EE-454 Computer Vision

Automated Video Summarization for Suspicious Event Detection By Making Pipeline

Muhammad Azeem Haider
Syed Muhammad Hussain
Muhammad Affanullah Habib

13 May 2023

What is the need for a smart surveillance system?

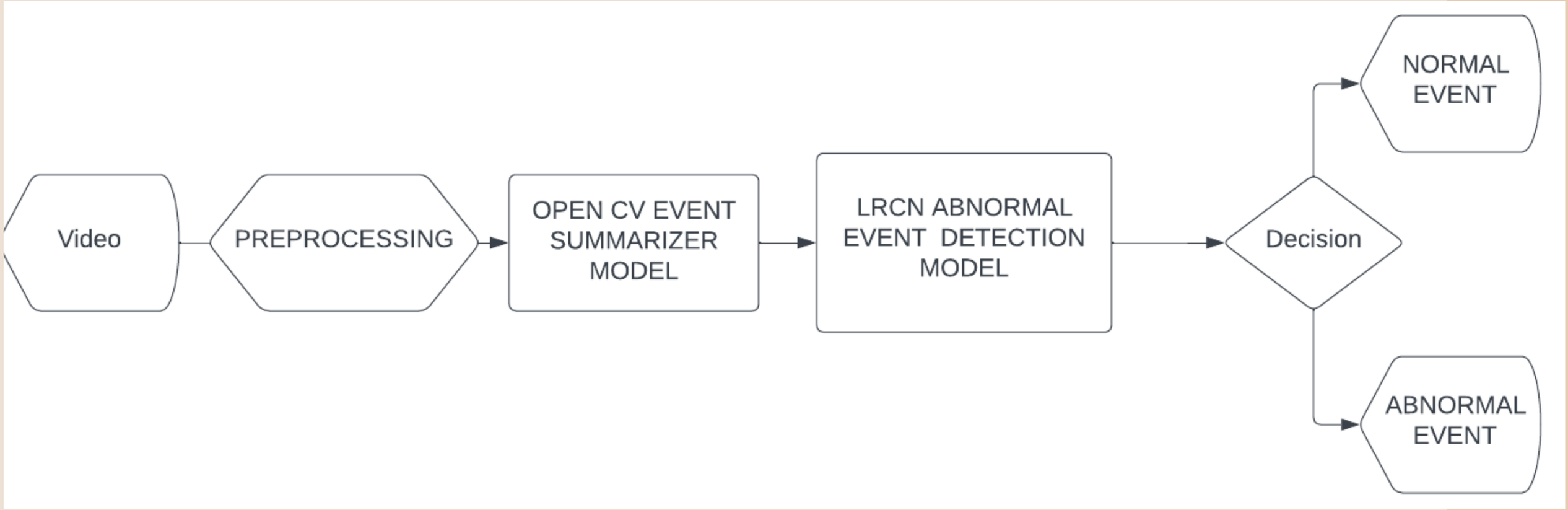
- According to the crime data released by Karachi police, Karachi experienced 81,000 law and order incidents in 2022.
- The number is expected to increase, with street crime increasing by 7% at the start of 2022.

Proposal

Dual Model System that summarizes and detects suspicious activities

- Model 1 - Summarizer using OpenCV
- Model 2 - Activity detection system

Flowchart of our Proposal



Dataset

- DCSASS (Distributed Camera System for Anomaly and Suspicious Spatio-Temporal event detection) dataset was used, publicly available on Kaggle.
- This dataset contains videos based on the following 13 classes: Abuse, Arrest, Arson, Assault, Accident, Burglary, Explosion, Fighting, Robbery, Shooting, Stealing, Shoplifting, and Vandalism.
- The video dataset consists of 3305 videos, each frame of the video labelled as normal or abnormal.

First-Stage Model- Summarizer

- The first-stage model carries out a frame-by-frame comparison of videos using the OpenCV library (CV2) to summarize the videos.
- Unique frames are determined by calculating the absolute difference between the current and previous frames. The current frame is considered unique if the difference is greater than or equal to 10. This simple yet effective method forms the basis of the motion detection algorithm used in the model.

