ANTAGONISM DETECTION IN VIDEO SEQUENCE

M Poonkudi, Ayush Shekhar,Pratik Singhal, Tarun Kumar Sharma
Assistant Professor (Sr.G), Student, Student, Student
Department of Computer Science and Engineering,
SRM Institute of Science and Technology, Chennai, India.

***Abstract*—**The world we live in today has an utmost importance to have a video surveillance system for detecting any kind of violent behaviours, for example, airports, railway stations, etc. In the not so distant past, the rate of violence has increased drastically. However, the traditional violence detection system uses low-level feature extraction along with other Sapio-Temporal features models developed before in order to extract high-level features. The existing systems in place are able to detect the violent footage in videos using the traditional methods. These methods include motion regions segmentation as per the distribution of optical flow fields. This is done by using low-level features, extracted from RGB images, using the very well known "Local Histogram of Oriented Gradient", or LHOG and from optical flow images using the method of "Local Histogram of Optical Flow", or LHOF. Further in the process, the features that were extracted from the images are coded using the "Bag of Words" model, or BoW, in order to eliminate all redundant information and as a result of this step, a specific-length vector is obtained for each of the video clips and then they are classified using "Support Vector Machine", or SVM. The proposed system uses high-level feature extraction using Convolutional Neural Networks(CNN).

IndexTerms - Violence Detection, CNN, Transfer Learning, Deep Learning, VGG16.

I. Introduction

With the fast paced progress in the field of digital media, video content has become abundantly available online. This has opened up the scope for a number of applications, including recognizing human activity in videos. In recent years, researchers have given a lot of attention to detecting violence in videos. The primary problem in finding a solution to this problem is that the perception of violence varies from one person to another. So, we decided to approach this problem using the definition provided in VSD[1], which defines violence as physical violence, or action resulting in injury.

In order to completely implement violence detection in surveillance video footage, we need to take into consideration the crowd scene analysis problem. [2]

However, for this system we are focusing on person-to-person interactions that are violent in nature in short distance video sequences. The conventional approach that has been implemented to attain these results include the bag of words approach pipeline for generic human actions.The essential aspects involved in this method is proper feature extraction to model human actions. Improved dense trajectory [3] can extract Motion Boundary Histogram (MBH), “Histogram of Oriented Gradients”, or HOG, and Histogram of Optical Flows (HOF), which are feature descriptors for every trajectory. For action recognition problems trajectory based methods have proven to be considerably effective. Since action recognition has very close similarities to violence detection, a lot of recent violence detection methods [4,5] have used the trajectory-based framework to develop a solution.

While these existing solutions are efficient and do produce results, they are not perfectly suitable for violence detection especially in real time surveillance, because these mentioned frameworks do not properly utilise the high level features of the videos which can potentially make the system run on less computational capacity and result in faster runtimes. The objective of this paper is to show that a versatile and robust system can be implemented using the Hockey Fights Dataset[6] which also involves camera movement.

II. Related Works

One of the initial proposals in Violence detection in videos was Nam et al. [7] which proposed detecting the elements of blood and fire to recognize a violent scene, as well as capturing the degree of motion, and the sounds that are characteristic to violent actions. Cheng et al. [8] proposed recognizing gunshots, explosions and car braking sounds using the Hidden Markov models, or HMM and the Gaussian mixture models. More recently, Gong et al. [9] have proposed a method to use low-level visual and auditory features and high-level audio effects to implement a violence detector for identifying potential violent content in movies.

While all of these methods and approaches have proven to give results, the questions which is still pertinent is that in a number of surveillance scenarios, the attributes of color and audio cues may not be present because the footage may not have audio or may be in grayscale, these situations can render the existing systems to be unimplementable.

This paper focuses on proposing a system that does not rely on audio or color cues, but can cover more practical scopes such as camera movement and low resolution video footage.

