Annexure-1

HUMAN FACE DETECTION

A Project Work

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND

ENGINEERING WITH

SPECIALIZATION IN AI AND ML

Submitted by:

HRITVIK MATHUR

20BCS6760

AMISHA KHANNA

20BCS6712

Under the Supervision of:

MR. Vijay Bharadwaj

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
APEX INSTITUE OF TECHNOLOGY

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413,

PUNJAB

NOVEMBER, 2022

Annexure-2

DECLARATION

We, **‘Hritvik Mathur’** and **‘Amisha Khanna’**, students of **‘Bachelor of Engineering in CSE with specialization in AIML’**, **session: 2020-2024**, Department of Computer Science and Engineering, Apex Institute of Technology, Chandigarh University, Punjab, hereby declare that the work presented in this Project Work entitled ‘**HUMAN FACE DETECTION’** is the outcome of our own bona fide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Hritvik Mathur
Candidate UID: 20BCS6760
Amisha Khanna
Candidate UID: 20BCS6712

Date: 10.11.2022

Place: Chandigarh University,
Gharuan, Punjab

ABSTRACT

With the phenomenal increase in image and video database there is an need of automatic understanding and examination of information by the intelligent systems as manually it is getting to be plainly distant. Our Face plays a major role in social intercourse for conveying feelings of a person and their identity. Human beings have not immense ability to identify different faces than machines. So, automatic face detection system plays an important role in face recognition, facial expression recognition, head-pose estimation, humancomputer interaction etc. FaceDetection is a technology that determines the location and size of a human face in a image. Face detection has been a standout amongst topics in the computer vision literature. This paper presents a comprehensive survey of various techniques explored for face detection in digital images. Different challenges and applications of face detection are also presented in this paper. In the end, different standard databases for face detection are also given with their features. Furthermore, we organize special discussions on the practical aspects towards the development of a robust face detection system and conclude this paper with several promising directions for future research.

ACKNOWLEDGEMENT

We are overwhelmed in all humbleness and gratefulness to acknowledge our depth to all those who have helped us to put these ideas, well above the level of simplicity and into something concrete.

We would like to express our special thanks of gratitude to our teacher, “Vijay Bharadwaj” who gave us the golden opportunity to do this wonderful project on the topic “HUMAN FACE DETECTION” which also helped us in doing a lot of Research and we came to know about so many new things. We are really thankful to them.
Any attempt at any level can 't be satisfactorily completed without the support and guidance of our parents and friends.

Thanking you,

Hritvik Mathur (20BCS6760)

Amisha Khanna (20BCS6712)

Annexure-3

Table of Contents

1 INTRODUCTION

With the rapid increase of computational powers and accessibility of innovative sensing, analysis and rendering equipment and technologies, computers are becoming more and more intelligent. Many research projects and commercial products have demonstrated the capability of a computer to interact with humans in a natural way by looking at people through cameras, listening to people through microphones, understanding these inputs, and reacting to people in a friendly manner. One of the fundamental techniques that enable such natural Human–Computer Interaction (HCI) is face detection. Face detection is the step stone to all facial analysis algorithms, including the face alignment, face modelling, face relighting, face recognition, face verification/authentication, head pose tracking, facial expression tracking/recognition, gender/age recognition, and many more. Face recognition is a very challenging research area in computer vision and pattern recognition due to variations in facial expressions, poses and illumination. Several emerging applications, from law enforcement to commercial tasks, demand the industry to develop efficient and automated face recognition systems. Although, many researchers have worked on the problem of face recognition for many years still several challenges need to be solved. Difference in illumination of the scene, changes in pose, orientation and expression are examples of some of the issues to be dealt carefully.

Within the last numerous years, numerous algorithms have been proposed for face detection. While lots of development has been made in the direction of spotting faces below small versions in lighting fixtures, facial expression and pose, dependable techniques for popularity beneath greater excessive variations have been established elusive. Face detection is essential to many face applications, together with face popularity and facial expression evaluation. But, the huge visible versions of faces, together with occlusions, massive pose variations, and excessive lighting, impose splendid demanding situations for those obligations in actual-world applications.

Cutting-edge deep gaining knowledge of structures is proved to be nearly perfect face detectors, which outperform human abilities in this location. The range of packages in these days' existence increases tremendously due to this reality. They would replace humans in regions wherein their accuracy is the most useful, as an instance, security

The cascade face detector proposed by using viola and jones makes use of haar-like features and AdaBoost to teach cascaded classifiers, which obtain correct performance with real-time performance. But, quite a few works [1, 3, 4] suggest that this detector may degrade substantially in real-global applications with large visual versions of human faces despite extra superior features and classifiers.

Recently, convolutional neural networks (CNN) achieved incredible progress in a selection of computer vision tasks, inclusive of picture class and face popularity. use cascaded CNNs for face detection, however, it requires bounding box calibration from face detection with a more computational price and ignores the inherent correlation among facial landmarks localization and bounding field regression. train deep convolutional neural networks for the facial characteristic reputation to reap excessive response in face areas which further yield candidate windows of faces

Fig. 1 A sample of faces

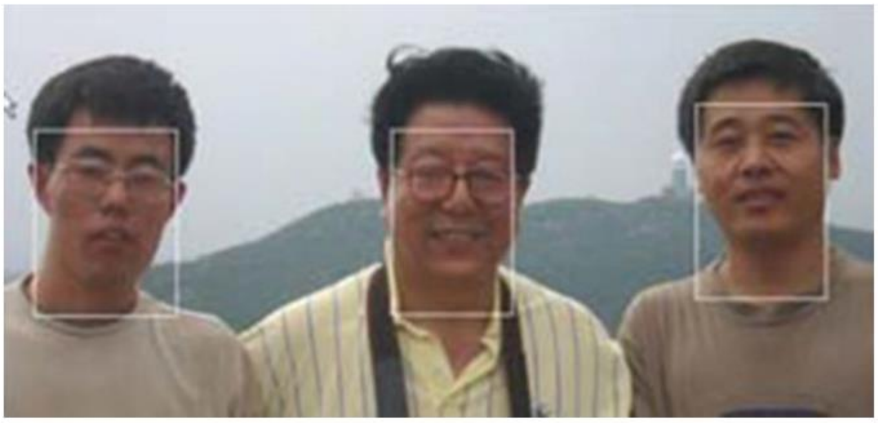


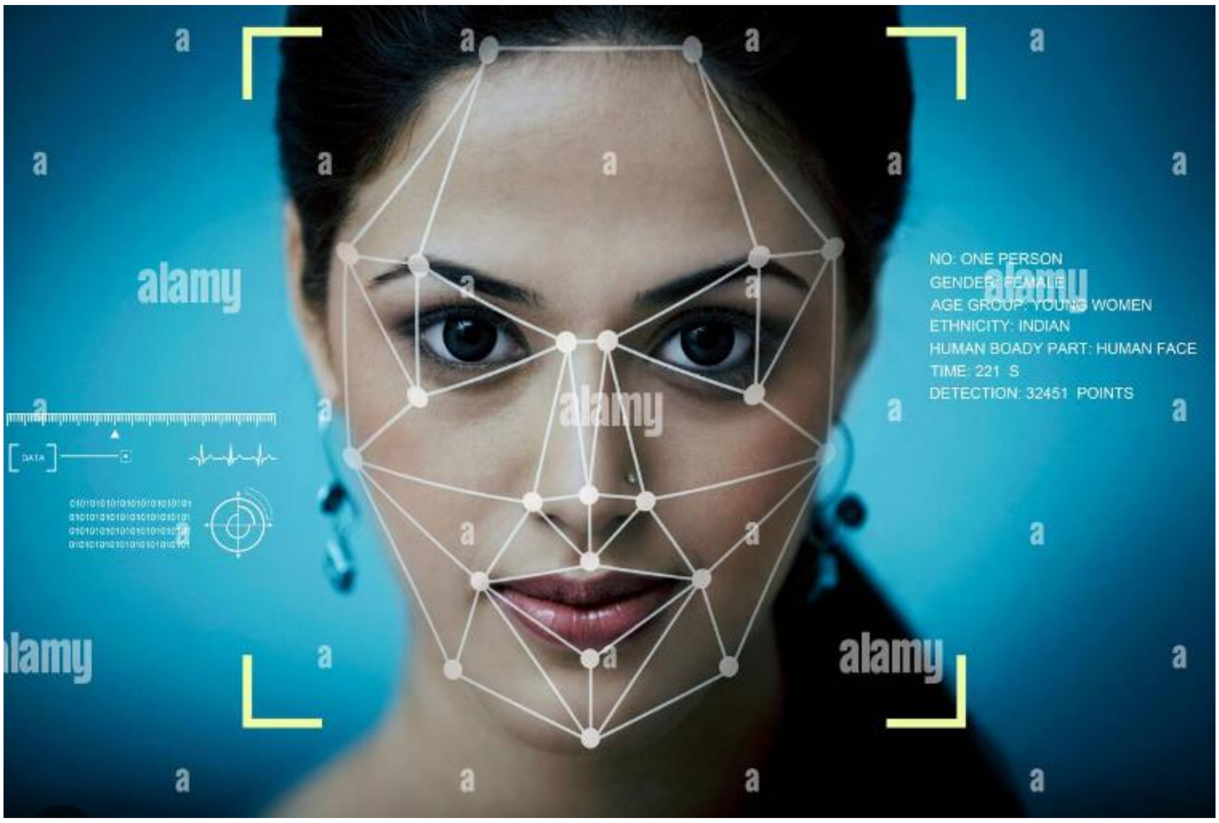
Fig. 2 Detected faces

A mature face detecting system commonly consists of image acquisition, photo pre-processing, face detection, face tracking, face alignment, function extraction, and evaluation. Among the extra critical steps are face detection, tracking, and face characteristic extraction. In recent years, face reputation structures were extensively used in channel bayonet structures which include clever access management and identity verification in high-pace railway stations. These channel bayonet face recognition structures have all or maximum of the face picture series, face detection, face alignment, face high-quality detection, face function extraction, face tracking, and different steps. However, some of those structures require an excessive degree of cooperation from people, a few are complex to put in force, and a few have high necessities for hardware along with computing gadgets.

1.1 PROBLEM DEFINITION

During software development, clones can occur in software intentionally or unintentionally. Our goal is to provide the users a wonderful experience. To keep pace with this fast-moving world, we need to be adaptable to changing circumstances. Face recognition is one of the most important aspects. And being able to develop these features is essential to growing a business. The computational models, which are implemented in this project, are chosen after extensive research, and the successful testing results confirm that the choices made by the researcher are reliable. The system with manual face detection and automatic face recognition did not have recognition accuracy over 90%, due to the limited number of eigen faces that were used for the PCA transform. This system is tested under very robust conditions in this experimental study and it is envisaged that real-world performance will be far more accurate. The fully automated frontal view face detection system displayed virtually perfect accuracy and in the researcher's opinion further work need not be conducted in this area. The fully automated face detection and recognition system was not robust enough to achieve a high recognition accuracy. The only reason for this was the face recognition subsystem did not display even a slight degree of invariance to scale, rotation or shift errors of the segmented face image.

1.2 PROJECT OVERVIEW/SPECIFICATIONS



INTRODUCTION

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as (computer)system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With an increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

HARDWARE REQUIREMENTS

Hardware environment includes a consumer-level PC with an Intel Core i5, M560 2.6GHz CPU, 4G RAM.

2 LITERATURE REVIEW

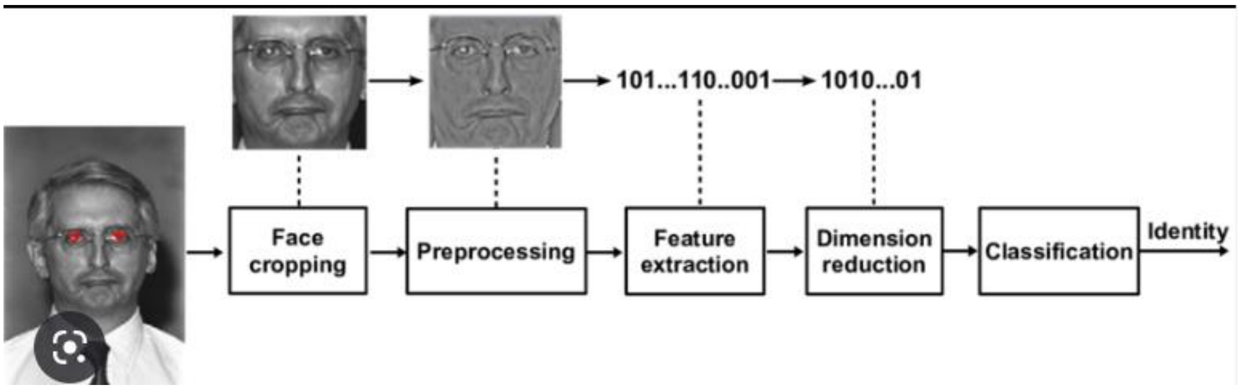
Face detection is a computer technology that determines the location and size of human face in arbitrary image. The facial features are detected and any other objects like trees, buildings and bodies etc. are ignored from the image. It can be regarded as a _specific‘case of object-class detection, where the task is finding the location and sizes of all objects in an image that belong to a given class. Face detection, can be regarded as a more general‘case of face localization. In face localization, the task is to find the locations and sizes of a known number of faces (usually one). Basically, there are two types of approaches to detect facial part in the given image i.e. feature base and image base approach. Feature base approach tries to extract features of the image and match it against the knowledge of the face features. While image base approach tries to get best match between training and testing images.

