PREDICTING CRIMINAL ACTIVITIES ANALYZING VIDEO SIGNAL USING MACHINE LEARNING

by

Syed Md. Sadman
16101073
Tabassum Kabir
16101130
Nairita Mostafa
16101139
Ahmed Ashik Chowdhury
16301128

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
April 2020

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.

2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Syed Md. Sadman

Syed Md. Sadman
16101073

Tabassum Kabir

Tabassum Kabir
16101130

Nairita Mostafa

Nairita Mostafa
16101139

Ashik

Ahmed Ashik Chowdhury
16301128

Approval

The thesis/project titled " CRIMINAL ACTIVITIES ANALYZING VIDEO SIGNAL USING MACHINE LEARNING" submitted by

1. Syed Md. Sadman (16101073)
2. Tabassum Kabir (16101130)
3. Nairita Mostafa (16101139)
4. Ahmed Ashik Chowdhury (16301128)

of Spring, 2020 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on April 7,2020.

Examining Committee:

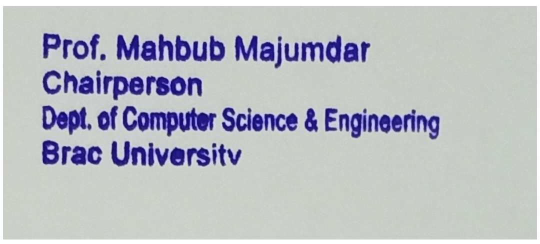
Supervisor:
(Member)

Dr. Jia Uddin
Assistant Professor (Research Track)
Department of Technology Studies, Endicott College
Woosong University, Daejeon, South Korea
Associate Professor (On leave)
Department of Computer Science and Engineering
Brac University

Thesis Coordinator:
(Member)

Dr. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)



Mahbubul Alam Majumdar
PhD Professor and Chairperson
Department of Computer Science and Engineering
Brac University

Abstract

Criminology is a method that is used to perceive wrongdoing and criminal qualities. The crooks and the wrongdoing occasion likelihood can be overviewed with the help of criminology frameworks. Video analysis and machine learning tasks have been moving from inferring the present state to predicting the future state. Law enforcement agencies can work effectively and respond faster if they have better knowledge about crime patterns in different geological points of a city. In this thesis, we proposed a system to predict criminal activities by using different neural networks and machine learning algorithms and approaches. The target of this proposed model is to break down dataset which comprise of various violations and anticipating the kind of crimes which may occur in future relying on different conditions. Contrasted with other existing models, we utilized another neural systems calculation called fastGRNN which is quicker and powerful. The experimentation is conducted on various datasets. Binary classifier, CNN, GRNN, Decision Tree, Support Vector Machine were used during experimentation. By implementing these algorithms, we came down to an accuracy of 89%.

Keywords: Crime; Video analysis Algorithms; Machine Learning; Neural Networks

Acknowledgement

First of All, we would like to thank Almighty Allah for everything. A major thanks goes to our honorable Supervisor Dr. Jia Uddin for his advice, support, feedback throughout the process of our work. He helped us every time when we needed. His support and advices helped us to reach our destination. Besides, we would like to thank our parents for their contribution in our life. Their prayer is our main weapon to do success in every sphere of life. Lastly, our friends, faculty member who also helped us lot, their advice, support was extraordinary throughout our journey.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Criminal detection may be a multi-faceted problem-solving challenge. surveillance, human-computer interaction and elderly protection devices, which has been and still remains a challenging problem on real-world conditions like partial occlusion. For criminal activities no society secure for human life. Suicide attempts, rape, terrorist attacks etc. become a thread for society development. It also destroys economic development. As an example, A website "dhakatribune.com" publish a report "A shocking 731 rapes reported in first six months of 2019" published on (July 8th, 2019) [9] that total 731 incidents of sexual violence, raped incidents were 592, gang-rape were 113 and 26 were killed after being raped. That is why there will always be an enormous demand of recent security cameras with advanced security technology to compete against the ever-evolving ways of crime that folks come up with. That is what triggered our thought regarding functioning on a project that helps alarming people before any unexpected situation takes place. It is beat attempts to make a safer environment, lowering crime, and most significantly keeping your valuable life, business and assets protected. We have got focused on certain aspects of the criminal zone analysis and have found certain issues. We believe that by fixing those issues and providing the analysts with a more analytical interface, criminal prediction project's support for deciding will be enhanced to an out sized extent. Our project is to predict criminal activity from video signals. The objective is to specialize in video footage around a selected area where criminal activities are high and so identifying suspicious activity by human. supported doubt, we are going to analyze the video footage still as pictures to predict the person may be a criminal or not.

1.2 Related Work

Recently, U-city control centers [6] have been built to observe and respond to criminal events through cooperation with the police force in real time. These centers use CCTV to collect behavior data and determine whether the situation is dangerous through crime prediction. However, the prediction accuracy is not high and detailed, and requires numerous human resources to determine the situation elaborately. Several works on this field were also published under the ADVISOR project [2]. The aim of this project was to detect vandalism acts, crowding situations and street fights. To accomplish this task, it was necessary to build the entire three-dimensional model of the monitored area. In the ADVISOR system, danger behaviors were previously defined by security experts who described relevant events using a description language. A more recent work carried out by Mecocci [1] introduces an architecture of an automatic real-time video surveillance system, capable of autonomously detecting anomalous behavioral events. The proposed system automatically adapts to different scenarios without human intervention, and applies self-learning techniques to automatically learn typical behavior of targets in each specific environment. Anomalous behaviors are detected whenever the observed trajectories deviate from the typical learned prototypes.

1.3 Objective

Criminal activities are now the biggest concern of the government as it is growing rapidly each and every year. It is affecting the quality of life and the development of a country. Crime prevention through prediction and rapid response to criminal events are becoming increasingly important social issues. Previously many researches have been done on this topic. In order to achieve more accuracy and efficiency we have tried to apply different algorithms. In this report, we have collected video datasets which is open data from UCF Crime .We have used binary classification to classify the elements of the datasets and neural networks algorithm like CNN to differentiate images from one another and reduce them into a form which are easier to process and faster GRNN to make the prediction faster. Results of different algorithms have been compared and most of the effective approach has also been documented.

1.4 Thesis Orientation

- Chapter 1- Introduction where foundation data of our exploration, the issue on which we are looking into, objectives, extent of the undertaking are examined.
- Chapter 2- Literature Review where past and related works of the equivalent issue are talked about and checked on.
- Chapter 3- Proposed model, here we talk about our proposed model methodology to the exploration.
- Chapter 4- Dataset, where we explain our dataset in details.
- Chapter 5- Algorithms where we talk about the pre-owned innovations and work process of our work in subtleties.
- Chapter 6- Includes Experimental Result Analysis, where we envision our information and investigate the outcome acquired from our model.
- Chapter 7- In Conclusion and Future works where the end comprises of our work till now and future works incorporates the extent of progress.

Chapter 2

Literature Review

Xhen [5] In this paper the performance for act recognition system using CNN is significantly successful in three public datasets with great robustness and high accuracy of partial video. In here they present a unique activity auto-completion (AAC) model for activity prediction. Two datasets are created for high-level activity analysis, and both of them comprises six differing types of human interaction activities shake-hands, hug, kick, point, punch and push. during this paper they discuss about Activity representation, Approach, Refining Detectors, Video Segmentation, Joint Feature Mapping, histogram intersection, Activity prediction, Activity Representation. They show a way to capture the essence of partial videos by mining discriminative patches. within the paper, a partial video and a full video are considered as a prefix and a question, respectively. They first discussed a way to represent videos by mining discriminative mid-level patches then video segmentation supported clustering then the details of the proposed AAC model. They divide a full video into a set of segments, which they call activity characters. This operation also reduces the quantity of prefixes. They formulate activity prediction problem as an auto-completion problem. For Represent an activity, build a compact video representation supported discriminative patches. For Experiments test the activity auto-completion AAC model on UT-Interaction and UT-Interaction get two datasets are created for high-level activity. They choose Rank SVM as learning model. Particularly, just one ranking model is formed for all activity classes. AAC model can automatically complete any new activity prefix AAC ranker, get a ranking of those activity candidates in line with relevance outputs. They also discuss to keep up the balance of detectors refine representative SVM detectors for every class supported patch temporal distribution. But sampling is not applied in other sections of this paper. According to [3] they have the problems in two subproblems. One is to detect the human in the video and another one is to track them in those video frames. They have used machine learning approach for the object detection problem. they have used Haar-like features for object detection. For the second phase, the tracking phase they have used particle filter system. In the proposed approach, uncertainty about human's state (position) is represented as a set of weighted particles, each particle representing one possible condition. Tracking of a human object is done from one video frame to another by foreseeing its position within the following outline when the position of the frame is known within the past outline. they have experimented their human detection and tracking technique on a number of videos. As in a video human can be found in different poses. So, they have divided the videos in 4 categories. Fist one is Single Human Videos, second one is Multiple Human Videos with Occlusion, third one is Multiple Human Videos Without Occlusion, last one is Multiple Objects Videos. Their algorithm can detect and track the all different poses of a man in those videos. Their average detection and tracking accuracy are 87.44% and the average execution speed is 18 frames per second.

The paper titled "FastGRNN: A Fast, Accurate, Stable and Tiny Kilobyte Sized Gated Recurrent Neural Network" by Kusupati, Aditya Singh stated that, Productive expectation calculations have regularly been acquired by making sparsity and low-position presumptions. Most unitary strategies viably use a low-position portrayal of the state transition matrix to control prediction and training complexity. Sparsity, low-position, and quantization were demonstrated to be successful in RNNs, CNNs, trees and closest neighbor classifiers. FastGRNN expands on these plans to use low-position, scanty and quantized portrayals for learning kilobyte measured classifiers without settling on classification accuracy. Different ways to deal with accelerate RNN preparing and forecast depend on supplanting consecutive concealed state changes by parallelizable convolutions or on learning skip associations in order to abstain from assessing all the shrouded states. Such strategies are correlative to the ones proposed right now can be utilized to additionally improve FastGRNN's presentation.

Chapter 3

Proposed Model

Since the world's trouble makers are expanding altogether, we have to step up to stop the events or possibly attempt to limit where conceivable. In spite of the fact that the law authorization association is working heart and soul, the crime percentages going up step by step. In this paper, we have collected video datasets which are open data from UCF Crime. The purpose of our model is to predict criminal activities in a particular area. We used the train_test_split function for our project. The train_test_split function is for splitting a single dataset for two different purposes: train and testing. We are going to train 80 percent of the data and we will test over 20percent of the data to get the higher accuracy in final results. We have used mainly three algorithms for our project. We have used binary classifiers, fast GRNN and CNN. Finally, for better results we found First GRNN is the best suited algorithm.

The proposed approach (in Fig. 3.1) starts with dividing surveillance videos into a fixed number of fragments during training. These portions make examples in a pack. Utilizing both positive (anomalous) and negative (normal) values, we train the abnormality recognition model utilizing the proposed Machine Learning algorithms for best result.

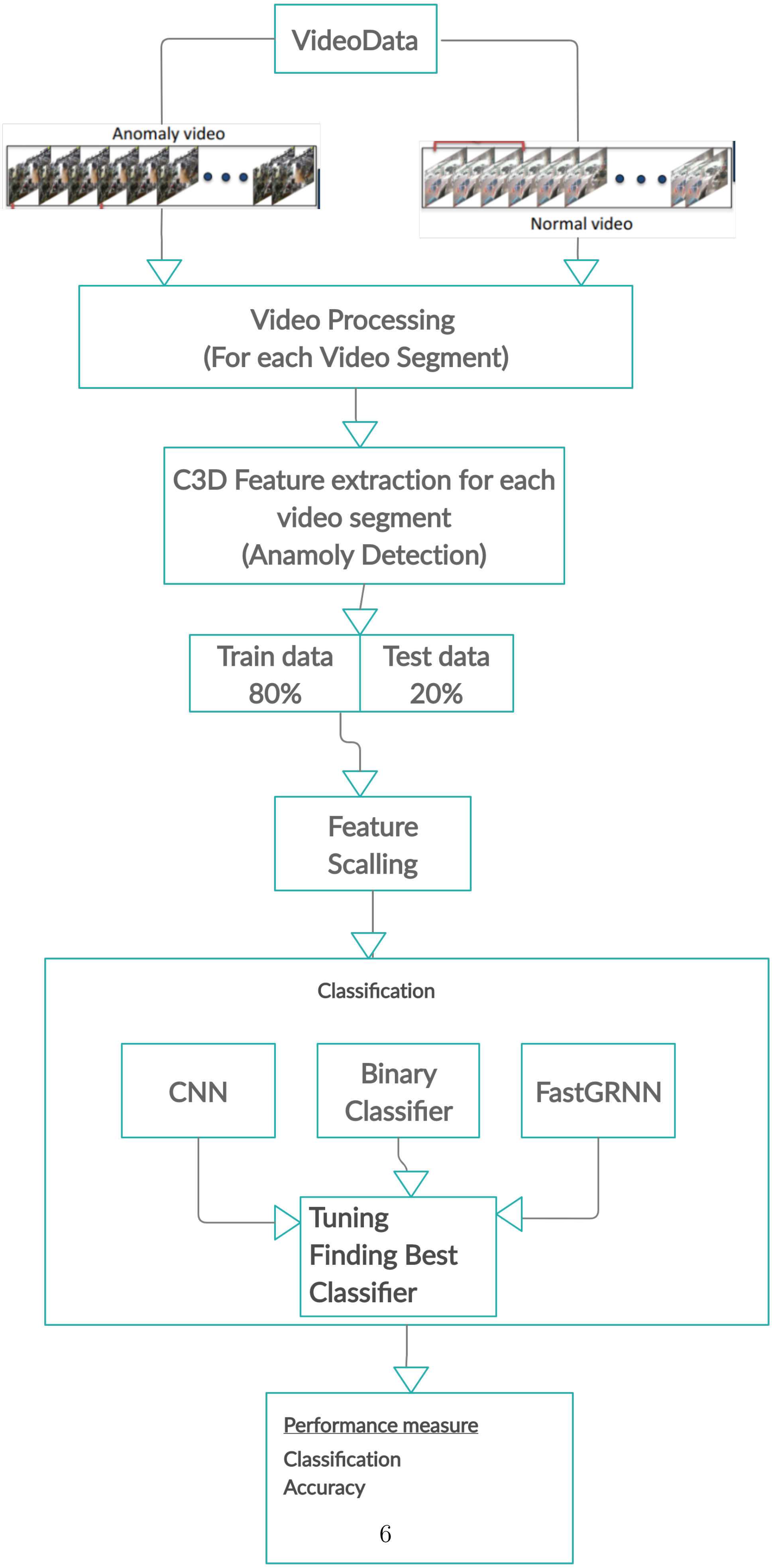


Figure 3.1: Proposed Model

Chapter 4

Dataset

Our experiment is conducted on a specific dataset. It is open data from UCF-Crime Dataset (Real-world Anomalies Detection in Videos) [10]. UCF-Crime dataset is another enormous scope first of its sort dataset of 128 hours of recordings. It comprises of 1900 long and untrimmed certifiable reconnaissance recordings, with 13 reasonable oddities including Abuse, Arrest, Arson, Assault, Road Accident, Burglary, Explosion, Fighting, Robbery, Shooting, Stealing, Shoplifting, and Vandalism. These peculiarities are chosen since they significantly affect open security. This dataset can be utilized for two assignments. To begin with, general abnormality discovery thinking about all peculiarities in a single gathering and every ordinary movement in another gathering. Second, for perceiving every one of 13 odd exercises. It provides many crime incidents videos that occurred around the world for the period of 1970 to 2017.

4.1 Attributes

The data in the datasets contains mp4 file. Based on the mp4 data there is a csv file containing 461 rows where the types of crime video (arrest, arson, assault, road accident, fighting etc.), normal or abnormal video, resolution are included based on whether these are anomaly or not. The attributes are given below.

Date	A Date when the crime occurred
Category	Types of crimes
Crime Description	Detailed description for a crime
iday	Day of week when crime occurred
imonth	Crimes on Month
Location of Attack	The location of crime
Terrorist attack	Whether it was terrorist attack or not
Successful/Unsuccessful	Whether it was successful or not
Was/Was not Suicide attack	Whether it was suicidal or not
Human freedom	Human freedom score (4-9)
Personal freedom	Personal freedom score (2-9)

Table 4.1: Attributes of the crime dataset

4.2 Dataset Description

UCF_Crimes folder contains following 3 subfolders:

Anomaly_Detection_splits
This folder contains training and testing partitions for anomaly detection experiments, i.e., Anomaly_Test.txt and Anomaly_Train.txt.

