

**CUSTOMER ANALYSIS WITH MACHINE
VISION**

TIONG WEI JIE

UNIVERSITI TUNKU ABDUL RAHMAN

CUSTOMER ANALYSIS WITH MACHINE VISION

TIONG WEI JIE


**A project report submitted in partial fulfilment of the
requirements for the award of Bachelor of Mechatronics
Engineering with Honours**

**Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman**

May 2023

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature :  _____

Name : Tiong Wei Jie


ID No. : 1805100

Date : 20/5/2023

APPROVAL FOR SUBMISSION

I certify that this project report entitled “**CUSTOMER ANALYSIS WITH MACHINE VISION**” was prepared by **TIONG WEI JIE** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Mechatronics Engineering with Honours at Universiti Tunku Abdul Rahman.

Approved by,

Signature :  _____

Supervisor : Tee Yee Kai _____

Date : 20/05/2023 _____

Signature : _____

Co-Supervisor : _____

Date : _____

The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of Universiti Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© 2023, Tiong Wei Jie. All right reserved.

ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my research supervisor, Dr. Tee Yee Kai for his invaluable advice, guidance and his enormous patience throughout the development of the research.

In addition, I am deeply grateful to my caring parents and friends for their love, encouragement and help. Their generous support and assistance were instrumental in bringing this project to fruition. I am truly blessed to have their unwavering support.

Once again, I would like to extend my sincere gratitude to each and every individual who played a part in making this project a success. Thank you all.

ABSTRACT

CCTVs usually installed in a business establishment can yield additional customer information, providing valuable insights for marketing analysis. However, manually analyzing the sheer volume of videos can be taxing for humans. Therefore, this study endeavors to develop a computer-vision solution that automates customer analysis on CCTV videos. The proposed solution must be able to fulfil the requirements for customer counting, customer recognition and gender classification.

This study aimed to improve the human detection model by eliminating the imperfections in existing models that have a high false rate in detecting the cartoons as humans. These cartoons may be human-like stickers that are placed around retail shops, and false detection may result in inaccurate customer analysis. To evaluate the performance of existing detection models, metrics such as accuracy, precision, recall, F1 score, false detection rate, model size, and parameters are used. To address the issue, the latest algorithms, such as YOLOv5, YOLOv8 and mobilenet ssd, were selected for retraining. The retraining process involved utilization of a dataset consists of 2 classes: human and cartoon, with 11k images per class. The instances in the dataset were well labelled before splitting into train, validation and test sets. Each selected model is then retrained, evaluated and compared to the existing models. The study found that the best model is the retrained YOLOv8n, which achieved a false detection rate of 8.16 %, outperforming all the pretrained models. Meanwhile, it has enhanced the accuracy and F1 score in human detection, improving by 5.38 % and 2.85 % respectively when compared to the best pretrained model, YOLOv8m. Hence, the retrained YOLOv8n has been selected as the human detection model for the proposed solution.

When the retrained YOLOv8n detects a customer in the CCTV video, human tracking takes place to track the customer. When the customer passes through a counting line drawn by the system, customer counting occurs, and the system will crop their faces for facial recognition and gender classification.

Due to time constraints, several components and algorithms could not be addressed in this study. Future work will focus on improving facial recognition and proposing new methods to explore different approaches.

TABLE OF CONTENTS

DECLARATION		i
APPROVAL FOR SUBMISSION		ii
ACKNOWLEDGEMENTS		iv
ABSTRACT		v
TABLE OF CONTENTS		vi
LIST OF TABLES		viii
LIST OF FIGURES		x
LIST OF SYMBOLS / ABBREVIATIONS		xiii
CHAPTER		
1	INTRODUCTION	1
1.1	General Introduction	1
1.2	Importance of the Study	1
1.3	Problem Statement	3
1.4	Aim and Objectives	4
1.5	Scope and Limitation of the Study	4
2	LITERATURE REVIEW	6
2.1	Human Detection	6
2.1.1	Face feature based	6
2.1.2	Motion Feature based	8
2.1.3	Body appearance based	9
2.1.4	Deep Learning based	10
2.2	Human Tracking	19
2.3	Facial Recognition	21
2.4	Gender classification	27
3	METHODOLOGY AND WORK PLAN	31
3.1	Improvement of Human Detection	31
3.1.1	Testing Dataset	32
3.1.2	Pretrained Object Detection Models	33

3.1.3	Evaluation Metrics	35
3.1.4	TP, FP and FN of Human	37
3.1.5	False Detection of Cartoons as Humans	39
3.1.6	Model Training	41
3.2	Overall Framework of Customer Analysis with Machine Vision	48
3.2.1	Human Detection, Tracking and Customer Counting	50
3.2.2	Face Detection, Face Recognition and Gender Classification	57
4	RESULTS AND DISCUSSION	61
4.1	Human Detection Improvement	61
4.1.1	Pretrained Models	61
4.1.2	Retrained Models	67
4.1.3	Summary of Pretrained and Retrained Models	72
4.1.4	Retrained Models: Classification of Result Based on 2 Classes	72
4.1.5	YOLOv8 series	78
4.2	Overall Result	81
4.2.1	Input Videos and Customer Database	81
4.2.2	Human Detection, Human Tracking and Customer Counting	82
4.2.3	Facial Recognition	83
4.2.4	Gender Classification	89
4.2.5	Updated Customer Database	90
4.2.6	Weakness of the Algorithm	90
5	CONCLUSIONS AND RECOMMENDATIONS	93
5.1	Conclusions	93
5.2	Recommendations for future work	94
	REFERENCES	95

LIST OF TABLES

Table 3.1:	List of pretrained detection models used for human detection.	34
Table 3.2:	Hardware specifications of Google Colab platform provided.	44
Table 4.1:	Result of the pretrained models.	61
Table 4.2:	Result of pretrained model.	62
Table 4.3:	Result for retrained models.	67
Table 4.4:	Result for retrained models.	68
Table 4.5:	Confusion Matrix of 2 classes detection for YOLOv5n.	72
Table 4.6:	Confusion Matrix of 2 classes detection for YOLOv5s.	73
Table 4.7:	Confusion Matrix of 2 classes detection for YOLOv5m.	73
Table 4.8:	Confusion Matrix of 2 classes detection for YOLOv8n.	73
Table 4.9:	Confusion Matrix of 2 classes detection for YOLOv8s.	73
Table 4.10:	Confusion Matrix of 2 classes detection for YOLOv8m.	74
Table 4.11:	Confusion Matrix of 2 classes detection for Mobilenet ssd.	74
Table 4.12:	Result of the 2 classes detection.	74
Table 4.13:	Result of YOLOv8 series on mix dataset.	78
Table 4.14:	Result of YOLOv8 series on mix dataset.	79
Table 4.15:	Result of Face Detection in Entrance Video.	84
Table 4.16:	Result of Face Detection in Exit Video.	84
Table 4.17:	Result of Face Recognition in Entrance Video.	85
Table 4.18:	Result of Face Recognition in Exit Video.	86
Table 4.19:	Overall Combined Result of Facial Recognition in Entrance Video.	87

Table 4.20:	Overall Combined Result of Facial Recognition in Exit Video.	88
Table 4.21:	Confusion Matrix of Gender Classification in Entrance Video.	89
Table 4.22:	Confusion Matrix of Gender Classification in Exit Video.	89

LIST OF FIGURES

Figure 2.1:	General framework of human detection. (Ansari and Singh, 2021)	6
Figure 2.2:	General framework of R-CNN. (Ansari and Singh, 2021)	11
Figure 2.3:	Overall framework of Fast R-CNN. (Ansari and Singh, 2021)	12
Figure 2.4:	Overall workflow of Faster R-CNN. (Ansari and Singh, 2021)	13
Figure 2.5:	Overall framework of YOLO. (Ansari and Singh, 2021)	14
Figure 2.6:	Network architecture of YOLOv5. (Xu, et al., 2021)	15
Figure 2.7:	Architecture of SSD. (Ansari and Singh, 2021)	17
Figure 2.8:	Convolutional feature pyramid based on backbone network ConvNext. (Lechen, 2022)	18
Figure 2.9:	Demonstration of MOSSE tracker. (Bolme, et al., 2010)	20
Figure 2.10:	Face image under lightning variation. (Adjabi, et al., 2020)	21
Figure 2.11:	Face bunch graph. (Adjabi, et al., 2020)	23
Figure 2.12:	Example calculation of LBP. (Adjabi, et al., 2020)	24
Figure 2.13:	Steps of 1DLBP. (Adjabi, et al., 2020)	26
Figure 2.14:	Example of landmarks applied. (Rai and Khanna, 2012)	28
Figure 2.15:	SIFT key points. (Ng, Tay and Goi, 2015)	29
Figure 2.16:	CNN architecture. (Levi and Hassner, 2015)	30
Figure 3.1:	General steps for improving human detection model.	31
Figure 3.2:	Portion of the human and cartoon testing dataset.	32
Figure 3.3:	Examples of counted instances and excluded instances.	33
Figure 3.4:	Details of YOLOv5 variant. (Ultralytics, 2020)	33
Figure 3.5:	Steps to acquire TP, FP and FN of human.	37

Figure 3.6:	Steps to acquire false detection of cartoons as humans.	39
Figure 3.7:	Steps for retraining a model.	41
Figure 3.8:	Examples of the training dataset.	42
Figure 3.9:	Examples of labelled instances.	42
Figure 3.10:	Data that has undergone data augmentation.	43
Figure 3.11:	Number of instances for cartoon and human.	43
Figure 3.12:	Steps for YOLOv5/ YOLOv8 retraining.	45
Figure 3.13:	Steps for MobileNet SSD retraining.	46
Figure 3.14:	Process flow of the proposed solution.	48
Figure 3.15:	Process flow of human detection, tracking and customer counting.	50
Figure 3.16:	Object ID assigned to centroids. (Rosebrock, 2018)	54
Figure 3.17:	Latest and existing centroids in the subsequent frame. (Rosebrock, 2018)	55
Figure 3.18:	Object ID assumption. (Rosebrock, 2018)	55
Figure 3.19:	ID assigned to new human. (Rosebrock, 2018)	56
Figure 3.20:	Flowchart of face detection, face recognition and gender classification.	57
Figure 3.21:	128 Measurements of Facial Image. (Geitgey, 2016)	59
Figure 4.1:	Graph of pretrained model vs accuracy.	63
Figure 4.2:	Graph of pretrained model vs precision & recall.	64
Figure 4.3:	Graph of pretrained model vs F1 score.	65
Figure 4.4:	Graph of pretrained model vs false rate.	65
Figure 4.5:	Graph of pretrained model vs params & model size.	66
Figure 4.6:	Graph of retrained model vs accuracy.	68
Figure 4.7:	Graph of retrained model vs precision & recall.	69
Figure 4.8:	Graph of retrained model vs F1 score.	69

Figure 4.9:	Graph of retrained model vs false rate.	70
Figure 4.10:	Graph of retrained model vs params & model size.	71
Figure 4.11:	Graph of retrained model's human class vs accuracy, precision, recall & F1 score.	75
Figure 4.12:	Graph of retrained model's cartoon class vs accuracy, precision, recall & F1 score.	76
Figure 4.13:	Graph of retrained model vs combined accuracy.	77
Figure 4.14:	Examples of the mix dataset.	78
Figure 4.15:	Graph of YOLOv8 series vs accuracy, precision, recall & F1 score.	79
Figure 4.16:	Examples of the scenario of human detection result.	80
Figure 4.17:	Real-life practical set videos.	81
Figure 4.18:	Facial Image in the Customer Database.	81
Figure 4.19:	Part of the customer information in customer database.	82
Figure 4.20:	Demonstration of human detection, tracking and customer counting.	82
Figure 4.21:	Human detection and tracking in different frame.	83
Figure 4.22:	Missing count of human.	83
Figure 4.23:	Updated customer database.	90
Figure 4.24:	Failure of human tracking.	91

LIST OF SYMBOLS / ABBREVIATIONS

H^l	correlation filter
\bar{G}	desired correlation output
F^l	grayscale image
$\overline{F^k}$	complex conjugation
A_t	numerator of correlation filter
B_t	denominator of correlation filter
y_{trans}	translation correlation
W	width of resized frame
E_d	Euclidean distance
x_1	point 1 of x-coordinate
x_2	point 2 of x-coordinate
y_1	point 1 of y-coordinate
y_2	point 2 of y-coordinate
\bar{y}	mean value of y-coordinate centroids
s_x	starting x-coordinate of bounding box
s_y	starting y-coordinate of bounding box
e_x	ending x-coordinate of bounding box
e_y	ending y-coordinate of bounding box
λ	regularization term
η	learning rate parameter
TP	True Positive
FP	False Positive
FN	False Negative
FD	False Detection
IoU	Intersection over Union

CHAPTER 1

INTRODUCTION

1.1 General Introduction

Closed circuit television (CCTV), also known as video surveillance, is commonly used in retail shops for security purposes. It enables the monitoring of customer activity to prevent potential criminal activities. In addition to security, retail shops can utilize CCTV to gain valuable information about their customers. Demographics such as age and gender can be recorded from the footage to understand customer behavior and preferences. This information can help retail shops provide better products and services and increase market sales (Mora, Nalbach, and Werth, 2019). Unfortunately, it is not practical for human workers to continuously monitor the footage and record customer information due to fatigue. To address this issue, machine vision is employed on the CCTV system to perform customer analysis.

Machine vision is a technology that mimics human sight to observe and analyze data with incredible speed and accuracy. It can detect and process minute details quickly, making it a valuable tool in various industries. In CCTV systems, machine vision technology comprises a camera, software, and output (Labudzki, Legutko, and Raos, 2014).

Human detection, as one of the artificial intelligence techniques utilized for customer analysis, involves detecting the location of people in images or videos. It is used for a variety of applications, ranging from developing marketing strategies to personal security and even pedestrian detection in traffic. This technology is highly versatile and useful. In customer analysis with machine vision, human detection typically serves as the first process in a pipeline, leading to higher-level reasoning modules such as people counting, people identification, and gender classification (Davis James and Sharma, 2009).

1.2 Importance of the Study

Customer analytics plays a crucial role in helping retailers gather and analyze customer data, allowing them to make informed decisions for their business

plans. Without customer analysis, retailers would struggle to deliver relevant and personalized offerings to their customers, potentially leading to their eventual elimination from the marketplace. Therefore, it is significant for retailers to perform customer analysis to gain valuable insights into customers' demographics and behavior. By doing so, retailers can build better consumer relationships and create immersive shopping environments (Cutittoi, 2022). According to research by McKinsey (Palmatier and Martin, 2019), retailers that utilize customer analytics outperform their competitors in terms of sales growth and gross margins by more than 25%, demonstrating that the advantages of customer analysis outweigh the risks. More than 85% of organizations claim that extensive use of customer analytics has resulted in significant value contribution, including improved in-store shopping experiences, lower marketing costs, deeper customer understanding, and customer loyalty.

Traditional retail shops have historically struggled to obtain customer information such as interests and demographics compared to online stores. Online stores can easily identify customer interests and recommend relevant items with the help of AI and big data. Thus, it is important for traditional retail shops to keep pace with online stores and improve by quickly analyzing customer needs to remain competitive. With the advancement of technology, machine vision solves the bottleneck of data collection and enables data-based innovations that assist physical retail shops in capturing customer data and optimizing behavior patterns (Bratu and Sabau, 2022). Therefore, retailers can formulate better business plans and enhance the shopping experience for their customers (Mora, Nalbach and Werth, 2019). As a result, retailers are able to strengthen their market position and avoid elimination from the marketplace.

On the other hand, human detection is a crucial component of customer analytic systems. It enables businesses to detect the presence of humans, followed by implementing people counting, identification of individuals, etc. The valuable information such as people flow and customer demographic gained by the system will ultimately improve businesses performance. The goal of human detection is to enable a computer system to accurately locate humans in images or videos while producing as few false positives as possible (Davis James and Sharma, 2009). Despite its importance,

human detection remains a challenging task for computer systems, as the level of accuracy achieved by the human brain is difficult to replicate. To address this challenge, numerous algorithms and techniques have been developed, with recent attention focused on deep learning for computer vision applications (Bam, Choudhary and Bhoir, 2021). The key focus of automated human detection is to minimize false detections, in order to increase accuracy for customer analysis.

1.3 Problem Statement

To remain competitive in the market, retail shops must gain insights into their target customers, analyze this data, and learn from it. Membership card programs and customer surveys are commonly used to extract customer information, but these methods are often ineffective due to their slow speed and the fact that not every customer is willing to participate. Therefore, the easiest and most effective way to extract valuable customer information is through CCTV video footage. CCTV cameras can provide vital information for customer analysis, such as age, gender, shopping period, frequency of visits, customer behavior, and the areas/products that are most popular.

However, manually analyzing the CCTV footage is not feasible due to the huge volume of videos that need to be watched in real-time. Human workers cannot keep an eye on all customers at the same time. One possible solution is to hire more workers to manually analyze the CCTV footage, but this is costly and impractical. Therefore, an automated computer vision solution that can perform customer analysis should be implemented. With advancements in technology, machine vision and artificial intelligence can be used to develop an algorithm that can provide real-time output analysis results. While automated computer vision solutions can save costs by replacing human workers, they may still face practical challenges such as occlusion by masks, variation in pose, and illumination.

Human detection is a useful technique that serves as the first step in identifying all humans in video footage before passing data to other algorithms. However, the human detection model may encounter cases where cartoons or human-like stickers are mistakenly identified as humans due to similarities in structure. Possible scenarios where cartoon stickers appear include anime or

cartoon lovers decorating their retail shops with them, bunting banners promoting anime movies in front of movie theaters, and so on. False detections of cartoons can lead to imperfections in the human detection model, resulting in inaccurate customer analysis such as incorrect customer counting and facial recognition of cartoons. Furthermore, false detections also consume unnecessary algorithm resources. Therefore, a review to address this issue will be undertaken.

1.4 Aim and Objectives

The aim of the project is to develop a computer vision solution that can extract valuable information such as customers' demographic through the CCTV videos. The project objectives are:

1. To address the issue of false detecting cartoons as humans in human detection model.
2. To perform human detection and tracking them to count the flow of customer within the retail shop.
3. To perform customer analysis for recognizing the customers' faces and classify their gender.
4. To save the customers' information in a database.

1.5 Scope and Limitation of the Study

To achieve real-time customer analysis through CCTV videos, it is essential to minimize the computational cost of the algorithm to ensure the smooth processing of the video, allowing it to be synchronized with real-time events. To maintain consistency, only the CPU will be used for running the algorithm, while the GPU will be utilized for training the model. However, the project is facing challenges in implementing the algorithm in practical situations due to various factors, including detecting cartoons as humans, people wearing masks, high customer traffic in the store, and varying illumination. As a result, the project will prioritize addressing issues such as improving human detection accuracy, reduce the false detections of cartoon as humans and improve processing video speed within the given time constraints and limitations.

Window 11, AMD Ryzen 7 5700U with Radeon Graphics @ 1.80 GHz, 8.0 GB RAM, x64-based processor, Python programming language,

Python Integrated Development Environment such as Pycharm, Python modules, pretrained model, retrained model and videos will be used to proposed the solution.

CHAPTER 2

LITERATURE REVIEW

2.1 Human Detection

Human detection is a computer vision technique that involves using algorithms to identify and locate all human beings present in an image. Figure 2.1 provides an overview of the general process of human detection. Initially, the camera captures a video sequence that serves as the input for the algorithm. Next, object detection occurs, wherein the algorithm locates and identifies objects within the video sequence. Subsequently, the algorithm classifies the detected objects as either human or non-human (Ansari and Singh, 2021).

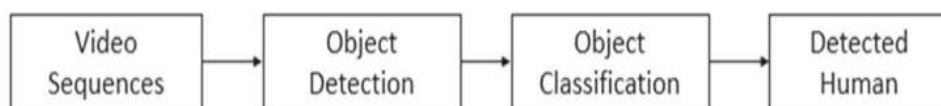


Figure 2.1: General framework of human detection. (Ansari and Singh, 2021)

According to Ansari and Singh (2021), there are four main types of human detection techniques based on different models. The first type is the face feature based module, which detects humans by identifying their faces in the image. The second type is the motion feature based module, which tracks moving pixels in successive frames to detect humans. The third type is the body appearance based algorithm, which classifies objects as human by analyzing their body shape, curves, and components. Finally, the fourth type is the deep learning based technique, which uses the power of graphic processing units (GPUs) to perform complex computational calculations and extract important features of objects to determine whether they are humans or not.

2.1.1 Face Feature Based

The face feature based technique can be subdivided into three main methods: skin color, face geometry, and face texture. Each of these methods uses distinct approaches to detect the human face.

2.1.1.1 Skin Color

The skin color based technique is used to detect a human face by analyzing the skin pixels within an image. This algorithm maps every pixel in a color image to its corresponding color value and then assigns each pixel an intensity value of either '1' to indicate that it is a skin pixel or '0' to indicate that it is not a skin pixel, using a threshold value. The color space used plays an important role in determining the threshold value. Al-mohair, Mohamad-saleh, and Suandi (2013) found that the YIQ color space performs better than other color spaces when differentiating between skin and non-skin pixels. Ansari and Singh (2021) suggest that using multiple color spaces can yield better results than using a single color space for skin detection. Maheswari and Korah (2017) also noted that the Cb and Cr components in the YCbCr color space are effective in detecting skin pixels, regardless of skin color variation. However, skin color based methods still face challenges in dealing with issues such as illumination, occlusion, and reflection.

2.1.1.2 Face Geometry

The face geometry based module uses the Head Curve Geometry technique to extract geometric face parameters, including curves, points, and lines, from human faces. The algorithm performs point searching to identify various points within the image, such as the head point, right neck point, and mean point. It then draws curves to connect these points based on the previously searched points. According to Wong, et al. (2011), this technique achieves an accuracy score of up to 91.4%. However, this method only works well when detecting a single human face, and the face geometry-based algorithm still struggles when multiple humans are in the same frame.

2.1.1.3 Face Texture

The face texture based module detects human faces by utilizing spatial and textural information. Two techniques used in this module are Viola Jones and Local Binary Pattern Histogram (LBPH). The Viola Jones technique extracts textural information from the face using haar feature selection, followed by Adaboost training and Cascade classifier to improve detection accuracy. On the other hand, LBPH evaluates features and creates a histogram based on

them to build classifiers for human detection. According to Ahmed, et al. (2018), LBPH has a real-time detection speed of 15 fps. However, this method has limitations as it is effective only when the frontal face is clearly visible in the image. Although the face texture based module is faster, it still lacks accuracy.

2.1.2 Motion Feature Based

A motion feature based module utilizes four main techniques: frame differencing, histogram of oriented optical flow (HOOF), space geometry and optical flow, and motion vector. All of these techniques detect humans based on the concept of moving pixels.

2.1.2.1 Frame Differencing

Frame differencing techniques include both background subtraction and frame differencing methods. According to Piccardi (2004), the background subtraction technique is a simple algorithm that searches for moving pixels in a video sequence by subtracting the current frame from the background frame. Next, the pixels are classified as either moving or foreground pixels using a threshold value. Although this technique can detect moving objects, it cannot differentiate between humans and animals. The frame differencing technique works similarly to the background subtraction technique, but instead subtracts the current frame from the previous frame. The limitation of this algorithm is that if a moving object stops, it may be recognized as part of the background.

2.1.2.2 Histogram of Oriented Optical Flow (HOOF)

HOOF is a technique that can efficiently track moving objects (Hsu, Gubbi and Palaniswami, 2013). In addition to extracting motion information, HOOF produces a histogram derived from optical flow. The classifier is then used to determine whether the moving object is human or non-human. Dalal, Triggs, and Schmid (2006) suggest that combining visual HOOF with motion HOOF produces better accuracy. However, generating a histogram of optical flow using HOOF takes time and is therefore not suitable for real-time human detection.

2.1.2.3 Space Geometry

Han and Tong (2013) proposed the space geometry technique for detecting humans, which uses region segmentation and classification based on optical flow and Bandelet transform. However, this algorithm has certain drawbacks as its performance can be negatively impacted by factors such as noise and varying lighting conditions.

2.1.2.4 Optical Flow and Motion Vector

The optical flow and motion vector technique is a popular method used for real-time video analysis to detect moving objects. Han and Tong (2013) explained that this method utilizes both optical flow and motion vector estimation to detect and track objects within a video frame sequence.

2.1.3 Body Appearance Based

The body appearance feature based module can be divided into four main methods, namely histogram of oriented gradient (HOG), census transform histogram (CENTRIST), edgelet based, and component based. These techniques rely on the natural appearance of the human body to extract information and detect the presence of humans.

2.1.3.1 Histogram of Oriented Gradient (HOG)

The HOG algorithm can detect human contours in a video by evaluating features and building a classifier through gradient computation, histogram generation, and normalization, as proposed by Dalal, Triggs, and Schmid (2006). The image is first divided into cells, and the algorithm calculates the gradient magnitude and orientation of edges. Based on these values, a histogram is generated. To account for variations in lighting conditions, normalization is used.

2.1.3.2 Census Transform Histogram (CENTRIST)

CENTRIST is a technique derived from the traditional Census transform, which addresses the limitations of the Census transform, such as incapability to match with high dimension sparse space and irregular histogram distribution (Chiachia, et al., 2011). CENTRIST is capable of providing better information

than the Census transform to detect the presence of humans. Wu, Geyer, and Rehg (2011) suggest that the CENTRIST descriptor is a fast and efficient way to detect humans, as it can extract human contours at 20 fps at a resolution of 640 x 480.

2.1.3.3 Edgelet Based

The Edgelet based method is a robust algorithm that can detect humans without being affected by reflections and shadows (Bhuvaneswari and Rauf, 2009). Wu and Nevatia (2007) suggest that edgelet features are effective in representing various body parts such as hands, legs, head, and body, and can be used with edgelet-based classifiers to improve human detection accuracy. However, this algorithm may not perform well in harsh environments, leading to decreased accuracy.

2.1.3.4 Component Based

Component-based algorithms are specialized in detecting different parts of the human body, such as the head, arms, and legs. They can handle variations in lighting and occlusion and are view-invariant, meaning that they can detect humans from different angles in images (Chakraborty, et al., 2007). However, this algorithm comes with a higher computational cost and is time-consuming, as it requires the detection of each body component separately.

2.1.4 Deep Learning Based

The deep learning approach is a modern method used for human detection. It utilizes convolutional neural network (CNN) based techniques to extract information from images and learn from them. However, to process the deep layers of CNN, powerful computational power is required. With the growth of GPU technology in recent years, it is now possible to support the large amount of computational power needed for this technique and speed up the algorithm. Compared to the traditional methods discussed in section 2.1.1, 2.1.2, and 2.1.3, deep learning based CNN algorithms offer better accuracy for object detection in images and have become more popular. There are two subgroups of deep learning (CNN) based object detectors: double stage and single stage. The double stage detector is a two-stage framework that utilizes region

proposal network or selective search to extract a set of region proposals first before localizing the object’s position and classifying them. The R-CNN series family of algorithms is an example of this approach. On the other hand, a single stage detector only requires a single step that utilizes a single or multilayer neural network to predict the object locations and classify them. The YOLO series family and SSD-based algorithms are popular examples of this approach.

2.1.4.1 R-CNN Series Family: R-CNN, Fast R-CNN, Faster R-CNN

Girshick, et al. (2014) proposed a region-based convolutional neural network (R-CNN) that utilizes Selective Search to extract 2k regions from the colored input image in the first stage. These regions, also known as region proposals, are warped to a square-shaped block before being fed into a convolutional neural network to generate 4,096 feature vectors. These feature vectors are then fed into an SVM classifier to classify the object present within the corresponding region proposal and localize the bounding box. Figure 2.2 provides a visual representation of the general framework of R-CNN. According to the authors, R-CNN was initially proposed to address the limitations of traditional approaches that select a large number of regions for object classification, resulting in time-consuming processing. Although R-CNN is faster than traditional approaches, it still requires a significant amount of time (>10 s) to train the network. Therefore, it is not suitable for real-time video implementation.

