

# Eve-Teasing Detection from Video Footage using Computer Vision and Artificial Intelligence

by

Abdullah Rapheo

18101012

A.T.M. Masum Billah

18101008

Lamisha Islam

18101003

Abu Shale MD Yahia Mahim

18101014

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

School of Data and Sciences

Brac University

January 2023

© 2023. Brac University  
All rights reserved.

# Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing the 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 that 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:



---

Abdullah Rapheo

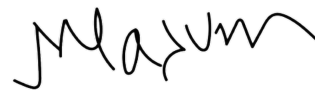
18101012



---

Lamisha islam

18101003



---

A.T.M. Masum Billah

18101008



---

Abu Shale MD Yahia Mahim

18101014

# Approval

The thesis/project titled “Eve-Teasing Detection from Video Footage using Computer Vision and Artificial Intelligence” submitted by

1. Abdullah Rapheo (18101012)
2. A.T.M. Masum Billah (18101008)
3. Lamisha Islam (18101003)
4. Abu Shale MD Yahia Mahim (18101014)

Of Fall, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering on January 17, 2023.

## Examining Committee:

Supervisor:  
(Member)



---

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

Program Coordinator:  
(Member)

---

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

Head of Department:  
(Chair)

---

Sadia Hamid Kazi  
Associate Professor  
Department of Computer Science and Engineering  
Brac University

## Abstract

We present computer vision approaches combined with machine learning techniques to detect eve-teasing from any video material, which may be used in any situation. Eve teasing is a colloquial term for public sexual harassment or sexual assault committed primarily against women by men. Eve-teasing is a common and profoundly distressing reality for young women, particularly young girls. Moreover, in Bangladesh, the number of reported occurrences of Eve-Teasing is increasing at an alarming rate. Already eve-teasing is leading to rape and then murder. Due to a lack of proof and adequate supervision, the vast majority of criminals can get away with their crimes. In order to eliminate this problem, we suggested computer vision approaches for detecting eve-teasing in any video clip, which may be used in any critical situation. Our proposed method uses machine learning and computer vision to detect human Gender, expression, posture, and gesture and combine them to verify a matching human behavior in a critical situation to justify. When employing the combination of male-female identification, human behavior detection, and CCTV-based monitoring systems or video footage, the system can identify such vital conditions in our suggested method. Also included is the ability to determine who is participating and who is not in the scenario. Women who want to prove eve-teasing or harassment may find the procedures proposed helpful while ensuring the actual offender is punished.

**Keywords:** Machine Learning; Eve teasing; detection; CNN; XGBoost; vgg16; Gender classification; Posture detection;



## **Acknowledgement**

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.

Secondly, to our supervisor Md. Golam Rabiul Alam Sir for his constant support and guidance in our work. He always helped with his suggestions whenever we needed help.

Thirdly, to our co-advisor Saily Roy ma'am for her advice and help. Without her support, we could not have completed our work.

And finally, to our parents without their constant support it may not have been possible. With their kind support and prayers, we are now on the verge of our graduation.

# Table of Contents

Declaration	i
Approval	ii
Abstract	iii
Acknowledgment	iv
Table of Contents	v
List of Figures	vii
List of Tables	viii
Nomenclature	ix
<b>1 Introduction</b>	<b>1</b>
1.1 Thoughts behind the Model . . . . .	1
1.2 Problem Statement . . . . .	2
1.3 Research Objective . . . . .	2
<b>2 Related Work</b>	<b>4</b>
<b>3 Methodology</b>	<b>9</b>
3.1 Methodology behind the proposed Model . . . . .	9
3.1.1 Data Procurement . . . . .	10
3.1.2 Data Splitting . . . . .	11
3.2 Neural Network Models . . . . .	11
3.3 Gender Detection . . . . .	13
3.3.1 Dataset . . . . .	13
3.3.2 Data Preprocessing . . . . .	14
3.3.3 Modeling . . . . .	15
3.3.4 Training the Model . . . . .	16
3.3.5 Performance Evaluating Methods . . . . .	16
3.4 Posture Detection . . . . .	17
3.4.1 Methodology Behind Posture Detection . . . . .	17
3.4.2 Feature Extraction for key points . . . . .	18
3.5 XGBoost Model . . . . .	21
3.5.1 Bagging and Boosting . . . . .	21
3.5.2 Math Behind XGBoost . . . . .	22

3.5.3	Hyperparameter . . . . .	23
3.5.4	Hyperparameter Tuning . . . . .	23
3.6	VGG16 . . . . .	24
3.6.1	Data Preprocessing . . . . .	25
3.6.2	Modeling . . . . .	25
3.7	Transfer Learning . . . . .	27
<b>4</b>	<b>Implementation &amp; Result Analysis</b>	<b>29</b>
4.1	Result Analysis of Gender Detection . . . . .	29
4.1.1	Evaluating Model Performance . . . . .	30
4.2	Result Analysis of XGBoost . . . . .	31
4.3	Result Analysis of CNN . . . . .	33
4.3.1	Accuracy . . . . .	33
4.3.2	Specificity . . . . .	33
4.3.3	Precision . . . . .	33
4.3.4	Recall . . . . .	33
4.3.5	F1 score . . . . .	33
4.3.6	Evaluating Model Performance . . . . .	33
4.4	Result analysis of Transfer Learning . . . . .	35
4.5	Final Result . . . . .	36
<b>5</b>	<b>Conclusion</b>	<b>38</b>
<b>6</b>	<b>Limitation and Future direction</b>	<b>39</b>
	<b>Bibliography</b>	<b>41</b>

# List of Figures

3.1	Workflow diagram of Proposed Model . . . . .	10
3.2	Workflow diagram of gender detection . . . . .	13
3.3	Dataset images of man faces . . . . .	14
3.4	Dataset images of woman faces . . . . .	14
3.5	Keras layer to classify gender . . . . .	16
3.6	33 pose landmarks of human body along with their indexes . . . . .	18
3.7	Posture detection using all 17 key points . . . . .	19
3.8	Coordinate vector of a keypoint . . . . .	19
3.9	L1,L2 norm and polar coordinates of a keypoint . . . . .	20
3.10	Bagging and boosting . . . . .	22
3.11	Hyperparameter Tuning Values . . . . .	24
3.12	Hyperparameter Tuning . . . . .	24
3.13	Workflow of VGG16 Model . . . . .	25
3.14	Plotting Keypoints on a white background . . . . .	26
3.15	Workflow of Transfer Learning . . . . .	27
4.1	Gender Detection in Good Light . . . . .	29
4.2	Gender Detection in low Light . . . . .	30
4.3	Score Analysis of the Model . . . . .	30
4.4	Gender Detection Accuracy and Loss . . . . .	30
4.5	Summary of our Sequential Model . . . . .	31
4.6	XGBoost train and validation result . . . . .	31
4.7	XGBoost score analysis . . . . .	32
4.8	XGBoost confusion matrix . . . . .	32
4.9	VGG16 Accuracy and Loss . . . . .	34
4.10	VGG16 confusion matrix . . . . .	35
4.11	Transfer learning train and validation result . . . . .	35
4.12	Transfer learning score analysis . . . . .	36
4.13	Transfer learning confusion matrix . . . . .	36
4.14	Score comparison of all the models . . . . .	37
4.15	Eve teasing Detection . . . . .	37

# List of Tables

3.1	Data Splitting with Parameters . . . . .	11
3.2	Data Splitting with Parameters(gender) . . . . .	15

# Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

$\theta$     Theta

*AI*    Artificial Intelligence

*CNN* Convolutional Neural Network

*GBM* Gradient Boosting Machine

*HOG* Histogram of Oriented Gradients

*ML*    Machine Learning

*SGD* Stochastic Gradient Descent

# Chapter 1

## Introduction

### 1.1 Thoughts behind the Model

The true goal of engineering is to reduce human effort and increase safety while making our lives more pleasant and safer. If we look around us, we can see that many crimes are happening. Although some crimes have been reduced with the emergence of technology, some crimes remain the same. One of these problems is Eve-teasing, which is now a leading problem globally. Nowadays, eve-teasing is a pressing issue that primarily affects girls, but it can also affect boys. Eve teasing is a term for sexual harassment in public or sexual assault most commonly used by men against women. Eve teasing has been defined as staring, stalking, passing comments, and unwanted physical contact with the opposite sex. According to a survey by Daily Prothom Alo, A woman in South Asia gets teased every 51 seconds. In our country's perspective, Eve-teasing happens every day from men to women, which can be a traumatic case. In article [23] they stated that 98 percent of the women have stated that sexual harassment on roads has affected their personal or academic development in one way or the other. According to the Bangladesh National Lawyer Association, there were 24 eve-teasing reports in newspapers in 2006, 39 in 2008, 54 in 2009, and 52 in 2010 [7]. However, Odhiker recorded 672 eve-teasing instances in 2011. In 2010, Odhiker reported 129 events [7]. In a survey, From this poll [22] , 65 to 70 percent of eve-teasing occurs at Bangladeshi educational institutions. According to the participants, the perceived repercussions of eve-teasing included severe limits on victims' mobility, an inability to attend a school or a job, girls being blamed the most, and generating problems at home. Sometimes the victims feel so vulnerable that this causes the loss of life. According to the World Health Organization (WHO), about 1.53 million people will be dead by 2020 through suicide, and plenty of them will cause eve-teasing. Although eve-teasing is a crime, most of the time, this crime is overlooked and taken lightly. The victims suffer, and the culprits roam freely and continue to do this horrendous crime. There is no specific group of people that does kind of act. The teasers can be anyone from a beggar to even a student. According to a JNU study, eve-teasing affects everyone from a coworker to her supervisor in the office, a female student to her male teacher, and maids to their masters. According to one study, approximately 33% [22] of eve-teasers are middle-aged males, and their employer abuses 11% of women at work. Students make up 32% [22] of eve-teasers, while other antisocial, worthless, and disgusting minors make up 35% [22]. Most of the cases are unreported do not have proper and necessary proof. Our task's

motivation came from the growing rate of eve-teasing. There are many cameras around different points of the streets. However, the issue is that these cameras cannot detect if any crime is happening or not. We have proposed our project that uses Computer Vision and machine learning to identify Eve-teasing. The proposed system will detect behaviors like groping, going too close to a woman/man, gestures, and posture of the subjects in the video clips. This method will be used both ways. Firstly, It can predict any unexpected occurrence using computer vision. Secondly, it can also detect any suspicious behavior from any video evidence. So the benefit is that these video clips and detection could be used as proof for victims to get proper justice.

## 1.2 Problem Statement

Eve teasing has become one of the alarming problems of Bangladesh. It has been shown that 90% of girls between the ages of 10-18 are affected by eve-teasing. The situation has grown to such proportions that some are forced to give up their everyday lives. Some are forced to stay at home, and young people get married to avoid this problem. Even the suicide rate has increased among women in the past three years. This whole eve-teasing scenario is much worse globally. It was shown that 35% of women globally had faced physical or sexual violence. Most of the cases of eve-teasing occur in public places. When girls go to school/college, people pass vulgar comments, winks or claps, or sometimes unnecessary touching and pushing. These things are also every day in public transports. A 2018 study by BRAC has revealed that 94% of women commuting in public transport in Bangladesh have experienced sexual harassment in verbal, physical, and other forms. These numbers are too high to ignore. Moreover, eve-teasing is also seen in workplaces. The question that arises is why this problem has not yet been solved. As shocking as it sounds, there is no specific law against eve-teasing in Bangladesh. There could be various reasons why we cannot overcome this heinous act. One of the reasons can be that, in most cases, there is no proof of the incident. Moreover, the victim does not get proper justice because of this lack of evidence. Furthermore, sometimes the victims do not protest as they know they do not have any proper evidence, and even if they protest for the lack of evidence, they cannot prove it, so they do not get justice. There is no security system for women in public places, public transport, or workplaces. So, even if some incidents happen, there is no way to identify the actual culprits and punish them. This is where our solution comes in. We will use our algorithm to identify any teasing activity from footage and even send an emergency request to the closest police control room.

## 1.3 Research Objective

The number of reported incidents of eve-teasing continues to spiral out of control across the world. It is not safe for women to go about their daily lives outside the home. Our goal is to make the world a better and safer place for women so that they can move freely all over the place and at any time. In addition, assisting them in gaining self-empowerment so that they don't have to keep a watchful eye on their surroundings or live in constant terror of some unfavorable occurrence is essential.



In addition, one of our goals is to devise a method that will demonstrate to the criminals that their every move is being closely monitored at all times. Because of this, people who commit offenses will give it some serious consideration before carrying it out. Every action that they take will be continuously checked, and if there are any offensive behaviors among them, those actions will be flagged as such. Our goal is to develop a computer program that can identify each and every criminal responsible for the offense. To summarize:

- For detecting eve teasing first gender classification needs to be done.
- Recognizing human behavior pattern from a video footage
  - Extracting human gesture
  - Extracting different human postures
  - Extracting facial expressions.
  - Combining the extracted data to recognize human behavior patterns.
- Training the human behavior classifier to figure out the posture and expression of the offenders in a critical situation.
- Consider the potential areas of future study that have accumulated on the basis of the paper.