Action_Recognition_splits
This folder contains four training and testing partitions for action recognition experiments. The experimental results reported in the paper are done using four-fold cross- validation. In order to report the results, please use all four partitions.

Videos
This folder contains the complete dataset. It has 16 subfolders. The names of folders are self-explanatory.

- 13 folders correspond to each of the anomaly.
- Normal_Videos_event corresponds to normal videos for action detection experiments.
- Testing_Normal_Videos_Anomaly contains normal testing videos for anomaly detection experiment.
- Training_Normal_Videos_Anomaly contains normal training videos for training a network for anomaly detection experiments.

4.3 Crime Classification

There are 13 types of crime in UCF-Crime dataset. The data has crime record from 1970-2017. The frequency changes in every 10year interval. The frequency changes mostly, based on location. The frequency of crime varies one country to another. Sometimes the frequencies are high and sometimes low, based on the situation of that time. These anomalies are selected because they have a significant impact on public safety. A short depiction of each anomalous occasion is given underneath.
Abuse: This occasion contains recordings which show awful, brutal or savage conduct against youngsters, elderly individuals, creatures, and ladies. **Burglary:** This occasion contains recordings that show individuals going into a structure or house with the expectation to submit robbery. It does exclude utilization of power against individuals. **Robbery:** This occasion contains recordings demonstrating thieves taking cash unlawfully forcibly or risk of power. These recordings do exclude shootings. **Stealing:** This occasion contains recordings demonstrating individuals taking property or cash without consent. They do exclude shoplifting. **Shooting:** This occasion contains recordings indicating demonstration of firing somebody with a firearm. **Shoplifting:** This occasion contains recordings demonstrating individuals taking products from a shop while acting like a customer. **Assault:** This occasion contains recordings indicating an abrupt or rough physical assault on somebody.

Note that in these recordings the individual who is attacked doesn't retaliate. **Fighting:** This occasion contains recordings showing two are more individuals assaulting each other. **Arson:** This occasion contains recordings demonstrating individuals intentionally burning down property. **Explosion:** This occasion contains recordings indicating dangerous occasion of something blowing separated. This occasion does exclude recordings where an individual deliberately sets a fire or sets off a blast. **Arrest:** This occasion contains recordings indicating police capturing people. **Road Accident:** This occasion contains recordings demonstrating car crashes including vehicles, walkers or cyclists. **Vandalism:** This occasion contains recordings indicating activity including intentional demolition of or harm to open or private property. The term incorporates property harm, for example, spray painting and disfigurement coordinated towards any property without authorization of the proprietor. **Normal Event:** This occasion contains recordings where no wrongdoing happened. These recordings incorporate both indoor, (for example, a shopping center) and open-air scenes just as day and evening time scenes.

4.4 Features of Dataset

Let us picture the information with graphical representations.

4.4.1 Crime Map

Here the black spot indicates the crime zone. It indicates that crime frequencies were not same in every 10years.

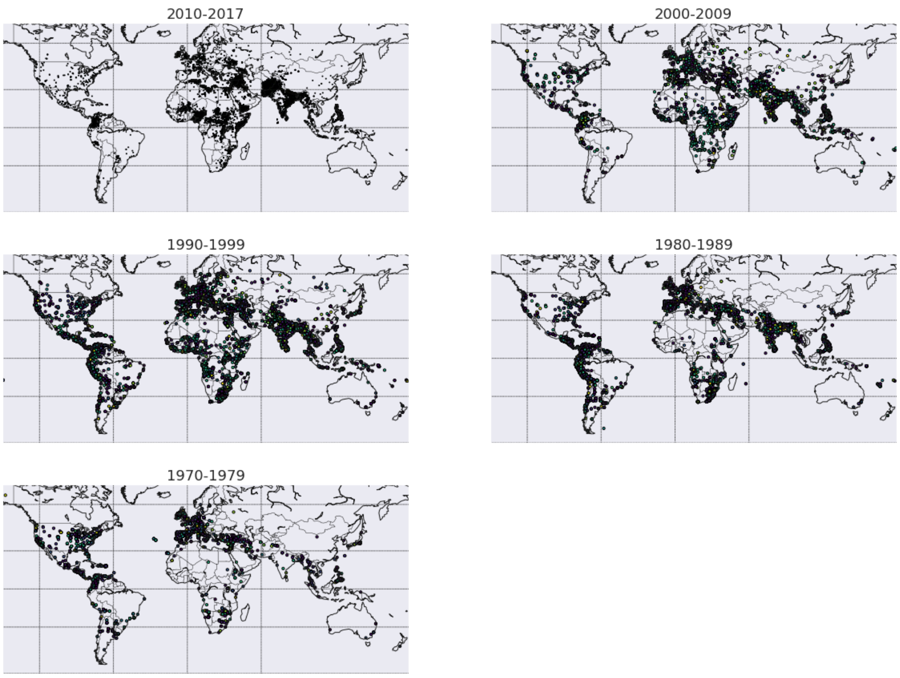


Figure 4.1: The crime map of last 47 year

We can see from 4.1 that how the rate of crime is changing in every 10 years. The rate of crime was not stable. It depends on situation, environment, socio-economic structure etc. From that maps we understand the frequency of crimes changes by the time.

