 **FACULTY OF ENGINEERING**

**TANTA UNIVERSITY**

**Automatic Fight Detection**

**In Surveillance Videos**

**By**

1. Shady Abdelaziz ElGhareb Sakr
2. Abdelrahman Atif Attia Elfiky
3. Abdelrahman Ali Sadik Abdella
4. Moaz Abdeltawab Ali ELzoughby
5. Mohamed Mortada Elasayed Salem
6. Yasser Abdelfatah Ahmed Nasr

Supervised by :

**Dr. Heba Ali Elkhobby**

(Assistant Professor)

This Project Documentation is submitted in partial fulfillment of the requirement for the Degree of B.Sc. In Electronics and Communication Engineering at Tanta University

**July 17, 2022**

**DEDICATION**

To our families, fathers, mothers, brothers, sisters and cousins All our friends, relatives, teachers and lecturers, for the support and care.

**ACKNOWLEDGEMENT**

We would like to express my gratitude to everyone who has been a part of this unforgettable adventure. When we look back from where our journey began, we recollect all the people who brought this project to completion.

First and foremost,We would like to thank our supervisor

**Dr. Heba Ali Elkhobby**

for her support, patience, continued guidance an

encouragement throughout the project.

We would like to express sincere gratitude to all the teaching and non-teaching faculty of the Electronics and Communication Engineering Department, our beloved parents, seniors and my dear friends for their constant support during the course of work.

**ABSTRACT**

Violence rates however have been brought down about 57% during the span of the past 4 decades yet it doesn't change the way that the demonstration of violence actually happens, unseen by the law. Violence can be mass controlled sometimes by higher authorities, however to hold everything in line one must "Micro Govern" over each movement occurring in every road of each square. To address the butterfly effects impact in our setting, We present an efficient method for detecting fight scenes in videos. Recent applications of convolution neural networks have shown promises of convolution layers for object detection and recognition, especially in images. However, convolution neural networks are supervised and require labels as learning signals.The proposed method is a deep learning based automatic detection approach that uses Convolutional Neural Network to detect violence present in a video. But, the disadvantage of using just CNN is that it requires a lot of time for computation and is less accurate. Hence, a pre-trained model, MobileNet, which provides higher accuracy and acts as a starting point for the building of the entire model.

We propose an architecture for fight detection in videos including crowded scenes. Our project helps detect the fight scenes in videos with high accuracy, thus saving time for organizations and individuals who would have to go through the entire footage instead.

CONTENTS

**[DEDICATION](#_tyjcwt)** 1

**[ACKNOWLEDGEMENT](#_3dy6vkm)** 2

**[ABSTRACT](#_1t3h5sf)** 4

**[Chapter 1: Introduction](#_4d34og8)** 9

[1.1 Problem Statement](#_17dp8vu) 10

[1.2 Objectives of the project](#_3rdcrjn) 10

**[Chapter2: Background](#_lnxbz9)** 11

[2.1 Artificial Intelligence](#_35nkun2) 11

[2.1.1 Artificial Intelligence Definition](#_2jxsxqh) 11

[2.1.2 The Necessity of Learning AI](#_3j2qqm3) 12

[2.1.3 Artificial intelligence Field of study](#_1y810tw) 13

[2.1.4 Artificial intelligence Mechanism](#_2xcytpi) 14

[2.1.5 Artificial Intelligence Importance](#_3whwml4) 16

[2.1.6 Artificial Intelligence Advantages](#_qsh70q) 18

[2.1.7 Artificial Intelligence Disadvantages](#_1pxezwc) 18

[2.1.8 Artificial Intelligence Applications](#_2p2csry) 18

[2.1.9 Machine Learning Definition](#_3o7alnk) 20

[2.2.10 Types of Machine Learning (ML)](#_23ckvvd) 20

[2.2.11 Most Common Machine Learning Algorithms](#_2grqrue) 22

[2.2.12 Deep learning Definition](#_4f1mdlm) 25

[2.2.13 Deep learning Mechanism](#_19c6y18) 25

[2.2.14 Types of Algorithms used in Deep Learning](#_nmf14n) 26

[2.2.15 A Brief History of Artificial Intelligence](#_1mrcu09) 30

[2.2 Computer Vision](#_46r0co2) 33

[2.2.1 Computer Vision Hierarchy](#_2lwamvv) 33

[2.2.2 Computer Vision Vs Image Processing](#_111kx3o) 38

[2.2.3 Computer vision Applications](#_206ipza) 38

**[Chapter3 :SYSTEM REQUIREMENT SPECIFICATION](#_4k668n3)** 40

[3.1 Hardware Requirement](#_2zbgiuw) 40

[3.2 Software Requirement](#_1egqt2p) 40

[3.3 Non-Functional Requirements](#_3ygebqi) 40

[3.3.1 Usability](#_2dlolyb) 40

[3.3.2 Assumption](#_sqyw64) 40

[3.3.3 Performance](#_3cqmetx) 41

[3.3.4 Reliability](#_1rvwp1q) 41

[3.4 Functional Requirements](#_4bvk7pj) 41

[3.5 Operating Environment](#_kgcv8k) 42

[3.5.1 Why Python?](#_34g0dwd) 42

[3.5.2 Features of Python](#_2iq8gzs) 43

[3.6 Mathematics Libraries](#_3hv69ve) 44

[3.6.1 NumPy](#_1x0gk37) 44

[3.6.2 Matplotlib](#_4h042r0) 46

[3.6.3 Seaborn](#_1baon6m) 46

[3.6.4 OpenCV](#_2afmg28) 47

[3.6.5 Keras](#_39kk8xu) 48

[3.6.6 TensorFlow](#_1opuj5n) 49

[3.6.7 Sklearn](#_1302m92) 51

[3.6.8 Time](#_3mzq4wv) 51

[3.6.9 Collections](#_2250f4o) 51

[3.6.10 Tqdm](#_319y80a) 52

[3.6.11 Pickle](#_1gf8i83) 52

[3.6.12 imageio](#_2fk6b3p) 53

[3.6.13 imgaug](#_upglbi) 53

**[Chapter 4: Implementation](#_4du1wux)** 54

[4.1 Method](#_2szc72q) 54

[4.2 Dataset](#_3s49zyc) 54

[4.2.1 Hockey fights](#_45jfvxd) 55

[4.2.2 Movies](#_zu0gcz) 56

[4.2.3 Real Life Violence Dataset](#_1yyy98l) 56

[4.3 Pre-Processing](#_2y3w247) 56

[4.4 Training](#_2ce457m) 60

[4.4.1 MobileNet v2 :](#_rjefff) 60

[4.4.2 Optimization for training](#_4anzqyu) 63

**[Chapter 5: RESULTS AND DISCUSSIONS](#_243i4a2)** 70

**[Chapter 6 : CONCLUSION](#_42ddq1a)** 72

[6.1 Future scope](#_2hio093) 72

**[REFERENCES](#_3gnlt4p)** 73

**[APPENDIX](#_1vsw3ci)** 755

LIST OF FIGURES

[Figure 1 Objectives of the project 1](#_Toc8585)0

[Figure 2 AI 1](#_Toc9400)2

[Figure 3 Vectors 2](#_Toc15281)3

[Figure 4 CNN Extractor 2](#_Toc15848)7

[Figure 5 LSTM 2](#_Toc15220)7

[Figure 6 RNN 2](#_Toc19004)8

[Figure 7 RNN 2](#_Toc32177)8

[Figure 8 MLP 2](#_Toc25904)9

[Figure 9 Aotuencoders 2](#_Toc3702)9

[Figure 10 History of AI 3](#_Toc13529)0

[Figure 11 Images 3](#_Toc19232)5

[Figure 12 method 5](#_Toc15657)4

[Figure 13 Dataset 5](#_Toc28612)5

[Figure 14 cv2 5](#_Toc30831)7

[Figure 15 Image Processing 5](#_Toc14018)8

[Figure 16 Cv2.Resize 5](#_Toc26394)9

[Figure 17 Training 6](#_Toc30557)0

[Figure 18 mobilenetv2 (1) 6](#_Toc29885)1

[Figure 19 mobilenetv2 (2) 6](#_Toc26788)2

[Figure 20 Training Code 6](#_Toc28716)3

[Figure 21 Adam 6](#_Toc15411)4

[Figure 22 Activation FN 6](#_Toc10009)8

[Figure 23 Sigmoid FN 7](#_Toc10009)0

[Figure 24 Training and Validation Acc 7](#_Toc10009)1

[Figure 25 Training and Validation Loss 7](#_Toc10009)1

[Figure 26 Output frame that recognized violence 7](#_Toc10009)2

[Figure 27 Output frame that recognized Non-violence 7](#_Toc10009)2

**Chapter 1: Introduction**

Violent behavior in public places is an issue that has to be addressed. Communities are also eroded by violence, which reduces productivity,lowers property values, and disrupts social services. Across the world, violence is a severe public health issue. It affects people at various phases of life, from infants to the elderly.

Recognizing violence is challenging since it must be done on real-time videos captured by a large number of surveillance cameras at any time and in any location. It should be able to make reliable real-time detection and alert corresponding authorities as soon as violent activities occur.

Public video surveillance systems are widespread around the world and can provide accurate and complete information in many security applications. However, having to watch videos for hours reduces your ability to make quick decisions. Video surveillance is essential to prevent crime and violence. In this regard, several studies have been published on the automatic detection of scenes of violence in video. This is so that authorities do not have to watch videos for hours to identify events that only last a few seconds. Recent studies have highlighted the accuracy of deep learning approaches to violence detection.

In this work we will be discussing the implementation of an Automatic Fight Detection in Surveillance Videos system using MobileNetv2.

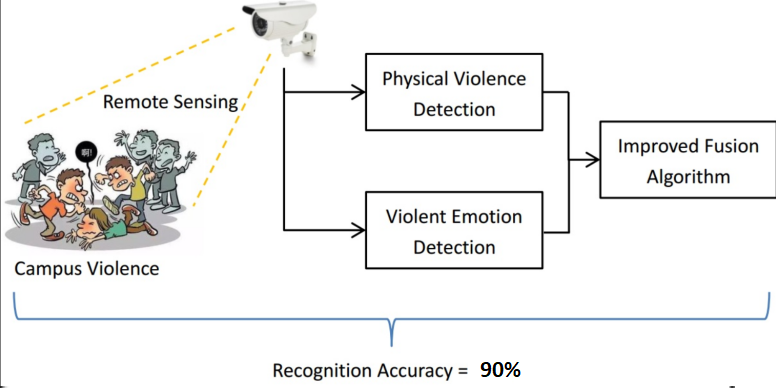
**1.1 Problem Statement**

* CCTV Surveillance is used to a greater extent but still it lacks the feature of automatic violence detection.
* Manual monitoring is not a feasible task and the time taken to respond to the situation is also crucial.
* A Real-time violence alert system is proposed.

**1.2 Objectives of the project**

The main objective of this system is to detect and localize fight

scenes, suspicious behaviors or irregular events in a scene. The method detects fight regions in a video and it is fast enough for real-time applications.

****

**Figure 1 Objectives of the project**

**Chapter2: Background**

**2.1 Artificial Intelligence**

Since the invention of computers or machines, their capability to perform various tasks has experienced an exponential growth. Humans have developed the power of computer systems in terms of their diverse working domains, their increasing speed, and reducing size with respect to time.

A branch of Computer Science named Artificial Intelligence pursues creating computers or machines as intelligent as human beings.

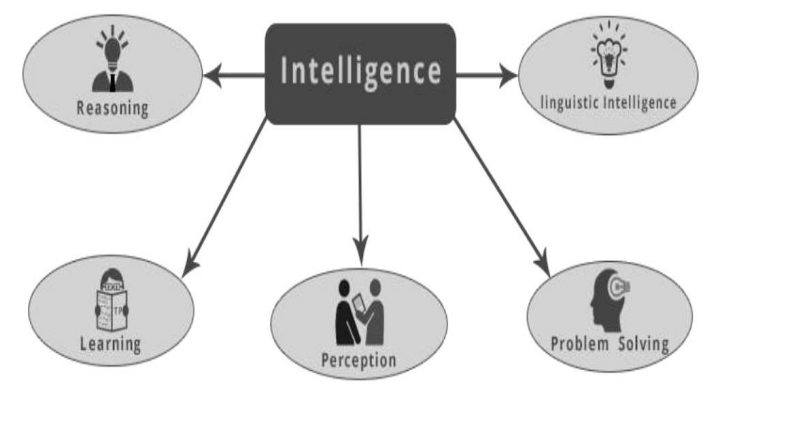
**2.1.1 Artificial Intelligence Definition**

According to the father of Artificial Intelligence, John McCarthy, it is “The science and engineering of making intelligent machines, especially intelligent computer programs”.

