A project Submitted [illegible] partial [illegible] of the requirements for the [illegible] of the Degree of

[illegible] of [illegible]

DECLARATION OF CERTIFICATE

I the undersigned solemnly declare that the project report on "ATTENDANCE SYSTEM USING FACE RECOGNIZATION" is based on my work carried out during the course of our study under the supervision of Asst. Professor

I assert the statements made and conclusions drawn are an outcome of my research work. I further certify that

- **The work contained in the report is original and has been done by me under the general supervision of my supervisor.**
- **The work has not been submitted to any other Institution for any other degree/diploma/certificate in this university or any other University.**
- **We have followed the guidelines provided by the University in writing the report.**
- **Whenever we have used materials(data,theoretical analysis, and text) from other sources, we have given due credit to them in the text of the report and giving their details in the references.**

Date:

Signature

Asst. Professor

Application

External Examiner

CERTIFICATE OF APPROVAL

This is to certify that the project report entitled “ATTENDANCE SYSTEM USING FACE RECOGNIZATION”,carried out b from Department of Computer Application At DR.Shyama Prasad University R is hereby approved after proper examination and evaluation as a creditable work for the partial fulfillment of the requirement for the awarding the degree of Bachelor of Science IN Computer Application from the ye

Department of CA&IT

ACKNOWLEDGEMENT

I would like to express my gratitude and appreciation to all those who gave me the opportunity to complete this report. A special thanks to my supervisor, Dr. Sayed Jaffar Abbas, for his guidance, patience, and support. I consider myself very fortunate to be able to work with someone as considerate and encouraging as him.

I owe many thanks to my classmate and all of my friends, especiall[redacted] that always support and give full attention for me to solve my problem. They always help me in exchanging any ideas and give the enjoyable studying environment. They made my life at [redacted] a truly memorable experience and their friendships are invaluable to me.

A special thanks to @Barmer from stackoverflow for helping with the correction of syntax.I appreciate myself for persevering with this endeavour and coming up with so many amusing ideas.

I want to appreciate such YouTube channels for improving their courses since they enabled me to have a thorough understanding of Python.

****Follow The Light...!!!****

ABSTRACT

The face is one of the easiest ways to distinguish the individual identity of each other. Face recognition is a personal identification system that uses personal characteristics of a person to identify the person's identity.
Human face recognition procedure basically consists of two phases, namely face detection, where this process takes place very rapidly in humans, except under conditions where the object is located at a short distance away, the next is the introduction, which recognize a face as individuals.
Stage is then replicated and developed as a model for facial image recognition (face recognition) is one of the much-studied biometrics technology and developed by experts.
Haar Cascade is an Object Detection Algorithm used to identify faces in an image or a real time video. The algorithm used edge or line detection features proposed by Viola and Jones.

The management of the attendance can be a great burden on the teachers if it is done by hand. To resolve this problem, smart and auto attendance management system is being utilized. By utilizing this framework, the problem of proxies and students being marked present even though they are not physically present can easily be solved. This system marks the attendance using live video stream. The frames are extracted from video using OpenCV. The main implementation steps used in this type of system are face detection and recognizing the detected face, for which dlib is used. After these, the connection of recognized faces ought to be conceivable by comparing with the database containing student's faces. This model will be a successful technique to manage the attendance of students.
Mysql is been used for various queries offered by the database server for performining various operations to operational data.

Keywords: face detection,python ,mysql,Haar Cascade,OpenCV,Attendance system, Automated attendance, Image Processing, Face detection, Feature matching, Face recognition

Extension: There are vast number of applications from this face detection project, this project can be extended that the various parts in the face can be detect which are in various directions and shapes.

TABLE OF CONTENT

TOPIC **PAGE**

INTRODUCTION

Human face plays an important role in our day to day life mostly for identification of a person. Face recognition is a part of biometric identification that extracts the facial features of a face, and then stores it as a unique face print to uniquely recognize a person. Biometric face recognition technology has gained the attention of many researchers because of its wide application. Face recognition technology is better than other biometric based recognition techniques like finger-print, palm-print, iris because of its non-contact process. Recognition techniques using face recognition can also recognize a person from a distance, without any contact or interaction with person.

The face recognition techniques are currently implemented in social media websites like Facebook, at the airports, railway stations. The, at crime investigations. Face recognition technique can also be used in crime reports, the captured photo can be stored in a database, and can be used to identify a person.

