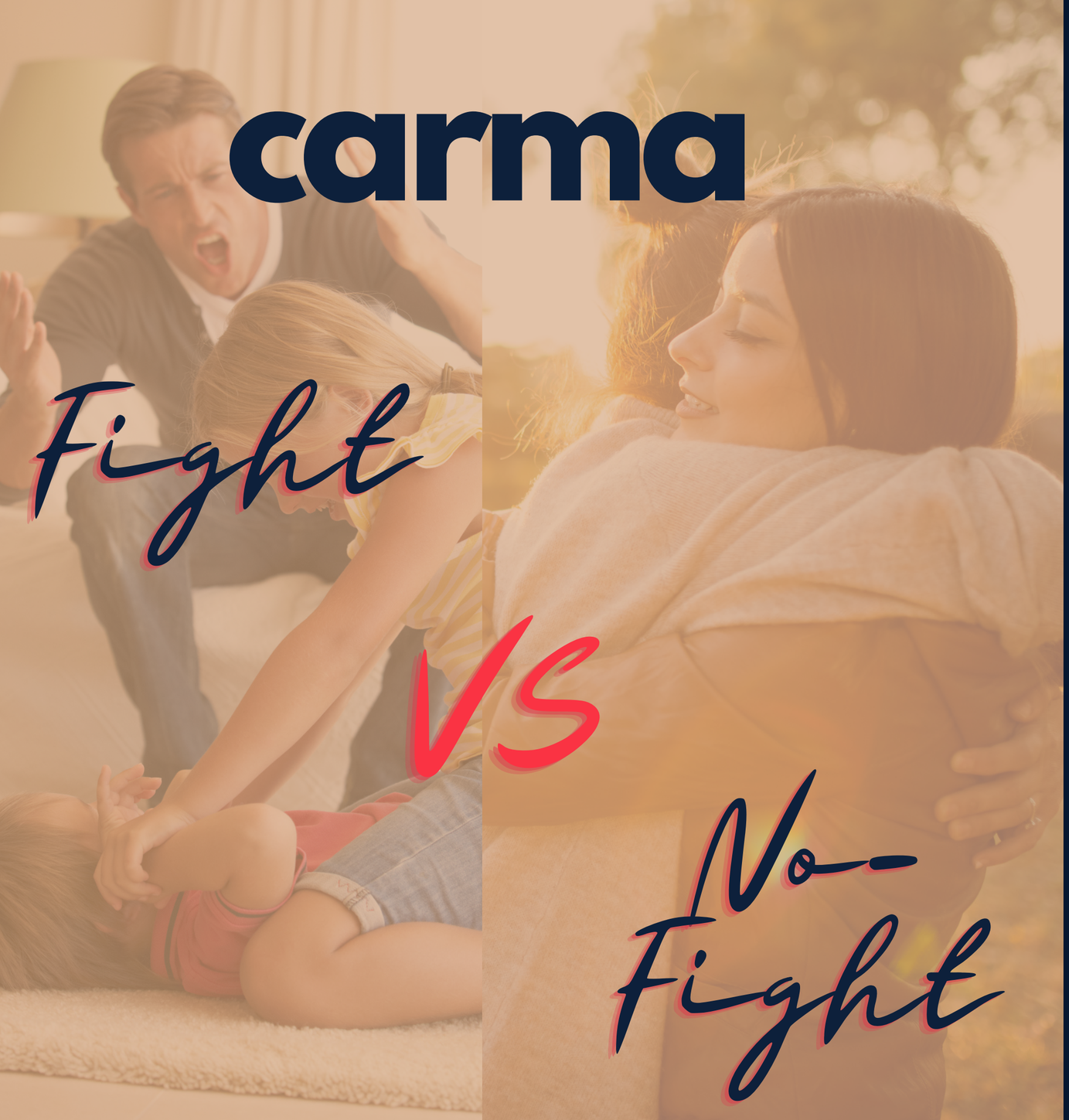


Violence Detection from Real-time Surveillance Cameras

- Group 10

https://github.com/ssachis/CRIME-SURVEILLANCE

Meet the team !

Ananya
(Team Lead, model building, feature addition)

Joel
(model building,app features, documentation)

Arnav
(Front-end,model building, deployment)

Saran
(model building, documentation)

Sachi
(model building, deployment, feature addition)

Arjun
(model building, hyperparameter optimization)