# Chapter 2

## Related Work

Regarding our topic, not a lot of work was done in this sector. Although, some work was done in other ways than our proposed method. We will start by discussing the alternative methods that were previously explored. In the paper [20] [13], a low-cost women's safety device that they made. They used a SIM900A GSM module with Arduino UNO interface, NE06M GPS module, RF Transmitter and receiver module, a buzzer, an accelerometer MPU6050, and a voice recorder module ISD1820. According to their research, most of the security apps in smartphones and devices used to ensure women's safety are not that useful. Moreover, the security devices currently on the market are pretty expensive and are still not accessible to rural people, where the cases happen the most. So the device that they made has two separate devices. A transmitter part that is fitted is an effortlessly wearable wristband from where the buttons will be pressed in emergencies. The other part will have the receiver part, which will carry the GSM, GPS module, accelerometer, and voice recorder. When a button is pressed on the wristband, the GPS module will calculate the current latitude and longitude. This location will be sent via the GSM module and the emergency message to the registered contacts. Furthermore, the buzzer will make noise to alert nearby people, and the voice recorder will turn on, which can record up to 20 seconds. Also, if the victim does not get a chance to press the button and falls to the ground, the MPU6050 sensor will detect the fall and send a similar emergency message to the contacts with the current location. In addition, there is also a feature that, when the button is accidentally pressed, there is another button that sends a new message will go to the emergency contacts that says "Everything is fine" and stops the buzzer sound and voice recording. If we now talk about the real-world usability of this device, there is a mixed opinion. Firstly, let us talk about the pros of this device. This proposed device is helpful in many ways for women's safety, and it is, of course, way better than the previously attempted ones. because the components they have used are cheap, thus making this device is a very affordable one compared to the devices available in the market right now. Additionally, this device can be easily carried as a wrist band and jacket, so others are unrecognizable as a safety gadget. Furthermore, the device does not require any internet or mobile data connection. By considering all these features and advantages, this device feels like an almost perfect solution for this issue. However, this device still has a few flaws, and some improvements can be made to make it even more functional. For example, they can use cameras to record short videos, which can later be used as a piece of evidence. Besides, as the sensors are cheap

they don't always give the right reading. The paper [9] states an intelligent device that can keep track of any unusual activities. They achieved this using a flying device which looks like a bird, as a processor Arduino Uno, for image processing they have used mat-lab tool, for communication they have used RFID system and GPS/GSM module. They have used the bird's architecture, and it flies following the aerodynamic model. They have used a stepper motor for 360-degree rotation of the camera, which takes pictures and sends them to administration to easily identify and keep track of unusual activity around the city. Batteries have been used for power supply to the bird. A GPS module has been used to track where the bird goes and find its exact location. Mat lab tool has been used for image processing at every state and tracking the suspect object. It tracks objects using the SIFT Algorithm. The robotics bird, first of all, takes images and sends them to the administrator to check. After finding the suspect, the robotic bird starts tracking using image processing. If it loses its signal for any reason, it sends the signal to the nearest police station using Dijkstra's Algorithm immediately. However, the robotics bird can not send signals and send pictures to the administrator in some places because of using GPS and GSM modules. The GPS module does not work if there is any building and tree around, and the GSM sensor does not sometimes work because of voltage drop. If the suspect person gets into the car or any building, it can not keep its tacking. Also, the car moves into the tunnel, and it loses its tracking. Also, if the signal is not strong enough, the pictures will take time to send to the administrator, and the robotics bird can not get the feedback immediately. In paper [16] states an IoT-based device to prevent harm to women by ensuring automated alerts (message) to their dear ones and nearby police stations to help them rescue. They achieved this using a pressure sensor, pulse-rate sensor, a temperature sensor (NTC Thermistor temperature sensor module), GPS, and GSM (SIM808) module. For power, they used a 12V rechargeable Li-ion battery. They also added a push button and a buzzer for manual trigger and buzzer to alert nearby people. The pressure sensor detects if any pressure is being applied to the woman beyond a threshold limit, the pulse-rate measures any abnormalities in the pulse rate of the woman, the temperature sensor detects any deviation in the woman's body temperature, thus combining all the data with matching them with a critical situation scenario threshold. All the calculations and control are done with an Arduino Uno. When the data of all three cross a threshold limit, the GSM module sends an alert message to relatives and nearby police stations with the victim's location using the GPS module. This system has great potential because of its automated system. As in a critical situation, manually massaging someone may not be possible. So, an automated system will significantly help in critical situations to ensure safety. However, they proposed an Adriano Uno-based system with many modules that made their prototype a bit bulky and made it hard to carry all the time. One of the cons of this system is not having proof of the situation as it does not have any visual aspect. For this reason, the offender may not be brought to justice. Thus, he will continue his offenses in the future, leaving the world as it is. It may help in the short term, but this kind of inhuman offense will continue to happen in the long term. In [8], they proposed exciting research. The system uses CCTV surveillance systems in work environments, public places, and transport to automatically identify scenes of interpersonal crime. Interpersonal crimes like bullying, harassment, assault, and eve-teasing. To achieve their goal, they used both image and speech processing.

CCTVs installed everywhere nowadays, and it is an elegant approach to solving an ever-existing issue. They implemented this by setting a “critical situation” threshold to trigger their algorithm. A logistic regression-based clustering algorithm is used to set the critical situation threshold because the office environment has a privacy situation, so the recording of video and audio will start only if the threshold limit is crossed. They are calling this system “Conditional video and voice recording”. Furthermore, if a person is alone in a room in an office environment, there will be no audio or video recording to maintain privacy. Also, notes and scribbling on a board or screens will be blurred for personal purposes while recording any documentation. Moreover, they suggest that image processing alone may not be able to verify a situation, so they will use speech recognition in conjunction with image processing to increase accuracy. In public places, privacy will not be an issue anymore. So, the algorithm for human detection and tracking can keep running. As mentioned, after the critical situation is triggered, the heavy algorithm, but before that, a preliminary test algorithm keeps on running. For their preliminary algorithm, multiple moving object tracking algorithms were used based on Gaussian mixture models. They made a bounding box to track persons. The Kalman filter does the motion tracking. It works by watching an object for a set of frames before removing its position in succeeding frames and calculating mistakes for future predictions. As a result, the bounding boxes include collision detection; when two or more boxes contact, a critical situation is triggered, and an alarm is sounded to alert authorities. Specific keywords that could be rude or inappropriate if identified would set off the critical condition in voice processing. They employed the Mel frequency cepstral coefficient as a characteristic to characterize the keywords in their keyword spotting method. The characteristics or coefficients derived from the inverse Fourier transform of the logarithm of the power signal are then used to see whether any keywords are present that could trigger the critical scenario. Their heavy algorithm suggested some great ideas though none of them were implemented in this paper. They suggested using human motion, human expression to detect any inappropriate situation. It may be a great approach to detect eve-teasing that we propose using in our paper. They alert the authority if the critical situation threshold is reached. Their approach is appreciated. They used a straightforward but effective way to detect and prevent any inappropriate situation. Although using colliding boxes may not always be helpful as people will walk close to each other most of the time, their audio processing may come in handy in these cases. However, audio filtering also may be confusing from time to time because of some situations. Thus false alarms will be set often. So, not a lot of work is done in the computer vision field related to eve teasing. But the paper [8] talked about some interesting ideas that they haven’t implemented, we shall be working towards those ideas. We propose using human motion, expression, and gesture after detecting their gender to find any inappropriate situations marked as eve-teasing. Papers related to these topics are described now. For gender detection from video footage, Liangliang Coa from [2] detected gender from a frontal view and a back view. Gender is quite tough to detect from an image or video because of the nature and type of human bodies and their clothing. To solve this, they made a three-way approach. First, to handle different human body types and characteristics, each image was represented by patch features to model different body parts to provide clues for gender recognition, feeding it to their ensemble learning algorithm to combine all the clues to detect gender with a fixed image constraint. Lastly, for

mixed view images, they trained a flexible classifier with almost the same accuracy to relax the constraint. SEXNET first did gender recognition, but from faces, later much paper tried, but all were human face based. This paper was the first to do gender recognition from the entire human body. Histogram of Oriented Gradients (HOG) was used to represent the image with hard edges with a magnitude-weighted histogram, that is grouped according to edge directions. Each body is divided into 3 x 3 blocks, each block is represented by a HOG feature, with a length 8 vector describing the gradient in 8 orientations. After preprocessing the data, they used Adaboost and Random Forests (RF) classification methods. Adaboost is a weighted additive model that creates a strong classifier by combining all the weak classifiers. Here they chose the most discriminative decision stumps. After that Random forest is used to train successive decision trees and combine them for prediction. They introduced their Part-Based Gender Recognition (PBGR) algorithm that specifies the characteristics of human body images. It may be hard to describe the exact gender characteristics, but it may be more accurate to say that a woman will have long hair in general and a person wearing a skirt will likely be a female. As we discussed they used grid sampling to partition the image into patches. Here HOGs come into play again. Their grid is 6 x 19 here. Their belief is that gender is more related to body parts than the whole. For this reason, they are considering each body part individually. Now to their method, by getting all the cues from all the body parts they used the ensemble learning algorithm which constructs a similar group of weak classifiers. The model first selects the most discriminative patch position and then computes the optimal classifier using only a single patch. Each classifier is based on distinct body components and uses weighted voting to determine recognition results. Individual body parts may be doubtful at a certain time but combining all the cues will prove robust. They came up with an extraordinary way to solve this problem. Their algorithm can be held as a variant of the Adaboost algorithm as it also generates weak classifiers based on subset features instead of the whole image. Also, their approach out-formed the traditional approaches. Also, they got a 75.0% accuracy in both frontal and back view. We just have to see how good it will hold for individual video frames rather than static images. Moving to expression detection, in paper [4][6], For frame extraction and face detection, they used MLP neural network and Gabor feature extraction together to handle live streaming frames. To detect expressions, face expressions are clustered together using principal component analysis. Feature point location The utmost pixel in a window is used to extract feature vectors from a face image. Feature vectors are formed. They used the composition of Gabor wavelet coefficients sat the feature points for feature vectors. They used the similarity function to compare. Moreover, for the face comparison, each of the feature vectors of test faces is compared to a reference image, and OSi is used to find their similarity to that of the test image. They used creating clusters for the learning phase. Respective expression using PCA. Then, they took aimless images from the dataset and used pattern recognition. Their work is very efficient, and also they can work more on the details. In this sector papers [3] [1], eyes and mouth detect edges and calculate the gradient between the two regions' region figures. However, it will be difficult as most of the videos do not have high resolution, and if an image is noisy, it will be hard to detect expression as the facial features will not be apparent. We might need to detect face to recognize human expression. In [5] states human posture recognition using Microsoft Kinect sensor.

It helps to automatically identify any human postures. 9 features were represented to utilize human body posture such as thigh, forearm, spine, crus, etc. But this proposed method showed a final overall accuracy of 99.14% success rate to recognize all the postures. By using FFAST( Flexible Action and Articulated Skeleton Toolkit), it can pre-defined 27 human postures. But using the proposed method reformed the recognition of the human posture. It has used depth image, skeleton information, feature extraction for posture learning. For poster recognition, it has used the trained model, depth images, skeleton information, feature extraction. By extracting the human feature it can easily recognize the human posture. These features represent all the important information for poster recognition by using its proposed method. To recognize it, it defined 22 different postures including the most common postures and complex postures also. All the data has been collected from Kinect at the rate of 30 fps(Frame per second). For posture recognition, first of all, investigate the features from depth images of body posture, then visualize the distribution of body postures. All the body postures are separated from each other except lean forward and stand up. In the reduced feature space, the posture can be clustered with the features. The cost-efficiency of C and the gamma efficiency of G which are associated to recognize the posture is the critical step. Also in the reduced feature space, the two postures can not be clustered which are lean body and stand up because of noisy skeleton data and lean forward is misclassified as a stand-up. There is a con like the rotation of the human body can create problems to identify the human posture. It needs to be more robust for human posture recognition from any angle and from any height. From the top we looked into the related worked and found little to non work being done with computer vision and machine learning field to tackle eve teasing. There are no direct model to solve or detect eve teasing from a video footage. Our proposed method will be the first direct model to detect eve teasing from a video footage. From studying the papers, we can summarize the approach of combining our methods to figure out human behavior from a scenario to detect eve teasing.

# Chapter 3

## Methodology

### 3.1 Methodology behind the proposed Model

Our target of this project is to find a solution for the social problem known as 'Eve-teasing'. We have proposed a system where we will use our own algorithm to solve this problem. For this approach, we have to make the algorithm using some detection. The whole work process has several detection parts. They are gender detection and posture detection. In this paper, we have described the methodology of our detection models. We also implemented the models. The first part of our methodology is to detect gender from video footage. That means we will differentiate between males and females. Secondly, we will detect the posture of these males and females and record the accuracy of the prediction. To start detecting eve-teasing from video footage, we first need to determine if there are any males or females in the data frame. In a word, we need to detect gender in video footage. [2] From their paper, we get a general idea of recognizing gender from a static image. Moreover, we can use [8] their methodology to set a bounding box to the detected male and female, then use them to detect collusion to trigger our following methods to detect posture and determine the situation and increase our accuracy. For now, we have been able to do gender recognition by using a gender classifier dataset[14]. We used the sequential model to train our classifier. For 'Posture Detection', we used real-world videos of eve-teasing (that will work as our data set) to find matches of these critical situations in any video footage. We tried to recognize different postures from our method model and use them in the detected male-female dataset. Next, we used the gender detection model and used the posture detection model on top of it. So, now we have detected males and females in a video frame with key points. Now we used the data in a GBM. For the GBM we have selected the XGBoost model. Then we converted our key points into linear values like L1, and L2 norms and polar coordinates  $r$  and  $\theta$ . We also used the CNN model VGG16. In the VGG16 model as the features, we plotted our already detected body points on a white background. We used a white background for better clarification of key points. After that, we trained the model with these datasets. Furthermore, we predicted the feature output of the VGG16 model. Now for better prediction, we used our own methodology. In this step, we took the output weight of the prediction from our CNN model VGG16. And feed it to the XGBoost classifier. Combining both the output weight of CNN and the prediction from the XGBoost classifier, we got the final detection of eve-teasing.