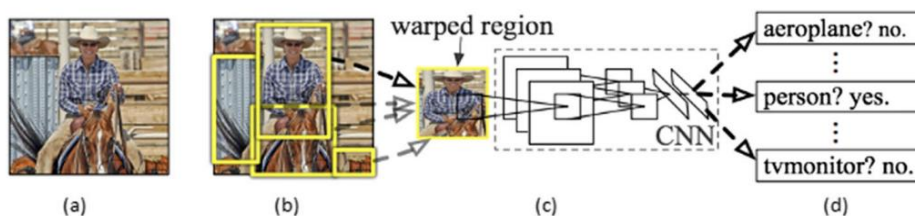


Figure 2.2: General framework of R-CNN. (Ansari and Singh, 2021)

In 2015, Girshick (2015) proposed a new version of R-CNN, called Fast R-CNN, which aimed to address several drawbacks in the original R-CNN framework. Fast R-CNN and R-CNN are similar in many aspects, but in

Fast R-CNN, the input colored image is fed directly to the CNN to produce a convolutional feature map, instead of feeding the region proposal to CNN in R-CNN. The region proposals are then extracted from the convolutional feature map and reshaped into fixed size blocks using the ROI pooling layer. The fixed size block is fed into a fully connected layer, and the proposed region's class and bounding box offset values are predicted from the ROI feature vector using the softmax and regression layers. Figure 2.3 illustrates the overall algorithm of Fast R-CNN. Compared to R-CNN, Fast R-CNN is faster because the convolution operation is performed once per image to generate the feature map. However, the selective search algorithm is still time-consuming in proposing regions, and thus, the performance of Fast R-CNN is still considered slow and unsuitable for real-time implementation.

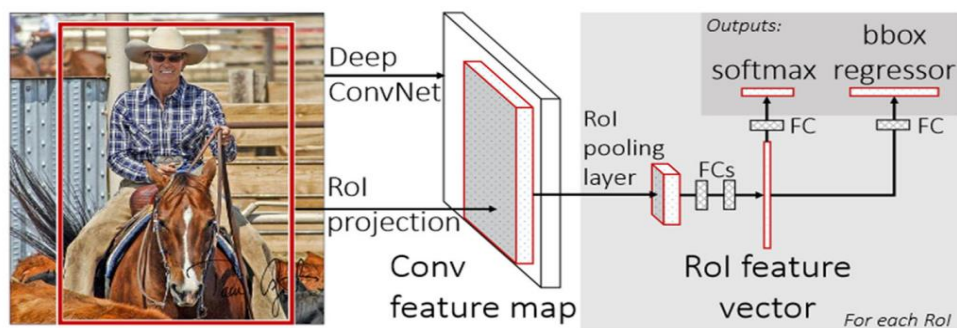


Figure 2.3: Overall framework of Fast R-CNN. (Ansari and Singh, 2021)

Both R-CNN and Fast R-CNN use selective search algorithms to extract region proposals, which is time-consuming. To address this issue, Ren, et al. (2015) proposed an improved framework called Faster R-CNN, which replaces selective search with a Region Proposal Network (RPN). Like Fast R-CNN, the Faster R-CNN inputs the colored image directly to a CNN to produce a convolutional feature map. The RPN is then used to extract region proposals from the feature map. These proposals are reshaped into fixed-size blocks using ROI pooling, and the fixed-size blocks are passed to a fully connected layer. Finally, the softmax and regression layers are used to predict the class of the proposed region and the offset values for the bounding box. Figure 2.4 illustrates the overall workflow of Faster R-CNN. The author reports that Faster R-CNN is 10 times faster than Fast R-CNN and 250 times

faster than R-CNN. The Faster R-CNN can process images at a speed of 5 fps, but it is still not suitable for real-time human detection. Overall, the R-CNN family series is more accurate than traditional approaches, but its speed is still not sufficient for smooth real-time video processing.

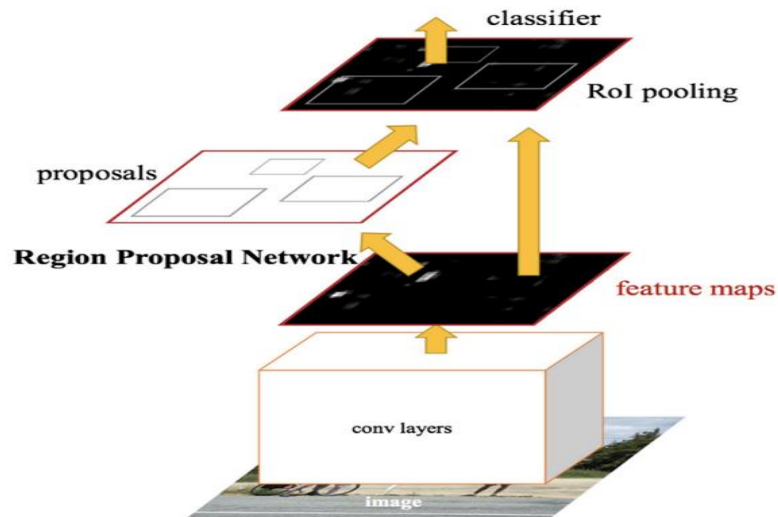


Figure 2.4: Overall workflow of Faster R-CNN. (Ansari and Singh, 2021)

2.1.4.2 YOLO (You Only Look Once) Series Family: YOLOv1, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv8

In 2015, Redmon, et al. (2015) introduced YOLO, a single stage object detector that differs from the two-stage approach of the R-CNN family series. YOLO relies solely on a fully connected layer to perform all predictions. Once the input image is fed into the YOLO algorithm, it is divided into an $S \times S$ grid where each of the N grids serves as a bounding box responsible for detecting and localizing objects within it. Only detections with higher class probability values than the predefined threshold are classified to avoid duplicate predictions from multiple cells containing the same object with different bounding box predictions. Figure 2.5 illustrates the YOLO framework. YOLO v1 achieved 57.9% accuracy on the VOC 2012 dataset with a speed of 45 fps, according to the authors.

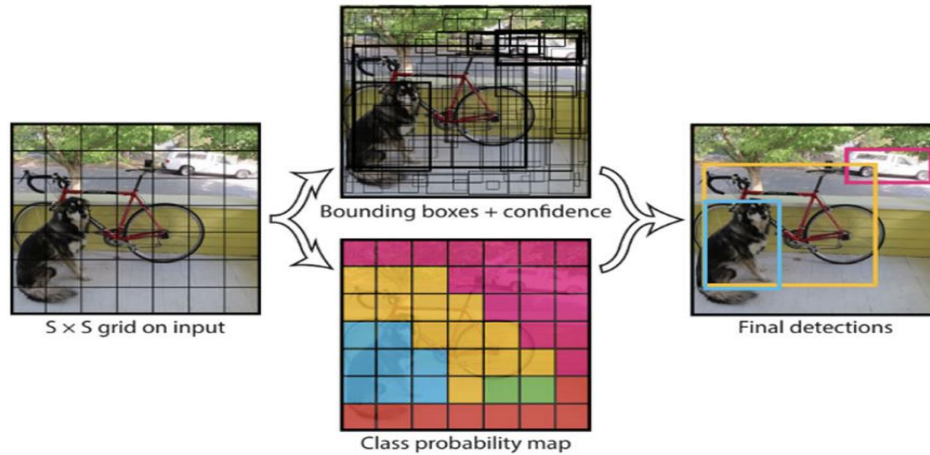


Figure 2.5: Overall framework of YOLO. (Ansari and Singh, 2021)

In 2017, Redmon and Farhadi (2017) introduced an improved version of YOLO, called YOLO v2 or YOLO 9000. The algorithm employed Darknet-19 as its model backbone and utilized anchor boxes to easily locate objects. To increase speed, extra features such as fine-grained features, a high-resolution classifier, and multiscale training were added. According to the authors, YOLO v2 achieved an accuracy of 73.4 mAP on the VOC 2012 dataset with a speed of 67 fps.

In 2018, Redmon and Farhadi (2018) further enhanced YOLO with YOLO v3, which uses DarkNet-53 as its model backbone to replace DarkNet-19 in YOLO v2. This architecture is more complex and accurate for extracting tiny features. The algorithm employs multiscale prediction to generate finely-tuned information from feature maps. Logistic regression is then used to assign a score and binary cross-entropy loss is used to classify the object. YOLO v3 is better at detecting smaller objects than YOLO v1 and YOLO v2. According to the authors, YOLO v3 achieved a 28.2 mAP at 45 fps when tested on the COCO dataset.

In 2020, Bochkovskiy, et al. (2020) presented YOLOv4, which is not merely a new algorithm but rather a comprehensive investigation into ways of enhancing YOLO-style object detectors. The study includes extensive experiments involving hyperparameter tuning, resulting in various best practices for selecting the CNN architecture, activation functions, loss functions, data augmentation techniques, regularization methods, and normalization approaches. Additionally, the authors offer a comparison of

hyperparameters related to the training algorithm. As a result, the YOLOv4 implementation achieved some crucial yet gradual enhancements in training time, accuracy, and inference time.

In the same year, Ultralytics LLC introduced the YOLOv5 model as a novel approach to object detection (Ultralytics, 2020). YOLOv5 utilizes PyTorch and incorporates CSPDarknet53 as its backbone. The use of CSPDarknet53 enhances accuracy by integrating gradient changes into the feature map, thus resolving the redundancy of gradient information found in wide backbones. As a result, the model's inference speed is improved, and the number of parameters is reduced, leading to a smaller model size. YOLOv5 also implements PANet, a path aggregation network that employs a feature pyramid network (FPN) with multiple bottom-up and top-down layers to enhance information flow. This improves the propagation of low-level features in the model and enhances the localization accuracy of detected objects, particularly at lower levels. In addition, the head of YOLOv5 has the same structure as that of YOLOv3, producing three separate feature maps to enable multi-scale prediction. The image is initially fed into the CSPDarknet53 network, which extracts image features, and then into PANet for feature fusion. Finally, the YOLO layer generates the outputs, as shown in Figure 2.6.

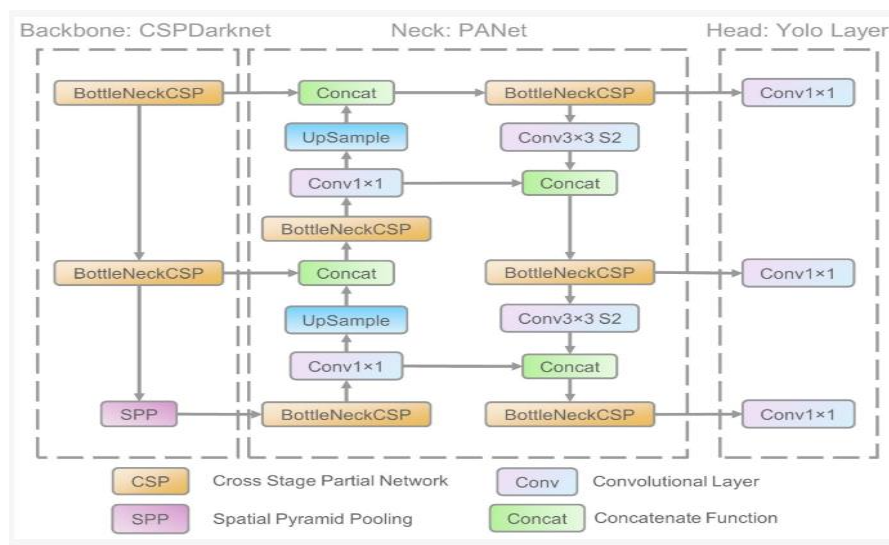


Figure 2.6: Network architecture of YOLOv5. (Xu, et al., 2021)

Ultralytics, the company behind the development of YOLOv5, recently released YOLOv8 in January 2023 (Ultralytics, 2023). YOLOv8 is the newest version of the YOLO series of detection models, which are recognized for their ability to perform joint detection and segmentation. Like YOLOv5, YOLOv8 features a backbone, head, and neck in its architecture. YOLOv8 boasts an upgraded architecture, improved convolutional layers in the backbone, and a more advanced detection head, making it an ideal choice for real-time object detection. It also supports the latest computer vision algorithms, such as instance segmentation, enabling the detection of multiple objects in images or videos. The model utilizes the faster and more accurate Darknet-53 backbone network, which improves its precision and speed compared to the previous YOLO network. YOLOv8 employs an anchor-free detection head to predict bounding boxes. The model is more effective than its predecessors because of its larger feature map and enhanced convolutional network. Additionally, YOLOv8 incorporates feature pyramid networks to recognize objects of varying sizes. The model also has a user-friendly API, simplifying its implementation in various applications (Aboah, et al., 2023).

2.1.4.3 SSD (Single Shot Detector)

Prior to 2015, Faster R-CNN was the algorithm that provided high accuracy for object detection, but it was not fast enough for real-time processing. In 2015, Liu, et al. (2015) proposed a new algorithm called SSD, which eliminated the use of region proposal networks to speed up the object detection process. Instead, SSD utilized default boxes and multi-scale features to recover the drop in accuracy caused by the elimination of RPN. These improvements allowed SSD to process the algorithm with lower resolution images, further enhancing its speed. The architecture of SSD consists of a base network, extra feature layers, and prediction layers, as illustrated in Figure 2.7. The base network of SSD uses VGG-16 as its model backbone, and extra feature layers extract useful information effectively at multiscale and shrink the input size. Lastly, the prediction layer predicts the classification scores for bounding boxes.

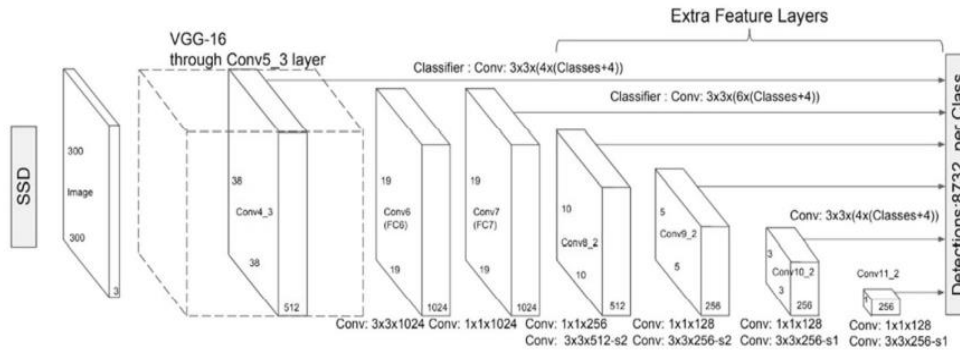


Figure 2.7: Architecture of SSD. (Ansari and Singh, 2021)

Although the SSD algorithm is designed to balance the speed of YOLO and the accuracy of R-CNN, it may perform slowly on devices that lack a GPU. This makes it unsuitable for real-time applications that require fast response to constantly changing environments.

2.1.4.4 SSDLITE

In 2018, Sandler, et al. (2018) proposed a method which is SSDLite. SSDLite is a variant of SSD that uses separable convolutions, consisting of Depthwise and 1×1 projection, to replace all of the regular convolutions in the detection layer. This modification reduces the number of parameters and computational cost of the model, making it more suitable for deployment on embedded devices compared to the conventional SSD model.

2.1.4.5 MobileNet SSD

Howard, et al. (2017) proposed MobileNet SSD, a lightweight CNN network that uses depthwise convolution instead of standard convolution to significantly reduce the required number of parameters in SSD. According to Gao, Zhai, and Guo (2021), MobileNet SSD achieves accuracy comparable to that of SSD, with only a 0.9% decrease when tested on the ImageNet dataset. However, the computational cost of MobileNet SSD is reduced by 27 times and the number of parameters is reduced by 33 times, making it suitable for real-time video processing without the need for GPU acceleration.

2.1.4.6 RetinaNet

Lin, et al. (2018) introduced the RetinaNet network in 2018 in their paper "Focal Loss for Dense Object Detection". This single-stage network is based on the ConvNext backbone network and generates a feature pyramid using the feature pyramid network (FPN) backbone, as illustrated in Figure 2.8. By using FPN, the network can create a rich and multi-scale convolutional feature pyramid, simplifying the network design and allowing for greater focus on the newly proposed focal loss. This focal loss reduces the accuracy gap between single-stage and two-stage networks, while maintaining high speed (Lechen, 2022).

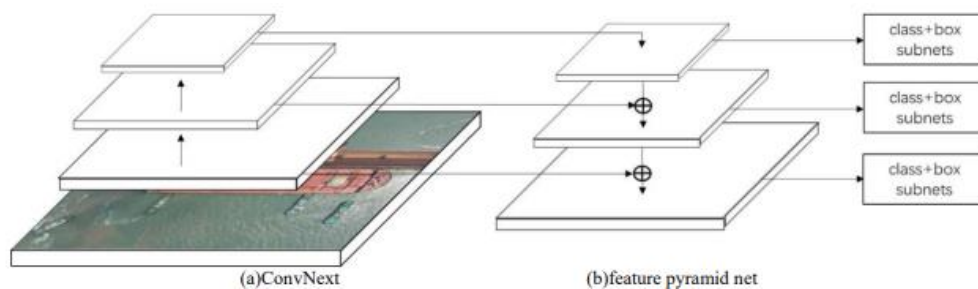


Figure 2.8: Convolutional feature pyramid based on backbone network ConvNext. (Lechen, 2022)

2.1.4.7 Mask R-CNN

Mask R-CNN is an extension of the Faster R-CNN approach that adds a function for predicting instance masks. While Faster R-CNN focuses on bounding box detection, Mask R-CNN generates more detailed spatial information about detected objects by predicting masks at the pixel level. This is achieved through the addition of a fully connected convolutional network. When a region of interest (RoI) is passed through the post-processing branch, Mask R-CNN produces both bounding boxes and masks for each instance, as opposed to compressing the feature information into vectors as in the case of bounding box features (He, et al., 2020).

2.1.4.8 FCOS

Recent research has introduced a novel approach to object detection, FCOS, which adopts a per-pixel prediction strategy similar to semantic segmentation.

Unlike traditional methods, FCOS eliminates the need for anchor boxes and proposals. By removing anchor boxes, FCOS avoids complex calculations associated with overlapping boxes during training and eliminates anchor box hyper-parameters, which can negatively impact detection performance. FCOS simplifies the detection process, requiring only non-maximum suppression (NMS) as post-processing. However, the current version of FCOS is limited by its large, complex network structure, which hinders both efficiency and accuracy. Improvements are needed to address these challenges (Tian, et al., 2019).

2.2 Human Tracking

Human detection is a useful technique to identify humans in a video, but it can be computationally expensive to perform in every frame. To address this issue, human detection is typically performed at regular intervals, with human tracking used to identify and track humans in subsequent frames. This technique helps to save computational resources and increase the speed of video processing. While many robust human tracking methods have been proposed, including MILTrack, GBDL, FragTrack and IVT, these techniques often involve complex algorithms that limit processing speed to 25 to 30 frames per second, making them unsuitable for real-time applications.

To overcome these limitations, Bolme, et al. (2010) proposed the Minimum Output Sum of Square Error (MOSSE) algorithm, which uses correlation filters to track objects in a video. MOSSE initializes the tracked target with a small tracking window and then uses correlation filters to track the target over the search window in the following frame. The target's new position is indicated by the maximum value in the correlation output. This method is robust enough to handle variation in pose, illumination, and non-rigid transformation. Moreover, MOSSE uses the peak-to-sidelobe ratio (PSR) to measure the correlation peak, which helps to track occluded or missing humans when they reappear in the video frame with similar appearances. According to Bolme, et al., MOSSE has been tested on various datasets, such as car4, car11, davidin 300, dudek, fish, sylv, and trellis70, and has shown excellent performance in locating even the most challenging parts of the video. The MOSSE tracker is 20 times faster than traditional methods and can

process video at a speed of 669 frames per second. A part of the results is illustrated in Figure 2.9, demonstrating how the MOSSE tracker can track humans even in challenging situations.

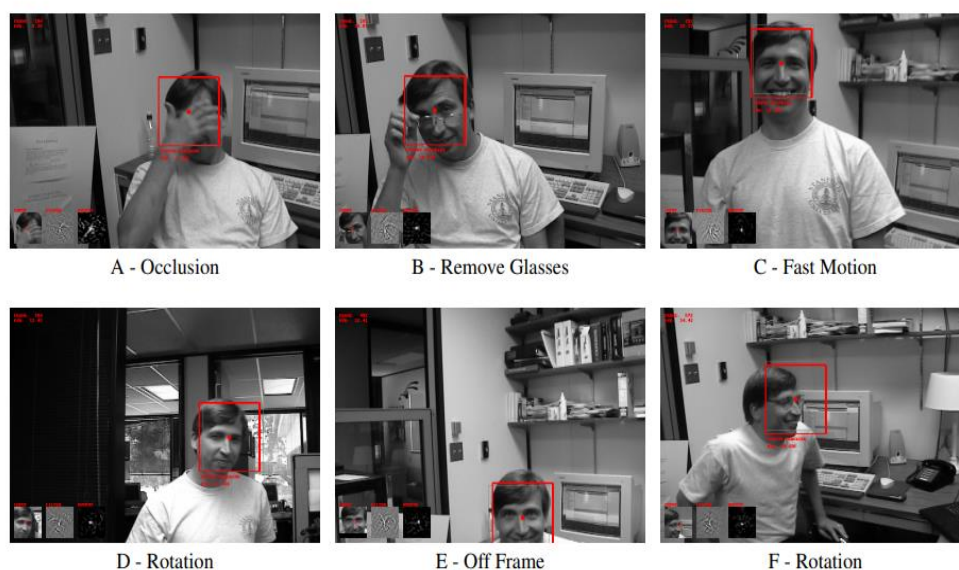


Figure 2.9: Demonstration of MOSSE tracker. (Bolme, et al., 2010)

In 2012, Henriques, et al. (2012) proposed the CSK tracker, which has a similar baseline to the MOSSE algorithm-based tracker. The CSK tracker works by training a classifier using the kernelized least-squares of the target appearance. Like the MOSSE tracker, the CSK tracker can handle variations in illumination, pose, and occlusion, among others. However, both trackers have limitations when it comes to handling frames that involve significant scale variations. In 2014, Danelljan, et al. (2014) proposed an accurate scale estimation algorithm for human tracking. The algorithm is based on the MOSSE tracker's discriminative correlation filters and trains the classifier on a scale pyramid to estimate the target scale. Once the ideal translation has been found, the method can estimate the target scale independently. As the scale estimation algorithm is independent, it can be incorporated into other tracking frameworks that lack scale variation techniques. According to the author, the method can track the object at a frame rate 25 times faster.

2.3 Facial Recognition

A face recognition system is composed of three main steps: face detection, feature extraction, and face recognition. In the first step, the face detection algorithm determines if a face is present in the image and its location is segmented out. Then, the segmented face goes through feature extraction to extract unique characteristics using a feature vector. Finally, verification or identification is used to match the extracted features with a database to recognize the face (Jayaraman, et al., 2020).

Despite advancements in face recognition technology, it remains a challenging task as no system can recognize all faces perfectly. This is due to various factors that affect and degrade the accuracy of the algorithm, which can be classified as intrinsic and extrinsic factors. Intrinsic factors are those that depend on the physical state of the face, such as facial expression and age. Facial expressions are so unique that they can change the geometry of the face, making it difficult for the face recognition system to identify the same person. As for age, facial appearance can vary with aging. On the other hand, extrinsic factors are external factors that change the appearance of the face when the image is captured, such as illumination, noise, occlusion, pose, and resolution. Variations in illumination can affect the accuracy of the face recognition system as it may extract features differently under different lighting conditions. Figure 2.10 shows an example of a person captured under different lighting conditions. Noise in digital images, although sometimes minimal, can lead to poor detection. Occlusion occurs when the face is covered by something like a mask or sunglasses, which can affect the system's ability to work properly. Pose variation can make it difficult for the camera to capture certain parts of the face, leading to faulty results. Lastly, low-resolution images are difficult to compare with high-resolution images in the database (Anwarul and Dahiya, 2020).



Figure 2.10: Face image under lightning variation. (Adjabi, et al., 2020)

Face detection is a crucial step in locating and identifying faces within an image. In 2000, Viola and Jones (2004) introduced a method for face detection, which was the first to achieve satisfactory results at that time. The Viola Jones face detector is based on three main concepts, namely the integral image, classifier learning using AdaBoost, and an attentional cascade structure. These ideas enable the algorithm to run in real-time by quickly computing Haar-like features using the integral image, finding accurate hypotheses using AdaBoost, and rejecting negative sub-windows with the attentional cascade. However, Viola Jones has some limitations, as it is not suitable for detecting frontal and wild faces in a 24x24 image. To address these issues, several methods such as Histogram of Gradient (HOG), Scale Invariant and Feature Transform (SIFT), and Aggregated Channel Features (ACF) have been proposed. In 2009, King (2009) introduced Dlib, a popular method that utilizes a support vector machine (SVM) as a classifier to enhance the robustness of face detection. In 2016, Liao, Jain, and Li (2016) proposed Normalised Pixel Differencing (NPH) to distinguish between pixel intensities, which can be used in unconstrained face detectors.

There have been numerous feature extraction and face recognition techniques proposed over the years. In 1964, Bledsoe (1964), an American researcher, introduced a semi-automated method for face recognition that required operators to manually enter twenty computer measures, such as the size of the eyes and mouth. In 1977, the method was improved by adding twenty-one additional markers, such as hair color and lip width. In 1991, Turk and Pentland (1991) proposed the first successful facial recognition technology called Principal Component Analysis (PCA). When a face is detected and cropped, the resulting 2D-face image matrix is converted into a 1D vector and fed into a data matrix to compute the covariance matrix. The face is then represented as eigenpictures, which constitute the best coordinate system of the face. The Karhunen-Love transform is then applied to the cropped face images to represent them as a vector and compute the mean face vector for each face. Finally, the eigenvectors, which are the primary components that make up the facial image, are calculated. The PCA has been tested on a dataset and achieved an accuracy range of 64% while

accommodating varying illumination, size, and orientation (Turk and Pentland, 1991).

In 1997, Belhumeur, Hespanha, and Kriegman (1997) introduced a 3D linear subspace method to address the issue of lighting conditions and changes in facial expressions. The method employs a projection technique to generate isolated classes in a low-dimensional subspace under different illumination and facial expression variations. In comparison to PCA, the method produced a lower error rate when tested on Harvard and Yale Face databases. Another method proposed in the same year was by Wiskott, et al., (1997) which employed elastic bunch graph matching based on Dynamic Link Structures. This method represents the human face using fiducial points and creates nodes for these points. Corresponding nodes are then joined through an edge to generate a face bunch graph as shown in Figure 2.11. The method obtained an accuracy of 89% when tested on the FERET database for manual positioning. In 1999, Li and Yang (1999) introduced a method to overcome the variability in light. The method used three images to calculate the covariance matrix obtained under various illumination conditions.

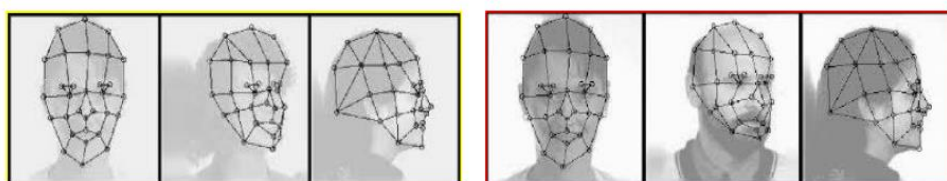


Figure 2.11: Face bunch graph. (Adjabi, et al., 2020)

In 2002, Bartlett, Movellan, and Sejnowski (2002) introduced independent component analysis (ICA) as a method for pattern recognition that utilizes high-order statistics derived from the high-order relationship among pixels. ICA has two different architectures. The first architecture treats the image as a set of random variables, and the pixels are the results. The second architecture treats the pixels as random variables, and the images are the results. The first architecture produces spatially local basis pictures for the faces, while the second architecture defines a fractional facial code. ICA

outperforms PCA, achieving an accuracy of 86% when tested on FERET databases.

In 2002, Ojala, Pietikäinen, and Mäenpää (2002) introduced local binary patterns (LBP), a method that can represent facial images using local texture descriptors based on the micro-patterns of the face. LBP converts each image pixel into a corresponding grayscale value and assigns a binary value to each pixel by applying a threshold on its 3x3 neighborhood. Two years later, Ahonen, Hadid, and Pietikäinen (2004) improved LBP by combining its features using a histogram as a texture descriptor. They also introduced weights to certain facial features such as the eyes, which were deemed more important for face recognition. The weights were assigned based on the significance of the information in each region. Figure 2.12 illustrates the overview of the LBP method. LBP achieved an accuracy of 95% when tested on the FERET database.