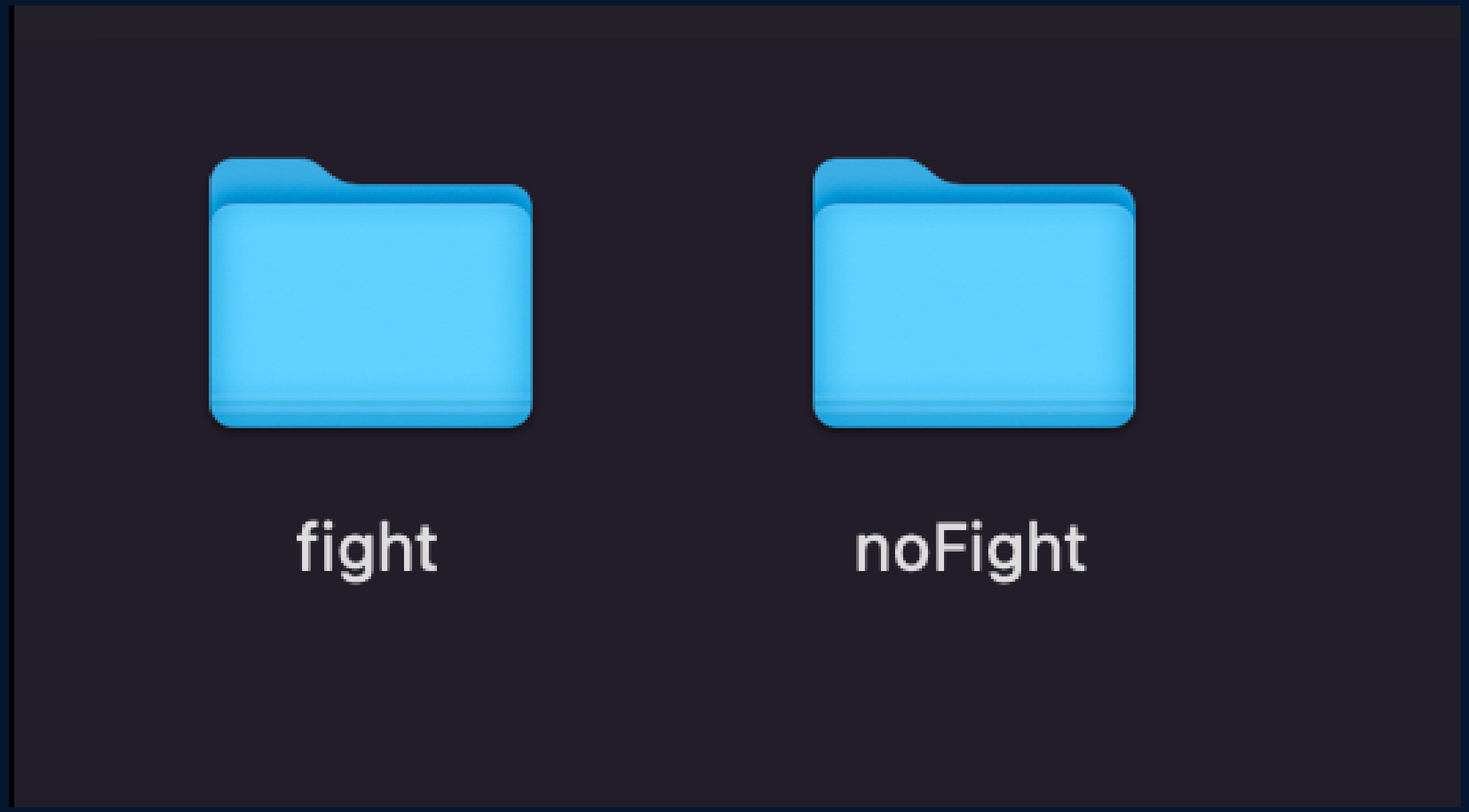
Problem Statement :-

In video surveillance, to critically assure public safety, hundreds and thousands of surveillance cameras are deployed within cities, but it is almost impossible to manually monitor all cameras to keep an eye on violent activities. Rather, there is a significant requirement for developing automated video surveillance systems to automatically track and monitor such activities.

Technology Used

Data Visualization

Dataset



-150 fight videos
-150 no fight videos

The model extracts the frames and during training the following labels are generated:

Data Preprocessing

Step 1 : Extracted the frames from the videos

Step 2 : Made a dataframe using the extracted frames

-Dataframe contains the following:

a) features - list of all the frames

b) labels - they have the class index

c) video file paths - paths

Step 3 : Using the formula given below the model skips certain frames while extracting them from the video.