III. Methodology

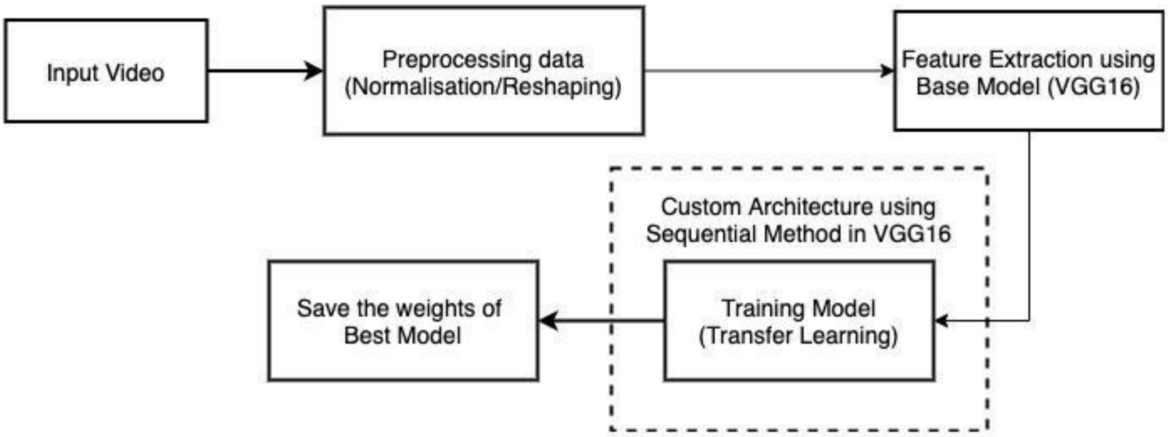

Input Video
Preprocessing data (Normalisation/Reshaping)
Feature Extraction using Base Model (VGG16)
Custom Architecture using Sequential Method in VGG16
Training Model (Transfer Learning)
Save the weights of Best Model

Fig: Architecture Diagram

Model and Algorithm:

Convolutional Neural Network or CNN is used in the implementation of the proposed system. Similar to neural networks, CNNs are formed by a number of neurons which entail the capacity to receive several inputs, and then take a weighted sum over them to pass it through an activation function and finally respond with an output. A loss function is present for the whole network.

We chose to implement our system with CNN because of the proven success that researchers have achieved by implementing this architecture on the Imagenet database and used it for a variety of scopes like Image Recognition, Object detection, etc.

Optimization Algorithm (ADAM):

Adam is an enhancement calculation that can be used in place of the old-style stochastic gradient strategy to refresh and organize loads iteratively dependent on training data.

The algorithm is called Adam and it is not an acronym and is not written as “ADAM”. Adam is deduced from Adaptive Moment Estimation.

Adam understands the advantages of “RMSProp” and “AdaGrad”. “Adaptive Gradient Algorithm”, or AdaGrad that keeps up a per-parameter learning rate that improves execution on issues with scanty slopes (for example natural language and computer vision issues).

“Root Mean Square Propagation”, or RMSProp that additionally keeps up per-parameter learning rates that are adjusted depending on the normal of ongoing sizes of the slopes for the weight (for example how rapidly it is evolving). This implies the calculation excels on both non-stationary issues, for example noisy, and the web. Rather than adjusting the parameter learning rates dependent on the normal first minute (the mean) as in RMSProp, Adam additionally utilizes the average of the second moments of the gradients.

What the algorithm specifically does, is that it calculates an exponential moving average of the both the gradient and the squared gradient. Also the two parameters beta1 and beta2 are responsible for controlling the rates of decay for these moving averages.

The initial value of the two parameters beta1 and beta2, and the moving averages, value close to 1.0, which is recommended, and it therefore causes the bias of moment estimates towards zero.This bias is overcome by first computing the estimates that are biased and then calculating the estimates that are bias-corrected.