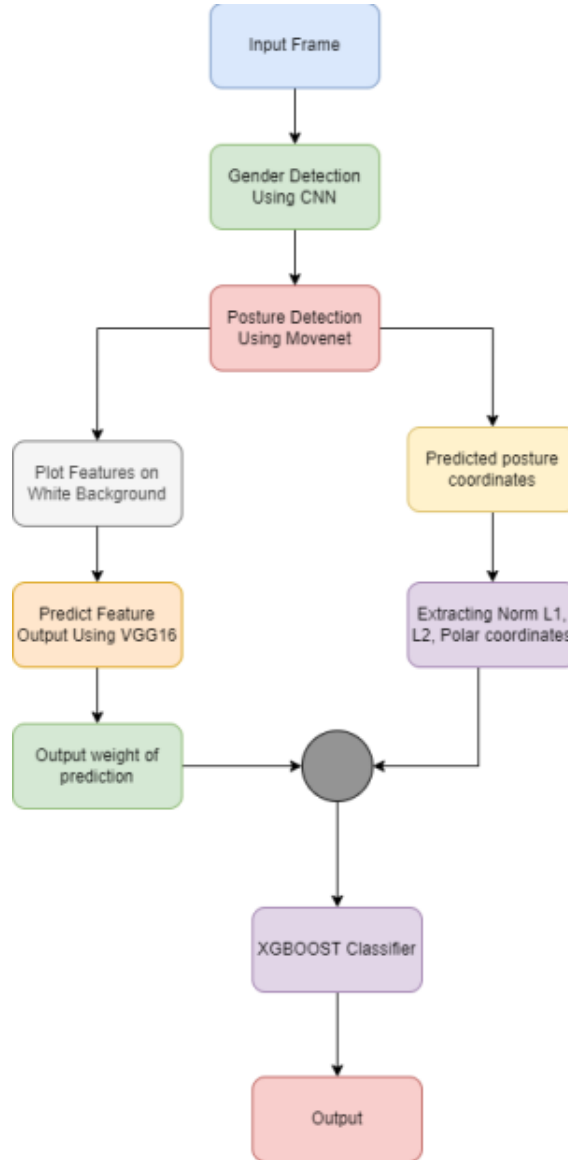


Figure 3.1: Workflow diagram of Proposed Model

### 3.1.1 Data Procurement

For our research’s proposed algorithms to work as expected, they need to be run on more than one dataset. So, a clean, well-organized dataset is thought to be one of the most important parts of any research. The convenience of the dataset largely determines how well the algorithms perform in research. We have two sections. One task is to determine whether a person is male or female based on their surroundings. And another section is to detect the posture of that male or female. The thing is that it is very tough to find labeled data for this kind of dataset. For ML research, Kaggle is a good choice for both classified and training datasets. Therefore, we found a good and resourceful dataset for our purpose. For our first section to determine gender, we have used the Gender Classification Dataset. This dataset contains cropped images of humans that have been categorized as male or female. It has been split into two directories. One is training, and another is a validation directory. The Training Directory contains more than 23,000 images of each class. Thus, there are more than 23,000 images for males and females, respectively. and



the validation directory contains around 5,500 images of each class. Our model can be trained with 23000 images. Having the labeled images will help the model use the resources efficiently. The dataset we have used is open-source and free to use. It can be found here: <https://www.kaggle.com/datasets/cashutosh/gender-classification-dataset> Now that we have all the data we need for gender recognition, We will take input for our next step, which is posture detection. Here we will use real-life videos as inputs. Here, the video source can be any real-world footage. For example, we have used scenes from movies, CCTV footage, short films, and awareness videos related to eve-teasing and the primary dataset. These videos provide real-life scenarios of eve-teasing. So it will be more accurate and effective for posture detection in actual cases.

### 3.1.2 Data Splitting

For machine learning projects we need to divide the data into parts for training, testing, and validating. This is Data splitting. Our Dataset for Eve-teasing contains 5434 pictures which split the dataset into the three categories of training, testing, and validation data. In our case, we used 3808 pictures for training data, 538 pictures for testing data and 1088 pictures for validation data with the 7:1:2 ratio. The total dataset is divided into 70 percent training data, 10 percent testing data, and 20 percent validation data.

Data Type	Percentage	Parameters
Training Data	70%	trainX trainY
Testing Data	10%	testX testY
Validation Data	20%	validX validY

Table 3.1: Data Splitting with Parameters

## 3.2 Neural Network Models

Due to its ability to process enormous amounts of data, deep learning has emerged during the past two decades as a very successful technology. Particularly in pattern recognition, the utilization of hidden layers has surpassed that of traditional methods. One of the most popular deep neural networks in deep learning is convolutional neural networks. A specific kind of network called a CNN is made for deep learning algorithms and is used for operations like image recognition and pixel data processing. CNNs are the chosen network architecture for detecting and recognizing objects in deep learning, despite the fact that there are several types of neural networks. The CNN is a different kind of neural network that can identify significant information using both time series and image data. It is therefore particularly useful for image-based applications including pattern recognition, object categorization, and image identification. To discover patterns in an image, a CNN employs linear algebraic ideas, such as matrix multiplication. CNNs are particularly helpful for image identification, image classification, and computer vision (CV) applications because they produce incredibly accurate results, especially when a large amount of data

is involved. As the object data passes through the CNN's numerous layers, it also picks up the features of the item over time. The requirement for manual feature extraction is eliminated by this direct (and deep) learning (feature engineering).

## **CNN Layers**

There are three layers in CNN. One is A convolutional layer, the second one is a pooling layer and the last one is a fully connected layer.

From the convolutional layer to the FC layer, the CNN gets more complicated. Due to the increasing complexity, CNN is able to recognize bigger and more complicated parts of an image until it can identify the whole thing.

### **Convolutional layer**

Most calculations are done in the convolutional layer, which is in the middle of a CNN. After the first convolutional layer, another convolutional layer could be added. During the convolution process, a kernel or filter in this layer moves over the image's receptive fields to figure out if a feature is there or not.

The kernel looks at the whole image over and over again. At the end of each iteration, a dot product is made between the pixels that were sent in and the filter. When you connect the dots in a certain way, you get a feature map or a convolved feature. In this layer, the image is turned into numbers that the CNN can understand and use to find important patterns.

The kernel scans the image over and over again. Each cycle calculates a filter-pixel dot product. A feature map or convolved feature is made up of connected dots. This layer turns the picture into numbers that the CNN can understand and use to find patterns.

### **Pooling layer**

Like the convolutional layer, the pooling layer sweeps a kernel or filter across the image it is given. The pooling layer is different from the convolutional layer in that it has less input parameters but also loses some information. This layer makes the CNN easier to use and makes it more effective.

The kernel scans the image over and over again. Each cycle calculates a filter-pixel dot product. A feature map or convolved feature is made up of connected dots. This layer turns the picture into numbers that the CNN can understand and use to find patterns.

### **Fully connected layer**

In the FC layer of the CNN, a picture is put into a category based on the features that were taken from the layers above it. In this case, "fully connected" means that every activation unit or node on the next layer is linked to every input or node on the layer before it.

All of CNN's layers are not fully connected, because if they were, the network would be too dense. It would cost a lot to calculate, cause more losses, and change the quality of the output.

## ConvNet

A convolutional neural network, which is a type of artificial neural network, is also called a ConvNet. A convolutional neural network is made up of many hidden layers, an input layer, an output layer, and an input layer. VGG16, a type of CNN (Convolutional Neural Network), is one of the best computer vision models we have right now. The people who made this model looked at the networks and improved the depth by using an architecture with very small (3,3) convolution filters. This was a big improvement over the current best setups. They added 16 to 19 weight layers, which adds up to about 138 trainable parameters. We used VGG, which is a convolutional network made for classifying and locating.

### 3.3 Gender Detection

In our project, the first step to detect eve-teasing is gender classification. We will have to classify males and females from a frame in order to do further detection. For this gender detection, we have used python, CNN, Keras, and OpenCV. In this methodology, we will explain step by step how we did gender detection from video footage. The workflow that we have followed step by step is given below.

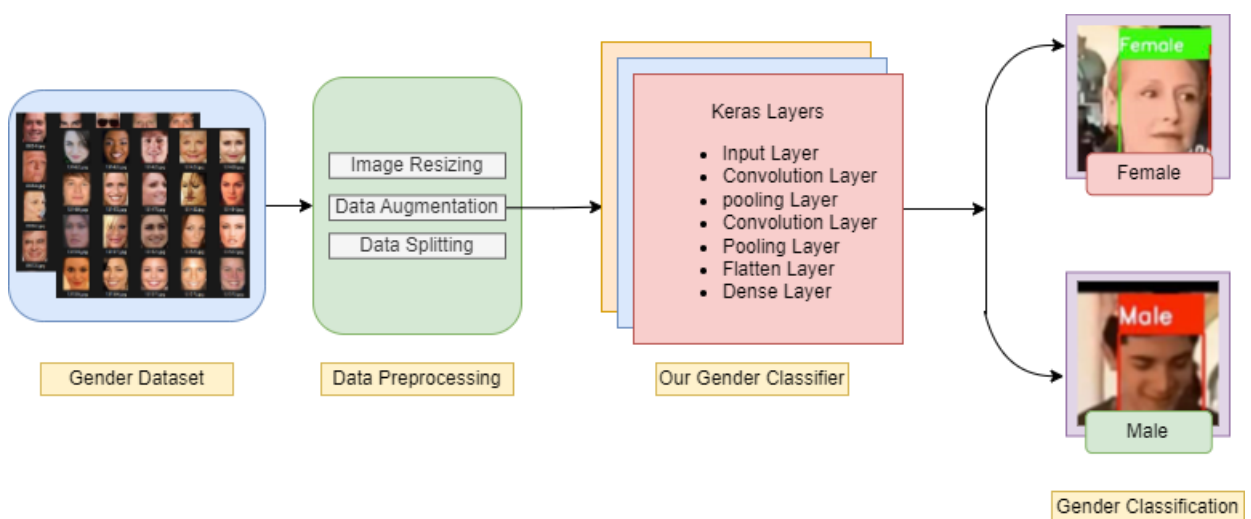


Figure 3.2: Workflow diagram of gender detection

#### 3.3.1 Dataset

For the dataset we took the dataset from the GitHub library which is linked below: [https://github.com/balajisrinivas/Gender-Detection/tree/master/gender\\_dataset\\_face](https://github.com/balajisrinivas/Gender-Detection/tree/master/gender_dataset_face). This dataset has two folders. One is a folder of men which contains the faces of men. There are around 1173 images of men's faces. Another folder is of women which contain about 1134 images of women's faces. The images were taken from Kaggle and google images. These were used to create a model which will be able to detect if a given image is a man or a woman.

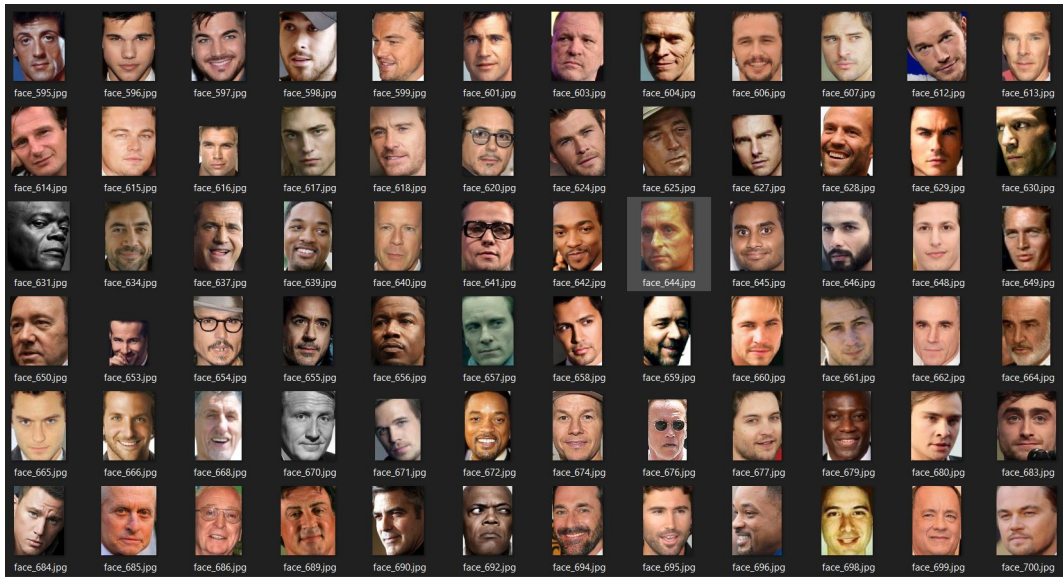


Figure 3.3: Dataset images of man faces

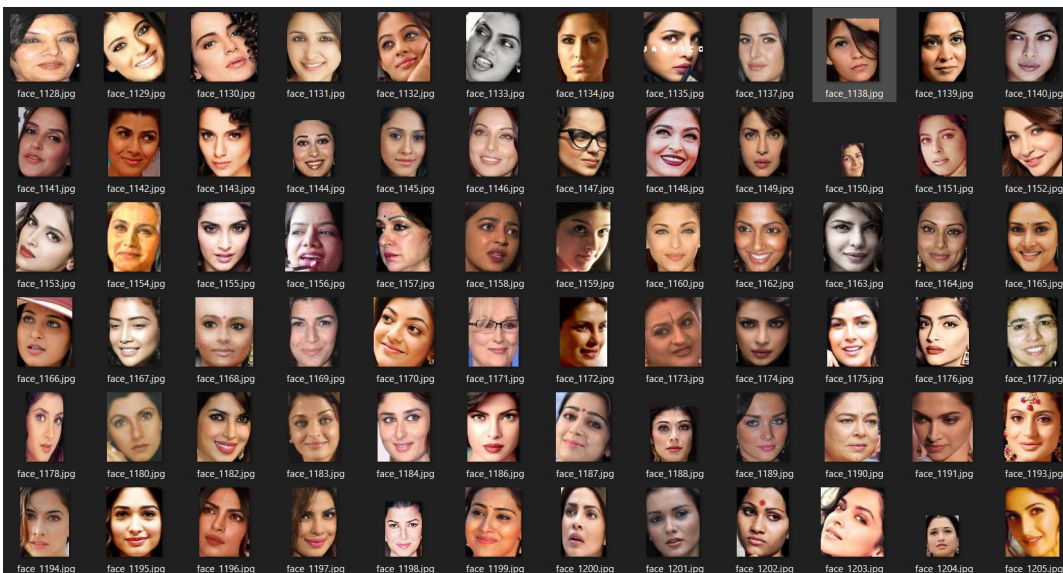


Figure 3.4: Dataset images of woman faces

### 3.3.2 Data Preprocessing

To preprocess our data the first thing we did was to turn our images into arrays. The images were resized to 96 by 96 pixels and normalized. We created two arrays one with the images and one with the labels. Next, we combined these two arrays and turned them into a single one. For an ideal training environment, we shuffled the selection of images that will be taken for training. As per the standard requirement, all images in the array had to be uniform so we resized the images to 96\*96. We also divided the data by 255 to uniformly keep the values of our image between 0 to 1. We also did turn the labels into categorical data. We used the ImageDataGenerator for data augmentation.