skip_frames_video = video_frame_count/sequence_length

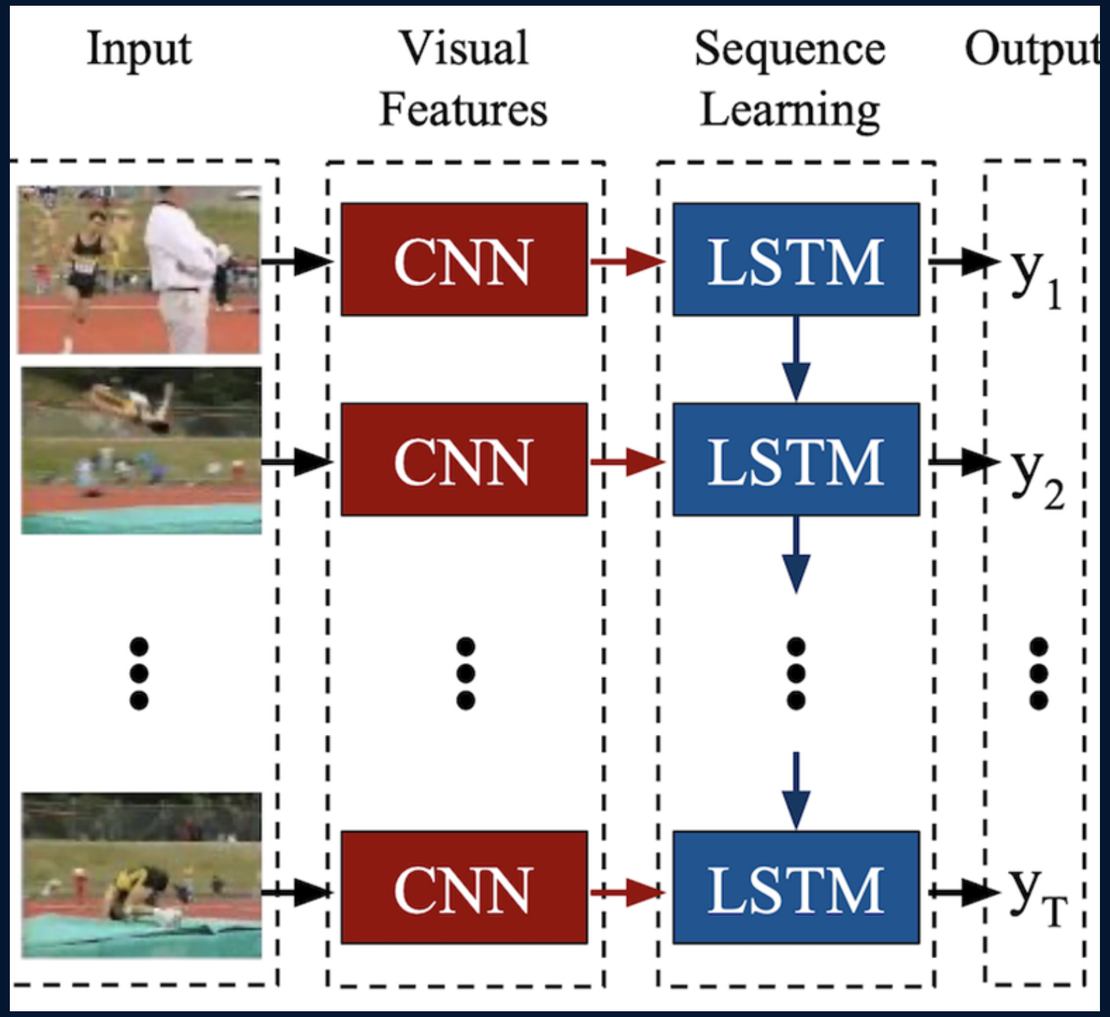
where sequence_length=20

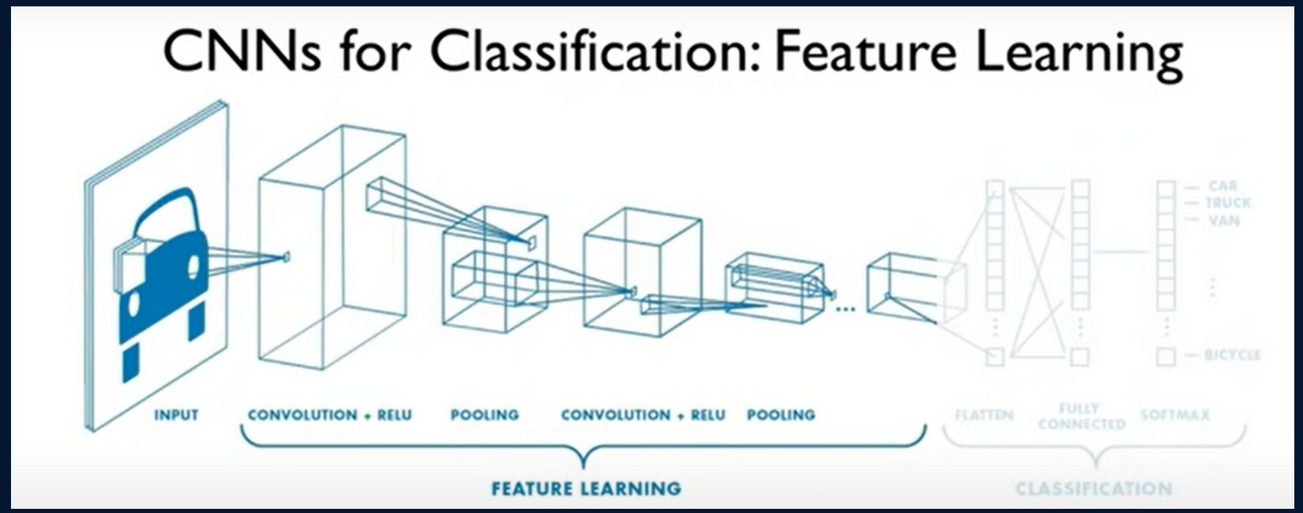
What did we build ?

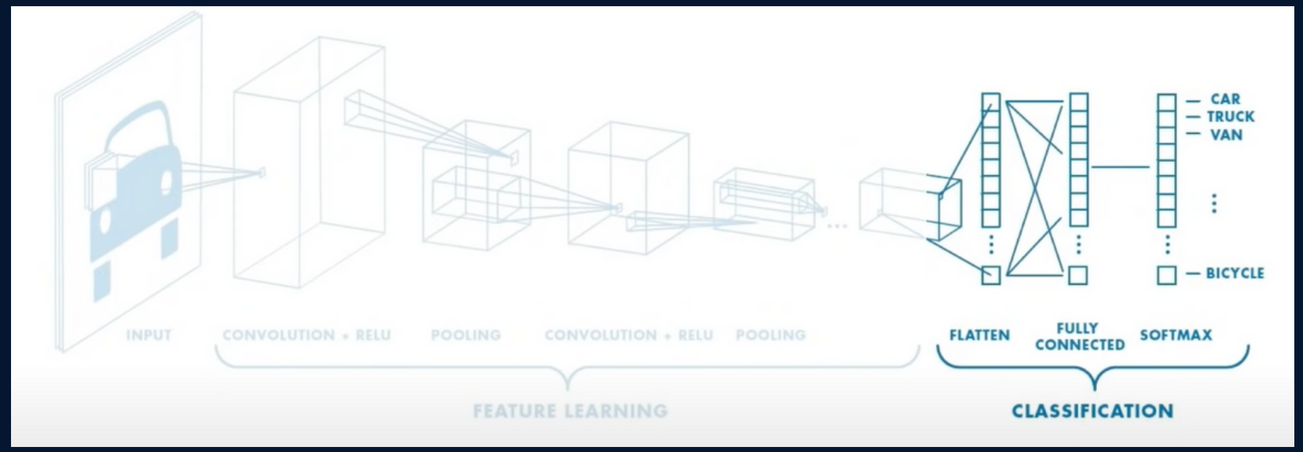
Model 1 : LRCN (CNN + LSTM)