Artificial Intelligence is a way of making a computer, a computer-controlled robot, or a software think intelligently, in the similar manner the intelligent humans think.

AI is accomplished by studying how the human brain thinks and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems. While exploiting the power of the computer systems, the curiosity of humans, led him to wonder, “Can a machine think and behave like humans do?”

Thus, the development of AI started with the intention of creating similar intelligence in machines that we find and regard high in humans.



**Figure 2 Ai**

**2.1.2 The Necessity of Learning AI**

As we know, AI pursues creating machines as intelligent as human beings. There are numerous reasons for us to study AI. The reasons are as follows:

* **AI can learn through data**

In our daily life, we deal with huge amounts of data and the human brain cannot keep track of so much data. That is why we need to automate things. For doing automation, we need to study AI because it can learn from data and can do repetitive tasks with accuracy and without tiredness.

* **AI can teach itself**

It is very necessary that a system should teach itself because the data itself keeps changing and the knowledge which is derived from such data must be updated constantly. We can use AI to fulfill this purpose because an AI enabled system can teach itself.

* **AI can respond in real time**

Artificial intelligence with the help of neural networks can analyze the data more deeply. Due to this capability, AI can think and respond to the situations which are based on the conditions in real time.

* **AI achieves accuracy**

With the help of deep neural networks, AI can achieve tremendous accuracy. AI helps in the field of medicine to diagnose diseases such as cancer from the MRIs of patients.

* **AI can organize data to get the most out of it**

The data is an intellectual property for the systems which are using self-learning algorithms. We need AI to index and organize the data in a way that it always gives the best results.

* **Understanding Intelligence**

With AI, smart systems can be built. We need to understand the concept of intelligence so that our brain can construct another intelligence system like itself.

**2.1.3 Artificial intelligence Field of study**

Artificial intelligence is a vast area of study. This field of study helps in finding solutions to real world problems. Let us now see the different fields of study within AI:

* **Machine Learning**

It is one of the most popular fields of AI. The basic concept of this field is to make machine learning from data as the human beings can learn from his/her experience. It contains learning models on the basis of which the predictions can be made on unknown data.

* **Logic**

It is another important field of study in which mathematical logic is used to execute computer programs. It contains rules and facts to perform pattern matching, semantic analysis, etc.

* **Searching**

This field of study is basically used in games like chess, tic-tac-toe. Search algorithms give the optimal solution after searching the whole search space.

* **Artificial neural networks**

This is a network of efficient computing systems the central theme of which is borrowed from the analogy of biological neural networks. ANN can be used in robotics, speech recognition, speech processing, etc.

* **Genetic algorithms**

Genetic algorithms help in solving problems with the assistance of more than one program. The result would be based on selecting the fittest.

* **Knowledge Representation**

It is the field of study with the help of which we can represent the facts in a way that is understandable to the machine. The more efficiently knowledge is represented; the more intelligent the system would be.

**2.1.4 Artificial intelligence Mechanism**

AI works by combining large amounts of data with fast, iterative processing and intelligent algorithms, allowing the software to learn automatically from patterns or features in the data. AI is a broad field of study that includes many theories, methods and technologies, as well as the following major subfields:

* **Machine learning**

automates analytical model building. It uses methods from neural networks, statistics, operations research and physics to find hidden insights in data without explicitly being programmed for where to look or what to conclude.

* **A neural Network**

is a type of machine learning that is made up of interconnected units (like neurons) that processes information by responding to external inputs, relaying information between each unit. The process requires multiple passes at the data to find connections and derive meaning from undefined data.

* **Deep learning**

uses huge neural networks with many layers of processing units, taking advantage of advances in computing power and improved training techniques to learn complex patterns in large amounts of data. Common applications include image and speech recognition.

* **Computer Vision**

relies on pattern recognition and deep learning to recognize what’s in a picture or video. When machines can process, analyze and understand images, they can capture images or videos in real time and interpret their surroundings.

* **Natural Language Processing (NLP)**

is the ability of computers to analyze, understand and generate human language, including speech. The next stage of NLP is natural language interaction, which allows humans to communicate with computers using normal, everyday language to perform tasks.

additionally, several technologies enable and support AI:

* **Graphical Processing Units**

are key to AI because they provide the heavy compute power that’s required for iterative processing. Training neural networks requires big data plus compute power.

* **The Internet Of Things**

generates massive amounts of data from connected devices, most of it unanalyzed. Automating models with AI will allow us to use more of it.

* **Advanced Algorithms**

are being developed and combined in new ways to analyze more data faster and at multiple levels. This intelligent processing is key to identifying and predicting rare events, understanding complex systems and optimizing unique scenarios.

* **APIs,Application Programming Interface**

are portable packages of code that make it possible to add AI functionality to existing products and software packages. They can add image recognition capabilities to home security systems and Q&A capabilities that describe data, create captions and headlines, or call out interesting patterns and insights in data.

In summary, the goal of AI is to provide software that can reason on input and explain on output. AI will provide human-like interactions with software and offer decision support for specific tasks, but it’s not a replacement for humans – and won’t be anytime soon.

**2.1.5 Artificial Intelligence Importance**

* **AI automates repetitive learning and discovery through data,**Instead of automating manual tasks, AI performs frequent, high-volume, computerized tasks. And it does so reliably and without fatigue. Of course, humans are still essential to set up the system and ask the right questions.
* **AI adds intelligence to existing products**.Many products you already use will be improved with AI capabilities, much like Siri was added as a feature to a new generation of Apple products. Automation, conversational platforms, bots and smart machines can be combined with large amounts of data to improve many technologies. Upgrades at home and in the workplace, range from security intelligence and smart cams to investment analysis.
* **AI adapts through progressive learning algorithms** to let the data do the programming. AI finds structure and regularities in data so that algorithms can acquire skills. Just as an algorithm can teach itself to play chess, it can teach itself what product to recommend next online. And the models adapt when given new data.
* **AI analyzes more and deeper data using neural networks** that have many hidden layers. Building a fraud detection system with five hidden layers used to be impossible. All that has changed with incredible computer power and big data . You need lots of data to train deep learning models because they learn directly from the data.
* **AI achieves incredible accuracy** through deep neural networks. For example, your interactions with Alexa and Google are all based on deep learning. And these products keep getting more accurate the more you use them. In the medical field, AI techniques from deep learning and object recognition can now be used to pinpoint cancer on medical images with improved accuracy.
* **AI gets the most out of data**. When algorithms are self-learning, the data itself is an asset. The answers are in the data. You just have to apply AI to find them. Since the role of the data is now more important than ever, it can create a competitive advantage. If you have the best data in a competitive industry, even if everyone is applying similar techniques, the best data will win.

**2.1.6 Artificial Intelligence Advantages**

* It defines a more powerful and more useful computers
* It introduces a new and improved interface for human interaction
* It introduces a new technique to solve new problems.
* It handles the information better than humans.
* It is very helpful for the conversion of information into knowledge
* It improves work efficiency so it reduces the duration of time to accomplish a task in comparison to humans.

**2.1.7 Artificial Intelligence Disadvantages**

* The implementation cost of AI is very high.
* The difficulties with software development for AI implementation are that the development of software is slow and expensive. Few efficient programmers are available to develop software to implement artificial intelligence.
* A robot is one of the implementations of Artificial intelligence with them replacing jobs and leading to unemployment.
* Machines can easily lead to destruction if the implementation of machines is put in the wrong hands. The results are hazardous for human beings.

**2.1.8 Artificial Intelligence Applications**

* **Natural Language Processing**

It is possible to interact with a computer that understands natural language spoken by humans.

* **Expert Systems**

There are some applications which integrate machine, software, and special information to impart reasoning and advising. They provide explanations and advice to the users. Vision Systems These systems understand, interpret, and comprehend visual input on the computer. For example,

* A spying aeroplane takes photographs, which are used to figure out spatial information or map of the areas.
* Doctors use a clinical expert system to diagnose the patient.
* Police use computer software that can recognize the face of a criminal with the stored portrait made by a forensic artist.
* **Speech Recognition**

Some intelligent systems are capable of hearing and comprehending the language in terms of sentences and their meanings while a human talks to it. It can handle different accents, slang words, noise in the background, change in human’s noise due to cold, etc.

* **Handwriting Recognition**

The handwriting recognition software reads the text written on paper by a pen or on screen by a stylus. It can recognize the shapes of the letters and convert it into editable text.

* **Intelligent Robots**

Robots are able to perform the tasks given by a human. They have sensors to detect physical data from the real world such as light, heat, temperature, movement, sound, bump, and pressure. They have efficient processors, multiple sensors and huge memory, to exhibit intelligence. In addition, they are capable of learning from their mistakes and they can adapt to the new environment

**2.1.9 Machine Learning Definition**

It may be defined as the field of computer science, more specifically an application of artificial intelligence, which provides computer systems the ability to learn with data and improve from experience without being explicitly programmed.

Basically, the main focus of machine learning is to allow computers to learn automatically without human intervention.

**2.2.10 Types of Machine Learning (ML)**

Machine Learning Algorithms help computer systems learn without being explicitly programmed. These algorithms are categorized into supervised or unsupervised. Let us now see a few algorithms:

**Supervised machine learning algorithms**

This is the most commonly used machine learning algorithm. It is called supervised because the process of algorithm learning from the training dataset can be thought of as a teacher supervising the learning process.In this kind of ML algorithm, the possible outcomes are already known and training data is also labeled with correct answers.

In simple words, we can say that in unsupervised learning there will be no correct answer and no teacher for guidance. Algorithms help to discover interesting patterns in data.

Mainly supervised learning problems can be divided into the following two kinds of problems:

* **Classification:** A problem is called classification problem when we have categorized output such as “black”, “teaching”, “non-teaching”, etc.
* **Regression:** A problem is called a regression problem when we have the real value output such as “distance”, “kilogram”, etc.
* **Decision tree, random forest, knn, logistic regression** are the examples of supervised machine learning algorithms.

**Unsupervised machine learning algorithms**

These kinds of machine learning algorithms do not have any supervisor to provide any sort of guidance. That is why unsupervised machine learning algorithms are closely aligned with what some call true artificial intelligence

Unsupervised learning problems can be divided into the following two kinds of problems:

* **Clustering:** In clustering problems, we need to discover the inherent groupings in the data.
* **Association:** A problem is called an association problem because such kinds of problems require discovering the rules that describe large portions of our data.
* **K-means for clustering, Apriori algorithm for association** are the examples of unsupervised machine learning algorithm

**Reinforcement machine learning algorithms**

These kinds of machine learning algorithms are used very less. These algorithms train the systems to make specific decisions.

Basically, the machine is exposed to an environment where it trains itself continually using the trial and error method.

These algorithms learn from past experience and try to capture the best possible knowledge to make accurate decisions. Markov Decision Process is an example of reinforcement machine learning algorithms

**2.2.11 Most Common Machine Learning Algorithms**

**Linear Regression**

It is one of the most well-known algorithms in statistics and machine learning. Linear regression is of the following two types:

* **Simple linear regression:** A linear regression algorithm is called simple linear regression if it has only one independent variable.
* **Multiple linear regression:** A linear regression algorithm is called multiple linear regression if it is having more than one independent variable.

**Basic concept:** Mainly linear regression is a linear model that assumes a linear relationship between the input variables and the single output variable The relationship between variables can be established by fitting a best line.

**Logistic Regression**

It is a classification algorithm and also known as **logit** regression.

Mainly logistic regression is a classification algorithm that is used to estimate the discrete values like 0 or 1, true or false, yes or no based on a given set of independent variables. Basically, it predicts the probability hence its output lies in between 0 and 1.

