

# Crime Detection

Post Graduate Diploma in ARTIFICIAL INTELLIGENCE  
From C-DAC, ACTS (Pune)



Presented by:

John Ealias 200240128011

Lokendra Carpenter 200240128014

Prince Pal 200240128020

Shivam Saini 200240128027

Guided by:

Mr. Manish Gupta

# Introduction

- Surveillance cameras are widely deployed in public places, and the general crime rate has been reduced significantly due to these devices.
- These cameras provide cues and evidence after crimes are conducted, while they are rarely used to prevent or stop criminal activities in time.
- It is both time and labor consuming to manually monitor a large amount of video data from surveillance cameras.
- We have tried various approaches to detect crime and have mentioned the methodology and the results that were obtained from each method.
- Our best approach was by using Flow Gated Network. that utilizes both the merits of 3D-CNNs and optical flow.

# Literature Survey

- **Step-by-Step Deep Learning Tutorial to Build Video Classification Model**

Article to learn how you can use computer vision and deep learning techniques to work with video data

- **Real-world Anomaly Detection in Surveillance Videos**

This article focus upon detecting anomaly data from videos using Multiple instance Learning Approach.

- **RWF-2000: An Open Large Scale Video Database for Violence Detection**

This article focus upon detecting anomaly data from videos using Gated Flow Network with Convolution 3D and Optical Flow.

# **Approaches**

# Previous Approaches

1. Numpy - LSTM Approach
2. Keras Video Generator

# Numpy – LSTM

- This approach involves extracting the frames from each video and converting them into numpy files as part of preprocessing.
- The preprocessed numpy files for each frame are bagged together into a bag of 60 frames.
- Time distributed convolution 2D networks along with max pooling and average pooling are used to process the numpy files and extract image information of each bag.
- The output of time distributed CNNs are passed onto an LSTM network that will extract the sequence information within the frames inside a bag.

# Main Challenges and drawbacks

- Highly unstable accuracy and loss curves at every epoch
- Training one video at a time due to limited ram size
- Both loss and accuracy suffered a fluctuation.
- To remove this we tried the approach using Keras Video Generator and instead of taking full video for normal and anomaly part, we put smaller chunks of video in different folders for normal and anomaly.

# Keras Video Generator

- As part of initial pre-processing, we divided the entire video into anomalous and normal videos by extracting the frames using annotations and created normal videos and anomalous videos of similar sizes.
- Generating videos with frame lengths between 120 and 240 frames.
- These preprocessed videos are then accessed using `keras_video` `VideoFrameGenerator`.
- `VideoFrameGenerator` extracts the videos and labels them as the name of the folder.
- 80 frames are extracted from each video and then fed into a model with time distributed convolution 2D networks.



# Drawback:

- We got a final accuracy of 68% for the training set but the model was not stable.
- Memory wasn't sufficient enough, system was getting out of memory.

# **Final Approach**

# Gated Flow Network

Our model contains four parts:

- RGB channel
  - Optical Flow channel
  - Merging Block
  - Fully Connected Layer
- 
1. RGB channel and Optical Flow channel are made of cascading 3-D CNNs, and they have consistent structures so that their output could be fused.
  2. Merging Block is also composed of basic 3-D CNNs, which is used to process information after self-learned temporal pooling.
  3. The FC layers generate the output.

# Preprocessing

A. Temporal data

B. Uniform sampling

C. Manually cleaned videos

D. Augmentation and Normalisation

# Temporal data

- Going through each video we created an array of
  - Video name
  - Category of the video
  - Start 1 - End 1/Start2- End2 (frames which contains anomaly)
- We created a Dataset containing Temporal Data of each video

# Uniform Sampling

- Extracting anomaly and normal frames of the video separately using dataset.
- Uniformly taking frames from the video.
- Making bags of 64 frame of each category .
- Storing videos in folder category wise for further processing

# Manually cleaned videos

- Going through each video we got some video of different category.
- We seperated videos category wise to make data more clean for training the model