4.4.2 Area of the crimes

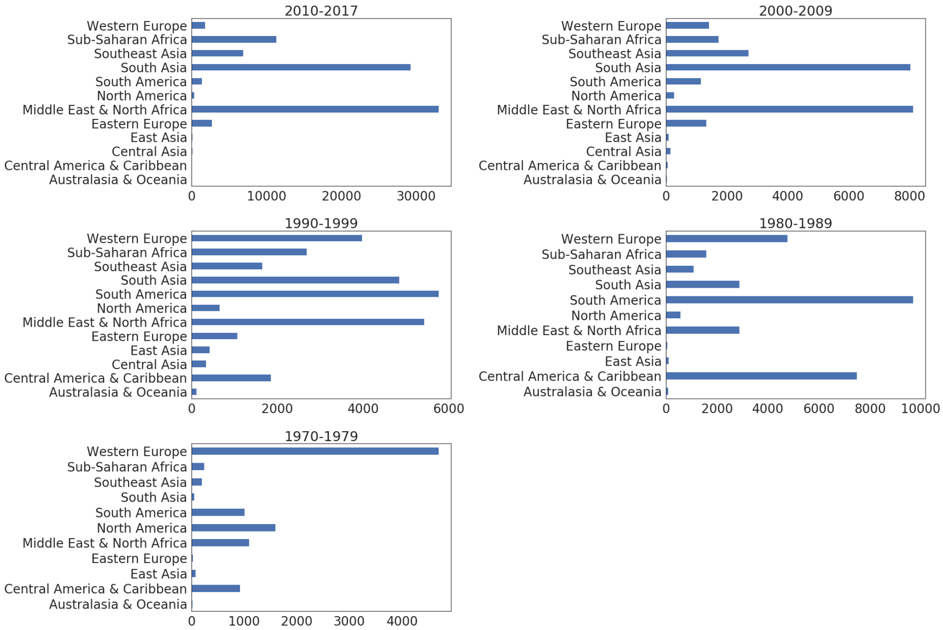


Figure 4.2: Crime area with frequency from 1970 to 2017

4.4.3 Types of the crimes

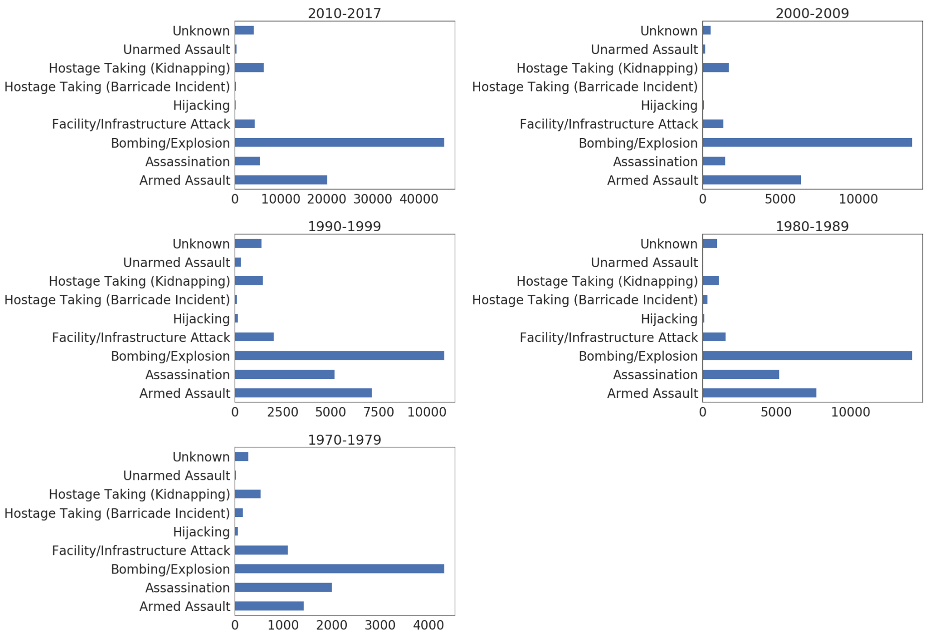


Figure 4.3: Crime Frequency

4.4.4 Targeted Attack

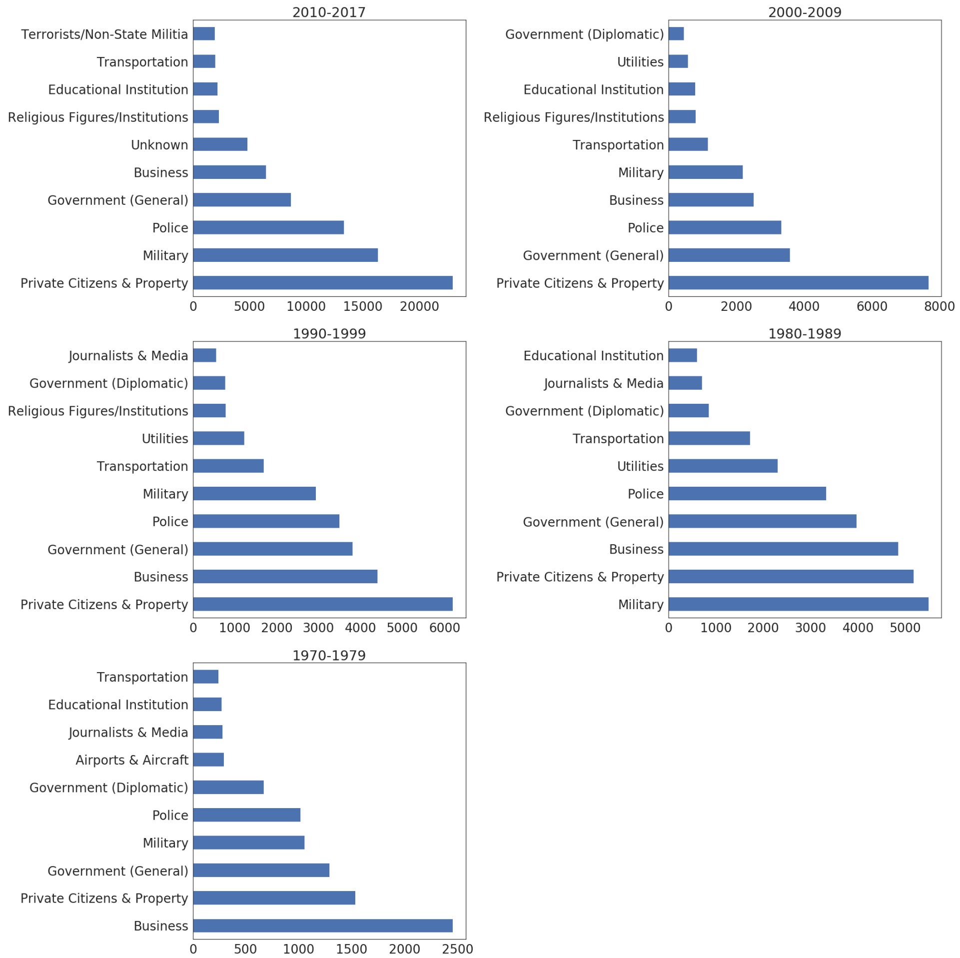


Figure 4.4: Targeted attack

4.4.5 Terrorist Attack Map

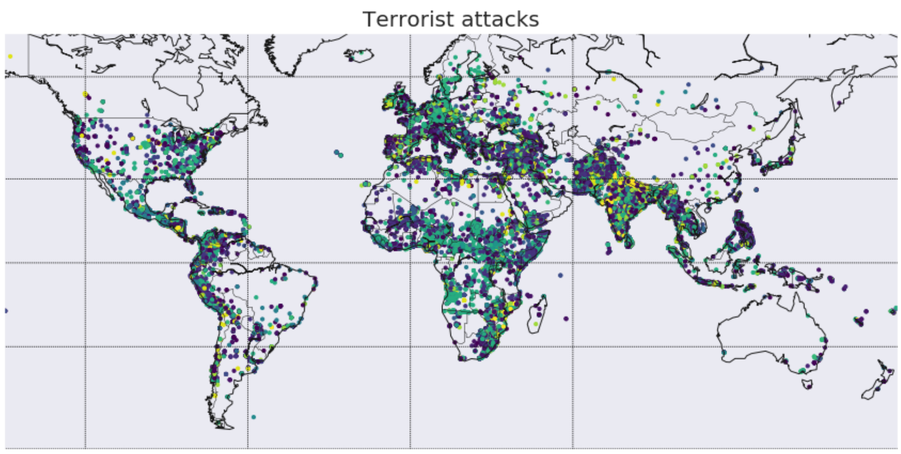


Figure 4.5: Terrorist attack map

4.4.5.1 Terrorist Attack Count (Attempted)

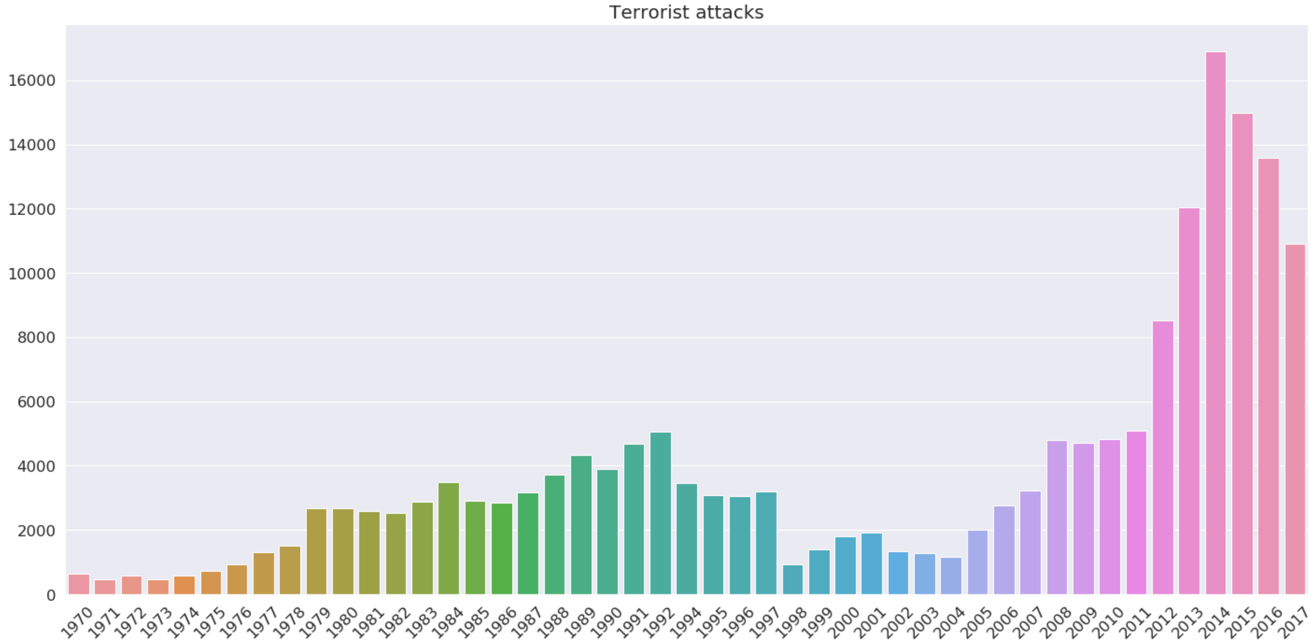


Figure 4.6: Number of terrorist attack 1970 to 2017

4.4.5.2 Terrorist Attack in Days (Attempted)

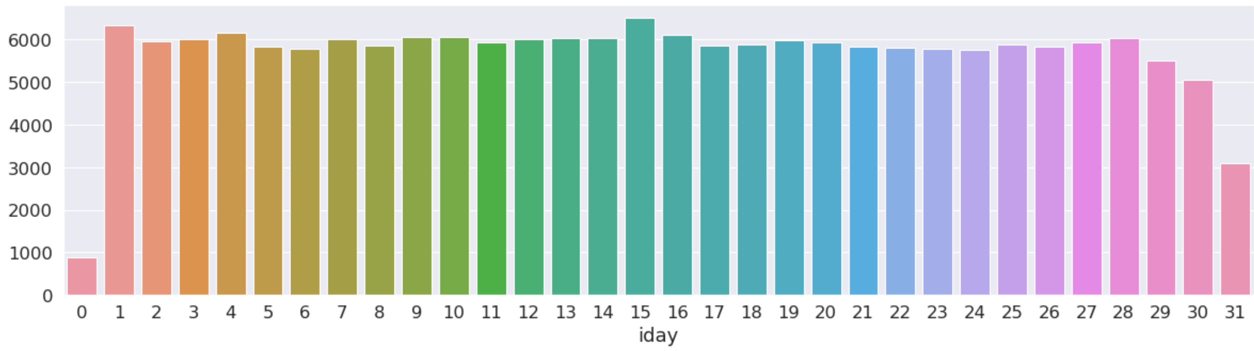


Figure 4.7: Frequency of attack per day

4.4.5.3 Terrorist Attack in Month (Attempted)

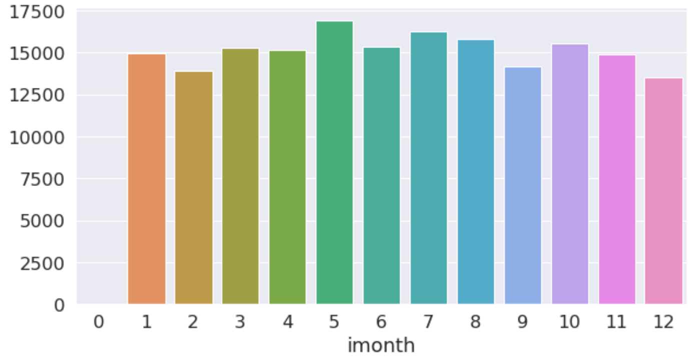


Figure 4.8: Attacks in Month

4.4.6 Successful/Unsuccessful Terrorist Strike Map

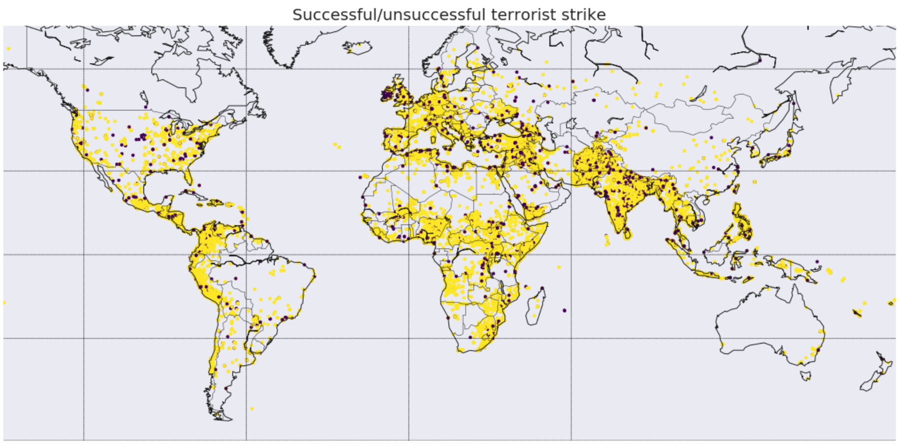


Figure 4.9: Successful/Unsuccessful map

4.4.6.1 Successful/Unsuccessful Terrorist Strike Count

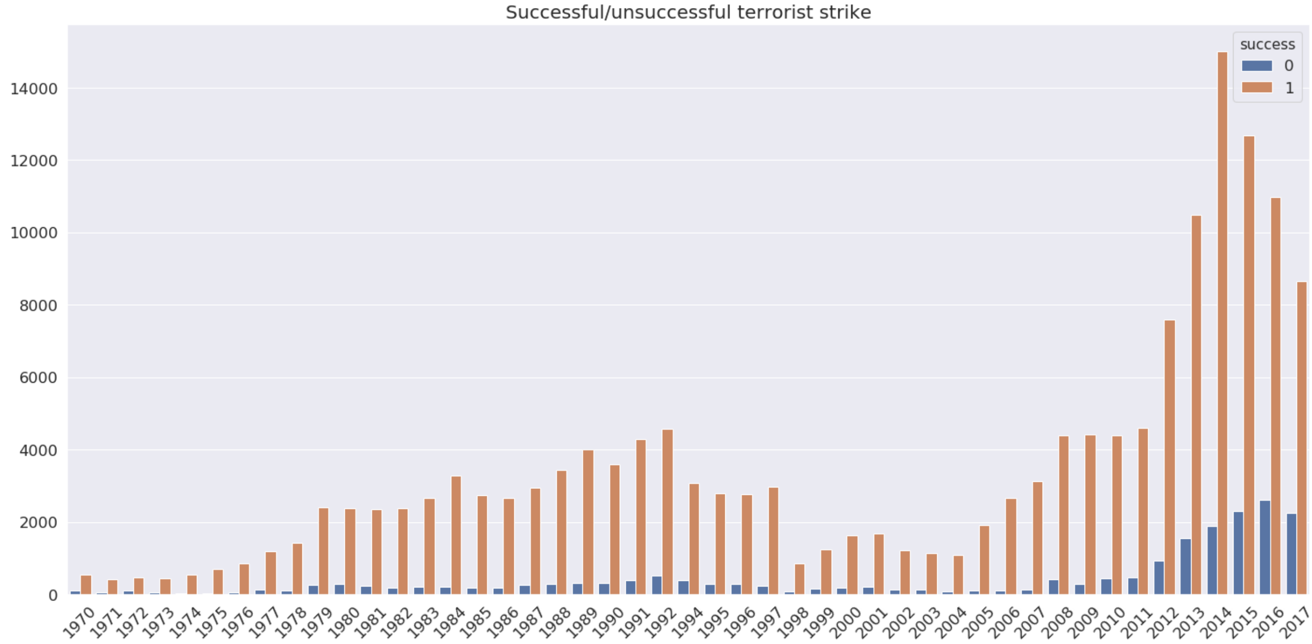


Figure 4.10: Successful/Unsuccessful count (0 refers to unsuccessful, 1 successful)

4.4.7 Incident was/was not a Suicidal Attack

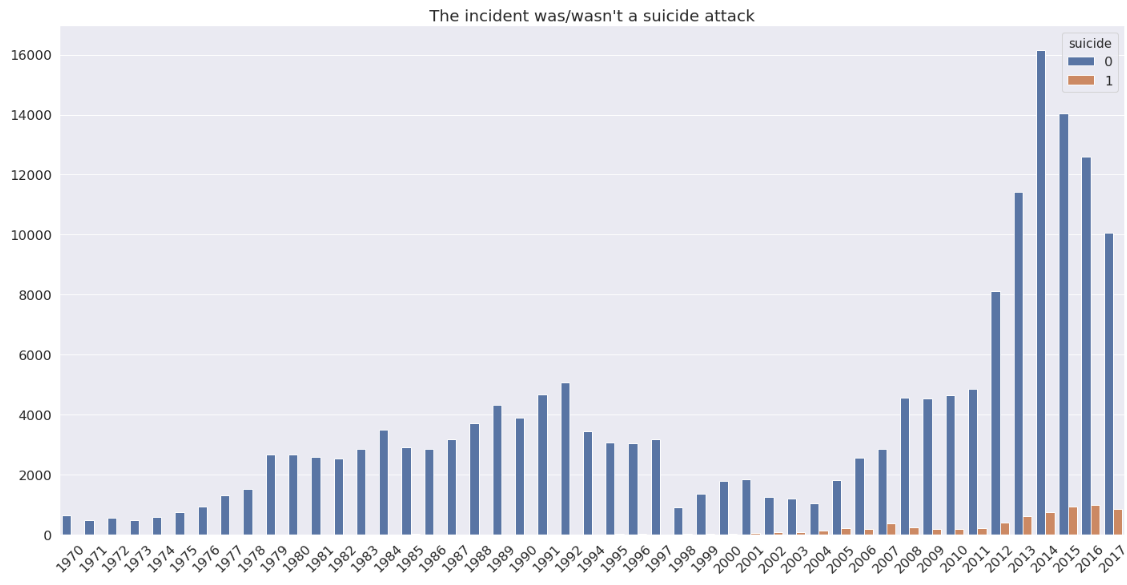


Figure 4.11: Suicidal attack (0 not suicidal, 1 suicidal)

4.4.8 Human Freedom (Score)

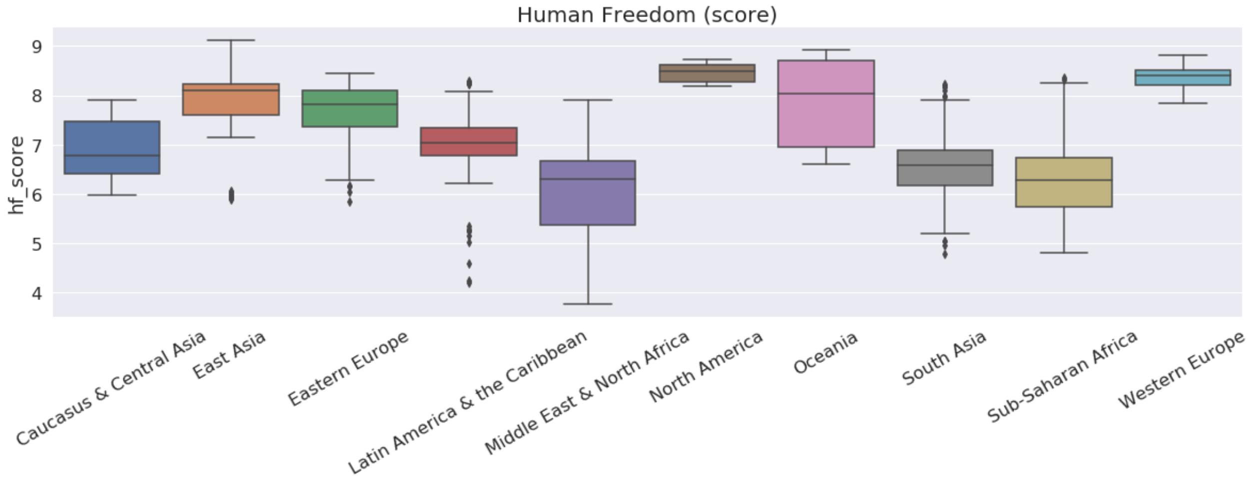


Figure 4.12: Human freedom based on region

4.4.9 Personal Freedom (Score)

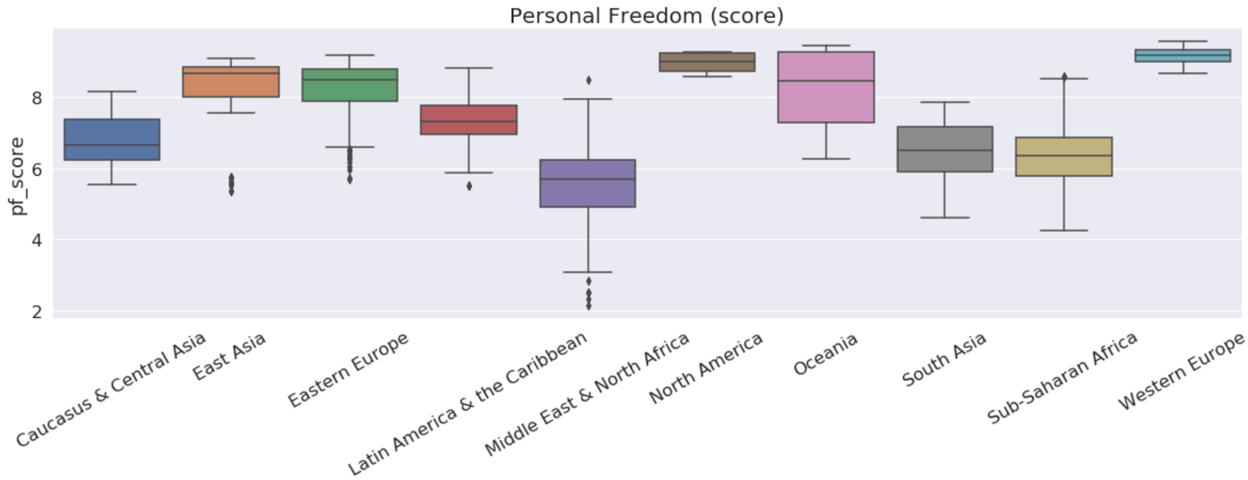


Figure 4.13: Personal freedom based on region

We can see from Fig. 4.5 that; it is a map of all terrorist attack where we can understand the terrorist attack frequency given in Fig. 4.6 from 1970 t0 2017. Besides, In Fig. 4.7 we can also see the amount of attacks per day and in Fig. 4.8 the attacks in months. These all attacks are attempted, from these terrorist attack some are successful and some unsuccessful, and we represented it in Fig. 4.10. In Fig. 4.11 we can understand that, which attacks were suicidal and which were not. Based on these from Fig. 4.12 and Fig. 4.13 we established Human Score and personal freedom score depending on location.

Chapter 5

Algorithms

5.1 Neural Networks

Neural networks are a method for doing AI, where a PC figures out how to play out some undertaking by investigating preparing models. As a rule, the models have been hand-marked ahead of time. An item acknowledgment framework, for example, may be taken care of thousands of marked pictures of vehicles, houses, espresso cups, etc., and it would discover visual examples in the pictures that reliably associate with specific names. There are so many neural networks exists. In this paper, we particularly used the CNN, GRNN to predict the crime activities.

5.1.1 CNN

A Convolutional Neural System (ConvNet/CNN) is a Profound Learning calculation which can take in an information picture, dole out significance (learnable loads and predispositions) to different angles/protests in the picture and have the option to separate one from the other. The pre-preparing required in a ConvNet is a lot of lower when contrasted with other arrangement calculations. While in crude strategies channels are hand-designed, with enough preparing, ConvNets can gain proficiency with these channels/attributes.

The engineering of a ConvNet is practically equivalent to that of the availability example of Neurons in the Human Cerebrum and was enlivened by the association of the Visual Cortex. Singular neurons react to boosts just in a confined locale of the visual field known as the Open Field. An assortment of such fields covers to cover the whole visual territory.

A ConvNet can effectively catch the Spatial and Fleeting conditions in a picture through the utilization of important channels. The engineering plays out a superior fitting to the picture dataset because of the decrease in the quantity of parameters included and reusability of loads. As such, the system can be prepared to comprehend the modernity of the picture better.

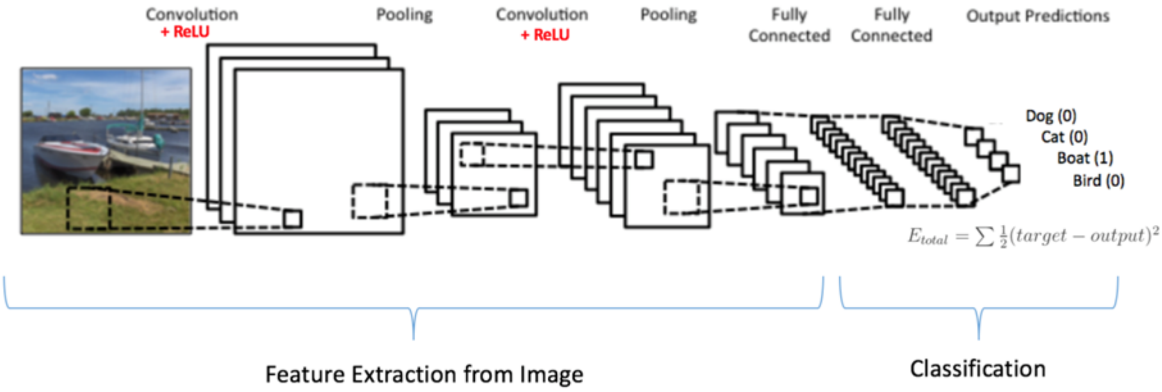


Figure 5.1: Working Method Of CNN

5.1.2 FastGRNN

Past methodologies have improved precision to the detriment of forecast costs making them infeasible for asset compelled and ongoing applications. Unitary RNNs have expanded exactness to some degree by limiting the scope of the state change framework's solitary qualities yet have additionally expanded the model size as they require a bigger number of shrouded units to compensate for the misfortune in expressive influence. Gated RNNs have gotten cutting edge exactness's by including additional parameters in this way coming about in much bigger models.