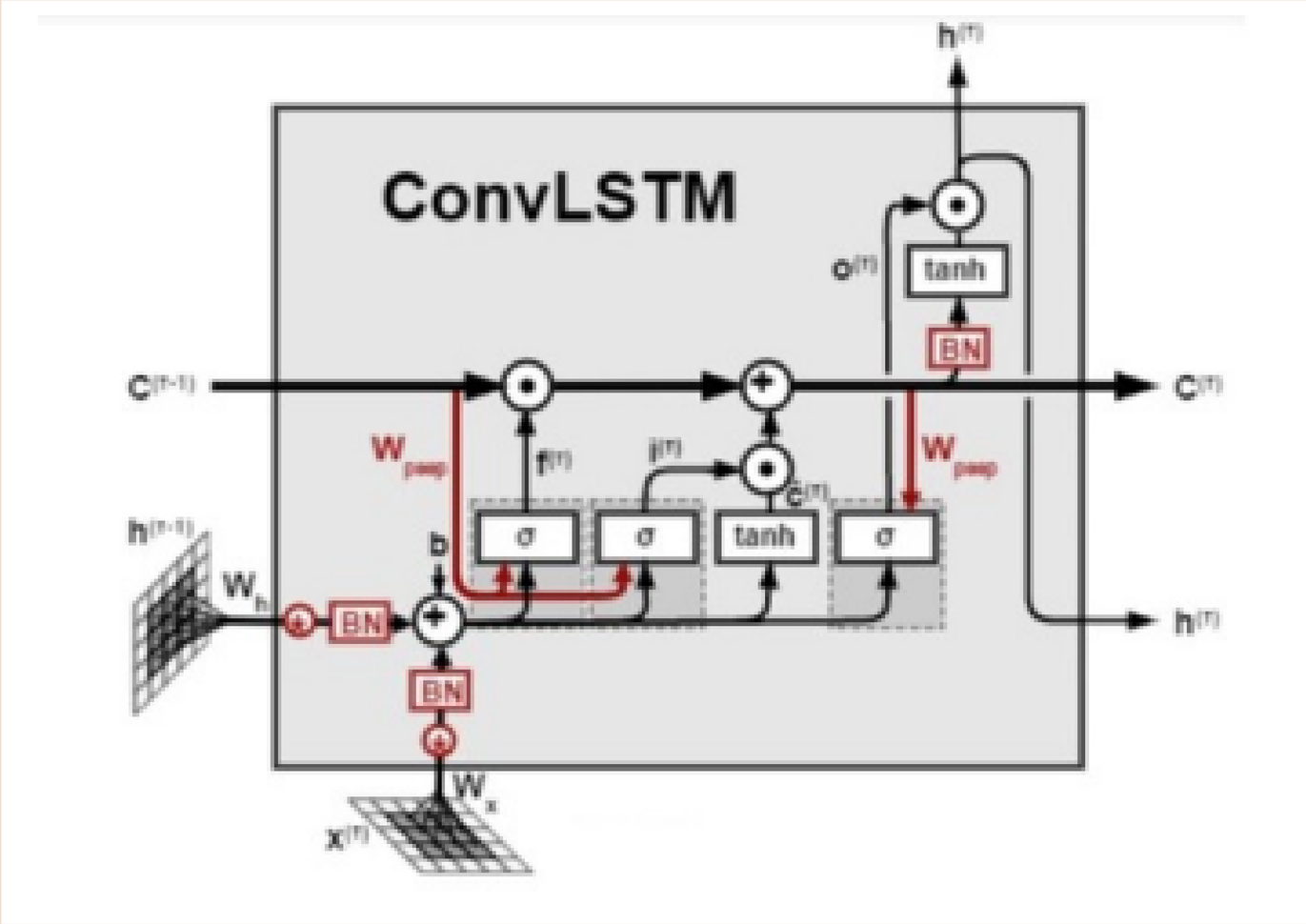
Further Explanation

- The model sets a threshold value to determine whether a frame contains significant changes or not. If the absolute difference between consecutive frames exceeds the threshold, the frame is considered unique and saved in the output video file. This threshold-based approach allows customization and fine-tuning of the sensitivity of the motion detection.
- The model utilizes counters to keep track of the number of unique frames, common frames, and total frames processed. This provides valuable insights into the motion patterns within the video and helps evaluate the effectiveness of the motion detection algorithm.