Model 2 :ConvLSTM

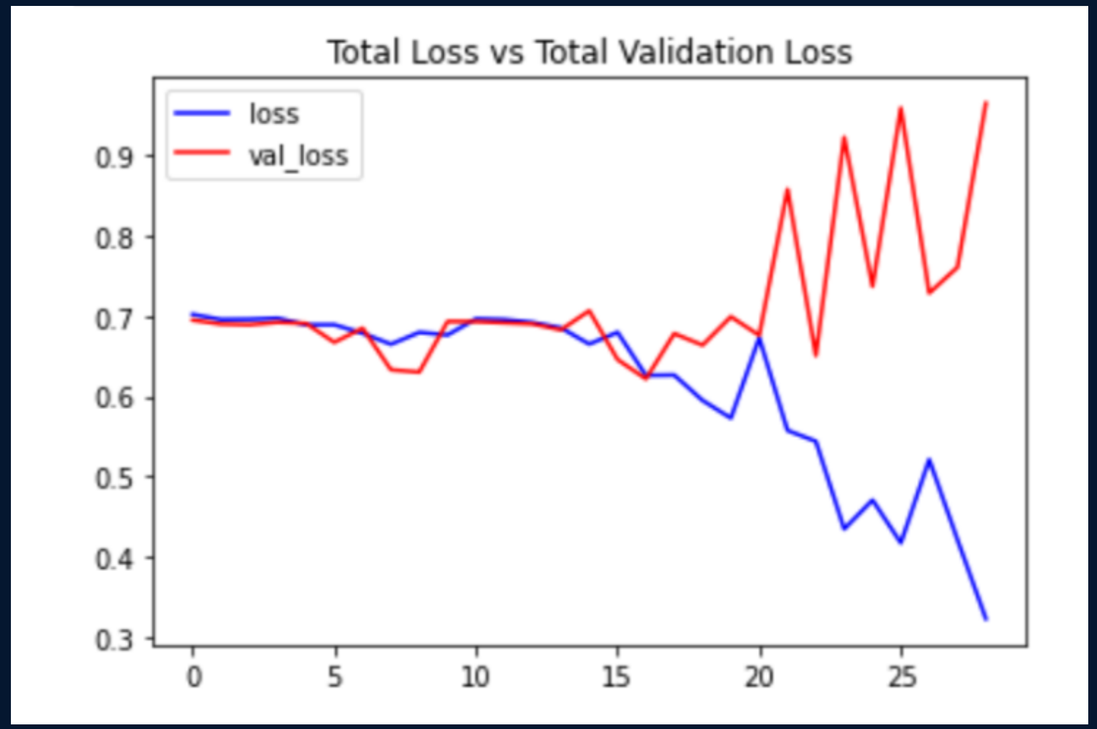
LRCN Model

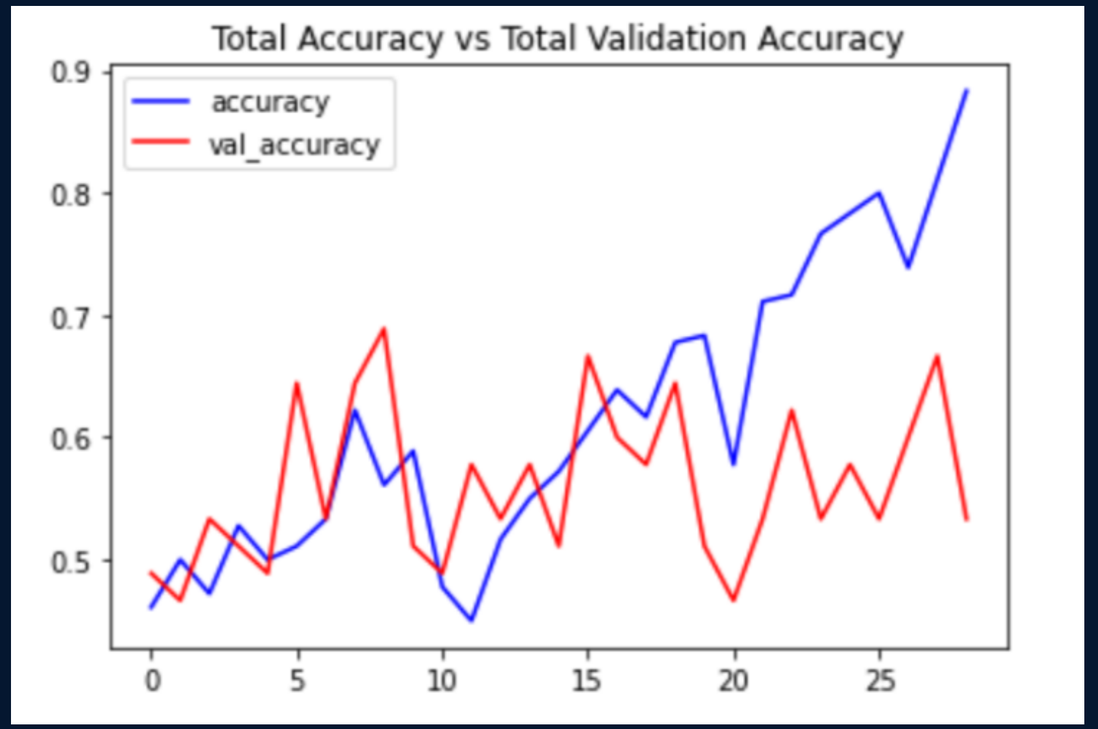

Input
Visual Features
Sequence Learning
Output
CNN
CNN
CNN
LSTM
LSTM
LSTM
y_1
y_2
y_T


CNNs for Classification: Feature Learning
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
CAR
TRUCK
VAN
BICYCLE
FEATURE LEARNING
CLASSIFICATION


INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
CAR
TRUCK
VAN
BICYCLE
FEATURE LEARNING
CLASSIFICATION

LRCN Model (accuracy - 73%)






```
Epoch 29/70
45/45 [==============================] - 5s 105ms/step - loss: 0.3231 - accuracy: 0.8833 - val_loss: 0.9646 - val_accuracy: 0.5333

5]: model_evaluation_history = LRCN_model.evaluate(features_test, labels_test)#without regularization patience=10

3/3 [==============================] - 1s 133ms/step - loss: 0.5880 - accuracy: 0.7333
```