Facebook uses the facial recognition technique for automating the process of tagging people. For face recognition we require large dataset and complex features to identify a person in all conditions like change of illumination, age, pose, etc.

Recent researches show there is a betterment in facial recognition systems. In the last ten years there is huge development in recognition techniques. But currently most of the facial recognition techniques is able to work fine only if the number of people in one frame is very few and under controlled illumination, proper position of faces and clear images. For face recognition purpose, there is a need for large data sets and complex features to uniquely identify the different subjects by manipulating different obstacles like illumination, pose and aging. During the recent few years, a good improvement has been made in facial recognition systems. In comparison to the last decade, one can observe an enormous development in the world of face recognition. Currently, most of the facial recognition systems perform well with limited faces in the frame. Moreover, these methodologies have been tested under controlled lighting conditions, proper face poses and non-blurry images. The system that is proposed for face recognition in this paper for attendance system is able to recognize multiple faces in a frame without any control on illumination, position of face.

In epidemic situations such as the novel coronavirus (COVID-19) pandemic, face masks have become an essential part of daily routine life. The face mask is considered as a protective and preventive essential of everyday life against the coronavirus. Many organizations using a fingerprint or card-based attendance system had to switch towards a face-based attendance system to avoid direct contact with the attendance system.
With further enhancement we can enable this project to detect faces with mask on.

The Attendance System using Face – Recognition is a replacement way method for the traditional way of marking attendance. The proposed system is python , tkinter based system supported with MySQL database.
This system can be implemented on a single faculty system of a particular institute. This system is proposed to be based on biometrics .i.e. Face Authentication. Since there is presence of biometrics, this system completely eliminates the chances of fake attendance which is a problem with the traditional methods of attendance.

The Attendance management is the significant process that were carry out in every institute to monitor the performance of the student. Every institute does this is its own way. Some of there institute use the old paper or file-based system and some have adopted strategies of automated attendance system using some biometric technique. A facial recognition system is a computerized software which is suited for determining or validating a person by performing comparisons on patterns based on their facial appearances. In this system OpenCV & Face Recognition libraries were used which are one of the popular libraries for face detection by using these libraries system first capturing the student photos and storing them into the database which were further used for the training purpose after that at the time of attendance when system camera get on system will detect the faces that were present in the frame the faces were detected by using HOG i.e. (Histogram of Oriented Gradients) which were carrier out in the system. after that if image that were present in the frame is tilted then Face Landmark Estimation algorithm will be carried out and face will be transformed to be as close as possible to perfectly centered. After that system will encode all the images that were present in the database as well as the face which were detected in the frame. For performing encoding Deep Conversional Neural Network algorithm will get carried out & for each face 128 measurements were generated then the measurements of the face that were detected in frame it get compared with the measurements of the faces that were present in the image which is earlier stored in the database. So at last by using simple liner SVM algorithm system will find the person in database of know peoples (i.e. capture at the starting of the project) who has closest measurements to the image that were detected by camera .

After finding perfect match system will generate the name of the person and store the entry in mysql database along with date and time.attendance is marked for the whole day is only one. If a person tries to mark his/her attendance repeatedly, the system ignores the person's marking with the initial data received from the beginning.

HISTORY

The dawn of Facial Recognition – 1960s

The earliest pioneers of facial recognition were Woody Bledsoe, Helen Chan Wolf and Charles Bisson. In 1964 and 1965, Bledsoe, along with Wolf and Bisson began work using computers to recognise the human face.

Due to the funding of the project originating from an unnamed intelligence agency, much of their work was never published. However, it was later revealed that their initial work involved the manual marking of various "landmarks" on the face such as eye centres, mouth etc. These were then mathematically rotated by a computer to compensate for pose variation. The distances between landmarks were also automatically computed and compared between images to determine identity

Advancing the accuracy of Facial Recognition – 1970s

Carrying on from the initial work of Bledsoe, the baton was picked up in the 1970s by Goldstein, Harmon and Lesk who extended the work to include 21 specific subjective markers including hair colour and lip thickness in order to automate the recognition.

While the accuracy advanced, the measurements and locations still needed to be manually computed which proved to be extremely labour intensive yet still represents an advancement on Bledsoe's RAND Tablet technology.

Using linear algebra for Facial Recognition – 1980s/90s

It wasn't until the late 1980s that we saw further progress with the development of Facial Recognition software as a viable biometric for businesses. In 1988, Sirovich and Kirby began applying linear algebra to the problem of facial recognition.

