# Violent / Non-violent Behavior Analysis from Crowd Video Footages

Himadri Sankar Chatterjee, Kavitha Br.

*Student (MCA),*
*SITE,*
*Vellore Institute of Technology,*
*Vellore.*
himadri.sanakr2019@vitstudent.ac.in

*Assistant Professor (Senior),*
*Department of Smart Computing, SITE,*
*Vellore Institute of Technology,*
*Vellore.*
kavitha.br@vit.ac.in

***Abstract*— With increase in the population, surveillance over the behavior of crowd have become a challenging task. With a greater number of surveillance cameras being installed to effectively carry out surveillance over erratic crowd behavior, the burden just increases. We propose a simple dee learning method to automate the task of monitoring these video footages to identify erratic or unusual behavior in the videos to help initiate prompt reaction.**

***Keywords*— crowd behaviour analysis, 3d-cnn, c3d network, video analysis, violent flow**

## I. Introduction

The world has seen a huge rise in the number of protest and sudden outbreak among common people, that has given birth to a larger crisis in the later stage. Governments have tried hard to initiate quick response teams and monitor the video footages to ultimately reduce the number of casualties that might occur. It's a huge task to keep track of all the events that are occurring and thus help in fighting for the cause. Automating the methods prove to be a challenging task due to the existence of highly complex interactions between the objects present in the video footages and the relative change of their interactions over a time period. This huge information being contained over a number of consecutive frames in a video is quite large to process and still takes a lot of time for computation, in some modern computers. Work has been in progress to reduce the time complexity of these pre-existing methods of automation to decrease the time constraint. Apart from hand-engineered feature extraction methods from video footages, there has been very less significant work on analysing crowd video footages and classifying the behaviour into "Violent" or "Non-violent". In this work, we try to apply a very basic deep learning model, the 3D Convolutional Neural Network Architecture, to classify video footages into the respective classes. We have used Violent-Flows dataset, consisting of around 200 videos to be classified into violent/non-violent.

## II. Literature Review

A simple analysis of previous work:

Previous work has been carried out to perform action recognition and simple behavior detection over video footages. The methods used were:

- Acquiring *Flow Based Features* for classification.
    - Optical Flow
    - Particle Flow
    - Streak Flow
- Acquiring *Local Spatio-Temporal Features* for classification
    - Spatio-temporal Gradients
- Trajectory / Tracklets

.

### *1) With the help of Flow Based Features*

For high density crowded scenes, while tracking of each individual is impossible, but the movement of the crowd together provides all the necessary information. Optical Flow is to compute pixel-wise instantaneous movements between consecutive frames. Horn et al. in the paper "Learning motion patterns in crowded scenes using motion flow field" [10], executed the Optical Flow analysis on crowd footages for various classes. Particle Flow, is based on the notion of movement of particles from fluid dynamics, also attained fairly good result in the task of modelling the movement of the crowd from video footages. S Ali et al. in their paper "A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis", attained good results in analysing the crowd behaviour with Particle Flow.

Our work is originally based on the paper by Tal Hassner et al. titled as Violent Flows: Real-Time Detection of Violent Crowd Behaviour [2]. They estimated the optical flow between two consecutive frames of the video footages, to create the Violence Flow vector that is then used for the task of classification.

### *2) By Acquiring Local Spatio-Temporal Features*

Some extremely crowded scenes are les structural due to the high variability of pedestrian movement. Spatio-temporal Gradients, or the 3D gradients of each pixel collectively represent characteristic motion pattern within the patch. L Kratz et al. in the paper "Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models" [6], carried out spatio-temporal analysis and was successful in identifying "unusual activities" in videos.

### *3) With the help of Tracklets*

Typical crowd behavior can be better analyzed based on motion features extracted from the trajectories of objects. The Tracklets methods is more semantic and is more suited for classification of the video footages. B Zhou et al. in their paper "Random field topic model for semantic region analysis in crowded scenes from tracklets" [9], introduces the idea of tracklets which were better at tracking individuals throughout the video footages.

Another work by R Pillai et al. in the paper titled “Crowd behavior Analysis Using 3d Convolutional Neural Network” [15], performed the task of classification of video footages of crowd movement in to two specific categories:

1. *Blocking:* A unusual gathering or block of crowd that acts as a blockage to the normal movement of people.
2. *Lanes:* The movement of the crowd is following the pattern of an almost straight line.