Further Explanation

- The model generates an output video file that contains only the frames with significant content changes. Additionally, it prints the total number of frames, the number of unique frames, and the number of common frames detected. These statistics provide a quantitative measure of the motion activity within the video and aid in understanding the overall motion dynamics.

Model 2 - ConvLSTM



ConvLSTM

In this step, we implemented 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.

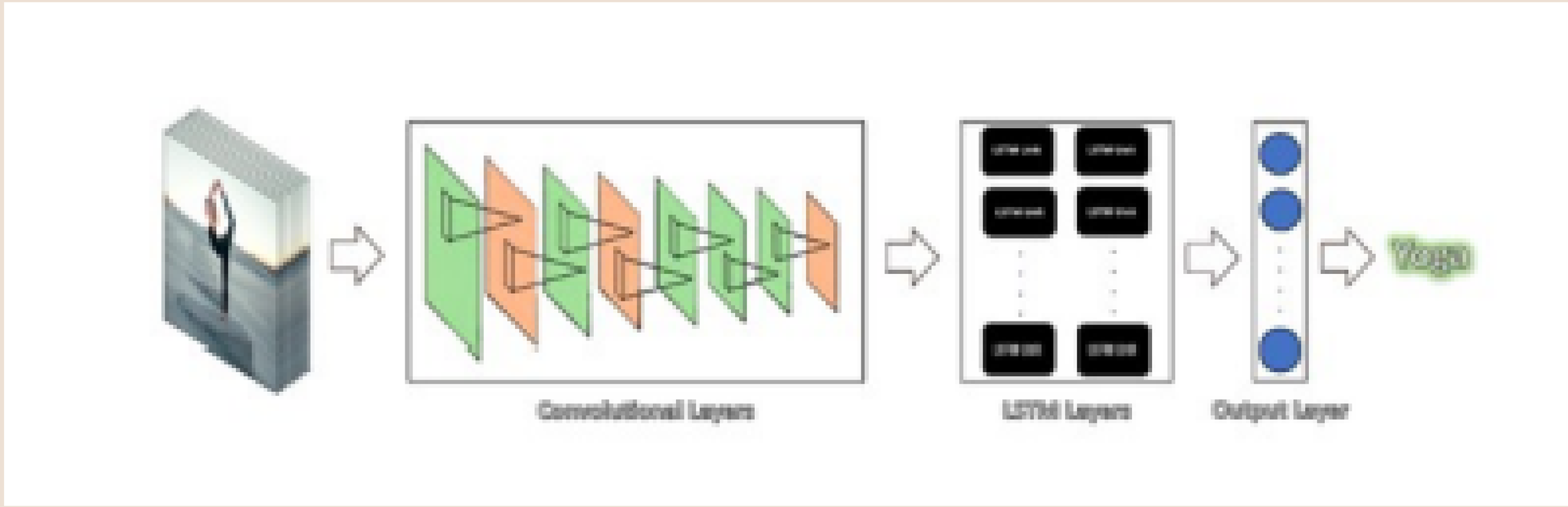
ConvLSTM

For video classification, this approach effectively captures the spatial relation in the individual frames and the temporal relation across the different frames. As a result of this convolution structure, the ConvLSTM is capable of taking in 3-dimensional input (width, height, num_of_channels) whereas a simple LSTM only takes in 1-dimensional input hence an LSTM is incompatible for modeling Spatio-temporal data on its own.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv_lstm2d (ConvLSTM2D)    (None, 10, 62, 62, 4)     1024

 max_pooling3d (MaxPooling3D  (None, 10, 31, 31, 4)    0
 )

 time_distributed (TimeDistr  (None, 10, 31, 31, 4)    0
 ibuted)

 conv_lstm2d_1 (ConvLSTM2D)  (None, 10, 29, 29, 8)     3488

 max_pooling3d_1 (MaxPooling  (None, 10, 15, 15, 8)    0
 3D)

 time_distributed_1 (TimeDis  (None, 10, 15, 15, 8)    0
 tributed)

 conv_lstm2d_2 (ConvLSTM2D)  (None, 10, 13, 13, 14)    11144

 max_pooling3d_2 (MaxPooling  (None, 10, 7, 7, 14)     0
 3D)