2.1 FEATURE BASE APPROCH:

Active Shape Model Active shape models focus on complex non-rigid features like actual physical and higher-level appearance of features Means that Active Shape Models (ASMs) are aimed at automatically locating landmark points that define the shape of any statistically modelled object in an image. When of facial features such as the eyes, lips, nose, mouth and eyebrows. The training stage of an ASM involves the building of a statistical

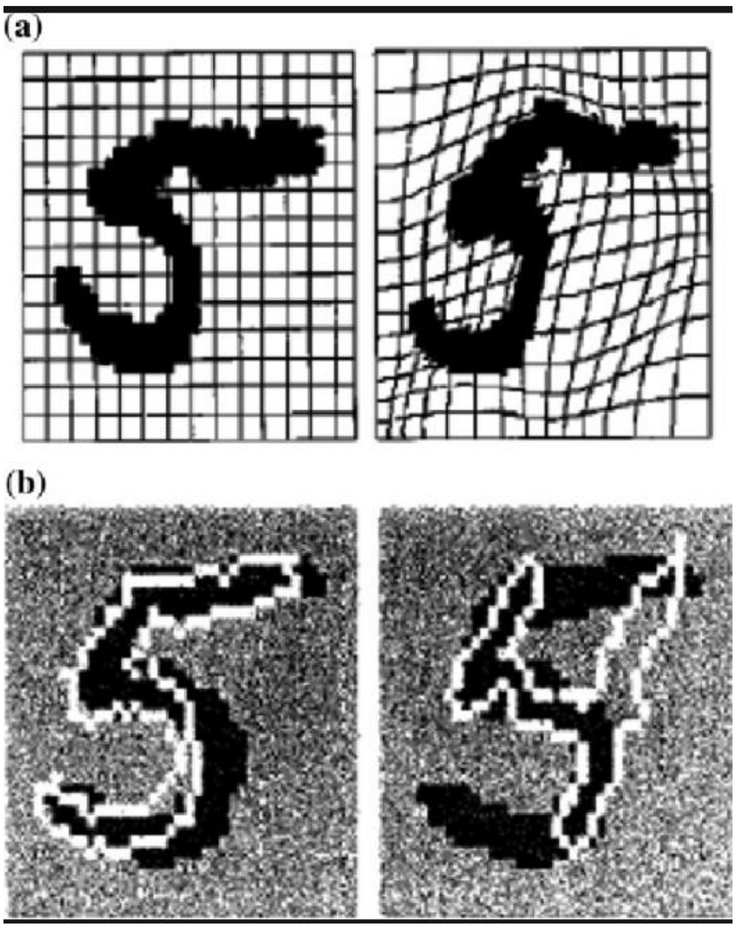
a) facial model from a training set containing images with manually annotated landmarks. ASMs is classified into three groups i.e., snakes, PDM, Deformable templates

b) 1.1)Snakes: The first type uses a generic active contour called snakes, first introduced by Kass et al. in 1987 Snakes are used to identify head boundaries [8,9,10,11,12]. In order to achieve the task, a snake is first initialized at the proximity around a head boundary. It then locks onto nearby edges and subsequently assume the shape of the head. The evolution of a snake is achieved by minimizing an energy function, Esnake (analogy with physical systems), denoted asEsnake = Einternal + EExternal WhereEinternal and EExternal are internal and external energy functions.Internal energy is the part that depends on the intrinsic properties of the snake and defines its natural evolution. The typical natural evolution in snakes is shrinking or expanding. The external energy counteracts the internal energy and enables the contours to deviate from the natural evolution and eventually assume the shape of nearby features—the head boundary at a state of equilibria.Two main consideration for forming snakes i.e. selection of energy terms and energy minimization. Elastic energy is used commonly as internal energy. Internal energy is vary with the distance between control points on the snake, through which we get contour an elastic-band characteristic that causes it to shrink or expand. On other side external energy relay on image features. Energy minimization process is done by optimization techniques such as the steepest gradient descent. Which needs highest computations. Huang and Chen and Lam and Yan both employ fast iteration methods by greedy algorithms. Snakes have some demerits like contour often becomes trapped onto false image features and another one is that snakes are not suitable in extracting non convex features.



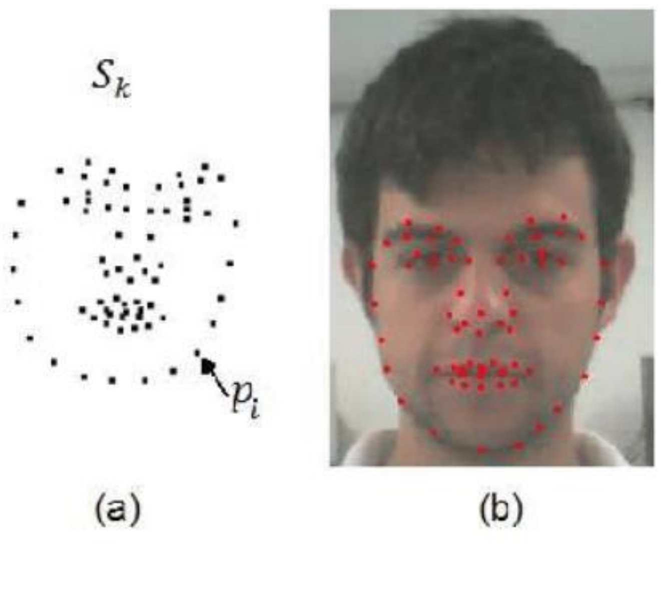
2.1.1 Deformable Templates:

Deformable templates were then introduced by Yuille et al. to take into account the a priori of facial features and to better the performance of snakes. Locating a facial feature boundary is not an easy task because the local evidence of facial edges is difficult to organize into a sensible global entity using generic contours. The low brightness contrast around some of these features also makes the edge detection process. Yuille et al. took the concept of snakes a step further by incorporating global information of the eye to improve the reliability of the extraction process. Deformable templates approaches are developed to solve this problem. Deformation is based on local valley, edge, peak, and brightness. Other than face boundary, salient feature (eyes, nose, mouth and eyebrows) extraction is a great challenge of face recognition. $E = E_v + E_e + E_p + E_i + E_{internal}$; where E_v , E_e , E_p , E_i , $E_{internal}$ are external energy due to valley, edges, peak and image brightness and internal energy



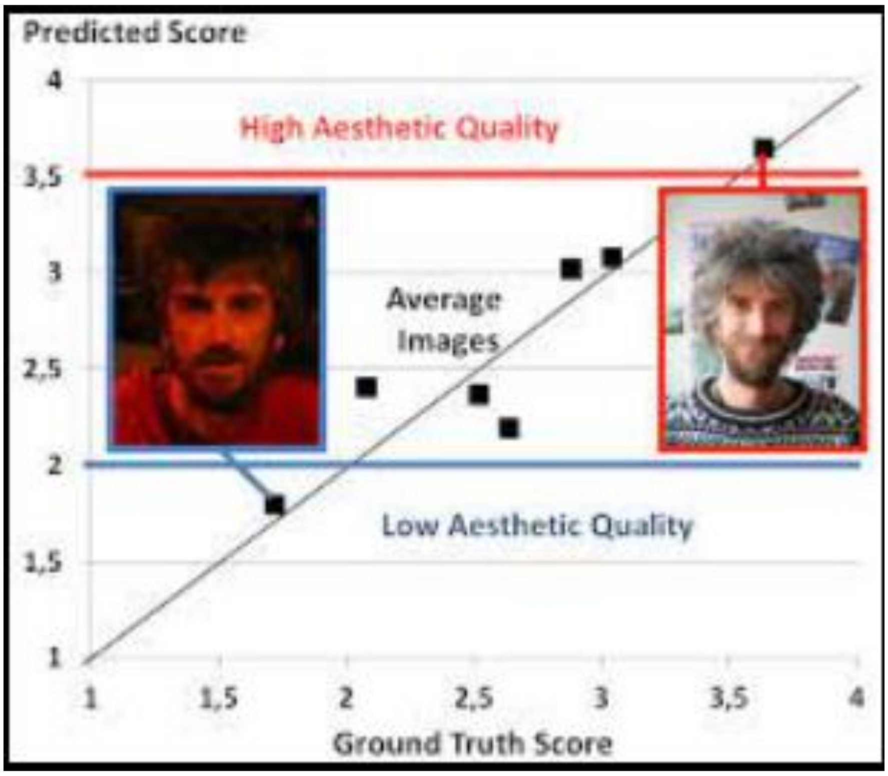
2.1.2 PDM (Point Distribution Model):

Independently of computerized image analysis, and before ASMs were developed, researchers developed statistical models of shape . The idea is that once you represent shapes as vectors, you can apply standard statistical methods to them just like any other multivariate object. These models learn allowable constellations of shape points from training examples and use principal components to build what is called a Point Distribution Model. These have been used in diverse ways, for example for categorizing Iron Age broaches. Ideal Point Distribution Models can only deform in ways that are characteristic of the object. Coots and his colleagues were seeking models which do exactly that so if a beard, say, covers the chin, the shape model can \override the image" to approximate the position of the chin under the beard. It was therefore natural (but perhaps only in retrospect) to adopt Point Distribution Models. This synthesis of ideas from image processing and statistical shape modelling led to the Active Shape Model.The first parametric statistical shape model for image analysis based on principal components of inter-landmark distances was presented by Cootes and Taylor in. On this approach, Cootes, Taylor, and their colleagues, then released a series of papers that cumulated in what we call the classical Active Shape Model.



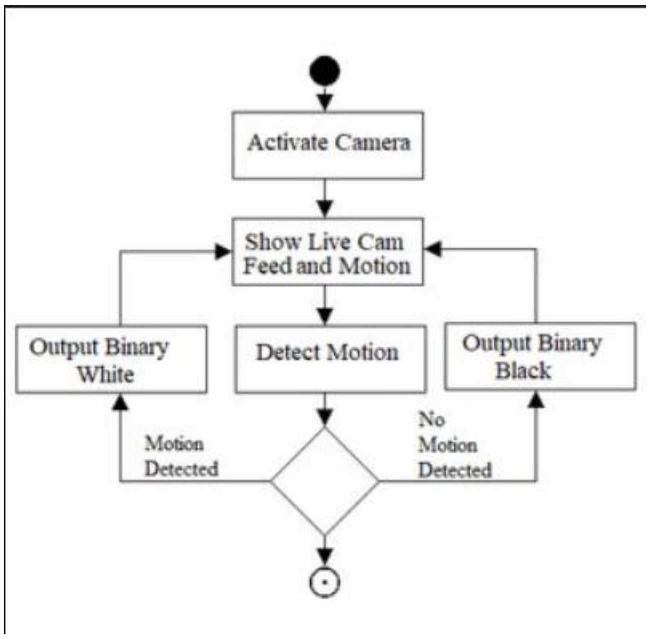
2.2) LOW LEVEL ANALYSIS:

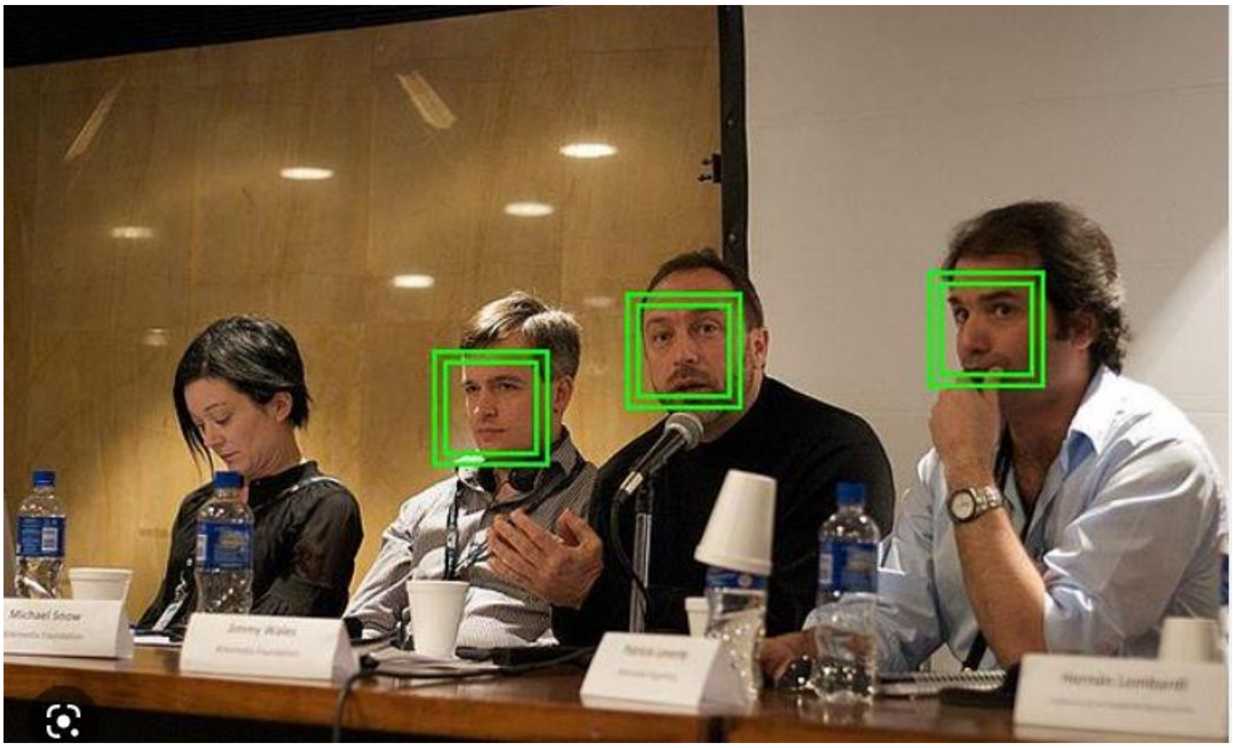
Based on low level visual features like colour, intensity, edges, motion etc. Skin Colour .Base Colour is avital feature of human faces. Using skin-color as a feature for tracking a face has several advantages. Colour processing is much faster than processing other facial features. Under certain lighting conditions, colour is orientation invariant. This property makes motion estimation much easier because only a translation model is needed for motion estimation. Tracking human faces using colour as a feature has several problems like the colour representation of a face obtained by a camera is influenced by many factors (ambient light, object movement, etc.