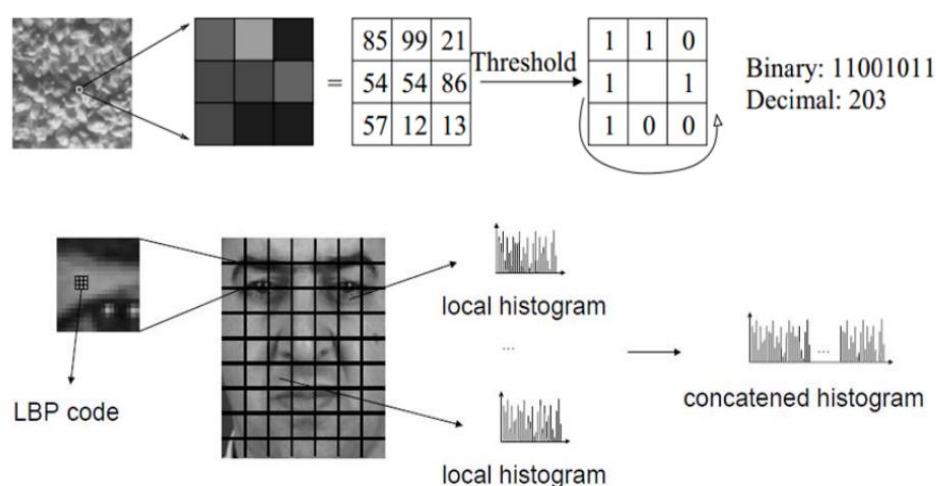


Figure 2.12: Example calculation of LBP. (Adjabi, et al., 2020)

In 2002, Kim, Jung and Kim (2002) proposed kernel PCA (KPCA), which utilizes non-linear mapping to convert the input image into a feature space and applies PCA to it. This method generates a higher correlation between input pixels, thereby enhancing the analysis of facial images. The support vector machine (SVM) is then used to recognize the face based on the obtained features. KPCA has been proven to have better accuracy than PCA, with a recorded accuracy of 83.3% when tested on the Olivetti Research

Laboratory (ORL) database. In 2004, Yang, et al. (2004) proposed 2D-image principal component analysis (2D-IPCA), which differs from PCA in that the 2D-face image matrix is not transformed into a 1D-vector. Instead, the original image matrices are directly used to build the covariance matrix. Once the feature vector for an image has been created, a nearest neighbor classifier is used to classify the image. According to experiments, 2D-IPCA achieved an accuracy of 84% on the ORL, Yale, and AR datasets, which is better than both PCA and KPCA.

In 2005, Dalal and Triggs (2005) proposed the Histogram of Oriented Gradients (HOG) as a face recognition method that is invariant to scaling and 2D rotation. This method segments the image into small cells and computes the histogram of edge orientations for each cell. The final HOG descriptor is represented by normalized histogram counts. HOG achieves an accuracy of around 95.5% when tested on the FERET database. In 2010, Wang, Jiang, and Li (2010) proposed linear techniques to extract features for face recognition, specifically the Discrete Cosine Transform (DCT) and the Discrete Wavelet Transform (DWT), which are typically used for feature selection and image compression. The facial image is first decomposed using 2D-DWT, followed by 2D-DCT to calculate the low-frequency image. Finally, the DCT coefficients are used for matching. The authors claimed that DCT and DWT showed better results compared to PCA when tested on ORL databases.

In 2013, Benzaoui and Boukrouche (2013) proposed a one-dimensional local binary pattern (1DLBP) for face recognition, which involves five main steps illustrated in Figure 2.13. Firstly, the input image was partitioned into multiple blocks of equal size. Next, each block was projected onto a one-dimensional space vertically. The projected block was then processed by the 1DLBP descriptor. Furthermore, a global vector was generated by combining the vectors produced by each block. Lastly, the global vector was reorganized using PCA to reduce its dimensionality while retaining relevant information. The similarity between faces was calculated using the Chi-square distance. When tested on an AR database, the method achieved an accuracy of 96.9%.

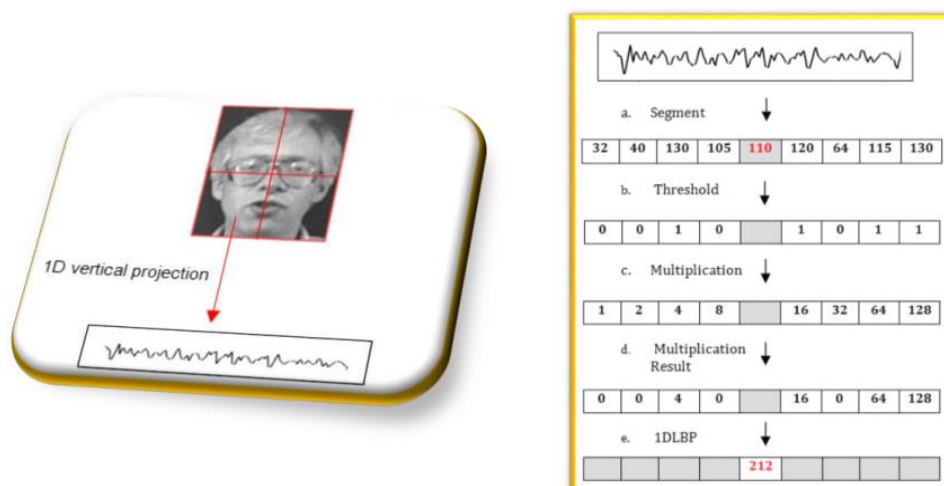


Figure 2.13: Steps of 1DLBP. (Adjabi, et al., 2020)

In 2014, Taigman, et al. (2014) proposed DeepFace, an initial deep learning architecture that utilizes a 3D shape model for face alignment. The method involves a 9-layered convolutional neural network that derives facial representation. The last two layers of the architecture are fully connected layers, whose output passes to the softmax function for classifying K number of classes. According to the experiment, DeepFace achieves an accuracy of 97.35% when tested on the LFW database. In 2015, Schroff, Kalenichenko, and Philbin (2015) suggested FaceNet, a model from Google that uses 128-dimensional representations to represent face features. FaceNet applies deep CNN to train 200 million facial images using the triplet loss function in the last layer. The triplet comprises two facial images of the same person and one facial image of another person. The algorithm then adjusts the weights of the neural network such that the loss for the same face is minimal and higher from another different face. Therefore, the 128-dimensional representations for the same face are closest.

In 2015, Abhishree, et al. (2015) proposed the use of Gabor filters (GFs) to enhance the functionality of face recognition systems by extracting features. The GFs utilize the orientation, frequency, and size of images through the Gaussian window to capture the facial characteristics. Aligned facial features at specific angles are then processed using GFs. A feature selection optimization technique is employed to identify the best feature space. The method has demonstrated good performance in dealing with challenges

such as pose variations, illumination changes, and expression variations. Experimental results showed its effectiveness when tested on the ORL and FERET databases.

In 2018, Wu, et al. (2018) proposed a light CNN framework that utilizes large amounts of data with noisy labels to learn face representation. The framework introduces the Max- Feature Map (MPM) as a maxout activation to replace ReLU, which is used to separate the noisy labels from informative labels. To reduce computational costs, three networks are introduced to decrease parameter numbers. The method achieves an accuracy of 97.50% when tested on the LFW database. In 2020, Ling, et al. (2020) suggested an attention-based neural network (ACNN) for embedding discriminative facial features. The method aims to study global feature relationships of aligned facial images and eliminate unnecessary extra information. The attention module comprises two building blocks, namely, the spatial attention block and channel attention block.

2.4 Gender Classification

A gender classification system is utilized to identify the gender of a face as male or female. Determining the gender at the initial stage can help reduce the search time for recognized faces in the database by half, assuming an equal number of males and females in the database (Rai and Khanna, 2012). Over the past few decades, various studies have been conducted on this topic, and the approaches for gender classification can be categorized into two types: geometric-based and appearance-based methods.

The geometric-based approach for gender classification is based on marking significant points on the face, known as facial landmarks. This approach utilizes only the geometric relationship between these points to perform gender classification (Ng, Tay, and Goi, 2015). In 1993, Brunelli and Poggio (1993) proposed a method that trains a hyper-function network classifier using 18 point-to-point distances. In 1997, Fellous (1997) proposed a semi-automated method that required human operators to manually select 40 points from the facial image to calculate 22 horizontal and vertical fiducial distances. Gender can be classified by deriving the five dimensions using discriminant analysis. In 2008, Xu, Lu, and Shi (2008) proposed the Active

Appearance Model (AAM), which generates 3403 geometry features from 83 landmarks on the face. However, only the ten most important features are chosen for gender classification after normalization. The 83 landmarks applied to the face image are shown in Figure 2.14.

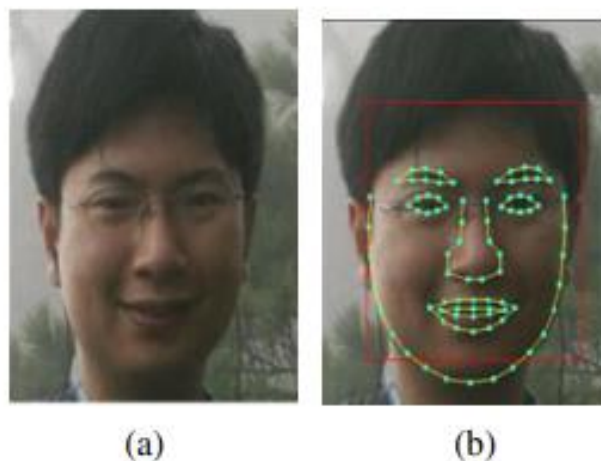


Figure 2.14: Example of landmarks applied. (Rai and Khanna, 2012)

The appearance-based approach treats an image as a high dimensional vector and extracts useful features from its statistical data, without considering the subject matter of the image. This method is generally quicker than the geometric-based approach (Rai and Khanna, 2012). In 1991, Golomb, Lawrence, and Sejnowski (1991) proposed a method that uses a multi-layer neural network to train face images and classify gender. In 1995, Wiskott, et al. (1995) represented face images as graphs with topographical labels and regional templates, using a similar function to perform comparisons to recognize faces and classify gender. In 2002, Moghaddam and Yang (2002) used SVM directly on image intensities for gender classification. The AdaBoost classifier was also used to replace SVM for the same purpose.

In 2007, Lian and Lu (2007) proposed the use of Local Binary Pattern (LBP) to extract facial features and Support Vector Machines (SVM) for gender classification. This approach achieved an accuracy of 94.81% as reported by Shan in 2012. In 2008, Cao, et al. (2008) introduced a part-based gender recognition (PBGR) method that used full body images, including front or back view images, to classify gender. This was the first method to utilize

static human body images for gender recognition, achieving an accuracy rate of 75%. Yuan, Pang and Li (2010) proposed a method that used footwear appearance for gender classification in 2010. The approach employed Histogram of Oriented Gradients (HOG) to represent the footwear image and non-linear SVM for gender classification. The method achieved an accuracy rate of 85.49%.

In 2010, Jabid, Kabir, and Chae (2010) proposed a method called Local Directional Pattern (PCD), which acts as a texture descriptor to represent facial images and classify gender. This approach divides the face area into small regions and applies LDP to extract features into histograms, which are combined into a single vector for efficient representation of the face image. In the same year, Scale Invariant Feature Transform (SIFT) was introduced. This technique derives feature descriptors from important image points, as shown in Figure 2.15. SIFT has the advantage of being insensitive to illumination, image rotation, and scaling, allowing it to extract features from images captured from different views. Wang, et al. (2010) utilized SIFT descriptors and combined them with AdaBoost to classify gender.



Figure 2.15: SIFT key points. (Ng, Tay and Goi, 2015)

In 2012, Basha and Jahangeer (2012) suggested the use of continuous wavelet transform to extract useful features from images and linear kernel SVM to classify gender. Also in the same year, Won, et al. (2012) proposed a motion-based method that uses machine learning and gestures obtained from Microsoft Kinect to recognize gender. The results of the method showed an

accuracy of 83%, but it requires a significant amount of computational resources compared to other methods because it needs image sequencing to record movements, which takes approximately 10 seconds. In 2015, Linder, Wehner, and Arras (2015) proposed a depth-based tessellation learning strategy that can learn the best selection and scale of a set of simple point cloud characteristics, achieving an accuracy of 90%.

In 2015, Levi and Hassner (2015) proposed a simple convolutional neural network architecture for gender classification. This method is able to work effectively even with a limited amount of training data. The method consists of three convolutional neural networks and two fully connected layers, as shown in Figure 2.16. Prior to entering the neural network, the input image is scaled to 256×256 and cropped into 227×227 . The first convolutional layer comprises 96 filters of $3 \times 7 \times 7$ pixels, a ReLU operator, and a max-pooling of 3×3 . The second and third layers are similar, except that the second layer consists of 256 filters of $96 \times 7 \times 7$ pixels and the third layer consists of 384 filters of $256 \times 3 \times 3$ pixels. Next, the output goes through two fully connected layers, each with 512 neurons, a ReLU operator, and a dropout layer. Finally, the output is fed to the softmax layer to generate probability scores and classify the gender. According to the authors, the method achieved an accuracy of 86.8% when tested on the Adience benchmark. This accuracy is considered high in real-world applications, as the Adience benchmark is designed to capture extreme variations in illumination conditions, head pose, and other factors.

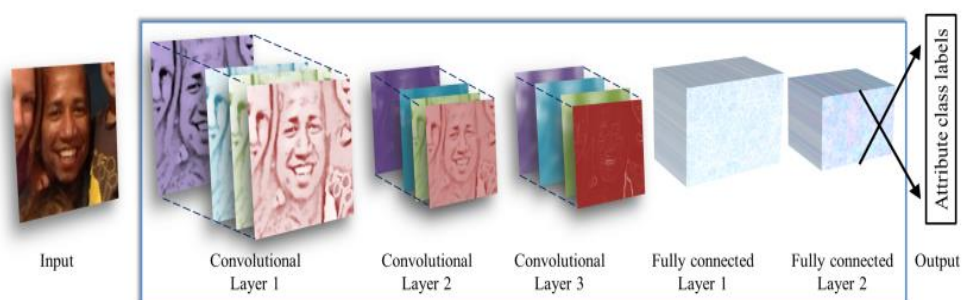


Figure 2.16: CNN architecture. (Levi and Hassner, 2015)

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Improvement of Human Detection

Figure 3.1 illustrates the general steps for improving human detection model to address the issue of identifying cartoons as humans.

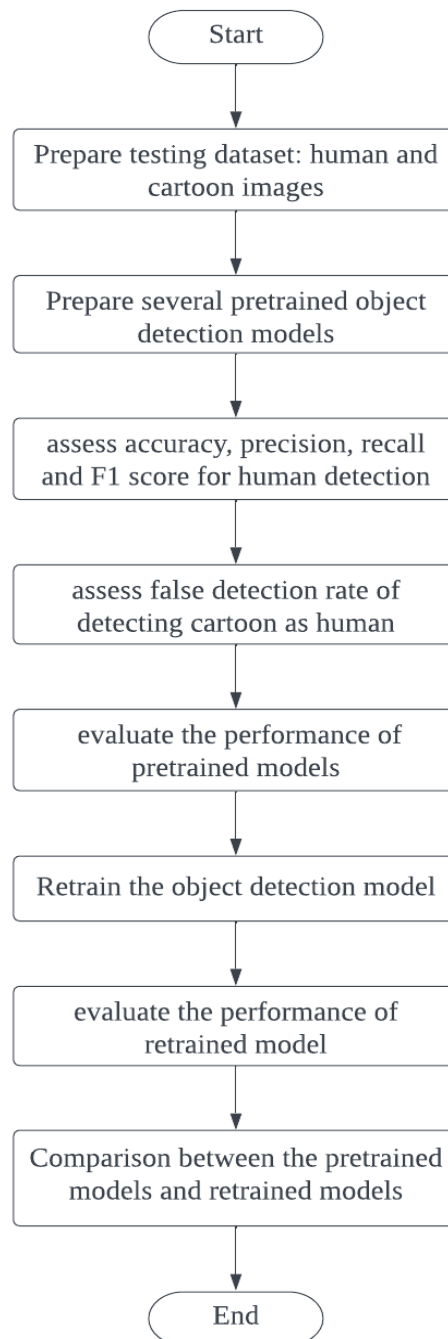


Figure 3.1: General steps for improving human detection model.

3.1.1 Testing Dataset

The testing dataset was composed of 300 human images and 300 cartoon images. Each set of 300 images contained only one type of content, either human or cartoon, with no mixed images. The human images included only real humans and no cartoons, while the cartoon images contained only animated characters and no real humans. The human images were extracted from the Internet and various human datasets, while cartoon images were sourced from the iCartoonFace dataset available on Github (luxiangju, 2021). A portion of both the human and cartoon testing datasets was depicted in Figure 3.2 (a) and (b) respectively.

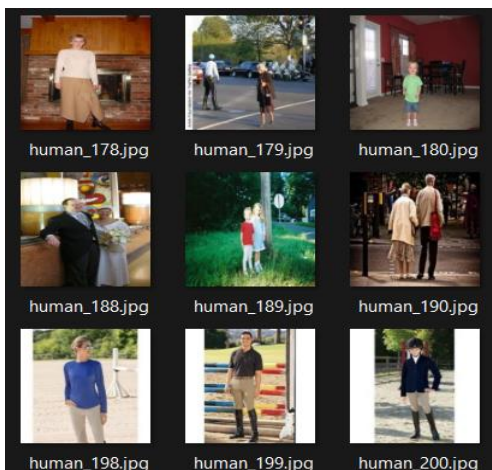


Figure 3.2 (a)



Figure 3.2 (b)

Figure 3.2: Portion of the human and cartoon testing dataset.

The 300 human images contained a total of 427 instances of humans, while the 300 cartoon images contained 515 instances of cartoons. However, instances that appeared very small or where only a portion of the body was visible in the image were excluded from this count (427 instances and 515 instances). The excluded instances would not have affected the result of the detection model. This approach ensured a fair comparison of the detection models, as these excluded instances were considered as difficult cases to detect. For example, in Figure 3.3 (a) and (b), only one human and one cartoon respectively, were counted as instances in the image using a red bounding box. The humans and cartoons that were excluded from the count of instances were shown using a green bounding box, for illustration purposes.

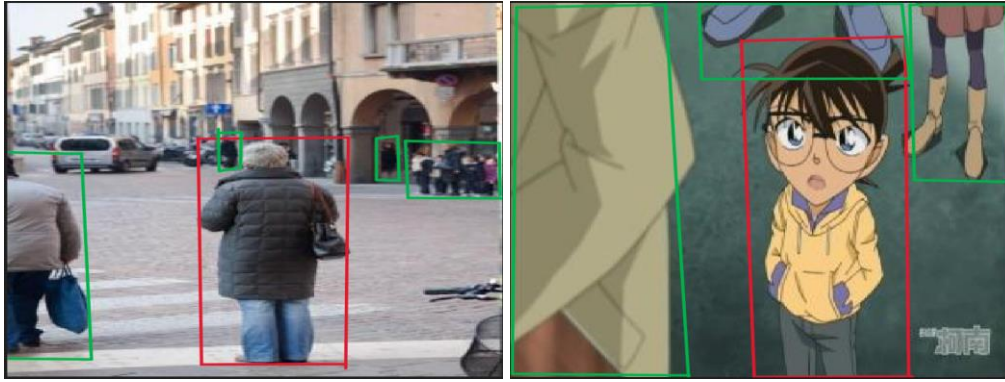


Figure 3.3 (a)

Figure 3.3 (b)

Figure 3.3: Examples of counted instances and excluded instances.

The ratio of the human images to cartoon images was 1: 1.21, which translated to a percentage distribution of 45.30 % and 54.70 % respectively.

3.1.2 Pretrained Object Detection Models

The pretrained object detection models must include human as one of their pretrained classes, so that the model could be utilized in the project to detect humans. The list of pretrained networks utilized to detect humans included models from YOLO family, and models that utilize PyTorch, TensorFlow 1.0, Caffe, OpenCV as backends to run respectively.

The YOLO family models included YOLOv3, YOLOv4, YOLOv5n, YOLOv5s, YOLOv5m, YOLOv8n, YOLOv8s and YOLOv8m. For YOLOv5 and YOLOv8 series, there are five standard variants: nano model (n), small model (s), medium model (m), large model (l) and xlarge model (x). As the model size increases from the n model to the x model, the accuracy improves but the inference speed slows down. Figure 3.4 illustrates the details of the YOLOv5 variant (Ultralytics, 2020).

Model	size (pixels)	mAP ^{val} ₅₀₋₉₅	mAP ^{val} ₅₀	Speed CPU b1 (ms)
YOLOv5n	640	28.0	45.7	45
YOLOv5s	640	37.4	56.8	98
YOLOv5m	640	45.4	64.1	224
YOLOv5l	640	49.0	67.3	430
YOLOv5x	640	50.7	68.9	766

Figure 3.4: Details of YOLOv5 variant. (Ultralytics, 2020)

The n, s, and m models for YOLOv5 and YOLOv8 were included in the testing list, while the l and xl models were excluded due to their long inference time, which was not practical for real-life implementation without a GPU. Other models that were utilized included frcnn-resnet, retinanet, frcnn-mobilenet, ssd_vgg16, ssdlite320_mobilenet_v3, fcos-resnet, maskrcnn-resnet, and keypointrcnn-resnet, which use pytorch as the backend, and frcnn-inception_v2, maskrcnn-inception_v2, ssd-inception_v2, ssdlite_mobilenet_v2, and ssd_mobilenet_v2, which use tensorflow as the backend. Additionally, mobilenet-ssd which uses Caffe, and HOG method which is built-in in the opencv module, were also included. Table 3.1 provides a full list of the pretrained detection models included in the project.

Table 3.1: List of pretrained detection models used for human detection.

No	Pretrained Model
1	YOLOv3
2	YOLOv4
3	YOLOv5n
4	YOLOv5s
5	YOLOv5m
6	YOLOv8n
7	YOLOv8s
8	YOLOv8m
9	Mobilenet ssd
10	HOG
11	frcnn-resnet
12	retinanet
13	frcnn-mobilenet
14	ssd_vgg16
15	ssdlite320_mobilenet_v3
16	fcos-resnet
17	maskrcnn-resnet
18	keypointrcnn-resnet
19	frcnn-inception_v2
20	maskrcnn-inception_v2
21	ssd-inception_v2
22	ssdlite_mobilenet_v2
23	ssd_mobilenet_v2

3.1.3 Evaluation Metrics

This section details the metrics that were used to assess the performance of human detection models. To evaluate the human detection task, various metrics were adopted from the traditional classification problem, including accuracy, precision, recall, F1 score (a harmonic mean of precision and recall), false detection rate and confusion matrix (Gonealves, et al., 2021). In addition, the project also employed metrics such as parameters and model size. Several fundamental concepts for evaluating performance were declared:

True Positive (TP): defines the correct prediction which matches the ground truth.

False Positive (FP): defines the wrong detection which does not matches the ground truth.

False Negative (FN): defines ground truth is not detected, resulting in a negative prediction that should have been positive.

True Negative (TN): defines the prediction is negative and the ground truth is negative.

Accuracy: Measure the correct predictions over all the number of predictions.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} \quad (3.1)$$

Precision: Measure how accurate the predictions are, finds the percentage of correct predictions over all positive predictions, the equation is given by as:

$$\text{Precision} = \frac{TP}{TP+FP} \quad (3.2)$$

Recall: Measure how good the model finding all the ground truth, the equation is given as:

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3.3)$$

F1 score: Measure the harmonic mean between precision and recall, the equation is given as:

$$\text{F1 score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3.4)$$

Intersection over Union (IoU): It is used to measure the accuracy of the bounding box, depends on the similarity between the predicted bounding box and the ground truth bounding boxes. It is also known as Jaccard Index, the equation is given as below:

$$\text{IoU} = \frac{\text{area}(\text{box}(\text{ground truth}) \cap \text{box}(\text{predict}))}{\text{area}(\text{box}(\text{ground truth}) \cup \text{box}(\text{predict}))} \quad (3.5)$$

3.1.4 TP, FP and FN of Human

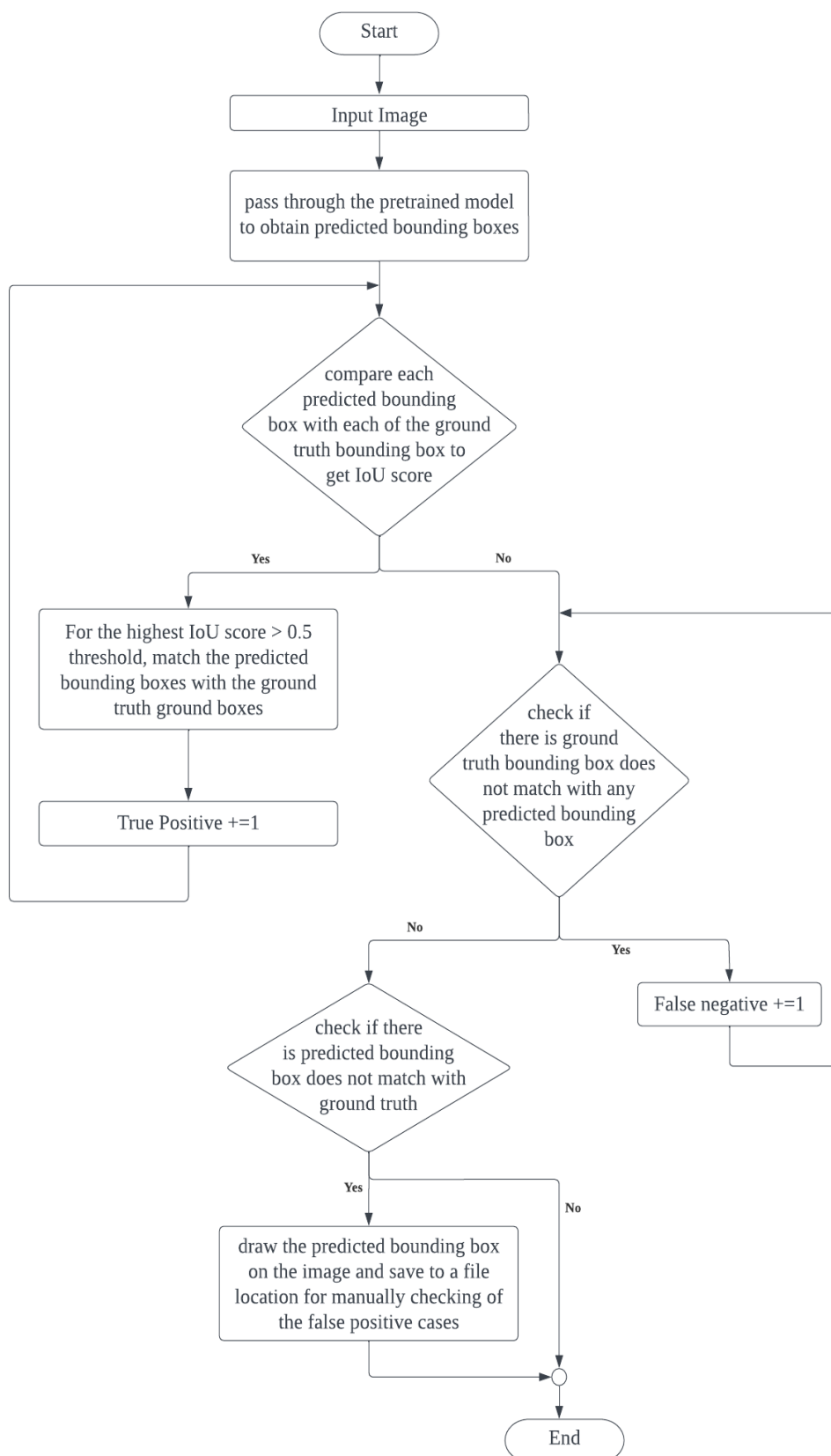


Figure 3.5: Steps to acquire TP, FP and FN of human.

Each pretrained model was tested on 300 human images. To obtain accuracy, precision, recall, and F1 score of the human class, it was necessary to determine the total number of true positive, false negative, and false positive cases. Firstly, the input image was loaded into the pretrained model to obtain the detection results. To ensure that the detection results only contained human predictions, irrelevant predictions that based on other classes were filtered out. Next, the filtered detection results had to pass through a confidence check with a threshold of 0.5 score to filter out weak detections.