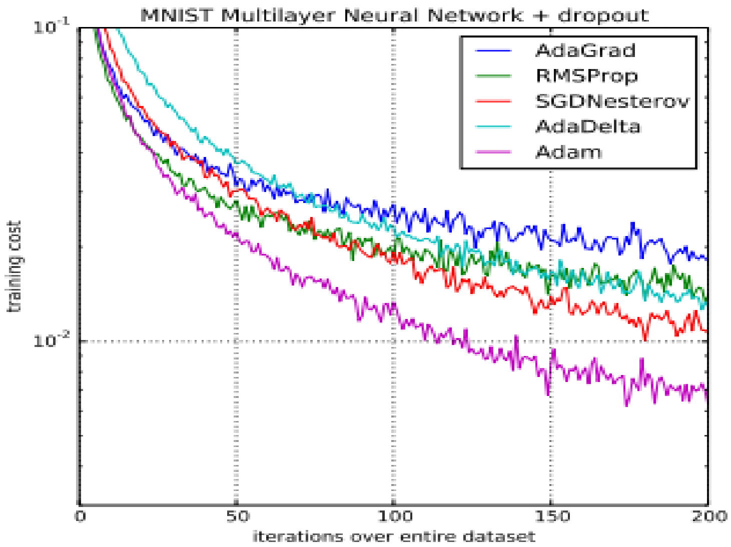


Fig: Comparison of Adam with Other Optimization Algorithms while performing the training of a “Multilayer Perceptron”.

Transfer Learning:

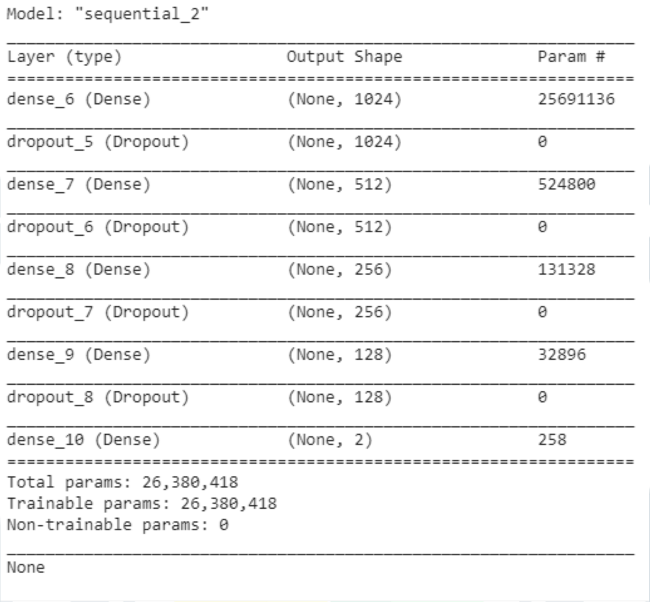
Transfer learning is the process of utilizing the knowledge stored as a result of solving a machine learning problem and applying it to solve a different problem. The process of transfer learning is widely used across the industry for a variety of commercial applications as it is simple, fast and convenient.

The preprocessing required in CNN is significantly less when put in comparison with other classification algorithms. In crude techniques the filters are manually designed but with considerable training CNNs can become familiar with these filters and attributes.

The architecture of a CNN closely resembles that of the network example of Neurons in the Human Cerebrum and was enlivened by associating it with the Visual Cortex. A receptive field is the limited area of the visual field in which the singular neurons react to upgrades. An assortment of such fields overlap so that the whole visual region is covered. In this system, we will be using a pre-trained CNN model VGG16, and retrain it to utilize our dataset for classifying videos as Violent or otherwise.

Creating Model Architecture:

We fetch the pre-trained VGG16 model from Keras and define a custom architecture for the top five layers. We follow the sequential method of arranging layers in the VGG16 model. We provide Rectified Linear Unit as a parameter for ensuring that negative values are not passed on to the successive layers and the final layer is configured with the SOFTMAX parameter to output a value of 0 or 1 depending on the confidence that the model has regarding the label of the image.



```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_6 (Dense)              (None, 1024)              25691136  
_________________________________________________________________
dropout_5 (Dropout)          (None, 1024)              0         
_________________________________________________________________
dense_7 (Dense)              (None, 512)               524800    
_________________________________________________________________
dropout_6 (Dropout)          (None, 512)               0         
_________________________________________________________________
dense_8 (Dense)              (None, 256)               131328    
_________________________________________________________________
dropout_7 (Dropout)          (None, 256)               0         
_________________________________________________________________
dense_9 (Dense)              (None, 128)               32896     
_________________________________________________________________
dropout_8 (Dropout)          (None, 128)               0         
_________________________________________________________________
dense_10 (Dense)             (None, 2)                 258       
=================================================================
Total params: 26,380,418
Trainable params: 26,380,418
Non-trainable params: 0
_________________________________________________________________
None
```