 time_distributed_2 (TimeDis  (None, 10, 7, 7, 14)     0
 tributed)

 conv_lstm2d_3 (ConvLSTM2D)  (None, 10, 5, 5, 16)      17344

 max_pooling3d_3 (MaxPooling  (None, 10, 3, 3, 16)     0
 3D)

 flatten (Flatten)           (None, 1440)              0

 dense (Dense)               (None, 2)                 2882

=================================================================
Total params: 35,882
Trainable params: 35,882
Non-trainable params: 0
_________________________________________________________________
Model Created Successfully!
```

ConvLSTM PARAMETERS

Model 2 - LRCN



LRCN

we implemented another approach known as the Long-term Recurrent Convolutional Network (LRCN), which combines CNN and LSTM layers in a single model. The Convolutional layers are used for spatial feature extraction from the frames, and the extracted spatial features are fed to LSTM layer(s) at each time-steps for temporal sequence modeling. This way the network learns spatiotemporal features directly in an end-to-end training, resulting in a robust model.

LRCN

We also used TimeDistributed wrapper layer, which allows applying the same layer to every frame of the video independently. So it makes a layer (around which it is wrapped) capable of taking input of shape (no_of_frames, width, height, num_of_channels) if originally the layer's input shape was (width, height, num_of_channels) which is very beneficial as it allows to input the whole video into the model in a single shot.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 time_distributed_3 (TimeDis  (None, 10, 64, 64, 16)   448
 tributed)

 time_distributed_4 (TimeDis  (None, 10, 16, 16, 16)   0
 tributed)

 time_distributed_5 (TimeDis  (None, 10, 16, 16, 16)   0
 tributed)

 time_distributed_6 (TimeDis  (None, 10, 16, 16, 32)   4640
 tributed)

 time_distributed_7 (TimeDis  (None, 10, 4, 4, 32)     0
 tributed)

 time_distributed_8 (TimeDis  (None, 10, 4, 4, 32)     0
 tributed)

 time_distributed_9 (TimeDis  (None, 10, 4, 4, 64)     18496
 tributed)

 time_distributed_10 (TimeDi  (None, 10, 2, 2, 64)     0
 stributed)

 time_distributed_11 (TimeDi  (None, 10, 2, 2, 64)     0
 stributed)

 time_distributed_12 (TimeDi  (None, 10, 2, 2, 64)     36928
 stributed)

 time_distributed_13 (TimeDi  (None, 10, 1, 1, 64)     0
 stributed)

 time_distributed_14 (TimeDi  (None, 10, 64)           0
 stributed)

 lstm (LSTM)                 (None, 32)                12416

 dense_1 (Dense)             (None, 2)                 66

=================================================================
Total params: 72,994
Trainable params: 72,994
Non-trainable params: 0
```

LRCN PARAMETERS

Results

We successfully trained our two models on three classes of datasets: Explosion, Fighting, Stealing

The result will compare the performance of the two models for each class and the accuracy achieved on the test data for each class.

ConvLSTM versus LRCN

- Why did LRCN perform better in terms of validation accuracy when compared to the ConvLSTM model?
- This is because of how LRCN works and how our data is collected, making it suitable for LRCN to provide higher accuracy.
- The stack of Recurrent sequence models makes it better for LRCN to recognize from a dataset which is frames taken from a video.

Results on ConvLSTM

TABLE I
COMPARISON BETWEEN TRAINING AND VALIDATION ACCURACY

	Explosion	*Fighting*	*Stealing*
Training Accuracy	0.9433	0.9216	0.9607
Validation Accuracy	0.9099	0.8462	0.8614