The remaining detection results were human predictions that composed a list of predicted bounding boxes. For each of the predicted bounding boxes, it was compared with the ground truth bounding boxes to obtain the IoU score. For the highest IoU score in the comparison, if its IoU score is greater than 0.5 threshold, the predicted bounding box was considered a match with the ground truth bounding box. Since the criteria of 0.5 IoU score was met, the predicted bounding box was considered a true positive. The predicted bounding boxes and ground truth bounding boxes that had not match yet were further compared to find a match, provided they met the IoU threshold of 0.5. In the end of comparison, if there was ground truth bounding box that did not find a match with the predicted bounding box, it meant the ground truth was not detected. Hence, it was considered a false negative. On the other hand, the predicted bounding boxes that did not have a match with the ground truth were considered extra boxes predicted by the model. However, the extra bounding boxes would not directly be considered as false positive. This was because, in the testing dataset, humans that appeared very tiny or had only a partial body view in the image were not considered as instances. However, some robust pretrained models could detect these difficult cases. Therefore, the extra predicted bounding boxes were drawn on the image and saved to a file location for manual checking of the false positive cases. Only the extra predicted bounding boxes that were totally unrelated to humans were considered false positives.

The process was repeated for all 300 images to obtain the total number of true positive, false negative, and false positive cases. The results were then used to calculate the accuracy, precision, recall, and F1 score using the equations mentioned in section 3.1.3.

3.1.5 False Detection of Cartoons as Humans

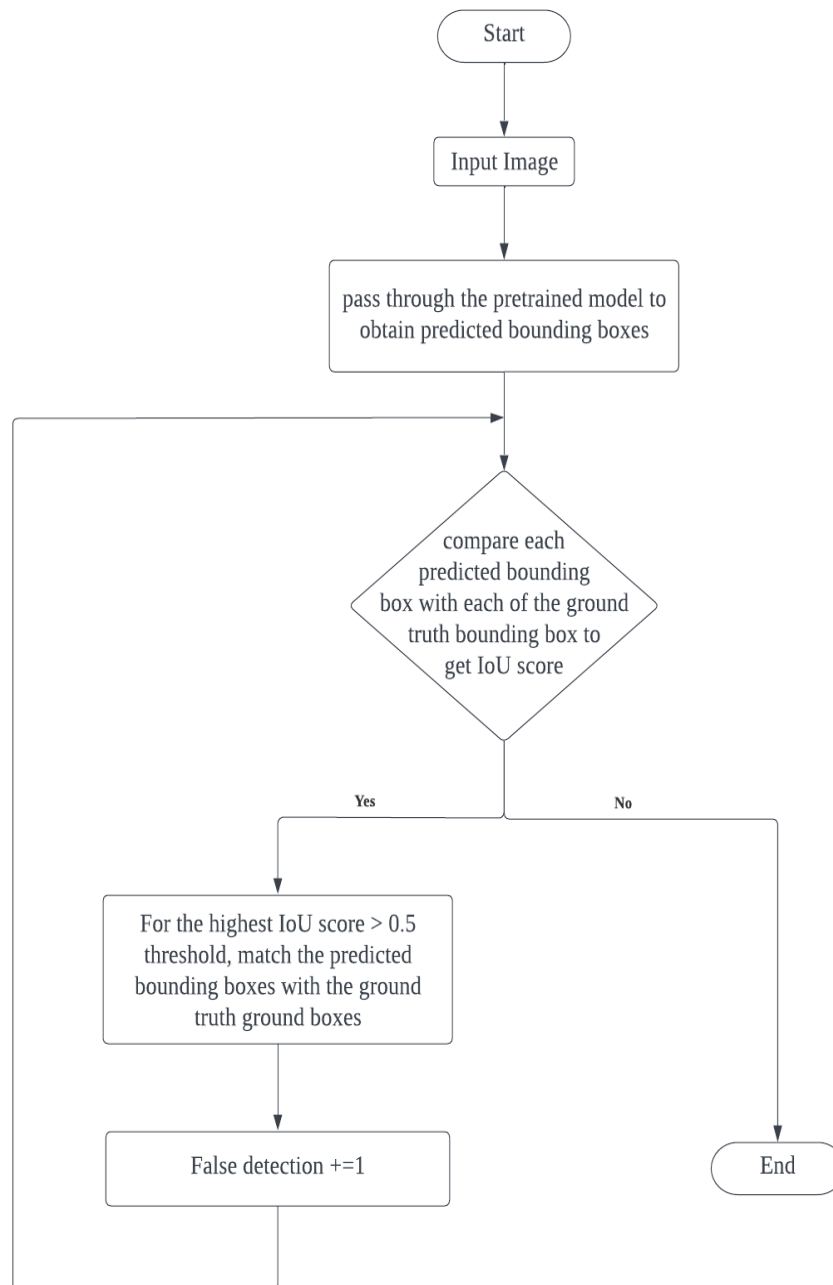


Figure 3.6: Steps to acquire false detection of cartoons as humans.

Each pretrained model was tested on 300 cartoon images to obtain the total number of false detections where the algorithm predicted cartoons as humans. First, the image was loaded into the pretrained algorithm to obtain the predicted result. The prediction result went through class filtering and confidence checks to ensure the filtered prediction result consisted of predicted humans only, with a minimum confidence threshold of 0.5. The filtered

prediction result consisted of a list of predicted bounding boxes. The predicted bounding boxes were compared with each of the ground truth bounding boxes to obtain the IoU score. For the highest IoU score with the condition that met the minimum IoU threshold of 0.5, it meant the predicted bounding box matched the ground truth bounding box. Note that the predicted bounding boxes were the human prediction whereas the ground truth bounding boxes were the cartoon instance. Thus, if there was a match between the predicted bounding box with the ground truth bounding box, it was considered a false detection. The predicted bounding boxes and ground truth bounding boxes that had not match yet were further compared to find a match, provided they met the IoU threshold of 0.5. Since this section only calculated false detections, the remaining ground truth bounding boxes and extra predicted bounding boxes that contribute to false negatives and false positives were ignored. This was to ensure that false detections consisted of predictions that truly detected the cartoon as a human. The process was repeated for 300 cartoon images to obtain the total number of false detections.

3.1.6 Model Training

Figure 3.7 illustrates the steps to perform model retraining.

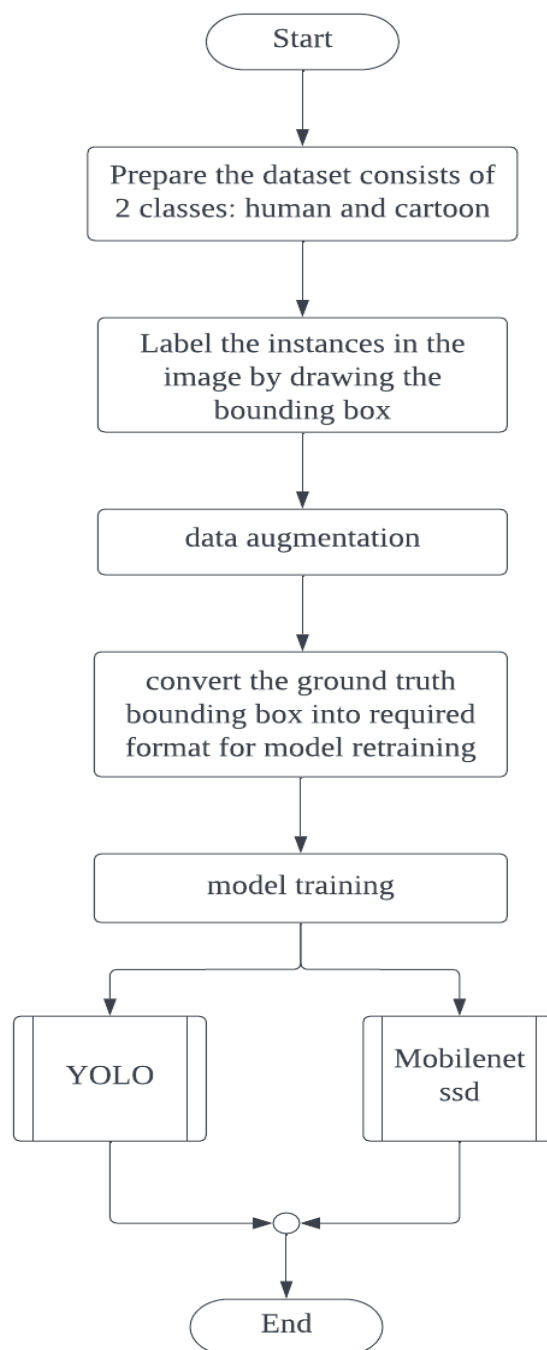


Figure 3.7: Steps for retraining a model.

The retrained model included YOLOv5 series models, YOLOv8 series models and Mobilenet SSD, all of which were trained as two-class models distinguishing between humans and cartoons. To begin, a training dataset with two classes: human and cartoon was prepared. The human dataset was a

combination of the VOC dataset and a random pedestrian dataset that extracted from Roboflow platform (Dwyer and Nelson, 2022). On the other hand, the cartoon dataset was obtained from the iCartoonFace dataset, which could be found on Github (luxiangju, 2021). The ratio of human images to cartoon images was 1:1. Some examples of the dataset were illustrated in Figure 3.8 (a) and (b):

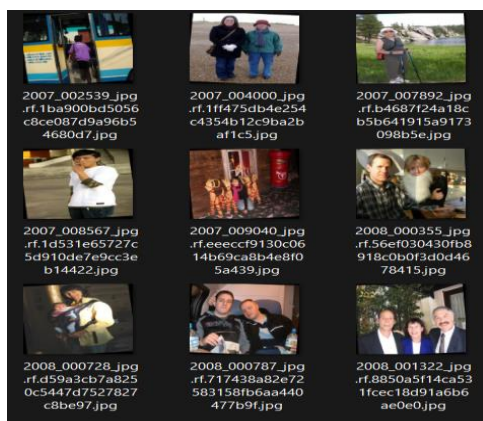


Figure 3.8 (a)

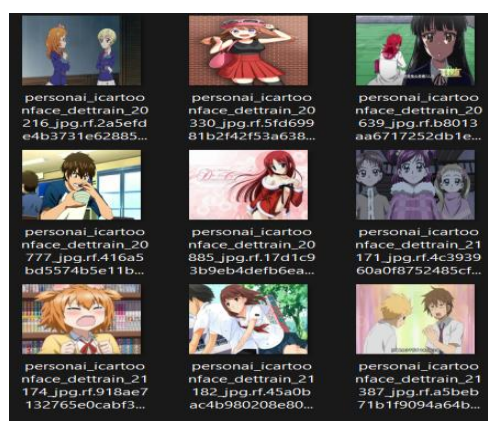


Figure 3.8 (b)

Figure 3.8: Examples of the training dataset.

Next, the instances in the dataset were labelled using the Roboflow platform, with bounding boxes drawn to indicate the location of humans and cartoons in each image. Figure 3.9 shows examples of human instances and cartoon instances, with their corresponding bounding boxes.



Figure 3.9 (a)

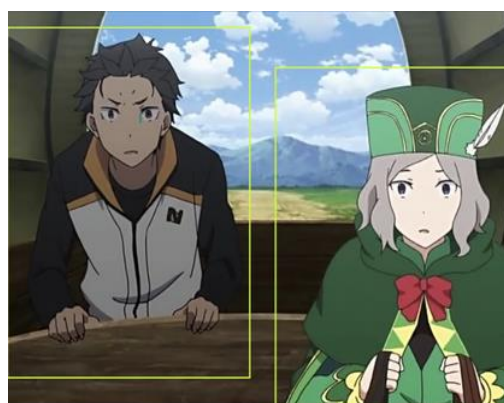


Figure 3.9 (b)

Figure 3.9: Examples of labelled instances.

After labeling, data augmentation was performed to increase the model's robustness, which included techniques such as flipping, 90-degree rotation, and rotation. Figure 3.10 illustrates an example of an image that underwent data augmentation.



Figure 3.10: Data that has undergone data augmentation.

After performing data augmentation, the ratio of instances for human and cartoon was measured, as shown by Figure 3.11. The ratio of cartoons to humans was approximately 1:2.

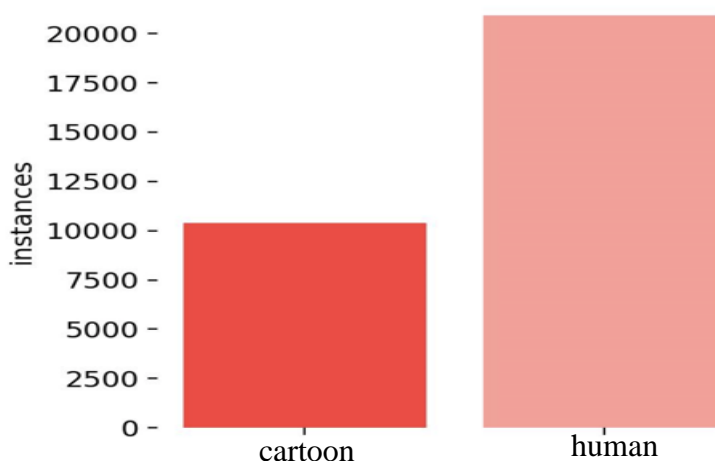


Figure 3.11: Number of instances for cartoon and human.

The next step involved converting the bounding boxes into the required format for model retraining. For YOLOv5 and YOLOv8, the format required of class ID, center-x coordinate, center-y coordinate, width, and height. Meanwhile, MobileNet SSD required bounding boxes to be in XML

format. Once the format of training dataset had been converted, it was ready for model training. Additionally, separate validation and testing datasets were prepared for humans and cartoons, but without undergoing data augmentation.

3.1.6.1 Training Hardware

The models were retrained using the Google Colab platform, which provides hardware specifications as shown in Table 3.2.

Table 3.2: Hardware specifications of Google Colab platform provided.

Type	Model
GPU	Tesla T4 graphic card
System RAM	12.7 GB
GPU RAM	15.0 GB
Programming language	Bash, Python
Framework	Caffe, Pytorch

3.1.6.2 Training of YOLOv5 and YOLOv8

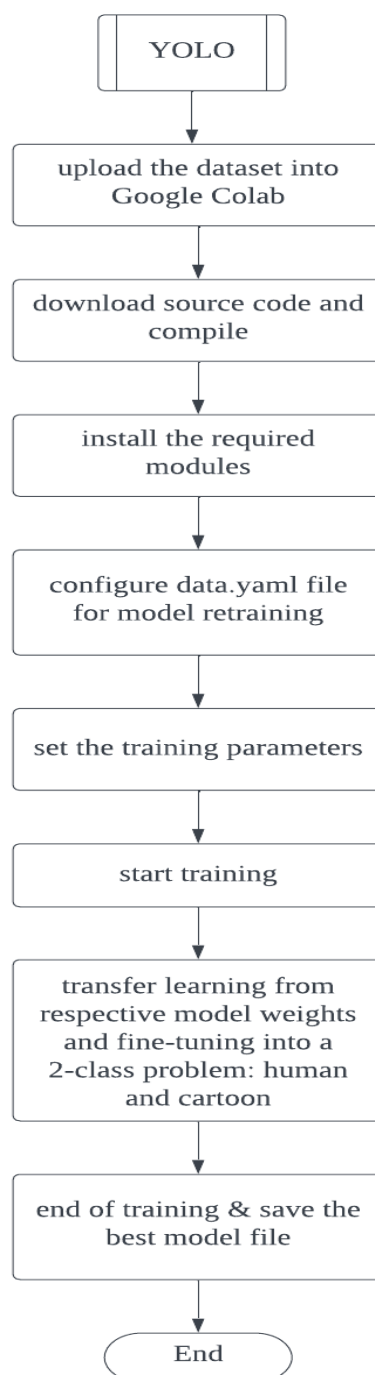


Figure 3.12: Steps for YOLOv5/ YOLOv8 retraining.

The training of YOLOv5 and YOLOv8 began by uploading the dataset into Google Colab. Then, the source code was downloaded from GitHub and compiled, followed by installing the required modules for the model. After that, the data.yaml file was configured to specify the pathway to the training,

validation, and testing images, the number of classes involved in the retrained model, and their respective class names. The training parameters were set to achieve the best possible results. For instance, the feed image size was set to 640, the batch size was set to 32, and the number of epochs was set to 100. The model then started training by transfer learning from the respective model weights and fine-tuning into a 2-class problem: human and cartoon. The training epochs were run until completion. At the end of training, the best model file was saved for performance evaluation purposes.

3.1.6.3 Training of Mobilenet SSD

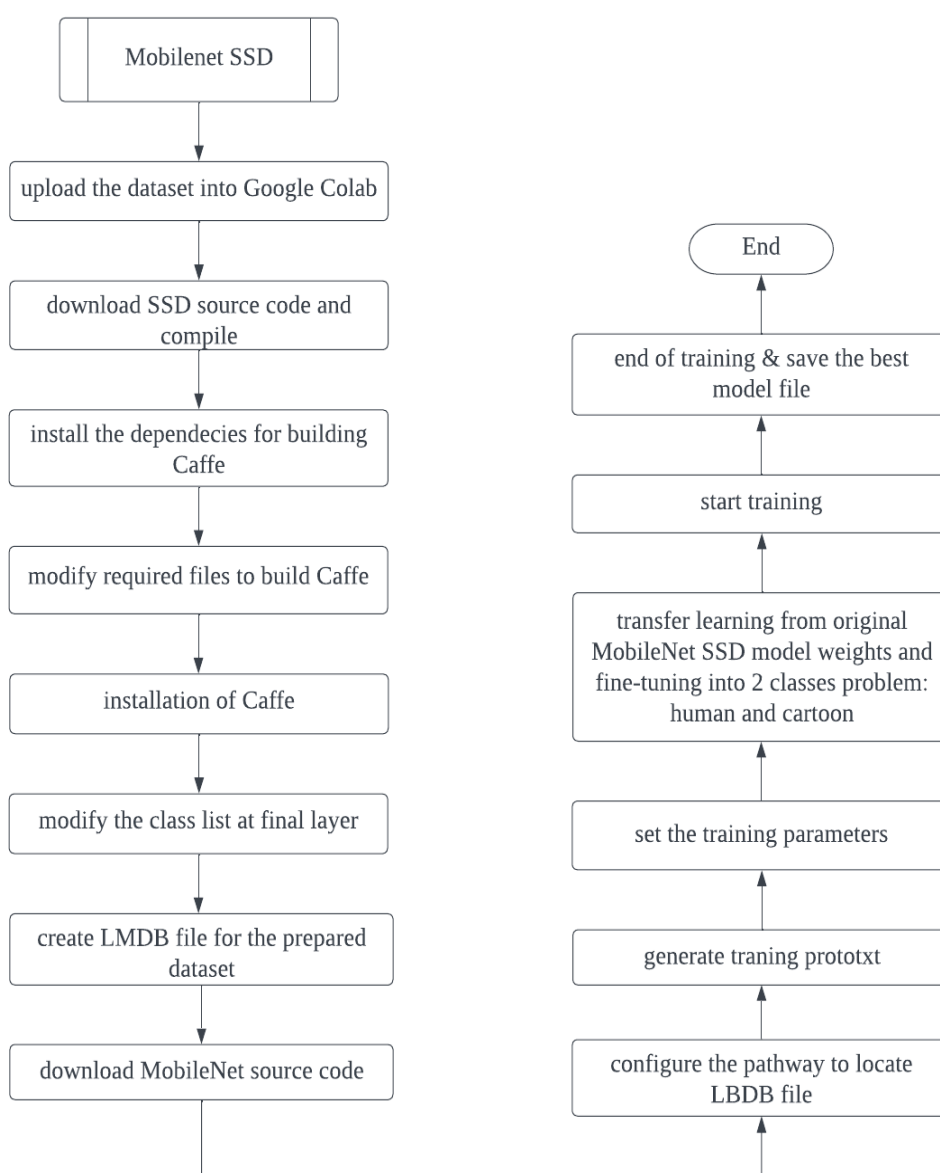


Figure 3.13: Steps for MobileNet SSD retraining.

The training process of MobileNet SSD began with the dataset being uploaded to Google Colab. To build MobileNet SSD, the SSD source code was downloaded and compiled. To run the SSD, Caffe needed to be built and all necessary packages installed. The required files, such as Makefile.config and caffe file, were modified to avoid errors while building Caffe. Once Caffe was installed, the class list in the labelmap_voc.prototxt was modified, and the LMDB file was created for the prepared dataset. The LMDB file was then fed into the MobileNet SSD training. Next, the source code of MobileNet was downloaded from GitHub, and the pathway to locate the LMDB file in the training and testing file was configured. A training prototxt was generated, consisting of three classes, two of which were human and cartoon, and the other one was the background. The training parameters, such as the number of epochs and learning rate, were set. The number of epochs was set to 10 000 iterations, and the learning rate was 0.0002. The training process began by transfer learning from the original MobileNet SSD model weights, followed by fine-tuning it into a two-class problem, human and cartoon. The training process was waited until the epochs finished running, and the best model file was saved with the highest evaluation score on the testing set.

3.2 Overall Framework of Customer Analysis with Machine Vision

Figure 3.14 illustrates the general flow of the proposed solution to perform customer analysis with machine vision.

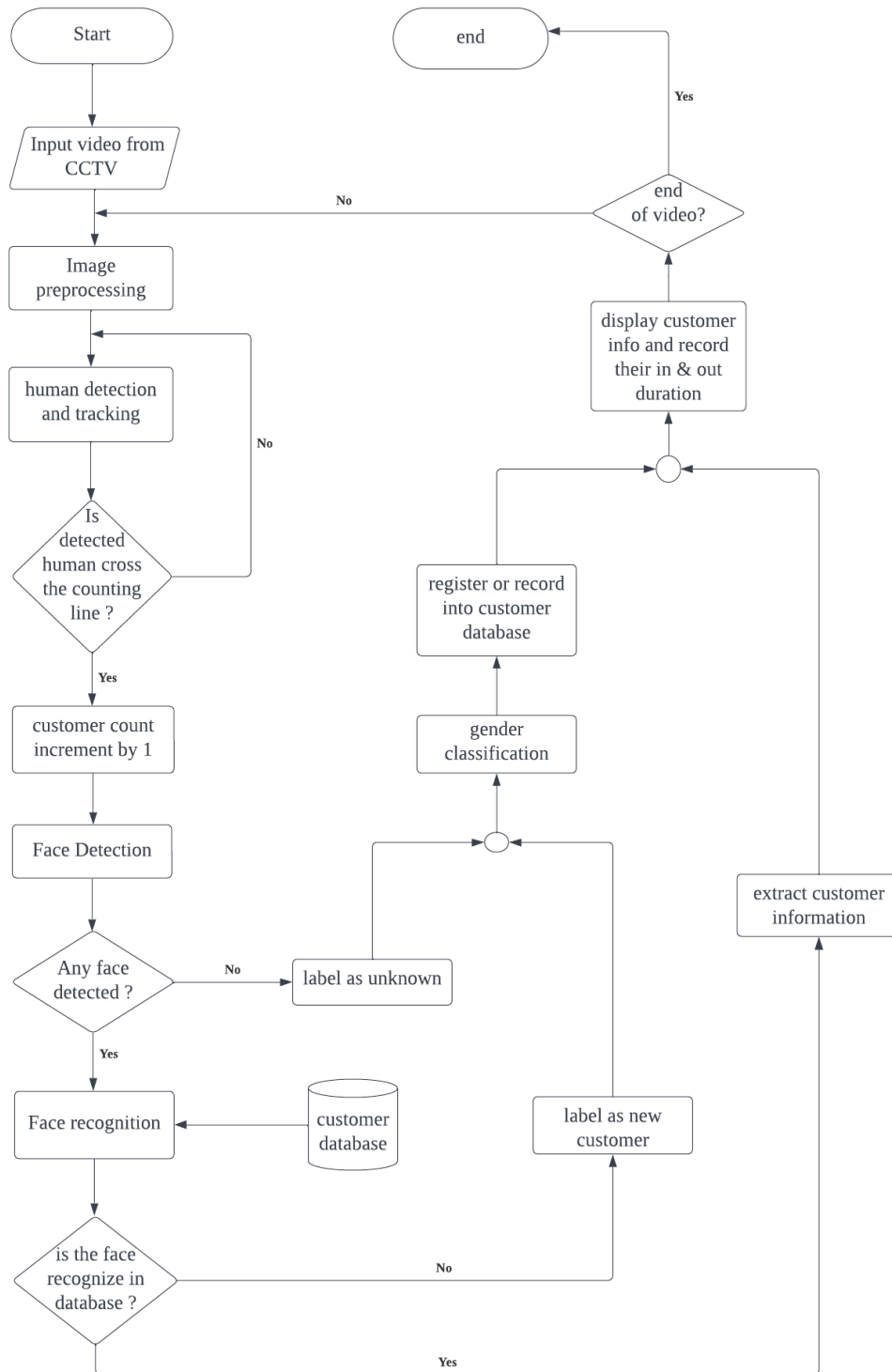


Figure 3.14: Process flow of the proposed solution.

The proposed solution initiates by capturing the real-time video frame from CCTV, which undergoes a series of image preprocessing steps, including resizing and conversion from BGR to RGB. Subsequently, the human detection and tracking phase commences to locate and track humans in subsequent frames. The customer counting increments by 1 if a detected human crosses the counting line. Otherwise, the algorithm will keep tracking the human until the human crosses the counting line. Upon detection, the algorithm captures and crops the customer's image to perform facial recognition and analysis. If the algorithm detects a face in the cropped image, it compares the face with the faces in the databases. The algorithm will extract the customer information if the same face is found. Else, the algorithm will label the human as a new customer. On the contrary, the human will be labelled as unknown if no face was found in the cropped image of that particular human. Since the customer databases do not have the information of new customers and unknown customers, they will go through a further step which is gender classification to determine their gender. After that, their information will register or record into the customer database. Finally, all of the customers' information will be displayed and their in & out time will be recorded as well. The details of human detection, human tracking, customer counting, face detection, face recognition and gender classification will be further illustrated in section 3.2.1.1, 3.2.1.2 and 3.2.1.3, 3.2.2.1, 3.2.2.2 and 3.2.2.3.

3.2.1 Human Detection, Tracking and Customer Counting

Figure 3.15 illustrates the general flowchart of human detection, human tracking and customer counting.

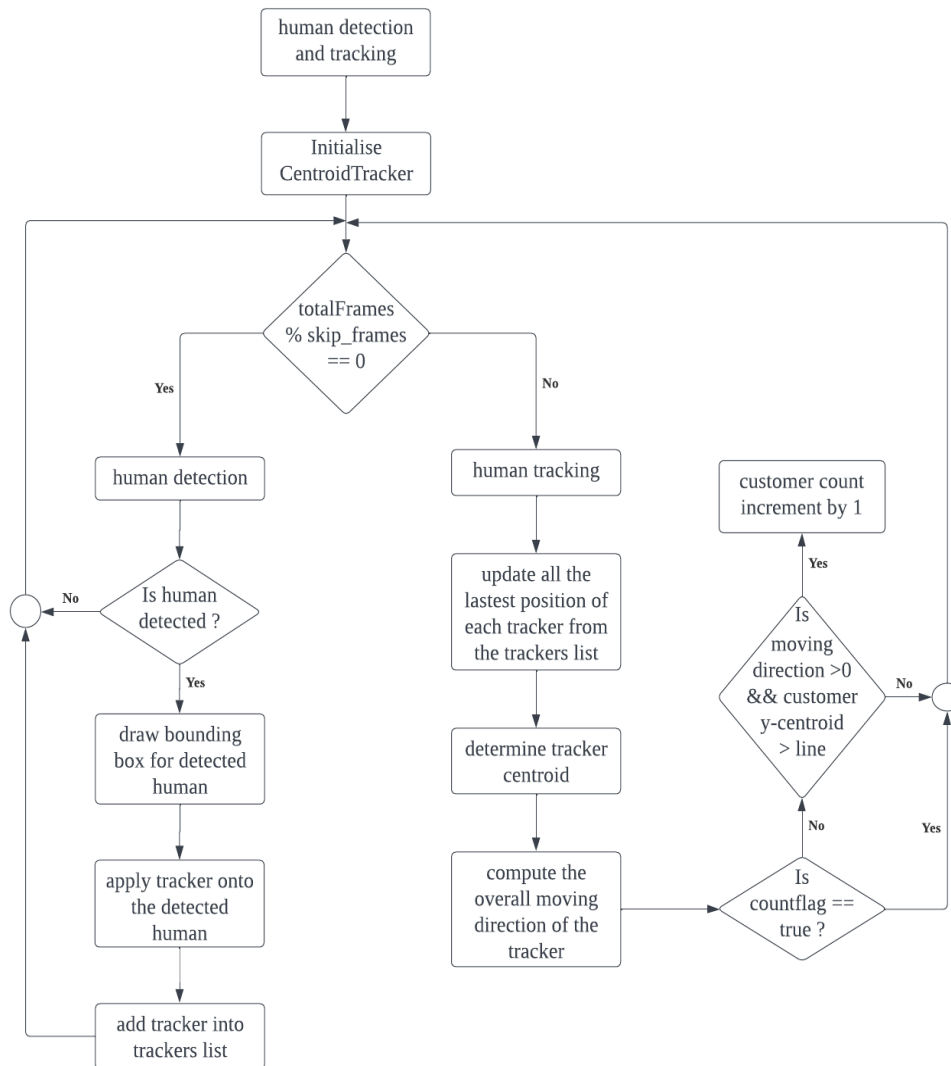


Figure 3.15: Process flow of human detection, tracking and customer counting.