2.3) MOTION BASE:

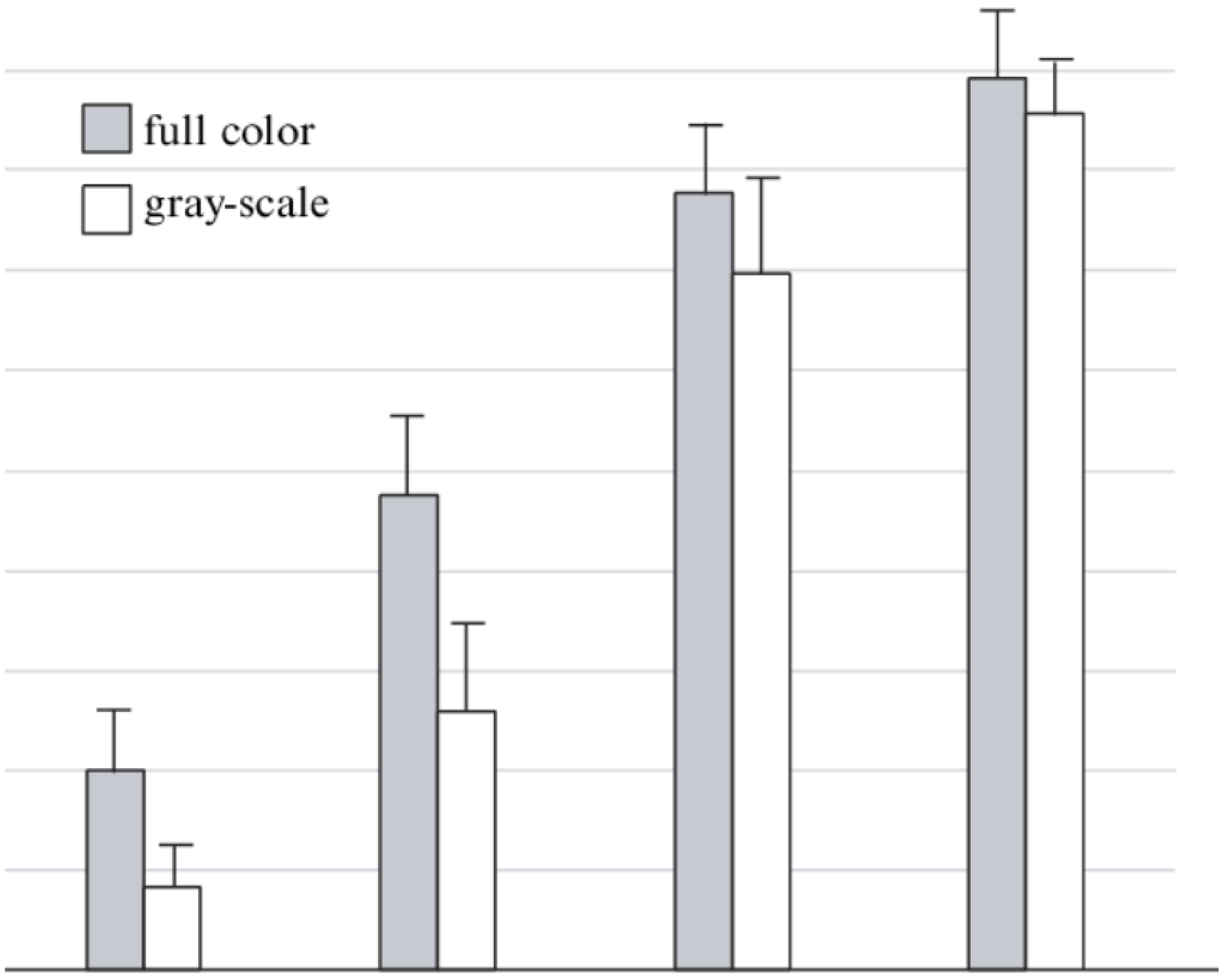
When use of video sequence is available, motion information can be used to locate moving objects. Moving silhouettes like face and body parts can be extracted by simply thresholding accumulated frame differences. Besides face regions, facial feature scan be located by frame differences.





2.3.1 Gray Scale Base:

Gray information within a face can also be treat as important features. Facial features such as eyebrows, pupils, and lips appear generally darker than their surrounding facial regions. Various recent feature extraction algorithms search for local gray minima within segmented facial regions. In these algorithms, the input images are first enhanced by contrast-stretching and gray-scale morphological routines to improve the quality of local dark patches and thereby make detection easier. The extraction of dark patches is achieved by low-level gray-scale thresholding. Based method and consist three levels. Yang and huang presented new approach i.e. faces gray scale behaviour in pyramid (mosaic) images. This system utilizes hierarchical Face location consist three levels. Higher two level based on mosaic images at different resolution. In the lower level, edge detection method is proposed. Moreover this algorithms gives fine response in complex background where size of the face is unknown



2.3.2 Edge Base:

Face detection based on edges was introduced by Sakai et al. This work was based on analysing line drawings of the faces from photographs, aiming to locate facial features. Than later Craw et al. proposed a hierarchical framework based on Sakai et al.‘swork to trace a human head outline. Then after remarkable works were carried out by many researchers in this specific area. Method suggested by Anila and Devarajan was very simple and fast. They proposed frame work which consist three stepsi.e. initially the images are enhanced by applying median filter for noise removal and histogram equalization for contrast adjustment. In the second step the edge image is constructed from the enhanced image by applying sobel operator. Then a novel edge tracking algorithm is applied to extract the sub windows from the enhanced image based on edges. Further they used Back propagation Neural Network (BPN) algorithm to classify the sub-window as either face or non-face.

2.4 FEATURE ANALYSIS

These algorithms aim to find structural features that exist even when the pose, viewpoint, or lighting conditions vary, and then use these to locate faces. These methods are designed mainly for face localization 2.4.1 Feature Searching Viola Jones Method: Paul Viola and Michael Jones presented an approach for object detection which minimizes computation time while achieving high detection accuracy. Paul Viola and Michael Jones [39] proposed a fast and robust method for face detection which is 15 times quicker than any technique at the time of release with 95% accuracy at around 17 fps. The technique relies on the use of simple Haar-like features that are evaluated quickly through the use of a new image representation. Based on the concept of an ―Integral Image‖ it generates a large set of features and uses the boosting algorithm AdaBoost to reduce the overcomplete set and the introduction of a degenerative tree of the boosted classifiers provides for robust and fast interferences. The detector is applied in a scanning fashion and used on gray-scale images, the scanned window that is applied can also be scaled, as well as the features evaluated. Department of ECE Page 10 Gabor Feature Method: Sharif et al proposed an Elastic Bunch Graph Map (EBGM) algorithm that successfully implements face detection using Gabor filters. The proposed system applies 40 different Gabor filters on an image. As are result of which 40 images with different angles and orientation are received. Next, maximum intensity points in each filtered image are calculated and mark them as fiducial points. The system reduces these points in accordance to distance between them. The next step is calculating the distances between the reduced points using distance formula. At last, the distances are compared with database. If match occurs, it means that the faces in the image are detected. Equation of Gabor filter is shown below

Face detection is a computer technology that determines the location and size of human face in arbitrary image. The facial features are detected and any other objects like trees, buildings and bodies etc. are ignored from the image. It can be regarded as a ‗specific‘case of object-class detection, where the task is finding the location and sizes of all objects in an image that belong to a given class.

Face detection, can be regarded as a more general‘case of face localization. In face localization, the task is to find the locations and sizes of a known number of faces (usually one). Basically, there are two types of approaches to detect facial part in the given image i.e. feature base and image base approach.

Feature base approach tries to extract features of the image and match it against the knowledge of the face features. While image base approach tries to get best match between training and testing images.

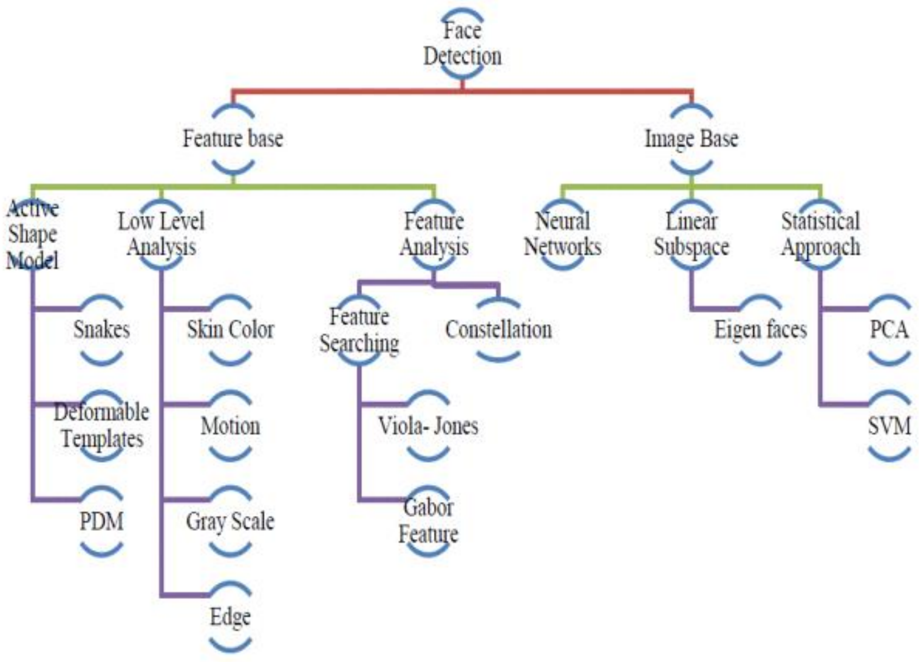


Fig 2.1 detection methods

2.1 FEATURE BASE APPROCH:

Active Shape Model Active shape models focus on complex non-rigid features like actual physical and higher level appearance of features Means that Active Shape Models (ASMs) are aimed at automatically locating landmark points that define the shape of any statistically modelled object in an image. When of facial features such as the eyes, lips, nose, mouth and eyebrows. The training stage of an ASM involves the building of a statistical

a) facial model from a training set containing images with manually annotated landmarks. ASMs is classified into three groups i.e. snakes, PDM, Deformable templates

b) 1.1)Snakes: The first type uses a generic active contour called snakes, first introduced by Kass et al. in 1987 Snakes are used to identify head boundaries [8,9,10,11,12]. In order to achieve the task, a snake is first initialized at the proximity around a head boundary. It then locks onto nearby edges and subsequently assume the shape of the head.

The evolution of a snake is achieved by minimizing an energy function, Esnake (analogy with physical systems), denoted asEsnake = Einternal + EExternal WhereEinternal and EExternal are internal and external energy functions.Internal energy is the part that depends on the intrinsic properties of the snake and defines its natural evolution. The typical natural evolution in snakes is shrinking or expanding. The external energy counteracts the internal energy and enables the contours to deviate from the natural evolution and eventually assume the shape of nearby features—the head boundary at a state of equilibria.Two main consideration for forming snakes i.e. selection of energy terms and energy minimization. Elastic energy is used commonly as internal energy. Internal energy is vary with the distance between control points on the snake, through which we get contour an elastic-band characteristic that causes it to shrink or expand. On other side external energy relay on image features. Energy minimization process is done by optimization techniques such as the steepest gradient descent. Which needs highest computations. Huang and Chen and Lam and Yan both employ fast iteration methods by greedy algorithms. Snakes have some demerits like contour often becomes trapped onto false image features and another one is that snakes are not suitable in extracting non convex features.

2.1.1 Deformable Templates:

Deformable templates were then introduced by Yuille et al. to take into account the a priori of facial features and to better the performance of snakes. Locating a facial feature boundary is not an easy task because the local evidence of facial edges is difficult to organize into a sensible global entity using generic contours. The low brightness contrast around some of these features also makes the edge detection process. Yuille et al. took the concept of snakes a step further by incorporating global information of the eye to improve the reliability of the extraction process. Deformable templates approaches are developed to solve this problem. Deformation is based on local valley, edge, peak, and brightness. Other than face boundary, salient feature (eyes, nose, mouth and eyebrows) extraction is a great challenge of face recognition. E = Ev + Ee + Ep + Ei + Einternal ; where Ev , Ee , Ep , Ei , Einternal are external energy due to valley, edges, peak and image brightness and internal energy

2.1.2 PDM (Point Distribution Model):

Independently of computerized image analysis, and before ASMs were developed, researchersdeveloped statistical models of shape . The idea is that once you represent shapes asvectors, you can apply standard statistical methods to them just like any other multivariateobject. These models learn allowable constellations of shape points from training examplesand use principal components to build what is called a Point Distribution Model. These havebeen used in diverse ways, for example for categorizing Iron Age broaches.Ideal Point Distribution Models can only deform in ways that are characteristic of the object. Cootes and his colleagues were seeking models which do exactly that so if a beard, say, covers the chin, the shape model can \override the image" to approximate the position of the chin under the beard. It was therefore natural (but perhaps only in retrospect) to adopt Point Distribution Models. This synthesis of ideas from image processing and statistical shape modelling led to the Active Shape Model.The first parametric statistical shape model for image analysis based on principal components of inter-landmark distances was presented by Cootes and Taylor in. On this approach, Cootes, Taylor, and their colleagues, then released a series of papers that cumulated in what we call the classical Active Shape Model.

2.2) LOW LEVEL ANALYSIS:

Based on low level visual features like colour, intensity, edges, motion etc. Skin Colour .Base Colour is avital feature of human faces. Using skin-color as a feature for tracking a face has several advantages. Colour processing is much faster than processing other facial features. Under certain lighting conditions, colour is orientation invariant. This property makes motion estimation much easier because only a translation model is needed for motion estimation. Tracking human faces using colour as a feature has several problems like the colour representation of a face obtained by a camera is influenced by many factors (ambient light, object movement, etc.

2.3) MOTION BASE:

When use of video sequence is available, motion information can be used to locate moving objects. Moving silhouettes like face and body parts can be extracted by simply thresholding accumulated frame differences. Besides face regions, facial feature scan be located by frame differences.

2.3.1 Gray Scale Base:

Gray information within a face can also be treat as important features. Facial features such as eyebrows, pupils, and lips appear generally darker than their surrounding facial regions. Various recent feature extraction algorithms search for local gray minima within segmented facial regions. In these algorithms, the input images are first enhanced by contrast-stretching and gray-scale morphological routines to improve the quality of local dark patches and thereby make detection easier. The extraction of dark patches is achieved by low-level gray-scale thresholding. Based method and consist three levels. Yang and huang presented new approach i.e. faces gray scale behaviour in pyramid (mosaic) images. This system utilizes hierarchical Face location consist three levels. Higher two level based on mosaic images at different resolution. In the lower level, edge detection method is proposed. Moreover this algorithms gives fine response in complex background where size of the face is unknown

2.3.2 Edge Base:

Face detection based on edges was introduced by Sakai et al. This work was based on analysing line drawings of the faces from photographs, aiming to locate facial features. Than later Craw et al. proposed a hierarchical framework based on Sakai et al.‘swork to trace a human head outline. Then after remarkable works were carried out by many researchers in this specific area. Method suggested by Anila and Devarajan was very simple and fast. They proposed frame work which consist three stepsi.e. initially the images are enhanced by applying median filter for noise removal and histogram equalization for contrast adjustment. In the second step the edge image is constructed from the enhanced image by applying sobel operator. Then a novel edge tracking algorithm is applied to extract the sub windows from the enhanced image based on edges. Further they used Back propagation Neural Network (BPN) algorithm to classify the sub-window as either face or non-face.