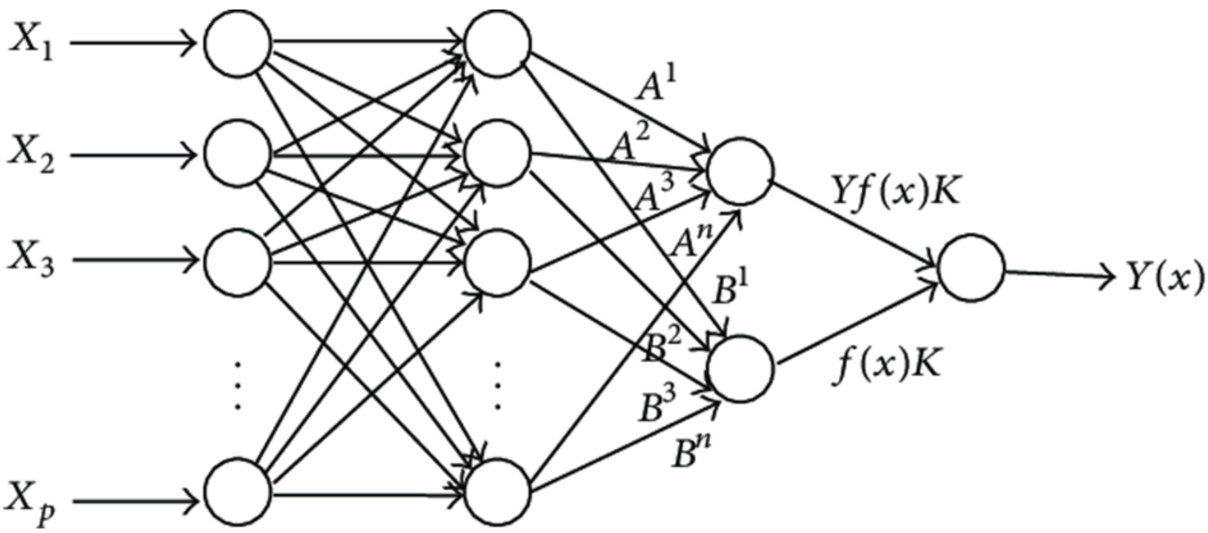


Figure 5.2: Structure of FastGRNN

5.2 Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves. [7] There are two types of machine learning algorithms: Supervised and Unsupervised. In order to get better accuracy, we used some Supervised machine learning algorithms such as Binary Classification, Decision Tree and Support Vector Machine.

5.2.1 Binary Classification

Binary or binomial classification is the task of classifying the elements of a given set into two groups (predicting which group each one belongs to) on the basis of a classification rule.
The objective of binary classification is to gain proficiency with a capacity F(x) that limits the misclassification likelihood PyF(x) <0, where y is the class name with + 1 for positive and -1 for negative. There are numerous powerful binary classifications strategies, for example, Decision tree, Support Vector Machine (SVM), Logistic Regression, Confusion Matrix etc. Decision tree and SVM were used in this case.

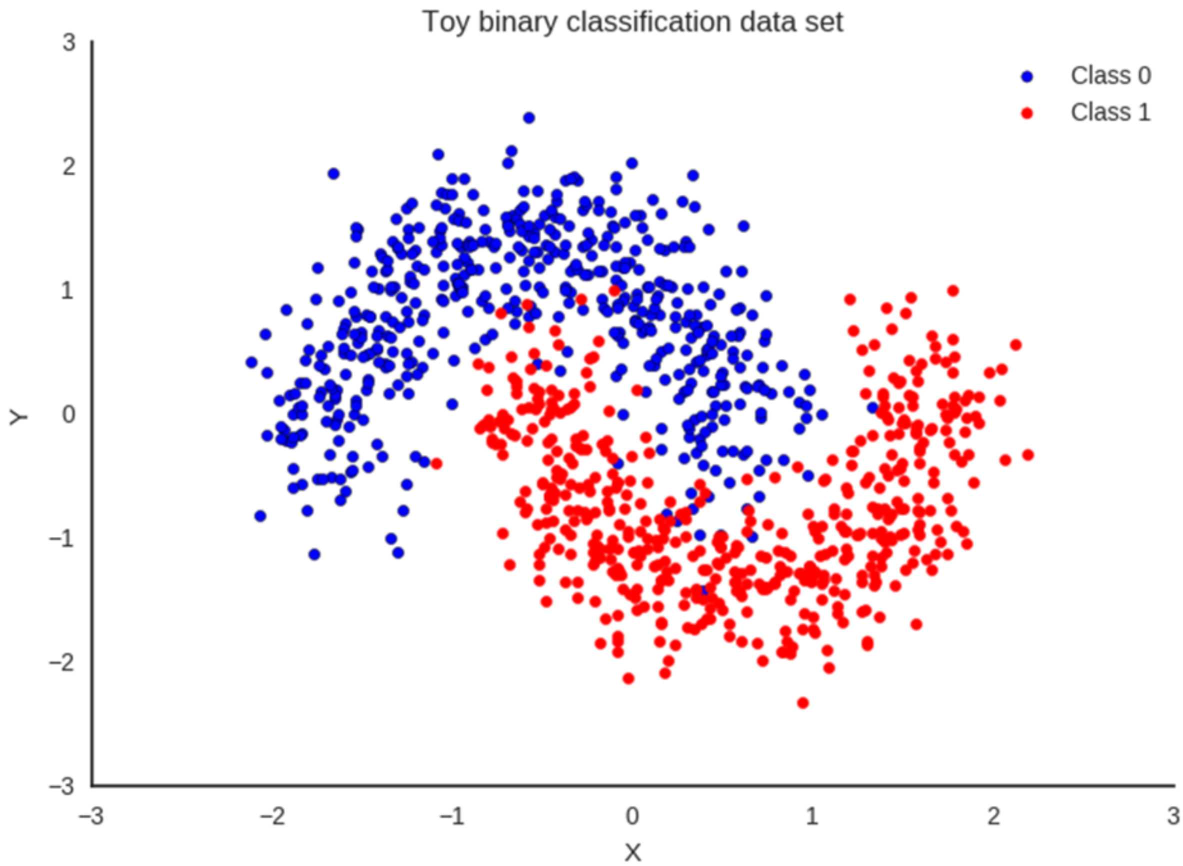


Figure 5.3: Binary Classifier

5.2.2 Decision Tree

Decision tree learning is a managed AI strategy for actuating a decision tree from training data. We used decision tree to test our datasets. A decision tree (also referred to as a classification tree or a reduction tree) is a predictive model which is a mapping from observations about an item to conclusions about its target value. In the tree structures, leaves represent classifications (also referred to as labels), non-leaf nodes are features, and branches represent conjunctions of features that lead to the classifications [5]. Building a decision tree that is consistent with a given data set is easy. The challenge is to build good decision trees, which means the smallest decision trees. Over fitting pruning can be utilized to keep the tree from being over fitted only for the training set. This method makes the tree general for unlabeled information and can endure some erroneously marked training data.

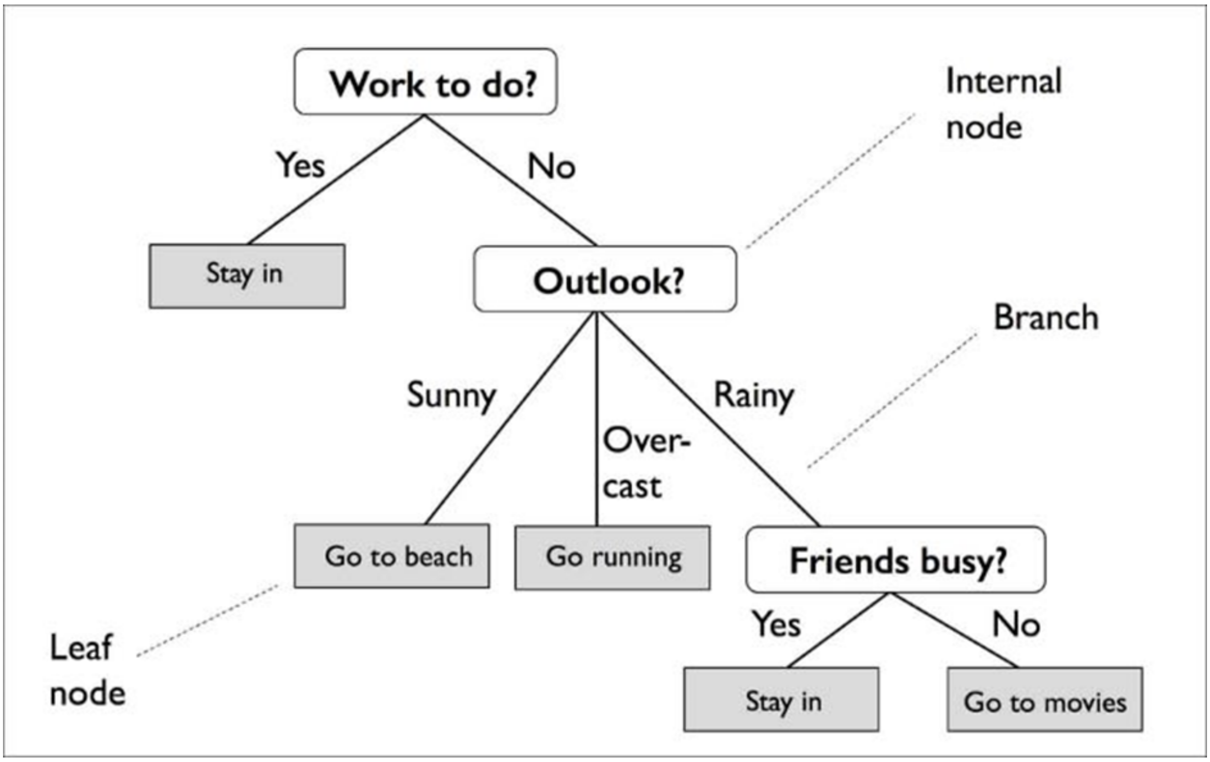


Figure 5.4: Decision Tree

5.2.3 Support Vector Machine

The target of support vector machine calculation is to discover a hyperplane in a N-dimensional space (N — the quantity of highlights) that unmistakably orders the data points. To isolate the two classes of data points, there are numerous conceivable hyper planes that could be picked. Our goal is to locate a plane that has the most extreme edge, i.e. the greatest separation between data points of the two classes. Expanding the edge separation gives some support so future information focuses can be grouped with more certainty.

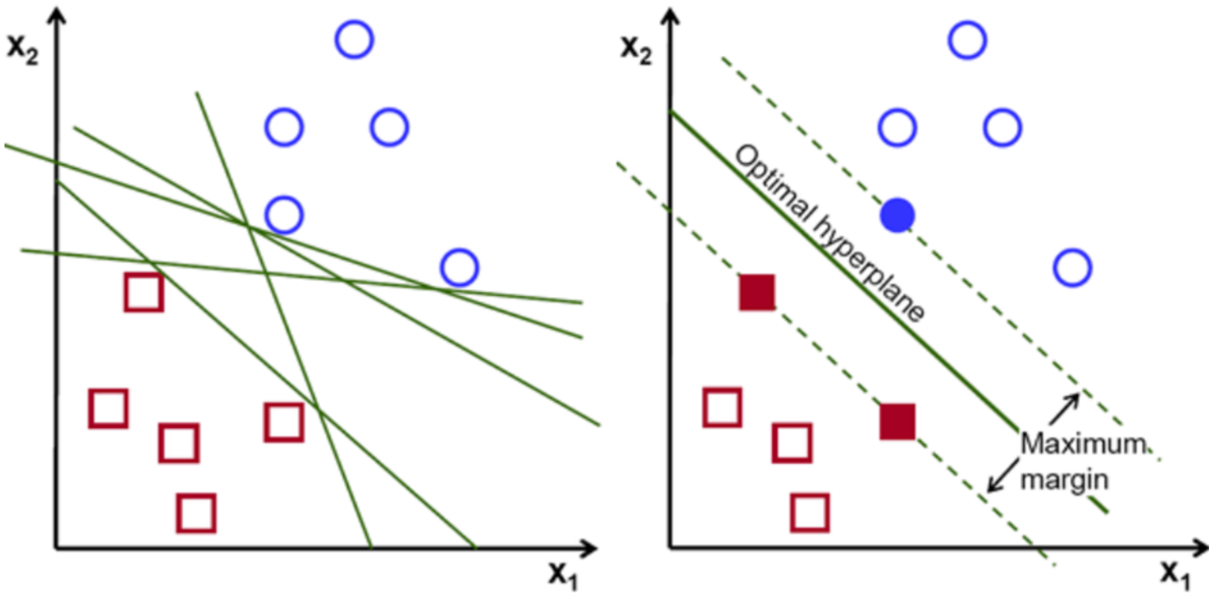


Figure 5.5: Possible Hyperplanes

Hyperplanes are choice limits that help group the data points. Data points falling on either side of the hyperplane can be ascribed to various classes. Additionally, the element of the hyperplane relies on the quantity of highlights.

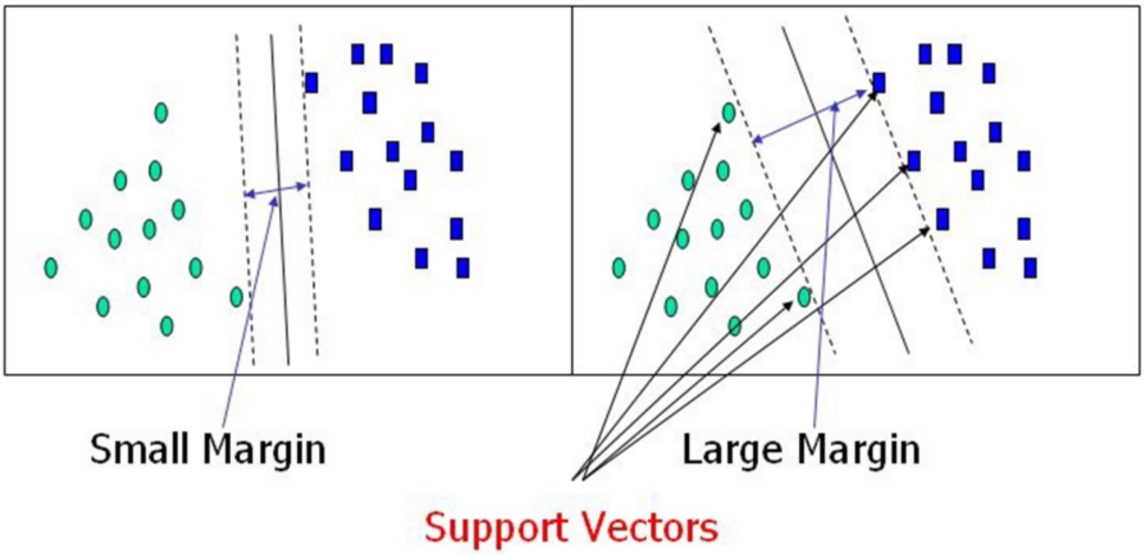


Figure 5.6: Support Vectors

Support vectors are data points that are nearer to the hyperplane and impact the position and direction of the hyperplane. Utilizing these support vectors, we amplify the edge of the classifier. Erasing the support vectors will change the situation of the hyperplane. These are the focuses that assisted us with building the SVM.

Chapter 6

Result Analysis

In order to get result, we used three Machine Learning Algorithm (CNN, Binary Classification, FastGRNN). we allotted 20% of the data to perform the testing just to decide the performance.

6.1 Implementation Details

For our anomaly discovery technique, as it were video-level marks are required for train. Be that as it may, all together to assess its exhibition on testing recordings, we need to know the transient comments, for example the beginning and completion casings of the peculiar occasion in each testing strange video. To this end, we appoint similar recordings to different annotators to mark the worldly degree of every peculiarity. The last transient explanations are acquired by averaging explanations of various annotators.
we remove visual highlights from the completely associated (FC) layer FC6 of the C3D arrange [4]. Prior to processing highlights, we re-size every video casing to 240x320 pixels and fix the edge rate to 30 fps. We process C3D highlights for each 16-outline video cut followed by l2 standardization. To acquire highlights for a video fragment, we take the normal of every one of the 16-outline cut highlights inside that portion.