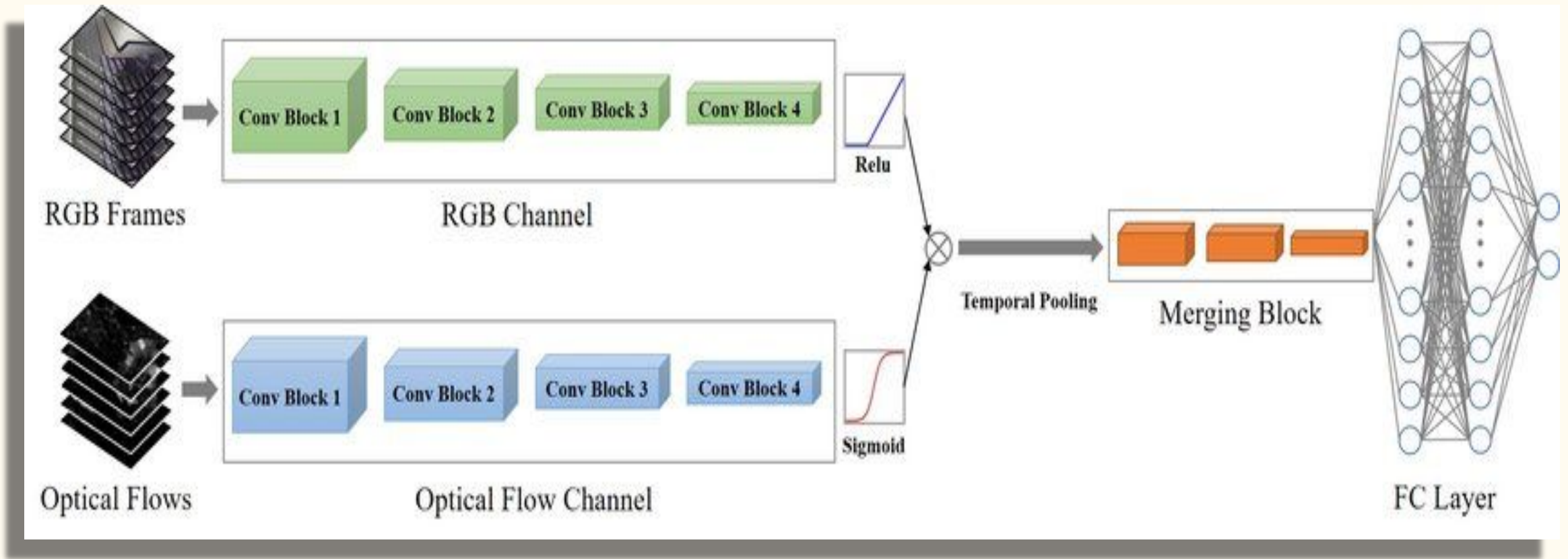
# Augmentation and Normalisation

- Before feeding data to the model for training we did some preprocessing on the data
  - Read videos and converted into numpy array
  - we extracted optical flow from the video and added to the numpy array.
  - Added color jitter, performed Data augmentation and Normalisation.