2.4 FEATURE ANALYSIS

These algorithms aim to find structural features that exist even when the pose, viewpoint, or lighting conditions vary, and then use these to locate faces. These methods are designed mainly for face localization 2.4.1 Feature Searching Viola Jones Method: Paul Viola and Michael Jones presented an approach for object detection which minimizes computation time while achieving high detection accuracy. Paul Viola and Michael Jones [39] proposed a fast and robust method for face detection which is 15 times quicker than any technique at the time of release with 95% accuracy at around 17 fps. The technique relies on the use of simple Haar-like features that are evaluated quickly through the use of a new image representation. Based on the concept of an ―Integral Image‖ it generates a large set of features and uses the boosting algorithm AdaBoost to reduce the overcomplete set and the introduction of a degenerative tree of the boosted classifiers provides for robust and fast interferences. The detector is applied in a scanning fashion and used on gray-scale images, the scanned window that is applied can also be scaled, as well as the features evaluated. Department of ECE Page 10 Gabor Feature Method: Sharif et al proposed an Elastic Bunch Graph Map (EBGM) algorithm that successfully implements face detection using Gabor filters. The proposed system applies 40 different Gabor filters on an image. As are result of which 40 images with different angles and orientation are received. Next, maximum intensity points in each filtered image are calculated and mark them as fiducial points. The system reduces these points in accordance to distance between them. The next step is calculating the distances between the reduced points using distance formula. At last, the distances are compared with database. If match occurs, it means that the faces in the image are detected. Equation of Gabor filter is shown below

$$\psi_{u,v}(z) = \frac{\| k_{u,v} \|^2}{\sigma^2} e^{\left(\frac{\|k_{u,v}\|^2 \|z\|^2}{2\sigma^2}\right)} \left[e^{i\vec{k}_{u,v} z} - e^{-\frac{\sigma^2}{2}} \right]$$

Where

$\phi_u = \frac{u\pi}{8}, \quad \phi_u \in [0, \pi)$ gives the frequency,

What is Viola Jones algorithm?

Viola Jones algorithm is named after two computer vision researchers who proposed the method in 2001, Paul Viola and Michael Jones in their paper, “Rapid Object Detection using a Boosted Cascade of Simple Features”. Despite being an outdated framework, Viola-Jones is quite powerful, and its application has proven to be exceptionally notable in real-time face detection. This algorithm is painfully slow to train but can detect faces in real-time with impressive speed.

Given an image(this algorithm works on grayscale image), the algorithm looks at many smaller subregions and tries to find a face by looking for specific features in each subregion. It needs to check many different positions and scales because an image can contain many faces of various sizes. Viola and Jones used Haar-like features to detect faces in this algorithm.

The Viola Jones algorithm has four main steps, which we shall discuss in the sections to follow:

1. Selecting Haar-like features
2. Creating an integral image
3. Running AdaBoost training
4. Creating classifier cascades

What are Haar-Like Features?

In the 19th century a Hungarian mathematician, Alfred Haar gave the concepts of Haar wavelets, which are a sequence of rescaled “square-shaped” functions which together form a wavelet family or basis. Voila and Jones adapted the idea of using Haar wavelets and developed the so-called Haar-like features.

Haar-like features are digital image features used in object recognition. All human faces share some universal properties of the human face like the eyes region is darker than its neighbour pixels, and the nose region is brighter than the eye region.

A simple way to find out which region is lighter or darker is to sum up the pixel values of both regions and compare them. The sum of pixel values in the darker region will be smaller than the sum of pixels in the lighter region. If one side is lighter than the other, it may be an edge of an eyebrow or sometimes the middle portion may be shinier than the surrounding boxes, which can be interpreted as a nose This can be accomplished using Haar-like features and with the help of them, we can interpret the different parts of a face.

There are 3 types of Haar-like features that Viola and Jones identified in their research:

1. Edge features
2. Line-features
3. Four-sided features

Edge features and Line features are useful for detecting edges and lines respectively. The four-sided features are used for finding diagonal features.

The value of the feature is calculated as a single number: the sum of pixel values in the black area minus the sum of pixel values in the white area. The value is zero for a plain surface in which all the pixels have the same value, and thus, provide no useful information.

Since our faces are of complex shapes with darker and brighter spots, a Haar-like feature gives you a large number when the areas in the black and white rectangles are very different. Using this value, we get a piece of valid information out of the image.

To be useful, a Haar-like feature needs to give you a large number, meaning that the areas in the black and white rectangles are very different. There are known features that perform very well to detect human faces:

For example, when we apply this specific haar-like feature to the bridge of the nose, we get a good response. Similarly, we combine many of these features to understand if an image region contains a human face.

What are Integral Images?

In the previous section, we have seen that to calculate a value for each feature, we need to perform computations on all the pixels inside that particular feature. In reality, these calculations can be very intensive since the number of pixels would be much greater when we are dealing with a large feature.

The integral image plays its part in allowing us to perform these intensive calculations quickly so we can understand whether a feature of several features fit the criteria.

An integral image (also known as a summed-area table) is the name of both a data structure and an algorithm used to obtain this data structure. It is used as a quick and efficient way to calculate the sum of pixel values in an image or rectangular part of an image.

How is AdaBoost used in viola jones algorithm?

Next, we use a Machine Learning algorithm known as AdaBoost. But why do we even want an algorithm?

The number of features that are present in the 24×24 detector window is nearly 160,000, but only a few of these features are important to identify a face. So we use the AdaBoost algorithm to identify the best features in the 160,000 features.

In the Viola-Jones algorithm, each Haar-like feature represents a weak learner. To decide the type and size of a feature that goes into the final classifier, AdaBoost checks the performance of all classifiers that you supply to it.

To calculate the performance of a classifier, you evaluate it on all subregions of all the images used for training. Some subregions will produce a strong response in the classifier. Those will be classified as positives, meaning the classifier thinks it contains a human face. Subregions that don’t provide a strong response don’t contain a human face, in the classifiers opinion. They will be classified as negatives.

The classifiers that performed well are given higher importance or weight. The final result is a strong classifier, also called a boosted classifier, that contains the best performing weak classifiers.

So when we’re training the AdaBoost to identify important features, we’re feeding it information in the form of training data and subsequently training it to learn from the information to predict. So ultimately, the algorithm is setting a minimum threshold to determine whether something can be classified as a useful feature or not.

What are Cascading Classifiers?

Maybe the AdaBoost will finally select the best features around say 2500, but it is still a time-consuming process to calculate these features for each region. We have a 24×24 window which we slide over the input image, and we need to find if any of those regions contain the face. The job of the cascade is to quickly discard non-faces, and avoid wasting precious time and computations. Thus, achieving the speed necessary for real-time face detection.

We set up a cascaded system in which we divide the process of identifying a face into multiple stages. In the first stage, we have a classifier which is made up of our best features, in other words, in the first stage, the subregion passes through the best features such as the feature which identifies the nose bridge or the one that identifies the eyes. In the next stages, we have all the remaining features.

When an image subregion enters the cascade, it is evaluated by the first stage. If that stage evaluates the subregion as positive, meaning that it thinks it’s a face, the output of the stage is maybe.

When a subregion gets a maybe, it is sent to the next stage of the cascade and the process continues as such till we reach the last stage.

If all classifiers approve the image, it is finally classified as a human face and is presented to the user as a detection.

Now how does it help us to increase our speed? Basically, If the first stage gives a negative evaluation, then the image is immediately discarded as not containing a human face. If it passes the first stage but fails the second stage, it is discarded as well. Basically, the image can get discarded at any stage of the classifier

Using a Viola-Jones Classifier to detect faces in a live webcam feed

In this section, we are going to implement the Viola-Jones algorithm using OpenCV and detect faces in our webcam feed in real-time. We will also use the same algorithm to detect the eyes of a person too. This is quite simple and all you need is to install OpenCV and Python on your PC. You can refer to this article to know about OpenCV and how to install it

In OpenCV, we have several trained Haar Cascade models which are saved as XML files. Instead of creating and training the model from scratch, we use this file. We are going to use "haarcascade_frontalface_alt2.xml" file in this project. Now let us start coding.

The first step is to find the path to the "haarcascade_frontalface_alt2.xml" and "haarcascade_eye_tree_eyeglasses.xml" files. We do this by using the os module of Python language.

```
import os

cascPathface = os.path.dirname(

    cv2.__file__) + "/data/haarcascade_frontalface_alt2.xml"

cascPatheyes = os.path.dirname(

    cv2.__file__) + "/data/haarcascade_eye_tree_eyeglasses.xml"
```

The next step is to load our classifier. We are using two classifiers, one for detecting the face and others for detection eyes. The path to the above XML file goes as an argument to CascadeClassifier() method of OpenCV.

```
faceCascade = cv2.CascadeClassifier(cascPath)
```

```
eyeCascade = cv2.CascadeClassifier(cascPatheyes)
```

After loading the classifier, let us open the webcam using this simple OpenCV one-liner code

```
video_capture = cv2.VideoCapture(0)
```

Next, we need to get the frames from the webcam stream, we do this using the read() function. We use the infinite loop to get all the frames until the time we want to close the stream.

```
while True:

    # Capture frame-by-frame

    ret, frame = video_capture.read()
```

The read() function returns:

1. The actual video frame read (one frame on each loop)
2. A return code

The return code tells us if we have run out of frames, which will happen if we are reading from a file. This doesn’t matter when reading from the webcam since we can record forever, so we will ignore it.

For this specific classifier to work, we need to convert the frame into greyscale.

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

The faceCascade object has a method detectMultiScale(), which receives a frame(image) as an argument and runs the classifier cascade over the image. The term MultiScale indicates that the algorithm looks at subregions of the image in multiple scales, to detect faces of varying sizes.

```
faces = faceCascade.detectMultiScale(gray,
                                    scaleFactor=1.1,

                                    minNeighbors=5,

                                    minSize=(60, 60),

                                    flags=cv2.CASCADE_SCALE_IMAGE)
```

Let us go through these arguments of this function:

- scaleFactor – Parameter specifying how much the image size is reduced at each image scale. By rescaling the input image, you can resize a larger face to a smaller one, making it detectable by the algorithm. 1.05 is a good possible value for this, which means you use a small step for resizing, i.e. reduce the size by 5%, you increase the chance of a matching size with the model for detection is found.
- minNeighbors – Parameter specifying how many neighbours each candidate rectangle should have to retain it. This parameter will affect the quality of the detected faces. Higher value results in fewer detections but with higher quality. 3~6 is a good value for it.
- flags –Mode of operation
- minSize – Minimum possible object size. Objects smaller than that are ignored.

The variable faces now contain all the detections for the target image. Detections are saved as pixel coordinates. Each detection is defined by its top-left corner coordinates and width and height of the rectangle that encompasses the detected face.

To show the detected face, we will draw a rectangle over it.OpenCV’s rectangle() draws rectangles over images, and it needs to know the pixel coordinates of the top-left and bottom-right corner. The coordinates indicate the row and column of pixels in the image. We can easily get these coordinates from the variable face.

Also as now, we know the location of the face, we define a new area which just contains the face of a person and name it as faceROI.In faceROI we detect the eyes and encircle them using the circle function.

```
for (x,y,w,h) in faces:

        cv2.rectangle(frame, (x, y), (x + w, y + h),(0,255,0), 2)

        faceROI = frame[y:y+h,x:x+w]

        eyes = eyeCascade.detectMultiScale(faceROI)

        for (x2, y2, w2, h2) in eyes:

            eye_center = (x + x2 + w2 // 2, y + y2 + h2 // 2)

            radius = int(round((w2 + h2) * 0.25))

            frame = cv2.circle(frame, eye_center, radius, (255, 0, 0), 4)
```

The function rectangle() accepts the following arguments:

- The original image
- The coordinates of the top-left point of the detection
- The coordinates of the bottom-right point of the detection
- The colour of the rectangle (a tuple that defines the amount of red, green, and blue (0-255)).In our case, we set as green just keeping the green component as 255 and rest as zero.
- The thickness of the rectangle lines

Next, we just display the resulting frame and also set a way to exit this infinite loop and close the video feed. By pressing the ‘q’ key, we can exit the script here

```
cv2.imshow('Video', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

The next two lines are just to clean up and release the picture.

```
video_capture.release()

cv2.destroyAllWindows()
```

Here are the full code and output.

```
import cv2

import os

cascPathface = os.path.dirname(

    cv2.__file__) + "/data/haarcascade_frontalface_alt2.xml"

cascPatheyes = os.path.dirname(

    cv2.__file__) + "/data/haarcascade_eye_tree_eyeglasses.xml"



faceCascade = cv2.CascadeClassifier(cascPathface)

eyeCascade = cv2.CascadeClassifier(cascPatheyes)



video_capture = cv2.VideoCapture(0)

while True:

    # Capture frame-by-frame

    ret, frame = video_capture.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(gray,
                                         scaleFactor=1.1,
                                         minNeighbors=5,
                                         minSize=(60, 60),
                                         flags=cv2.CASCADE_SCALE_IMAGE)
    for (x,y,w,h) in faces:
        cv2.rectangle(frame, (x, y), (x + w, y + h),(0,255,0), 2)
        faceROI = frame[y:y+h,x:x+w]
        eyes = eyeCascade.detectMultiScale(faceROI)
        for (x2, y2, w2, h2) in eyes:
            eye_center = (x + x2 + w2 // 2, y + y2 + h2 // 2)
            radius = int(round((w2 + h2) * 0.25))
            frame = cv2.circle(frame, eye_center, radius, (255, 0, 0), 4)