6.2 Performance of the Proposed Method

We mentioned earlier that we have applied three models and now we are going to analyze the results, compare the results and find the best result for us. The proposed approach is that gives a great deal of positive and negative recordings with video-level marks, the system can naturally figure out how to anticipate the area of the anomaly in the video [8]. To accomplish this objective, the system ought to figure out how to produce high scores for atypical video portions during training iterations.

6.3 Visualization

To training dataset we used some models like bc model c3d model with an abstract.

Best Library:tensorflow, Numpy, PCA, matplotlib. pyplot as plt, seaborn, pandas

6.3.1 Crime HOF Visualization

Here We visualize all crime classes by condensing our HOF (Histograms of Optical Flow) features using PCA (Principal Component Analysis).
We go analyze the distinction between the crime and the normal part of the video
We view the distribution of video lengths for both datasets in consideration
Each row of 'Temporal_Anomaly_Annotation.csv' is the annotation for a video, for example: Abuse028_x264.mp4 Abuse 165 240 -1 -1

- The first column is the name of the video.
- The second column is the name of the anomalous event.
- The third column is the starting frame of the event (convert each video to image frames first) .
- The fourth column is the ending frame of the event.
- For videos in which second instance of event occurs, fifth and sixth contains starting and ending frames of second instance.
  Negative number means no anomalous event instance. In this example, abuse (instance) only occurs once.
  Note: The videos have 30 frames per second.
  Crime labels = ['Arrest','Burglary','Fighting','Explosion','Assualt','Normal','Abuse','Arson']
  After condensing our HOF feature using PCA we visualize all crime classes below.

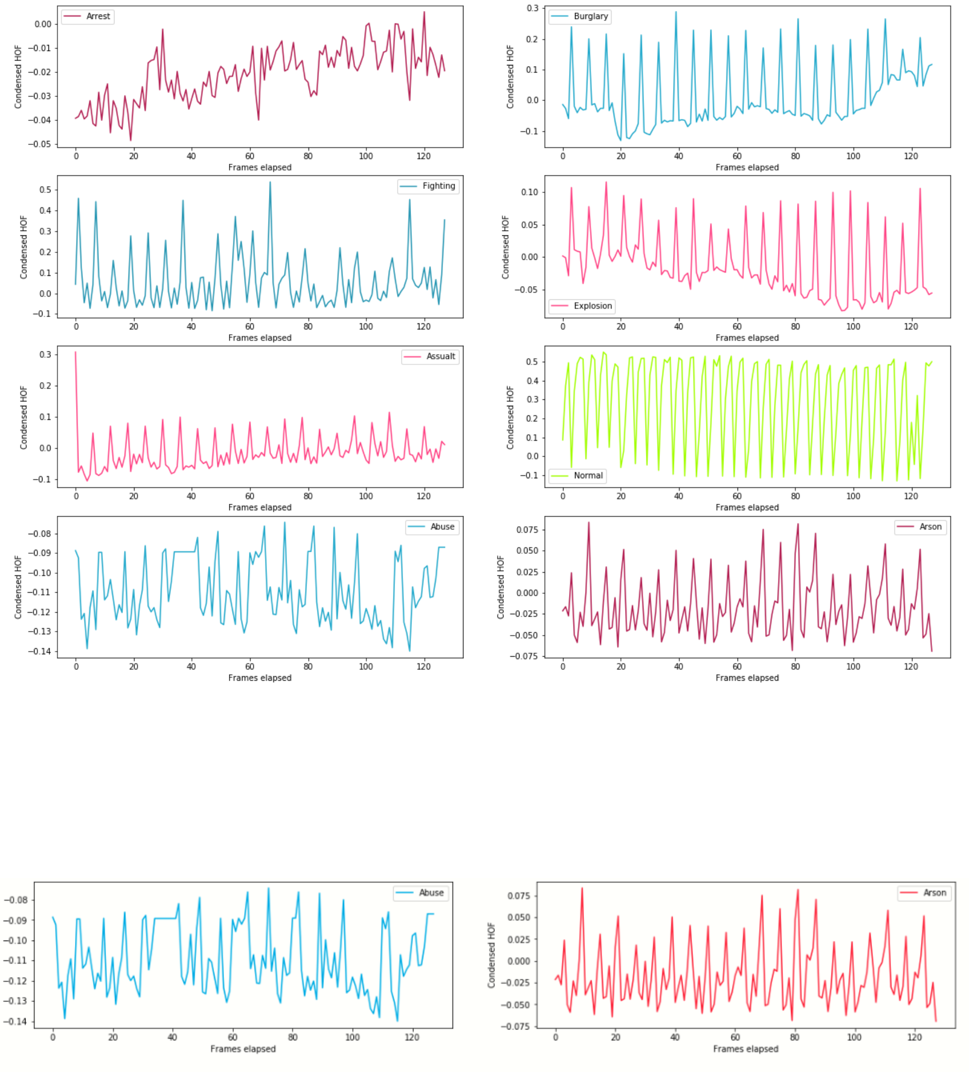


Figure 6.1: Condensed HOF vs Frame elapsed

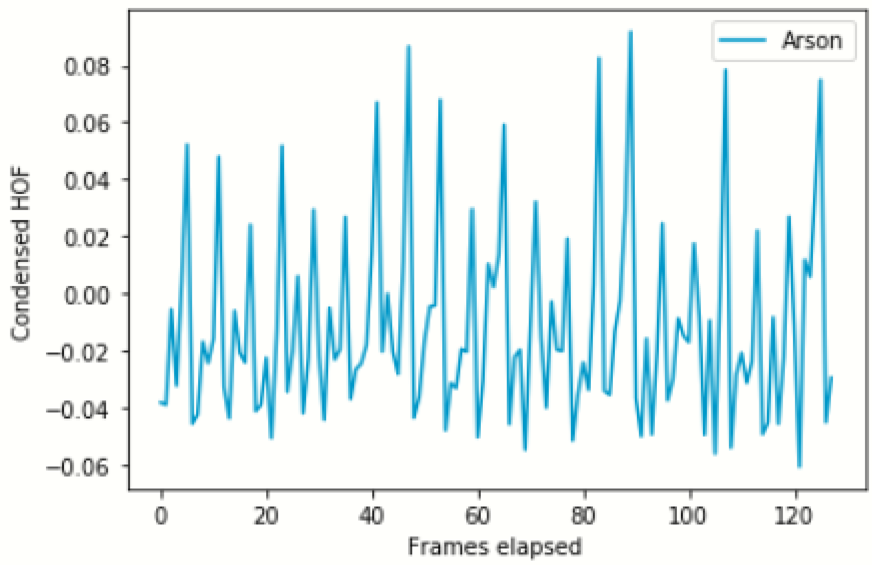
(x = 'Frames elapsed', y = 'Condensed HOF'). Frame elapse range 128. ['Frames elapsed','Condensed HOF'] for eight of them sequentially (67456, 1), (129408, 1), (27264, 1), (13824, 1), (26112, 1), (639104, 1), (29056, 1), (39680, 1).

	Frames elapsed	Condensed HOF
0	0	-0.038105
1	1	-0.038878
2	2	-0.005329
3	3	-0.032183
4	4	0.004425
...	...	...
123	123	0.005818
124	124	0.037033
125	125	0.074902
126	126	-0.045039
127	127	-0.029412

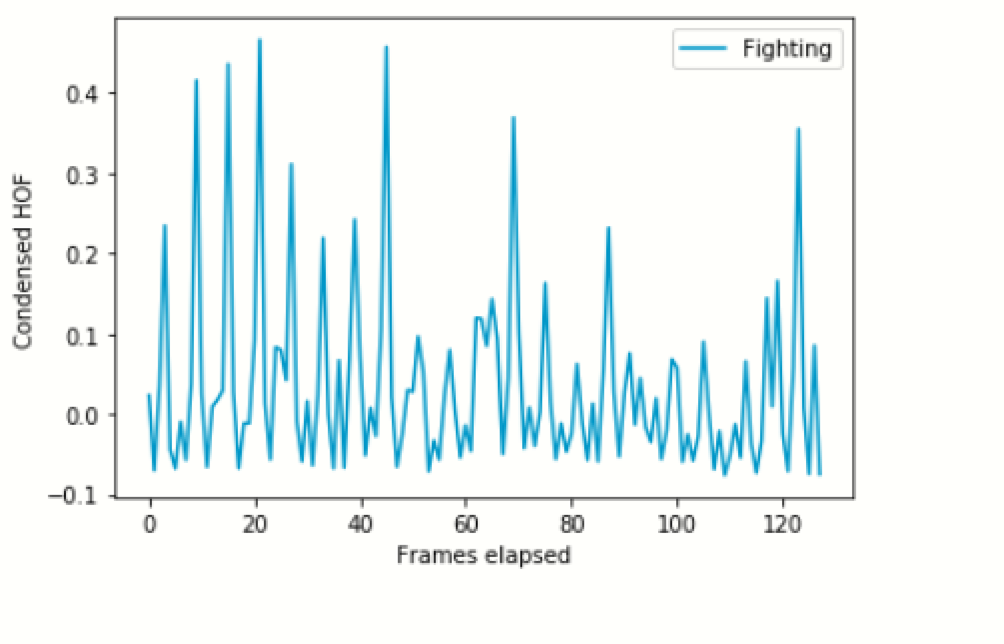
128 rows × 2 columns

Figure 6.2: Frame elapse with condensed HOF for range 0 to 127.

6.1 shows the graphical visualization of eight crime classes video based on their frame elapsed rate and condensed HOF value. The value of frame elapsed and condensed HOF varies one to another crimes. For this visualization we took frame elapsed range 128. In Fig. 6.2 shows the tabular representation of frame elapsed rate with condensed HOF within range 128.



(a) plt. Axis ([0,40, -.07,.09])



(b) plt. Axis ([0,40, -.07,.09])

Figure 6.3: Matplot for Arson Fighting

6.3.2 Get HOF for a Video with Crime visualization

We use principal component analysis to represent the HOF in an interpretable visual format with lesser dimensions.

	Abuse001_x264.mp4	**1**	**230**	**365**	**-1**	**-1.1**
0	Abuse002_x264.mp4	1	1	250	-1	-1
1	Abuse003_x264.mp4	1	833	3588	-1	-1
2	Abuse004_x264.mp4	1	870	2000	-1	-1
3	Abuse005_x264.mp4	1	692	930	-1	-1
4	Abuse006_x264.mp4	1	323	1822	-1	-1
...	...	...	...	...	...	...
944	Vandalism046_x264.mp4	14	750	1130	-1	-1
945	Vandalism047_x264.mp4	14	1040	1130	2060	2400
946	Vandalism048_x264.mp4	14	2960	5000	-1	-1
947	Vandalism049_x264.mp4	14	240	750	3570	4230
948	Vandalism050_x264.mp4	14	1	880	-1	-1

949 rows × 6 columns

Figure 6.4: Anomaly annotation

In 6.4 shows the Temporal-anomaly-Annotation.csv file of our video dataset. This is the annotation for videos. As we mentioned earlier that, the first column is the name of the video, the second column is the name of the anomalous event, the third column is the starting frame of the event, the fourth column is the ending frame of the event.

Sample videos Burglary.mp4 videos:

	feat_0	feat_1	feat_2	feat_3	feat_4	feat_5	feat_6	feat_7	feat_8	feat_9	...
0	0.000000	0.000000	0.0	0.000000	0.006425	0.0	0.0	0.0	0.0	0.009698	...
1	0.000000	0.000000	0.0	0.000000	0.003843	0.0	0.0	0.0	0.0	0.000007	...
2	0.000000	0.000000	0.0	0.000000	0.010548	0.0	0.0	0.0	0.0	0.020364	...
3	0.000000	0.000000	0.0	0.000000	0.008066	0.0	0.0	0.0	0.0	0.006224	...
4	0.000000	0.000000	0.0	0.000000	0.021878	0.0	0.0	0.0	0.0	0.010284	...
...	...	...	...	...	...	...	...	...	...	...	...
15789	0.000000	0.000000	0.0	0.000000	0.004184	0.0	0.0	0.0	0.0	0.000004	...
15790	0.000000	0.000000	0.0	0.000000	0.010785	0.0	0.0	0.0	0.0	0.026619	...
15791	0.000000	0.000000	0.0	0.000000	0.024141	0.0	0.0	0.0	0.0	0.015726	...
15792	0.032930	0.000000	0.0	0.000000	0.028387	0.0	0.0	0.0	0.0	0.010820	...
15793	0.029587	0.000624	0.0	0.000662	0.028967	0.0	0.0	0.0	0.0	0.007613	...

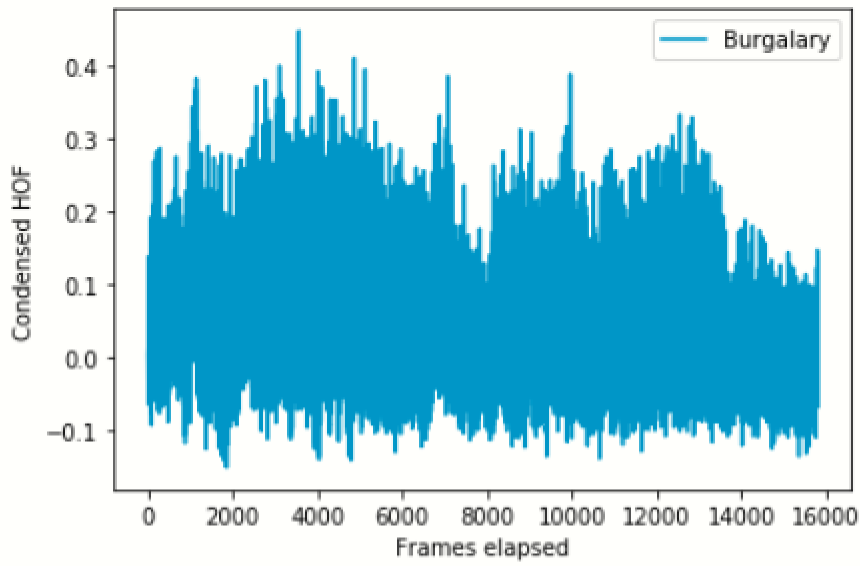
15794 rows × 541 columns

Range (15794)

Figure 6.5: HOF for burglary videos

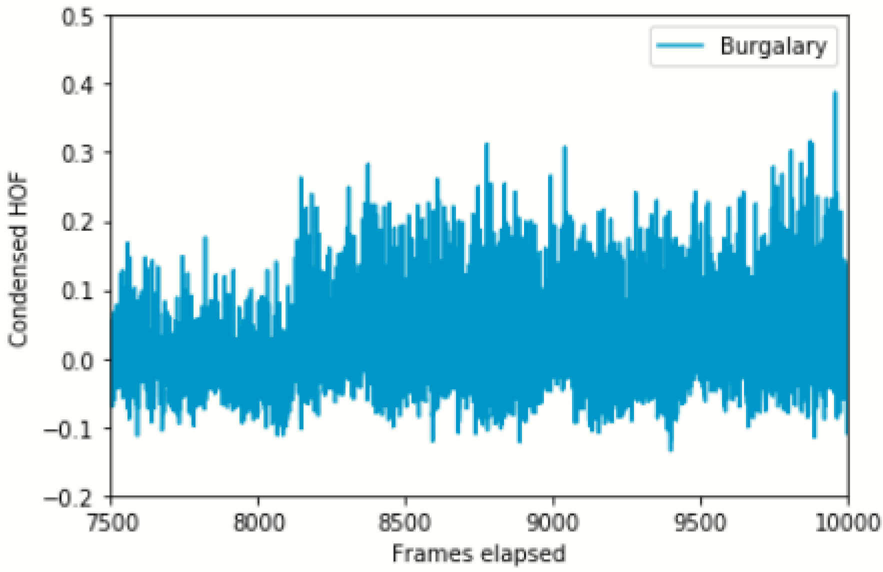
Using PCA to condense 540 features to 1 dimension
annotations['Abuse001_x264.mp4'
== 'Burglary032_x264.mp4']