Training the model:

Here the idea is to classify each frame in either category using Convolutional Neural Network, specifically the VGG16 pre-trained model. Once training of the model is complete, the weights of the model are explicitly saved in a file. The optimum model is decided on the basis of validation loss. The model is first compiled using a particular loss function, optimizer and metric.

IV. Experimental Analysis

Setup:

The experiment was carried out in a Google Colaboratory hosted cloud environment. The environment had 25.51 GB of RAM, 68.40 GB of Disk Space, and Google Compute Engine based GPU acceleration. We use Keras with a Tensorflow 1.x backend and Python 3 to implement the system.

Dataset:

The Hockey Fights Data set that we are making use of is a standard dataset that is used by researchers and academicians worldwide for similar projects. It is best suited to prepare a proof of concept for real-time violence detection in surveillance because the videos have a sub-standard resolution and capture only the relevant actions, important to the training of the deep learning model.

Using the dataset:

Using the git tool for version control, the Hockey Fights Dataset is imported into the Google Colaboratory environment from a 3rd party open source repository, which has obtained the data from the official Academic Torrents source.

Preprocessing the data:

We have defined a standard convention for naming the dataset video files and stored the list of file names in txt files that act as config files and drive the entire system.

Processing the dataset:

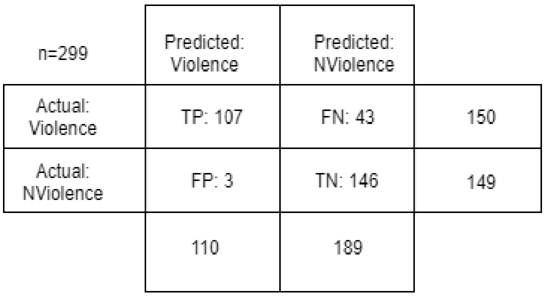
The next step is to create the actual dataset of images from the obtained videos, which will be used for training our model. Using OpenCV, a popular open-source framework for processing images and videos, we extract the individual frames from the videos, catalogue them and store them for training in a separate directory.

Cleaning and normalizing the extracted frames:

First, the target size of the extracted frames are changed and then the pixel values for the extracted frames are normalized. After normalizing, the frames are stored in an array. The size of the extracted frames gets changed once again once they are passed through model architecture.

Evaluating the model:

To evaluate the model efficiency, the validation data and the weights saved from the model, when it was trained, are used. Two lists are created, one to store the actual tags and the other to store the predicted tags. Then a function is used to check the accuracy of the predicted tags as compared to that of the actual tags using the lists that were created.



n=299	Predicted: Violence	Predicted: NViolence	
Actual: Violence	TP: 107	FN: 43	150
Actual: NViolence	FP: 3	TN: 146	149
	110	189	

Fig. Confusion Matrix

- Accuracy(Overall, how often is the classifier correct?):
  (TP+TN)/Total Samples = (107+146)/299 = 0.846

- Misclassification Rate (Overall, how often is it wrong?):
  (FP+FN)/Total Samples = (3+43)/299 = 0.15

- True Positive Rate (When it's actually violence, how often does it predict violence?):
  TP/actual Violence = 107/150 = 0.71

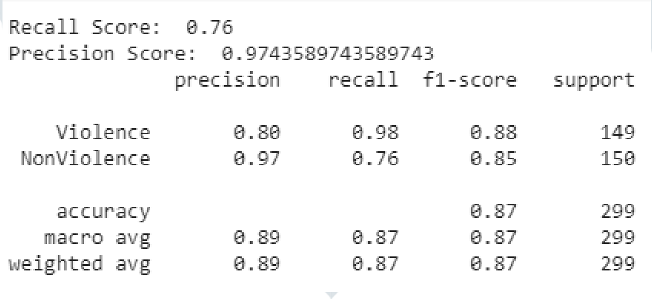
- False Positive (When it's actually NViolence, how often does it predict Violence?):
  FP/ actual NViolence = 3/149 = 0.02