Data Type	Percentage	Parameters
Training Data	70%	trainX trainY
Testing Data	10%	testX testY
Validation Data	20%	validX validY

Table 3.2: Data Splitting with Parameters(gender)

### 3.3.3 Modeling

For building the model, we had to set some parameters first. The width and height both were 96, the depth of the model was 3 because of the 3 color channels, and there were 2 classes for men and women. We used the Keras Sequential model. We added 32 filters in the 2D Convolution Layer and the convoluted image size to 3\*3.

#### BatchNormalization

We did the batch normalization. It was done to our convolutional layer after the 'relu' activation function. The normalization was done to balance out the data points. Some data points may have a very high value and some have very low values. So if we do not normalize the high values might overshadow the lower values.

#### MaxPooling2D

It is used to reduce noise from the data samples. For example, in our dataset, the beard or mustache is an important feature for identification. And unwanted dots or pimples are noise in the samples. So Maxpooling will reduce unwanted things and keep the features for better accuracy.

#### Dropout

Dropout is used to limit the neurons which will learn from the data. So if we do not limit the dropout, the model will become overfitting. This means the model will understand the training data very well but will fail to understand any dataset that is outside our training dataset. We added 0.25 dropout which will deactivate 25% of neurons every time during the front propagation and the backpropagation.

#### Flatten and Dense

To convert the two-dimensional convolutional and max pooling layers to one-dimensional to feed into the dense layers. We then added 1024 neurons into our dense layer. In the dense layer, we used 0.5 dropouts to deactivate 50% of neurons.

#### Sigmoid

For the output activation layer, we used sigmoid as our activation layer. The reason for using sigmoid is that it is a probability-based accuracy function. And the output value they give is 0 to 1.

#### Optimizer

For the optimizer, we have used Adam. In the parameters, we have the learning rate 1e-3, epochs = 10, and batch size to be 32.

### 3.3.4 Training the Model

For gender detection, we used The Sequential model[15]. We applied this model because, For most situations, the sequential API enables us to build models layer by layer. We also used adam optimizer as it outperforms all other optimization algorithms in terms of performance, computing time, and tuning parameters. It works well on large data. In this model, we need to use CNN and Maxpull[18] together because It lets you build up a model one layer at a time and reduces the number of pixels in the output of the previous convolutional layer, which makes images smaller. And to get better results, we modified it and used Dense and flatten layers along with it, as Flatten layers can be used when you have a multidimensional output and want to make it linear so you can send it to a Dense layer. The dense layer changes the number of dimensions of the previous layer's output so that the model can easily define the relationship between the values of the data it is working with.

We used ReLU as our activation function.

$$f(y) = \max(0, x)$$

here y is the output and x is the input. y is always linear for all the positive values and 0 for all the negative values. It helps to coverage faster.

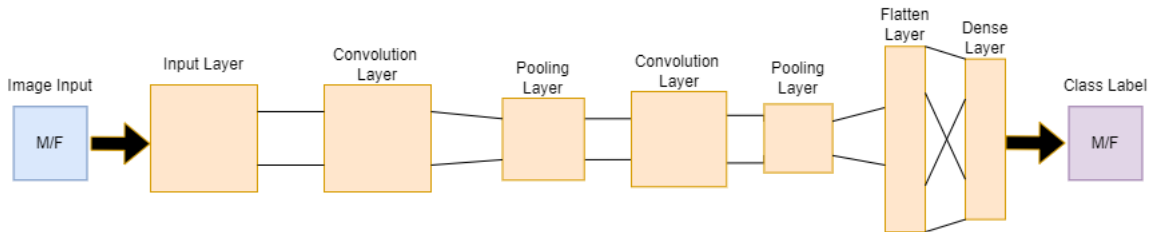


Figure 3.5: Keras layer to classify gender

### 3.3.5 Performance Evaluating Methods

We evaluated precision, recall, accuracy, sensitivity, specificity, and F1 score to evaluate the model performance. Precision [12] is the ratio of correctly labeled positive identification by all positive identification. A low precision will result in a significant number of false positives. Recall [12] is the number of true positives divided by the number of true positives plus the number of false negatives. it takes false negative cases into consideration. The F1 score is the average mean of precision and recall. Specificity [12] is the ratio of correctly labeled negative identification by the number of true negative plus false positive.

$$\text{Accuracy}_i = \frac{\text{TP}_i + \text{TN}_i}{\text{TP}_i + \text{TN}_i + \text{FP}_i + \text{FN}_i}$$

$$\text{Precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}$$

$$\text{Sensitivity}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i}$$

$$\text{Specificity}_i = \frac{\text{TN}_i}{\text{TN}_i + \text{FP}_i}$$

Where,

- TP= True Positive.
- TN = True Negative.
- FP = False Positive
- FN = False Negative.
- i = identification.

## 3.4 Posture Detection

### 3.4.1 Methodology Behind Posture Detection

For posture detection, one of the most widely used methods is Mediapipe. Mediapipe delivers a solution to predict 3D landmarks or key points in a human body in real-time[21]. It has 33 landmarks on a human body for identification. It has two parts. One is localizing humans in a certain frame by using a detector. And the second part is to use the pose key points to predict the landmarks in the frame. It uses mp.solution.drawing\_utils[21] to visualize the landmarks on the image/video. After that, it passes the input image/video to the ML pipeline to perform the pose detection. While this Mediapipe model is much more accurate and robust to use even on low-end devices but there is a huge limitation. That is, it cannot detect posture for multiple people. To detect multiple postures, we have to use other models, which will be for identifying multiple people and then use the Mediapipe pose() to detect the posture. Thus, this way of proceeding is quite computationally expensive and complex. We then opted for another approach which is Movement—multi-pose model.

For pose detection, we have used Movement. Multi pose model. This model allows us to detect the posture of multiple people in a data frame in real time. This model was created by Google. They have used the same SOTA[19] architecture as all other pose detection models. But they have retained the inference time as low as they can. For this reason, this model provides more accurate vital points. We know for Deep Neural Networks(DNN), we need a large set of learnable parameters. We also need a large scale of labeled data so that the parameters can take full advantage of the data and generalize them. In keypoint detection, we need to augment our key points, which means we need to use a combination of rotation, scaling, and flipping. Our human body has a total of 33 pose landmarks or key points.[19] They are used



to identify movements of different parts of the body. In Movement, we use 17 key points to measure movements. These key points are scattered throughout the whole body in a specific position. We take these key points and embed them in our inputs. Then following the algorithm, the model identifies movements in order of the key points from one position to another. The 17 keypoint joints are :nose, left eye, right eye, left ear, right ear, left shoulder, right shoulder, left elbow, right elbow, left wrist, right wrist, left hip, right hip, left knee, right knee, left ankle, right ankle.

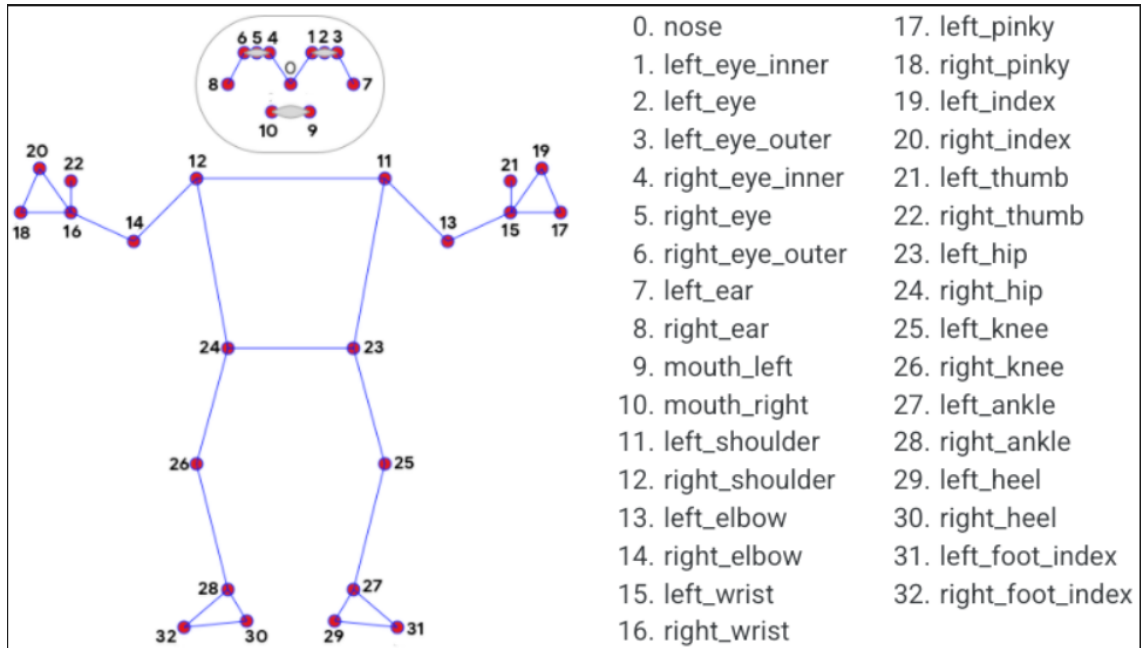


Figure 3.6: 33 pose landmarks of human body along with their indexes [21]

The way this model works is that it uses heatmaps for localizing human key points[19]. This is also called the bottom-up estimation model.[19]. This model has a feature extractor and a set of prediction heads. The prediction heads stick to CenterNet. The internet makes notable changes to improvise both speed and accuracy. For training, the model TensorFlow Object Detection API[19] is used. The feature extractor uses MobileNetV2 CNN architecture along with an attached feature pyramid network (FPN)[19]. This is used for high-resolution feature maps. On the other hand, the prediction heads have four attached to the feature extractor. The prediction heads can predict various types of cases. For example, it can speculate person center heatmap, person keypoint, 2D keypoint heatmaps, and keypoint regression fields. As we can see from the above pictures, our model is able to detect multiple postures with very good accuracy. In most situations, the model is able to get all the 17 key points that it is looking for. But there is a limitation that the model can only detect objects within 6 ft. So when the method crosses the range we are not getting the 17 key points.

### 3.4.2 Feature Extraction for key points

Now we have 17 key points of a single person after using the posture detection model. Each of these 17 key points is represented in a coordinate vector format. Which is





Figure 3.7: Posture detection using all 17 key points

displayed as two points  $x$  and  $y$ . The  $(x,y)$  coordinate is used to show the exact location of that specific key point. In the image below, we can see that the nose keypoint has 2 decimal values. Like that, every 17 keypoint will have 2 coordinates. so for each person, it will have values like this. This representation is easy to locate the key points but it is not ideal if we want to use this output anywhere else.

	A
nose	
	[0.3535996, 0.6220994]
	[0.3222369, 0.38716125]
	[0.3830207, 0.61464065]
	[0.31943765, 0.38016886]
	[0.3390567, 0.34802008]
	[0.2922291, 0.43098208]
	[0.32764554, 0.3579652]
	[0.7759363, 0.72701615]
	[0.33494234, 0.7099506]
	[0.30757275, 0.39362094]
	[0.36723888, 0.4370316]
	[0.31170514, 0.50597286]
	[0.36750787, 0.44424963]
	[0.2982125, 0.5234018]
	[0.32052726, 0.51255625]
	[0.2858046, 0.5162056]
	[0.36396676, 0.45788747]
	[0.29295665, 0.5182259]
	[0.36321354, 0.46513388]
	[0.46417844, 0.93177]
	[0.36840773, 0.46191502]
	[0.46855414, 0.9083357]
	[0.337329, 0.37163597]
	[0.33455747, 0.59919894]
	[0.3474999, 0.57945657]
	[0.34589738, 0.42177424]
	[0.33039942, 0.3650481]

Figure 3.8: Coordinate vector of a keypoint

We now have the posture data that we got from our model and we want to use it in a gradient-boosting method. We have selected XGBoost as the GBM. But there is a problem. As we have shown earlier that the posture data is in a coordinate vector format. But XgBoost only takes one value as a feature. So we will have to find a way to convert our vector to a linear form. so that our key points are made into a more meaningful value to feed to the XGBoost feature. We decided to use Vector norm for conversion[10]. The vector norm is basically the length of the vector or the

magnitude. This is a non-negative number that measures the size of the vector[10]. We have used L1 and L2 norms as a replacement for our coordinate vector values.