Hyperparameter Optimization

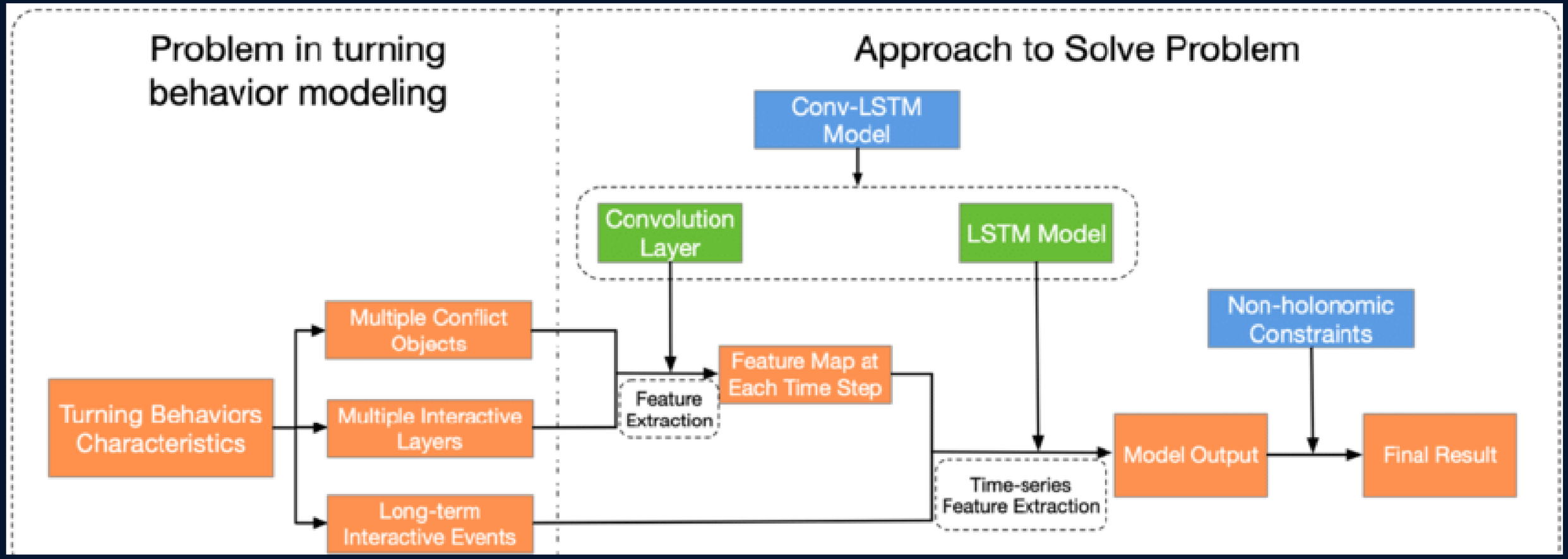
Activation in hidden layers	Dropout in I/P layer	Dropout in hidden layers	Batch size	Epochs	Early stopping	Testing accuracy(%)
RELU	0.25	0.25	4	50	No	62.67
RELU	0.8	0.6	4	70	No	58.67
Leaky RELU (alpha=0.01)	0.8	0.6	8	100	No	56.67
Leaky RELU (alpha=0.1)	0.25	0.25	4	70	Yes	64.67
Tanh	0.3	0.3	4	80	Yes	68
Tanh	0.25	0.25	4	70	Yes	73.33

Confusion Matrix and F-1 Score

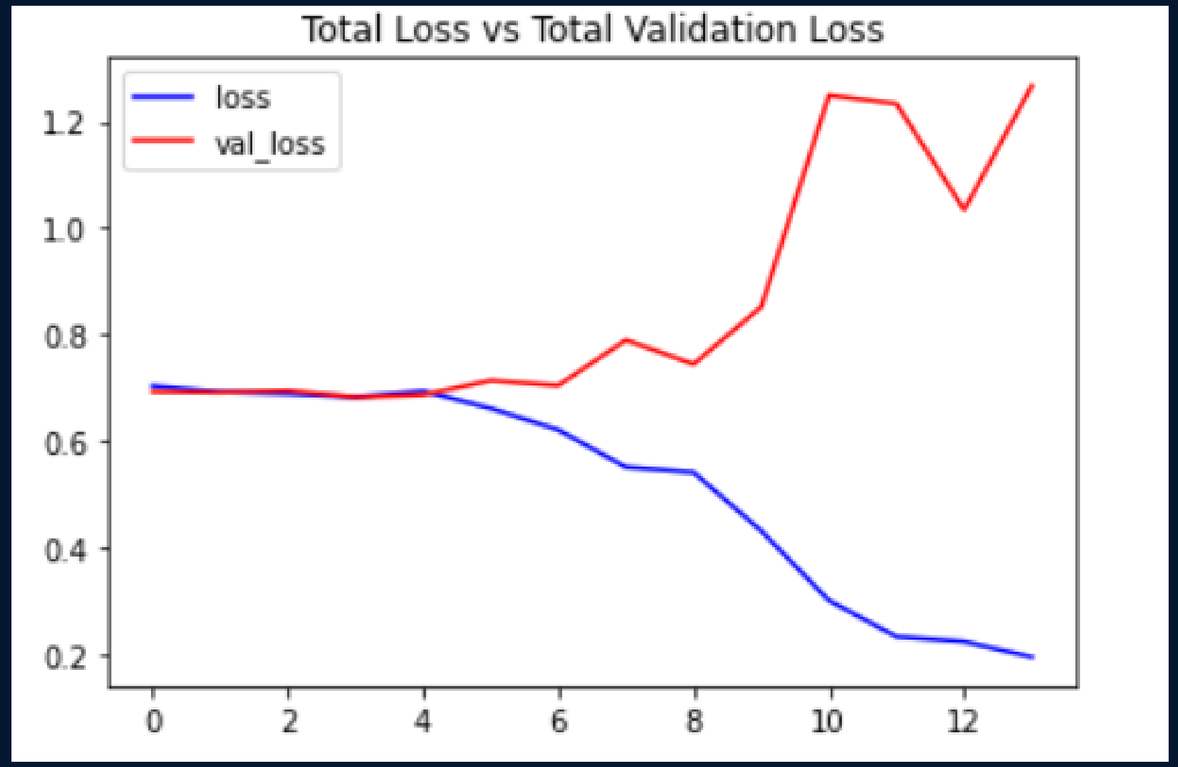
```
y_pred = LRCN_model.predict_classes(features_test)
new_array = to_categorical(y_pred,2)


from sklearn.metrics import confusion_matrix
print(confusion_matrix(labels_test.argmax(axis=1), new_array.argmax(axis=1)))

from sklearn.metrics import f1_score
print(f1_score(labels_test.argmax(axis=1), new_array.argmax(axis=1)))
```