A system that came to be known as Eigenface showed that feature analysis on a collection of facial images could form a set of basic features. They were also able to show that less than one hundred values were required in order to accurately code a normalized facial image.

In 1991, Turk and Pentland carried on the work of Sirovich and Kirby by discovering how to detect faces within an image which led to the earliest instances of automatic facial recognition. This significant breakthrough was hindered by technological and environmental factors, however, it paved the way for future developments in Facial Recognition technology.

FERET Programme – 1990s/2000s

The Defence Advanced Research Projects Agency (DARPA) and the National Institute of Standards and Technology (NIST) rolled out the Face Recognition Technology (FERET) programme in the early 1990s in order to encourage the commercial facial recognition market. The project involved creating a database of facial images. Included in the test set were 2,413 still facial images representing 856 people. The hope was that a large database of test images for facial recognition would inspire innovation and may result in more powerful facial recognition technology.

Face Recognition Vendor Tests – 2000s

The National Institute of Standards and Technology (NIST) began Face Recognition Vendor Tests (FRVT) in the early 2000s. Building on FERET, FRVTs were designed to provide independent government evaluations of facial recognition systems that were commercially available, as well as prototype technologies. These evaluations were designed to provide law enforcement agencies and the U.S. government with the information necessary to determine the best ways to deploy facial recognition technology.

Face Recognition Grand Challenge – 2006

Launched in 2006, the primary goal of the Face Recognition Grand Challenge (FRGC) was to promote and advance face recognition technology designed to support existing face recognition efforts in the U.S. Government[2].

The FRGC evaluated the latest face recognition algorithms available. High-resolution face images, 3D face scans, and iris images were used in the tests. The results indicated that the new algorithms were 10 times more accurate than the face recognition algorithms of 2002 and 100 times more accurate than those of 1995, showing the advancements in facial recognition technology over the past decade.

Social Media – 2010-Current

Back in 2010, Facebook began implementing facial recognition functionality that helped identify people whose faces may feature in the photos that Facebook users update daily. The feature was instantly controversial with the news media, sparking a slew of privacy-related articles. However, Facebook users by and large did not seem to mind. Having no apparent negative impact on the website's usage or popularity, more than 350 million photos are uploaded and tagged using face recognition each day.

iPhone X – 2017

Facial Recognition technology advanced rapidly from 2010 onwards and September 12, 2017, was another significant breakthrough for the integration of facial recognition into our day to day lives. This was the date that Apple launched the iPhone X – the first iPhone users could unlock with FaceID – Apple's marketing term for facial recognition.

NEC and Facial Recognition

Border controls, airlines, airports, transport hubs, stadiums, mega-events, concerts, and conferences. Biometrics is playing a growing role not only in the real-time policing and securing of increasingly crowded and varied venues worldwide but also in ensuring a smooth, enjoyable experience for the citizens who visit them.

As a long-time committed pioneer of biometric research and solutions, NEC has developed multi-modal technologies including face, iris and voice recognition, finger and palmprint identification, and ear acoustic authentication, and supplemented them with AI and data analytics to enhance situational awareness and facilitate effective real-time or post-event action in both law-enforcement and consumer-oriented spheres.

Face recognition can often prove one of the best biometrics because images can be taken without touching or interacting with the individual being identified, and those images are recorded and instantly checked against existing databases.

NEC's face recognition offers high-performance, scalable solutions for the most demanding real-time or post-event requirements. With face surveillance, search, identification and verification functions all on a single platform, it can be easily integrated into existing surveillance systems to extract faces in real-time, match them against an existing database or watchlist and produces real-time alerts to help reduce public safety risks.

With the ability to process and analyse multiple camera feeds and thousands of faces per minute, NEC's powerful face recognition is able to address the largest and most difficult security challenges with unparalleled efficiency, sensitivity, and perception.

NEC Face Recognition in Action

In response to booming demand for varied biometrics, NEC has expanded its traditional range of face recognition applications from law-enforcement and security provision to new areas, and now boasts more than 1,000 active systems in over 70 countries and regions spanning police, immigration control agencies, national ID, banking, entertainment, stadium, conference venue systems, and many more.

Right now, many of our face recognition technology solutions are blazing a trail for first-time use in new areas and venues worldwide, such as end-to-end airport travel experiences in the U.S., mega-event surveillance in Japan, and EU Summit security in Europe.