Before the video frame undergoes human detection and tracking, it undergoes image preprocessing to facilitate further analysis. Firstly, the frame is resized from its original size (usually 1920 x 1080 pixels) to a smaller ratio, ensuring that the maximum width of the frame does not exceed 500 pixels. This is done to reduce the computational cost of the algorithm while still maintaining an acceptable level of detail. Secondly, the resized frame undergoes color space conversion from BGR to RGB, as the OpenCV function assumes the color order of the video frame to be BGR.

To begin the human detection and tracking phase, a `CentroidTracker` class is created and initialized to store all the trackable humans' information. During the human detection phase, the algorithm scans the video frame to determine if any human is present. Upon detection, a bounding box is drawn around the human, indicating start x-coordinate, start y-coordinate, end x-coordinate and end y-coordinate of the detected human. The algorithm then applies a tracker to the detected human based on the bounding box coordinates for tracking purposes. All applied trackers are added to the tracker list, indicating the number of humans that need to be tracked in the human tracking phase. While human detection is essential for detecting humans, performing it in every frame is computationally expensive and can slow down the outcome video. Thus, the human detection process will run only once for every N frames to reduce the computational burden of the algorithm. In the meantime, human tracking takes place. The equation for determining the alternation between human detection and human tracking is defined below:

$$\text{totalFrames \% skip_frames} \quad (3.6)$$

where `totalFrames` is the total frames that the algorithm is processing and `skip_frames` is the number of frames, N to undergo human detection. If the outcome of the equation is equal to zero, the algorithm will undergo human detection. Otherwise, the algorithm will perform human tracking.

In the human tracking phase, the algorithm will update the latest position of each detected human from the tracker list to determine their current location in the video frame. The centroid of each human will then be calculated using the bounding box coordinates. The formula for calculating the centroid is as follows:

$$cX = \frac{\text{startX} + \text{endX}}{2.0} \quad (3.7)$$

$$cY = \frac{\text{startY} + \text{endY}}{2.0} \quad (3.8)$$

where `cX` is the centroid coordinate in the x-axis, `cY` is the centroid coordinate in the y-axis, whereas `startX`, `endX`, `startY` and `endY` respectively represent

start x-coordinate, end x-coordinate, start y-coordinate and end y-coordinate of the bounding box. The next step in the algorithm is to compute the overall moving direction of the detected human to determine whether they are entering or leaving the shop. To count the customers, the algorithm will increment by 1 if the detected human fulfills three conditions. The first condition is that the human is moving downwards, which indicates that they are entering the shop instead of leaving it. The second condition is that they cross the counting line to ensure that they have entered the shop. Finally, a count flag is used to prevent the duplicate counting of the same human.

3.2.1.1 Human Detection

The algorithm utilized for human detection is retrained YOLOv8n, which is the best model evaluated from section 3.1. It was selected because it has the overall good performance considering its speed, its ability to detect the human accurately and not detecting cartoons as humans.

Before feeding the video frame (resized frame) to the retrained YOLOv8 model, the video frame needs to append into a list. Next, the list with the video frame was fed to the YOLO model to obtain the detection result. The obtained detection result was filtered to have 'human' class only and the confidence values of 0.6 was set to filter out the weak detections.

3.2.1.2 Human Tracking

The proposed method for human tracking utilizes a correlation tracker based on Danelljan et al. (2014). This particular tracker was chosen for its ability to handle variations in illumination, occlusion, and human pose. Additionally, it excels in handling significant scale variation, which is essential for this project since video frames involve a lot of resizing to reduce computational cost. Unlike other correlation trackers, the chosen method can handle these variations well. Furthermore, its accessibility in the dlib library makes it easy to implement.

Assume the correlation tracker iterates at every time step (t), the input is given as image (I_t), previous scale (s_{t-1}), previous target position (p_{t-1}), previous scale model A_{t-1}^{scale} , B_{t-1}^{scale} and previous translation model A_{t-1}^{trans} , B_{t-1}^{trans} . To perform a robust translation estimation, the translation sample z_{trans}

is first obtained from I_t at s_{t-1} and p_{t-1} . Next, the correlation filter H^l is defined using the equation below:

$$H^l = \frac{\bar{G}F^l}{\sum_{k=1}^d \bar{F}^k F^k + \lambda} \quad (3.9)$$

where \bar{G} is the desired correlation output, F^l is the grayscale image, and \bar{F}^k is the complex conjugation and λ is the parameter that controls the impact of the regularization term. However, the equation 3.9 is costly to solve which involving a $d \times d$ linear system of equations. Thus, the equation 3.9 that represent the correlation filter, H^l is separate into the numerator A_t and denominator B_t and updated accordingly to obtain a robust approximation. The formula of A_t and B_t is computed using the equation below:

$$A_t = (1 - \eta) A_{t-1} + \eta \bar{G}_t F_t \quad (3.10)$$

$$B_t = (1 - \eta) B_{t-1} + \eta \sum_{k=1}^d \bar{F}_t^k F_t^k \quad (3.11)$$

where η is the parameter for learning rate. By using the z_{trans} , A_t and B_t , the translation correlation y_{trans} is computed using the equation below:

$$y = F^{-1} \left\{ \frac{\sum_{l=1}^d \bar{A}^l Z^l}{B + \lambda} \right\} \quad (3.12)$$

where F^{-1} is the inverse discrete fourier transform operator and z is the feature map. Finally, the target position p_t is set by maximizing the score y and the translation model A_t^{trans} and B_t^{trans} is updated. The steps above are repeated from the scale model A_t^{scale} and B_t^{scale} except that the scale sample z_{scale} is obtained from I_t at s_{t-1} and p_t .

3.2.1.3 Customer Counting

The human tracking method discussed in section 3.2.1.2 is useful for tracking humans that have been detected in each frame of the video. However, this method does not recognise each human individually, as it only provides the coordinates of the bounding box. Therefore, in order to perform customer

counting, the algorithm needs to recognise each individual customer and ensure that they are not confused with one another. To achieve this, a CentroidTracker class will be created and initialised, which assigns a unique object ID to each detected human. The CentroidTracker will update the position of each human's centroid in every frame. Centroid tracking will then be performed using these object IDs and updated centroid positions.

In Step 1 of the process, the algorithm first performs human tracking, which generates bounding box coordinates. These coordinates are then passed to the CentroidTracker, which uses Equation 3.7 and 3.8 defined earlier to calculate the centroids and assigns object IDs to each human. This is demonstrated in Figure 3.16.

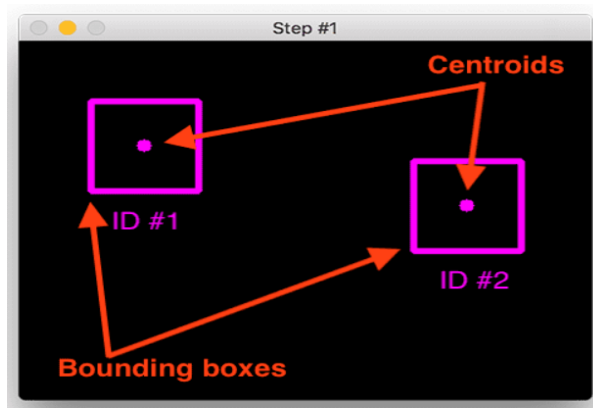


Figure 3.16: Object ID assigned to centroids. (Rosebrock, 2018)

In the subsequent video frame, the algorithm repeats step 1 to calculate the latest centroids based on the current frame. At this point, both the latest centroids (yellow dots) and existing centroids (purple dots) will be present in the frame, as shown in Figure 3.17. However, the job of assigning IDs to the centroids will not be performed yet because some of the latest centroids and existing centroids might correspond to the same person. Figure 3.17 shows that there are three centroid points in the current frame, indicating that an extra person has been detected. If there are two identical humans in the video frame, the algorithm will use Euclidean distance to recognize them as the same person.

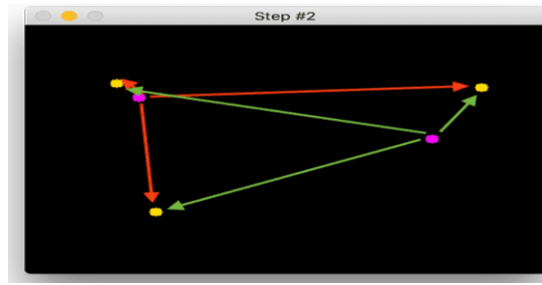


Figure 3.17: Latest and existing centroids in the subsequent frame. (Rosebrock, 2018)

Based on Figure 3.17, the Euclidean distance will be calculated between each of the latest centroids and existing centroids. The equation for calculating Euclidean distance is defined below:

$$E_d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (3.13)$$

where E_d represents Euclidean distance, x_1 and y_1 represent coordinate of first point and x_2 and y_2 represent coordinate of second point. Based on the calculations, the algorithm selects the centroid with the smallest Euclidean distance between the latest centroid and the existing centroid, assuming it is the same person. This is because humans are likely to move between subsequent frames, but the distance moved between the current and previous frames should be the smallest. The assumption made is illustrated in Figure 3.18. To classify as the same person, the maximum distance allowed between subsequent frames is set to 50. This is because humans cannot move such a great distance in just one frame. If the distance is greater than 50, the centroid point will be classified as a new object.

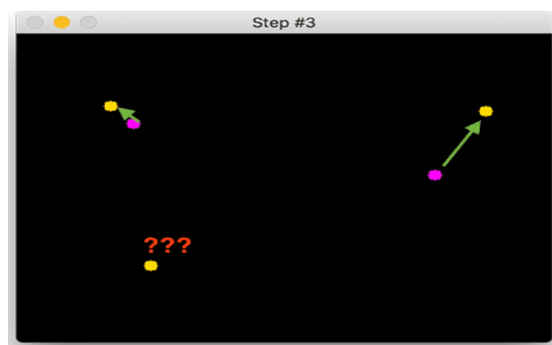


Figure 3.18: Object ID assumption. (Rosebrock, 2018)

Based on Figure 3.18, there is a latest centroid point still left out. Thus, the centroid point will be registered and assigned a new object ID as shown in Figure 3.19.



Figure 3.19: ID assigned to new human. (Rosebrock, 2018)

Regarding the case where a human has disappeared or left the view of the video frame, the centroid tracking algorithm should deregister the object ID. If the detected human is lost for 40 consecutive frames, the object ID will be deregistered in the algorithm.

Since the centroid tracking algorithm can effectively recognize each human, customer counting can be done more efficiently. Normally, the customer will move back and forth around the entrance of the shop. To restrict the duplicate counting of the customer, a count flag will be initialised for each customer. Once the customer crosses the counting line, the count flag will be set to true. This ensures that the customer will not be counted again, even if they move back and forth across the counting line. In this algorithm, the counting line is set at the y-coordinate value of $(H // 2 + 10)$, where H is the height of the video frame. Additionally, to ensure accurate counts, movement in only one direction will be considered for counting. The equation to determine the moving direction is defined as:

$$\text{moving direction} = y_{\text{centroid}} - \bar{y} \quad (3.14)$$

where y_{centroid} is the y-coordinate of the customer's centroid in the current frame and \bar{y} is the mean value of all the y-coordinate centroids in the previous frames. If the moving direction is positive (greater than 0), it indicates that the customer is moving towards the inside of the shop, and the counting condition

will be fulfilled. On the other hand, if the moving direction is negative or zero, the customer will be ignored for counting.

3.2.2 Face Detection, Face Recognition and Gender Classification

Figure 3.20 illustrates the general flowchart of face detection, facial recognition and gender classification.

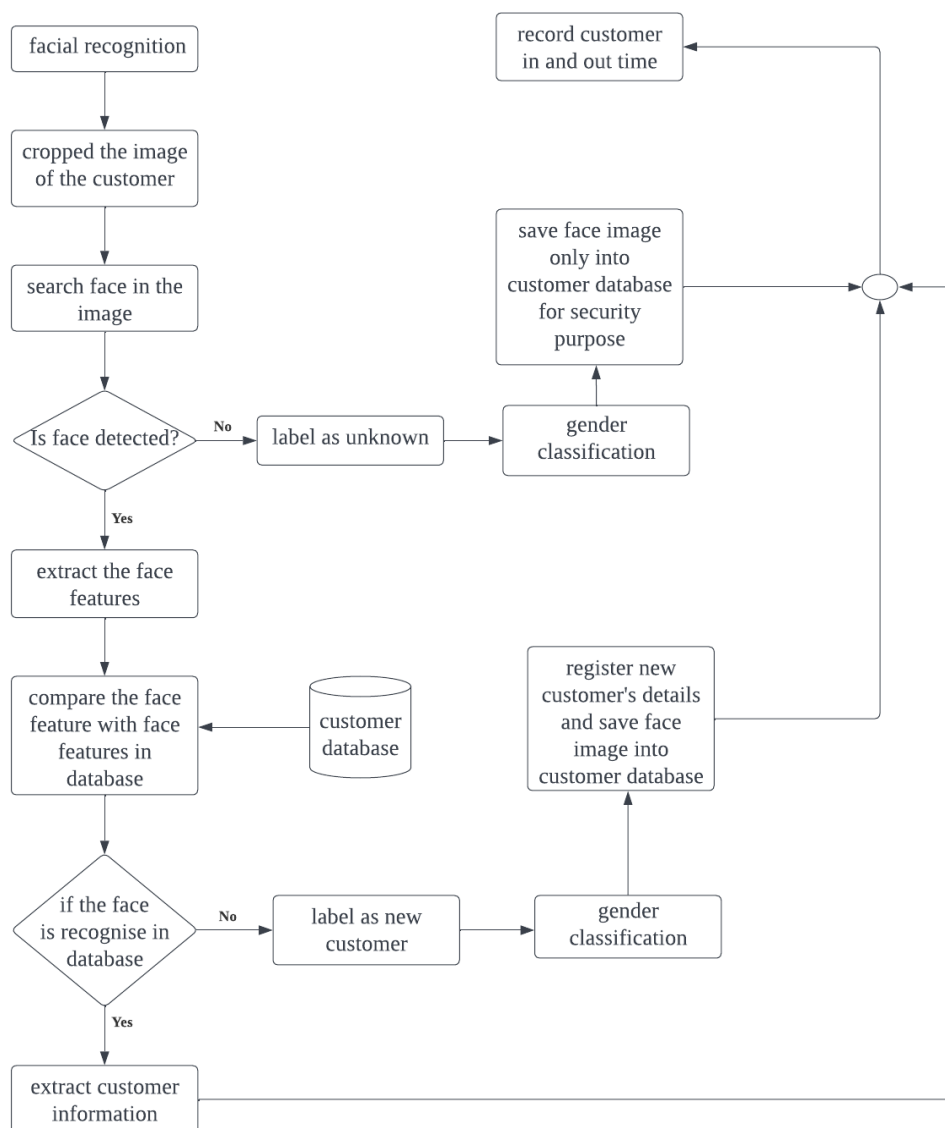


Figure 3.20: Flowchart of face detection, face recognition and gender classification.

When the customer passes the counting line, the face of the customer will be immediately captured at the moment using the original frame instead of using the resized frame. This is to increase the quality of the image for better

analysis in face detection and facial recognition. To accurately locate the customer's face in the original frame, the method first obtains the resized ratio using the following equation:

$$\text{resize_ratio} = \text{frame.shape}[1] / W \quad (3.15)$$

where $\text{frame.shape}[1]$ is to obtain the width of the original frame and W is the width of the resized frame. Next, to obtain the centroid of the customer in the original frame, the centroid of the customer in the resized frame is multiplied by the resize_ratio . Then, a bounding box coordinate is created from the customer centroid to roughly estimate the cropping zone that contains the customer's face. The equation used for creating the bounding box coordinate is defined:

$$(s_x, e_x) = ((\text{cent_x} - 120), (\text{cent_x} + 90)) \quad (3.16)$$

$$(s_y, e_y) = ((\text{cent_y} - 280), \text{cent_y}) \quad (3.17)$$

Where s_x , e_x , s_y , and e_y represent the starting x-coordinate, ending x-coordinate, starting y-coordinate, and ending y-coordinate of the bounding box, respectively. Meanwhile, cent_x and cent_y represent the x-coordinate and y-coordinate of the customer's centroid, respectively.

After cropping the image, the algorithm searches for any face present in the image using face detection. If a face is detected, the algorithm extracts its features and compares them with the known faces in the database. If a match is found, the customer's information is extracted. Otherwise, the customer is classified as a new customer and the gender model is used to determine their gender. The new customer's gender and facial features are then registered in the customer database, along with the cropped image for future reference. If the algorithm fails to detect a face, the customer is labeled as an unknown customer. Their image is still fed into the gender model for customer analysis. As the facial features are not available, the information of unknown customers is not registered into the customer database. However, the cropped image of unknown customers is still saved into the customer database for security purposes.

3.2.2.1 Face Detection

The proposed method for face detection is using the HOG model in the face recognition module. The face recognition module consists of two methods to perform the face detection which are the HOG and CNN model. Although CNN model has a higher adaptation against the challenges such as pose variation and occlusion and result in higher accuracy to detect face, it is slower and not suitable for real-time scenarios. Hence, the HOG model is selected as it is less computationally demanding and faster to run in real-time. However, the HOG model has limitations in detecting faces that are not frontal.

If a face is detected, the proposed method returns the bounding box of the human face in the image using the CSS style format, with the order of top, right, bottom, and left.

3.2.2.2 Face Recognition

The proposed solution for facial recognition is FaceNet, which was suggested by Schroff, Kalenichenko, and Philbin in 2015. This algorithm extracts the face features of an individual by embedding the face identity into 128-dimensional representations, as shown in Figure 3.21. Since FaceNet method is being built into the face recognition module and the pre-trained network is trained by King (2017) with 3 million facial images, it is highly available and easy to access.

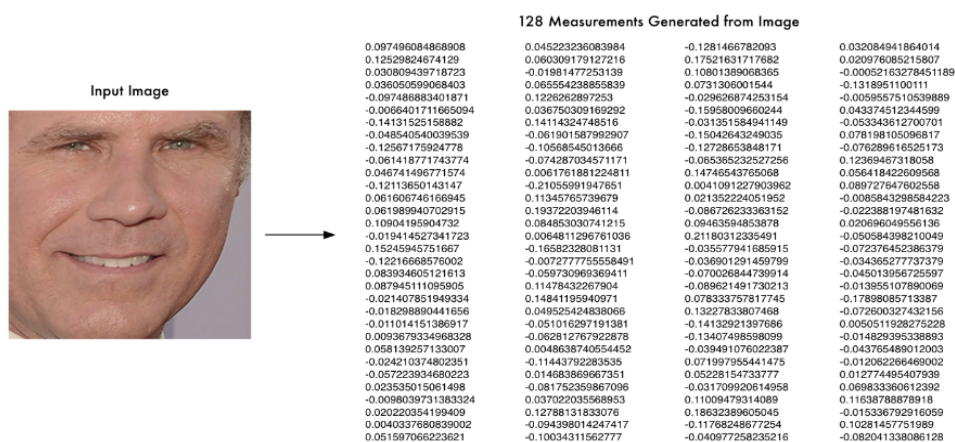


Figure 3.21: 128 Measurements of Facial Image. (Geitgey, 2016)

3.2.2.3 Gender Classification

The proposed solution for gender classification utilizes a simple CNN architecture developed by Levi and Hassner (2015). This method was chosen because it has shown higher accuracy when tested on challenging datasets with variations in pose and lighting conditions.

To implement gender classification, the cropped face image was first converted into a blob to perform mean subtraction and scaling. The mean subtraction was set to 127.5 and no scaling was performed, with the value set to 1. The blob was then passed through the CNN network to obtain the result. Based on the result, the face was classified as male or female, depending on the higher probability score for each gender.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Human Detection Improvement

The result and performance of both the pretrained and retrained models will be discussed on the subchapter 4.1.1 and 4.1.2.

4.1.1 Pretrained Models

The pretrained detection models were evaluated on the testing dataset using a confidence threshold value of 0.5 and IoU score of 0.5. The result of the detections were classified as True Positive (TP), False Positive (FP) and False Negative (FN) for the human class while False Detection (FD) for the cartoon. Since the pretrained detection models does not have ‘cartoon’ as its pretrained class, the result of pretrained models was classified based on the human class only. Thus, True Negative (TN) was not considered in the result as it merely represents the background of the image that does not contribute any true negative to the result. Since the detection models were pretrained to recognize humans only, any cartoon detected as a human will be considered as a false detection. The results are presented in Table 4.1.

Table 4.1: Result of the pretrained models.

Pretrained Model	TN	FN	FP	FD
YOLOV3	387	40	1	114
YOLOV4	370	57	1	134
YOLOV5n	224	203	0	68
YOLOv5s	309	118	1	81
YOLOv5m	366	61	2	128
YOLOv8n	376	51	1	186
YOLOv8s	387	40	1	193
YOLOv8m	392	35	1	202
Mobilenet ssd	389	38	1	247
HOG	127	300	22	73
frcnn-resnet	396	31	58	320

retinanet	387	40	6	193
frcnn-mobilenet	378	49	5	230
ssd_vgg16	357	70	2	229
ssdlite320_mobilenet_v3	375	52	1	224
fcos-resnet	395	32	21	240
maskrcnn-resnet	416	11	28	280
keypointrcnn-resnet	395	32	32	325
frcnn-inception_v2	389	38	11	216
maskrcnn-inception_v2	390	37	17	240
ssd-inception_v2	367	60	6	189
ssdlite_mobilenet_v2	363	64	2	144
ssd_mobilenet_v2	355	72	3	135

These results were further calculated to obtain accuracy, precision, recall, and F1 score for human detection, as well as the false detection rate for the cartoon. Additionally, the evaluation will include parameters and model size for each detection model. Table 4.2 illustrates the results.

Table 4.2: Result of pretrained model.

Pretrained Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	False Rate (%)	Params (M)	Model Size (MB)
YOLOv3	90.42	99.74	90.63	94.97	22.14	61.9	242.20
YOLOv4	86.45	99.73	86.65	92.73	26.02	64.3	251.68
YOLOv5n	52.46	100.00	52.46	68.82	13.20	1.9	3.97
YOLOv5s	72.20	99.68	72.37	83.85	15.73	7.2	14.46
YOLOv5m	85.31	99.46	85.71	92.08	24.85	21.2	41.80
YOLOv8n	87.85	99.73	88.06	93.53	36.12	3.2	6.38
YOLOv8s	90.42	99.74	90.63	94.97	37.48	11.2	22.05
YOLOv8m	91.59	99.75	91.80	95.61	39.22	25.9	50.90
Mobilenet ssd	90.89	99.74	91.10	95.23	47.96	5.8	22.61
HOG	28.29	85.23	29.74	44.10	14.17	3.3	3.78
frcnn-resnet	81.65	87.22	92.74	89.90	62.14	41.8	163.58
retinanet	89.38	98.47	90.63	94.39	37.48	34.0	149.54
frcnn-mobilenet	87.50	98.69	88.52	93.33	44.66	19.4	76.02
ssd_vgg16	83.22	99.44	83.61	90.84	44.47	35.6	139.25
ssdlite320_mobilenet_v3	87.62	99.73	87.82	93.40	43.50	3.4	13.74

fcos-resnet	88.17	94.95	92.51	93.71	46.60	32.3	126.58
maskrcnn-resnet	91.43	93.69	97.42	95.52	54.37	46.4	181.47
keypointrcnn-resnet	86.06	92.51	92.51	92.51	63.11	59.1	231.48
frcnn-inception_v2	88.81	97.25	91.10	94.07	41.94	13.3	55.82
maskrcnn-inception_v2	87.84	95.82	91.33	93.53	46.60	92.4	65.57
ssd-inception_v2	84.76	98.39	85.95	91.75	36.70	20.1	99.59
ssdlite_mobilenet_v2	84.62	99.45	85.01	91.67	27.96	4.5	19.45
ssd_mobilenet_v2	82.56	99.16	83.14	90.45	26.21	6.8	68.06

4.1.1.1 Accuracy

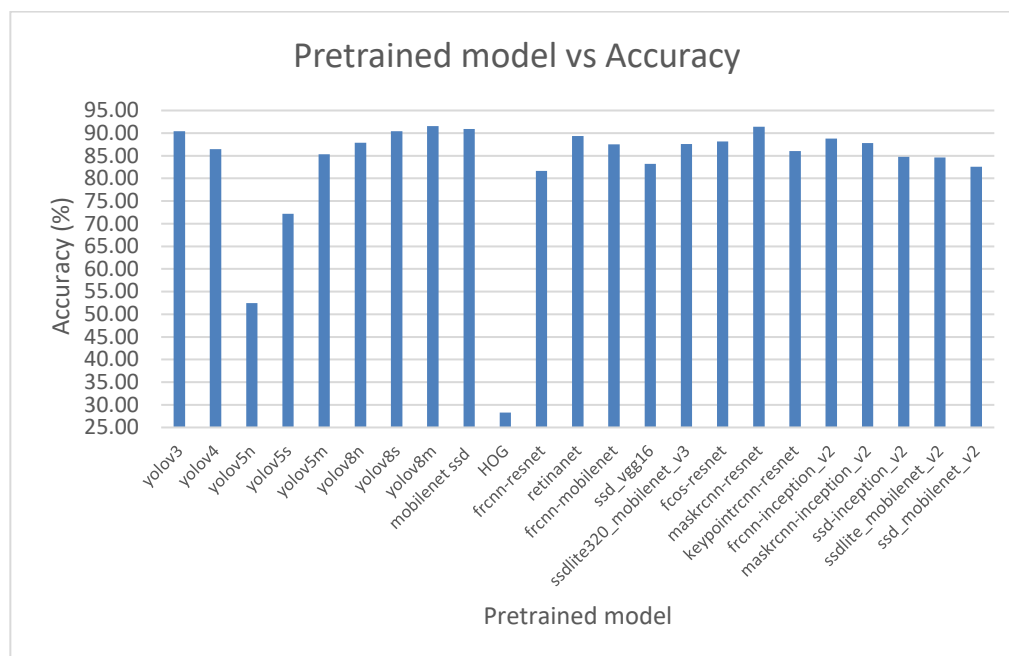


Figure 4.1: Graph of pretrained model vs accuracy.

Figure 4.1 illustrates the accuracy of each pretrained model in detecting humans. According to the Figure 4.1, YOLOv8m achieved the highest accuracy of 91.59 %, followed by maskrcnn-resnet, YOLOv3, and YOLOv8s, which also attained a high accuracy of above 90 % in detecting 427 instances of humans. YOLOv4, YOLOv5m, YOLOv8n, retinanet, frcnn-mobilenet, ssdlite320_mobilenet_v3, fcos-resnet, keypointrcnn-resnet, frcnn-inception_v2, maskrcnn_inception_v2, and ssd-inception were also able to achieve human detection above 85 %. On the other hand, HOG had the lowest accuracy of 28.29 % in detecting humans. This was attributed to its limitation as an older method, as it lacks the ability to capture intricate detail compared to the deep-learning based method.