- True Negative Rate (When it's actually NViolence, how often does it predict NViolence?):
  TN/ actual NViolence = 146/149 = 0.97

- Precision (When it predicts violence, how often is it correct?)
  TP/predicted Violence = 107/110 = 0.972

- Prevalence (How often does the violence condition actually occur in our sample?)
  Actual violence/total samples = 150/299= 0.501

Performance Measures Used:



```
Recall Score:  0.76
Precision Score:  0.9743589743589743
              precision    recall  f1-score   support

    Violence       0.80      0.98      0.88       149
 NonViolence       0.97      0.76      0.85       150

    accuracy                           0.87       299
   macro avg       0.89      0.87      0.87       299
weighted avg       0.89      0.87      0.87       299
```

V. CONCLUSION

The project is created as a proof of concept for a system that can classify videos as Violent or Nonviolent in real-time in surveillance footage. The inferences that we have made from implementing this system are that there is a need to develop a better process for normalizing the extracted frames for better prediction accuracy, and faster implementation. Also, the results of our experiment have shown that a simple process and a commercially accepted algorithm, which is Transfer Learning, can be used to reduce the complexity of this overall process and make the concept of real-time video classification commercially feasible for deployment at scale. The next steps in accomplishing this objective can be to port this system for running on mobile platforms, or platforms with much lower hardware configurations like a Raspberry Pi board, which can be plugged into the video feed of the surveillance system and automate the entire process.

REFERENCES

[1] Claire-Heilene, D.: VSD, a public dataset for the detection of violent scenes in movies: design, annotation, ananlysis and evaluation. In: The Handbook of Brain Theory and Neural Networks, vol. 3361 (1995)

[2] Zhan, B., Monekosso, D.N., Remagnino, P. *et al.* Crowd analysis: a survey. *Machine Vision and Applications* 19, 345–357 (2008).

[3] Wang, H., Schmid, C.: Action recognition with improved trajectories. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3551–3558 (2013)

[4] Ionescu, B., Schl¨uter, J., Mironica, I., Schedl, M.: A naive mid-level concept-based fusion approach to violence detection in hollywood movies. In: Proceedings of the 3rd ACM International Conference on Multimedia Retrieval, pp. 215–222. ACM (2013)

[5] de Souza, F.D., Ch´avez, G.C., do Valle, E.A., Araujo, A.D.: Violence detection in video using spatio-temporal features. In: 2010 23rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 224–230. IEEE (2010)

[6] Nievas, Enrique Bermejo and Suarez, Oscar Deniz and Garcia, Gloria Bueno and Sukthankar, Rahul: Hockey Fight detection Dataset (2011)

[7] Nam, J., Alghoniemy, M., Tewfik, A.: Audio-visual content-based violent scene characterization. In: Proceedings of ICIP, pp. 353–357 (1998)

[8] Cheng, W.H., Chu, W.T., Wu, J.L.: Semantic context detection based on hierarchical audio models. In: Proceedings of the ACM SIGMM workshop on Multimedia information retrieval, pp. 109–115 (2003)

[9] Gong, Y., Wang, W., Jiang, S., Huang, Q., Gao, W.: Detecting violent scenes in movies by auditory and visual cues. In: Proceedings of the 9th Pacific Rim Conference on Multimedia, pp. 317–326. Springer, Heidelberg (2008)

[10] Lin, J., Wang, W.: Weakly-supervised violence detection in movies with audio and video based co-training. In: Muneesawang, P., Wu, F., Kumazawa, I., Roeksabutr, A., Liao, M., Tang, X. (eds.) PCM 2009. LNCS, vol. 5879, pp. 930–935. Springer, Heidelberg (2009)