        # Display the resulting frame
    cv2.imshow('Video', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
video_capture.release()
cv2.destroyAllWindows()
```

Output:

This brings us to the end of this article where we learned about the Viola Jones algorithm and its implementation in OpenCV.

Applications of face detection system

• Gender classification Gender information can be found from human being image. • Document control and access control Control can be imposed to document access with face identification system.

• Human computer interaction system It is design and use of computer technology, focusing particularly on the interfaces between users and computers.

• Biometric attendance It is system of taking attendance of people by their finger prints or face etc.

• Photography Some recent digital cameras use face detection for autofocus. Face detection is also useful for selecting regions of interest in photo slideshows.

• Facial feature extraction Facial features like nose, eyes, mouth, skin-color etc. can be extracted from image.

• Face recognition A facial recognition system is a process of identifying or verifying a person from a digital image or a video frame. One of the ways to do this is by comparing selected facial features from the image and a facial database. It is typically used in security systems.

• Marketing Face detection is gaining the interest of marketers. A webcam can be integrated into a television and detect any face that walks by. The system then calculates the race, gender, and age range of the face. Once the information is collected, a series of advertisements can be played that is specific towards the detected race/gender/age.

DIGITAL IMAGE PROCESSING

Interest in digital image processing methods stems from two principal application areas:

1. Improvement of pictorial information for human interpretation
2. Processing of scene data for autonomous machine perception In this second application area, interest focuses on procedures for extracting image information in a form suitable for computer processing. Examples includes automatic character recognition, industrial machine vision for product assembly and inspection, military recognizance, automatic processing of fingerprints etc. Image: Am image refers a 2D light intensity function f(x, y), where(x, y) denotes spatial coordinates and the value of f at any point (x, y) is proportional to the brightness or gray levels of the image at that point. A digital image is an image f(x, y) that has been discretized both in spatial coordinates and brightness. The elements of such a digital array are called image elements or pixels. A simple image model: To be suitable for computer processing, an image f(x, y) must be digitalized both spatially and in amplitude. Digitization of the spatial coordinates (x, y) is called image sampling. Amplitude digitization is called gray-level quantization. The storage and processing requirements increase rapidly with the spatial resolution and the number of gray levels. Example: A 256 gray-level image of size 256x256 occupies 64k bytes of memory.

Types of image processing

• Low level processing
• Medium level processing
• High level processing

Low level processing means performing basic qperations on images such as reading an image resize, resize, image rotate, RGB to gray level conversion, histogram equalization etc…, The output image obtained after low level processing is raw image. Medium level processing means extracting regions of interest from output of low level processed image. Medium level processing deals with identification of boundaries i.e edges .This process is called segmentation. High level processing deals with adding of artificial intelligence to medium level processed signal.

Viola-Jones algorithm:

There are different types of algorithms used in face detection. Here, we have used Viola-Jones algorithm
for face detection using MATLAB program. This algorithm works in following steps:

1. Creates a detector object using Viola-Jones algorithm
2. Takes the image from the video
3. Detects features
4. Annotates the detected feature

There are two predominant approaches to the face recognition problem: Geometric (feature based) and
photometric (view based). As researcher interest in face recognition continued, many different algorithms
were developed, three of which have been well studied in face recognition literature.

Recognition algorithms can be divided into two main approaches:

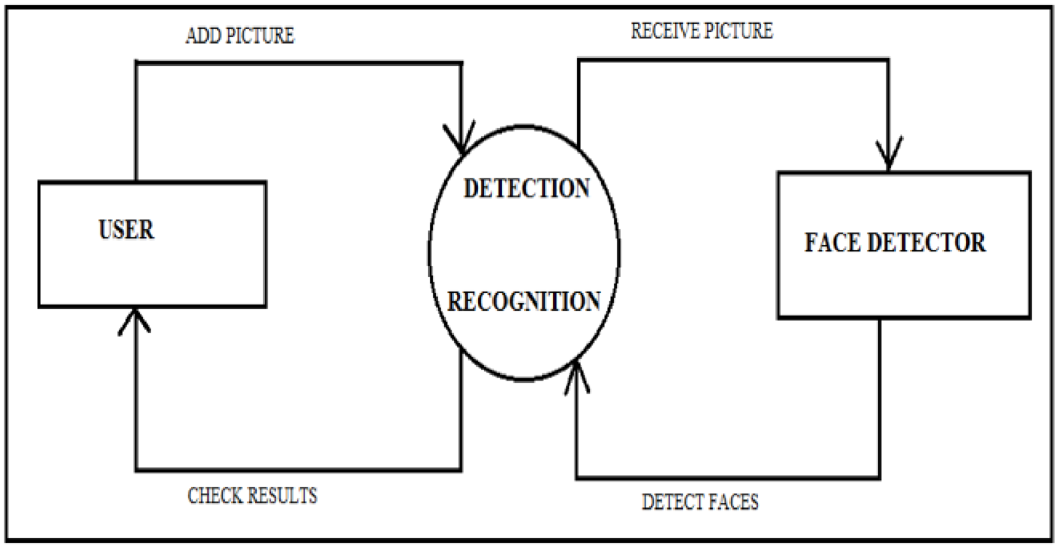
1. Geometric:

It is based on geometrical relationship between facial landmarks, or in other words the spatial
configuration of facial features. That means that the main geometrical features of the face such as the eyes,
nose and mouth are first located and then faces are classified on the basis of various

geometrical distances.

2. Photometric:

It is used to recover the shape of an object from a number of images taken under different lighting
conditions. The shape of the recovered object is defined by a gradient map, which is made up of an array
of surface.



3 PROBLEM FORMULATION

During software development, clones can occur in software intentionally or unintentionally. Developers tend to clone fragments of software during development to save efforts and expedite the development process.

From the literature review, it is observed that studies highlight the need of efficient and scalable approach for detecting code clones having software vulnerability. The existing techniques are not able to detect all types of vulnerable code clones. Different approaches suffer from high false negative rate and not scalable to large software systems due to high time complexity. So firstly, there is a need to make an efficient model

4 RESEARCH OBJECTIVES

The proposed work is aimed to carry out work leading to the development of an approach for better Human Face Detection System. The proposed aim will be achieved by dividing the work into following objectives:

a. Planned approach towards working: - The working in the organization will be well planned and organized. The data i.e. Image will be stored properly in database stores which will help in retrieval of information as well as its storage.

b. Accuracy: - The level of accuracy in the proposed system will be higher. All operation would be done correctly and it ensures that whatever information is coming from the center is accurate.

c. Reliability: - The reliability of the proposed system will be high due to the above stated reasons. The reason for the increased reliability of the system is that now there would be proper storage of information.

d. No Redundancy: - In the proposed system utmost care would be that no information is repeated anywhere, in storage or otherwise. This would assure economic use of storage space and consistency in the data stored.

e. Immediate retrieval of information: - The main objective of proposed system is to provide for a quick and efficient detection of required information. Any type of detection would be available whenever the user requires.

f. Immediate storage of information: - In manual system there are many problems to store the largest amount of information for processing.

g. Easy to Operate: - The system should be easy to operate and should be such that it can be developed within a short period of time and fit in the limited budget of the user.

5 METHODOLOGY

Every Biometric system has four main features which are shown in Figure. 1: face Detection, preprocessing, Feature Extraction, and Face Recognition.

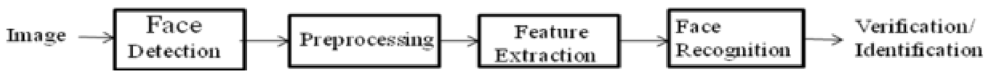


Figure 1. Architecture of Face Recognition System

Fundamental steps in image processing are

1. Image acquisition: to acquire a digital image
2. Image pre-processing: to improve the image in ways that increases the chances for success of the other processes.
3. Image segmentation: to partitions an input image into its constituent parts of objects.
4. Image segmentation: to convert the input data to a from suitable for computer processing.

5. Image description: to extract the features that result in some quantitative information of interest of features that are basic for differentiating one class of objects from another.

6. Image recognition: to assign a label to an object based on the information provided by its description.

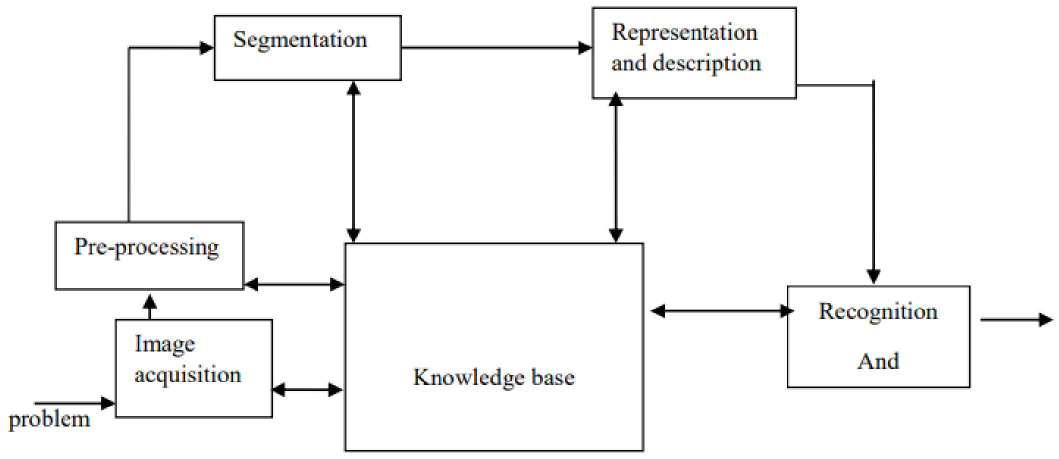


fig. Fundamental steps in digital image processing

2.1. Face Detection

The main function of this step is to detect the face from capture image or the selected image from the database. This face detection process actually verifies that weather the given image has face image or not, after detecting the face this output will be further given to the pre-processing step.

2.2. Pre-processing

This step is working as the pre-processing for face recognition, In this step the unwanted noise, blur, varying lightening condition, shadowing effects can be remove using pre-processing techniques. Once we have fine smooth face image then it will be used for the feature extraction process.

2.3. Feature Extraction

In this step features of face can be extracted using feature extraction algorithm. Extractions are performed to do information packing, dimension reduction, salience extraction, and noise cleaning. After this step, a face patch is usually transformed into a vector with fixed dimension or a set of fiducial points and their corresponding locations.

2.4. Face Recognition

Once feature extraction is done step analyzes the representation of each face; this last step is used to recognize the identities of the faces for achieving the automatic face recognition, for the recognition a face database is required to build. For each person, several images are taken and their features are extracted and stored in the database. Then when an input face image comes for recognition, then it first performs face detection, preprocessing and feature extraction, after that it compare its feature to each face class which stored in the database. There are two general applications of face recognition, one is called identification and another one is called verification. Face identification means given a face image, can be used to determine a person's identity even without his knowledge. While in face verification, given a face image and a guess of the identification, the system must to tell about the true or false about the guess. Face recognition can be largely classified into two different classes of approaches, the local feature-based method and the global feature-based method. The Human faces can be characterized both on the basis of local as well as of global features global features are easier to capture, they are generally less discriminative than localized features local features on the face can be highly discriminative, but may suffer for local changes in the facial appearance or partial face occlusion. Now a day’s face recognition system is recognize the face using multiple-views of faces, these Multi-view face recognition techniques has proposed by some authors for detecting each view of face such as left, right, front, top, and bottom etc.

Materials and Methods:

MATERIAL:

we have imported “haarcascade_frontalface_default.xml" Pre train model in our IDEL

The Haar-Cascade Face Detection Algorithm is a sliding-window type of algorithm that detects objects based upon its features.

Haar Face Features

The Haar-Cascade model, employs different types of feature recognition that include the likes of

- Size and location of certain facial features. To be specific, nose bridge, mouth line and eyes.
- Eye region being darker than upper-cheek region.
- Nose bridge region being brighter than eye region.

Intel's 'haarcascade_frontalface_default.xml'

This 'XML' file contains a pre-trained model that was created through extensive training and uploaded by Rainer Lienhart on behalf of Intel in 2000.

Rainer's model makes use of the Adaptive Boosting Algorithm (AdaBoost) in order to yield better results and accuracy.

METHOD:

We will use viola and jones’s algorithm’s haar-like features and AdaBoost to teach cascaded classifiers, which obtain correct performance with real-time performance.

Available facial recognition APIs :

• Kairos Offers a wide variety of image recognition solutions through their API. Their API endpoints include identifying gender, age, emotional depth, facial recognition in both photo and video, and more.

• Trueface.ai One flaw with some facial recognition APIs is that they are unable to differentiate between a face and a picture of a face. TrueFace.ai solves that problem with their ability to do spoof detection through their API.

• Amazon recognition This facial recognition API is fully integrated into the Amazon Web Service ecosystem. Using this API will make it really easy to build applications that make use of other AWS products.

• Face recognition and face detection by Lambda Labs With over 1000 calls per month in the free pricing tier, and only $0.0024 per extra API call, this API is a really affordable option for developers wanting to use a facial recognition API.

• EmoVu by Eyeris This API was created by Eyeris and it is a deep learning-based emotion recognition API. EmoVu allows for great emotion recognition results by identifying facial micro-expressions in real-time.

• Microsoft face API One cool feature that I found while doing research on the Microsoft Face API, is that the API has the ability to do “similar face search.” When this API endpoint is given a collection of faces, and a new face as a query, the API will return a collection of similar faces from the collection.

• Animetrics face recognition Using advanced 2D-to-3D algorithms, this API will convert a 2D image into a 3D model. The 3D model will then be used for facial recognition purpose.

• Face++ This API also has an offline SDK for iOS and Android for you to use. The offline SDK does not provide face recognition, but it can perform face detection, comparing, tracking and landmarks, all while the phone does not have cell service.

• Google cloud vision By being integrated into the Google Cloud Platform, this API will be a breeze for you to integrate into applications that are already using other Google Cloud Platform products and services.

• IBM Watson visual recognition Whether it is faces, objects, colors, or food, this API lets you identify many different types of classifiers. If the included classifiers aren’t enough, then you can train and use your own custom classifiers.

Convolutional Neural Network Cascade

Haoxiang Liy, Zhe Linz, Xiaohui Shenz, Jonathan Brandtz, Gang Hua [8] has proposed Convolutional Neural Network Cascade for Face Detection this method has built with very powerful discriminative capability, while maintaining high performance. The proposed CNN cascade operates at multiple resolutions, quickly rejects the background regions in the fast low-resolution stages, and carefully evaluates a small number of challenging candidates in the last high-resolution stage.

To improve localization effectiveness, and reduce the number of candidates at later stages, author has introduced a CNN-based calibration stage after each of the detection stages in the cascade. The motivation of applying the calibration is the most confident detection window may not be well aligned to the face. As the result has generated, without the calibration step, the next CNN in the cascade will have to evaluate more regions to maintain a good result. The overall detection has to increases the result at run time. This problem generally exists in object detection.

He has analysed this problem with CNNs in this work. Instead of training a CNN for bounding boxes regression as in R-CNN, he trained a multi-class classification CNN for calibration. He observed that a multi-class calibration CNN can be easily trained from limited amount of training data while a regression CNN for calibration requires more training data. He observed that the discretization decreases the difficulty of the calibration problem so that he can achieve good calibration accuracy with simpler CNN structures, after calibration the detection bounding box is better aligned to the real face centre. As the result has generated, the calibration nets enable more accurate face localization using coarser scanning windows across fewer scales.

The output of each calibration stage is used to adjust the detection window position for input to the subsequent stage. The proposed method runs at 14 FPS on a single CPU core for VGAresolution images and 100 FPS using a GPU, and achieves state-of-the-art detection performance on two public face detection benchmark

GOAL

Our goal is to provide the users a wonderful experience of studying and gathering knowledge. The computational models, which were implemented in this project, were chosen after extensive research, and the successful testing results confirm that the choices made by the researcher were reliable. The system with manual face detection and automatic face recognition did not have recognition accuracy over 90%, due to the limited number of eigen faces that were used for the PCA transform. This system was tested under very robust conditions in this experimental study and it is envisaged that real-world performance will be far more accurate. The fully automated frontal view face detection system displayed virtually perfect accuracy and in the researcher's opinion further work need not be conducted in this area. The fully automated face detection and recognition system was not robust enough to achieve a high recognition accuracy. The only reason for this was the face recognition subsystem did not display even a slight degree of invariance to scale, rotation or shift errors of the segmented face image. This was one of the system requirements identified in section However, if some sort of further processing, such as an eye detection technique, was implemented to further normalize the segmented face image, performance will increase to levels comparable to the manual face detection and recognition system. There are better techniques such as iris or retina recognition and face recognition using the thermal spectrum for user access and user verification applications since this need a very high degree of accuracy. The real-time automated pose invariant face detection and recognition system would be ideal for crowd surveillance applications. The implemented fully automated face detection and recognition system (with an eye detection system) could be used for simple surveillance applications such as ATM user security, while the implemented manual face detection and automated recognition system is ideal of mug shot matching., were we obtained in this study, which was conducted under adverse conditions. Implementing an eye detection technique would be a minor extension to the implemented system and would not require a great deal of additional research. All other implemented systems displayed commendable results and reflect well on the deformable template and Principal Component Analysis strategies.