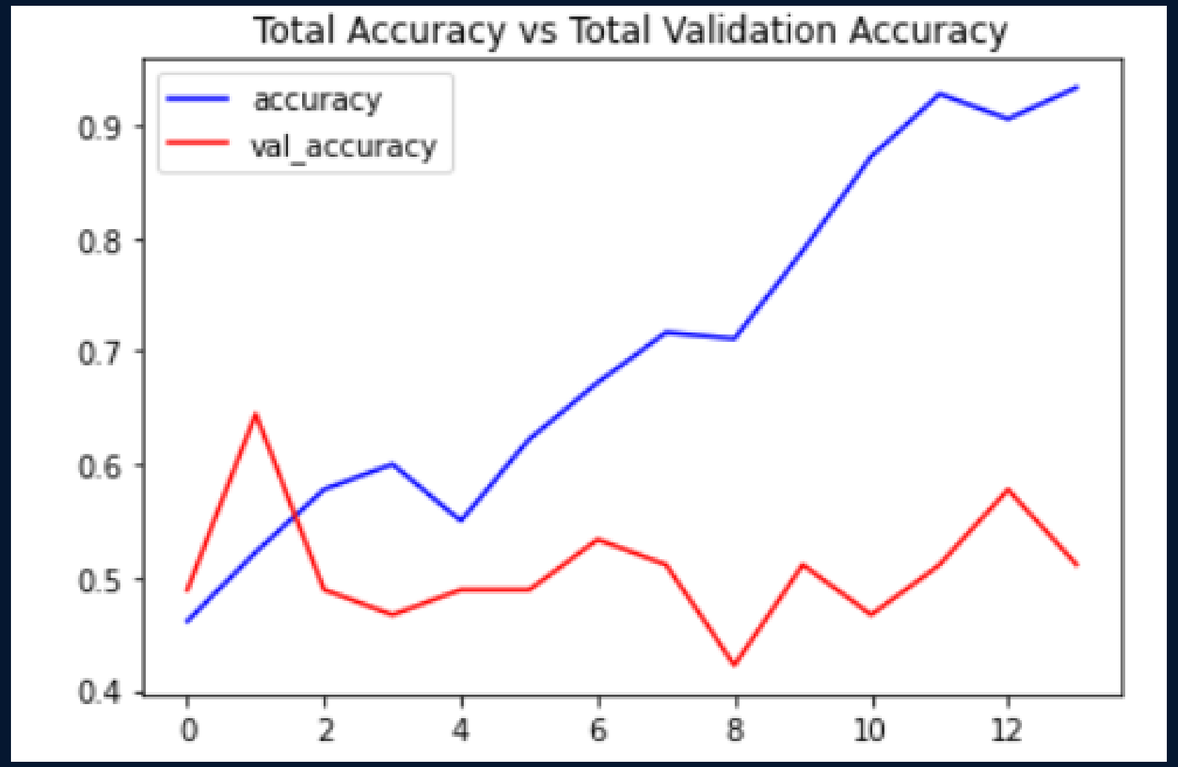
```
...  [[33  4]
      [16 22]]
     0.6875
```

ConvLSTM model



ConvLSTM






```
Epoch 17/40
45/45 [==============================] - 42s 938ms/step - loss: 0.2816 - accuracy: 0.8611 - val_loss: 1.0636 - val_accuracy: 0.
6000

In [33]: model_evaluation_history = convlstm_model.evaluate(features_test, labels_test)

3/3 [==============================] - 3s 852ms/step - loss: 0.6457 - accuracy: 0.6400
```