**Decision Tree**

Decision tree is a supervised learning algorithm that is mostly used for classification problems. Basically it is a classifier expressed as a recursive partition based on the independent variables.Decision tree has nodes which form the rooted tree. Rooted tree is a directed tree with a node called “root”. Root does not have any incoming edges and all the other nodes have one incoming edge. These nodes are called leaves or decision nodes

**Support Vector Machine (SVM)**

It is used for both classification and regression problems. But mainly it is used for classification problems.

**The main concept of SVM** is to plot each data item as a point in n dimensional space with the value of each feature being the value of a particular coordinate. Here n would be the features we would have. Following is a simple graphical representation to understand the concept of SVM:

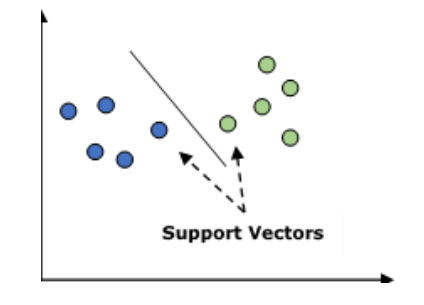


Figure 3 Vectors

In the above diagram, we have two features hence we first need to plot these two variables in two dimensional space where each point has two coordinates, called support vectors. The line splits the data into two different classified groups. This line would be the classifier.

**K-Nearest Neighbors (KNN)**

It is used for both classification and regression of the problems. It is widely used to solve classification problems. **The main concept** of this algorithm is that it is used to store all the available cases and classifies new cases by majority votes of its k neighbors. The case being then assigned to the class which is the most common amongst its K-nearest neighbors, measured by a distance function. The distance function can be Euclidean, Minkowski and Hamming distance.

Consider the following to use KNN:

* Computationally KNN is more expensive than other algorithms used for classification problems.
* The normalization of variables needed otherwise higher range variables can bias it.
* In KNN, we need to work on pre-processing stage like noise removal

**K-Means Clustering**

As the name suggests, it is used to solve the clustering problems. It is basically a type of unsupervised learning. The main logic of the K-Means clustering algorithm is to classify the data set through a number of clusters.

Follow these steps to form clusters by K-means:

* K-means picks k number of points for each cluster known as centroids.
* Now each data point forms a cluster with the closest centroids, i.e., k clusters.
* Now, it will find the centroids of each cluster based on the existing cluster members.
* We need to repeat these steps until convergence occurs.

**Random Forest**

It is a supervised classification algorithm. The advantage of the random forest algorithm is that it can be used for both classification and regression kind of problems. Basically it is the collection of decision trees (i.e., forest) or you can say ensemble of the decision trees. The basic concept of random forest is that each tree gives a classification and the forest chooses the best classifications from them. Followings are the advantages of Random Forest algorithm:

* Random forest classifiers can be used for both classification and regression tasks.
* They can handle the missing values.
* It won’t overfit the model even if we have more trees in the forest.

**2.2.12 Deep learning Definition**

Deep learning is a type of machine learning and artificial intelligence (AI) that imitates the way humans gain certain types of knowledge. Deep learning is an important element of data science, which includes statistics and predictive modeling. It is extremely beneficial to data scientists who are tasked with collecting, analyzing and interpreting large amounts of data; deep learning makes this process faster and easier

**2.2.13 Deep learning Mechanism**

Each algorithm in the hierarchy applies a nonlinear transformation to its input and uses what it learns to create a statistical model as output.

iterations continue until the output has reached an acceptable

level of accuracy. The number of processing layers through which data must pass is what inspired the label.

initially, the computer program might be provided with training data.

A computer program that uses deep learning algorithms can be shown a training set and sort through millions of images, accurately identifying which images have dogs in them within a few minutes.

To achieve an acceptable level of accuracy, deep learning programs require access to immense amounts of training data and processing power, neither of which were easily available to programmers until the era of big data and cloud computing.

Because deep learning programming can create complex statistical models directly from its own iterative output, it is able to create accurate predictive models from large quantities of unlabeled, unstructured data. This is important as the internet of things (IoT) continues to become more pervasive because most of the data humans and machines create is unstructured and is not labeled.

**2.2.14 Types of Algorithms used in Deep Learning**

* **Convolutional Neural Networks (CNNs)**

CNN's, also known as ConvNets, consist of multiple layers and are mainly used for image processing and object detection.

Yann LeCun developed the first CNN in 1988 when it was called LeNet. It was used for recognizing characters like ZIP codes and digits.

CNN's are widely used to identify satellite images, process medical images, forecast time series, and detect anomalies.

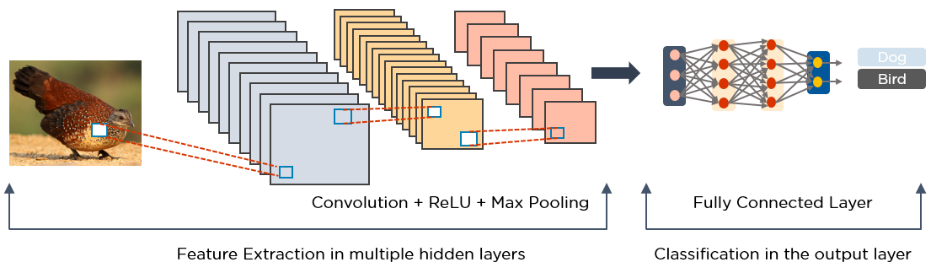
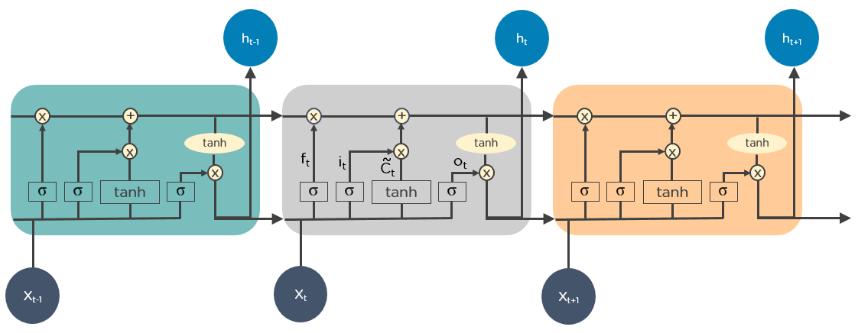


Figure 4 CNN Extractor

* **Long Short Term Memory Networks (LSTMs)**

LSTMs are a type of Recurrent Neural Network (RNN) that can learn and memorize long-term dependencies. Recalling past information for long periods is the default behavior.

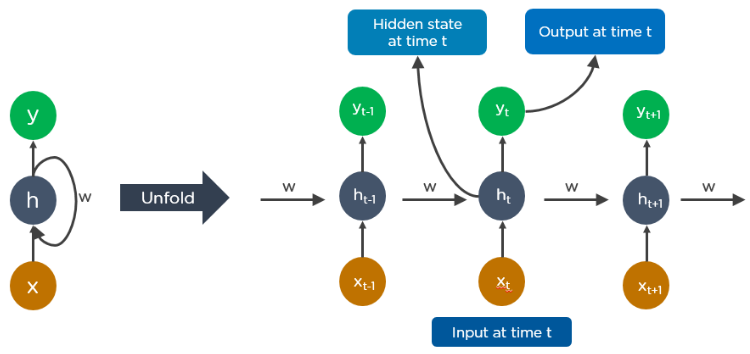
LSTMs retain information over time. They are useful in time-series prediction, LSTMs are typically used for speech recognition, music composition, and pharmaceutical development.

Figure 5 LSTM

* **Recurrent Neural Networks (RNNs)**

RNNs have connections that form directed cycles, which allow the outputs from the LSTM to be fed as inputs to the current phase.

The output from the LSTM becomes an input to the current phase and can memorize previous inputs due to its internal memory. RNNs are commonly used for image captioning, handwriting recognition, and machine translation.

****

**Figure 6 RNN**

Here is an example of how Google’s autocompleting feature works:

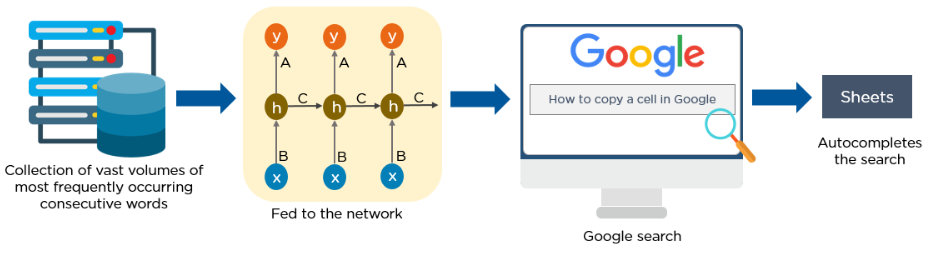


Figure 7 RNN

* **Multilayer Perceptrons (MLPs)**

MLPs are an excellent place to start learning about deep learning technology.

MLPs belong to the class of feedforward neural networks with multiple layers of perceptrons that have activation functions.

MLPs consist of an input layer and an output layer that are fully connected. They have the same number of input and output layers but may have multiple hidden layers and can be used to build speech-recognition, image-recognition, and machine-translation software.

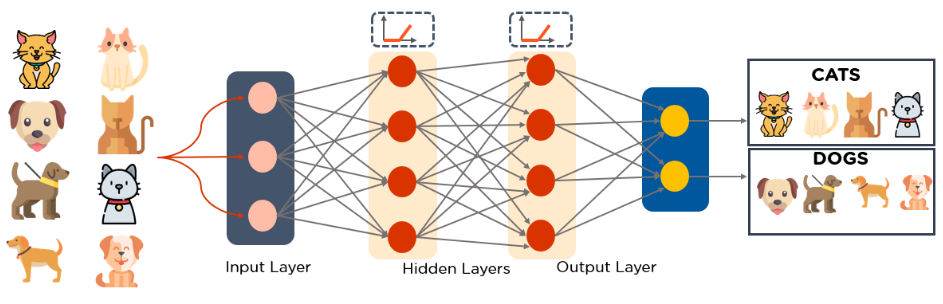
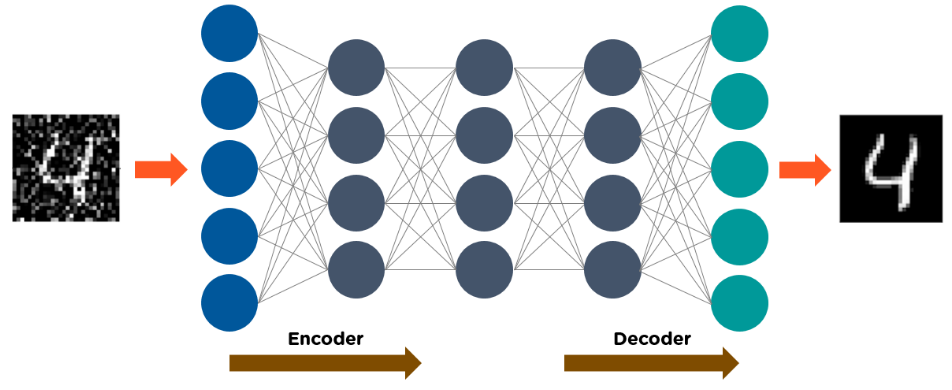


Figure 8 MLPS

* **Autoencoders**

Autoencoders are a specific type of feedforward neural network in which the input and output are identical. Geoffrey Hinton designed autoencoders in the 1980s to solve unsupervised learning problems. They are trained neural networks that replicate the data from the input layer to the output layer. Autoencoders are used for purposes such as pharmaceutical discovery, popularity prediction, and image processing.

****

**Figure 9 Autoencoders**

**2.2.15 A Brief History of Artificial Intelligence**

Artificial Intelligence is not a new word and not a new technology for researchers. This technology is much older than you would imagine. Even there are the myths of Mechanical men in Ancient Greek and Egyptian Myths. Following are some milestones in the history of AI which defines the journey from the AI generation to till date developme

****Figure 10 History of Ai

**Maturation of Artificial Intelligence (1943-1952)**

* **Year 1943:** The first work which is now recognized as AI was done by Warren McCulloch and Walter pits in 1943. They proposed a model of **artificial neurons**.
* **Year 1949:** Donald Hebb demonstrated an updating rule for modifying the connection strength between neurons. His rule is now called **Hebbian learning**.
* **Year 1950:** Alan Turing who was an English mathematician and pioneered Machine learning in 1950. Alan Turing published "Computing Machinery and Intelligence" in which he proposed a test. The test can check the machine's ability to exhibit intelligent behavior equivalent to human intelligence, called a **Turing test**.