I. Motivation

From Instagram filters to self-driving cars, Computer Vision technologies are now deeply integrated into the lifestyle of many people. One important Computer Vision application is the ability to have a computer detects objects in images. Among those objects, the human face receives the most attention since it has many useful applications in security and entertainments. Hence, this article focuses on a popular face detection framework called the Viola-Jones Object Detection Framework.

II. The Concepts

Developed by Paul Viola and Michael Jones back in 2001, the Viola-Jones Object Detection Framework can quickly and accurately detect objects in images and works particularly well with the human face (Viola & Jones, 2001). Despite its age, the framework is still a leading player in face detection along side many of its CNNs counter parts. The Viola-Jones Object Detection Framework combines the concepts of Haar-like Features, Integral Images, the AdaBoost Algorithm, and the Cascade Classifier to create a system for object detection that is fast and accurate. Thus to understand the framework, we first need to understand each of these concepts individually and then figure out how they connect together to form the framework.

Haar-like Features

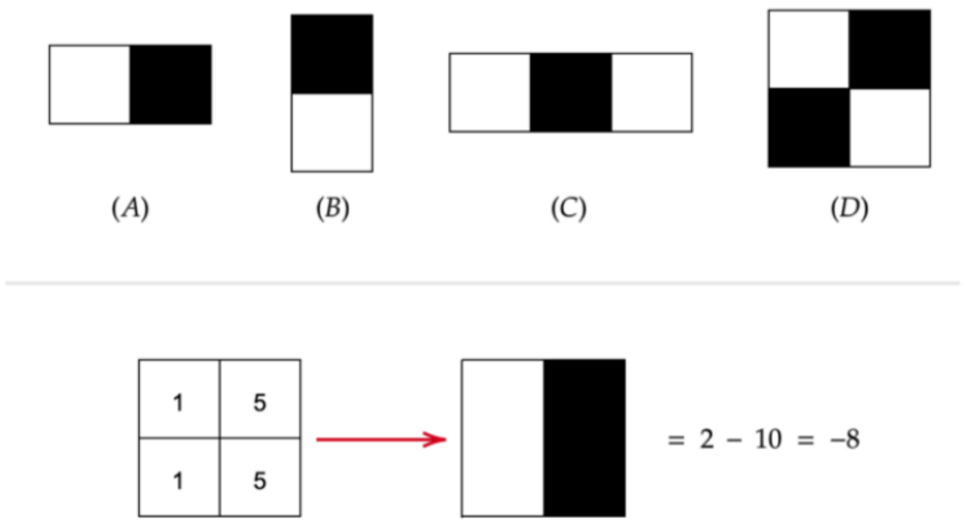


Figure 1: Haar-like features (top) and how to calculate them (bottom).

Often in Computer Vision, features are extracted from input images rather than using their intensities (RGB values, etc) directly. Haar-like features are one example. Other examples include Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), etc. A Haar-like feature consists of dark regions and light regions. It produces a single value by taking the sum of the intensities of the light regions and subtract that by the sum of the intensities of dark regions. There are many different types of Haar-like features but the Viola-Jones Object Detection Framework only uses the ones in Figure 1. The different types of Haar-like features let us extract useful information from an image such as edges, straight lines, and diagonal lines that we can use to identify an object (i.e. the human face)

The Cascade Classifier

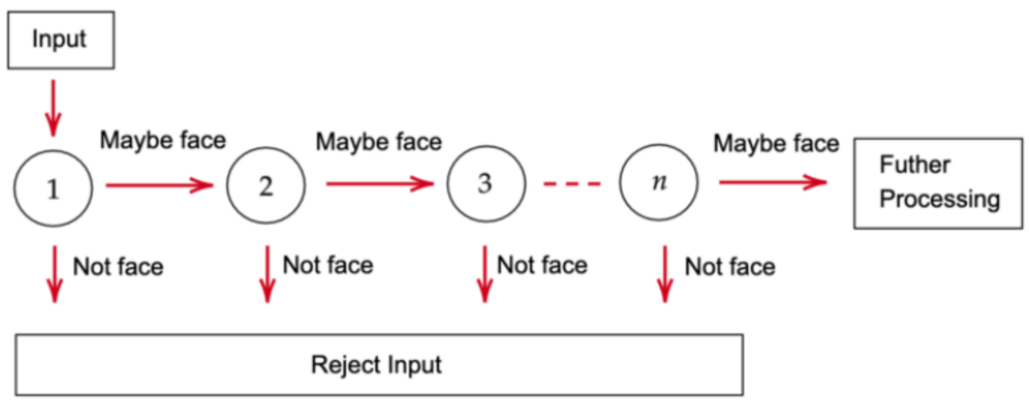


Figure 4: The Cascade Classifier

A Cascade Classifier is a multi-stage classifier that can perform detection quickly and accurately. Each stage consists of a strong classifier produced by the AdaBoost Algorithm. From one stage to another, the number of weak classifiers in a strong classifier increases. An input is evaluated on a sequential (stage by stage) basis. If a classifier for a specific stage outputs a negative result, the input is discarded immediately. In case the output is positive, the input is forwarded onto the next stage. According to Viola & Jones (2001), this multi-stage approach allows for the construction of simpler classifiers which can then be used to reject most negative (non face) input quickly while spending more time on positive (face) input.

III. Face Detection with the Viola-Jones Object Detection Framework

After learning about the major concepts used in the Viola-Jones Object Detection Framework, you are now ready to learn about how those concepts work together. The framework consists of two phases: Training and Testing/Application. Let’s look at each of them one by one.

Training

The goal of this phase is to produce a Cascade Classifier for a face that is able to accurately classify a face and discard non-faces quickly. To achieve that, you must first prepare your training data and then construct a Cascade Classifier by using a modified AdaBoost Algorithm on that training data.

1. **Data Preparation**

Concepts: Integral Image + Haar-like Features

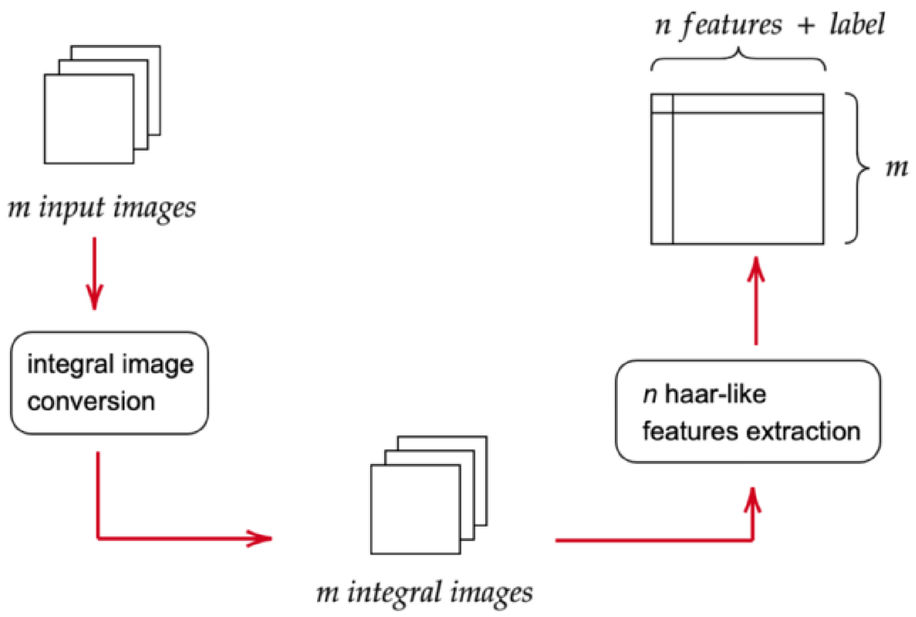
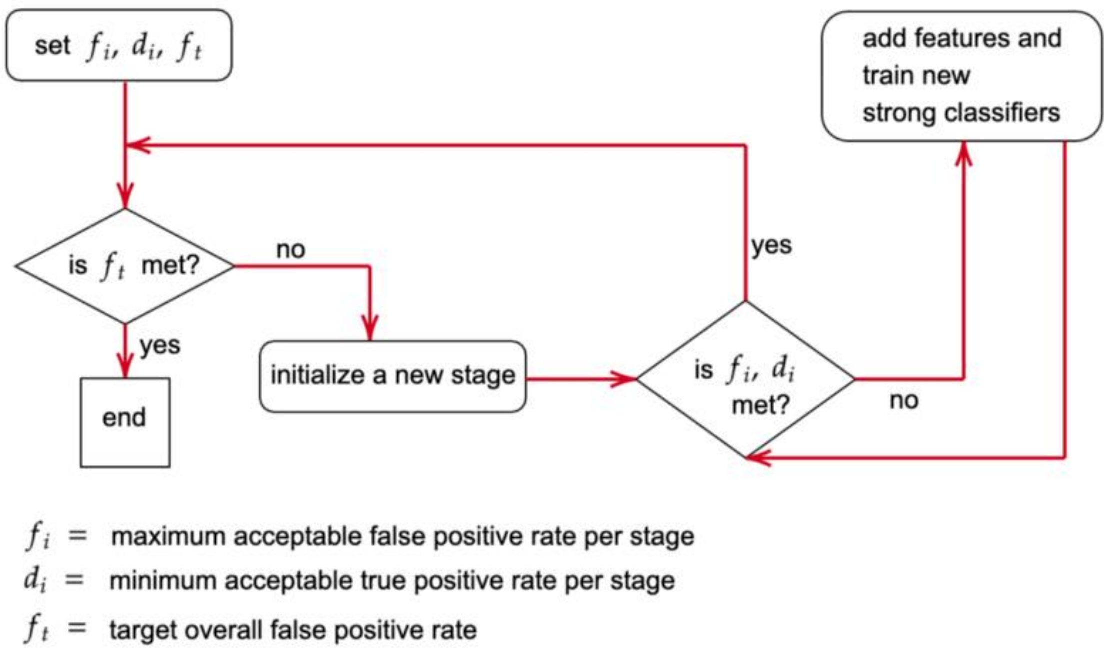


Figure 5: The Data Preparation Process

Assuming that you already have a training set consisting of positive samples (faces) and negative samples (non-faces), the first step is to extract features from those sample images. Viola & Jones (2001) recommends the images to be 24 x 24. Since each type of Haar-like features can have different sizes and positions in a 24 x 24 window, over 160,000 Haar-like features can be extracted. Nonetheless, in this stage all 160,000+ Haar-like features need to be calculated. Fortunately, the introduction of Integral Images helps speed up this process. Figure 5 illustrates the entire process of data preparation.

2. Constructing a Cascade Classifier with a modified AdaBoost Algorithm



set f_i, d_i, f_t
add features and train new strong classifiers
is f_t met?
no
yes
yes
end
initialize a new stage
is f_i, d_i met?
no
f_i = maximum acceptable false positive rate per stage
d_i = minimum acceptable true positive rate per stage
f_t = target overall false positive rate

Figure 6: The process to construct a cascade classifier

Concepts: the AdaBoost Algorithm + the Cascade Classifier.

As you can imagine, using all 160,000+ features directly is computationally inefficient. Viola & Jones (2001) proposed two solutions that can solve this. First, reduce the number of features to only a handful of useful features with the AdaBoost algorithm. Second, split the remaining features into stages and evaluate each input in a stage by stage (cascading) fashion. Viola & Jones (2001) devised a modified version of the AdaBoost algorithm to be able to train a Cascade Classifier. Figure 6 shows a simplified version of the algorithm provided by Ramsri in his video (Ramsri, 2012).

In their paper, Viola & Jones (2001) mentioned that their Cascade Classifier has 38 stages (38 strong classifiers) that are made up of over 6000 features.

Testing/Application

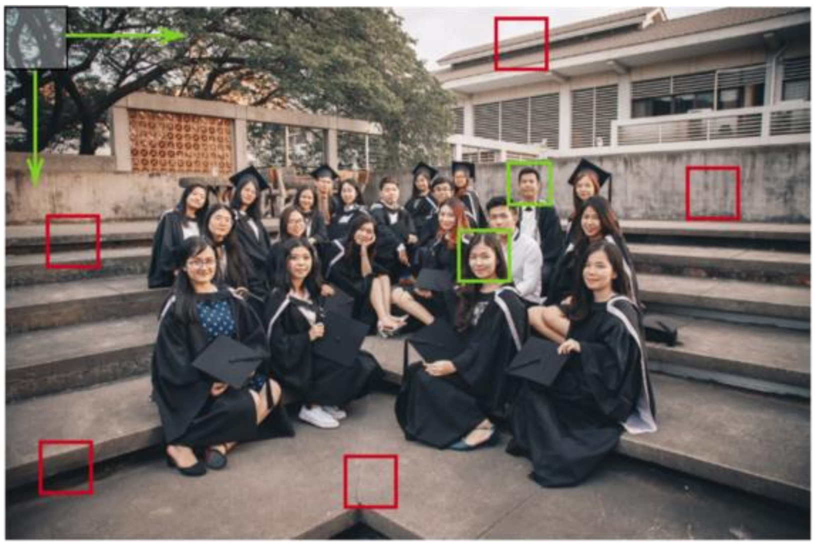


Figure 7: Sliding Windows Detection Process in the Viola-Jones Object Detection Framework

Imagine that we need to detect faces in the above image. Viola & Jones (2001) use a sliding window approach where window of various scales are slid across the entire image. The scale factor and the shifting step size are parameters for you to decide upon. So for the above image, there are *m* sub-windows to evaluate. For a sub-window *i*, the framework resize the image of that sub-window to a base size of 24 x 24 (to match training data), convert it into an integral image and feed it through a Cascade Classifier produced during the training phase. A face is detected if a sub-window passes through all the stages in the Cascade Classifier.

IV. Conclusion

To conclude, you have learnt about the Viola-Jones Object Detection Framework and its application for face detection. Many technologies today benefited from Paul Viola and Michael Jones's work. By understanding how the framework works, you can confidently implement your own version of their work or used an open source implementation like the one provided by OpenCV. I hope that my explanation moves you forward in that direction and compels you to create amazing technologies that uses this awesome framework.

How face detection works?

Face detection applications use algorithms and ML to find human faces within larger images, which often incorporate other non-face objects such as landscapes, buildings and other human body parts like feet or hands. Face detection algorithms typically start by searching for human eyes -- one of the easiest features to detect. The algorithm might then attempt to detect eyebrows, the mouth, nose, nostrils and the iris. Once the algorithm concludes that it has found a facial region, it applies additional tests to confirm that it has, in fact, detected a face.

To help ensure accuracy, the algorithms need to be trained on large data sets incorporating hundreds of thousands of positive and negative images. The training improves the algorithms' ability to determine whether there are faces in an image and where they are.

The methods used in face detection can be knowledge-based, feature-based, template matching or appearance-based. Each has advantages and disadvantages:

Knowledge-based, or rule-based methods, describe a face based on rules. The challenge of this approach is the difficulty of coming up with well-defined rules.
Feature invariant methods -- which use features such as a person's eyes or nose to detect a face -- can be negatively affected by noise and light.
Template-matching methods are based on comparing images with standard face patterns or features that have been stored previously and correlating the two to detect a face. Unfortunately these methods do not address variations in pose, scale and shape.
Appearance-based methods employ statistical analysis and machine learning to find the relevant characteristics of face images. This method, also used in feature

extraction for face recognition, is divided into sub-methods.
Some of the more specific techniques used in face detection include:

Removing the background. For example, if an image has a plain, mono-color background or a pre-defined, static background, then removing the background can help reveal the face boundaries.
In color images, sometimes skin color can be used to find faces; however, this may not work with all complexions.

Using motion to find faces is another option. In real-time video, a face is almost always moving, so users of this method must calculate the moving area. One drawback of this method is the risk of confusion with other objects moving in the background.

A combination of the strategies listed above can provide a comprehensive face detection method.

Detecting faces in pictures can be complicated due to the variability of factors such as pose, expression, position and orientation, skin color and pixel values, the presence of glasses or facial hair, and differences in camera gain, lighting conditions and image resolution. Recent years have brought advances in face detection using deep learning, which presents the advantage of significantly outperforming traditional computer vision methods.

Major improvements to face detection methodology came in 2001, when computer vision researchers Paul Viola and Michael Jones proposed a framework to detect faces in real time with high accuracy. The Viola-Jones framework is based on training a model to understand what is and is not a face. Once trained, the model extracts specific features, which are then stored in a file so that features from new images can be compared with the previously stored features at various stages. If the image under study passes through each stage of the feature comparison, then a face has been detected and operations can proceed.

Although the Viola-Jones framework is still popular for recognizing faces in real-time applications, it has limitations. For example, the framework might not work if a face is covered with a mask or scarf, or if a face is not properly oriented, then the algorithm might not be able to find it.

To help eliminate the drawbacks of the Viola-Jones framework and improve face detection, other algorithms -- such as region-based convolutional neural network (R-CNN) and Single Shot Detector (SSD) -- have been developed to help improve processes.

A convolutional neural network (CNN) is a type of artificial neural network used in image recognition and processing that is specifically designed to process pixel data. An R-CNN generates region proposals on a CNN framework to localize and classify objects in images.

While region proposal network-based approaches such as R-CNN need two shots -- one for generating region proposals and one for detecting the object of each proposal -- SSD only requires one shot to detect multiple objects within the image. Therefore, SSD is significantly faster than R-CNN.

Face detection vs. face recognition

Although the terms face detection and face recognition are often used together, facial recognition is only one application for face detection -- albeit one of the most significant ones. Facial recognition is used for unlocking phones and mobile apps as well as for Biometric verification. The banking, retail and transportation-security industries employ facial recognition to reduce crime and prevent violence.

In short, the term face recognition extends beyond detecting the presence of a human face to determine whose face it is. The process uses a computer application that captures a digital image of an individual's face -- sometimes taken from a video frame -- and compares it to images in a database of stored records.

Uses of face detection

Although all facial recognition systems use face detection, not all face detection systems are used for facial recognition. Face detection can also be applied for facial motion capture, or the process of electronically converting a human's facial movements into a digital database using cameras or laser scanners. This database can be used to produce realistic computer animation for movies, games or avatars.

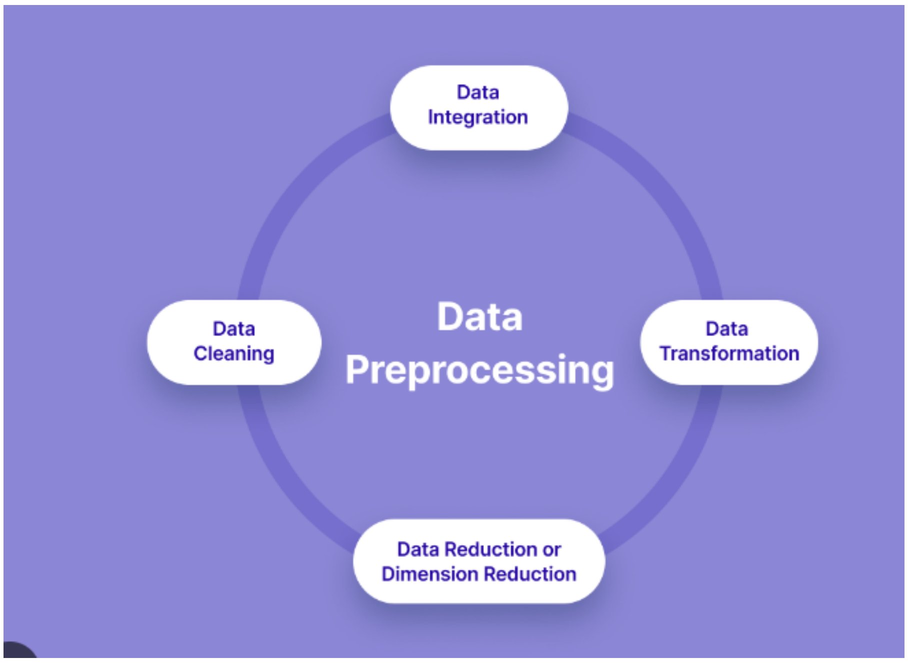
Face detection can also be used to auto-focus cameras or to count how many people have entered an area. The technology also has marketing applications -- for example, displaying specific advertisements when a particular face is recognized.

Another application for face detection is as part of a software implementation of emotional inference, which can, for example, be used to help people with autism understand the feelings of people around them. The program "reads" the emotions on a human face using advanced image processing.

An additional use is drawing language inferences from visual cues, or "lip reading." This can help computers determine who is speaking, which may be helpful in security applications. Furthermore, face detection can be used to help determine which parts of an image to blur to assure privacy.

Data collection and processing:

We will implement our use case using the Haar Cascade classifier. Haar Cascade classifier is an effective object detection approach which was proposed by Paul Viola and Michael Jones in their paper, “Rapid Object Detection using a Boosted Cascade of Simple Features” in 2000. This is basically a machine learning based approach where a cascade function is trained from a lot of images both positive and negative. Based on the training it is then used to detect the objects in the other images.



Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

Why do we need Data Preprocessing?

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

SOFTWARE REQUIREMENTS

VsCode:

Visual Studio Code is a free coding editor that helps you start coding quickly. Use it to code in any programming language, without switching editors. Visual Studio Code has support for many languages, including Python, Java, C++, JavaScript, and more.

Visual Studio Code is a free, lightweight but powerful source code editor that runs on your desktop and on the web and is available for Windows, macOS, Linux, and Raspberry Pi OS. It comes with built-in support for JavaScript, TypeScript, and Node.js and has a rich ecosystem of extensions for other programming languages (such as C++, C#, Java, Python, PHP, and Go), runtimes (such as .NET and Unity), environments (such as Docker and Kubernetes), and clouds (such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform).

Aside from the whole idea of being lightweight and starting quickly, Visual Studio Code has IntelliSense code completion for variables, methods, and imported modules; graphical debugging; linting, multi-cursor editing, parameter hints, and other powerful editing features; snazzy code navigation and refactoring; and built-in source code control including Git support. Much of this was adapted from Visual Studio technology.

Haarcascade_frontalface_default.xml:

It is an Object Detection Algorithm used to identify faces in an image or a real time video. The algorithm uses edge or line detection features proposed by Viola and Jones in their research paper “Rapid Object Detection using a Boosted Cascade of Simple Features” published in 2001. The algorithm is given a lot of positive images consisting of faces, and a lot of negative images not consisting of any face to train on them. The model created from this training is available at the OpenCV GitHub repository
https://github.com/opencv/opencv/tree/master/data/haarcascades.

The repository has the models stored in XML files, and can be read with the OpenCV methods. These include models for face detection, eye detection, upper body and lower body detection, license plate detection etc.

Data set:

In the Viola — Jones research, they had a total of 38 stages for something around 6000 features. The number of features in the first five stages are 1, 10, 25, 25, and 50, and this increased in the subsequent stages. The initial stages with simpler and lesser number of features removed most of the windows not having any facial features, thereby reducing the false negative ratio, whereas the later stages with complex and more number of features can focus on reducing the error detection rate, hence achieving low false positive ratio.

Detector.py