4.1.1.2 Precision x Recall

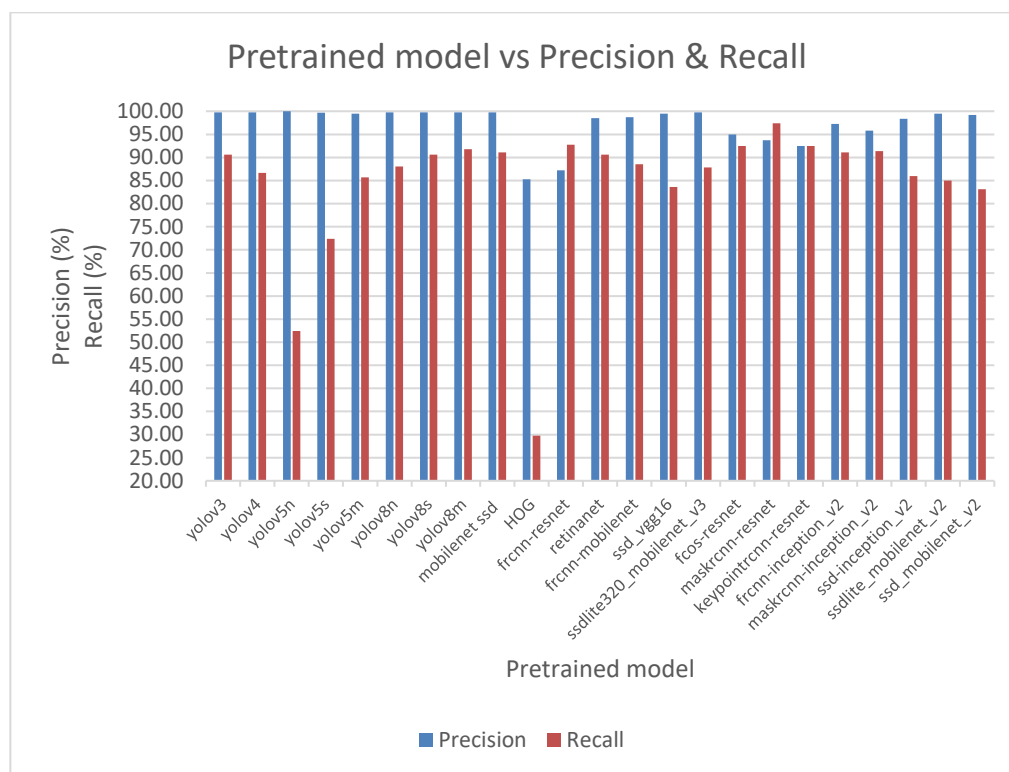


Figure 4.2: Graph of pretrained model vs precision & recall.

In Figure 4.2, the precision and recall relationship for each pretrained model is shown. It was observed that these models were able to achieve high precision in capturing human instances, which indicated their abilities without making many errors in the positive predictions. Only HOG, frfcn-resnet, fcos-resnet, maskrcnn-resnet, and keypointrcnn-resnet recorded precision values below 95%, with HOG recording the least precision at 85.23%. Regarding recall, the results for these pretrained models range from 29.74% to 94.72%. Maskrcnn-resnet achieved the highest recall at 94.72%, which indicated its ability to return human instances and not irrelevant instances. However, it had a lower precision value at 93.69% than other pretrained models, suggesting more false positive predictions. From Figure 4.2, the pretrained models that achieved a precision above 95% and recall above 90% were YOLOv3, YOLOv8s, YOLOv8m, mobilenet_ssd, retinanet, frfcn-inception_v2, and maskrcnn-inception_v2.

4.1.1.3 F1 Score

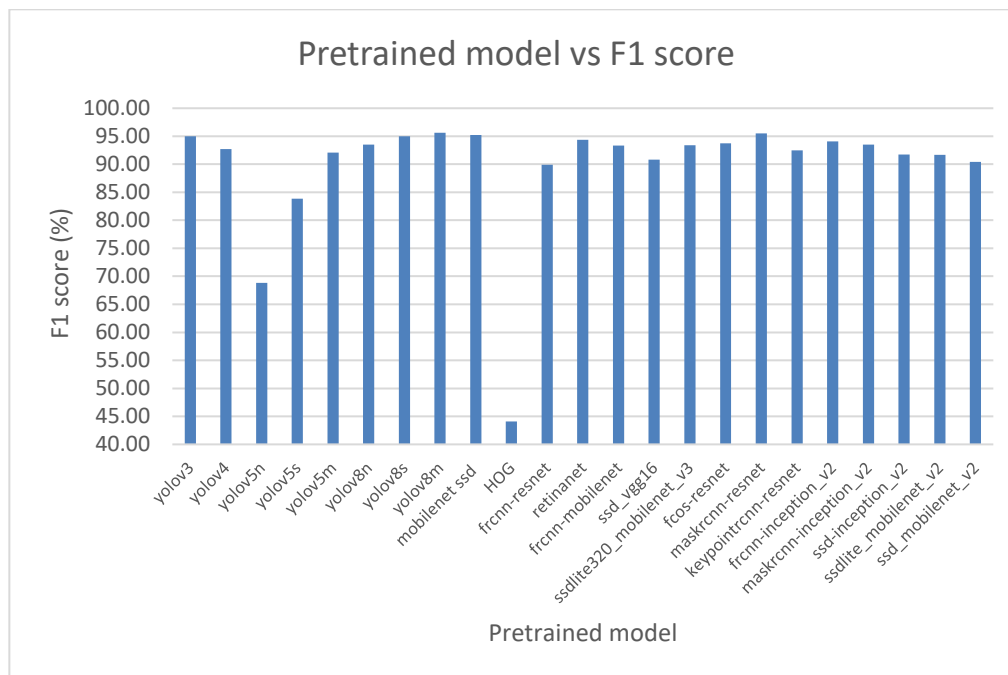


Figure 4.3: Graph of pretrained model vs F1 score.

The F1 score is a combined metric that takes into account both precision and recall. Based on the results presented in Figure 4.3, YOLOv8m, MobileNet SSD, and Mask R-CNN ResNet were able to achieve an F1 score of around 95 %. On the other hand, HOG had the lowest F1 score of 44.10 %.

4.1.1.4 False Rate

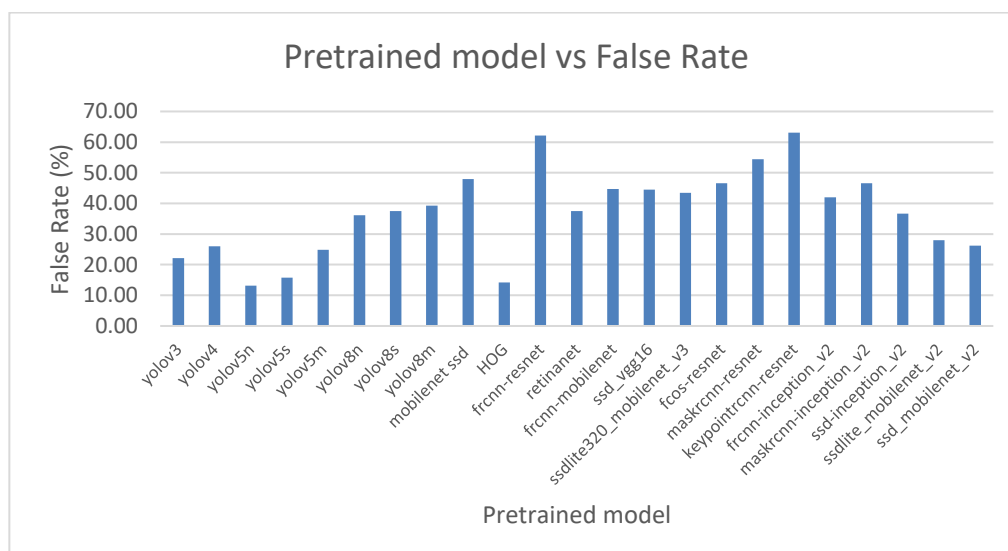


Figure 4.4: Graph of pretrained model vs false rate.

Figure 4.4 shows the false detection rate of each pretrained model in falsely identifying cartoons as humans. Only YOLOv5n, YOLOv5s, and HOG had false detection rates below 20 %. All other pretrained models had a false detection rate of at least 20 %, indicating that for every five cartoon instances, at least one would be incorrectly identified as human.

4.1.1.5 Params x Model Size

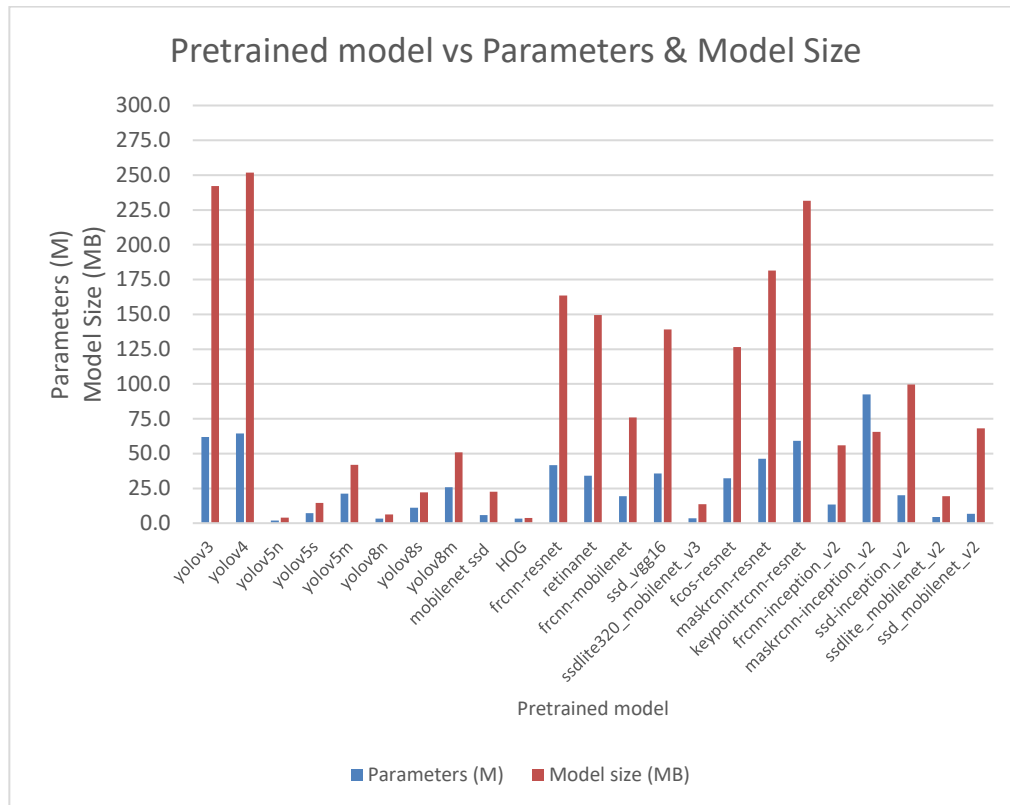


Figure 4.5: Graph of pretrained model vs params & model size.

Figure 4.5 presents the parameters and model size for each pretrained model. The parameters and model size of a pretrained model typically reflect its required time to infer an image. Smaller parameters and model size typically have faster inference speed compared to those that have larger parameters and model size because it involves fewer matrix multiplications (Wulfert, et al., 2022). Based on the Figure 4.5, YOLOv5n, YOLOv5s, YOLOv8n, YOLOv8s, mobilenet_ssd, HOG, ssdlite320_mobilenet_v3, and ssdlite_mobilenet_v2 were the pretrained models that had parameters and model size smaller than 25 M and 25 MB, respectively. This provided a greater advantage for them to be implemented in real-time processing.

4.1.1.6 Summary of the Pretrained Models

Based on the results, it seems that there was no single pretrained model that was able to achieve the perfect combination of high accuracy, precision, recall, F1 score, and low false detection rate while also having small parameters and model size. Each pretrained model had its own strengths and weaknesses.

For example, the mobilenet ssd model had a low parameters and model size, with an accuracy of 90.89 %, precision of 99.74 %, recall of 91.10 %, and F1 score of 95.23 %. However, it also had a high false detection rate at 47.96 %. On the other hand, YOLOv5n, YOLOv5s, and HOG had low false detection rates, but they did not have high accuracy and F1 score compared to other pretrained models.

In summary, when selecting a pretrained model, it is important to consider the specific needs and requirements of the application and weigh the trade-offs between accuracy, precision, recall, F1 score, false detection rate, and computational efficiency.

4.1.2 Retrained Models

The retrained models include YOLOv5n, YOLOv5s, YOLOv5m, YOLOv8n, YOLOv8s, YOLOv8m, and Mobilenet SSD. To enable a fair comparison with the pretrained models, the same classification of results were used for the retrained models. The results were classified into True Positive (TP), False Positive (FP), and False Negative (FN) for human, and False Detection (FD) for cartoon as shown in Table 4.3.

Table 4.3: Result for retrained models.

Retrain model	TN	FN	FP	FD
YOLOv5n	295	132	1	13
YOLOv5s	345	82	2	16
YOLOv5m	368	59	0	18
YOLOv8n	416	11	2	42
YOLOv8s	414	13	2	40
YOLOv8m	416	11	1	77
Mobilenet ssd	322	105	1	20

The results were converted as usual into accuracy, precision, recall, and F1 score for human detection, as well as the false detection rate for cartoon instances, as shown in Table 4.4.

Table 4.4: Result for retrained models.

Retrained Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	False Rate (%)	Params (M)	Model Size (MB)
YOLOv5n	68.93	99.66	69.09	81.60	2.52	0.517	3.81
YOLOv5s	80.42	99.42	80.80	89.15	3.11	1.971	14.10
YOLOv5m	86.18	100.00	86.18	92.58	3.50	5.984	41.23
YOLOv8n	96.97	99.52	97.42	98.46	8.16	-	6.09
YOLOv8s	96.50	99.52	96.96	98.22	7.77	-	21.97
YOLOv8m	97.20	99.76	97.42	98.58	5.24	-	50.79
Mobilenet ssd	75.23	99.69	75.41	85.87	3.88	-	22.61

4.1.2.1 Accuracy

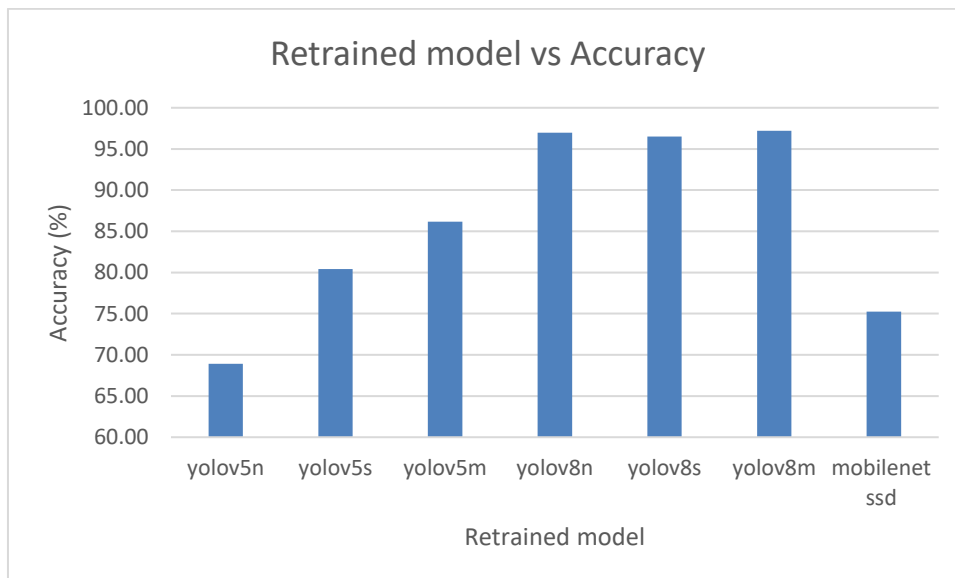


Figure 4.6: Graph of retrained model vs accuracy.

From the given Figure 4.6, it was observed that the YOLOv8 series outperformed the YOLOv5 series and mobilenet ssd in terms of accuracy. Specifically, YOLOv8n, YOLOv8s, and YOLOv8m recorded accuracy rates of 96.97 %, 96.50 %, and 97.20 %, respectively. Conversely, YOLOv5n recorded the lowest accuracy among the retrain models, with an accuracy of 68.73 %.

4.1.2.2 Precision x Recall

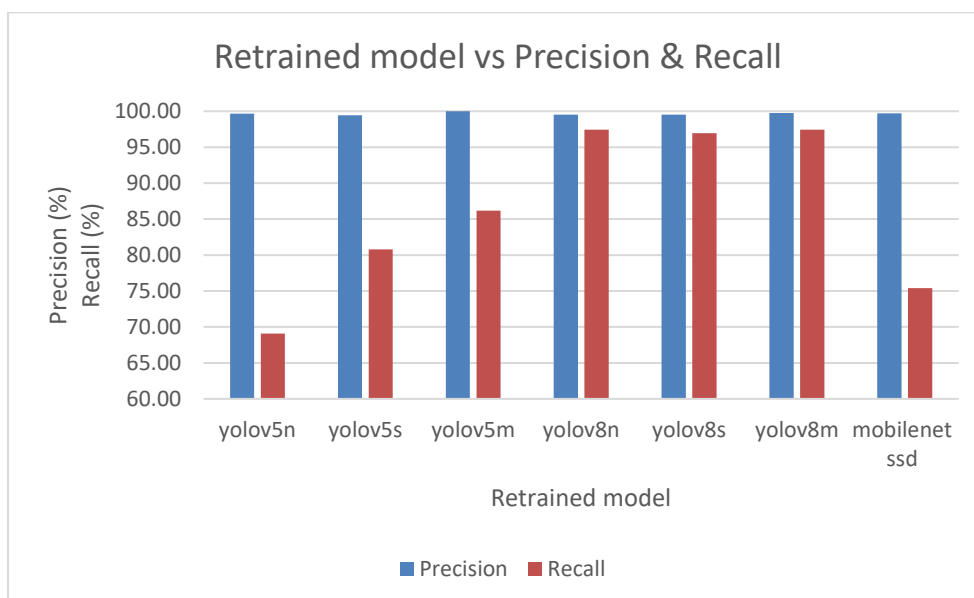


Figure 4.7: Graph of retrained model vs precision & recall.

The data presented in Figure 4.7 indicates that all the retrained models achieve high precision levels of at least 99 %. However, only the YOLOv8 series displayed high recall rates when compared to the other retrain models. Specifically, YOLOv8n, YOLOv8s, and YOLOv8m recorded recall of 97.42 %, 96.96 %, and 97.42 %, respectively. On the other hand, YOLOv5n exhibited the least recall amongst the retrain models, which was 69.09 %.

4.1.2.3 F1 Score

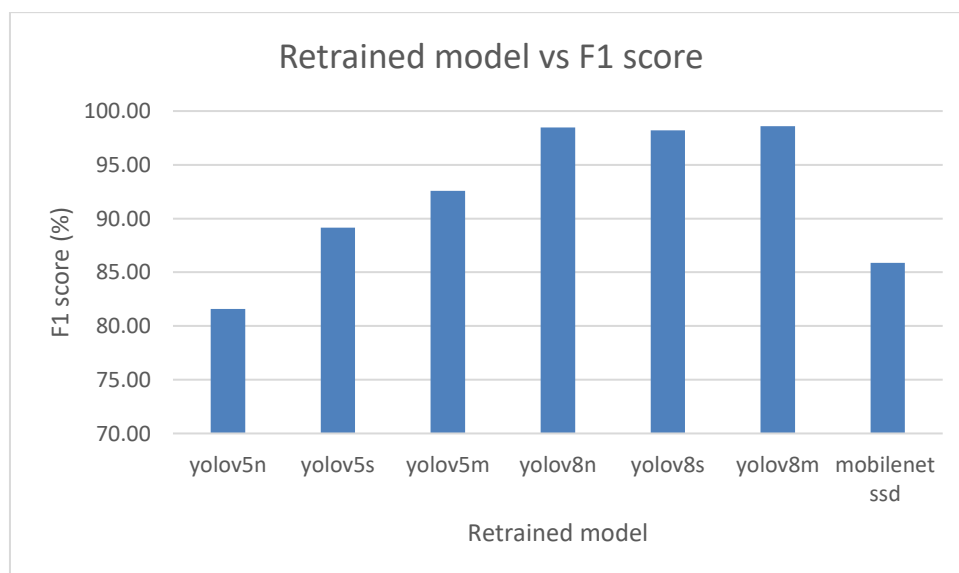


Figure 4.8: Graph of retrained model vs F1 score.

From Figure 4.8, it was observed that YOLOv8 series had the highest F1 score among all the other retrained models. They were able to achieve an average F1 score of 98 %. In comparison, mobilenet ssd had an F1 score of 85.87 %, while YOLOv5n, YOLOv5s, and YOLOv5m had F1 scores of 81.60 %, 89.15 %, and 92.58 %, respectively.

4.1.2.4 False Rate

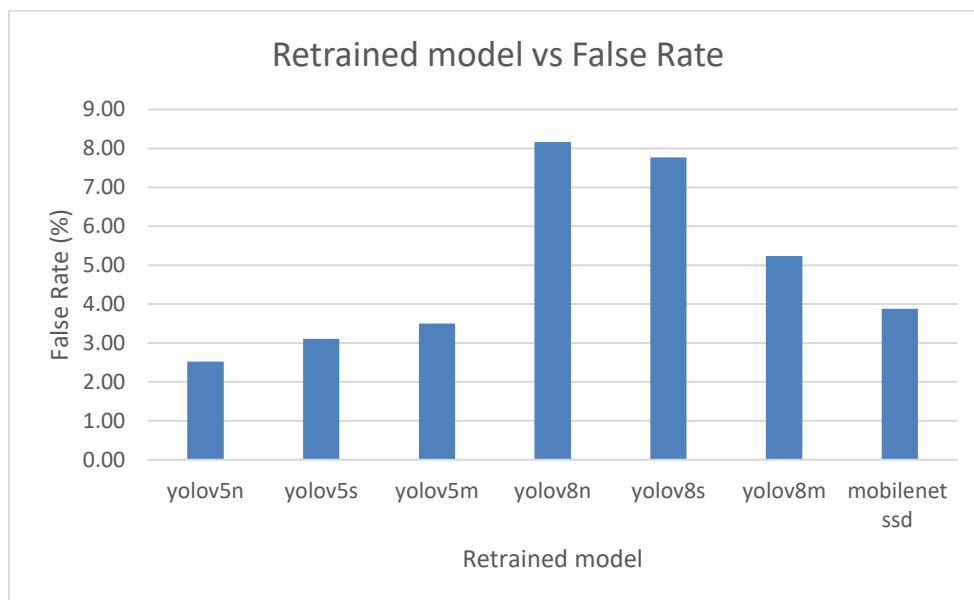


Figure 4.9: Graph of retrained model vs false rate.

According to Figure 4.9, the YOLOv5 series models had the lowest false detection rate among the other retrained models, indicating that they were capable of not detecting cartoons as humans. YOLOv5n, YOLOv5s, and YOLOv5m had false detection rates of 2.52 %, 3.11 %, and 3.50 %, respectively. YOLOv8n recorded the highest false detection rate at 8.16 %.

4.1.2.5 Params x Model Size

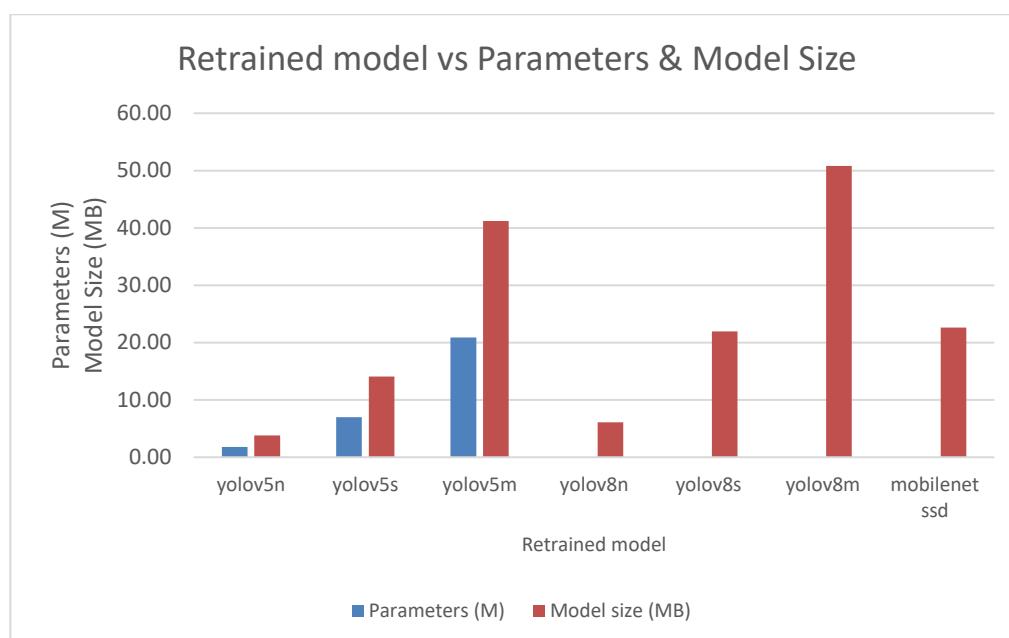


Figure 4.10: Graph of retrained model vs params & model size.

Figure 4.10 shows the model size of each retrained model. The smallest model size was achieved by YOLOv5n at 3.81 MB, followed by YOLOv8n at 6.09 MB, YOLOv5s at 14.10 MB, YOLOv8s at 21.97 MB, and mobilenet ssd at 22.61 MB. Among the YOLO family, the medium models had the largest model sizes, with YOLOv5m at 41.23 MB and YOLOv8m at 50.79 MB.

4.1.2.6 Summary of Retrained Models

When considering the overall performance, YOLOv8n emerged as the best retrained model. It achieved an accuracy of 96.97 %, precision of 99.52 %, recall of 97.42 %, and F1 score of 98.46 %, which surpasses the YOLOv5 series and mobilenet ssd. Although it had the highest false rate among the models at 8.16 %, the false rate was still acceptable, given that its model size was only 6.09 MB, indicating a faster inference speed. In contrast, YOLOv8s and YOLOv8m had similar performance with smaller false rates but their model sizes were almost 3 times and 8 times that of YOLOv8n, respectively, making them slower for real-time processing.

4.1.3 Summary of Pretrained and Retrained Models

The retrained models performed significantly better than the pretrained models in not detecting cartoons as humans. The model with the highest false rate among the retrained models was YOLOv8n with a percentage of 8.16 % which was much lower than all the pretrained models. YOLOv5n, the pretrained model with the least false rate, had a percentage of 13.20 %. Moreover, retrained YOLOv8n outperformed all the pretrained models in terms of accuracy, recall, and F1 score. Its model size of 6.09 MB was only larger than that of pretrained YOLOv5n and the HOG method. Overall, retrained YOLOv8n is the best-performing model among all the pretrained models.

4.1.4 Retrained Models: Classification of Result Based on 2 Classes

In the previous section, the results were presented based on the detection of humans and the false detection rate of cartoons to allow a fair comparison with the pretrained models. However, the retrained models themselves can be evaluated based on two main classes, which were human and cartoon. For detection problems, ‘background’ class would also be considered as part of the result to provide a more comprehensive analysis of model performance. The background class was necessary as it accounted for instances where the model fails to predict the ground truth human or cartoon in the image. These unpredicted instances would classify as false negatives of the background. Additionally, if there were extra predicted bounding boxes by the model where the prediction was not related to cartoon or human instances, it would classify into the false positive of the background. To evaluate the models' performance, a confusion matrix were generated for each retrained model using a confidence threshold of 0.5 and an IOU score of 0.5.

Table 4.5: Confusion Matrix of 2 classes detection for YOLOv5n.

Predicted	Human	295	13	1
	Cartoon	3	987	0
	Background	129	405	0
		Human	Cartoon	Background
		True		

Table 4.6: Confusion Matrix of 2 classes detection for YOLOv5s.

Predicted	Human	345	16	2
	Cartoon	3	140	0
	Background	79	359	0
		Human	Cartoon	Background
		True		

Table 4.7: Confusion Matrix of 2 classes detection for YOLOv5m.

Predicted	Human	368	18	0
	Cartoon	3	188	0
	Background	56	309	0
		Human	Cartoon	Background
		True		

Table 4.8: Confusion Matrix of 2 classes detection for YOLOv8n.

Predicted	Human	416	42	3
	Cartoon	1	253	0
	Background	10	220	0
		Human	Cartoon	Background
		True		

Table 4.9: Confusion Matrix of 2 classes detection for YOLOv8s.