## III. Our Approach

We have decided to approach our task using Deep Learning models, which have been proved to be effective if learning robustly from more features of the data that were previously unknown to us. We studied the various deep learning models designed to be applied to the task of action classification. One of the very early models among them is the 3D CNN architecture. Since, our goal is to classify the videos among *two* categories, we

### *A.* The C3D Model

Our model is the C3D model or the 3D Convolutional Neural Network model, that has performed well in the task of classification of videos into various classes based on the activity being performed that is being portrayed through it.

The key features of the model are:

- Repurposing 3D convolutional networks as feature extractors
- Extensive search for best 3D convolutional kernel and architecture
- Using deconvolutional layers to interpret model decision

C3D starts by focusing on appearance on the first few frames and then track the salient motion in the subsequent frames. The working concept is similar to that of the 2D CNN, except that it now works on a volume of image frames, instead of one. Fig 1.1 tries to explain the architecture used in implanting the C3D model.

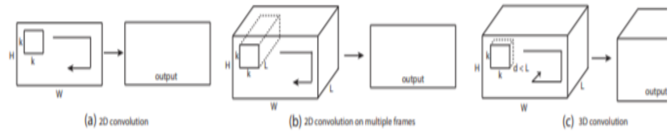

Fig 1.1

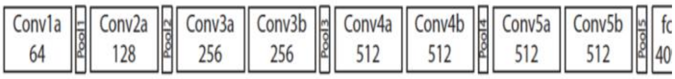

Fig 1.2

### *B.* Data Used

For our task, which will specifically classify the crowd video footages into two specific classes, viz. Violent / Non-Violent, we have decided to work on the Violent Flow dataset.

The Violent Flow Dataset is a database of real-world, video footage of crowd violence, along with standard benchmark protocols designed to test both violent/non-violent classification and violence outbreak detections.

The data set contains **246** videos. All the videos were downloaded from YouTube. The shortest clip duration is 1.04 seconds, the longest clip is 6.52 seconds, and the average length of a video clip is 3.60 seconds. Each of the videos are of 224 X 224 dimensions.

The data was eventually divided into 3 sets. We brought together all the videos under a single folder. This revealed various duplicated videos residing in the main dataset. We did manual data cleaning to remove the duplicates. We, then modified the different csv files for each set, to create a final csv file that contains the information of all the videos together. This revealed various complications, as in some cases the name of the video file did not math with the name in the csv file. We had to further jump into cleaning the data and correcting the names of the file in the csv dataset, to match it with the original file, that will help in loading the data during training and testing.

Our final csv files included the following information:

- **id:** This gives a unique id to the video in the list. This is used as a reference during random shuffling of the data at the preparation of the training and the testing data.
- **simplified_video_title:** This denotes the title of the original video on YouTube, from where a portion of data is clipped.
- **search_keyword:** This include various keywords related to the video. Like the scenes depicted, the theme of the video and others.
- **nickname:** This contains the nickname of the person who uploaded the video on YouTube.
- **video_identifier:** This denotes the unique video key to refer the video online.
- **video_url**: This include the complete video YouTube link.
- **range:** Since these are small clips of the original video, this denotes the starting and the ending point of the original video that is clipped to prepare the input in the dataset.
- **tag:** Finally, this include the tag of either *“violent”* or *“non-violent”*. This is the final prediction of each data.

For, out task, we needed only the name and the id of the video to refer it from the disk and the final tag of the video for learning and predicting. So, during the preparation of data, we extract only the **“id”, “simplified_video_title”** and the **“tag”** of the corresponding videos into our dataframe, that is sent to the model for training and testing.

### *C.* The Modules of the implementation

- **c3d_model.py:** This contains the implemented C3D model, based on the original architecture, but tweaked to perform more effectively for our task. We introduced a few more layers in the FC layer and modified the optimizers in few of the layers.

Fig 1.3: Frames of Non-Violent Video Footages

Fig 1.4: Frames of Violent Video Footages

- **frames_extract.py:** This script is used to process each video and extract each of the frames. These frames are then resized to 64 X 64 images, color corrected and then stored in the folders corresponding to each video.
- **process_getext.sh:** This shell script is used to prepare the Training and the Testing list, based on the 7:3 ratio of the total video dataset.
- **test.list & train.list:** Holds the path to the folders containing the frames of each video, that is referenced during the training and the testing phase, respectively.
- **train_c3d.py:** This python script contains the methods required to train the model on the image frames in batches of 16 videos together.
- **eval_c3d.py:** This python script contains the methods required to test the model on the image frames from the testing list.
- **input_data.py:** This script has the methods defined to read each image frames from the folder whenever they are accessed through the train_c3d.py methods or the eval_c3d.py methods.
- **video_classification.ipynb:** This includes the jupyter file that is used to run the training and the testing files in Google Colabs.