In 2019, NEC was named Frost & Sullivan Asia Pacific Biometrics Company of the Year in recognition of its leading position in the industry and foresight in innovating and developing future face recognition biometric solutions that maximize customer value and experience. In 2021, NEC was awarded with the 2020 Global Biometrics in Security Market Growth Innovation & Leadership Excellence Frost Radar™ Award by Frost & Sullivan.

The award reflects NEC's pivot toward biometric solutions, which are providing the company with significant growth opportunities with which to develop new, large-scale identity solutions and use cases outside of traditional biometric markets. This focus on identity solutions, powered by biometrics, has allowed NEC to go beyond traditional government or enterprise customers and to move toward a B2C strategy through tailored vertical-specific solutions.

Balancing security and privacy

As the leading global pioneer of face recognition and other biometric technologies, and champion of our NEC Safer Cities Vision, NEC is fully vested in developing biometric recognition solutions and services that contribute to the creation of safe, secure, equal and efficient communities around the world.

Having positioned safety business as one of our key pillars of growth, we are keen to encourage businesses, consumers, and governments to work together to help balance the need for privacy with the benefits of protecting our society, securing our borders and providing consumer convenience without the fear of negative consequences.

At NEC, we strongly believe that face recognition can add significant value to our lives, and we seek to advance these technologies in ways that respect the worldwide principles of freedom, justice, rights to privacy, transparency and continuous improvement..

Vision

Replacement the method for the traditional way of marking attendance:

Nowadays Educational institutions are concerned about regularity of student attendance. Even in pandemic situation attendance is still a major issue in schools and colleges. Mainly there are two conventional methods of marking attendance which are calling out the roll call or by taking student sign on paper. They both were more time consuming and difficult. Hence, there is a requirement of computer-based student attendance management system which will assist the faculty for maintaining attendance record automatically. In this project we have implemented the automated attendance system using ‘TKINTER’ and ‘PYTHON’. We have projected our ideas to implement an “Automated Attendance System Based on Face Recognition”.

To reduce the use of Paper:

If one class uses two attendance registers and twelve of the classes are there, it costs twenty-four registers every year. But storing the records automatically in a database enables us to cope with this problem.

To provide high-level information security :

This project categorises the authenticity into two individuals: the student and the administrator/teacher. Only by logging into their respective accounts can students view their personal information. A teacher can see an overall student's details. because a teacher can attend more than one class. They can do so by Logging into their repective accounts.

To save the time of both teacher and Student:

making attendance by automated system is fast over manual marking of attendance. It saves the time of both student and teacher.

To provide high accuracy of marking attendance:

The traditional way of marking attendance includes calling the name of a student and marking in the response of that student.

In this overall process, there may be flaws in mishearing or responding in the name of another student . This could be elemenated by the automated process of recognition.

To promote untouchability:

Many organizations using a fingerprint or card-based attendance system had to switch towards a face-based attendance system to avoid direct contact with the attendance system.

However, face mask adaptation brought a new challenge to already existing commercial biometric facial recognition techniques in applications such as facial recognition access control and facial security checks at public places

To Centralize the data:

Data is stored in same database and three different table with reduced data redudency and data inconsistency with in mind. Before inserting data in the database the entered data is checked to be ensure is there any data redudency. To save the storage space the data is fetched by another table .

To increase the Accuracy to Recognision:

Face alignment is another stage that could further improve the quality of recognition. The most important stage still remains the face representation. Designing new hybrid methods for the training as well as neural architecture search for optimizing the models for large-scale face datasets is a direction to look at. Designing training methods to minimize the negative effects of data label noise would increase the accuracy and make the recognition more robust for more extreme environments with low image quality, varying facial poses, and low illumination. Of course Accuracy is propotional to the sample image taken. Although Hundred Images are been taken for increase Accuracy of recognition.

To detect multiple Faces at a Same time:

Being able to recognise multiple multiple faces concorrently saves a bunch of time over recognising faces one by one.

Misson:

" Face Recognition Solution works by helping people to manage the daily threats that occur in a regular basis at premises. It helps in recovering from the unexpected ones and thereby meets the security requirements of people ."

We comprise of people, who believe in working together to reach the common goal by sharing variable thoughts and ideas, which could ultimately enhance the security at premises. By this we bring diverse talent at a common place and nourish it with the experiences gained from different fields and thereby we proceed to serve our customers.