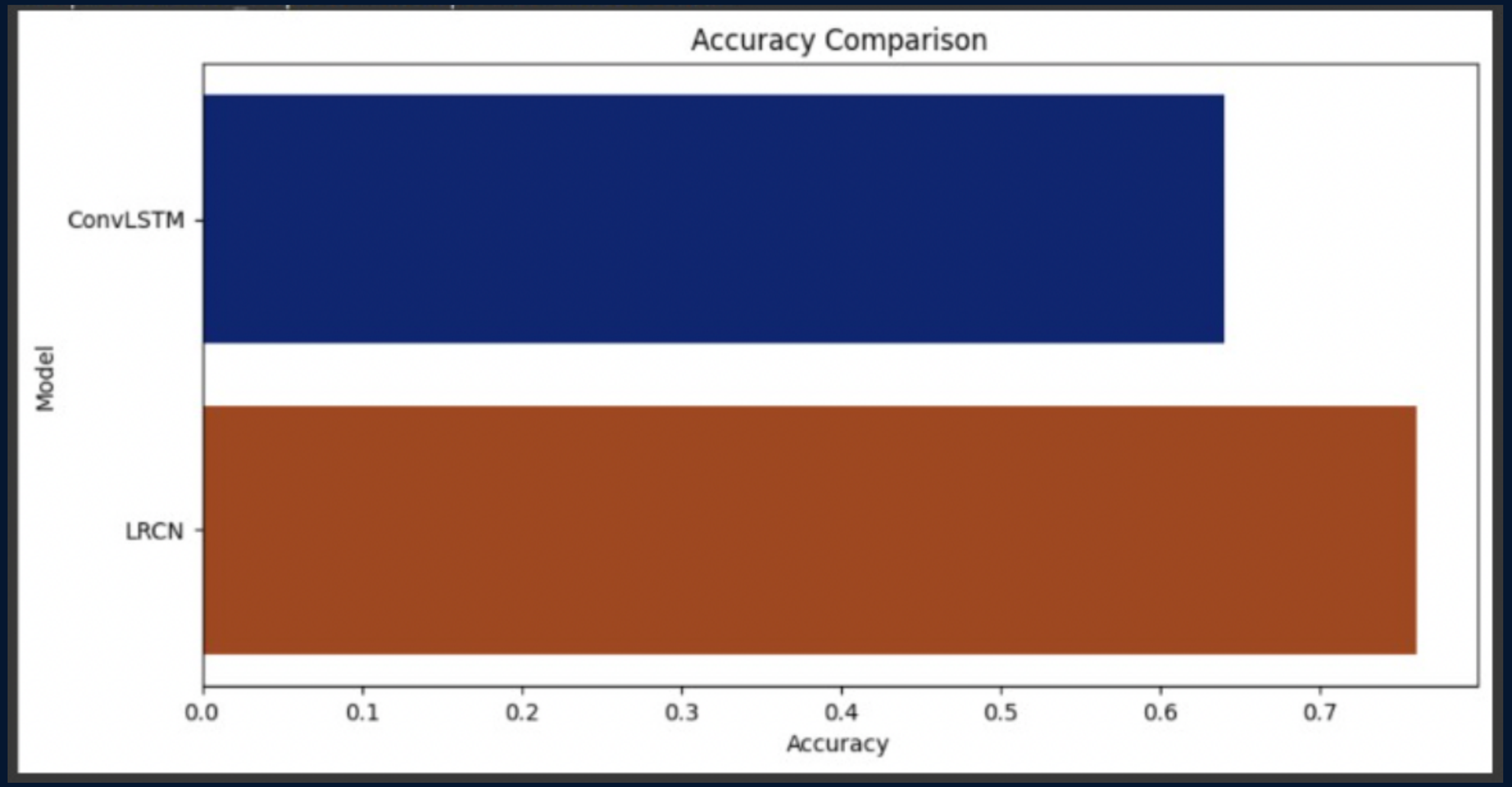
ConvLSTM

```
    return model
```

In [27]:
```
convlstm_model = create_convlstm_model()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv_lst_m2d_4 (ConvLSTM2D)  (None, 20, 62, 62, 4)     1024
_________________________________________________________________
max_pooling3d_4 (MaxPooling3 (None, 20, 31, 31, 4)     0
_________________________________________________________________
time_distributed_3 (TimeDist (None, 20, 31, 31, 4)     0
_________________________________________________________________
conv_lst_m2d_5 (ConvLSTM2D)  (None, 20, 29, 29, 8)     3488
_________________________________________________________________
max_pooling3d_5 (MaxPooling3 (None, 20, 15, 15, 8)     0
_________________________________________________________________
time_distributed_4 (TimeDist (None, 20, 15, 15, 8)     0
_________________________________________________________________
conv_lst_m2d_6 (ConvLSTM2D)  (None, 20, 13, 13, 14)    11144
_________________________________________________________________
max_pooling3d_6 (MaxPooling3 (None, 20, 7, 7, 14)      0
_________________________________________________________________
time_distributed_5 (TimeDist (None, 20, 7, 7, 14)      0
_________________________________________________________________
conv_lst_m2d_7 (ConvLSTM2D)  (None, 20, 5, 5, 16)      17344
_________________________________________________________________
max_pooling3d_7 (MaxPooling3 (None, 20, 3, 3, 16)      0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2880)              0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 5762
=================================================================
Total params: 38,762
Trainable params: 38,762
Non-trainable params: 0
_________________________________________________________________
```