Predicted	Human	413	39	3
	Cartoon	7	269	2
	Background	7	207	0
		Human	Cartoon	Background
		True		

Table 4.10: Confusion Matrix of 2 classes detection for YOLOv8m.

Predicted	Human	416	26	1
	Cartoon	0	285	4
	Background	11	204	0
	Human	Cartoon	Background	
	True			

Table 4.11: Confusion Matrix of 2 classes detection for Mobilenet ssd.

Predicted	Human	312	19	1
	Cartoon	71	281	3
	Background	44	215	0
	Human	Cartoon	Background	
	True			

The results from the confusion matrix were utilized to compute accuracy, precision, recall, F1 score, and combined accuracy for each class as shown in Table 4.12 below. The table presents the metric values for each class, as well as the overall combined accuracy of the retrained models at a 0.5 confidence threshold and 0.5 IoU score.

Table 4.12: Result of the 2 classes detection.

Retrain model	Human				Cartoon				Combined Accuracy (%)
	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	
YOLOv5n	84.52	95.47	69.09	80.17	55.36	97.00	18.83	31.54	41.57
YOLOv5s	89.41	95.04	80.80	87.34	59.96	97.90	27.18	42.55	51.38
YOLOv5m	91.83	95.34	86.18	90.53	64.97	98.43	36.50	53.25	59.02
YOLOv8n	94.07	90.24	97.42	93.69	72.17	99.61	49.13	65.80	70.79
YOLOv8s	94.09	90.77	96.72	93.65	73.07	96.76	52.23	67.84	72.02
YOLOv8m	95.99	93.91	97.42	95.63	75.29	98.62	55.34	70.90	74.02
Mobilenet ssd	85.73	93.98	73.07	82.22	67.44	79.15	54.56	64.59	62.68

4.1.4.1 Human Class x Accuracy, Precision, Recall, F1 score

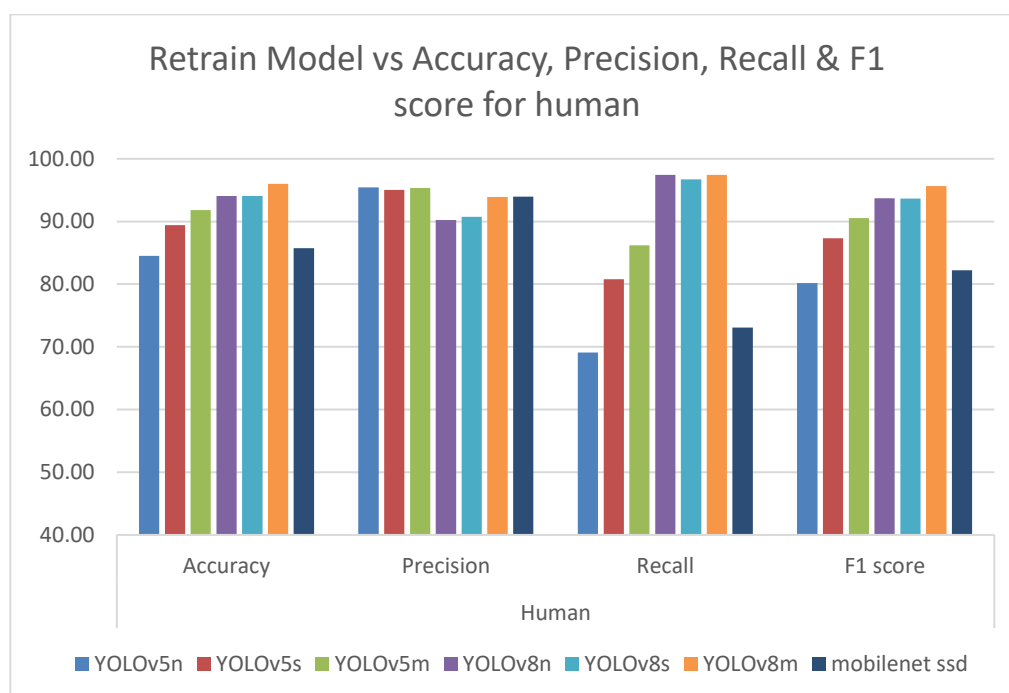


Figure 4.11: Graph of retrained model's human class vs accuracy, precision, recall & F1 score.

Figure 4.11 illustrates the accuracy, precision, recall, and F1 score of the human class for each retrained model. The results in this section were different from the section 4.1.2 since the result in the section 4.1.2 was not influenced by another class, cartoon. The true negative, false positive, and false negative contributed by the cartoon class may have affected the results. According to the Figure 4.11, the YOLOv8 series showed a significant advantage in accuracy, recall, and F1 score compared to the YOLOv5 series and mobilenet ssd. Although the precision for the YOLOv8 series was lower than the YOLOv5 series and mobilenet ssd, the recall for YOLOv8 was significantly better than the YOLOv5 series and mobilenet ssd, leading to a higher F1 score for YOLOv8 series, which was 93.69 % for YOLOv8n, 93.65 % for YOLOv8s, and 95.63 % for YOLOv8m. In contrast, for the YOLOv5 series, the highest F1 score was YOLOv5m at 90.53 % and the lowest F1 score was YOLOv5n at 80.17 %. The mobilenet ssd had an intermediate F1 score of 82.22 % between the YOLOv5 series. Based on the overall result, this matched the result in the previous section that the YOLOv8 series is the better retrained model than others.

4.1.4.2 Cartoon Class x Accuracy, Precision, Recall, F1 score

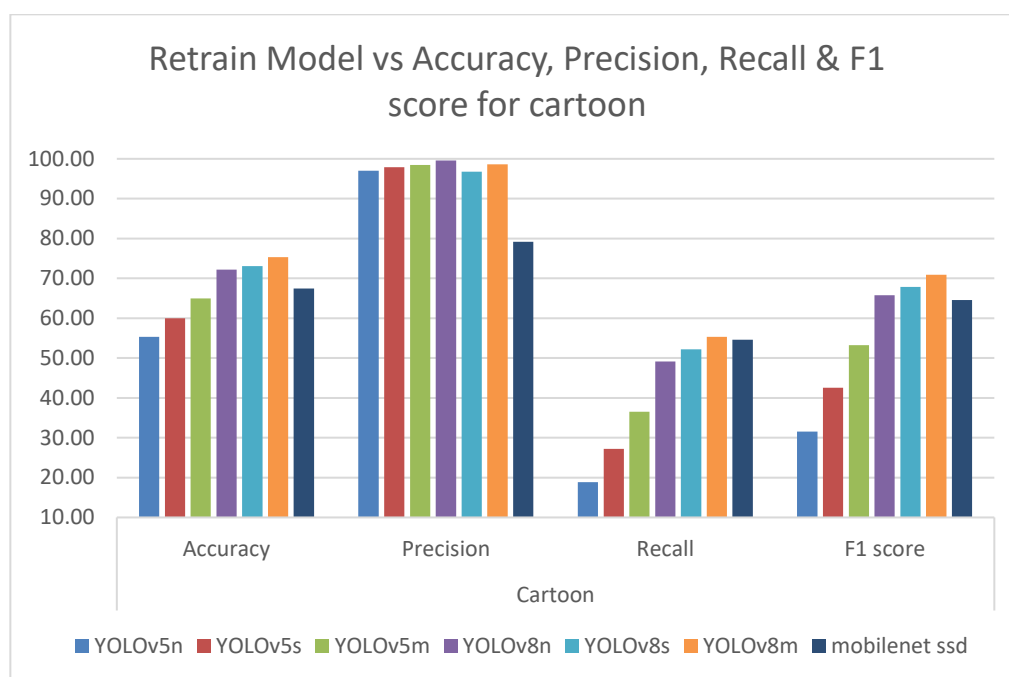


Figure 4.12: Graph of retrained model's cartoon class vs accuracy, precision, recall & F1 score.

Figure 4.12 presents the accuracy, precision, recall, and F1 score for the cartoon class. As shown in Figure 4.12, YOLOv8m achieved the highest accuracy in detecting the cartoon class correctly at 75.29 %, followed by its variants, YOLOv8s at 73.07 % and YOLOv8n at 72.17 %. Although mobilenet ssd had a lower accuracy than YOLOv8 series, it still outperformed the YOLOv5 series, with an accuracy of 67.44 %. YOLOv5n had the lowest accuracy among all models at 55.36 %. Regarding precision, YOLOv5 series and YOLOv8 series performed similarly well, while mobilenet ssd had a precision of only 79.15 %. All retrain models exhibited low recall percentage, below 60 %. YOLOv5n had the lowest recall percentage at 18.83 %, while YOLOv5m achieved the highest recall at 36.50 % among the YOLOv5 series. YOLOv8m had the highest recall percentage at 55.34 %. Regarding the F1 score, YOLOv8 series demonstrated the best performance, followed by mobilenet ssd and YOLOv5 series. YOLOv8n had an F1 score of 65.80 %, lower than its variants, YOLOv8s at 67.84 %, and YOLOv8m at 70.90 %.

4.1.4.3 Combined Accuracy

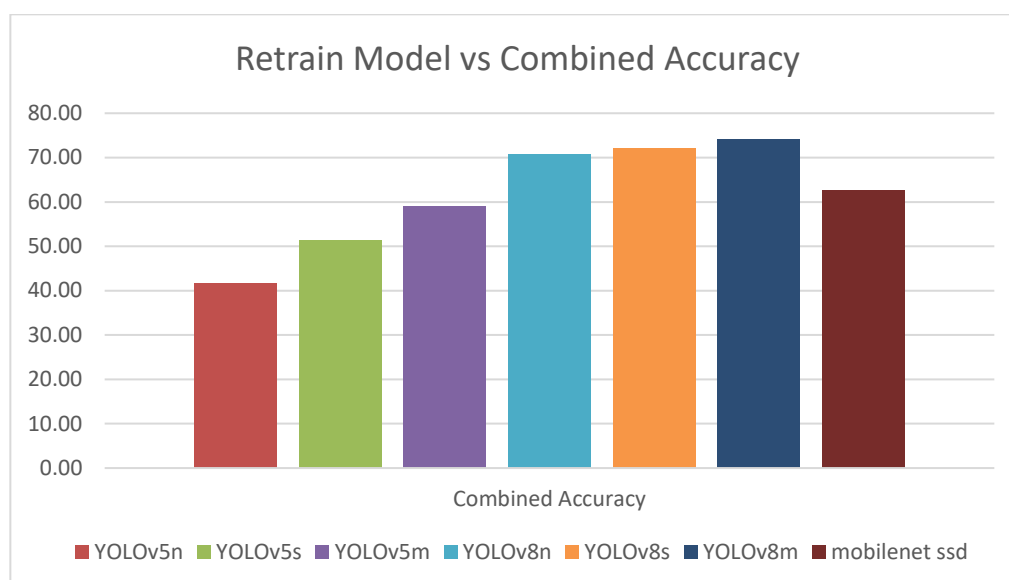


Figure 4.13: Graph of retrained model vs combined accuracy.

Figure 4.13 displays the combined accuracy for human and cartoon classes for each retrained model. Based on the results, the YOLOv8 series had a higher combined accuracy than the mobilenet ssd and YOLOv5 series. YOLOv8m had the highest combined accuracy at 74.02 % due to its higher accuracy in detecting human and cartoon. YOLOv8s was 2 % less combined accuracy, at 72.02 % while YOLOv8n at 70.79 %. Mobilenet ssd had the intermediate combined accuracy in between YOLOv5 series and YOLOv8 series, which was 62.68 %. YOLOv5 series had the least combined accuracy, which was 41.57 % for YOLOv5n, 51.38 % for YOLOv5s and 59.02 % for YOLOv5m.

4.1.4.4 Summary

After analyzing the classification results into two classes, it was observed that the retrained models have a higher ability to detect humans but a lower ability to detect cartoons accurately. However, the main objective of the project is to implement a human detection model that only detects humans and reduces the false detection of cartoons as humans. The model does not need to detect cartoons accurately, but rather to avoid detecting them as humans. Regardless of the classification method, whether one class or two classes, the YOLOv8 series consistently outperforms the other retrained models. Therefore, the YOLOv8 series can be considered the best model among the retrained models.

4.1.5 YOLOv8 series

The retrained YOLOv8 series models were only be tested on the mixed dataset. The dataset comprises 75 images, all of which contain a combination of both humans and cartoons. The dataset contains a total of 146 instances of humans and 259 instances of cartoons, which account for 36.05 % and 63.95 %, respectively. Figure 4.14 displays some of the images in the dataset.

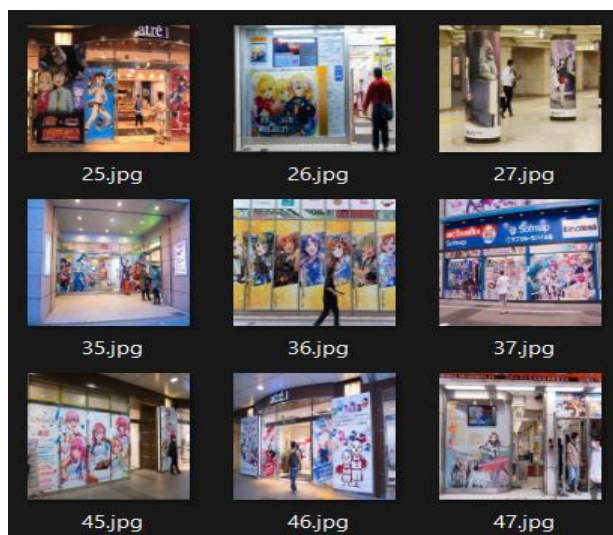


Figure 4.14: Examples of the mix dataset.

As the project's goal is solely to recognize humans and avoid falsely detecting cartoons as humans, the classification of results were based on the human class and the false detection rate of cartoons. The Table 4.13 displays the result.

Table 4.13: Result of YOLOv8 series on mix dataset.

Retrain Model	TP	FN	FP	FD
YOLOv8n	127	19	0	15
YOLOv8s	126	20	0	21
YOLOv8m	126	20	0	13

These results were further calculated to obtain accuracy, precision, recall, and F1 score for human detection, as well as the false detection rate for the cartoon. The results were shown as Table 4.14.

Table 4.14: Result of YOLOv8 series on mix dataset.

Retrained Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	False Rate (%)
YOLOv8n	86.99	100	86.99	93.04	5.79
YOLOv8s	86.30	100	86.30	92.65	8.11
YOLOv8m	86.30	100	86.30	92.65	5.02

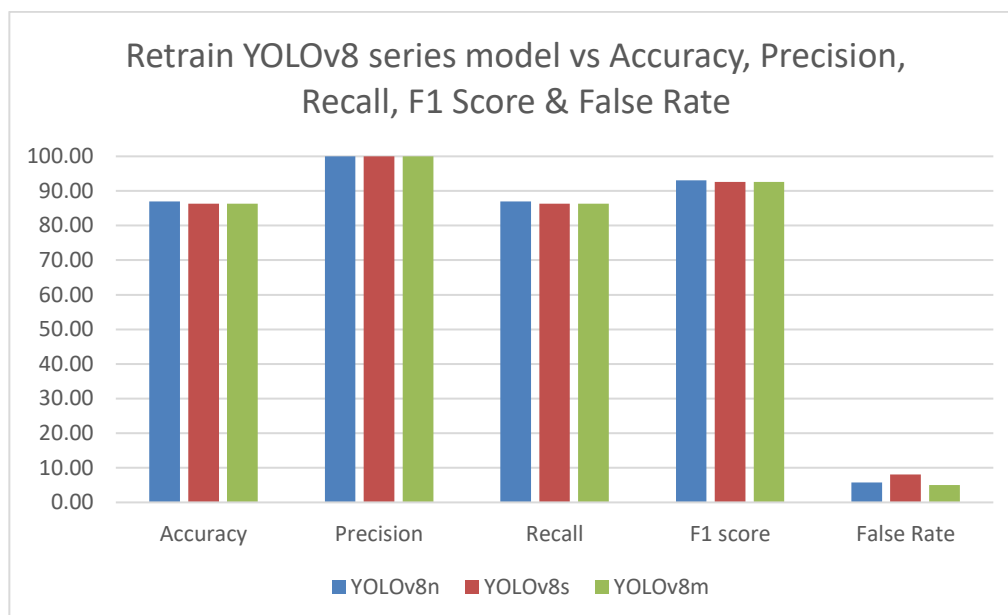


Figure 4.15: Graph of YOLOv8 series vs accuracy, precision, recall & F1 score.

Based on the results shown in Figure 4.15, it was observed that the YOLOv8 series models had very similar performance on the dataset. The main difference in performance was the false detection rate of cartoons as humans. YOLOv8m had the smallest false detection rate at 5.02 %, followed by YOLOv8n at 5.79 % and YOLOv8s at 8.11 %.

Figure 4.16 (a), (b), (c) and (d) depict the scenario of the human detection results. As shown in Figure 4.16 (a) and (b), the retrained YOLOv8n successfully detected all humans without producing any false detection of cartoons as humans. However, in Figure 4.16 (c) and (d), some of the cartoons were erroneously detected as humans, while all humans were accurately detected.



Figure 4.16 (a)



Figure 4.16 (b)

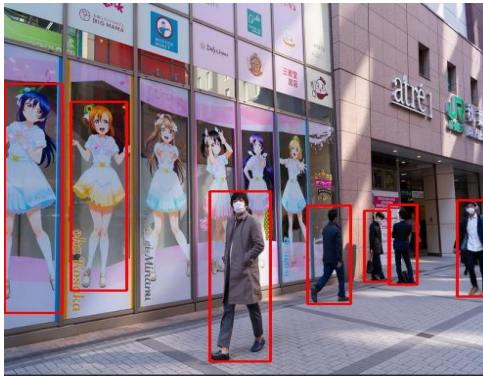


Figure 4.16 (c)



Figure 4.16 (d)

Figure 4.16: Examples of the scenario of human detection result.

Among the retained YOLOv8 series models, YOLOv8n was chosen for implementation in real-time processing. In terms of performance, the YOLOv8 series models did not exhibit significant differences. Although YOLOv8m achieved the best performance overall, its larger model size made it unsuitable for real-time processing. On the other hand, YOLOv8n had a smaller model size of 6.09 MB, making it faster to implement in real time. This represents a trade-off between accuracy and speed.

4.2 Overall Result

This section discussed about the result of proposed solution for customer analysis with machine vision. The retrained human detection model, YOLOv8n was utilised as the human detection model in the proposed solution.

4.2.1 Input Videos and Customer Database

The proposed solution for customer analysis with machine vision was conducted on a set of real-life practical videos that were captured by a CCTV system. The dataset included both an entrance video and an exit video, as illustrated in figure 4.17.



(a) Entrance Video

(b) Exit Video

Figure 4.17: Real-life practical set videos.

Figure 4.18 displays the facial images of the 17 known customers included in the customer database used for the real-life practical case study. Each customer was represented by their high-resolution frontal facial image, and their information, including their name, was encrypted and saved into the database. Similarly, Figure 4.19 shows a portion of the customer information, such as the 128-embedded point of the face and the respective name of each customer.



Figure 4.18: Facial Image in the Customer Database.

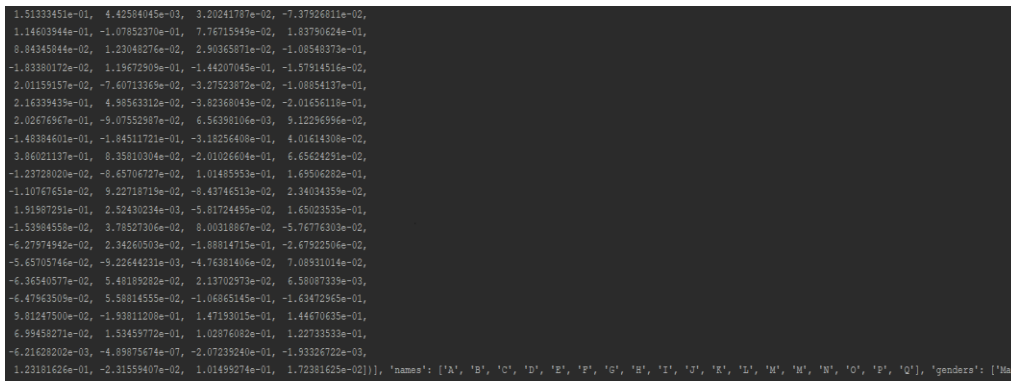


Figure 4.19: Part of the customer information in customer database.

4.2.2 Human Detection, Human Tracking and Customer Counting

In Figure 4.20, (a) displays the video frame when a human first appears and has not yet been detected by the algorithm. In (b), the algorithm has detected the human and assigned an ID to them. Once the human crossed the counting line, as shown in (c), the customer count was incremented by 1, indicating that they have entered the shop. (d) illustrates how the algorithm performed human tracking, following the human assigned with the ID in (b) as they moved.



Figure 4.20 (a)



Figure 4.20 (b)



Figure 4.20 (c)



Figure 4.20 (d)

Figure 4.20: Demonstration of human detection, tracking and customer counting.

From Figure 4.21, the algorithm was able to detect multiple humans at the same time as shown in Figure 4.21 (a) and tracked each of them correctly as shown in Figure 4.21 (b).



Figure 4.21 (a)

Figure 4.21 (b)

Figure 4.21: Human detection and tracking in different frame.

Throughout the case study, the algorithm successfully recorded the count of 13/13 humans in the entrance video and 15/16 in the exit video, resulting in an accuracy of 100% for the entrance video and 93.75% for the exit video. However, one human was not counted in the exit video as he was moving very quickly behind another human, making it difficult for the algorithm to detect him accurately before he passed the counting line. Therefore, the count was missed. Figure 4.22 illustrates this missed count of human.

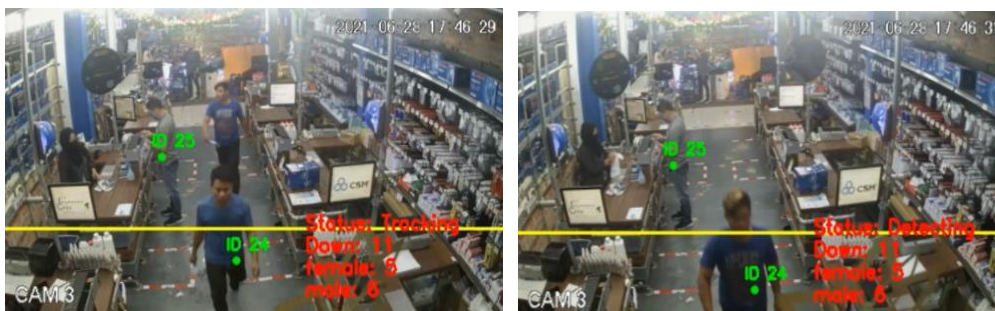


Figure 4.22: Missing count of human.

4.2.3 Facial Recognition

To perform facial recognition, the proposed algorithm cropped the image of the customer at the moment they crossed the counting line and checked if a face was present in the frame. If a human face was detected in the frame, the algorithm proceeds to feature extraction and face recognition.

4.2.3.1 Face Detection

Table 4.15 and Table 4.16 demonstrate the ability of the algorithm to detect the face.

Table 4.15: Result of Face Detection in Entrance Video.

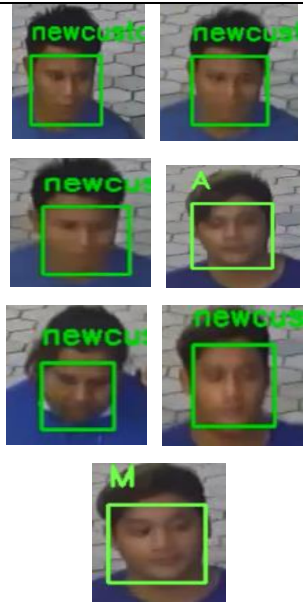




	Success	Fail
Face		
Total	7	4

Table 4.16: Result of Face Detection in Exit Video.

	Success	Fail
Face		

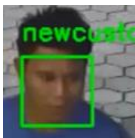

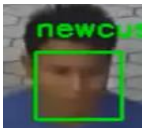

		
Total	5	10

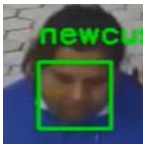
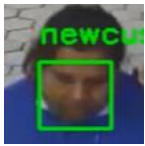


According to Table 4.15 and Table 4.16, the algorithm was successful in detecting frontal faces, faces looking left, and faces looking downwards up to 30 °. However, the algorithm failed to detect faces wearing masks, faces looking downwards at an angle greater than 30 °, or faces facing backward. This was due to the algorithm's inability to capture and detect faces under these challenging conditions. Additionally, low image resolution may contribute to the algorithm's failure to detect faces accurately. The results showed that the algorithm successfully detected 7 out of 11 faces in the entrance video and 5 out of 15 faces in the exit video. Therefore, the success rate of the algorithm to detect faces in the entrance and exit videos was 63.63 % and 33.33 % respectively.

4.2.3.2 Face Recognition

The facial recognition results were relied on the faces detected by the algorithm in section 4.3.1. Currently, the customer database for the entrance video and exit video were independent to allow for better analysis of the facial recognition process.

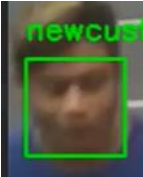




Table 4.17: Result of Face Recognition in Entrance Video.

Customer	Success	Fail
New customer 1 	 	

New customer 2 		
Q 		
Total	4	3

Based on the results from section 4.2.3.1, seven faces were detected in the entrance video. The algorithm was able to recognise two faces from the initial customer database, while the other faces were registered as new customers and recognised based on the updated database. Out of the seven faces, four were recognised successfully, while three faces failed to be recognised from the customer database. Therefore, the accuracy of the algorithm in recognising faces is 57.14 %.

Table 4.18: Result of Face Recognition in Exit Video.

Customer	Success	Fail
New customer 1 		
New customer 2 		
Total	4	1

Same technique was applied to the exit video. From the exit video, five faces were detected in the previous section. Based on it, the algorithm did not recognise any face from the initial customer database while registering two customers as new customers in the customer database. Out of the five faces, four faces were recognised successfully and one face was failed to recognise from the customer database. The recognition rate of the algorithm was 80 % for the exit video.

4.2.3.3 Overall Result for Facial Recognition

Facial recognition comprised the combination of face detection and face recognition. The combined result were classified in the Table 4.19 and Table 4.20.

Table 4.19: Overall Combined Result of Facial Recognition in Entrance Video.



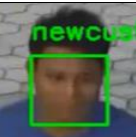
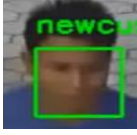



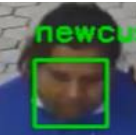
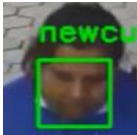





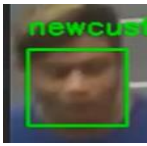
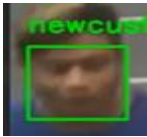
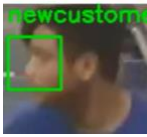

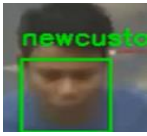
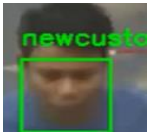
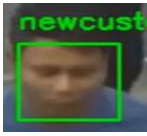





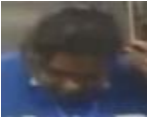








Customer	Success	Fail
New customer 1 	  	  
New customer 2 		 
Q 		 
Total Detected Face	4	3
Total Undetected face	0	4
Total Face	4	7

Table 4.20: Overall Combined Result of Facial Recognition in Exit Video.

Customer	Success	Fail
New customer 1 	 	
New customer 2 	 	 
Unknown 1 		   
Unknown 2 		 
Unknown 3 		
Unknown 4 		
Total Detected Face	4	1
Total Undetected face	0	10
Total Faces	4	11

According to Tables 4.19 and 4.20, the facial recognition algorithm's accuracy for the entrance and exit videos was 36.36 % and 26.67 %, respectively. The algorithm's low accuracy was primarily due to its inability to efficiently detect faces under challenging conditions.

4.2.4 Gender Classification

Table 4.21 and Table 4.22 summarized the result of gender classification in the entrance video and exit video.

Table 4.21: Confusion Matrix of Gender Classification in Entrance Video.

		Predicted		Total
		Male	Female	
Actual	Male	8	0	8
	Female	0	3	3
Total		8	3	11

Table 4.22: Confusion Matrix of Gender Classification in Exit Video.