We believe that our success rely on quality service, reliability, mutual trust and total customer satisfaction. In order to accomplish the above factors our employees give their best to achieve the same. We prioritize our customers first and therefore offer all the products and the solutions that can benefit them the most.

We want to put our joint / team effort to meet the present security requirements and thereby want to enhance the security solutions. We are in the hope to create a long lasting business relationship with our respectable customers.

In future, we are planning the same, as meeting the customer's expectations is our primary agenda therefore we will continue to follow the same way as we are currently following by providing the security solutions to enhance our customer's security. Our client's requirements will determine our path and our morals and business ethics will guide us in the right direction to proceed.

Objective:

The objective of face recognition is, from the incoming image, to find a series of data of the same face in a set of training images in a database. The great difficulty is ensuring that this process is carried out in real-time, something that is not available to all biometric facial recognition software providers.

The facial recognition process can perform two variants depending on when it is performed:

●The one in which, for the first time, a facial recognition system addresses a face to register it and associate it with an identity, in such a way that it is recorded in the system. This process is also known as digital onboarding with facial recognition.

●The variant in which the user is authenticated, prior to being registered. In this process, the incoming data from the camera is crossed with the existing data in the database. If the face matches an already registered identity, the user is granted access to the system with his credentials.

The face recognition procedure simply requires any device that has digital photographic technology to generate and obtain the images and data necessary to create and record the biometric facial pattern of the person that needs to be identified.

Unlike other identification solutions such as passwords, verification by email, selfies or images, or fingerprint identification, Biometric facial recognition uses unique mathematical and dynamic patterns that make this system one of the safest and most effective ones.

One of the most challenging tasks for visual form ('shape') analysis and object recognition is the understanding of how people process and recognize each other's face, and the development of corresponding computational models.

Attendance marking in a classroom during a lecture is not only a onerous task but also a time consuming one at that. Due to an unusually high number of students present during the lecture there will always be a probability of proxy attendance(s).Attendance marking with conventional methods has been an area of challenge. The growing need of efficient and automatic techniques of marking attendance is a growing challenge in the area of face recognition. In recent years, the problem of automatic attendance marking has been widely addressed through the use of standard biometrics like fingerprint and Radio frequency Identification tags etc., However,these techniques lack the element of reliability. In this proposed project an automated attendance marking and management system is proposed by making use of face detection and recognition algorithms. Instead of using the conventional methods, this proposed system aims to develop an automated system that records the student’s attendance by using facial recognition technology.

The main objective of this work is to make the attendance marking and management system efficient, time saving, simple and easy. Here faces will be recognized using face recognition algorithms. The processed image will then be compared against the existing stored record and then attendance is marked in the database accordingly. Compared to existing system traditional attendance marking system, this system reduces the workload of people. This proposed system will be implemented with 4 phases such as Image Capturing, Segmentation of group image and Face Detection, Face comparison and Recognition, Updating of Attendance in database.

Our primary goal is to help the lecturers, improve and organize the process of track and manage student attendance and absenteeism. Additionally, we seek to:

- Provides a valuable attendance service for both teachers and students.
- Reduce manual process errors by provide automated and a reliable attendance system uses face recognition technology.
- Increase privacy and security which student cannot presenting himself or his friend while they are not.
- Produce monthly reports for lecturers.
- Flexibility, Lectures capability of editing attendance records.
- Calculate absenteeism percentage and send reminder messages to students.

▪ To develop a portable Smart Attendance System which is handy and self-powered

SWOT Analysis:

Strength:

- The system stores the faces that are detected and automatically marks attendance.

  Simply put, a face recognition time clock makes use of facial recognition technology to identify and verify a person and mark attendance automatically.

  Fingerprint scanning systems are almost the standard for attendance systems but the recent struggle with the pandemic has brought forth the issue with systems that require physical contact. A facial recognition time clocking system is a contactless technology that provides freedom from any physical interaction between the man and the machine.

- Provide authorized access.

  Continuity of decision: Access control is verified before and during the access. Access to resources can be revoked after it has been granted due to a change of some object or subject attributes.

  Mutability of attributes: Subject's or object's attributes can be mutable (i.e., student,student details) or immutable (i.e., Exam Roll Number). Immutable attributes can’t be mutable.

- Ease of use.