```
# Import libraries
import cv2
import numpy as np

n=int(input("Select Option : \n 1.Face Detection From Live Video\n 2.Face
Detection From Images\n"))
```

```
if(n==1) :
   # Video capture using WebCam
   cap = cv2.VideoCapture(0)

   # print a feedback
   print('Camera On')

   # Load face detection classifier
   # Load face detection classifier ~ Path to face & eye cascade
   face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")  # Pre train model

   while True:
   # Original frame ~ Video frame from camera3
      ret, frame = cap.read()  # Return value (true or false) if the capture
work, video frame

      # Convert original frame to gray
      gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

      # Get location of the faces in term of position
      # Return a rectangle (x_pos, y_pos, width, height)
      faces = face_cascade.detectMultiScale(gray, 1.2, 5, minSize=(30, 30),
flags=cv2.CASCADE_SCALE_IMAGE)

      # Detect faces
      for (x, y, w, h) in faces:
         # Draw rectangle in the face
         cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 53, 18), 2)  # Rect for
the face

      # Load video frame
      cv2.imshow('Video Frame', frame)

      # Wait 1 millisecond second until q key is press
      # Get a frame every 1 millisecond
      if cv2.waitKey(1) == ord('q'):
         # Print feedback
         print('Camera Off')
         break

   # Close windows
   cap.release()  # Realise the webcam
   cv2.destroyAllWindows()  # Destroy all the windows

else:
   pixels = cv2.imread('1.jpg')
   # load the pre-trained model
   classifier = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")
   # perform face detection
   bboxes = classifier.detectMultiScale(pixels)
   # print bounding box for each detected face
```