	Abuse001_x264.mp4	1	230	365	-1	-1.1
230	Burglary032_x264.mp4	5	7782	9150	-1	-1

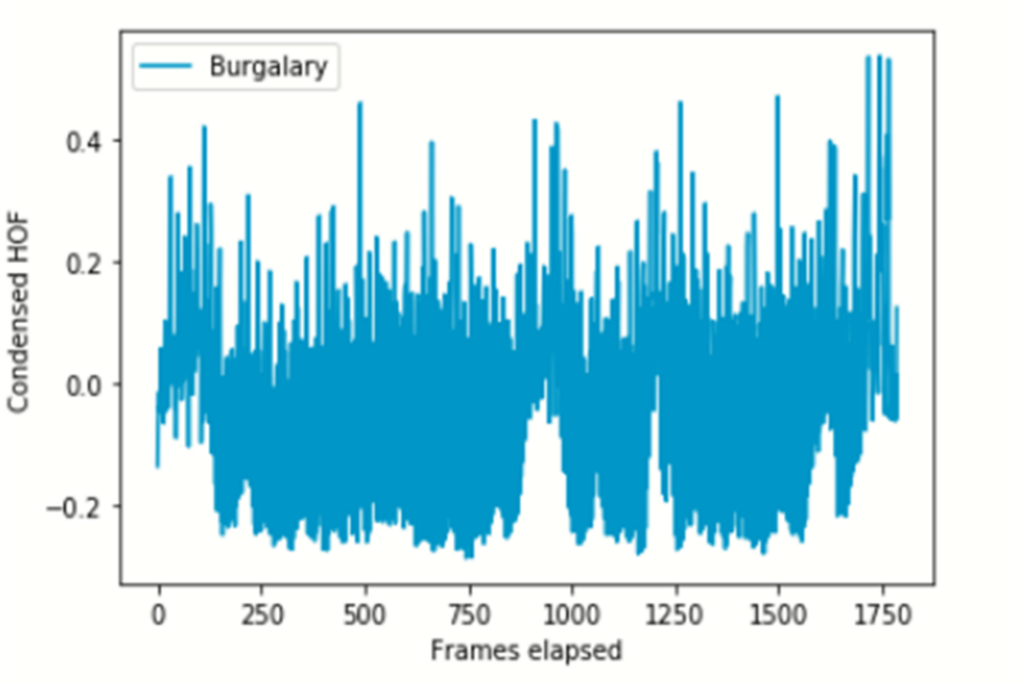


(a)

```
[7500, 10000, -0.2, 0.5]
```



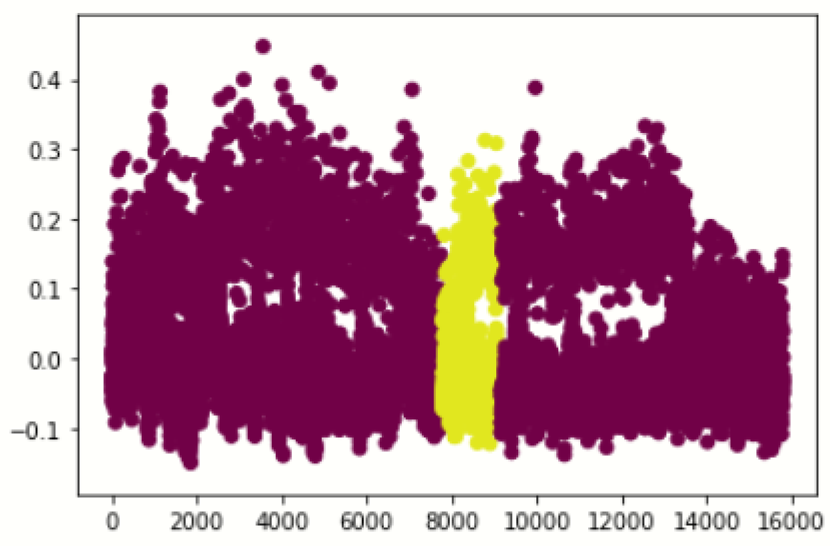
(b)



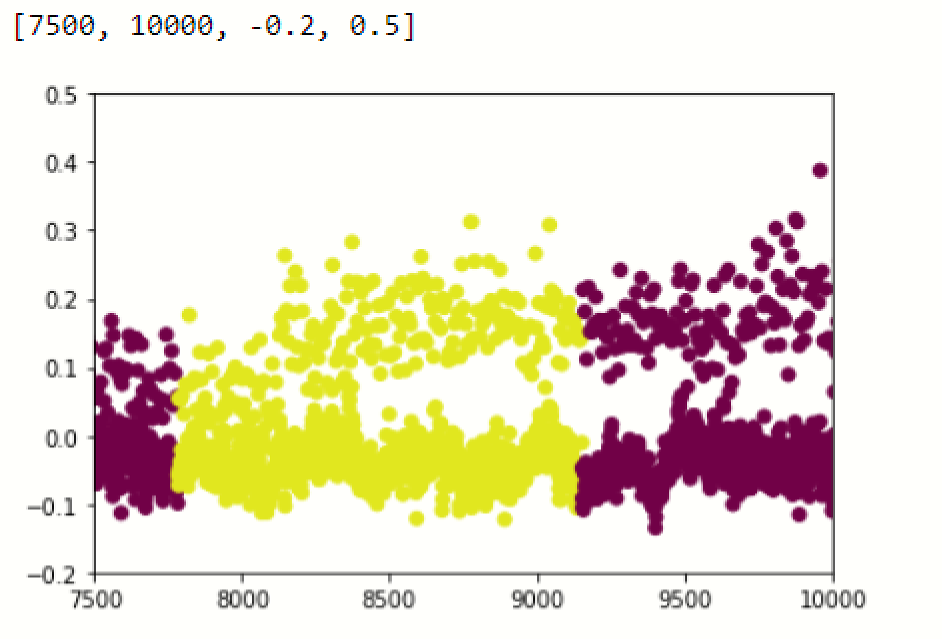
(c)

Figure 6.6: Burgalary frame elapse sequentially 15794 to 1786 (High to low)

In 6.6 shows how the plot changes according to the range. Reducing frame elapse gives better understanding. This if for Burgalary videos.



(a)



(b)

Figure 6.7: Scatter plot

Abuse visualization:

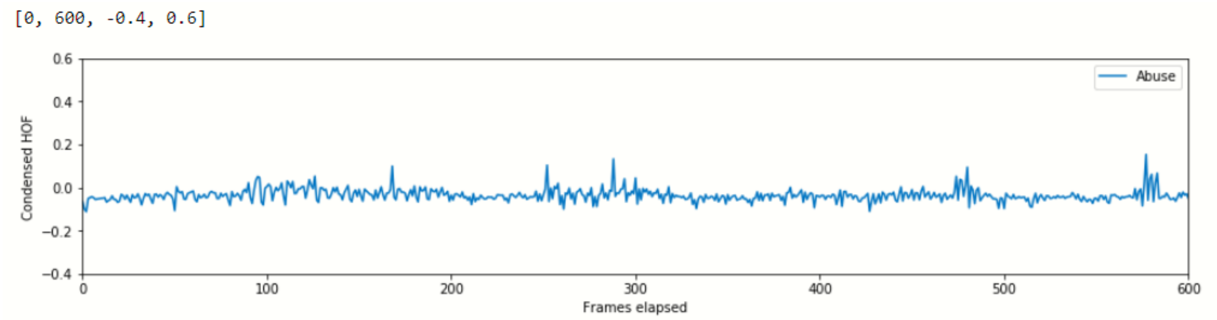


Figure 6.8: Abuse visualization for range 600.

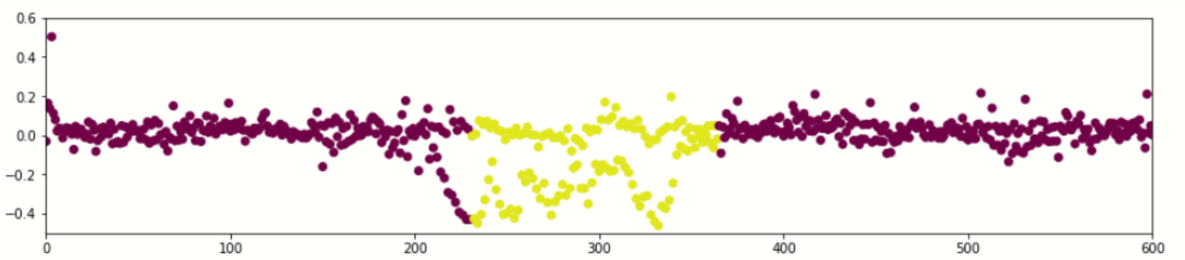


Figure 6.9: Scatter plot Abuse

In 6.8 shows how the line plot changes according to frame elapsed rate and HOF for abuse. Besides, 6.9 shows the scatter plot within this range.

Arson Visualization:

	Abuse001_x264.mp4	1	230	365	-1	-1.1
100	Arson002_x264.mp4	3	2183	3400	-1	-1

Total 2183 frames

No. of Frames / 30 FPS = Duration of Video in Seconds (2183/30 = 72.7666666667 sec.)

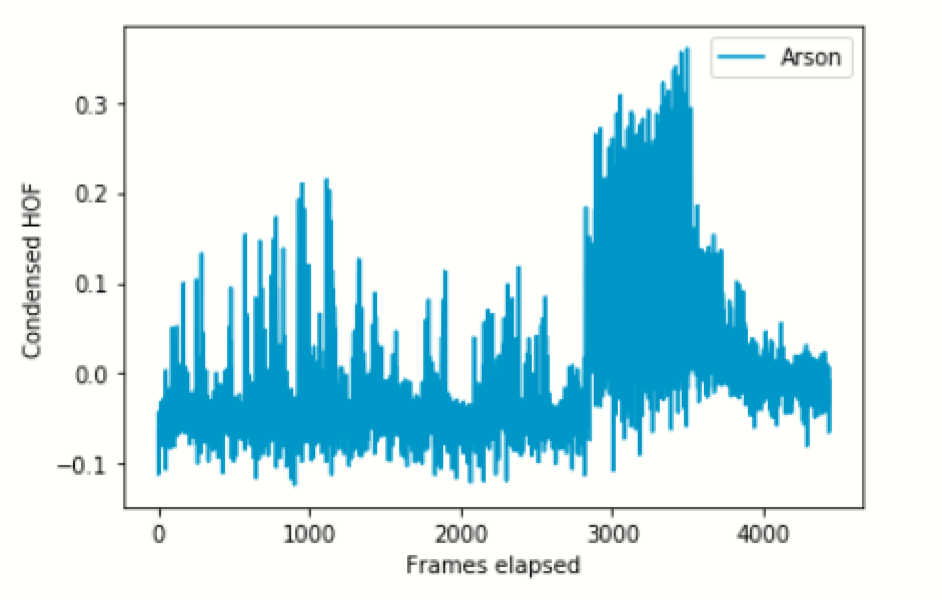
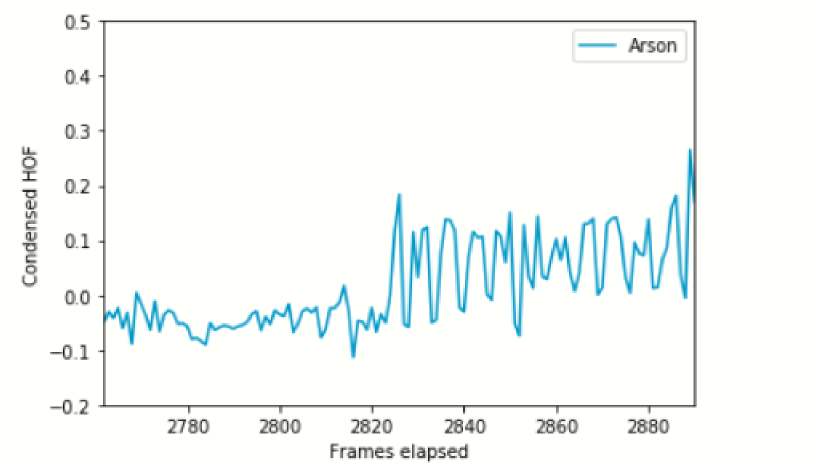


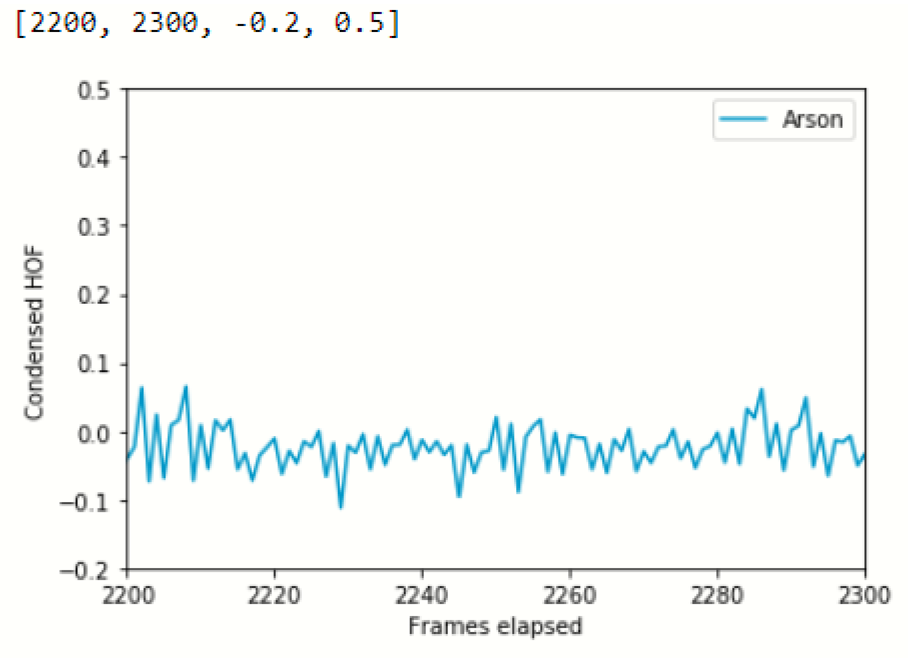
Figure 6.10: Arson visualization range (4438)

[2762, 2890, -0.2, 0.5]



2700/30 = 90.0 sec

Figure 6.11: Arson range (2762)



2200/30 = 73.333 sec

Figure 6.12: Arson range (2200).

Visual representation for arson videos in 6.10 6.11 and 6.12 shows the line plot changes for different range. As the videos have 30 frames per second. We can calculate the duration of each video

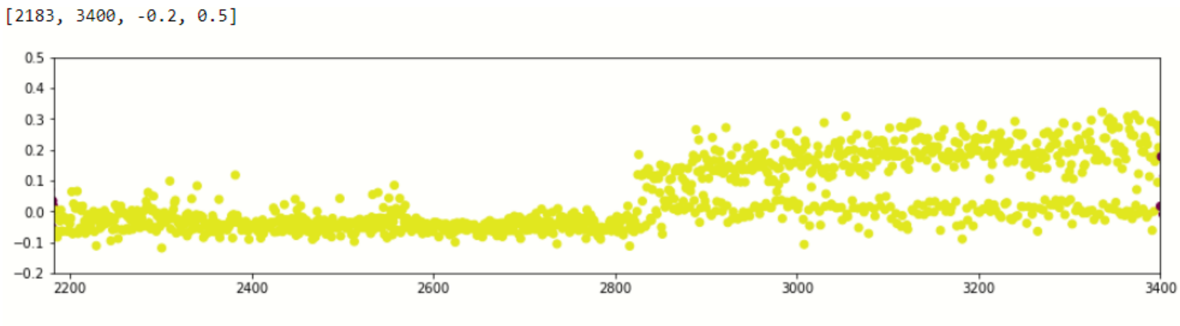


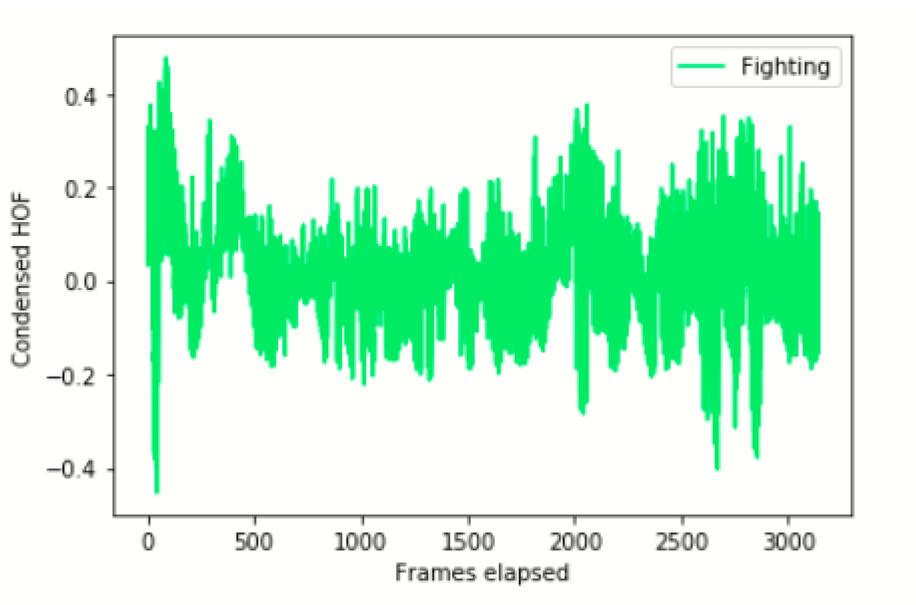
Figure 6.13: Scatter plot.