		Predicted		Total
		Male	Female	
Actual	Male	4	6	10
	Female	2	3	5
Total		6	9	15

Based on Table 4.21 and Table 4.22, the algorithm had an accuracy of 100 % and 46.67 % for gender classification in the entrance and exit videos, respectively. The discrepancy in accuracy between the two videos may be attributed to the higher concentration of humans in the exit video, which could make it more difficult for the algorithm to accurately classify gender. Additionally, misclassification of gender during the initial registration process could lead to incorrect customer information being stored in the database, resulting in subsequent incorrect gender recognition.

4.2.5 Updated Customer Database

Figure 4.23 (a) depicted the folders for new customers and unknown customers that were added or recorded in the customer database. If a customer's face was detected but not recognised by the algorithm, the customer will be registered as a new customer, as illustrated in Figure 4.23 (b). Conversely, if the algorithm failed to detect the face, the customer will be recorded as unknown, as shown in Figure 4.23 (c).

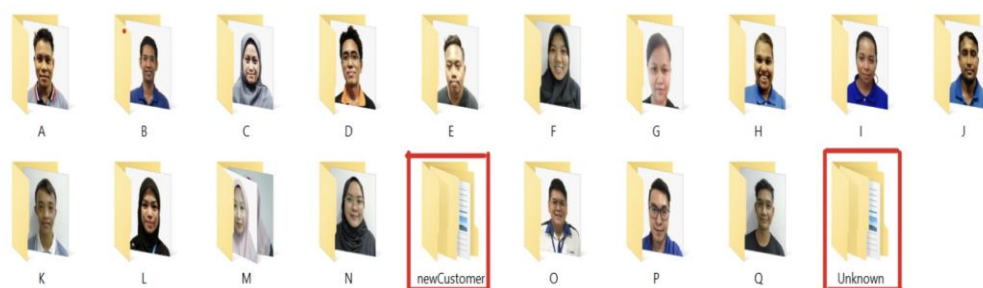


Figure 4.23 (a)

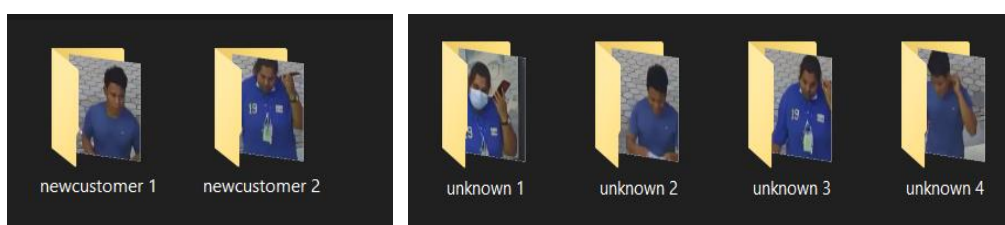


Figure 4.23 (b)

Figure 4.23 (c)

Figure 4.23: Updated customer database.

4.2.6 Weakness of the Algorithm

The algorithm consists of several major steps, including human detection and tracking, customer counting, facial recognition, and gender classification. However, most of the steps has its own limitations when facing certain situations, which can affect the overall performance of the algorithm.

During the human tracking phase, the algorithm may encounter a weakness when tracking people who are standing very close or overlapping with each other. As depicted in Figure 4.25, if the people were too close, the algorithm may wrongly assign IDs to the wrong individuals, leading to incorrect tracking results. In Figure 4.25 (a), for example, the algorithm assigned ID 10 to the customer with the yellow shirt but failed to detect the human with the black shirt, who was occluded by the previous human due to

overlapping. In Figure 4.25 (b), ID 10 was tracking the wrong individual with the black shirt, instead of the correct human with the yellow shirt. This was due to the centroid tracking algorithm, which always assumed that the minimum Euclidean distance between centroids in subsequent frames should belong to the same human, leading to the wrong assignment of IDs.

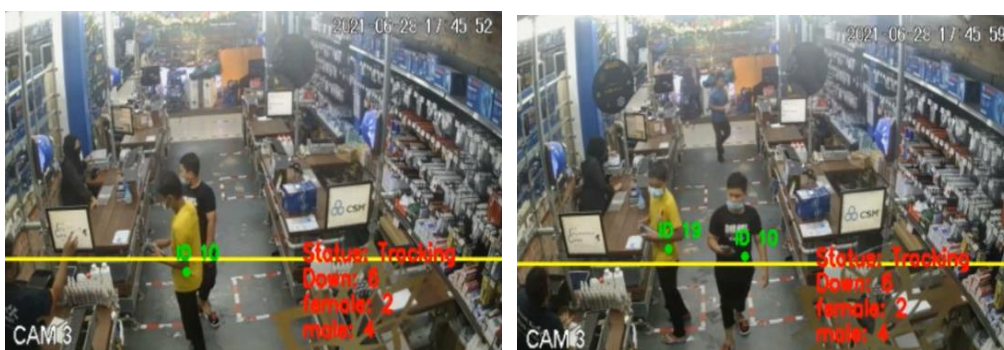


Figure 4.24 (a)

Figure 4.24 (b)

Figure 4.24: Failure of human tracking.

The face detection algorithm struggles to detect human faces in challenging situations, such as when faces are occluded by masks, when there are pose variations, and when faces are not looking directly at the camera. Section 4.2.3.1 highlighted that most of the algorithm's failures in detecting faces occurred when the person was wearing a mask and looking downwards at an angle greater than 45° . When a face is obscured by a mask, the algorithm finds it challenging to determine if a face is present since most of the facial features are covered. Additionally, if a person looks downwards, important features of the face may also be obstructed. However, this issue may be caused by the location of the CCTV, which is typically installed at a high angle, resulting in the algorithm always capturing the customers' face image from above.

The face recognition algorithm's weakness lies in its difficulty in accurately identifying humans based on only one face image embedded into 128 measurements in the customer database. The algorithm may have difficulty precisely differentiating between faces since one set of 128 measurements for each human does not allow for much tolerance. As a result, the algorithm may make mistakes in the face recognition step. However, this

weakness is difficult to overcome in reality as humans move quickly in and out of retail shops.

The pre-trained gender classification network used in the proposed solution was based on the Adience benchmark, which includes various challenging face images from around the world. However, the gender classification algorithm used in this solution was only based on Malaysians. As a result, the gender classification model may not perform well for Malaysian citizens, resulting in a low accuracy rate of 46.67 % in the exit video.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

The YOLOv8 series models outperform all other detection models in terms of their ability to detect humans and minimize false detection of cartoons as humans. In addition, the YOLOv8n model stands out for its fast processing speed, small parameters, and model size compared to other detection models discussed in this paper. Therefore, the retrained YOLOv8n model is considered to be the best-performing model overall, achieving 96.97 % accuracy, 99.52 % precision, 97.42 % recall, 98.46 % F1 score, a false detection rate of 8.16 %, and a model size of 6.09 MB. However, false detections of cartoons as humans cannot be completely eliminated with either pretrained or retrained models. The false detection rate can only be significantly reduced by retraining the model using a dataset consisting of two classes: humans and cartoons.

To perform customer analysis with machine vision, a combination of algorithms is utilized, including human detection, human tracking, centroid tracking, facial recognition, and gender classification. For human detection, the retrained YOLOv8n is used due to its high accuracy in detecting human, low false rate of detecting cartoons as humans and fast processing speed without GPU. The human tracking phase employs a scale estimation algorithm developed by Danelljan, et al. (2014) to track targets through scaling and translation. Centroid tracking assigns an ID to each human target to ensure accurate tracking. To count customers, a counting line is drawn in the video frame, with the algorithm recording an accuracy of 100 % and 93.75 % for the entrance and exit videos, respectively. The face detector used is the HOG model in the face recognition module in Python, which detects human faces with a success rate of 63.63 % and 33.33 % for the entrance and exit videos respectively. For facial recognition, the algorithm utilizes FaceNet which developed by Levi and Hassner (2015), extracts 128 measurements for facial features, with an accuracy of 57.14 % and 80 % for the entrance and exit

videos respectively. Overall, facial recognition records an accuracy of 36.36 % and 26.67 % for the entrance and exit videos, respectively. Lastly, gender classification uses a simple CNN architecture method by Levi and Hassner (2015), with an accuracy of 100 % and 46.67 % for the entrance and exit videos respectively.

5.2 Recommendations for future work

To improve the human detection algorithm, it is recommended to experiment with various combinations of learning parameters, so that it can achieve the optimal results. The learning parameters that can tune include number of epochs, batch size of images, fed-in image size, learning rate, etc. Furthermore, ensure that each instance in the dataset is labelled well to improve the quality of the dataset. It is also recommended to maintain a 1:1 ratio of human-to-cartoon instances while keeping the ratio of images per class at 1:1.

To enhance the algorithm's capabilities, it is recommended to lower the position of the CCTV and adjust its angle to capture frames that face the customer's frontal face. This adjustment would help reduce issues with face detection when customers look downwards and increase the algorithm's ability to detect faces. Additionally, a better CCTV with a higher resolution would be preferred, as the algorithm crops the face image of the customer when they cross the counting line, and sometimes, the cropped image may be blurry due to movement. With a higher quality image, the algorithm could capture and extract more precise face features for the face recognition phase.

Another recommendation is to increase the number of face images with different postures and angles in the customer database for training purposes. In the case study, each person only uses a single frontal face image for training, so the algorithm doesn't adapt to recognize humans if there are any changes in their posture or facial angle.

REFERENCES

- Abhishree, T.M., Latha, J., Manikantan, K. and Ramachandran, S., 2015. Face Recognition using Gabor filter based Feature Extraction with Anisotropic Diffusion as a pre-processing technique. In: *Procedia Computer Science*. <https://doi.org/10.1016/j.procs.2015.03.149>.
- Aboah, A., Wang, B., Bagci, U. and Adu-Gyamfi, Y., 2023. Real-time Multi-Class Helmet Violation Detection Using Few-Shot Data Sampling Technique and YOLOv8. [online] Available at: <http://arxiv.org/abs/2304.08256>.
- Adjabi, I., Ouahabi, A., Benzaoui, A. and Taleb-Ahmed, A., 2020. *Past, present, and future of face recognition: A review*. *Electronics (Switzerland)*, <https://doi.org/10.3390/electronics9081188>.
- Ahmed, A., Guo, J., Ali, F., Deebea, F. and Ahmed, A., 2018. LBPH based improved face recognition at low resolution. In: *2018 International Conference on Artificial Intelligence and Big Data, ICAIBD 2018*. <https://doi.org/10.1109/ICAIBD.2018.8396183>.
- Ahonen, T., Hadid, A. and Pietikäinen, M., 2004. Face recognition with local binary patterns. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3021. https://doi.org/10.1007/978-3-540-24670-1_36
- Al-mohair, H.K., Mohamad-saleh, J. and Suandi, S.A., 2013. Impact of Color Space on Human Skin Color Detection Using an Intelligent System. *Recent Advances in Image, Audio and Signal Processin*.
- Ansari, M.A. and Singh, D.K., 2021. Human detection techniques for real time surveillance: a comprehensive survey. *Multimedia Tools and Applications*, 80(6). <https://doi.org/10.1007/s11042-020-10103-4>.
- Anwarul, S. and Dahiya, S., 2020. A comprehensive review on face recognition methods and factors affecting facial recognition accuracy. In: *Lecture Notes in Electrical Engineering*. https://doi.org/10.1007/978-3-030-29407-6_36.
- Bartlett, M.S., Movellan, J.R. and Sejnowski, T.J., 2002. Face recognition by independent component analysis. *IEEE Transactions on Neural Networks*, 13(6). <https://doi.org/10.1109/TNN.2002.804287>
- Bam, A., Choudhary, P. and Bhoir, J., 2021. Real-Time Human Detection and Tracking in Motion Environment. *International Research Journal of Engineering and Technology*. [online] Available at: www.irjet.net.
- Basha, A.F., Jahangeer, G.S.B., 2012. Face gender image classification using various wavelet transform and support vector machine with various kernels, *International Journal of Computer Science Issues*, Vol. 9, No. 6, pp.150–157.

Belhumeur, P.N., Hespanha, J.P. and Kriegman, D.J., 1997. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7). <https://doi.org/10.1109/34.598228>.

Benzaoui, A. and Boukrouche, A., 2013. 1DLBP and PCA for face recognition. In: *Proceedings of the 2013 11th International Symposium on Programming and Systems, ISPS 2013*. <https://doi.org/10.1109/ISPS.2013.6581486>

Bhuvaneshwari, K. and Rauf, H.A., 2009. Edgelet based human detection and tracking by combined segmentation and soft decision. In: *2009 International Conference on Control Automation, Communication and Energy Conservation, INCACEC 2009*.

Bledsoe, W.W., 1964. The Model Method In Facial Recognition. *Panoramic Research Inc., Palo Alto, CA, Rep. PRI*, 15(47).

Bochkovskiy, A., Wang, C.-Y. and Liao, H.-Y.M., 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. [online] Available at: <http://arxiv.org/abs/2004.10934>.

Bolme, D.S., Beveridge, J.R., Draper, B.A. and Lui, Y.M., 2010. Visual object tracking using adaptive correlation filters. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2010.5539960>.

Bratu, S. and Sabau, R.I., 2022. *Digital Commerce in the Immersive Metaverse Environment: Cognitive Analytics Management, Real-Time Purchasing Data, and Seamless Connected Shopping Experiences*. [online] Addletonacademicpublishers.com. Available at: <<https://addletonacademicpublishers.com/contents-lpi/2445-volume-21-2022/4229-digital-commerce-in-the-immersive-metaverse-environment-cognitive-analytics-management-real-time-purchasing-data-andseamless-connected-shopping-experiences>> [Accessed 8 August 2022].

Brunelli, R. and Poggio, T., 1993. Face Recognition: Features versus Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10). <https://doi.org/10.1109/34.254061>.

Cao, L., Dikmen, M., Fu, Y. and Huang, T.S., 2008. Gender recognition from body. In: *MM'08 - Proceedings of the 2008 ACM International Conference on Multimedia, with co-located Symposium and Workshops*. <https://doi.org/10.1145/1459359.1459470>.

Chakraborty, B., Rius, I., Pedersoli, M., Mozerov, M. and González, J., 2007. *Component-based Human Detection*. [online] Digital.csic.es.

Chiachia, G., Marana, A.N., Ruf, T. and Ernst, A., 2011. Census histograms: A simple feature extraction and matching approach for face recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(8). <https://doi.org/10.1142/S0218001411009068>.

Chuanqi305, 2018. MobileNet-SSD. [source code]. Available at <https://github.com/chuanqi305/MobileNet-SSD>.

Cuțitoi, A.-C., 2022. Remote Patient Monitoring Systems, Wearable Internet of Medical Things Sensor Devices, and Deep Learning-based Computer Vision Algorithms in COVID-19 Screening, Detection, Diagnosis, and Treatment, *American Journal of Medical Research* 9(1): 129–144. doi: 10.22381/ajmr9120229

Danelljan, M., Häger, G., Khan, F.S. and Felsberg, M., 2014. Accurate scale estimation for robust visual tracking. In: *BMVC 2014 - Proceedings of the British Machine Vision Conference 2014*. <https://doi.org/10.5244/c.28.65>.

Dalal, N. and Triggs, B., 2005. Histograms of oriented gradients for human detection. In: *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*. <https://doi.org/10.1109/CVPR.2005.177>.

Dalal, N., Triggs, B. and Schmid, C., 2006. Human detection using oriented histograms of flow and appearance. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/11744047_33.

Davis James W. and Sharma, V. and T.A. and K.M., 2009. Human Detection and Tracking. In: A. Li Stan Z. and Jain, ed. *Encyclopedia of Biometrics*. [online] Boston, MA: Springer US. pp.708–712. https://doi.org/10.1007/978-0-387-73003-5_35.

Dwyer, B., Nelson, J. (2022), Solawetz, J., et. al. Roboflow (Version 1.0) [Software]. Available from <https://roboflow.com>. [computer vision](https://roboflow.com).

Eklund, H., n.d. OBJECT DETECTION: MODEL COMPARISON ON AUTOMATED DOCUMENT CONTENT INTERPRETATION A performance evaluation of common object detection models.

Fellous, J.M., 1997. Gender discrimination and prediction on the basis of facial metric information. *Vision Research*, 37(14). [https://doi.org/10.1016/S0042-6989\(97\)00010-2](https://doi.org/10.1016/S0042-6989(97)00010-2).

Gao, C., Zhai, Y. and Guo, X., 2021. Visual Object Detection and Tracking System Design based on MobileNet-SSD. In: *2021 7th International Conference on Computer and Communications, ICC 2021*. <https://doi.org/10.1109/ICCC54389.2021.9674450>.

Geitgey, A., 2016. *Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning*. [online] Medium. Available at: <<https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78>> [Accessed 31 August 2022].

Girshick, R., 2015. Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 1440–1448.

Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2014.81>.

Golomb, B.A., Lawrence, D.T. and Sejnowski, T.J., 1991. Sexnet: A neural network identifies sex from human faces. *Advances in Neural Information Processing Systems 3*, (July).

Gonealves, V.P.M., Silva, L.P., Nunes, F.L.S., Ferreira, J.E. and Araujo, L. V., 2021. Evaluation of Traditional and Deep Learning Human Detection Techniques Applied to Surveillance: A Performance Comparison at Distinct Object Sizes. In: *Proceedings of 2021 IEEE International Conference on Signal Processing, Communications and Computing, ICSPCC 2021*. Institute of Electrical and Electronics Engineers Inc. <https://doi.org/10.1109/ICSPCC52875.2021.9564442>.

Han, H. and Tong, M., 2013. Human detection based on optical flow and spare geometric flow. In: *Proceedings - 2013 7th International Conference on Image and Graphics, ICIG 2013*. <https://doi.org/10.1109/ICIG.2013.96>.

He, K., Gkioxari, G., Dollár, P. and Girshick, R., 2020. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2). <https://doi.org/10.1109/TPAMI.2018.2844175>.

Henriques, J.F., Caseiro, R., Martins, P. and Batista, J., 2012. Exploiting the circulant structure of tracking-by-detection with kernels. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-642-33765-9_50.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. and Adam, H., 2017. MobileNets. *arXiv preprint arXiv:1704.04861*.

Hsu, F.C., Gubbi, J. and Palaniswami, M., 2013. Human head detection using Histograms of Oriented optical flow in low quality videos with occlusion. In: *2013, 7th International Conference on Signal Processing and Communication Systems, ICSPCS 2013 - Proceedings*. <https://doi.org/10.1109/ICSPCS.2013.6723967>.

Jabid, T., Kabir, M.H. and Chae, O., 2010. Gender classification using local directional pattern (LDP). In: *Proceedings - International Conference on Pattern Recognition*. <https://doi.org/10.1109/ICPR.2010.373>.

Jayaraman, U., Gupta, P., Gupta, S., Arora, G. and Tiwari, K., 2020. Recent development in face recognition. *Neurocomputing*. <https://doi.org/10.1016/j.neucom.2019.08.110>.

Kim, K.I., Jung, K. and Kim, H.J., 2002. Face recognition using kernel principal component analysis. *IEEE Signal Processing Letters*, 9(2). <https://doi.org/10.1109/97.991133>.

King, D.E., 2009. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10.

Labudzki, R., Legutko, S. and Raos, P., 2014. The essence and applications of machine vision. *Tehnicki Vjesnik*, 21(4).

Lechen, C., 2022. IS-Net: Improved Ship Detector Based on RetinaNet. In: *Proceedings - 2022 2nd International Signal Processing, Communications and Engineering Management Conference, ISPCEM 2022*. Institute of Electrical and Electronics Engineers Inc. pp.367–372. <https://doi.org/10.1109/ISPCEM57418.2022.00079>.

Levi, G. and Hassner, T., 2015. Age and gender classification using convolutional neural networks. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. <https://doi.org/10.1109/CVPRW.2015.7301352>.

Li, Z. and Yang, Y.H., 1999. Theoretical analysis of illumination in PCA-based vision systems. *Pattern Recognition*, 32(4). [https://doi.org/10.1016/s0031-3203\(98\)00119-8](https://doi.org/10.1016/s0031-3203(98)00119-8).

Lin, T.Y., Goyal, P., Girshick, R., He, K. and Dollar, P., 2017. Focal Loss for Dense Object Detection. In: *Proceedings of the IEEE International Conference on Computer Vision*. <https://doi.org/10.1109/ICCV.2017.324>.

Lian, H.C. and Lu, B.L., 2007. Multi-view gender classification using multi-resolution local binary patterns and support vector machines. *International Journal of Neural Systems*, 17(6). <https://doi.org/10.1142/S0129065707001317>.

Liao, S., Jain, A.K. and Li, S.Z., 2016. A Fast and Accurate Unconstrained Face Detector. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2). <https://doi.org/10.1109/TPAMI.2015.2448075>.

Linder, T., Wehner, S. and Arras, K.O., 2015. Real-time full-body human gender recognition in (RGB)-D data. In: *Proceedings - IEEE International Conference on Robotics and Automation*. <https://doi.org/10.1109/ICRA.2015.7139616>.

Ling, H., Wu, J., Huang, J., Chen, J. and Li, P., 2020. Attention-based convolutional neural network for deep face recognition. *Multimedia Tools and Applications*, 79(9–10). <https://doi.org/10.1007/s11042-019-08422-2>.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y. and Berg, A.C., 2016. SSD: Single shot multibox detector. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-46448-0_2.

luxiangju, 2021. iCartoonFace. Available at < <https://github.com/luxiangju-PersonAI/iCartoonFace>> .

Maheswari, S. and Korah, R., 2017. Enhanced skin tone detection using heuristic thresholding. *Biomedical Research (India)*, 28(9).

Moghaddam, B. and Yang, M.H., 2002. Learning gender with support faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5). <https://doi.org/10.1109/34.1000244>.

Mora Hernandez, D.A., Nalbach, O. and Werth, D., 2019. How computer vision provides physical retail with a better view on customers. In: *Proceedings - 21st IEEE Conference on Business Informatics, CBI 2019*. <https://doi.org/10.1109/CBI.2019.00060>.

Ng, C.B., Tay, Y.H. and Goi, B.M., 2015. A review of facial gender recognition. *Pattern Analysis and Applications*, 18(4). <https://doi.org/10.1007/s10044-015-0499-6>.

Ojala, T., Pietikäinen, M. and Mäenpää, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7). <https://doi.org/10.1109/TPAMI.2002.1017623>

Palmatier, R.W. and Martin, K.D., 2019. Understanding and Valuing Customer Data. In: *The Intelligent Marketer's Guide to Data Privacy*. https://doi.org/10.1007/978-3-030-03724-6_7.

Piccardi, M., 2004. Background subtraction techniques: A review. In: *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*. <https://doi.org/10.1109/ICSMC.2004.1400815>.

Rai, P. and Khanna, P., 2012. Gender classification techniques: A review. In: *Advances in Intelligent and Soft Computing*. https://doi.org/10.1007/978-3-642-30157-5_6.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2016.9>.

- Redmon, J. and Farhadi, A., 2017. YOLO9000: Better, faster, stronger. In: *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*. <https://doi.org/10.1109/CVPR.2017.690>.
- Redmon, J. and Farhadi, A., 2018. Yolov3: an incremental improvement. arXiv:180402767
- Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In: *Advances in Neural Information Processing Systems*.
- Turk, M. and Pentland, A., 1991. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1). <https://doi.org/10.1162/jocn.1991.3.1.71>.
- Rosebrock, A., 2018. *Simple object tracking with OpenCV - PyImageSearch*. [online] PyImageSearch. Available at: <<https://pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/>> [Accessed 30 August 2022].
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L.C., 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2018.00474>.
- Schroff, F., Kalenichenko, D. and Philbin, J., 2015. FaceNet: A unified embedding for face recognition and clustering. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2015.7298682>.
- Shan, C., 2012. Learning local binary patterns for gender classification on real-world face images. *Pattern Recognition Letters*, 33(4). <https://doi.org/10.1016/j.patrec.2011.05.016>.
- Taigman, Y., Yang, M., Ranzato, M. and Wolf, L., 2014. DeepFace: Closing the gap to human-level performance in face verification. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2014.220>.
- Tian, Z., Shen, C., Chen, H. and He, T., 2019. FCOS: Fully convolutional one-stage object detection. In: *Proceedings of the IEEE International Conference on Computer Vision*. <https://doi.org/10.1109/ICCV.2019.00972>.
- Ultralytics, 2020. yolov5. [source code]. Available at <https://github.com/ultralytics/yolov5>.
- Ultralytics, 2023. Ultralytics YOLOv8. [source code]. Available at <https://github.com/ultralytics/ultralytics>.
- Viola, P. and Jones, M.J., 2004. Robust Real-Time Face Detection. *International Journal of Computer Vision*, 57(2). <https://doi.org/10.1023/B:VISI.0000013087.49260.fb>.

Wang, J.G., Li, J., Yau, W.Y. and Sung, E., 2010. Boosting dense SIFT descriptors and shape contexts of face images for gender recognition. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, CVPRW 2010*. <https://doi.org/10.1109/CVPRW.2010.5543238>.

Wang, M., Jiang, H. and Li, Y., 2010. Face recognition based on DWT/DCT and SVM. In: *ICCAASM 2010 - 2010 International Conference on Computer Application and System Modeling, Proceedings*. <https://doi.org/10.1109/ICCAASM.2010.5620666>.

Weiliu89, 2018. caffe. [source code]. Available at <https://github.com/weiliu89/caffe/tree/ssd>.

Wiskott, L., Fellous, J.-M., Krüger, N. and von der Malsburg, C., 1995. Face Recognition and Gender Determination. *Faces*.

Wiskott, L., Fellous, J.M., Krüger, N. and von Malsburg, C. der, 1997. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7). <https://doi.org/10.1109/34.598235>.

Won, A.S., Yu, L., Janssen, J.H. and Bailenson, J., 2012. Tracking gesture to detect gender. *Proceedings of the International Society for Presence Research Annual Conference*.

Wong, W.K., Hui, J.H., Loo, C.K. and Lim, W.S., 2011. Off-time swimming pool surveillance using thermal imaging system. In: *2011 IEEE International Conference on Signal and Image Processing Applications, ICSIPA 2011*. <https://doi.org/10.1109/ICSIPA.2011.6144091>.

Wu, B. and Nevatia, R., 2007. Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors. *International Journal of Computer Vision*, 75(2). <https://doi.org/10.1007/s11263-006-0027-7>.

Wu, J., Geyer, C. and Rehg, J.M., 2011. Real-time human detection using contour cues. In: *Proceedings - IEEE International Conference on Robotics and Automation*. <https://doi.org/10.1109/ICRA.2011.5980437>.

Wu, X., He, R., Sun, Z. and Tan, T., 2018. A light CNN for deep face representation with noisy labels. *IEEE Transactions on Information Forensics and Security*, 13(11). <https://doi.org/10.1109/TIFS.2018.2833032>.

Wulfert, L., Wiede, C., Verbunt, M.H., Gembaczka, P. and Grabmaier, A., 2022. Human Detection with A Feedforward Neural Network for Small Microcontrollers. In: *2022 7th International Conference on Frontiers of Signal Processing, ICFSP 2022*. Institute of Electrical and Electronics Engineers Inc. pp.14–22. <https://doi.org/10.1109/ICFSP55781.2022.9924667>.

Xu, R., Lin, H., Lu, K., Cao, L. and Liu, Y., 2021. A forest fire detection system based on ensemble learning. *Forests*, 12(2), pp.1–17. <https://doi.org/10.3390/f12020217>.

Xu, R., Lin, H., Lu, K., Cao, L. and Liu, Y., 2021. A forest fire detection system based on ensemble learning. *Forests*, 12(2), pp.1–17. <https://doi.org/10.3390/f12020217>.

Xu, Z., Lu, L. and Shi, P., 2008. A hybrid approach to gender classification from face images. In: *Proceedings - International Conference on Pattern Recognition*. <https://doi.org/10.1109/icpr.2008.4761883>.

Yang, J., Zhang, D., Frangi, A.F. and Yang, J.Y., 2004. Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1). <https://doi.org/10.1109/TPAMI.2004.1261097>

Yuan, Y., Pang, Y. and Li, X., 2010. Footwear for gender recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(1). <https://doi.org/10.1109/TCSVT.2009.2022795>.

Zhang, H. and Hong, X., 2019. Recent progresses on object detection: a brief review. *Multimedia Tools and Applications*, 78(19). <https://doi.org/10.1007/s11042-019-07> .