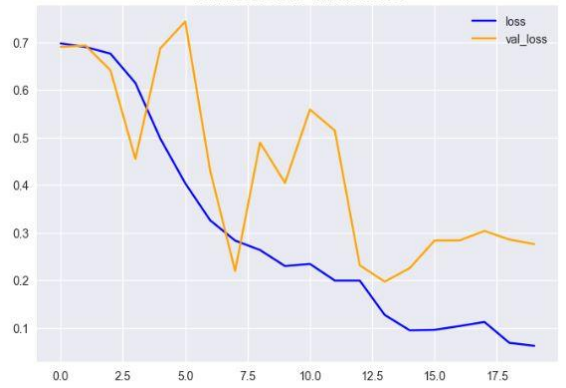
epochs = 30
batch_size = 8,
validation_split = 0.2
Early stopping callback
Binary Classification
Adam


```
plot_metric(MobBiLSTM_model_history, 'accuracy', 'val_accuracy', 'Total Accuracy vs Total '
```



epochs = 5
batch_size = 4
validation_split = 0.2
Early stopping callback
Categorical classification
SGD





epochs = 20
batch_size = 8,
validation_split = 0.2
Early stopping callback
Binary Classification
SGD

CORPORATE
GURUKUL

Inference

- **Batch sizes**

  Smaller showed faster convergence to optimal solutions, larger gave poor generalization
- **Early Stopping Callback**

  Separates bias and variance sections and **prevents overfitting**
- **Categorical Cross Entropy over Sparse Categorical Cross Entropy**

  Prediction against one sided solution
- **Dropout** for regularization

  Dropping neurons to prevent overfitting, too much leads to slow convergence
- **Optimizer**

  Adam converges faster, Adagrad shrinks learning rate, SGD converges to optimal solutions
- **Activation functions** - Tanh and ReLU

Optimal Case

The optimal case is the **BiLSTM model**

Parameters:

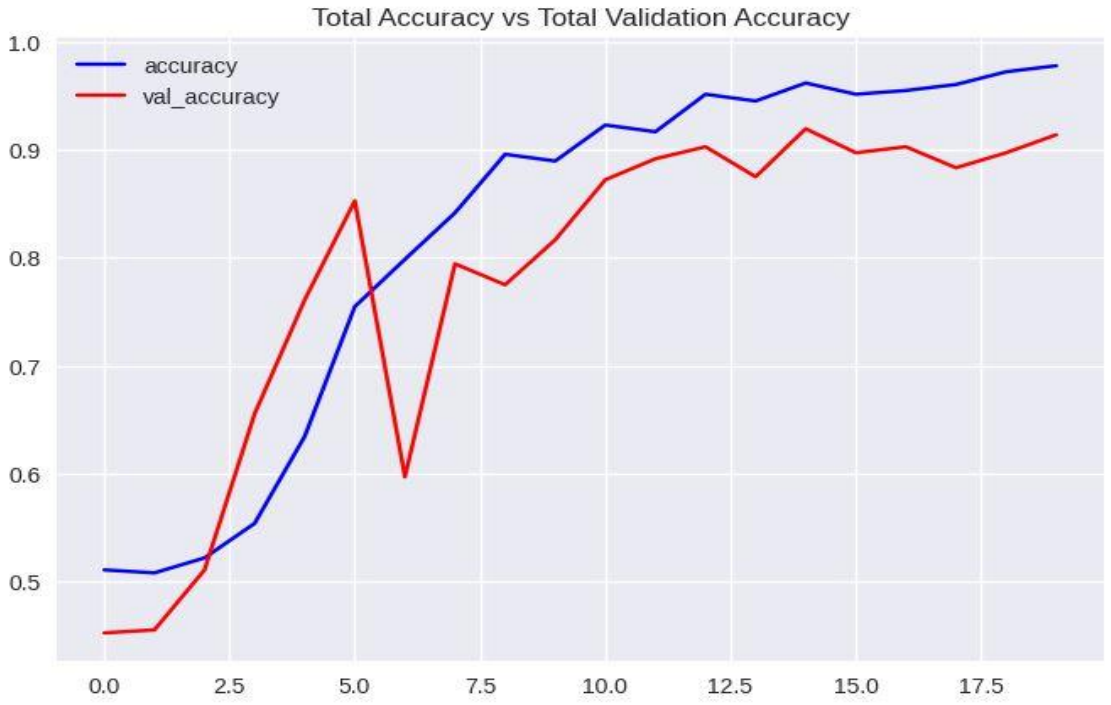
Number of Epochs: 20
Batch size: 8
Optimiser: SGD
Validation Accuracy: 0.93

Activation Function: ReLU
Dropout: 0.25
Loss Function: Categorical Cross Entropy
Validation Loss: 0.29

Early Stopping Callback to prevent overfitting
Validation split : 0.2

Bi-LSTM Visualizations

Model Accuracy and Loss:

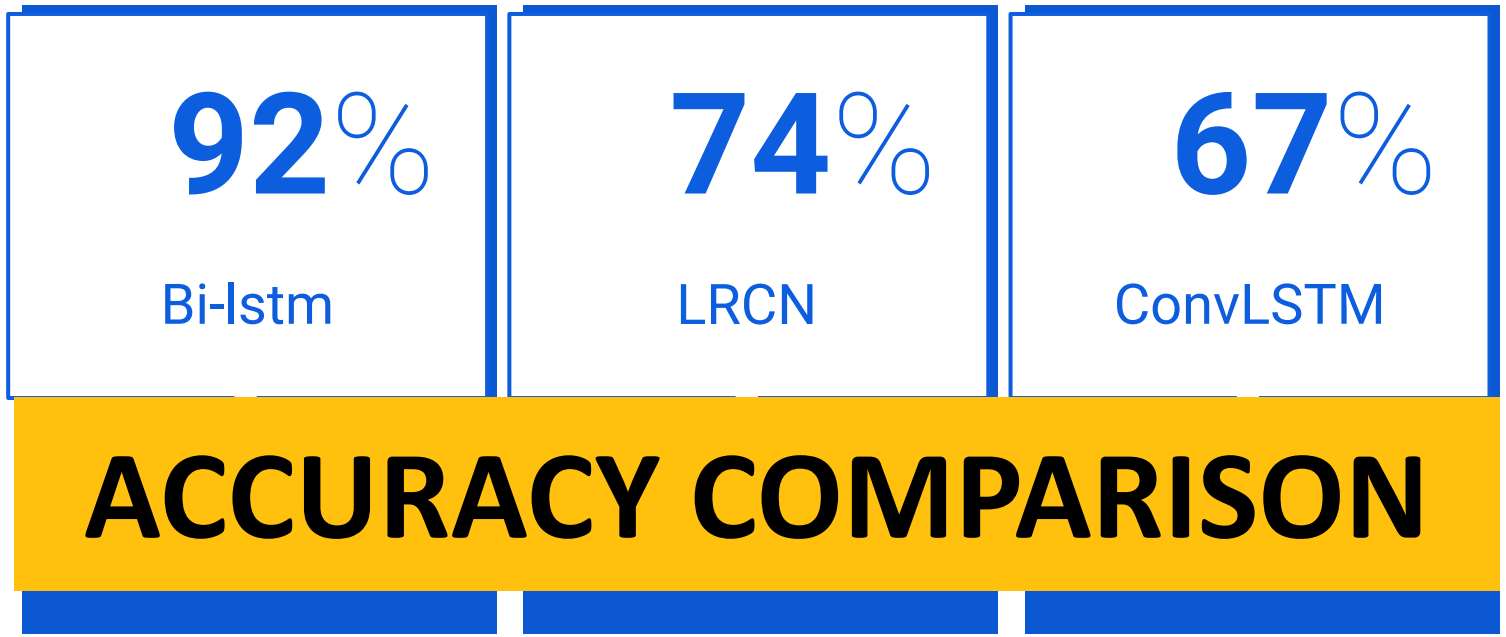


CORPORATE GURUKUL

Final Recommendation

Final Recommendation

- Bi-LSTM model with ReLU activation function and categorical cross entropy loss function
- Integration of object, speech & sentiment will improve the model



CORPORATE GURUKUL

Our Learning

1. Working with **HuggingFace** Spaces API and **Streamlit**
2. The team learned how to work with **TensorFlow** and **Keras** to build Deep Neural Networks
3. Performing data pre-processing activities like **resizing images, frame conversion** & **balancing dataset** using necessary python packages and libraries

TensorFlow Keras

CORPORATE GURUKUL

Future Enhancements

1. **Object detection**: Can help police departments identify and classify objects in real-time, aiding in threat assessment by distinguishing potential weapons or suspicious items.
2. **Speech recognition**: Using speech recognition technology, law enforcement can analyze spoken words and detect certain keywords or phrases that may indicate a potential threat, enabling them to take appropriate action.
3. **Face sentiment analysis**: Can interpret facial expressions and emotions, helping police departments gauge the emotional state of individuals. This information can be useful in determining if someone appears agitated, angry, or potentially threatening.

Demo

Let’s start the demo for SafetyNet!

Challenges Faced

1. **Data Collection**: Finding relevant dataset with a balanced set of violent and non-violent actions
2. **Data Pre-Processing**: Converting frames into videos
3. **Bias and Variance** while optimization
4. Underwhelming accuracies on **self-created dataset**
5. **Integrating live feed** with the model to predict outcomes in real times
6. Initial implementation of pose detection with **MediaPipe Holistics**
7. **Understanding** out how to get the **first few models** up and running, figuring out what kinds of datasets we need, when to add what kind of layers.
8. While training ran into issues about **limited computational resources**

GROUP 4

VIRAJ PARMAJ
Data Preprocessing
+ Data Preparation

MRIDUL SANGHAVI
Model Building
Model Preprocessing

M. ANANYA RAJU
Model Building
+ ML Deployment

ARYAMAN DEV
Hyperparameter Tuning
+ Business Logic

HRIDAY GUPTA
Data Collection
+ UI/UX

SNEHA AGARWAL
EDA
+ FrontEnd, Business Logic

CORPORATE GURUKUL

Thank You