### L1 Norm

The L1 norm is calculated as the summation of absolute vector values. The absolute values are denoted and added to find the l1 norm. The formula for calculating the L1 norm is :

$$L1\_Norm = |\mathbf{x}| + |\mathbf{y}| \quad (3.1)$$

### L2 Norm

The L2 norm calculates vector space vector coordinate distance from the origin. As it calculates the Euclidean distance from the origin it is also called the Euclidean norm[10]. It returns a value that is greater than zero for the resulting distance. Calculating the L2 norm involves taking the square root of the sum of the squared vector values derived from the l1 norm. The formula for calculating the L1 norm is :

$$L2\_Norm = \sqrt{|\mathbf{x}|^2 + |\mathbf{y}|^2} \quad (3.2)$$

### Polar Coordinates

For feeding the posture data as the XGBoost features. We also need to find the polar coordinates of the given keypoints. We know polar coordinates have two parts. one is the length of the vector from the coordinate (x,y) to the origin that is r and  $\theta$  which is the angle between the vector and the positive x-axis. the formula to find the polar coordinates are :

$$r = \sqrt{x^2 + y^2} \quad (3.3)$$

$$\theta = \tan^{-1} \frac{y}{x} \quad (3.4)$$

Now we have the l1 norm,l2 norm, and also the polar coordinates of each of the individual key points. So, now the values look like this:

A	B	C	D
nose_p1_norm_1	nose_p1_norm_2	nose_p1_polar	nose_p1_degree
0.763050616	0.540317535	0.540317503	48.03775839
0.584583759	0.555227399	0.555227439	3.115600559
0.581201971	0.546631396	0.546631417	3.748819747
0.575353742	0.541061878	0.541061883	3.75717596
0.810445786	0.585345685	0.585345681	56.75400246
0.996462822	0.741787434	0.741787415	63.21771598
0.980089307	0.729645431	0.729645431	63.22888384
0.961506367	0.715484798	0.715484783	63.14939031
0.647705257	0.458913058	0.458913059	48.62125288
0.910757065	0.702584565	0.702584577	68.56326759
0.883478642	0.637180209	0.63718024	56.35245502
0.753242791	0.533128858	0.533128826	47.49585901
0.570438206	0.558726013	0.558726019	1.21400416

Figure 3.9: L1,L2 norm and polar coordinates of a keypoint

We have 17 key points for each person. And each keypoint has 4 values(L1 norm, L2 norm, polar, degree) to represent them. For a single person, Feature = 17\*4 =

68. And we have 2 people in each frame. so each frame has  $68*2 = 136$  features. So each frame will consist of 136 features. These 136 features will be used as XGBoost features.

## 3.5 XGBoost Model

In our project, we used the XGBoost model. It is the short form of an extreme gradient boosting algorithm. XGBoost is a very advanced gradient-boosting algorithm used for machine learning. It is very efficient, very flexible, and convenient as a distributed gradient boosting library. It uses the supervised learning problem. The XGBoost algorithm has many advantages which makes it better than traditional gradient boosting methods. The XGBoost model has advantages like efficiency, accuracy, and also feasibility. Some of the benefits of using this model are that it produces minimum error while assembling more quickly using fewer steps. Some more of the benefits are mentioned below.

- The model has both the tree learning algorithm and also the linear model solver.
- It has the capability to do parallel computation in a single machine using OpenMP. Therefore, making it 10 times faster than traditional GBM.
- Simplified calculations also reduce computation costs and improve the speed of the process.
- This model has the capability to find important variables rather quickly and correctly and also for doing cross-validation.
- XGBoost can be used simultaneously by multiple users with multiple processors effortlessly while handling terabytes of real-world data[11].
- It uses cache memory effectively for faster computation.
- For handling missing values, it uses parallel tree learning.
- XGBoost has an improved and better way to find the threshold for splitting in a tree.
- It is an end-to-end highly scalable tree-boosting system.

Because of these advantages over other gradient-boosting algorithms, XGBoost is being used by many big tech companies like Google, Alibaba, Tencent, and many other startups [11]. To explain this algorithm we have to understand the way machine learning works. ML has two important concepts known as bagging and boosting.

### 3.5.1 Bagging and Boosting

**Bagging:** bagging is a concept where we take random samples of data and use these data to build our learning algorithm. Next, we take the simple means of these data to find the probabilities.

**Boosting:** boosting is somewhat similar to the bagging approach but in this way

the sample selection is a bit more intelligent. Rather than taking random samples, boosting gives samples subsequently more weight to it. So that the weaker samples can be converted into strong ones.

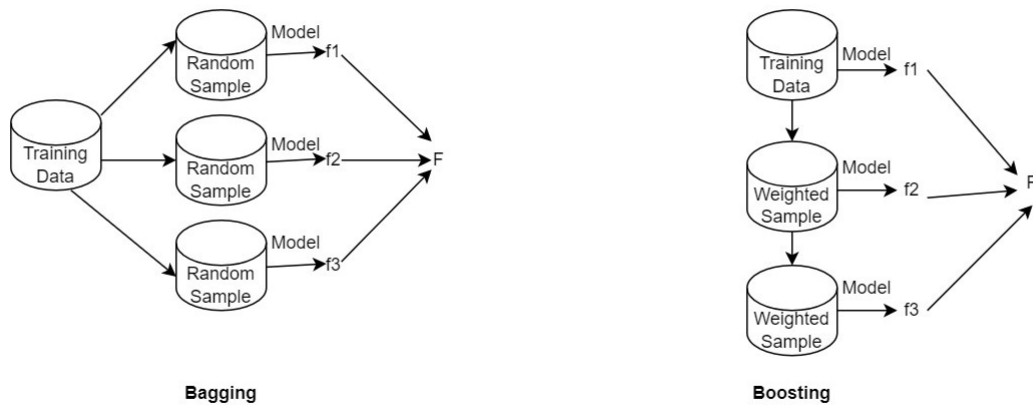


Figure 3.10: Bagging and boosting

### 3.5.2 Math Behind XGBoost

To understand how XGBoost works we need to explain some aspects of the algorithm to know what makes it so effective. We will be using the objective function and using Regularized learning objective. Regularization is done to help with overfitting and is used for being less sensitive to single samples. We will need the derivation quality score and output of a leaf. The quality score will help us to decide which threshold value we split on. And the output value will be used to predict new samples when we will do the final prediction. The equation for the output value and quality score can be written as this.

#### XGBoost Regression

$$Output\_Value = \frac{\sum R}{(n + \lambda)} \quad (3.5)$$

$$Quality\_Score = \frac{\sum R^2}{(n + \lambda)} \quad (3.6)$$

$$min\_child\_weight = n \quad (3.7)$$

#### XGBoost Classification

$$Output\_Value = \frac{\sum R}{\sum [ Previous_{Probability}_i * (1 - Previous_{Probability}_i) ]}$$

$$min\_child\_weight = \sum [ Previous_{Probability}_i * (1 - Previous_{Probability}_i) ] \quad (3.8)$$

### 3.5.3 Hyperparameter

#### **Learning\_Rate**

The learning rate is a hyperparameter of XGBoost. This denotes the pace or rate at which the algorithm will learn the estimation of the parameters and update accordingly.[17] This learning rate will determine the weight of our CNN and will control the gradient loss.

#### **Max\_Depth**

The max depth denotes the maximum depth of a single tree construction. To avoid the overfitting issue, we can try to keep the max\_depth as 3-4 at the beginning and then increase it to 1 gradually.

#### **N\_estimators**

n\_estimators represent the number of trees in the forest. Simply it means how many runs will XGBoost will need to learn. The higher number of trees is a benefit to learning the data.[17] But adding too many estimators or trees can affect the training process.

#### **Subsample**

What this parameter does is randomly selects the samples in the training data upon building the trees.[17] The number usually dictates the percentage of randomly selected samples used in constructing a tree.

#### **Colsample\_bytree**

This parameter randomly selects the number of columns/features in the training process while building the tree. Similar to the subsample, the number dictates the percentage of randomly selected features of training before constructing a tree.[17]

#### **Eval\_metric**

This is used to measure the quality of the statistical points of the model.[17] There are different types of evaluation metrics to measure the statistics of the ML model.

#### **Verbosity**

It is one of the most used general parameters. It is a log message to keep track of the progress of the work. Here 0 means silent, 1 is warning which is the default, 2 is info and 3 is debug.[17]

#### **Label\_encoder**

It refers to the conversion of labels which converts the numerical values to a more machine-readable form.[17] This allows the ML to decide how the labels are to be operated in a better way. It is a necessary preprocessing step in the dataset.

### 3.5.4 Hyperparameter Tuning

For the model, it had to be tweaked on some hyperparameters of the XGBoost model. The tuning of the parameters was done to produce a more accurate and refined output.[17] Here is how the tuning was done to the parameters.

```

model_xgboost_fin = xgboost.XGBClassifier(learning_rate=0.1,
max_depth=5,
n_estimators=1000,
subsample=0.50,
colsample_bytree=0.50,
eval_metric='auc',
verbosity=1,
use_label_encoder=False)

```

Figure 3.11: Hyperparameter Tuning Values

	rank_test_score	mean_test_score	mean_train_score	param_learning_rate	param_max_depth	param_n_estimators
33	1	0.842105	1.0	0.1	5	1000
30	1	0.842105	1.0	0.1	3	1000
31	3	0.841422	1.0	0.1	3	2000
34	3	0.841422	1.0	0.1	5	2000
35	5	0.840055	1.0	0.1	5	3000
32	5	0.840055	1.0	0.1	3	3000
15	7	0.838688	1.0	0.03	5	1000
12	7	0.838688	1.0	0.03	3	1000
0	9	0.837321	1.0	0.02	2	1000
9	10	0.836979	1.0	0.03	2	1000
13	10	0.836979	1.0	0.03	3	2000

Figure 3.12: Hyperparameter Tuning

### 3.6 VGG16

VGG16 classifies items. It classifies 1000 images into 1000 groups with 92.7 percent accuracy. Transfer learning is simple with this method of grouping photos. VGG16 has 16 weighted layers. VGG16 has 21 layers: 13 convolutional layers, 5 Max Pooling layers, 3 Dense layers. Only 16 are weight layers, also known as learnable parameters layers. VGG16's 224 by 244 input tensor features three RGB channels. The most unusual feature about VGG16 was that it preferred convolution layers of a 3x3 filter with stride 1 over a large number of hyper-parameters and always used the same padding and maxpool layers of a 2x2 filter with stride 2.

The architecture always orders the convolution and max pool layers. Conv-1, Conv-2, and Conv-3 each have 64, 128, and 256 filters. Conv-1 has 64 filters, Conv-2 128 filters, Conv-3 256 filters, Conv-4 and Conv-5 512 filters. Three FC layers follow convolutional layers. 1000-channel ILSVRC classification is done by the third. The first two FC levels have 4096 channels (one for each class). Sigmoid is last.

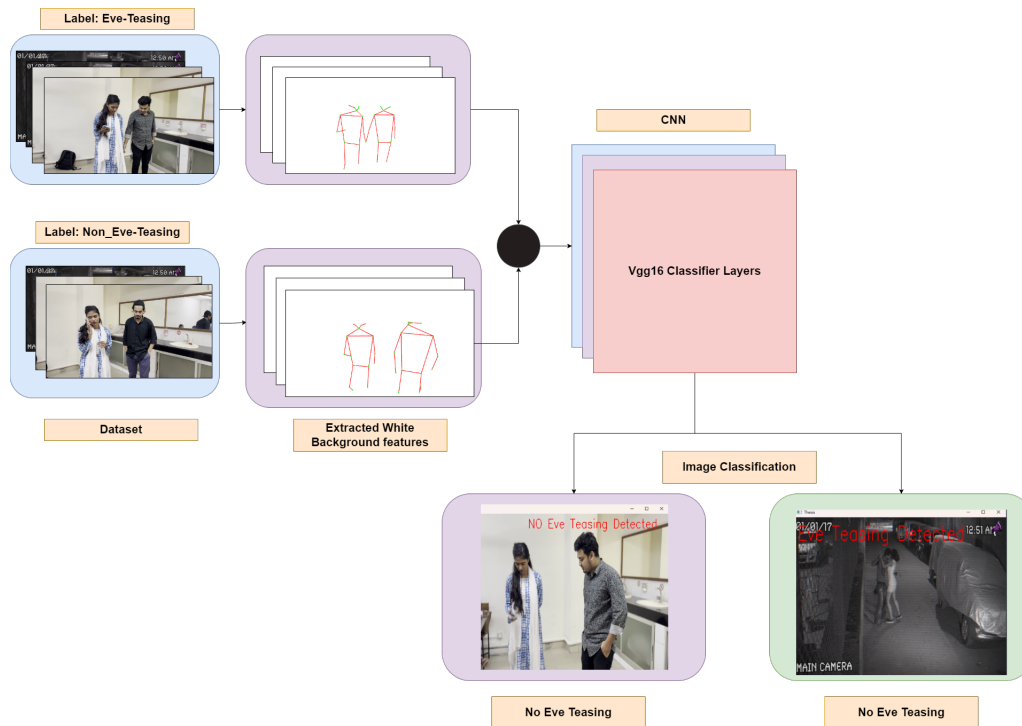


Figure 3.13: Workflow of VGG16 Model

### 3.6.1 Data Preprocessing

We did it by importing the required libraries. It is necessary to conduct an exploratory data analysis (EDA) of the dataset in a comprehensive manner. Checking the image size and dimension will allow you to identify any photographs that are an anomaly due to their unusually huge or small dimensions. Check for any extra EDA features that are associated with the image, such as blurriness, whiteness, aspect ratio, and so on. Import the dataset, then once it has been imported, normalize the data so that the VGG16 model can interpret it. There is a wide variety of body styles, wheelbase lengths, and pixel densities included in the Stanford automobile dataset. We change the value of the image input tensor to 224 so that we can utilize the VGG16 model. ImageDataGenerator's primary objective is to streamline the process of incorporating labeled data into the model. It is a very useful class because of the numerous operations that it can perform, such as rescaling, rotating, zooming, and flipping images. The fact that this class does not have any effect on the data that is stored on the disk is one of its strongest selling points. This class makes modifications to the data in real time while it is being provided to the model. The ImageDataGenerator will assign labels to each and every piece of information included within the folder automatically. Because of this, feeding input into the neural network is a simple process.