**The birth of Artificial Intelligence (1952-1956)**

* **Year 1955:** Allen Newell and Herbert A. Simon created the "first artificial intelligence program"Which was named as "Logic Theorist". This program has proved 38 of 52 Mathematics theorems, and found new and more elegant proofs for some theorems.
* **Year 1956:** The word "Artificial Intelligence" first adopted by American Computer scientist John McCarthy at the Dartmouth Conference. For the first time, AI was coined as an academic field.

**The golden years-Early enthusiasm (1956-1974)**

* **Year 1966:** The researchers emphasized developing algorithms which can solve mathematical problems. Joseph Weizenbaum created the first chatbot in 1966, which was named ELIZA.
* **Year 1972:** The first intelligent humanoid robot was built in Japan which was named as WABOT-1.

**The first AI winter (1974-1980)**

* The duration between 1974 to 1980 was the first AI winter duration. AI winter refers to the time period where computer scientists dealt with a severe shortage of funding from the government for AI research.
* During AI winters, an interest of publicity on artificial intelligence was decreased.

**A boom of AI (1980-1987)**

* **Year 1980:** After AI winter duration, AI came back with "Expert System". Expert systems were programmed that emulate the decision-making ability of a human expert.
* **In 1980,** the first national conference of the American Association of Artificial Intelligence was held at Stanford University.

**The second AI winter (1987-1993)**

* The duration between the years 1987 to 1993 was the second AI Winter duration.
* Again Investors and government stopped funding for AI research due to high cost but not efficient result. The expert system such as XCON was very cost effective.

**The emergence of intelligent agents (1993-2011)**

* **Year 1997:** In the year 1997, IBM Deep Blue beat world chess champion Gary Kasparov, and became the first computer to beat a world chess champion.
* **Year 2002:** for the first time, AI entered the home in the form of Roomba, a vacuum cleaner.
* **Year 2006:** AI came in the Business world till the year 2006. Companies like Facebook, Twitter, and Netflix also started using AI.

**Deep learning, big data and artificial general intelligence (2011-present)**

* **Year 2011:** In the year 2011, IBM's Watson won jeopardy, a quiz show, where it had to solve complex questions as well as riddles. Watson had proved that it could understand natural language and can solve tricky questions quickly.
* **Year 2012:** Google has launched an Android app feature "Google now", which was able to provide information to the user as a prediction.
* **Year 2014:** In the year 2014, Chatbot "Eugene Goostman" won a competition in the infamous "Turing test."
* **Year 2018:** The "Project Debater" from IBM debated on complex topics with two master debaters and also performed extremely well.
* Google has demonstrated an AI program "Duplex" which was a virtual assistant and which had taken hairdresser appointments on call, and the lady on the other side didn't notice that she was talking with the machine.

**2.2 Computer Vision**

Computer vision is the process of using machines to understand and analyze imagery (both photos and videos). While these types of algorithms have been around in various forms since the 1960’s, recent advances in Machine Learning, as well as leaps forward in data storage, computing capabilities, and cheap high-quality input devices, have driven major improvements in how well our software can explore this kind of content.

Computer vision is the broad parent name for any computations involving visual content – that means images, videos, icons, and anything else with pixels involved. But within this parent idea, there are a few specific tasks that are core building blocks:

In object classification, you train a model on a dataset of specific objects, and the model classifies new objects as belonging to one or more of your training categories.

For object identification, your model will recognize a specific instance of an object – for example, parsing two faces in an image and tagging one as Tom Cruise and one as Katie Holmes.

A classical application of computer vision is handwriting recognition for digitizing handwritten content (we’ll explore more use cases below). Outside of just recognition, other methods of analysis include:

Video motion analysis uses computer vision to estimate the velocity of objects in a video, or the camera itself.

In image segmentation, algorithms partition images into multiple sets of views.

Scene reconstruction creates a 3D model of a scene inputted through images or video.

In image restoration, noise such as blurring is removed from photos using Machine Learning based filters.

Any other application that involves understanding pixels through software can safely be labeled as computer vision.

One of the major open questions in both Neuroscience and Machine Learning is: how exactly do our brains work, and how can we approximate that with our own algorithms? The reality is that there are very few working and comprehensive theories of brain computation; so despite the fact that Neural Nets are supposed to “mimic the way the brain works,” nobody is quite sure if that’s actually true. Jeff Hawkins has an entire book on this topic called On Intelligence.

The same paradox holds true for computer vision – since we’re not decided on how the brain and eyes process images, it’s difficult to say how well the algorithms used in production approximate our own internal mental processes. For example, studies have shown that some functions that we thought happen in the brain of frogs actually take place in the eyes. We’re a far cry from amphibians, but similar uncertainty exists in human cognition.

Machines interpret images very simply: as a series of pixels, each with their own set of color values. Consider the simplified image below, and how grayscale values are converted into a simple array of numbers:

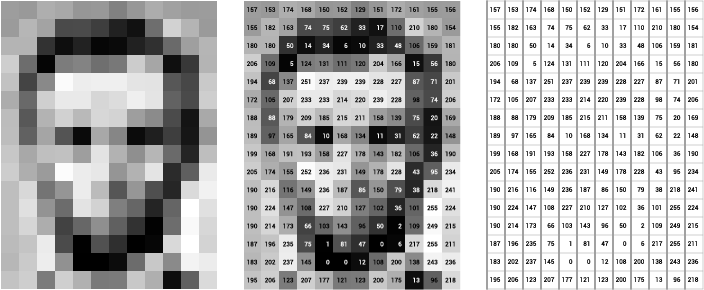


Figure 11 Images

Think of an image as a giant grid of different squares, or pixels (this image is a very simplified version of what looks like either Abraham Lincoln or a Dementor). Each pixel in an image can be represented by a number, usually from 0 – 255. The series of numbers on the right is what software sees when you input an image. For our image, there are 12 columns and 16 rows, which means there are 192 input values for this image.

When we start to add in color, things get more complicated. Computers usually read color as a series of 3 values – red, green, and blue (RGB) – on that same 0 – 255 scale. Now, each pixel actually has 3 values for the computer to store in addition to its position. If we were to colorize President Lincoln (or Harry Potter’s worst fear), that would lead to 12 x 16 x 3 values, or 576 numbers.

For some perspective on how computationally expensive this is, consider this tree:

Each color value is stored in 8 bits.

8 bits x 3 colors per pixel = 24 bits per pixel.

A normal sized 1024 x 768 image x 24 bits per pixel = almost 19M bits, or about 2.36 megabytes.

That’s a lot of memory to require for one image, and a lot of pixels for an algorithm to iterate over. But to train a model with meaningful accuracy – especially when you’re talking about Deep Learning – you’d usually need tens of thousands of images, and the more the merrier. Even if you were to use Transfer Learning to use the insights of an already trained model, you’d still need a few thousand images to train yours on.

With the sheer amount of computing power and storage required just to train deep learning models for computer vision, it’s not hard to understand why advances in those two fields have driven Machine Learning forward to such a degree.

Computer vision and convolutional neural networks

Much of the progress made in computer vision accuracy over the past few years is due in part to a special type of algorithm. Convolutional Neural Networks are a subset of Deep Learning with a few extra added operations, and they’ve been shown to achieve impressive accuracy on image-associated tasks.

Convolutional Neural Networks (CNNs or ConvNets) utilize the same major concepts of Neural Networks, but add in some steps before the normal architecture. These steps are focused on feature extraction, or finding the best version possible of our input that will yield the greatest level of understanding for our model. Ideally, these features will be less redundant and more informative than the original input.

The CNN uses three sorts of filters for feature extraction. For more detail and interactive diagrams, Ujjwal Karn’s walkthrough post on the topic is excellent.

Convolution

During the convolution process (perhaps why it’s called a CNN) the input image pixels are modified by a filter. This is just a matrix (smaller than the original pixel matrix) that we multiply different pieces of the input image by. The output – often called a Feature Map – will usually be smaller than the original image, and theoretically be more informative.

ReLU

This futuristic sounding acronym stands for Rectified Linear Unit, which is an easy function to introduce non-linearity into the feature map. All negative values are simply changed to zero, removing all black from the image. The formal function is y = max(0, x).

Pooling

In pooling, the image is scanned over by a set width of pixels, and either the max, sum, or average of those pixels is taken as a representation of that portion of the image. This process further reduces the size of the feature map(s) by a factor of whatever size is pooled.

**2.2.1 Computer Vision Hierarchy**

Computer vision is divided into three basic categories as follows:

* **Low-level vision:** It includes process images for feature extraction.
* **Intermediate-level vision:** It includes object recognition and 3D scene interpretation
* **High-level vision:** It includes conceptual description of a scene like activity, intention and behavior.

**2.2.2 Computer Vision Vs Image Processing**

Image processing studies image to image transformation. The input and output of image processing are both images. Computer vision is the construction of explicit, meaningful descriptions of physical objects from their image. The output of computer vision is a description or an interpretation of structures in a 3D scene.

**2.2.3 Computer vision Applications**

**Robotics**

* Localization-determine robot location automatically
* Navigation
* Obstacles avoidance
* Assembly (peg-in-hole, welding, painting)
* Manipulation (e.g. PUMA robot manipulator)
* Human Robot Interaction (HRI): Intelligent robotics to interact with and serve people

**Medicine**

* Classification and detection (e.g. lesion or cells classification and tumor detection)
* 2D/3D segmentation
* 3D human organ reconstruction (MRI or ultrasound)
* Vision-guided robotics surgery

**Security**

* Biometrics (iris, finger print, face recognition)
* Surveillance-detecting certain suspicious activities or behaviors

**Transportation**

* Autonomous vehicle Safety, e.g., driver vigilance
* monitoring

**Industrial Automation Application**

* Industrial inspection (defect detection)
* Assembly
* Barcode and package label reading
* Object sorting
* Document understanding (e.g. OCR)

**Chapter3 :SYSTEM REQUIREMENT SPECIFICATION**

**3.1 Hardware Requirement**

A desktop with 3.4GHz CPU, 16G memory, 32-bit or 64-bit processor, a running windows operating system. Basic hardware like monitor, mouse and keyboard are a must for input and output. A CD-ROM or a USB port to install the necessary softwares. An additional camera will be required for real-time video processing.

**3.2 Software Requirement**

Python with the version 3.9.x with the following modules installed, keras, numpy, os, time, tensorflow, pickle. Windows operating system. Camera modules installed for importing video files from camera.

**3.3 Non-Functional Requirements**

**3.3.1 Usability**

The system is easy to train and test thus navigates in the most expected way with less delay. Since the algorithm is written in such a way that a lot of parallel computation can be performed, extensively high frame rates can be achieved by inculcating multi-threading. Users will be allowed to add his videos easily and direct testing can be started on those videos. Thus it is user friendly, reducing complexity on users and getting them a better result in a faster way.

**3.3.2 Assumption**

Test videos are either in .mp4, .mkv, .web formats. Each video is taken to represent a lot of frames so a 200 second video is assumed to generate 200 frames or more. The minimum learning rate for training is chosen to be 0.00001 and the maximum is 0.0001 . Batch size is set to 16.

**3.3.3 Performance**

Pre-captured video frames will be analyzed fast as a significant amount of time is saved in not capturing the video. Detection of abnormality in the frames is done at a rate of 150fps using parallel processing of data in the testing phase. An accuracy of 90% is achieved with the sparse combination method implemented.

**3.3.4 Reliability**

The software is tested with various datasets mentioned in this report and the output is very close to the actual abnormal scenarios thus turning out to be reliable when all the assumptions mentioned are considered.

**3.4 Functional Requirements**

The system should process videos in the chosen video formats. (mp4, mkv, web) The system shall detect fight videos and display the exact time from the start at which the fight video is detected. The length of the test video can vary and no particular limit is imposed. In the presence of violence, the user should be displayed with a message that the frame is violence. After the whole frames are processed the user should see the portion of the frame where violence event was detected. The video is displayed and through it is clarified that there is violence or not .