**Final Approach : Gated Flow Network**

# High Level Diagram

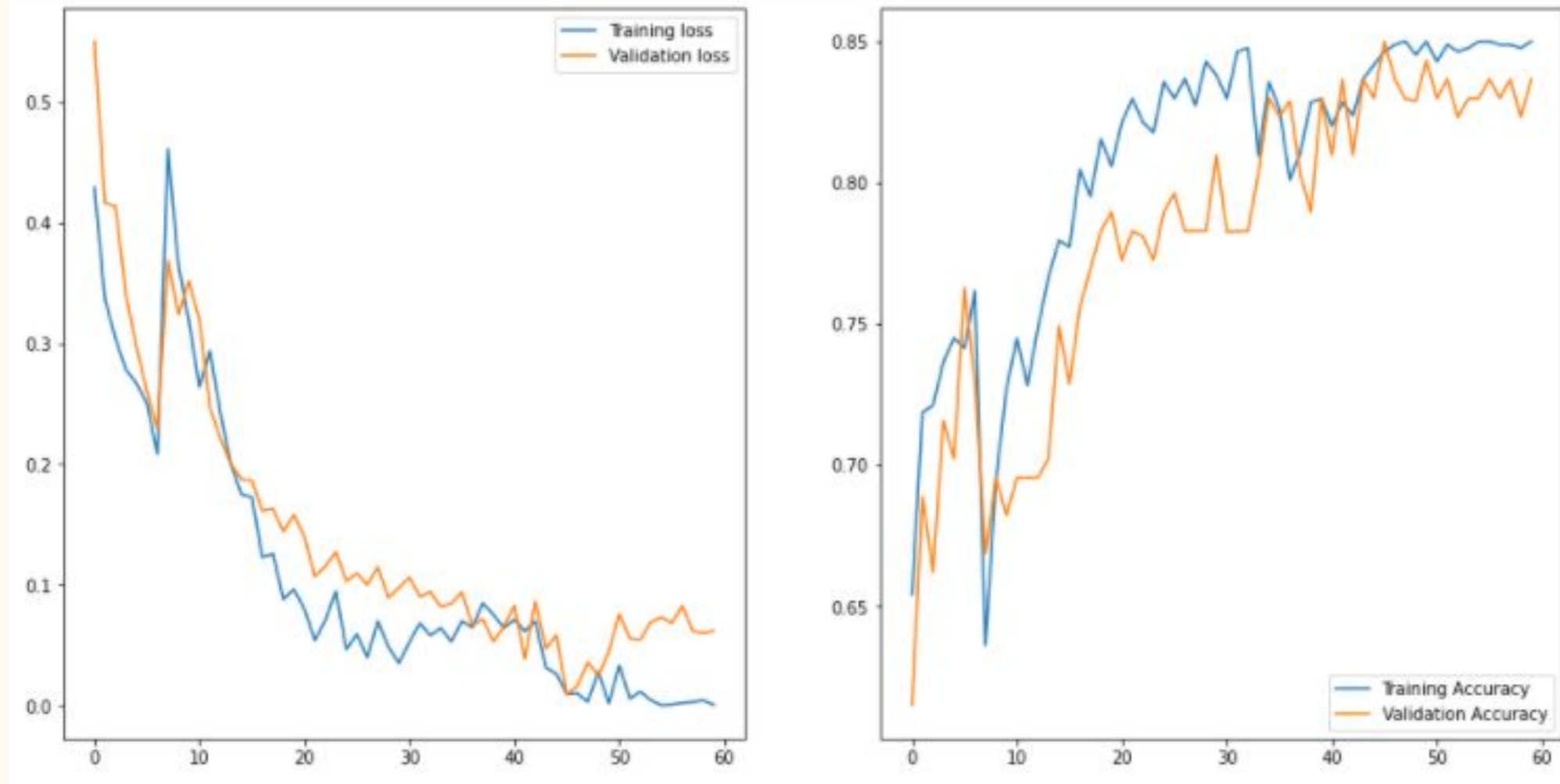


Block Name	Type	Filter Shape	t
RGB/Flow channels	Conv3d	$1 \times 3 \times 3 @ 16$	2
	Conv3d	$3 \times 1 \times 1 @ 16$	
	MaxPool3d	$1 \times 2 \times 2$	2
	Conv3d	$1 \times 3 \times 3 @ 32$	
	Conv3d	$3 \times 1 \times 1 @ 32$	
Fusion and Pooling	MaxPool3d	$1 \times 2 \times 2$	1
	Multiply	None	
	MaxPool3d	$8 \times 1 \times 1$	2
Merging Block	Conv3d	$1 \times 3 \times 3 @ 64$	
	Conv3d	$3 \times 1 \times 1 @ 64$	
	MaxPool3d	$2 \times 2 \times 2$	
FC layers	Conv3d	$1 \times 3 \times 3 @ 128$	1
	Conv3d	$3 \times 1 \times 1 @ 128$	
	MaxPool3d	$2 \times 2 \times 2$	
	FC layer	128	1
	FC layer	128	
	Softmax	2	

Table : Model parameters. The  $t$  represents the number of repeat.

# Model

# Result graph



We were able to get the accuracy of around 84%

# video result

Test Last Checkpoint: Yesterday at 02:24 (unsaved changes)

View Insert Cell Kernel Help Trusted Python


```
def video2numpy(cap):
    frameCount = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    frameWidth = 320
    frameHeight = 240

    buf = np.empty((frameCount, frameHeight, frameWidth, 3))

    fc = 0
    ret = True

    while (fc < frameCount and ret):
        ret, frame = cap.read()
        buf[fc] = frame
        fc += 1
    # print(buf.shape)
    return(buf)

def pred(video_with_OF):
    result_list=[]
    for i in range(len(video_with_OF)//64):
        segment=np.empty((64,224,224,5))
        for i in range(64):
            segment[i]=video_with_OF[i]
            result_list.append(model.predict(segment.reshape(1,64,224,224,5)))
            video_with_OF=video_with_OF[64:,:,:,:]
    return(result_list)
```



# Comparison table

Approaches	Results(Accuracy in percent)
Relationship between Frames within bags	unstable model with average accuracy of 52%
Relationship between bags	unstable model with average accuracy of 55%
Keras Video Generator	better model with slightly improved stability and final accuracy of 68%
Gated Flow Network	best accuracy among all models 84%

# Challenges faced

- Uncleaned data without temporal data of high size
- Gone through each video to create temporal data manually.
- Limited Resources.
- Managing project remotely.

# Conclusion and future scope

- After trying a few approaches and comparing the results, we came to the conclusion that flow gated approach works best in identifying anomaly in a video. Hence we have used this approach in our final model.
- This system can be very helpful and can be used to detect the criminal activities in public places and to stop them in time. Further functionality can also be added such as triggering a message to the nearest police station in case of any crime.
- Also this system is not limited to detect only criminal activities but can also be used to detect any other anomaly like road accidents, fire etc and report it to the nearest hospital, fire department or any other concerned authority in time and help in saving many lives .