### 3.6.2 Modeling

In this approach, we build a fresh dataset from our input photos using the architecture of the pre-trained model. The Convolutional and Pooling layers will be imported, however the model's "top portion" will not be imported (the Fully-Connected layer).

Its learned weights and convolutional layers are capable of detecting generic feature. Give the VGG16 model a definition as a sequential model. In order to prevent all negative values from being passed on to the following layer, we also apply relu (Rectified Linear Unit) activation to each layer. We flatten the vector that results from the convolutions before passing the data to the dense layer after completing all the convolutions. 512 units of dense layer, and 1 unit of the dense sigmoid layer. The keypoints extracted from the previous model were used. Only the keypoint was excluded and plotted on a white background. Here is a demo of the dataset after plotting it.

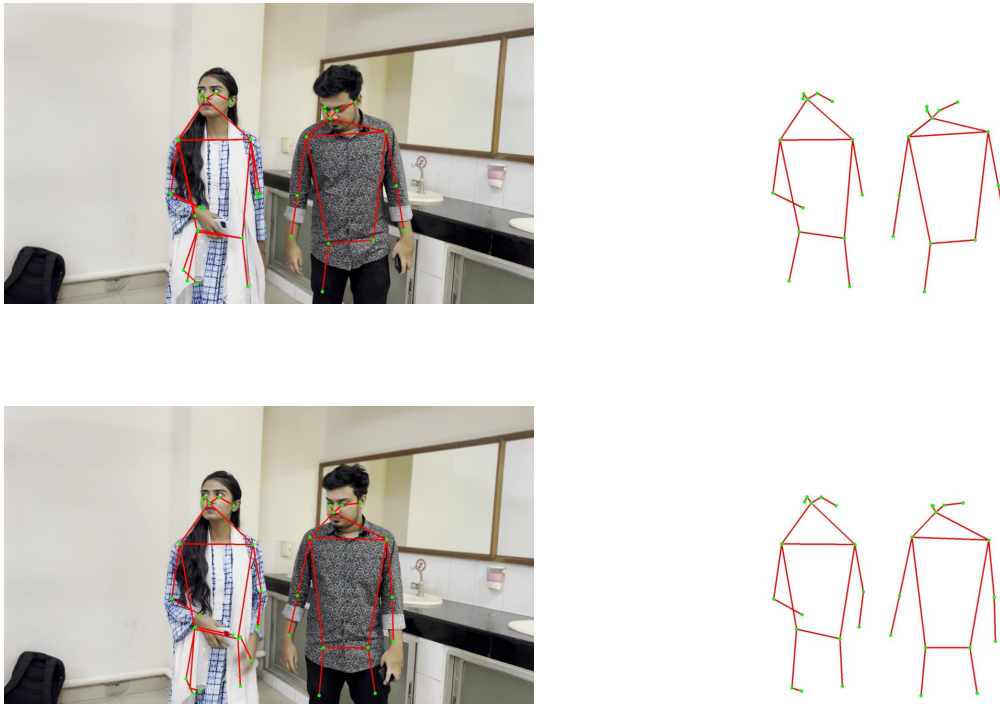


Figure 3.14: Plotting Keypoints on a white background

The output layer dimension should be changed to reflect the number of classes in the line below. Since there are 2 classes in the dataset, this information is also included in the output layer. Additionally, the activation function sigmoid is used because this algorithm classifies objects. Based on the model's confidence that the photos belong to a particular class, the softmax layer will produce a value between 0 and 1 for the output. Set the early ending option so that the training can end when the model's accuracy reaches the maximum level compared to that of the previous iterations. used the SGD (stochastic gradient descent) optimizer (although ADAM could also be used). Loss "Because the model has 2 classes, binary cross-entropy was used. If there are only two classes, binary cross-entropy can be used. The optimizer's learning rate is specified; in this instance, it is set to 1e-6. In order to obtain global minima, we must lower the learning rate if our training exhibits significant epoch-level bouncing. We classified data using pre-trained weights. Transfer learning is a technique used to avoid the need for extensive retraining and to save time and money. The creation of a model checkpoint allows us to save just the best models for later testing and model validation. At 20 epochs, the model was stopped due to early learning after achieving an accuracy of 85%. The model's training/validation accuracy can be seen after it has been trained. Pre-processing the image and passing



it to the model for output are required in order to make predictions using the trained model. To evaluate the model, use the saved version and load it. Next compile the model, the optimizer, and the loss metric.

### 3.7 Transfer Learning

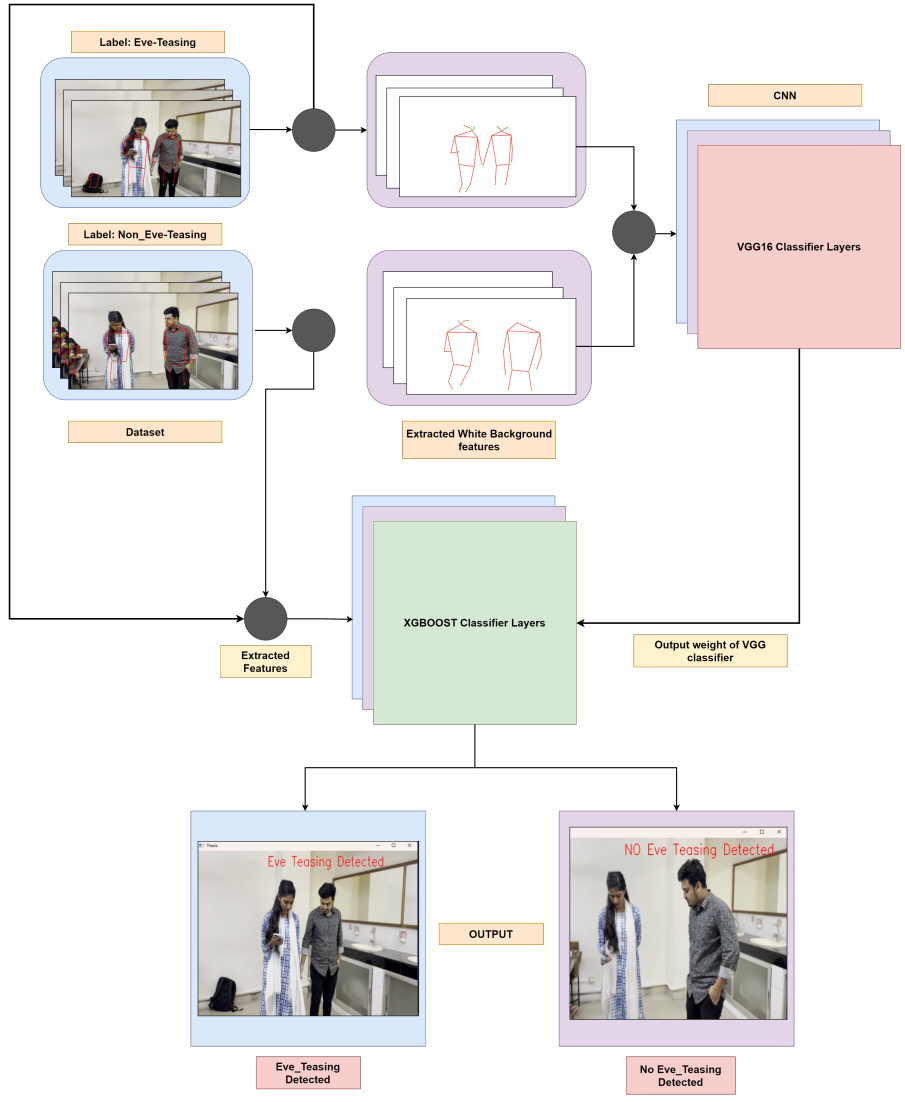


Figure 3.15: Workflow of Transfer Learning

Transfer Learning is a approach for machine learning where gained knowledge is stored while solving one problem and applying it to a different but related problem. In traditional transfer learning, the weight of the neural network of a pretrained model in a given scenario is stored and feed into another model for the purpose of getting a better prediction. In our approach, we didn't follow the traditional transfer learning approach. Rather, we propose a different approach in our scenario. We took the output weight rather than the total neural network of VGG16 and feed it into XGBoost. After extracting the posturer of the persons using mobile-net, we plotted the posture on a white background and use those images of the posture to label eve teasing and not eve teasing for training the VGG16. Furthermore, after detecting

using the VGG16 we took the given output of a scenario and stored it. Again, using the mobile-net model, extracting the posture's coordinates this time and converting them to norm and polar coordinates we got the extracted features of the postures. We got exactly 136 features from every frame. Now, along with those features, one more feature is added and that is the output weight of VGG16 prediction that were stored earlier. Now, with those 137 features combined we trained the XGBoost and got a better accuracy and precision overall.

# Chapter 4

## Implementation & Result Analysis

### 4.1 Result Analysis of Gender Detection

The findings from the experiments are partitioned into two distinct categories here. The findings were examined statistically in the first phase, and matplotlib was used in the second phase to illustrate the accuracy and loss of the model. Our analysis led us to the conclusion that it is possible to determine gender based on video evidence. The implementation of the proposed gender classification model is described in this section. Tensor flow 2.8.0 was used to implement and test the model. In this method, Conv2D layers make up our initial two layers. These convolution layers will deal with our 2-dimensional matrices as input images. It will contain 3x3 kernel sizes, and it can contain 32, 64, and 128 filters, in that order. In these layers, the resulting volume's spatial dimensions are then reduced via Max pooling. It gives an overview of the features in a part of the feature map that was made by a convolution layer. Then we used a 0.2 dropout to prevent the training data from overfitting. Then we used flattening layers and dense layers after the convolutional layers and before the output layer, go to fully connected layers. We tried using different combinations of layers until we got our expected results. keeping our epochs to only 20 as we started getting fine accuracy after 12 to 13 iterations.

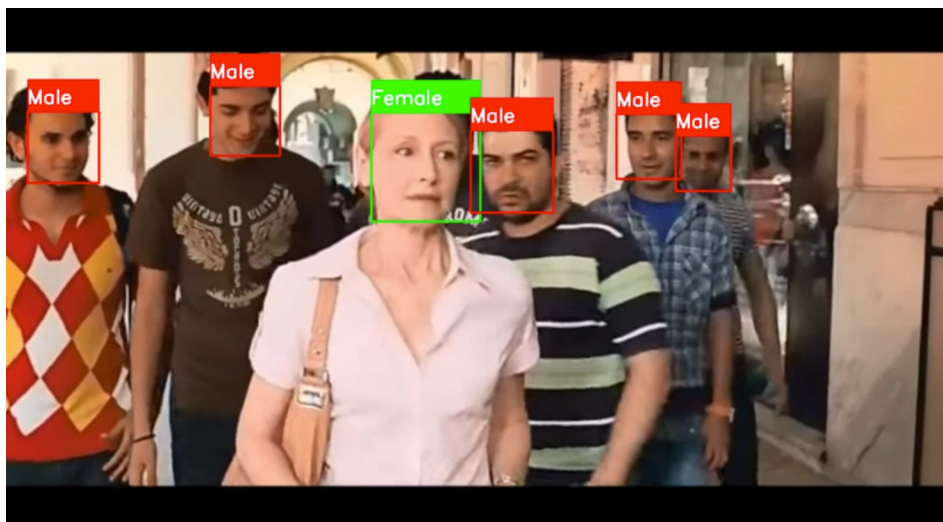


Figure 4.1: Gender Detection in Good Light

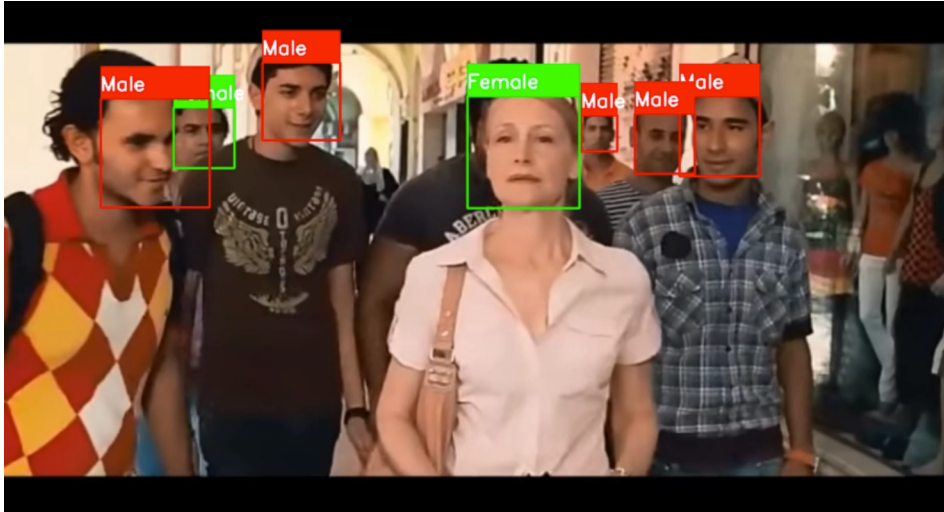


Figure 4.2: Gender Detection in low Light

As it is seen from the images, our model is able to differentiate male and female in a frame and label it accordingly. The model performs very well under good lighting conditions. But it seems in the second picture the model identifies the man in the back as female. It is because that part of the frame has not a good lighting condition. So the model identifies the male as a female. It is a limitation of the model.