**3.5 Operating Environment**

**Python** - The language used here is a popular programming language called Python which due to its huge number of pre-built libraries, is quite suitable for doing Machine Learning and Deep Learning projects. Python has libraries like Tensorflow and Open-CV which we have used in our project. Due to its easy to learn syntax, it is used world-wide for different purposes like web development etc..

**3.5.1 Why Python?**

* **Simple syntax & less coding**

Python involves very less coding and simple syntax among other programming languages which can be used for developing AI applications. Due to this feature, the testing can be easier and we can focus more on programming.

* **Inbuilt libraries for AI projects**

A major advantage for using Python for AI is that it comes with inbuilt libraries. Python has libraries for almost all kinds of AI projects. For example, **NumPy, SciPy, matplotlib, nltk, SimpleAI** are some of the important inbuilt libraries of Python.

* **Open source:** Python is an open source programming language. This makes it widely popular in the community.
* **Can be used for a broad range of programming:** Python can be used for a broad range of programming tasks like small shell script to enterprise web applications. This is another reason Python is suitable for AI projects.

**3.5.2 Features of Python**

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English words frequently whereas other languages use punctuation, and it has fewer syntactic constructions than other languages. Python's features include the following :

* **Easy-to-learn :**Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read :** Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
* **A broad standard library :** Python's bulk of the library is very portable and cross platform compatible on UNIX, Windows, and Macintosh
* **Interactive Mode:** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable :** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable:** We can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases:** Python provides interfaces to all major commercial databases.
* **GUI Programming:** Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

* **Scalable:** Python provides a better structure and support for large programs than shell scripting.

**3.6 Mathematics Libraries**

There are several very popular math libraries for Python that supply more advanced math functions than the standard built-in Python Math Library.

**3.6.1 NumPy**

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

There are several important differences between NumPy arrays and the standard Python sequences:

* NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.
* The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.
* NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.
* A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays.
* In other words, in order to efficiently use much (perhaps even most) of today’s scientific/mathematical Python-based software, just knowing how to use Python’s built-in sequence types is insufficient

**Why is NumPy Fast?**

Vectorization describes the absence of any explicit looping, indexing, etc., in the code - these things are taking place, of course, just “behind the scenes” in optimized, pre-compiled C code. Vectorized code has many advantages, among which are:

* vectorized code is more concise and easier to read
* fewer lines of code generally means fewer bugs
* the code more closely resembles standard mathematical notation (making it easier, typically, to correctly code mathematical constructs)
* vectorization results in more “Pythonic” code. Without vectorization, our code would be littered with inefficient and difficult to read for loops.

**3.6.2 Matplotlib**

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

Matplotlib is a collection of functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.

**3.6.3 Seaborn**

Seaborn is an open source  Python library providing high level API for visualizing the data using Python programming language.

Functions in the seaborn library expose a declarative, dataset-oriented API that makes it easy to translate questions about data into graphics that can answer them. When given a dataset and a specification of the plot to make, seaborn automatically maps the data values to visual attributes such as color, size, or style, internally computes statistical transformations, and decorates the plot with informative axis labels and a legend.

Many seaborn functions can generate figures with multiple panels that elicit comparisons between conditional subsets of data or across different pairings of variables in a dataset.

Seaborn is designed to be useful throughout the lifecycle of a scientific project. By producing complete graphics from a single function call with minimal arguments, seaborn facilitates rapid prototyping and exploratory data analysis.

**3.6.4 OpenCV**

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library.

The library has more than 2500 optimized algorithms,These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc..

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available.

A full-featured CUDA and OpenCL interfaces are being actively developed right now.

There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms.

OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

**3.6.5 Keras**

Keras is an Open Source Neural Network library written in Python that runs on top of Tensorflow. It is designed to be modular, fast and easy to use. It was developed by François Chollet, a Google engineer.

Keras High-Level API handles the way we make models, defining layers, or set up multiple input-output models. In this level, Keras also compiles our model with loss and optimizer functions, training process with fit function. Keras in Python doesn’t handle Low-Level API such as making the computational graph, making tensors or other variables

Keras is:

* **Simple** -- but not simplistic. Keras reduces developer cognitive load to free you to focus on the parts of the problem that really matter.
* **Flexible** -- Keras adopts the principle of progressive disclosure of complexity: simple workflows should be quick and easy, while arbitrarily advanced workflows should be possible via a clear path that builds upon what you've already learned.
* **Powerful** -- Keras provides industry-strength performance and scalability: it is used by organizations and companies including NASA, YouTube, or Waymo.

Keras empowers engineers and researchers to take full advantage of the scalability and cross-platform capabilities of TensorFlow 2: you can run Keras on TPU or on large clusters of GPUs, and you can export your Keras models to run in the browser or on a mobile device.

Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code.

The code is hosted on Github, and community support forums include the GitHub issues page, and a Slack channel.

In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch,normalization, and pooling..

**3.6.6 TensorFlow**

Developed by Google, TensorFlow is now an open-source library for computations and graphs and serves as a framework to develop neural networks and deep learning.TensorFlow is an end-to-end platform for machine learning. It supports Multidimensional-array based numeric computation ,GPU and distributed processing,Automatic differentiation,Model construction,training, and export,etc..

**Why TensorFlow?**

* **Easy model building**

TensorFlow offers multiple levels of abstraction so you can choose the right one for your needs. Build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy.

If you need more flexibility, eager execution allows for immediate iteration and intuitive debugging. For large ML training tasks, use the Distribution Strategy API for distributed training on different hardware configurations without changing the model definition.

* **Robust ML production anywhere**

TensorFlow has always provided a direct path to production. Whether it's on servers, edge devices, or the web, TensorFlow lets you train and deploy your model easily, no matter what language or platform you use.

Use TensorFlow Extended (TFX) if you need a full production ML pipeline. For running inference on mobile and edge devices, use TensorFlow Lite. Train and deploy models in JavaScript environments using **TensorFlow.js.**

* **Powerful experimentation for research**

Build and train state-of-the-art models without sacrificing speed or performance. TensorFlow gives you the flexibility and control with features like the Keras Functional API and Model Subclassing API for creation of complex topologies. For easy prototyping and fast debugging, use eager execution.

TensorFlow also supports an ecosystem of powerful add-on libraries and models to experiment with, including Ragged Tensors, TensorFlow Probability, Tensor2Tensor and BERT.

**3.6.7 Sklearn**

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. This library, which is largely written in Python, is built upon**NumPy, SciPy and Matplotlib**.

**3.6.8 Time**

This module provides various time-related functions  like waiting during code execution and measuring the efficiency of your code.

**3.6.9 Collections**

This module implements specialized container data types providing alternatives to Python’s general purpose built-in containers, dict,list,set and tuple

**3.6.10 Tqdm**

Instantly make your loops show a smart progress meter - just wrap any iterable with tqdm.In addition,tqdm uses smart algorithms to predict the remaining time and to skip unnecessary iteration displays, which allows for a negligible overhead in most cases.It can also be executed as a module with pipes.

**Overhead is low :**about 60ns per iteration and is unit tested against performance regression. By comparison, the well-established Progress Bar has an 800ns/iter overhead.

In addition to its low overhead, tqdm uses smart algorithms to predict the remaining time and to skip unnecessary iteration displays, which allows for a negligible overhead in most cases.

tqdm works on any platform (Linux, Windows, Mac, FreeBSD, NetBSD, Solaris/SunOS), in any console or in a GUI, and is also friendly with IPython/Jupyter notebooks.

tqdm does not require any dependencies , just Python and an environment supporting carriage return \r and line feed \n control characters.

**3.6.11 Pickle**

Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script.

**Advantages of using Pickle Module:**

* Recursive objects (objects containing references to themselves): Pickle keeps track of the objects it has already serialized, so later references to the same object won’t be serialized again. (The marshal module breaks for this.)
* Object sharing (references to the same object in different places): This is similar to self- referencing objects; pickle stores the object once, and ensures that all other references point to the master copy. Shared objects remain shared, which can be very important for mutable objects.
* User-defined classes and their instances: Marshal does not support these at all, but pickle can save and restore class instances transparently. The class definition must be importable and live in the same module as when the object was stored.

**3.6.12 imageio**

Imageio is a Python library that provides an easy interface to read and write a wide range of image data, including animated images, volumetric data, and scientific formats. It is cross-platform, runs on Python 3.5+, and is easy to install.

**3.6.13 imgaug**

imgaug is a library for image augmentation in machine learning experiments. It supports a wide range of augmentation techniques, allows to easily combine these and to execute them in random order or on multiple CPU cores, has a simple yet powerful stochastic interface and can not only augment images, but also keypoints/landmarks, bounding boxes, heatmaps and segmentation maps.

**Chapter 4: Implementation**

**4.1 Method**

The method described here is based on the principle that when a fight occurs in the Surveillance Videos, the fighting frames of video will be significantly different than the Non-fighting frames. We train a model that consists of a feature extractor and learns the temporal patterns of the input volume of frames. The model is trained with video volumes consisting of normal and fighting scenes. Footage from the surveillance camera is broken down into frames. The frames are given as input to MobileNet v2 classifier for detecting violent activities in the given sequence of input frames. If no violent activity is recognized the respective frames are discarded. The violence detected frame is obtained and it is enhanced for better clarity.

Figure 12 Method

**4.2 Dataset**

The dataset contains 3500 video clips which belong to two classes, violence and non-violence respectively. The average duration of the video clips is 5 seconds and majority of those videos are from CCTV footage. For training, 350 videos each from the violent and non-violent classes are taken at each epoch.

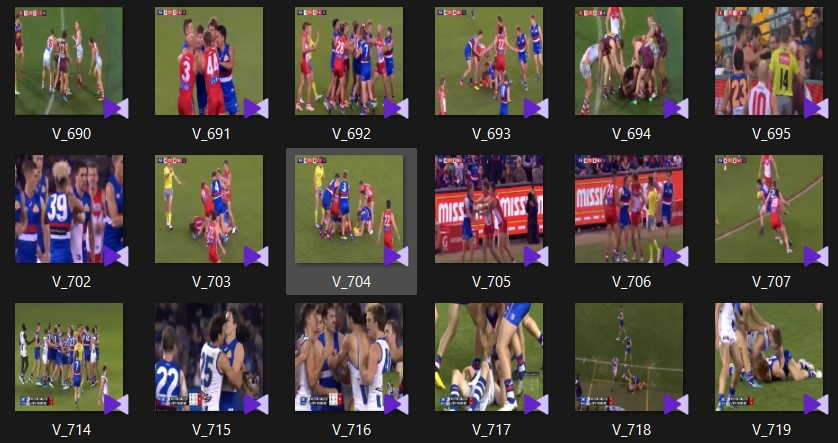


Figure 13 Dataset

To test our methodology, we work with these three datasets,Hockey Fight Dataset , Movies Dataset and ViolentFlows . the 3 datasets captured from closed- circuited-TV,Phone or high-resolution recorder, the quality, number of pixels and length varies between dataset.

**4.2.1 Hockey fights**

Dataset composed of equal numbers of violence and nonviolence action during hockey professional matches, usually Two players participating in close body interaction.

**4.2.2 Movies**

This dataset consists of fight sequences collected from movies, for the non-violence label - videos of general action activity gathered from movies. The dataset is made up of an equal number of violent movie clips and non-violent movie clips. Unlike the Hockey dataset, this dataset varies profoundly between samples.

**4.2.3 Real Life Violence Dataset**

This is a crowd violence dataset. Most of the crowd violence seen in this dataset are random clips

**4.3 Pre-Processing**

The pre-processing step includes importing the video frames and making it ready for training. It also involves feature extraction which is the input to the training algorithm. Initially the video is converted to frames. Each video will generate a set of frames which approximately denotes the number of seconds. This process of converting video to frame using python libraries is illustrated in the Figure .