### *D. Training of the data*

We begin by loading the complete dataset and the code on Google Colabs. Based on previous attempts, we realized that, it was not possible to fit the complete data into the of our systems to complete the training. So, we shifted to the free online solution available at hand. Google Colabs offers an initial 12GB of RAM and 328 GB of free storage with a NVIDIA Tesla K80 GPU, available for a span of twenty-four hours. We are training the model from scratch on the given dataset.

The following figure defines our workflow through the Flow Diagram:

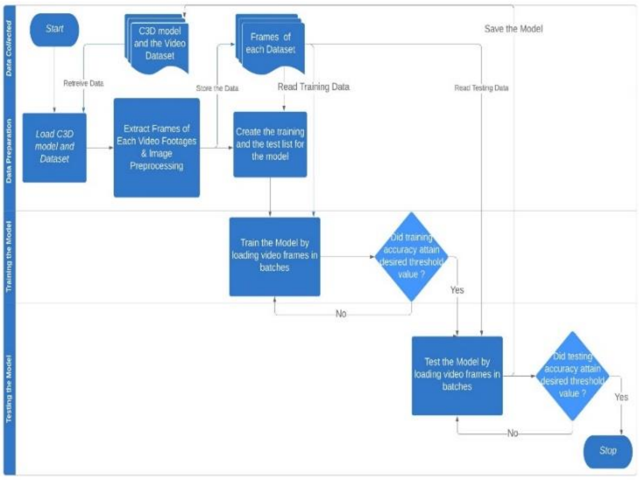
Data Collected
Data Preparation
Training the Model
Testing the Model
Start
C3D model and the Video Dataset
Frames of each Dataset
Save the Model
Retreive Data
Store the Data
Read Training Data
Read Testing Data
Load C3D model and Dataset
Extract Frames of Each Video Footages & Image Preprocessing
Create the training and the test list for the model
Train the Model by loading video frames in batches
Did training accuracy attain desired threshold value ?
Yes
No
Test the Model by loading video frames in batches
Did testing accuracy attain desired threshold value ?
Yes
No
Stop

Fig 1.5

We begin by loading the frame images of videos together in batches of 16. Each of the frames has already been cropped into 64X64 dimensions with 3 channels for RGB input and kept into their respective folders. The videos are chosen randomly from the training list and their corresponding set of images are loaded into the system for training. We then grain the model for 200 steps, for each batch, while calculating the validation accuracy of the model after each 100 steps.

We then proceed to load the next batch of data from the dataset and train the model. We continue this process for 5 epochs. Every time, we had to maintain the RAM, such that it does not overflows and reinitializes.

We were also constrained by the amount of free memory that is provided on Google Drive. So, it was not possible to save all the trained model after each batch in the drive. We had to manually delete all the pre-trained models except the last one and run training again. Every training would load the last trained model into RAM and further train this model on the dataset.

For the testing phase, we had to follow a similar procedure across all the batches of data tested. This would return a **testing_accuracy** for each bath of the **test_data.** The final **testing_accuracy** is the average of all the accuracy across the various batches.

The total time required to complete the training and testing of the model in Colab was around 9 hours.

## RESULTS

We completed the training of the model in Google Colab and recorded the training and the testing accuracy of the model.