Fighting class visualization:

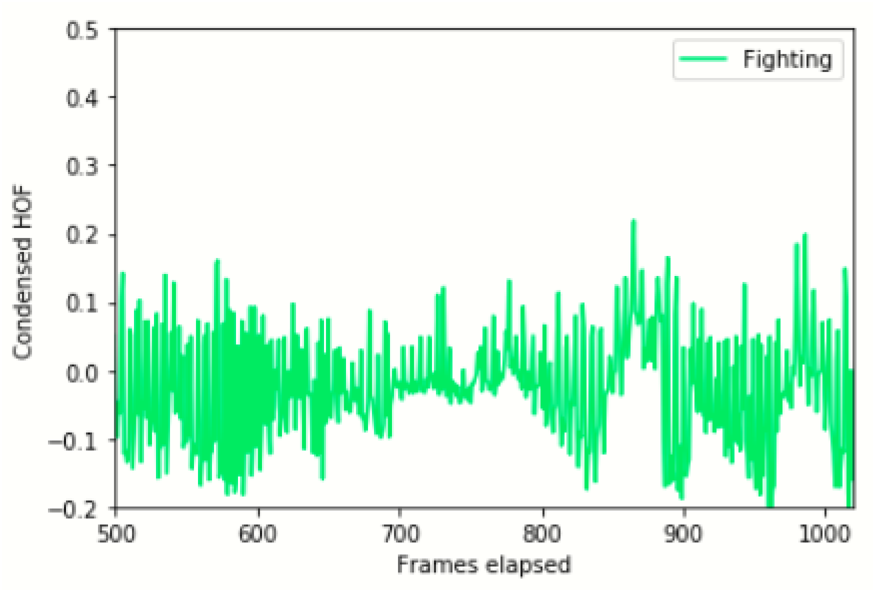
	Abuse001_x264.mp4	1	230	365	-1	-1.1
390	Fighting043_x264.mp4	7	500	1020	-1	-1

Frame = 500

Duration of video= 500/30 = 16.666666666667 sec.



range (31432)



Range (1020)

Figure 6.14: Fighting visualization

6.3.3 Box Plot for Number of Data Points and Data Distribution

Crime annotated video:

```
Total no of frames :
0    1007581
dtype: int64
                   0
count  5.950000e+02
mean  -3.100293e+16
std    5.342941e+17
min   -9.223372e+18
25%    2.320000e+02
50%    6.530000e+02
75%    1.550000e+03
max    1.079970e+05
```

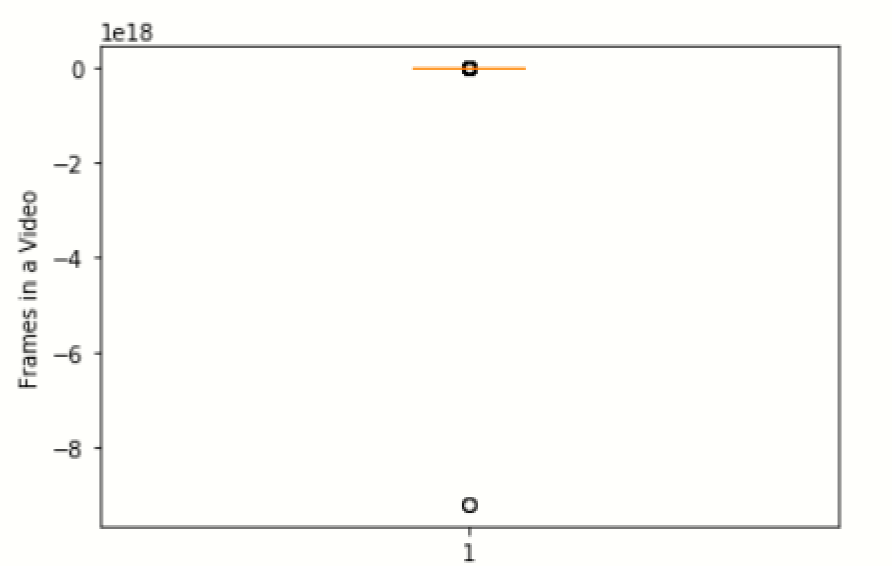


Figure 6.15: Box plot leveling.

Normal video frames:

```
Total no of frames :
0    70425
dtype: int64
                  0
count     50.000000
mean    1408.500000
std     1481.284182
min        0.000000
25%      312.250000
50%     1013.500000
75%     2222.250000
max     6177.000000
```

```
Total no of frames :
0    136136
dtype: int64
                  0
count    104.000000
mean    1309.000000
std     1186.285669
min        0.000000
25%      517.250000
50%     1093.000000
75%     1787.250000
max     8030.000000
```

```
Total no of frames :
0    31145
dtype: int64
                  0
count     72.000000
mean     432.569444
std      377.143408
min        0.000000
25%      122.750000
50%      311.000000
75%      707.500000
max     1301.000000
```

```
Total no of frames
0    16913
dtype: int64
                  0
count     54.000000
mean     313.203704
std      369.064978
min        0.000000
25%      100.000000
50%      196.500000
75%      365.750000
max     1850.000000
```

```
Total no of frames :
0    29061
dtype: int64
                  0
count     52.000000
mean     558.865385
std     1015.664180
min        0.000000
25%      102.000000
50%      340.500000
75%      594.000000
max     6751.000000
```

```
Total no of frames :
0    32346
dtype: int64
                  0
count     57.000000
mean     567.473684
std      611.254977
min        0.000000
25%      101.000000
50%      400.000000
75%      767.000000
max     2605.000000
```

```
Total no of frames :
0    42650
dtype: int64
                  0
count     53.000000
mean     804.716981
std     1045.341725
min        0.000000
25%      138.000000
50%      360.000000
75%     1071.000000
max     4500.000000
```

Figure 6.16: Frame vs video count, mean, std, min, max .

The 6.16 shows the frame count for total number of videos in each category. For that, we able to calculate std, min, min, max value. This gives clear idea about frame classification

Crime video frame:

```
Total no of frames :
0    648905
dtype: int64
                    0
count      150.000000
mean      4326.033333
std      12760.834906
min        209.000000
25%        918.500000
50%       1667.000000
75%       3503.500000
max     107997.000000
```

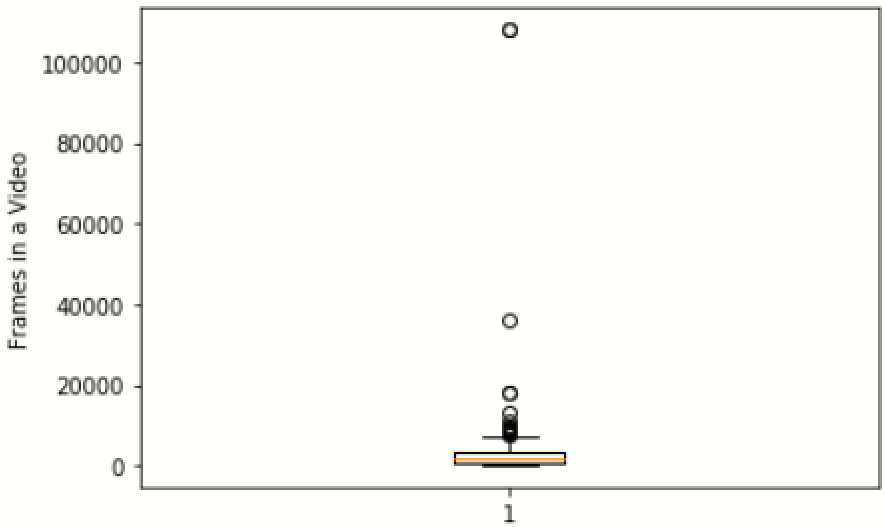


Figure 6.17: Total frames, video count, mean, std, min, max .

6.3.4 Feature Extraction Time Comparison

In this part we calculated total time, per frame time for Frame Embeds, HOF, HOG based on eight type of crime classes "abuse, arrest, arson, assault, burglary, explosion, fighting, normal"

1 Number of frames

2 FrameEmbedes/ HOF/ HOG

3 Total Time.

4 Per frame time

```
706 ____________________
FrameEmbeds
31.68400549888611
0.04487822852796603
HOF
4.746802568435669
0.006723548805409721
HOG
0.4184577465057373
0.0005927393186531391

1024 ____________________
FrameEmbeds
45.42604994773865
0.04436139389872551
HOF
6.912982702255249
0.00675097550265491
HOG
0.6584038734436035
0.0006429899949580431

41 ____________________
FrameEmbeds
1.8810088634490967
0.0458786138674108
HOF
0.3000943660736084
0.007319752762957317
HOG
0.045322418212890625
0.0011054248344607
```

```
195 ____________________
FrameEmbeds
8.629382371902466
0.04425333218696679
HOF
1.3428518772125244
0.006886507914616511
HOG
0.14068913459777832
0.0007215707730024289

1500 ____________________
FrameEmbeds
66.78902196884155
0.04452602672576904
HOF
10.209842681884766
0.006806573708852132
HOG
0.877178430557251
0.0005847970644632975

401 ____________________
FrameEmbeds
17.730401277542114
0.04421550555716726
HOF
2.756702423095703
0.006874616009338835
HOG
0.24434971809387207
0.0006093925371431651
```

```
270 ____________________
FrameEmbeds
11.838989973068237
0.04384817017449273
HOF
1.860790729522705
0.006891899638705783
HOG
0.17720532417297363
0.0006563822428585417

10025 ____________________
FrameEmbeds
442.756795167923
0.04416526811081275
HOF
68.07159161567688
0.00679018562868646
HOG
5.928550004959106
0.0005913782238662986
```

(Time comparison sequentially)

6.4 Method Prediction

We implemented a trainable model which can "Saves the current state of the model, Loads the saved model from the given export_dir, Train the model using the positive and negative batch"

6.4.1 CNN (Convolution Neural Network)

Result Analysis Before Iteration:
TPR defines how many correct positive results occur among all positive samples available during the test.
FPR defines how many incorrect positive results occur among all negative samples available during the test.


```
"true_values": [1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0],
"scores": [0.48498684, 0.46791032, 0.4888667, 0.4542461, 0.5012666,
  0.47919646, 0.47361252, 0.5059249, 0.46394345, 0.5383038, 0.49505943,
  0.5388621, 0.49089983, 0.4801692],
"predictions": [0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0],
"fpr": [0.0, 0.125, 0.125, 0.75, 0.75, 0.875, 0.875, 1.0, 1.0],
"tpr": [0.0, 0.0, 0.16666667, 0.16666667, 0.33333333, 0.33333333, 0.83333333, 0.83333333, 1.0],
"auc": 0.25
```


Figure 6.18: CNN before iteration.
Accuracy: 25%

Result Analysis After Iteration:

```
 0.83333333 0.83333333 1.        ]
Done. AUC:0.250000 Elapsed Time: 5.76576 sec
Training model...
Training EPOCH #0...
Finished. Cost: 1.44219. Time Elapsed: 3.86441 sec.
Training EPOCH #1...
Finished. Cost: 1.51430. Time Elapsed: 1.57607 sec.
Training EPOCH #2...
Finished. Cost: 1.47353. Time Elapsed: 1.56600 sec.
Training EPOCH #3...
Finished. Cost: 1.47450. Time Elapsed: 1.59275 sec.
Training EPOCH #4...
Finished. Cost: 1.49887. Time Elapsed: 1.58339 sec.
Training EPOCH #5...
Finished. Cost: 1.47148. Time Elapsed: 1.58152 sec.
Training EPOCH #6...
Finished. Cost: 1.49781. Time Elapsed: 1.59172 sec.
Training EPOCH #7...
Finished. Cost: 1.44835. Time Elapsed: 1.59357 sec.
Training EPOCH #8...
Finished. Cost: 1.44390. Time Elapsed: 1.59460 sec.
Training EPOCH #9...
Finished. Cost: 1.50889. Time Elapsed: 1.59019 sec.
Training EPOCH #10...
Finished. Cost: 1.51948. Time Elapsed: 1.59163 sec.
Training EPOCH #11...
Finished. Cost: 1.46465. Time Elapsed: 1.58261 sec.
Training EPOCH #12...
Finished. Cost: 1.46762. Time Elapsed: 1.59170 sec.
Training EPOCH #13...
Finished. Cost: 1.51702. Time Elapsed: 1.58223 sec.
Training EPOCH #14...
Finished. Cost: 1.49116. Time Elapsed: 1.59082 sec.
Training EPOCH #15...
Finished. Cost: 1.46097. Time Elapsed: 1.58113 sec.
Training EPOCH #16...
Finished. Cost: 1.46585. Time Elapsed: 1.57948 sec.
```

```
Training EPOCH #2988...
Finished. Cost: 0.68135. Time Elapsed: 1.60164 sec.
Training EPOCH #2989...
Finished. Cost: 0.84466. Time Elapsed: 1.61300 sec.
Training EPOCH #2990...
Finished. Cost: 0.64138. Time Elapsed: 1.62828 sec.
Training EPOCH #2991...
Finished. Cost: 0.65537. Time Elapsed: 1.63157 sec.
Training EPOCH #2992...
Finished. Cost: 0.76994. Time Elapsed: 1.60474 sec.
Training EPOCH #2993...
Finished. Cost: 0.81582. Time Elapsed: 1.60599 sec.
Training EPOCH #2994...
Finished. Cost: 0.67053. Time Elapsed: 1.59984 sec.
Training EPOCH #2995...
Finished. Cost: 0.65857. Time Elapsed: 1.60048 sec.
Training EPOCH #2996...
Finished. Cost: 0.61972. Time Elapsed: 1.60877 sec.
Training EPOCH #2997...
Finished. Cost: 0.63548. Time Elapsed: 1.60659 sec.
Training EPOCH #2998...
Finished. Cost: 0.68900. Time Elapsed: 1.61551 sec.
Training EPOCH #2999...
Finished. Cost: 0.60894. Time Elapsed: 1.59908 sec.
Done. Elapsed Time: 4813.87079 sec
Testing trained model...
TRUE LABELS:
[1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0]
SCORES:
[0.0016144643, 0.0438962, 0.07525045, 0.0048509124, 0.0027689121,
 0.12891941, 0.70771676, 0.0042794985, 0.12070409, 0.6477395,
0.0047873706, 0.0034672106, 0.26896137, 0.09113577]
PREDICTIONS:
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0]
FPR: [0.    0.    0.125 0.125 0.5   0.5   1.    1.   ]
TPR: [0.16666667 0.33333333 0.33333333 0.5        0.5        0.83333333
 0.83333333 1.        ]
Done. AUC:0.645833 Elapsed Time: 4.23288 sec
```

Figure 6.19: CNN after iteration.

The result Fig.6.19 shows that after 3000 iterations, the model was able to go from 0.250000 to 0.645833.

Final Accuracy for CNN: 64.5833%

6.4.2 FastGRNN

Result Analysis FastGRNN:


```
{
 "true_values": [0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1,
0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0],

 "scores": [0.15189523, 0.25026545, 0.36038864, 0.22268008, 0.8565128, 0.18306376, 0.15188594, 0.3298905, 0.51985943, 0.1670035,
0.34760967, 0.51975054, 0.14232656, 0.34746304, 0.7055121, 0.48649737, 0.46786496, 0.3272651, 0.4095528, 0.30418882, 0.25253287,
0.16101691, 0.6381502, 0.1289915, 0.16425905, 0.8775885, 0.69536287, 0.2889564, 0.25905627, 0.28935578, 0.2969002, 0.8691662,
0.5449785, 0.551839, 0.24923784, 0.78630143, 0.24160631, 0.2791857, 0.42169142, 0.91286457, 0.8485528, 0.13456152, 0.13077034,
0.34198308, 0.16443205, 0.11487684, 0.7456797, 0.6943358, 0.33060277, 0.5771214, 0.73160386, 0.6414973, 0.42065382, 0.5300108,
0.6370006, 0.22699, 0.40615577, 0.6253927, 0.42742717, 0.78792864, 0.17057657, 0.22522056, 0.26754326, 0.90559363, 0.5579381,
0.22991085, 0.6887664, 0.57749236, 0.17507955, 0.16135584, 0.84652966, 0.4615819, 0.23605183, 0.28863004, 0.8670522, 0.26185924,
0.70978457, 0.14418747, 0.7542475, 0.3426486, 0.87436926, 0.20853327, 0.82393855, 0.48720092, 0.18158075, 0.16088864, 0.19124207,
0.3147573, 0.88817835, 0.42370096, 0.22186215, 0.22955222],

 "predictions": [0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
1.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0,
0.0, 1.0, 1.0, 1.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0,
1.0, 0.0, 1.0, 0.0, 1.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0],

 "fpr": [0.0, 0.01694915, 0.01694915, 0.03389831, 0.03389831, 0.05084746, 0.05084746, 0.06779661, 0.06779661, 0.08474576, 0.08474576,
0.11864407, 0.11864407, 0.13559322, 0.13559322, 0.18644068, 0.18644068, 0.22033898, 0.22033898, 0.27118644, 0.27118644, 0.37288136,
0.37288136, 1.0],

 "tpr": [0.0, 0.0, 0.09090909, 0.09090909, 0.24242424, 0.24242424, 0.42424242, 0.42424242, 0.45454545, 0.45454545, 0.51515152,
0.51515152, 0.60606061, 0.60606061, 0.84848485, 0.84848485, 0.90909091, 0.9090909, 0.93939394, 0.93939394, 0.96969697, 0.96969697,
1.0, 1.0],

  "auc": 0.895737
```


Figure 6.20: FastGRNN result analysis.