  Ease of use is a basic concept that describes how easily users can use a product. Design teams define specific metrics per project—e.g., “Users must be able to tap Find within 3 seconds of accessing the interface.”—and aim to optimize ease of use while offering maximum functionality and respecting business limitations

- Multiple face detection.

  The face recognition algorithms based in PCA (Principal Component Analysis) do multiple comparisons and matches between a face detected and the trained images stored in binary database for this reason, and for improving the accuracy of recognition, you should add several images of the same person in different angles, positions and luminance conditions, this training makes this prototype solid and very accurate.

- Provide methods to maximize the number of extracted faces from an image.

  Provieded algorithm uses 100 images of the person to train.This provide accuracy to recognise and locate less resourse.

  The method of face detection in pictures is complicated because of variability present across human faces such as pose, expression, position and orientation, skin colour, the presence of glasses or facial hair, differences in camera gain, lighting conditions, and image resolution.

- Can store more than one image for a user to maximize face detection.
- High accuracy.

  Objective of any system is to make ensure the client that the proposed system is reliable . to gain relibility the system have to output to much accurate.
- Showing the names of who is absent or late in the screen to avoid errors.
- Search of specific student and his/her attendance
- Permitting the lecturer to modify the student absent or late.
- Student can show his /her percentage of attendance
- Speed and Responsiveness: Execution of operations should be fast.
- Auto semi-automatic system: System designed to less the client intervention.
- Modifiability: the system should be easy to modify, any wrong should be correct.
- Security: the system should be secure and saving student's privacy.
- Usability: the system should be easy to deal with and simple to understand.
- Maintainability: the maintenance group should be able to fix any problem occur suddenly.

Weekness:

- number of images taken Is propotional to time it takes to train images:

  Generating more data out of your existing data set, using transformations and generative models, helps you to increase the size of your dataset. In Neural networks the general observation is more data => better accuracy. This will help you in regularization of your NN and reduce overfitting as well, Especially if your dataset is not a complete representative of the domain where the NN will be used.

- More image more accuracy:

  Data tells a story only if you have enough of it. Every data sample provides some input and perspective to your data's overall story is trying to tell you. Perhaps the easiest and most straightforward way to improve your model's performance and increase its accuracy is to add more data samples to the training data.

  Doing so will add more details to your data and finetune your model resulting in a more accurate performance. Rember after all, the more information you give your model, the more it will learn and the more cases it will be able to identify correctly.

- To identify a person the person have to set in front of the camera:

  In the real world, though, accuracy rates are usually lower. According to the CSI story, the Facial Recognition Vendor Test found that the error rate for one algorithm rose from 0.1% when faces were matched against high-quality mugshots to 9.3% when matched to pictures of individuals captured in public. Error rates rose especially when subjects were not looking directly at the camera or were partially hidden by shadows or objects.

  Aging is another challenge. The Facial Recognition Vendor Test said that middle-tier facial recognition algorithms had error rates that jumped by nearly a factor of 10 when they attempted to match photos of subjects that had been taken 18 years earlier.

- Needs proper lighting fo better recognition:

  Illumination variation is among the several bottlenecks in a face recognition system because it can greatly affect the appearance of a face image, which causes a reduction in the face matching performance of the system. Using illumination preprocessing methods is an effective approach to overcome this problem. Despite the achievements made, however, each method still has its own demerits. In this paper, an efficient representation method insensitive to varying illumination based on a combination of Weber-face (WF) and the difference of Gaussians (DoG) methods is proposed for human face recognition using local descriptors.

- data entered by student only modified by student:

  To make a change in any information the student have to do so.it is certain that if data is not modified if necessary this can be a misleading information gathering.

- The accuracy of the system is not 100%.

  Facial recognition technology has improved dramatically over the past few years. In the latest round of testing conducted by the National Institute of Standards and Technology (NIST) in March 2020, showed that the best face identification algorithm had an error rate of just 0.08%. In 2014, the leading algorithm had an error rate of 4.1%.

  In 2018 Face Recognition Vendors Tests (FRVT) carried out by NIST, NEC's facial recognition technology ranked No. 1 for the fifth time, achieving an error rate of just 0.5% – a significant improvement from 2014 and the jump again in 2020 is significant.

- The instructor and trainingSet manager still have to do some work manually.

  The accuracy of the system is not 100%.Face detection and loading training data processes just a little bit slow.The instructor and trainingSet manager still have to do some work manually

- It can only detect face from a limited distance.