We evaluated the model on our test dataset in Google Colab and found that the model was able to achieve an accuracy of only 61.2%. This is almost good, compared to the existing hand-engineered classification algorithms. The following graph shows the variation of the training and the testing accuracy of the model throughout the steps:

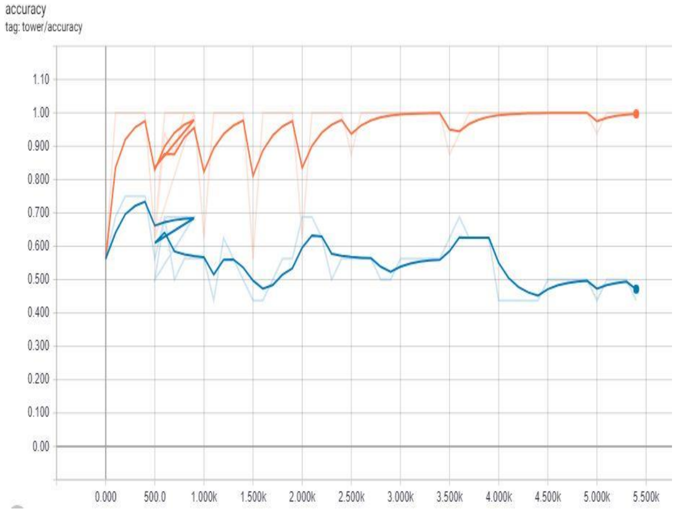

Fig 1.5

The orange line denotes the training accuracy. The sudden rise in the curve at the beginning suggested that the model acquires very high accuracy after some initial training. This slowly converges to achieving a perfect score.

The blue line denotes the validation accuracy for each batch that was trained.

## Conclusion, Constarints & Future Work

From the test result that we receive, we can conclude that the C3D model is not suitable enough for the task of classification of video footages. We might have to refine the dataset to include more variations to facilitate it to learn more features. Due to the constraint of limited computation power and access to limited online memory, out training have not been exactly based on the guidelines. Access to stable systems with better performance capabilities is a must to train these huge models, to help them make better classifiers.

In future work, we aim to collect more data for the task. The dataset must be more detailed so that the model is able to capture more data from the videos. We also plan on applying various other video classification models for the task of training the data and finalizing on the model achieving the best accuracy. For further improvement, we have to integrate Optical Flow, that gives valuable information from the data, that might be useful for the task at hand. The I3D model has shown better performance at classifying the videos into various classes. We also aim to implement this model for classifying the videos.

## References

[1] Ji, Shuiwang, et al. "3D convolutional neural networks for human action recognition." *IEEE transactions on pattern analysis and machine intelligence* 35.1 (2012): 221-231.

[2] Hassner, Tal, Yossi Itcher, and Orit Kliper-Gross. "Violent flows: Real-time detection of violent crowd behavior." *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2012.

[3] Du, Tran, et al. "C3d: Generic features for video analysis." *Eprint Arxiv* 2.8 (2014).

[4] Tran, Du, et al. "Learning spatiotemporal features with 3d convolutional networks." *Proceedings of the IEEE international conference on computer vision*. 2015.

[5] Ma, Ruihua, et al. "On pixel count based crowd density estimation for visual surveillance." *IEEE Conference on Cybernetics and Intelligent Systems, 2004.*. Vol. 1. IEEE, 2004.

[6] Kratz, Louis, and Ko Nishino. "Tracking pedestrians using local spatio-temporal motion patterns in extremely crowded scenes." *IEEE transactions on pattern analysis and machine intelligence* 34.5 (2011): 987-1002.

[7] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.

[8] Pang, Shuchao, et al. "Deep learning to frame objects for visual target tracking." *Engineering Applications of Artificial Intelligence* 65 (2017): 406-420.

[9] Zhou, Bolei, Xiaogang Wang, and Xiaoou Tang. "Random field topic model for semantic region analysis in crowded scenes from tracklets." *CVPR 2011*. IEEE, 2011.

[10] Horn, Berthold KP, and Brian G. Schunck. "Determining optical flow." *Techniques and Applications of Image Understanding*. Vol. 281. International Society for Optics and Photonics, 1981.

[11] Cao, Tian, et al. "Abnormal crowd motion analysis." *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2009.

[12] Najva, N., and K. Edet Bijoy. "SIFT and tensor based object detection and classification in videos using deep neural networks." *Procedia Computer Science* 93 (2016): 351-358.

[13] Arunnehru, J., G. Chamundeeswari, and S. Prasanna Bharathi. "Human action recognition using 3D convolutional neural networks with 3D motion cuboids in surveillance videos." *Procedia computer science* 133 (2018): 471-477.

[14] Sreenu, G., and MA Saleem Durai. "Intelligent video surveillance: a review through deep learning techniques for crowd analysis." *Journal of Big Data* 6.1 (2019): 48.

[15] Pillai, Divya R., and P. Nandakumar. "Crowd behavior Analysis Using 3d Convolutional Neural Network."