Results on ConvLSTM

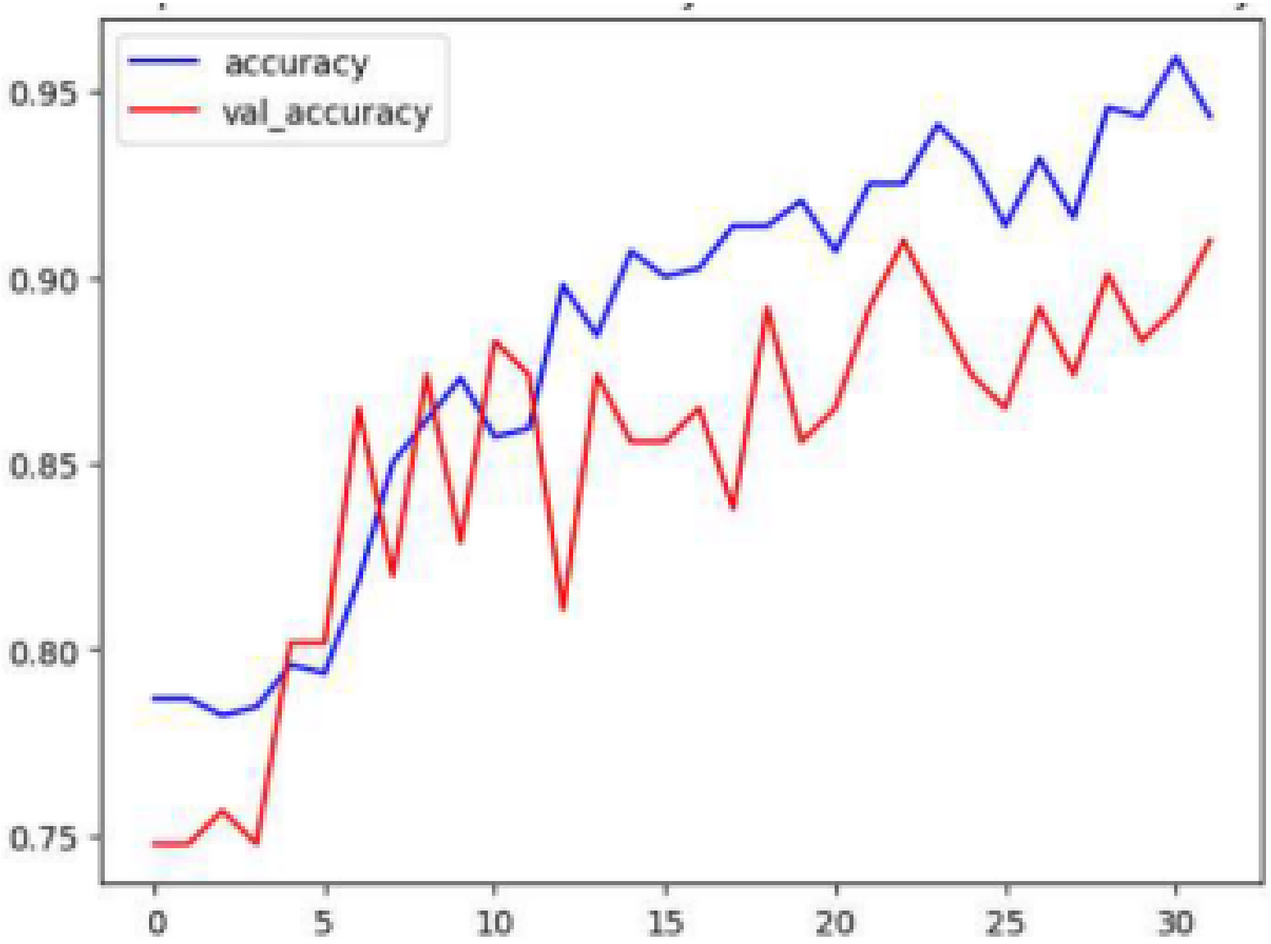


Fig. 4. Training and Validation accuracy for Explosion subclass

Results on ConvLSTM

TABLE II
COMPARISON BETWEEN TRAINING AND VALIDATION LOSS

	Explosion	*Fighting*	*Stealing*
Training Loss	0.1589	0.1776	0.1023
Validation Loss	0.3046	0.4358	0.4440

Results on LRCN

TABLE III
COMPARISON BETWEEN TRAINING AND VALIDATION ACCURACY

	Explosion	*Fighting*	*Stealing*
Training Accuracy	0.9456	0.9608	0.9517
Validation Accuracy	0.9459	0.9487	0.8554