#### 4.1.1 Evaluating Model Performance

Precision	Recall	Accuracy	Specificity	F1 score
0.95013231039	0.9526531	0.9513265491	0.9979592	0.95139102

Figure 4.3: Score Analysis of the Model

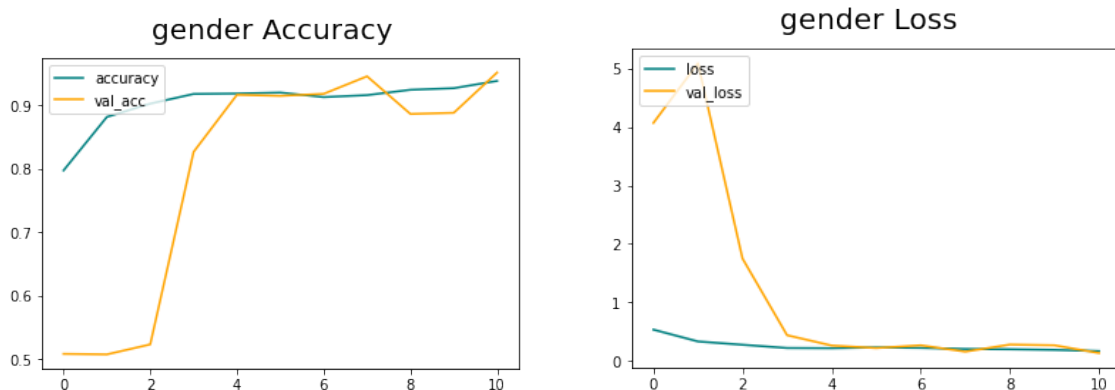


Figure 4.4: Gender Detection Accuracy and Loss

```

Model: "sequential_1"
-----
Layer (type)                Output Shape                Param #
-----
conv2d (Conv2D)              (None, 30, 30, 32)         320
conv2d_1 (Conv2D)            (None, 28, 28, 32)         9248
max_pooling2d (MaxPooling2D) (None, 14, 14, 32)         0
conv2d_2 (Conv2D)            (None, 12, 12, 64)         18496
conv2d_3 (Conv2D)            (None, 10, 10, 64)         36928
max_pooling2d_1 (MaxPooling2D) (None, 5, 5, 64)          0
dropout (Dropout)            (None, 5, 5, 64)          0
flatten (Flatten)            (None, 1600)                0
dense (Dense)                 (None, 64)                  102464
dropout_1 (Dropout)           (None, 64)                  0
dense_1 (Dense)               (None, 2)                   130
-----
Total params: 167,586
Trainable params: 167,586
Non-trainable params: 0
-----

```

Figure 4.5: Summary of our Sequential Model

## 4.2 Result Analysis of XGBoost

The XGBoost implementation was done and the accuracy was satisfactory. The result of the XGBoost model is given statistically and also graphically.

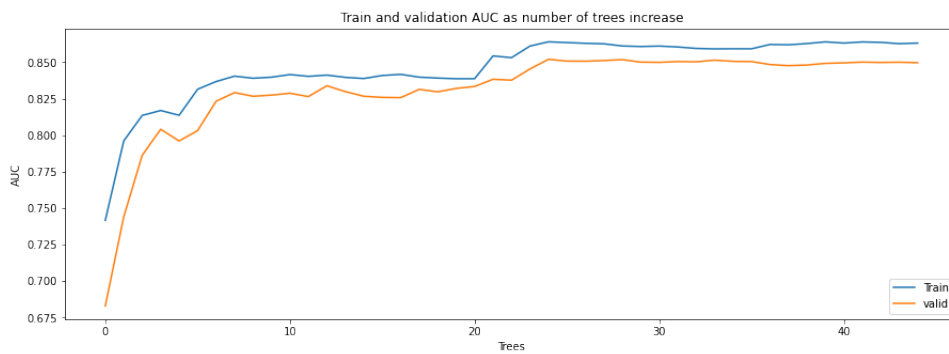


Figure 4.6: XGBoost train and validation result

Here, the blue line denotes the training data and the orange line denotes the validation data. It has trees on its x-axis and the area under the curve on the y-axis. As the tree increases the validation and training data increase in a similar rate.

Precision	Recall	Accuracy	Specificity	F1 score
0.93939393	0.3690476	0.86005092	0.486254	0.5833333

Figure 4.7: XGBoost score analysis

From the table, the scores of the model are satisfactory. We see that the model got and accuracy of 0.86005092 or 86%. The precision is 0.939393. The recall value is 0.3690476. The specificity is 0.486254. lastly, the f1 score that this model produces is 0.5833333.

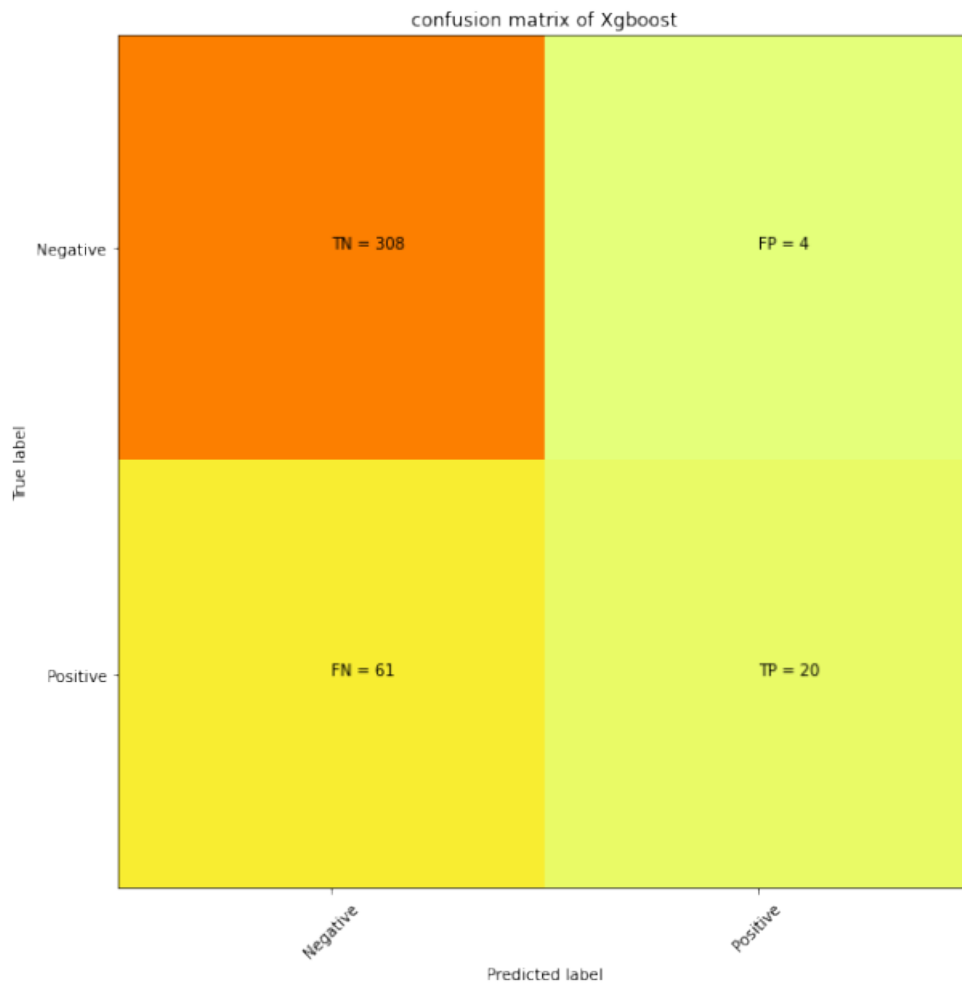


Figure 4.8: XGBoost confusion matrix

The confusion matrix has a true negative of 308 and a false positive of 4. And also, the false negative is 61 and the true positive is 20. Thus, the model is very good at correctly predicting the positive class and also has a low number of instances of incorrectly predicting the positive class. And it has a higher number of correct predictions of negative classes than incorrect predictions of negative classes.

## 4.3 Result Analysis of CNN

In the first phase, results were analyzed through a confusion matrix to visualize the accuracy and loss of the model. And our model came to the conclusion that it detected Eve's teasing from video footage.

### 4.3.1 Accuracy

The accuracy is a measurement of how well the algorithm is classifying the prediction correctly. It is calculated by the number of correct predictions divided by the number of total prediction.  $\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{FP} + \text{TN} + \text{FN})$ . Our model had an accuracy of 0.85294115, or 85%.

### 4.3.2 Specificity

A classifier's specificity is the difference between the number of times it correctly identifies negative data and the number of times it actually finds negative data. It is used in situations where the categorization of negatives is important. We achieved Specificity: 0.9813432693481445

### 4.3.3 Precision

The level of precision can be measured by determining, out of all the positives, how many were correctly labeled as positive.

$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$  We achieved Precision: 0.8774703741073608

### 4.3.4 Recall

Recall and sensitivity are identical. We achieved Recall: 0.8283582329750061  $\text{Recall} = \text{TruePositives} / (\text{TruePositives} + \text{FalseNegatives})$

### 4.3.5 F1 score

The f1 score is a measure of how well the model can classify things. It is found by finding the harmonic mean of the model's precision and recall. People think that the F1 score gives a more accurate picture of how well the classifier works than the standard accuracy metric.  $\text{F1 score} = 2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$   
As a result, Our F1 score: 0.852207284544955

### 4.3.6 Evaluating Model Performance

The blue in the learning curves for the VGG-16 represents train accuracy and the orange, actual accuracy.

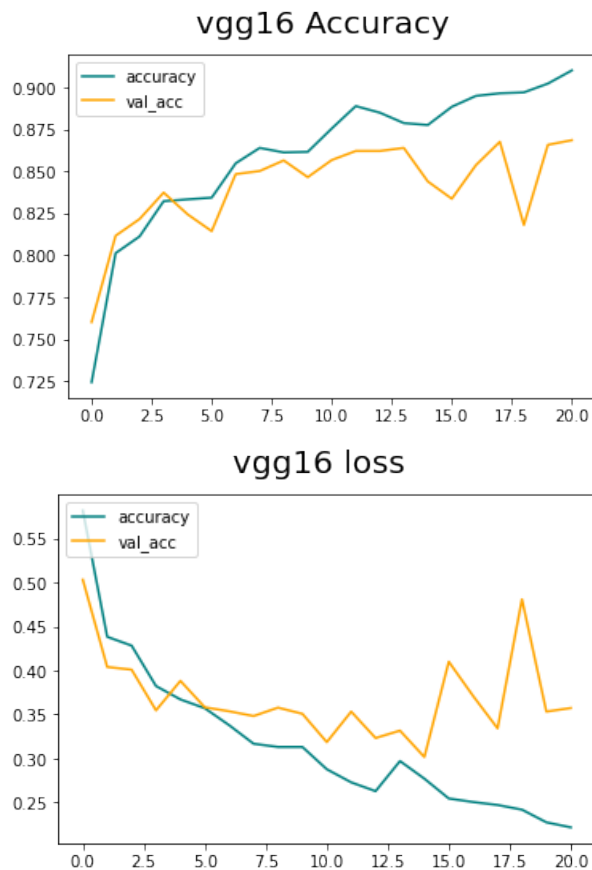


Figure 4.9: VGG16 Accuracy and Loss

### Confusion matrix

An evaluation of a classification algorithm's performance is done using a table called a confusion matrix. The output of a classification algorithm is shown and summarized in a confusion matrix. Figure 4.6 displays a confusion matrix showing in categorization issues to determine where model errors occurred.

Here,

True Positive (TP) = the number of times the model correctly predicted the positive class.

False Positive (FP) = the number of times the model incorrectly predicted the positive class.

False Negative (FN) = the number of times the model incorrectly predicted the negative class.

True Negative (TN) = the number of times the model correctly predicted the negative class.