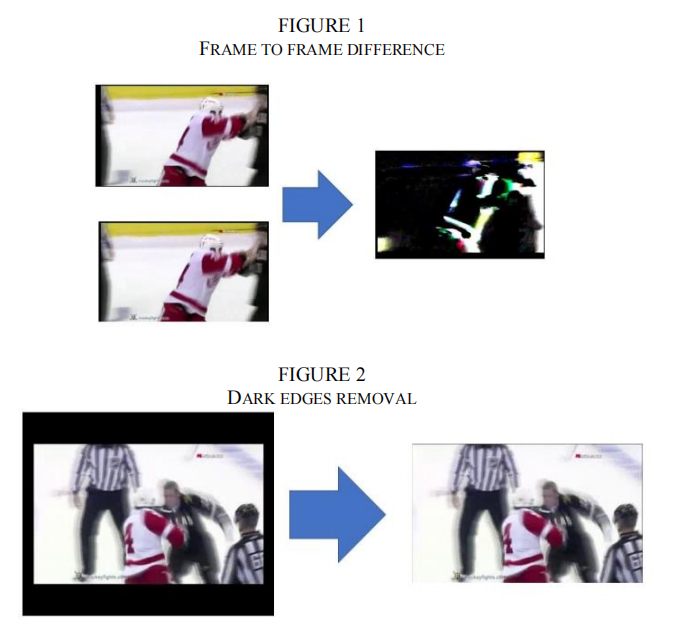


****

Figure 14 cv2

The task of this stage is to convert raw data to an acceptable

input for the model. Each frame is extracted from the raw videos are modified by rotating , flipping , zooming and random brightness

****

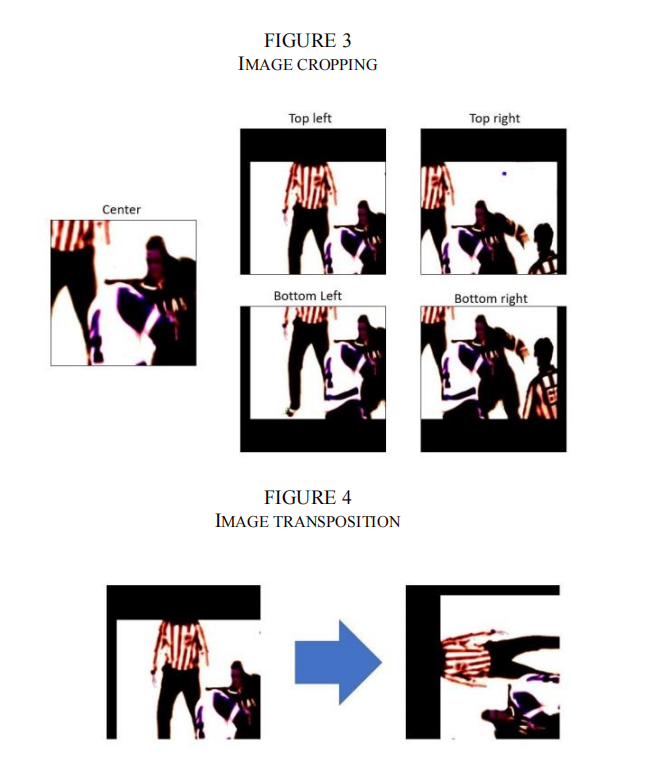
****

Figure 15 Image Processing

and resized to 128\*128. To ensure that the input images are all on the same scale, the pixel values are scaled between 0 and 1 for normalization.

Now the input is ready for model training.

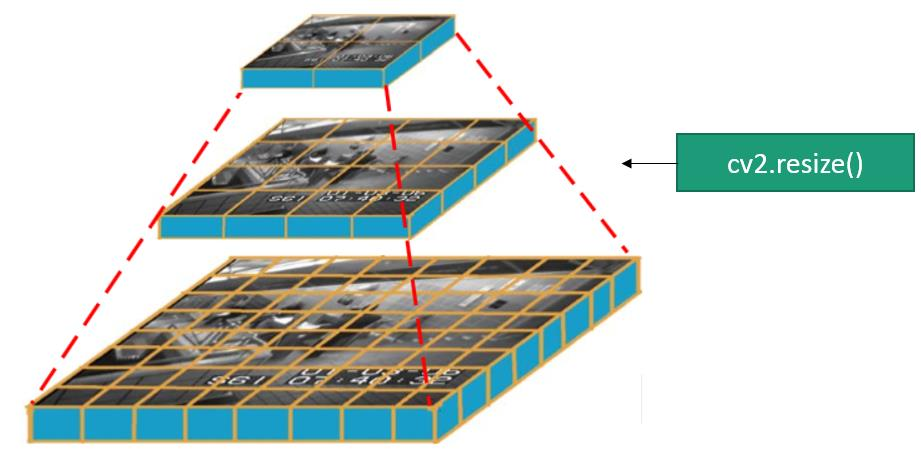


Figure 16 cv2.resize()

**4.4 Training**

The features extracted in the data pre-processing step are used as an initial input to the training algorithm. We will use MobileNet v2 to make our model , Adam optimization algorithm and Binary\_Crossentropy loss .

We use One Cycle Learning Rate with initial learning rate of 0.00001, with maximum learning rate of .0001 and learning rate decay rate of 0.8 per epoch . We use a batch size of 16.

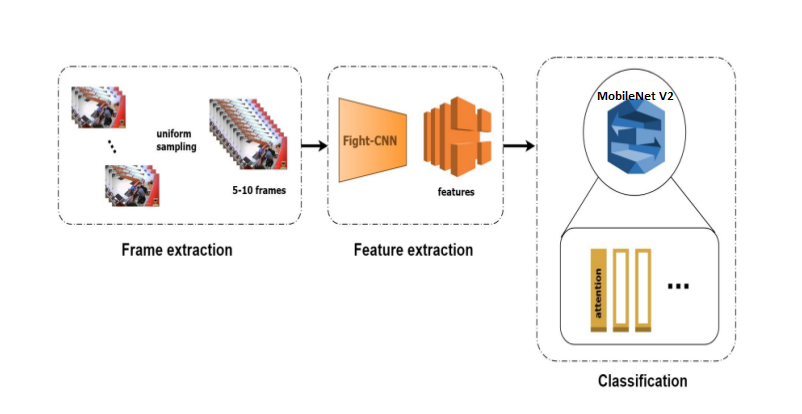


Figure 17 Training

**4.4.1 MobileNet v2 :**

MobileNet v2 is a convolutional neural network architecture . It is based on an inverted residual structure where the residual connections are between the bottleneck layers .

Since the images can be seen as a matrix of pixels and each pixel describes some features of the image, these technologies use filters to filter out a certain set of pixels in the images and results in the formation of output predictions about images.

CNN uses a lot of pre-defined and stored filters and does a convolution (X) of that filter with the pixel matrix of the image. This results in filtering the image’s objects and comparing them with a large set of predefined objects to identify a match between them. Hence in this way these models are able to predict the image.

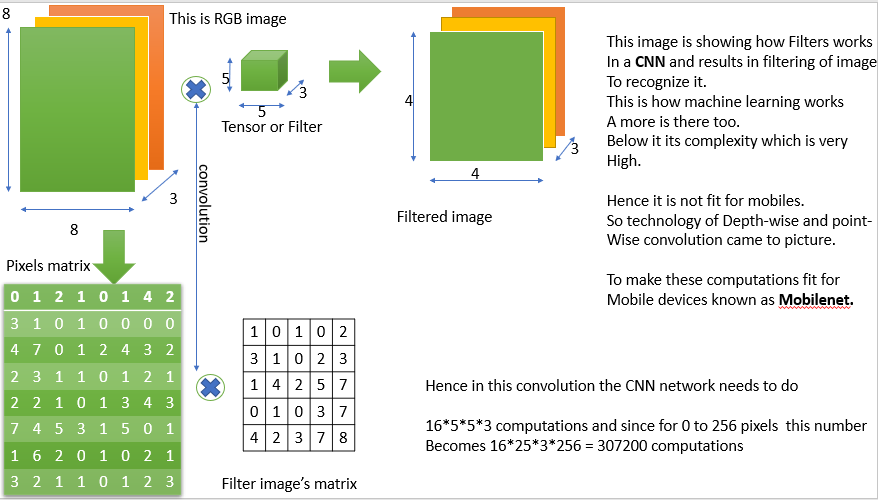


Figure 18 mobilenetv2 (1)

But these technologies require a high GPU to increase the comparison rate between millions of data which cannot be provided by any mobile device.Hence, here comes in action what is known as MobileNet.

Mobilenet is a model which does the same convolution as done by CNN to filter images but in a different way than those done by the previous CNN. It uses the idea of Depth convolution and point convolution which is different from the normal convolution as done by normal CNNs. This increases the efficiency of CNN to predict images and hence they can be able to compete in the mobile systems as well. Since these ways of convolution reduce the comparison and recognition time a lot, it provides a better response in a very short time and hence we are using them as our image recognition model.

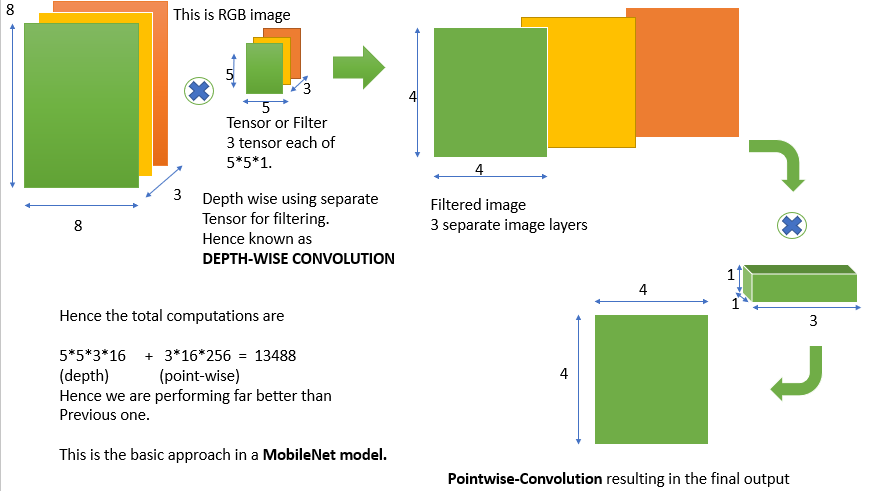


Figure 19 mobilenetv2 (2)

The main strategies introduced in MobileNetV2 were linear bottleneck and inverted residual blocks. In the linear bottleneck layer, the channel dimension of input is expanded to reduce the risk of information loss by nonlinear functions such as ReLU. It comes from the fact that information lost in some channels might be preserved in other channels. The inverted residual block has a ("narrow" -"wide"-""narrow") structure in the channel dimension.