```
for box in bboxes:
    # extract
    x, y, width, height = box
    x2, y2 = x + width, y + height
    # draw a rectangle over the pixels
    cv2.rectangle(pixels, (x, y), (x2, y2), (0,0,255), 1)
# show the image
cv2.imshow('face detection', pixels)
# keep the window open until we press a key
cv2.waitKey(0)
# close the window
cv2.destroyAllWindows()
```

Numpy:

which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package. It stands for 'Numerical Python'.

Pandas:

is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays.

Sklearn:

is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction

Matploatlib:

is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

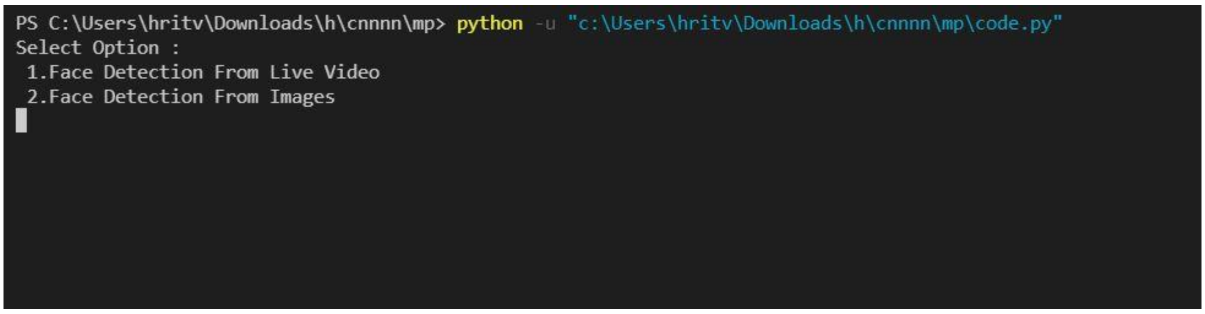
Scipy:

is a collection of mathematical algorithms and convenience functions built on the NumPy extension of Python. It adds significant power to the interactive Python session by providing the user with high-level commands and classes for manipulating and visualizing data.

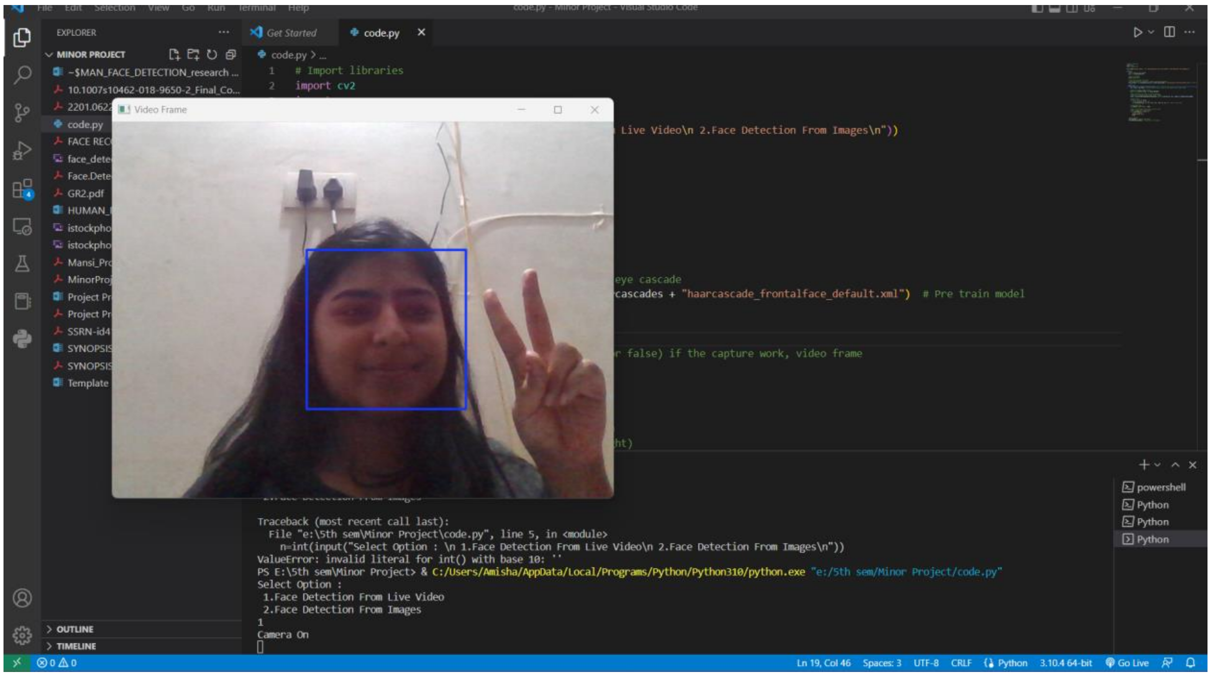
Open CV:

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

OUTPUT:

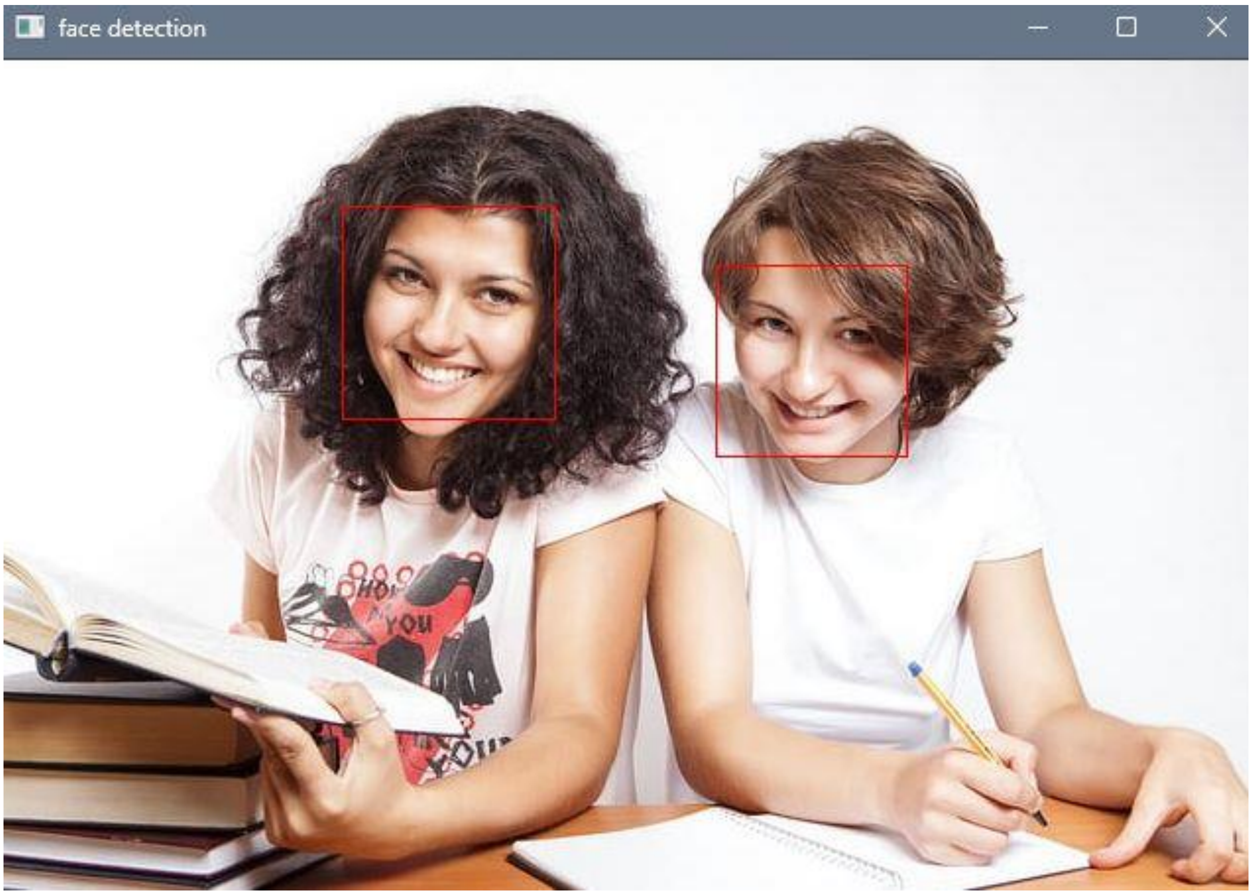


```
PS C:\Users\hritv\Downloads\h\cnnnn\mp> python -u "c:\Users\hritv\Downloads\h\cnnnn\mp\code.py"
Select Option :
 1.Face Detection From Live Video
 2.Face Detection From Images
```



```
# Import libraries
import cv2
```

```
Traceback (most recent call last):
  File "e:\5th sem\Minor Project\code.py", line 5, in <module>
    n=int(input("Select Option : \n 1.Face Detection From Live Video\n 2.Face Detection From Images\n"))
ValueError: invalid literal for int() with base 10: ''
PS E:\5th sem\Minor Project> & C:/Users/Amisha/AppData/Local/Programs/Python/Python310/python.exe "e:/5th sem/Minor Project/code.py"
Select Option :
 1.Face Detection From Live Video
 2.Face Detection From Images
1
Camera On
```



6 RESULTS AND DISCUSSION

In recent years face detection has achieved considerable attention from researchers in biometrics, pattern recognition, and computer vision groups. There is countless security, and forensic applications requiring the use of face recognition technologies. As you can see, face detection system is very important in our day-to-day life. Among the entire sorts of biometric, face detection and recognition system are the most accurate. In this article, we have presented a survey of face detection techniques. It is exciting to see face detection techniques be increasingly used in real-world applications and products. Applications and challenges of face detection also discussed which motivated us to do research in face detection. The most straightforward future direction is to further improve the face detection in presence of some problems like face occlusion and non-uniform illumination. Current research focuses in field of face detection and recognition is the detection of faces in presence of occlusion and non-uniform illumination. A lot of work has been done in face detection, but not in presence 12367 Face detection techniques: a review of problem of presence of occlusion and non-uniform illumination. If it happens, it will help a lot to face recognition, face expression recognition etc. Currently many companies providing facial biometric in mobile phone for purpose of access. In future it will be used for payments, security, healthcare, advertising, criminal identification etc.

Project:

Below, we can see step-by-step guide to Human Face Detector:

The methods of face detection can be classified into four categories:

• Methods for a priori. These methods based on the rules used to model the knowledge of what makes a face. Typically, these rules represent relationships in facial features.

• Approaches invariant features. These approaches are based on structural features that exist even when the pose, the view or the illumination conditions vary, and use them to locate faces.

• Methods based models. Several standard models of faces are used to define a face model or models of facial features separately.

• The correlation between the image and the models is evaluated for the presence of face.

• Methods for learning. In contrast to methods based models, the models are learned from a set of training images which should allow characterizing the variability of the appearance of a face. These models are then used to learn the detection.

Importing libraries:

```
# Import libraries
import cv2
import numpy as np

n=int(input("Select Option : \n 1.Face Detection From Live Video\n 2.Face
Detection From Images\n"))

if(n==1) :

  # Video capture using WebCam
  cap = cv2.VideoCapture(0)

  # print a feedback
  print('Camera On')

  # Load face detection classifier
  # Load face detection classifier ~ Path to face & eye cascade
  face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")  # Pre train model

  while True:
  # Original frame ~ Video frame from camera3
    ret, frame = cap.read()  # Return value (true or false) if the capture work, video
```

```
frame

    # Convert original frame to gray
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Get location of the faces in term of position
     # Return a rectangle (x_pos, y_pos, width, height)
    faces = face_cascade.detectMultiScale(gray, 1.2, 5, minSize=(30, 30),
flags=cv2.CASCADE_SCALE_IMAGE)

    # Detect faces
    for (x, y, w, h) in faces:
        # Draw rectangle in the face
        cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 53, 18), 2)  # Rect for the face

    # Load video frame
    cv2.imshow('Video Frame', frame)

    # Wait 1 millisecond second until q key is press
    # Get a frame every 1 millisecond
    if cv2.waitKey(1) == ord('q'):

        # Print feedback
        print('Camera Off')
        break

# Close windows
cap.release()  # Realise the webcam
cv2.destroyAllWindows()  # Destroy all the windows
```

```
else:
   pixels = cv2.imread('1.jpg')
   # load the pre-trained model
   classifier = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")
   # perform face detection
   bboxes = classifier.detectMultiScale(pixels)
   # print bounding box for each detected face
   for box in bboxes:
      # extract
      x, y, width, height = box
      x2, y2 = x + width, y + height
      # draw a rectangle over the pixels
      cv2.rectangle(pixels, (x, y), (x2, y2), (0,0,255), 1)
   # show the image
   cv2.imshow('face detection', pixels)
   # keep the window open until we press a key
   cv2.waitKey(0)
   # close the window
   cv2.destroyAllWindows()
```

After making use of my dataset to the MTCNN procedure, I determined the face of the images for approximately a hundred videos at a rate of 99%-100%. Right here, the end result suggests that a great final result has been finished: the use of multi-venture cascaded Convolutional networks. “Fig. 4”, presents a number of the pix from my dataset in which the face has been efficiently decided by the usage of the MTCNN manner

a.DATA FLOW DIAGRAM:

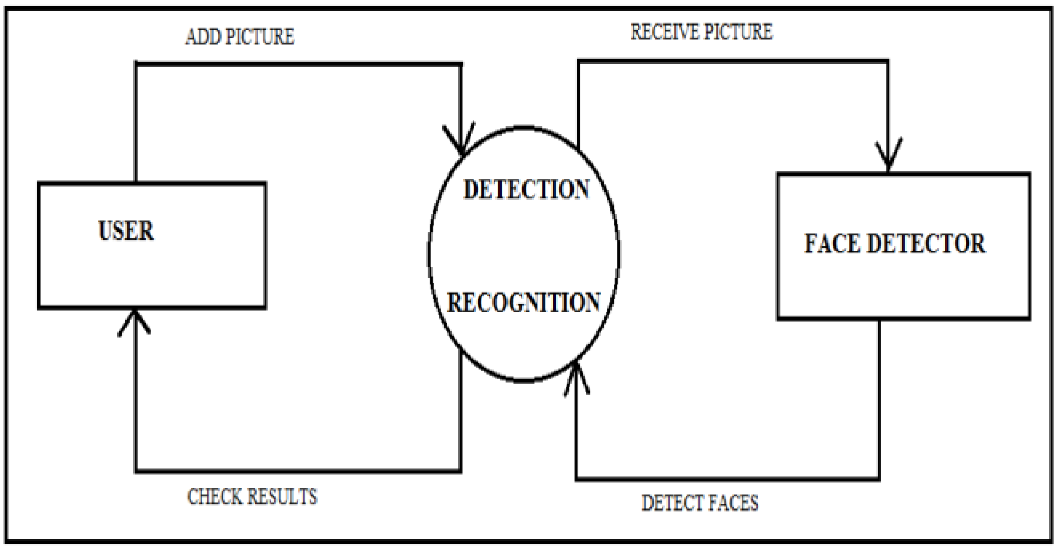


Fig1: data flow diagram (level 0)

b. ENTITY RELATIONSHIP DIAGRAM:

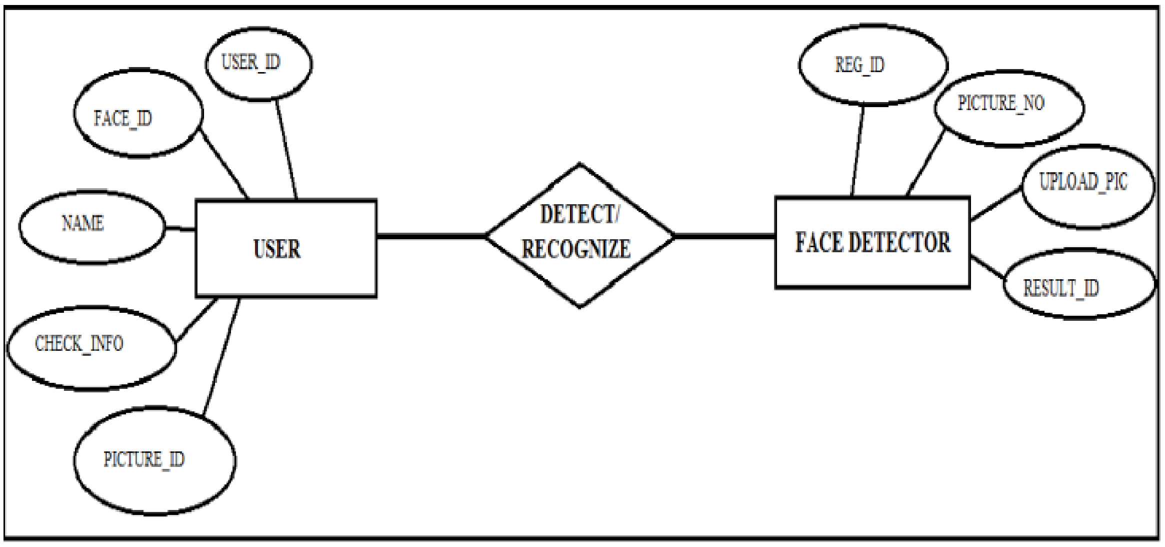


Fig2: E-R diagram

7 Future Scope:

Future work is to work on the same domain but to track a particular face in a video sequence. That is like avoiding all other faces except the face required. To evaluate various face detection and recognition methods, provide complete solution for image based face detection and recognition with higher accuracy, better response rate as an initial step for video surveillance.

8 REFERENCES

1. Face Recognition by Independent Component Analysis Marian Stewart Bartlett,Member, IEEE, Javier R. Movellan, Member, IEEE, and Terrence J. Sejnowski, Fellow,IEEE

2. Eigenspace-Based Face Recognition: A Comparative Study of Different ApproachesJavier Ruiz-del-Solar, Member, IEEE, and Pablo Navarrete, Student Member, IEEE

3. Face Recognition by Extending Elastic Bunch Graph Matching with Particle SwarmOptimization

4. Discriminant Analysis of Principal Components for Face Recognition.

5. www.wikipedia.com