Results on LRCN

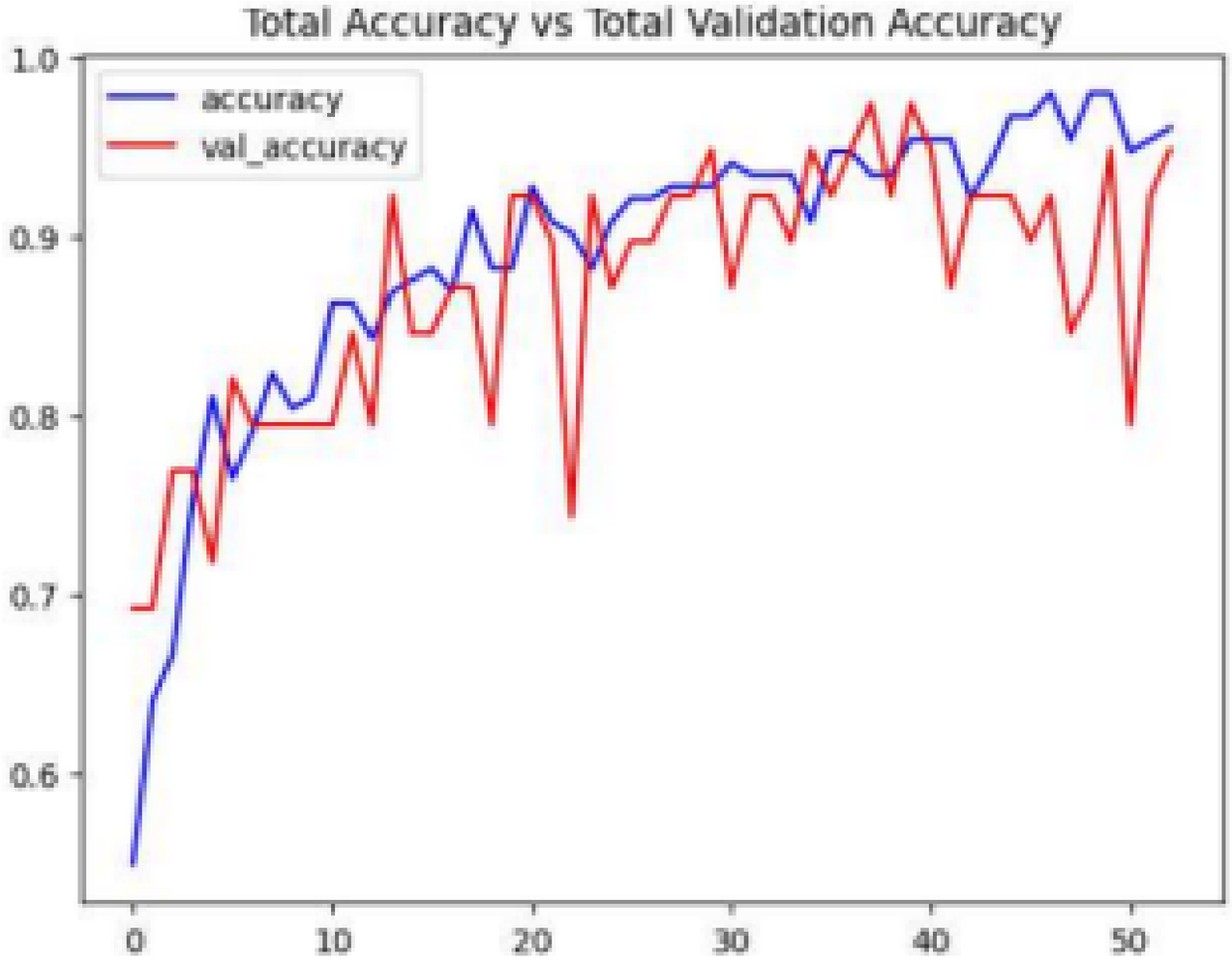


Fig. 5. Training and Validation accuracy for Fighting Subclass

Results on LRCN

TABLE IV
COMPARISON BETWEEN TRAINING AND VALIDATION LOSS

	Explosion	*Fighting*	*Stealing*
Training Loss	0.1524	0.0718	0.1078
Validation Loss	0.2503	0.3214	0.4675

Lackings and future plans

- Train on multiple sub-classes of suspicious activities.
- Train and test the model of local data collected from CCTV footage (through collaboration) and videos collected from internet resources.
- Train the model on all the classes one time which we were unable to do due to the lack of computation power.

Thank you for listening!