Figure 20 Training Code

**4.4.2 Optimization for training**

Optimizers are algorithms or methods used to change the attributes of your neural network such as weights and learning rate in order to reduce the losses

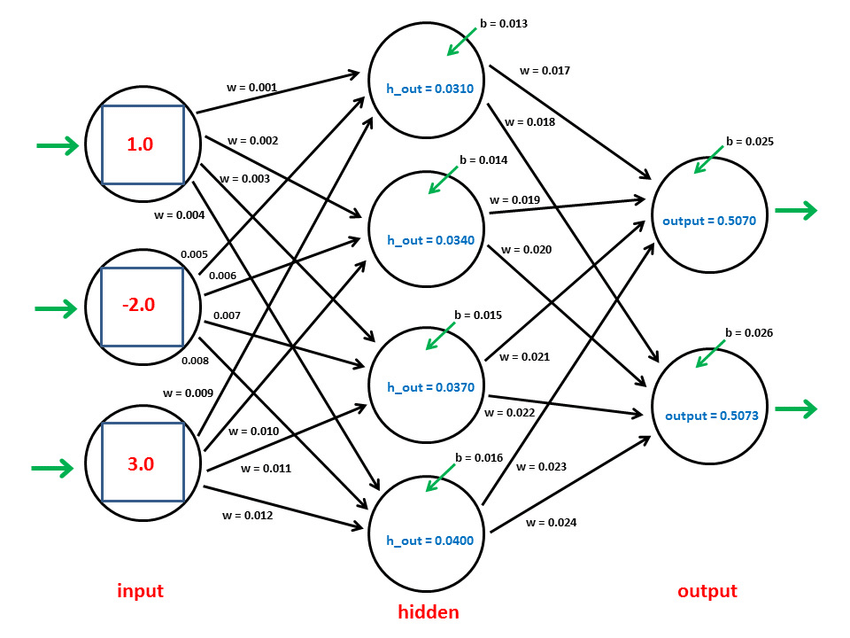


Figure 21 adam

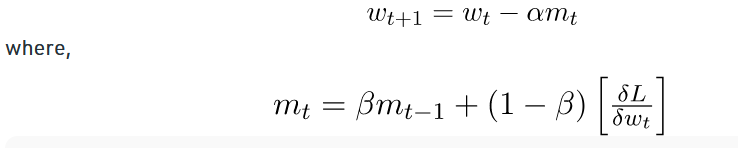
and We use **the “Adam” Optimizer :**

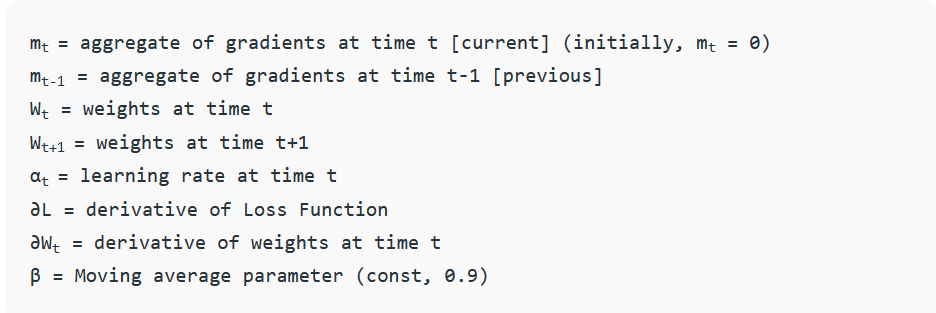
Adaptive Moment Estimation is an algorithm for optimization technique for gradient descent. The method is really efficient when working with large problems involving a lot of data or parameters. It requires less memory and is efficient. Intuitively, it is a combination of the ‘gradient descent with momentum’ algorithm and the ‘RMSP’ algorithm.

Adam optimizer involves a combination of two gradient descent methodologies:

**Momentum:**

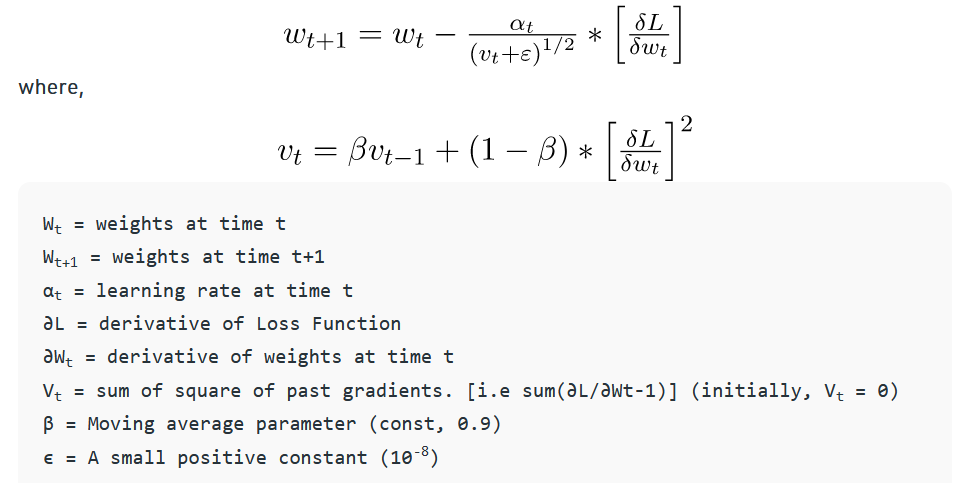
This algorithm is used to accelerate the gradient descent algorithm by taking into consideration the ‘exponentially weighted average’ of the gradients.



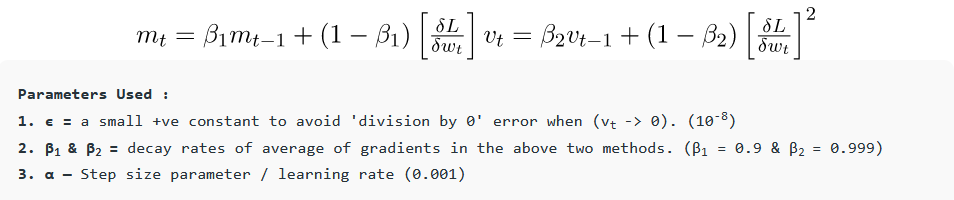


**Root Mean Square Propagation (RMSP):**

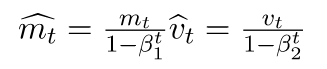
Root mean square prop or RMSprop is an adaptive learning algorithm that tries to improve performance on problems with gradients (e.g. natural language and computer vision problems).



**Mathematical aspect of Adam optimizer :**

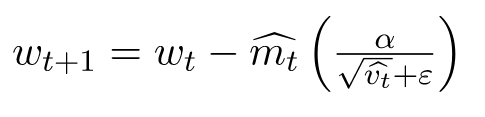
Taking the formulas used in the above two methods, we get  
  


Since mt and vt have both initialized as 0 (based on the above methods), it is observed that they gain a tendency to be ‘biased towards 0’ as both β1 & β2 ≈ 1. This Optimizer fixes this problem by computing ‘bias-corrected’ mt and vt. This is also done to control the weights while reaching the global minimum to prevent high oscillations when near it. The formulas used are:



We are adapting to the gradient descent after every iteration so that it remains controlled and unbiased throughout the process, hence the name Adam.

Now, instead of our normal weight parameters mt and vt , we take the bias-corrected weight parameters (m\_hat)t and (v\_hat)t. Putting them into our general equation, we get



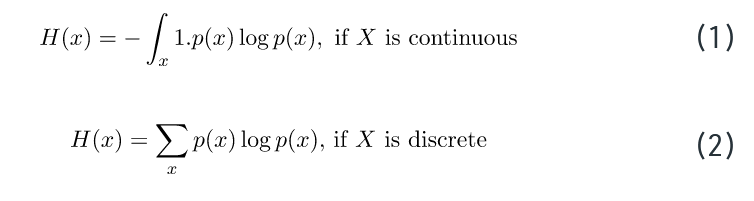
**Binary Crossentropy loss :**

In machine learning lingo, a ‘cost function‘ is used to evaluate the performance of a model.The cost function quantifies the difference between the actual value and the predicted value and stores it as a single-valued real number. The cost function can analogously be called the ‘loss function‘ and we are using here The Cross-Entropy Cost Function

**The idea behind Shannon entropies :**

The Entropy of a random variable X can be measured as the uncertainty in the variables’ possible outcomes. This means the more the certainty/probability, the lesser is the entropy.

The formula to calculate the entropy can be represented as:



**Binary cross-entropy cost function:**

In Binary cross-entropy, there is only one possible output. This output can have discrete values, either 0 or 1. For example, let an input of a particular fruit’s image be either that of an apple or that of an orange. Now, let us rewrite this sentence: A fruit is either an apple, or it is not an apple. There are only binary, true-false outputs possible.

Let us assume that the actual output is represented as a variable y:  
  


**Activation function :**

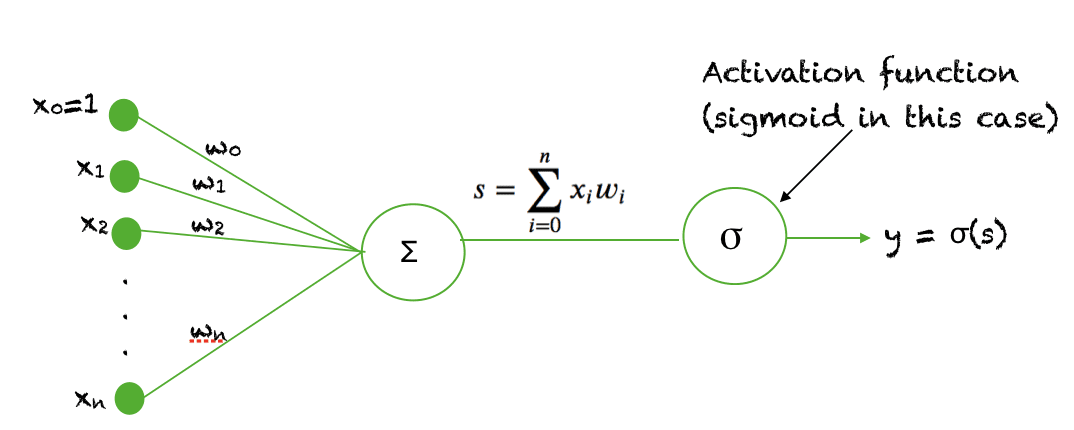
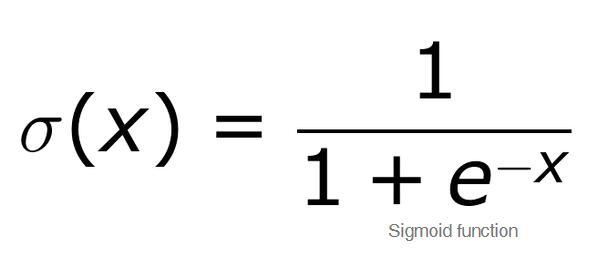


Figure 22 activation function

We use Sigmoid Function and it is a special form of the logistic function and is usually denoted by



When the activation function for a neuron is a sigmoid function it is a guarantee that the output of this unit will always be between 0 and 1. Also, as the sigmoid is a nonlinear function, the output of this unit would be a nonlinear function of the weighted sum of inputs.

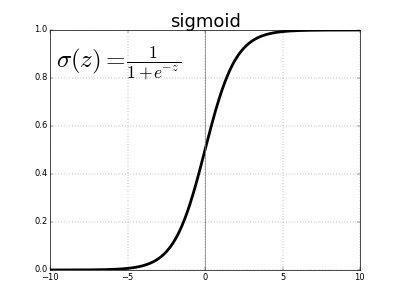


Figure 23 sigmoid function

**Chapter 5: RESULTS AND DISCUSSIONS**

In this section testing and training accuracy are displayed in the below given graphical representation.

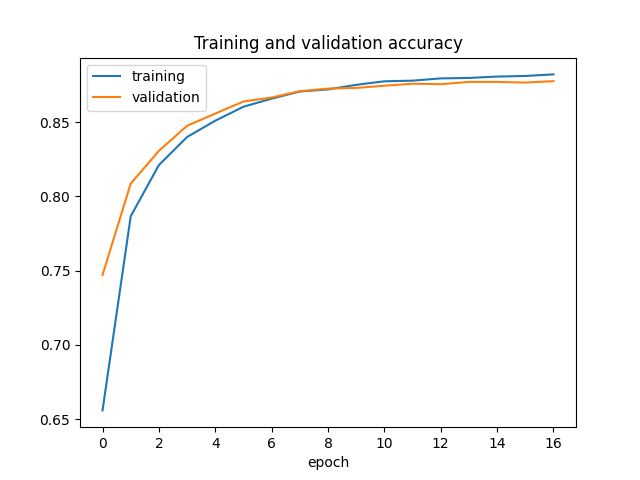


Figure 24 Training and Validation ACC

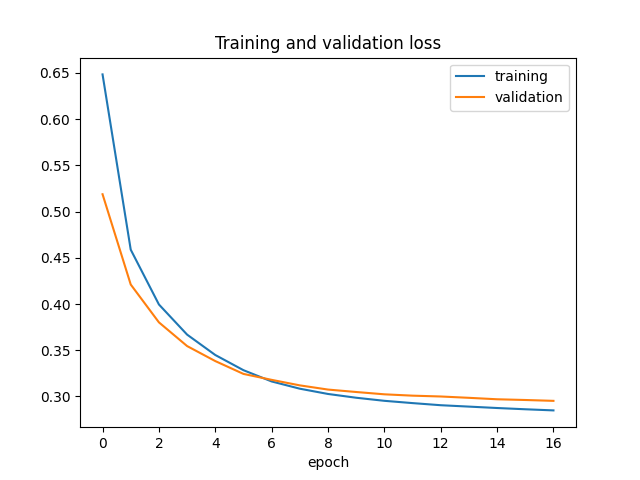


Figure 25 Training and Validation Loss

displays the training and testing accuracy and loss for the

MobileNet v2 model when a dataset containing 3500 videos of average duration 5 seconds is given as input. For each epoch 1750 videos from the fight class and 1750 videos from the non-fight are trained , the accuracy and loss comes to a constant level of increment and decrement after approximately 6 epochs. 90% accuracy was obtained on training.