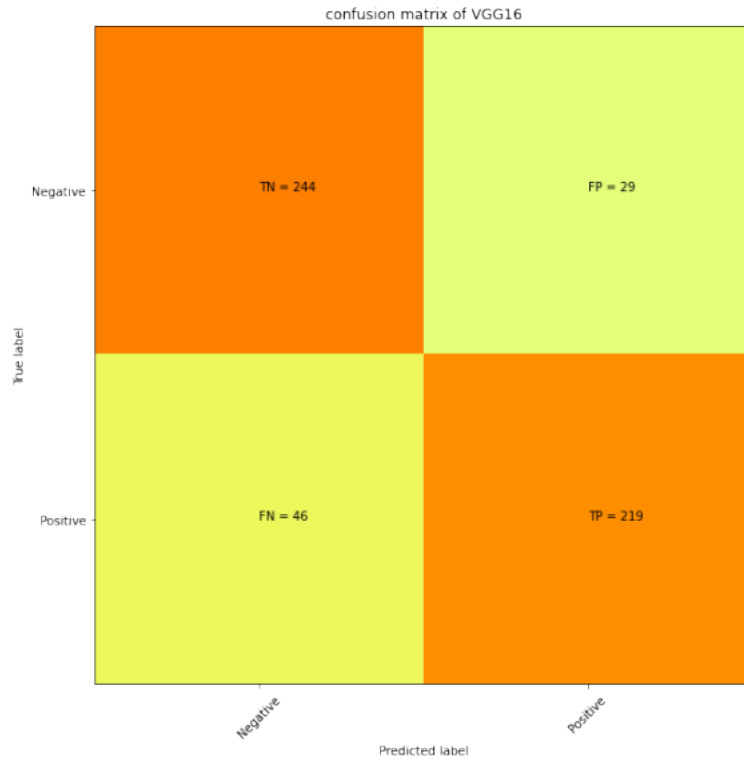


Figure 4.10: VGG16 confusion matrix

From the confusion matrix, the data that we get is the true negative, which is 244, which means the model can correctly predict negative classes more than it can incorrectly predict, which is 46. So it has a good prediction rate. Moreover, the true prediction of positive classes is 219, which is higher than the incorrect positive prediction of 29.

## 4.4 Result analysis of Transfer Learning

The result that we got after using the weight of the VGG16 model and the XGBoost classifier are listed below.

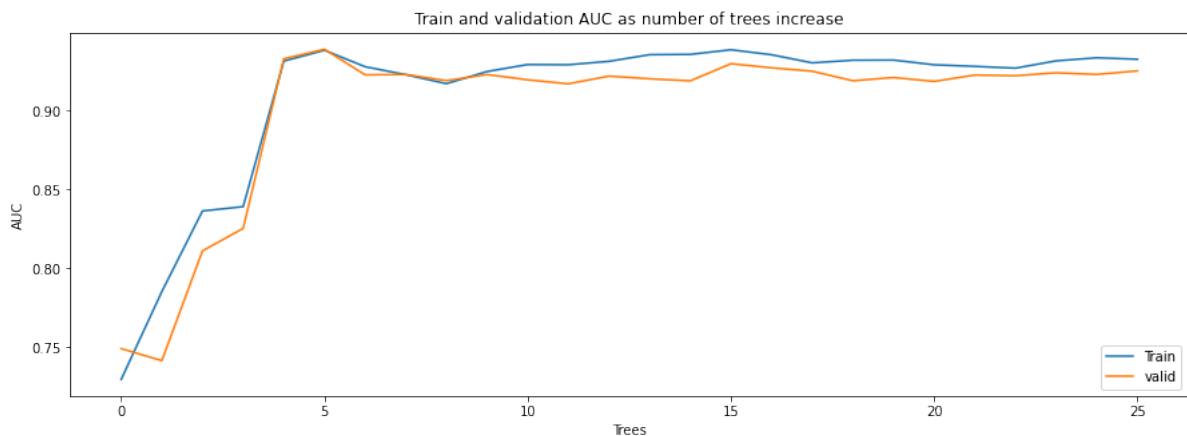


Figure 4.11: Transfer learning train and validation result

Here, the graph shows the validation and training data result in terms of trees increasing. The blue line denotes the training data and the orange line denotes the validation data. It has trees on its x-axis and the area under the curve on the y-axis. As the tree increases the validation and training data increases at a similar rate.

Precision	Recall	Accuracy	Specificity	F1 score
0.933333337	0.51851850	0.89312977	0.77564102	0.666666666

Figure 4.12: Transfer learning score analysis

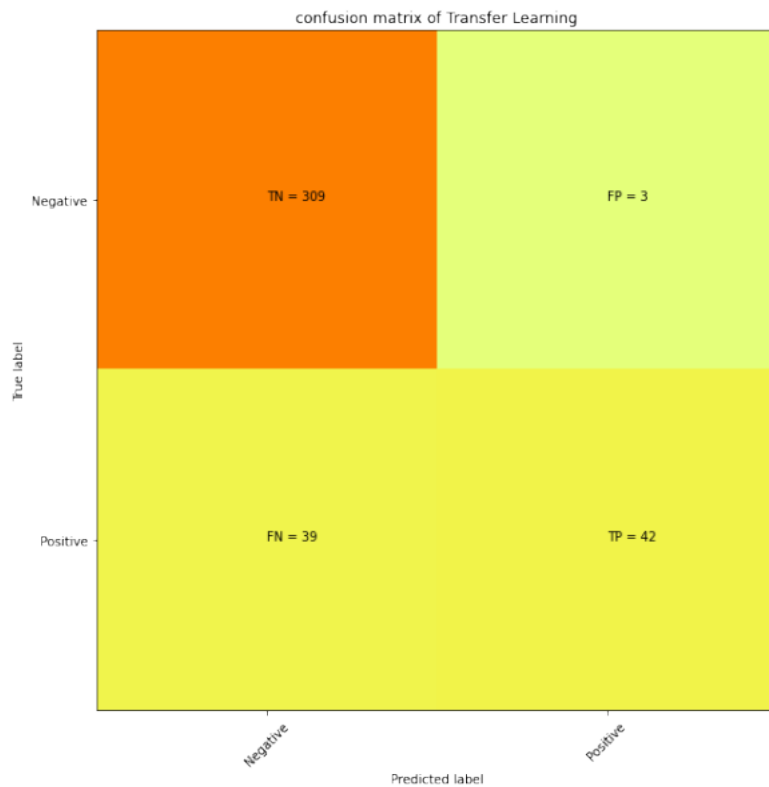


Figure 4.13: Transfer learning confusion matrix

The confusion matrix has a true negative of 309 and a false positive of 3. And also, the false negative is 39 and the true positive is 42. Thus, the model is very good at correctly predicting the positive class and also has a low number of instances of incorrectly predicting the positive class. And it has a higher number of correct predictions of negative classes than incorrect predictions of negative classes.

## 4.5 Final Result

On the first attempt, by training XGBoost model with 1962 datasets and splitting them into 0.8% train and 0.2% test data of the 136 features extracted from the posture of both male and female in one frame we got 86% accuracy. As it was not as bad, but we were getting more false-negative detection as shown in the confusion matrix. For improving the performance of XGBoost, the output weight or the detection accuracy is feed into XGBoost as a feature, along with the 136 posture

Model Name	Precision	Recall	Accuracy	Specificity	F1_score
Transfer learning	0.933333337	0.51851850	0.89312977	0.77564102	0.666666666
XGBoost	0.93939393	0.3690476	0.86005092	0.486254	0.5833333
VGG16	0.87747037	0.82835823	0.85294115	0.98148149	0.8522072845

Figure 4.14: Score comparison of all the models

features. Now, total 137 features is given in XGBoost to train. Same as before, train 80% and test 20% split. We got an increase accuracy of 89.31%. Furthermore, the confusion matrix also shows false-negative is now 39, nearly cut in half than before. Using the new approach of using the transfer learning of feeding the model with output data boosted the performance of XGBoost greatly for detecting eve-teasing.



Figure 4.15: Eve teasing Detection

# Chapter 5

## Conclusion

The algorithm that we have developed will be utilized to analyze video footage in search of cases of eve-teasing. By using the algorithm, any suspicious behavior as well as one of the appropriate offenders will be identified. Within the scope of this paper, we have discussed our methodology for developing a functional and effective algorithm for this cause. We were successful in determining the gender of a person in a video clip and correctly annotating it. As a result of this, we are able to create a bounding box around humans and then run our other algorithms on the locations within that box. Our model produces fully accurate results. The next step was to test out two different models for figuring out a person's posture: Mediapipe and MoveNet. multi posture. Both have certain limitations, but MoveNet functioned better at detecting various poses and delivered more accurate output. MoveNet also has some limitations. In the research that we did, we were able to pinpoint 17 important aspects that differentiate gesture from posture. The 17 keys were plotted against a white background, and the results were given to VGG16. In addition, we used the XGBoost technique to train it with the norms and polar additions that we had calculated for each of the 17 keypoints. After that, we combined the 136 features that were derived from posture detection with the output weight that we had previously obtained from VGG16. Therefore, as of right now, we have 137 features. In the end, we were able to contribute 137 features to XGBOOST. By doing things this way, we were able to incorporate all of the algorithms into one training model in order to identify eve teasing.

## Chapter 6

### Limitation and Future direction

The most challenging task is to detect gender in a blurry image. As most CCTV footages are not of good quality, thus making it harder for detecting the genders as we detect gender using face. We perceive, using full body and posture to detect gender is required. Also, our proposed method can only detect eve-teasing between two people. The model has another limitation, that is it cannot differentiate between the type of harassment that is happening. Eve-teasing has many types for example it can be verbal abuse, it can be of showing bad gestures toward a female, it can be making bad sounds, and also calling bad names, etc. The approach, we have taken is not so extensive in differentiating different types of eve-teasing. We focused mainly on the touching part and the posture detection part when there is a potential case of unwanted touching or grabbing. The model only detects and tells if there is eve teasing in a video frame. But it does not have a system to prevent the eve teasing from happening. For example, it does not trigger any alarm or send an alert to any security organization to help the victim in that situation. Rather the system is able to identify if the occurrence happened or not. We feel more data is required with more scenarios of eve-teasing to do a more extensive study for building a better model to be able to detect every type of eve-teasing along with detecting all the victims and perpetrators involved in each scenario. Hopefully, there will be more work on this topic and we will get more use out of this model that we tried our best to implement. We dream to see a sophisticated system that will detect even wrongful gestures toward women to make the world a better place for everyone.

# Bibliography

- [1] S. Ioannou, G. Caridakis, K. Karpouzis, and S. Kollias, “Robust feature detection for facial expression recognition,” *EURASIP Journal on Image and Video Processing*, vol. 2007, pp. 1–22, 2007.
- [2] L. Cao, M. Dikmen, Y. Fu, and T. S. Huang, “Gender recognition from body,” in *Proceedings of the 16th ACM international conference on Multimedia*, 2008, pp. 725–728.
- [3] I. Maglogiannis, D. Vouyioukas, and C. Aggelopoulos, “Face detection and recognition of natural human emotion using markov random fields,” *Personal and Ubiquitous Computing*, vol. 13, no. 1, pp. 95–101, 2009.
- [4] B. Gupta, S. Gupta, and A. K. Tiwari, “Face detection using gabor feature extraction and artificial neural network,” *Proceedings from IS CET*, pp. 18–23, 2010.
- [5] Z. Zhang, Y. Liu, A. Li, and M. Wang, “A novel method for user-defined human posture recognition using kinect,” in *2014 7th International Congress on image and signal processing*, IEEE, 2014, pp. 736–740.
- [6] A. De and A. Saha, “A comparative study on different approaches of real time human emotion recognition based on facial expression detection,” in *2015 International Conference on Advances in Computer Engineering and Applications*, IEEE, 2015, pp. 483–487.
- [7] M. A. Islam and T. Amin, “Eve teasing in bangladesh: An overview,” *Legal Service India - Law, Lawyers and Legal Resources*, 2016.
- [8] R. S. Sidhu and M. Sharad, “Smart surveillance system for detecting interpersonal crime,” in *2016 International Conference on Communication and Signal Processing (ICCSP)*, IEEE, 2016, pp. 2003–2007.
- [9] S. Bhattacharjee and G. Somashekhar, “Artificial intelligence to impart surveillance, tracking, actuation on suspicious activities,” in *2017 IEEE 7th International Advance Computing Conference (IACC)*, IEEE, 2017, pp. 1–5.
- [10] J. Brownlee, “Gentle introduction to vector norms in machine learning,” *Machine Learning Mastery*, 2018.
- [11] A. Admin, “Why xgboost? and why is it so powerful in machine learning,” *Abzooba*, 2019.
- [12] S. Ghoneim, “Accuracy, recall, precision, f-score specificity, which to optimize on?” *Towards Data Science*, 2019.
- [13] J. Agarkhed, A. Rathi, F. Begum, *et al.*, “Women self defense device,” in *2020 IEEE Bangalore Humanitarian Technology Conference (B-HTC)*, IEEE, 2020, pp. 1–5.

- [14] A. Chauhan, “Gender classification dataset,” *Multi-Person Pose Estimation with Python*, 2020.
- [15] F. Chollet, “The sequential model,” *Multi-Person Pose Estimation with Python*, 2020.
- [16] V. Hyndavi, N. S. Nikhita, and S. Rakesh, “Smart wearable device for women safety using iot,” in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, IEEE, 2020, pp. 459–463.
- [17] S. Raju, “Everything you need to know about xgboost,” *Medium*, 2020.
- [18] ActiveState, “What is a keras model and how to use it to make predictions,” *What is a Keras model and how to use it to make predictions*, 2021.
- [19] A. R. Naik, “Inside movenet, google’s latest pose detection model,” *DEVELOPERS CORNER*, 2021.
- [20] D. Tanwar, V. Nijhawan, P. Sinha, and R. Gupta, “Design of low-cost women safety system using gps and gsm,” in *2021 8th International Conference on Computing for Sustainable Global Development (INDIACom)*, IEEE, 2021, pp. 827–831.
- [21] A. P. Gulati, “Pose detection in image using mediapipe library,” *Analytics Vidhya*, 2022.
- [22] M. R. Islam, “Eve teasing as a social problem: A study on the school girls in dhaka city, bangladesh,”
- [23] H. Singh, “Eve teasing in india,” *Legal Service India - Law, Lawyers and Legal Resources*,