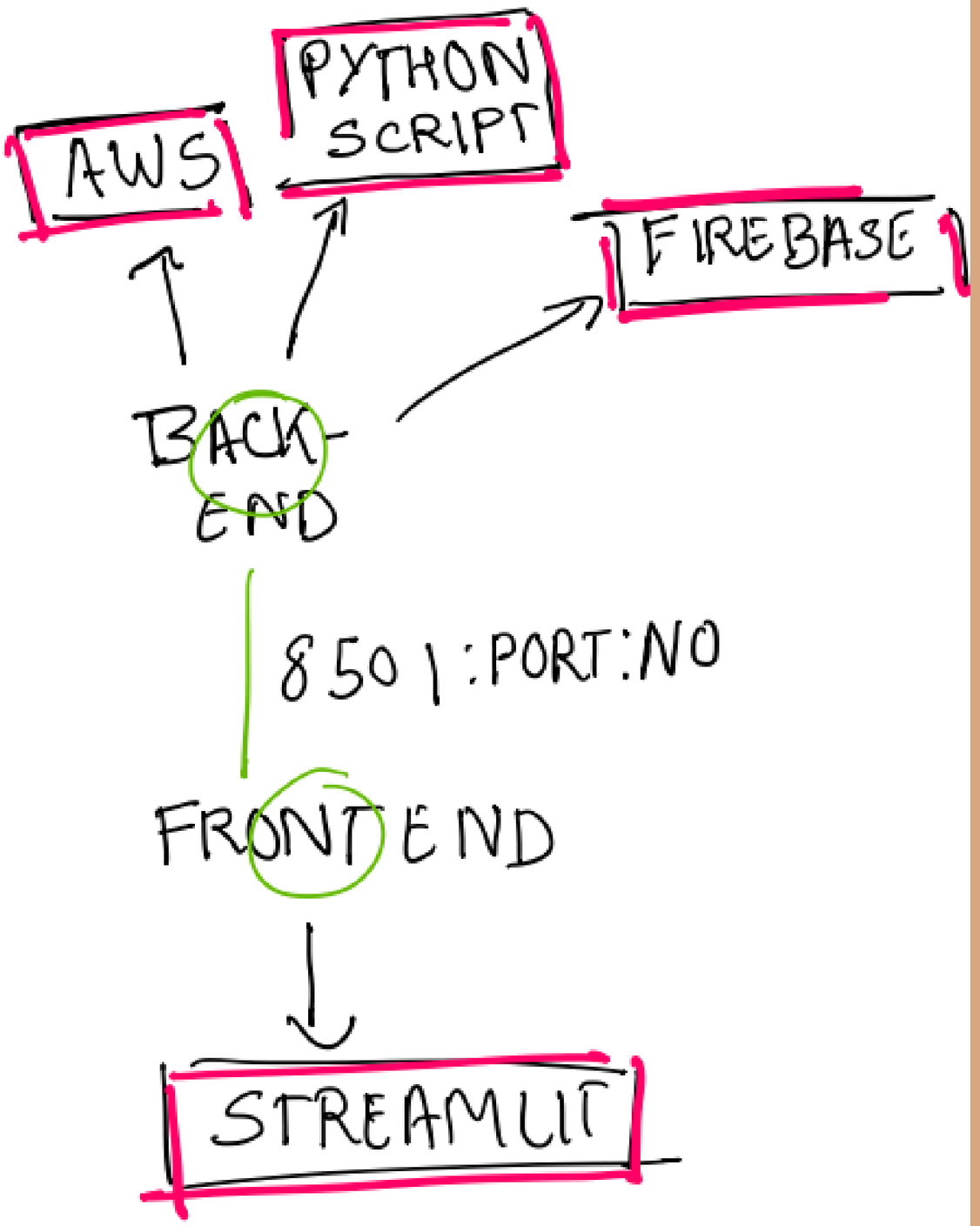
In [ ]:
```
early_stopping_callback = EarlyStopping(monitor = 'val_loss', patience = 10, mode = 'min', restore_best_weights = True)
```


Accuracy Comparison
ConvLSTM
LRCN
Model
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Accuracy

FRONTEND


Welcome to CarMa
Login/SignUp
SignUp
Enter your email address
Enter your password
Enter Department Name
Sign Up
Made with Streamlit

Flowchart of Carma



Features of carma

- MAP feature: After a fight has been detected, a marker is generated right on the location of the CCTV.

- Violence detection: Accurately detects if there is a fight going on or not.

CCTV Location Database

```
In [2]: data = {'Camera Name':  ['PGPR', 'Com2'],
                'Latitude': ['1.291654', '1.294108'],
                'Longitude': ['103.780445', '103.773765']
                }

        df = pd.DataFrame(data)

        print (df)
```

```
  Camera Name  Latitude   Longitude
0        PGPR  1.291654  103.780445
1        Com2  1.294108  103.773765
```

```
In [3]: x = int(input("Enter Camera Number: "))
        location = [df['Latitude'][x],df['Longitude'][x]]
        map = folium.Map(location=location, zoom_start=10)
        folium.Marker(location).add_to(map)
        map
```

```
Enter Camera Number: 1
```

Map with marker at COM 2

Out[3]:

Challenges faced

- ConvLSTM – It proved to be a really slow model because of the large training time and hence we had to choose a different model.
- Overfitting – The performance was increasing but the overfitting was increasing along with it.
- Data – It was tough to find the ideal dataset containing cctv footages for getting the best accuracy. This also happened due to copyright issues which prevented us from using many good datasets.
- Extracting of frames from the video dataset was also very time consuming.
- Instability of Neural Networks
- We had trouble in implementing the GoogleNet model (transfer learning)

Challenges faced

- We tried deploying our application using Azure. It was a new cloud platform for us and therefore we encountered several errors which we weren't able to rectify.
- We tried deploying using streamlit cloud and firebase as well but due to shortage of resources we were yet again not able to deploy it.
- Then finally we deployed using AWS.

Target audience

- Security Authorities and Police Force
- Army and Defence
- Hostels and School parks
- Prisons / Jails

Our Learning

- We learned how to combine CNN(image classification) and RNN(sequential learning) in order to do video classification.

- We learned about transfer learning while making the Googlenet model

- We made 3 models that used 3 different deep learning concepts.

- We Visualized our data using matplotlib through which we go to know how our model was faring and thus kept improving our model.

Future Enhancements

- We will connect our model to a cctv system for realtime violence detection.
- We are going to build a notification system that will send a ping to the app whenever it detection any sort of crime.
- The location should also get updated on the map without any delay so that the police can get to know where the crime is happening.
- Our app will also have a reporting feature so that the public can report any crime they have witnessed to the police and a cctv camera can be setup at that particular location.

References

- *DETECTION OF REAL-WORLD FIGHTS IN SURVEILLANCE VIDEOS*.
  - Mauricio Perez, Alex C. Kot, Anderson Rocha

  It is notable that the accuracy of our model has exceeded the accuracy achieved in this mail

- *VISION-BASED FIGHT DETECTION FROM SURVEILLANCE CAMERAS*
  - S¸ EYMANUR AKTI, GOZDE AYS¸E TATARO ¨ GLU, HAZIM KEMAL EKENEL

THANK YOU