For the output : A video with violence is given as input to the system. It shows one frame in the video that was labeled to have violent activity. Another video clip without violent activity was given as input. It shows one frame of that video which is rightly labeled as false or non-violence.



Figure 26 Output frame that recognized violence



Figure 27 Output frame that did not recognize violence

**Chapter 6 : CONCLUSION**

Violence scene detection in real-time is a challenging problem due to the diverse content and large variations in quality. In this research, we use the MobileNet v2 model to offer an innovative and efficient technique for identifying violent events in real-time

surveillance footage.

The proposed network has a good recognition accuracy in typical benchmark datasets, indicating that it can learn discriminative motion saliency maps successfully. It’s also computationally efficient, making it ideal for use on low-end devices.

**6.1 Future scope**

This model could be upgraded to work in multiple cameras connected by a single network in a concurrent fashion. A short video of the violent activity could be incorporated along with the alert message.

**REFERENCES**

[1] A. Datta, M. Shah and N. Da Vitoria Lobo, "Person-on-person violence detection in video data," Object recognition supported by user interaction for service robots, 2002, pp. 433-438 vol.1, doi: 10.1109/ICPR.2002.1044748.

[2] Yu Zhao and Rennong Yang and Guillaume Chevalier and Maoguo Gong (2017). Deep Residual Bidir-LSTM for Human Activity Recognition Using Wearable Sensors. CoRR, abs/1708.08989.

[3] Sudhakaran, Swathikiran, and Oswald Lanz. "Learning to detect violent videos using convolution long short-term memory." IAdvanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on, pp. 1-6. IEEE, 2017.

[4] Nievas, Enrique Bermejo and Suarez, Oscar Deniz and Garcia, Gloria Bueno and Sukthankar, Rahul, “Hockey Fight Detection Dataset”, 2016, hosted on bittorrent.

[5] T. Hassner, Y. Itcher, and O. Kliper-Gross, Violent Flows: Real-Time Detection of Violent Crowd Behavior, 3rd IEEE International Workshop on Socially Intelligent Surveillance and Monitoring (SISM) at the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Rhode Island, June 2012 .

[6] Nievas, E., Suarez, O., Garcia, G., & Sukthankar, R. (2011). Movies Fight Detection Dataset. In Computer Analysis of Images and Patterns (pp. 332–339).

[7] E. Acar, F. Hopfgartner and S. Albayrak, "Breaking down violence detection: Combining divide-et-impera and coarse-to-fine strategies", Neurocomputing, vol. 208, pp. 225-237, 2016.

[8] L.-H. Chen, H.-W. Hsu, L.-Y. Wang and C.-W. Su, "Violence detection in movies", International Conference on Computer Graphics Imaging and Visualization (CGIV), 2011.

[9] T. Giannakopoulos, A. Pikrakis and S. Theodoridis, "A multi-class audio classification method with respect to violent content in movies using bayesian networks", IEEE Workshop on Multimedia Signal Processing (MMSP), 2007.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun (2015). Deep Residual Learning for Image Recognition. arXiv preprint arXiv:1512.03385.

[11] Christian Szegedy and Vincent Vanhoucke and Sergey Ioffe and Jonathon Shlens and Zbigniew Wojna (2015). Rethinking the Inception Architecture for Computer Vision. CoRR, abs/1512.00567.

[12] Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.

[13] Zhang, T., Yang, Z., Jia, W. et al. A new method for violence detection in surveillance scenes. Multimed Tools Appl 75, 7327–7349 (2016).

[14] Abdarahmane Traoré, Moulay A. Akhloufi, "Violence Detection in Videos using Deep Recurrent and Convolutional Neural Networks", Systems Man and Cybernetics (SMC) 2020 IEEE International Conference on, pp. 154-159, 2020.

[15] Tal Hassner, Yossi Itcher, and Orit Kliper-Gross. Violent flows: Real Time detection of violent crowd behavior. In Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on, pages 1–6, IEEE, 2012.

**APPENDIX**

**Image Processing**

* Getting frames from videos
* Some image augmentations









**Split data in train/test sets**



**Model Training**





**CALLBACKS**

****



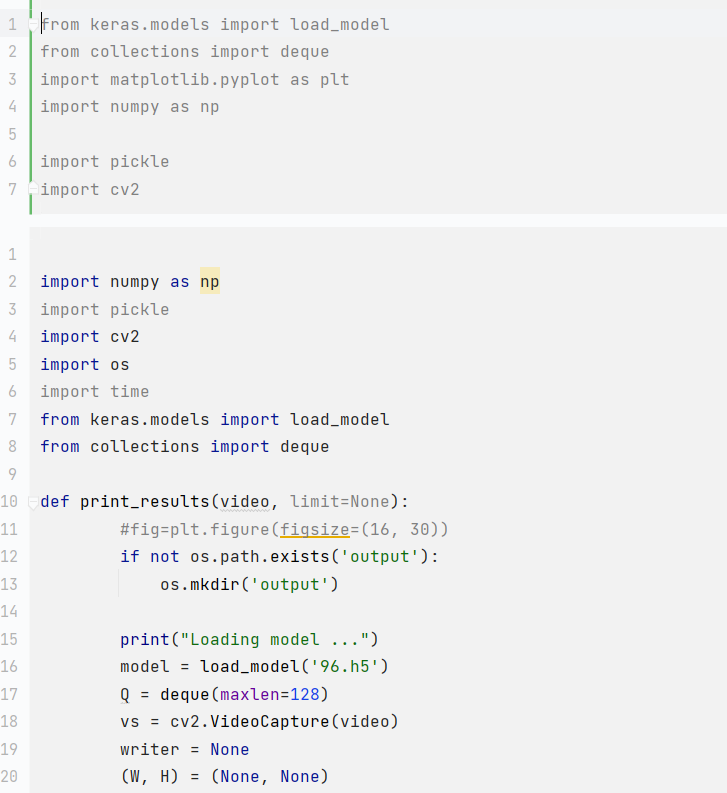
**Model Evaluation and Virtualization**

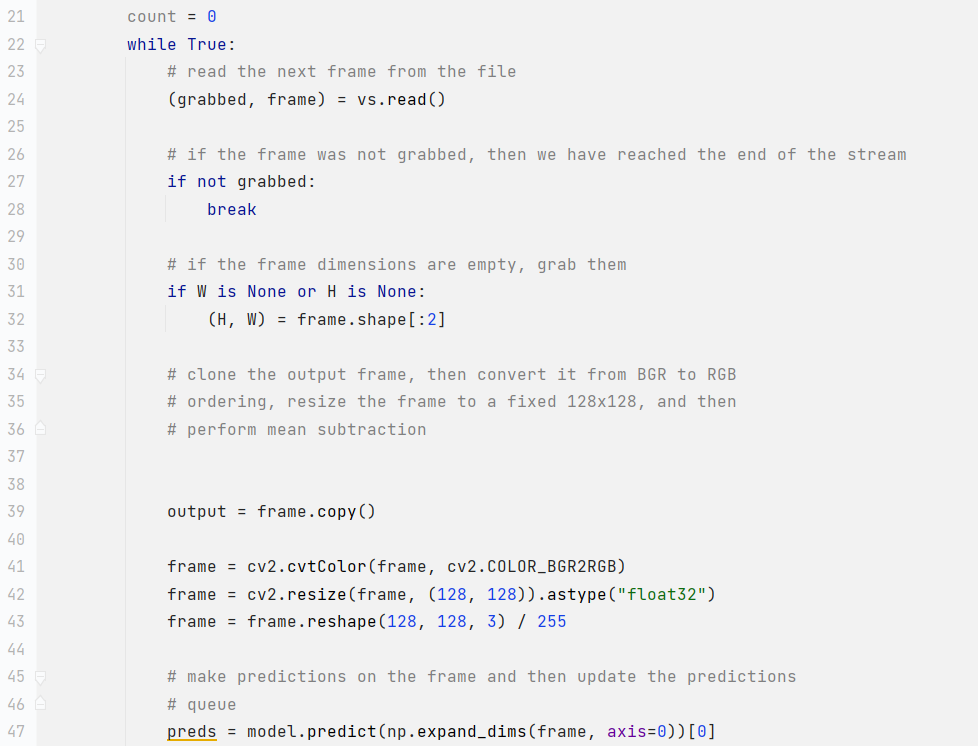


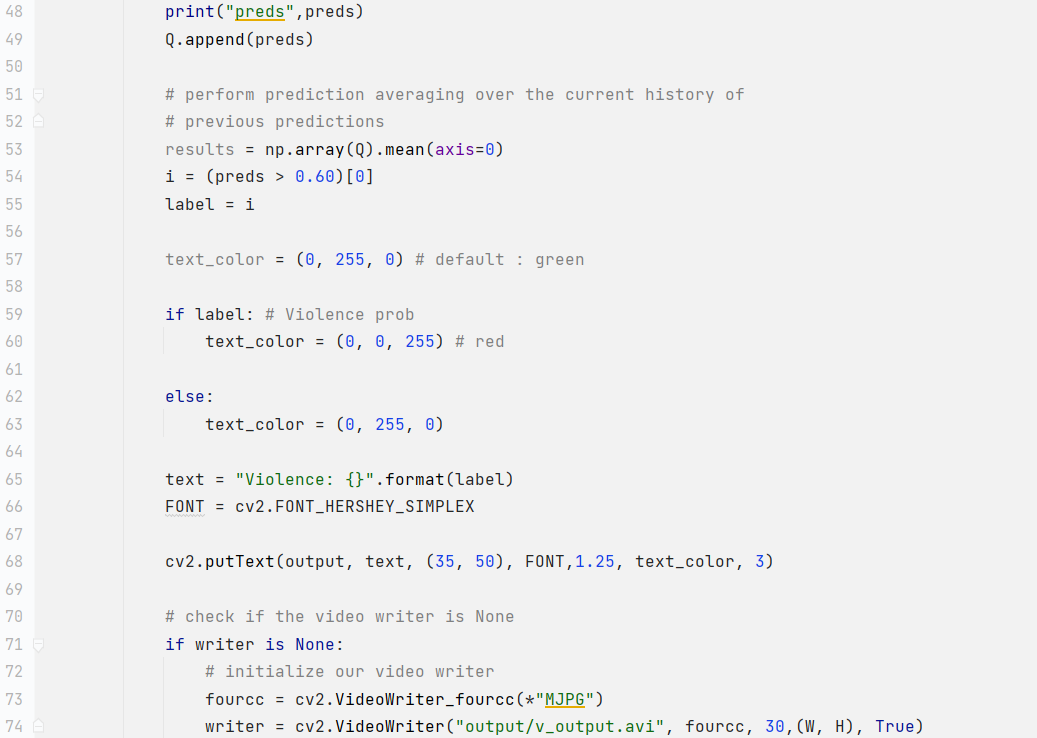
**Evaluation on Test Set**

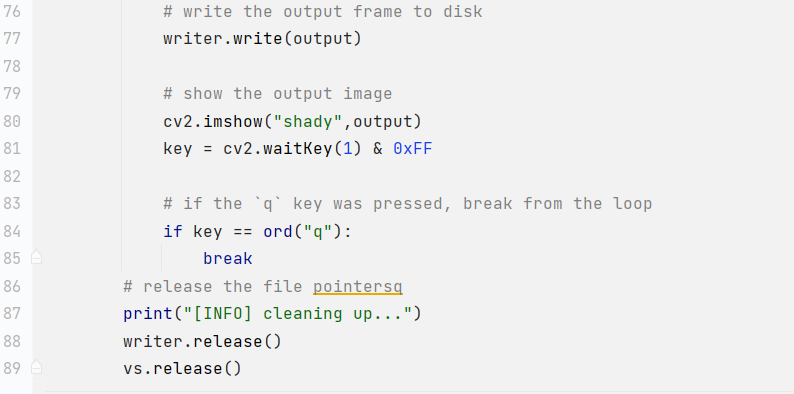




****







**Printing Results**