Accuracy: 89.5737%

6.4.3 Binary Classification

Library used: torchvision, torch.nn. functional, torchvision. models, torch. optima, torch. auto grad, os, matplotlib. python, numpy, cv2, sys.

Result analysis Binary Classification:
After 19 phase we got the final accuracy

```
--- Epoch 0 ---

--- Phase train ---
[Epoch 0/20] [Batch 437/438] [Loss: 2.001631 (1.511374), Acc: 0.00% (41.87%)]
train , acc: 41.86643835616438

--- Epoch 1 ---

--- Phase train ---
[Epoch 1/20] [Batch 437/438] [Loss: 1.151204 (1.305457), Acc: 50.00% (49.12%)]
train , acc: 49.11529680365297

--- Epoch 2 ---

--- Phase train ---
[Epoch 2/20] [Batch 437/438] [Loss: 1.231476 (1.179129), Acc: 50.00% (53.97%)]
train , acc: 53.96689497716895

--- Epoch 3 ---

--- Phase train ---
[Epoch 3/20] [Batch 437/438] [Loss: 0.774739 (1.041741), Acc: 75.00% (60.62%)]
train , acc: 60.61643835616438

--- Epoch 4 ---

--- Phase train ---
[Epoch 4/20] [Batch 437/438] [Loss: 0.276145 (0.809180), Acc: 75.00% (70.01%)]
train , acc: 70.00570776255708

--- Epoch 5 ---

--- Phase train ---
[Epoch 5/20] [Batch 437/438] [Loss: 1.330135 (0.614348), Acc: 50.00% (78.00%)]
train , acc: 77.99657534246575
```

```
--- Epoch 6 ---

--- Phase train ---
[Epoch 6/20] [Batch 437/438] [Loss: 1.023739 (0.401012), Acc: 75.00% (85.79%)]
train , acc: 85.78767123287672

--- Epoch 7 ---

--- Phase train ---
[Epoch 7/20] [Batch 437/438] [Loss: 2.653315 (0.342477), Acc: 25.00% (88.16%)]
train , acc: 88.15639269406392

--- Epoch 8 ---

--- Phase train ---
[Epoch 8/20] [Batch 437/438] [Loss: 0.026995 (0.223468), Acc: 100.00% (92.15%)]
train , acc: 92.15182648401826

--- Epoch 9 ---

--- Phase train ---
[Epoch 9/20] [Batch 437/438] [Loss: 0.071458 (0.188113), Acc: 100.00% (93.64%)]
train , acc: 93.63584474885845

--- Epoch 10 ---

--- Phase train ---
[Epoch 10/20] [Batch 437/438] [Loss: 0.077914 (0.125080), Acc: 100.00% (95.66%)]
train , acc: 95.662100456621

--- Epoch 11 ---

--- Phase train ---
[Epoch 11/20] [Batch 437/438] [Loss: 0.109610 (0.097384), Acc: 100.00% (96.89%)]
train , acc: 96.8892694063927
```

```
--- Epoch 12 ---

--- Phase train ---
[Epoch 12/20] [Batch 437/438] [Loss: 0.149528 (0.073297), Acc: 100.00% (97.57%)]
train , acc: 97.57420091324201

--- Epoch 13 ---

--- Phase train ---
[Epoch 13/20] [Batch 437/438] [Loss: 0.000738 (0.064217), Acc: 100.00% (97.75%)]
train , acc: 97.74543378995433

--- Epoch 14 ---

--- Phase train ---
[Epoch 14/20] [Batch 437/438] [Loss: 0.006020 (0.036686), Acc: 100.00% (98.97%)]
train , acc: 98.97260273972603

--- Epoch 15 ---

--- Phase train ---
[Epoch 15/20] [Batch 437/438] [Loss: 0.000287 (0.029383), Acc: 100.00% (99.00%)]
train , acc: 99.00114155251141

--- Epoch 16 ---

--- Phase train ---
[Epoch 16/20] [Batch 437/438] [Loss: 0.001510 (0.020584), Acc: 100.00% (99.40%)]
train , acc: 99.40068493150685
```

Figure 6.21: Binary Classification analysis.

Accuracy: 99.68607305936074% (Overfitting)

The result analysis 6.21 shows that after 19 phase of train we got the best accuracy considering all the loss value using binary classifier. But the we found that the result is overfitting.

Loss value:

```
[1.5113742563822499,
 1.305457015968349,
 1.17912877885174,
 1.041740705569585,
 0.8091797970201327,
 0.6143478534809531,
 0.4010123163461685,
 0.3424767658590726,
 0.22346783888666596,
 0.18811332580705756,
 0.1250801009538511,
 0.09738352649832425,
 0.07329675657291936,
 0.06421710863777491,
 0.03668630755927465,
 0.029382538033402673,
 0.02058421796587504,
 0.01861625916609481,
 0.012762131824340994,
 0.011054905242027213]
```

Figure 6.22: Loss values.

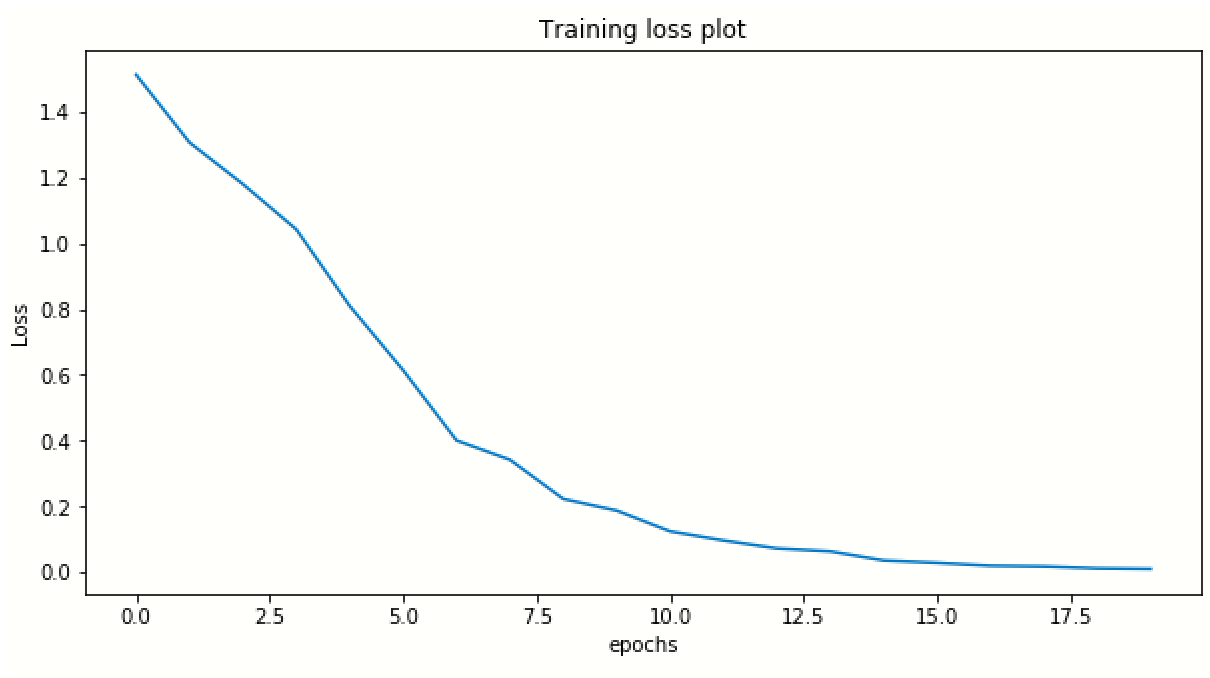


Figure 6.23: Training loss plot.

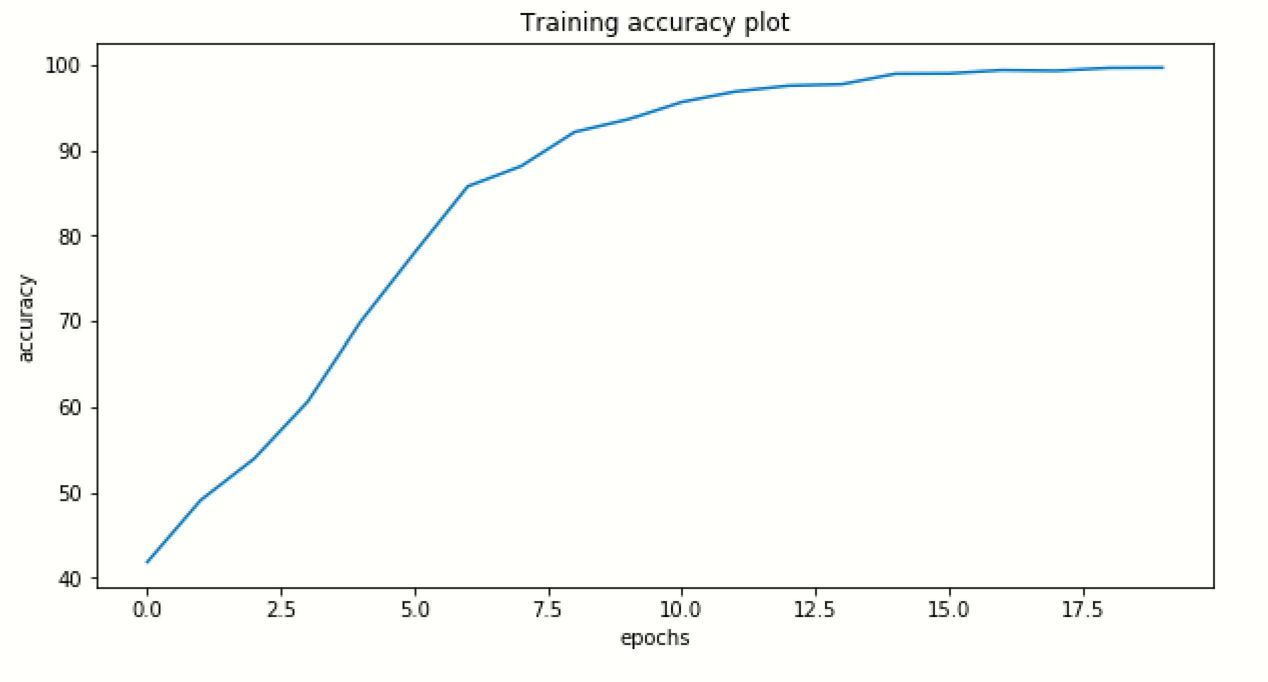


Figure 6.24: Training accuracy plot.

6.23 shows us a graphical representation of the loss value and 6.24 shows the training accuracy graph.

6.5 Comparison

Classifier	CNN	FastGRNN	Binary Classifier
Accuracy	64.5833%	89.5737%	99.68607305936074% (Overfitting)

Table 6.1: Comparison

Chapter 7

Conclusion

7.1 Summary

In our report, we have discussed different algorithms and approaches to predict criminal activities in a particular area. We have used open dataset from UCF crime. Using algorithms like binary classifier, CNN and faster GRNN. We have gotten low accuracy in CNN algorithm; binary classifier algorithm has given overfitting result. We have found fastGRNN algorithm best as its result is not overfitting and has given good accuracy. We have discussed data attributes and crime classification in our report. In conclusion we would like to say that implementing our system would reduce a greater number of incidents like rape, terrorist attacks, kidnapping etc. and provide everyone with safer and crime free environment.

7.2 Limitations

Given an extended untrimmed video, we wish to grasp whether it contains an anomalous event and where the event happens. Because of the large amount of video recordings and therefore the rare occurrence of anomalies, it is very challenging and expensive to get precise frame-level an-notations to coach a robust neural network. Other major limitations of our framework are, if we do not get the victim's characteristics or features that we are working with to predict the ultimate result then our system cannot predict the result. Moreover, Hyper Parameter Tuning may be a time-consuming process because it searches for each possible parameter by leading to the accuracy to work out best parameter.

7.3 Future Work

Technology makes each and every work productive. We can think of improving predicting criminal activities with the help of future technology and make them more futuristic. Directly off the beginnings we are wanting to set up the task appropriately and test it with bigger measure of certified records of crime. Now we will get our desired outcome, we will make this system more futuristic day by day for our Major Law and enforcement offices of our country as they can use this system for the betterment of our country. In future we have a plan to work on Yolo V3 and implement it in our system. It is till now one of the faster and accurate object detection algorithms. It makes detection at three different scales. It is also good to detect small objects so small objects can be detected easily in case of crime time. It will help this system to get better frame rate and analyzing live videos. The system can be launch to an android and iOS-based apps for the law and enforcement offices of our country.

Bibliography

[1] A. Mecocci, M. Pannozzo, and A. Fumarola, detection of anoma- lous behavioural events for advanced real-time video surveillance", in The 3rd International Workshop on Scientiicc Use of Submarine Cables and Related Technologies, 2003., IEEE, 2003, pp. 187192.

[2] D. Duque, H. Santos, and P. Cortez, of abnormal behaviors for intelligent video surveillance systems", in 2007 IEEE Symposium on Compu- tational Intelligence and Data Mining, IEEE, 2007, pp. 362367.

[3] A. K. S. Kushwaha, C. M. Sharma, M. Khare, R. K. Srivastava, and A. Khare, multiple human detection and tracking for visual surveillance sys- tem", in 2012 International Conference on Informatics, Electronics Vision (ICIEV), IEEE, 2012, pp. 326331.

[4] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, spa- tiotemporal features with 3d convolutional networks", in Proceedings of the IEEE international conference on computer vision, 2015, pp. 44894497.

[5] Z. Xu, L. Qing, and J. Miao, auto-completion: Predicting human activities from partial videos", in Proceedings of the IEEE international con- ference on computer vision, 2015, pp. 31913199.

[6]S. H. Bang, T. Kim, B. Ji, Y. Lim, and H. Cho, -time crime prediction method using criminal records", Jun. 2016.

[7] A. Kusupati, M. Singh, K. Bhatia, A. Kumar, P. Jain, and M. Varma, -grnn: A fast, accurate, stable and tiny kilobyte sized gated recurrent neu- ral network", in Advances in Neural Information Processing Systems, 2018, pp. 90179028.

[8] W. Sultani, C. Chen, and M. Shah, -world anomaly detection in surveillance videos", in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 64796488.

[9] S. T. Rahman, rape punishment in bangladesh: A comparative overview", PhD thesis, East West University, 2019.

[10] A. M. U. Wagdarikar and R. K. Senapati, based interesting region identi

cation for video watermarking", Journal of Information Security and Applications, vol. 49, p. 102 393, 2019.

[11] C. Lu, J. Shi, and J. Jia. Abnormal event detection at 150 fps in matlab.In ICCV, 2013

[12] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis. Learning temporal regularity in video sequences. In CVPR, June 2016.

[13] Akash Desai, G. (2017). Criminal Investigation Tracker with Suspect Prediction. IJESC.

[14] Shojaee, S., Mustapha, A., Sidi, F., Jabar, M. A. (2013). A study on classification learning algorithms to predict crime status. International Journal of Digital Content Technology and its Applications, 7 (9), 361

[15] Iqbal, R., Murad, M. A. A., Mustapha, A., Panahy, P. H. S., Khanahmadliravi, N. (2013). An experimental study of classification algorithms for crime prediction. Indian Journal of Science and Technology, 6 (3), 4219-4225.

[16] M. Valera, S.A. Velastin "A review of the State-of-the- Art in distributed surveillance systems", in Intelligent Distributed Systems (Eds. S.A. Velastin, P. Remagnino), IEE Publications 2005.