  The resolution of the images provided by security cameras. If the resolution is high enough, then face recognition can be done. What also matters is the distance of the camera to people's faces. The higher a camera is fixated, the lower the resolution that people's faces will be allocated on an image.

- Odd expressions. A human face might have an odd expression, making it difficult for facial detection algorithms to identify it as a face;
- Face occlusion. If a face is hidden by hair, a hat, a hand, glasses, or a scarf, it may result in a false negative;
- Illuminations. An image might not have uniform lighting effects; part of the image may be overexposed, while another part is very dark. Again, this can contribute to false negatives;
- Complex background. When lots of objects are present in an image, face detection's accuracy is reduced;
- Too many faces. If there is a large number of human faces in an image, face detection software may have a hard time distinguishing between some of them;
- Low resolution. If an image's resolution is poor, it is more difficult to detect faces;
- Skin color. If somebody's skin color falls outside of the gradient that is recognized by the algorithm,

  their face might not be detected.

Oppertunities:

- It can be modified to another subjects:

  A face recognition based attendance system is an application that performs two main tasks - identifying a person based on key parameters and verifying his identity and authorization using a database. This technology combines artificial intelligence and machine learning to create a sophisticated solution.

  Traditional attendance methods are both time consuming and labor intensive. In large organizations that employ many employees, it can be difficult to take attendance manually and there can be misalignment of time sheets resulting in inaccurate and inefficient attendance. Employees can easily get rid of time recording options by stealing time in a manual based time and attendance system .

  Therefore, we need a face recognition-based time and attendance system that not only automates the time and attendance process, but also prevents employees from cutting corners.

- It can take part as a identifier in subway/train or train systems:

  India is an emerging and developing country of Southern Asia. It is world's largest democracy and one of the fastest growing economies. Several steps and economic development plans have been taken to convert India to Smart India. It's the digital era and it's the high time to portray our India as Tech-India and the Indian government, in no time, has taken steps towards digitisation, safety, financial and economic growth.One of the main assets contributing to our economic growth is the Indian Railways. It's the only transport giant which operates throughout India with various zones and divisions. Indian Railways has shown its activeness and proved that it is one of the base and financial backbone of our country. Here, Indian Railways is planning to take step towards the safety of passengers by recognizing and identifying the criminals through digital means and disallow them from entering into railway premises. As Indian Railways is increasing its facilities and features; passengers can use them to book food online in train and even address their complaints on social media. Recently, it has announced to implement face recognition technology at railway stations.

Threats:

- Massive data storage

  Facial recognition software relies on machine learning technology, which requires massive data sets to “learn” to deliver accurate results. Such large data sets require robust data storage. Small and medium-sized companies may not have sufficient resources to store the required data.

- Breach of privacy

  The question of ethics and privacy is the most contentious one. Governments have been known to store several citizens' pictures without their consent. In 2020, the European Commission said it was considering a ban on facial recognition technology in public spaces for up to five years, to allow time to work out a regulatory framework to prevent privacy and ethical abuses.

- Scope for error

  Facial recognition data is not free from error, which could lead to people being implicated for crimes they have not committed. For example, a slight change in camera angle or a change in appearance, such as a new hairstyle, could lead to error. In 2018, Newsweek reported that Amazon’s facial recognition technology had falsely identified 28 members of the US Congress as people arrested for crimes.

- Vulnerability in recognition

  Facial recognition technology is indeed very accurate and no one can doubt that.

  At the same time, a slight change in the camera angle or even the change of appearance will inevitably lead to an error. Bam — and your new haircut keeps you invisible for the cameras.

  So this is a really serious flaw. You cannot guarantee that a person will stand still and face the camera so the results will not always be correct. And if the person changes the appearance, it would be almost impossible to recognize them.

Topic discussion:

If I have seen further it is by standing on the shoulders of giants.

-Sir Isaac Newton

Metaphor which means "Using the understanding gained by major thinkers who have gonebefore in order to make intellectual progress"

Likewise, Sir Newton said, improvement is a day-to-day process. It's not about the person, the whole community is involved in a specific region. And here we are using the inprovent.

This section we should taking about the improvement in face recognition

Likewise, Sir Newton said, improvement is a day-to-day process. It's not about the person, the whole community is involved in a specific region. And here we are using the accompolishment.

In this section, we should be talking about the improvements that have taken place over time and about their accomplishers.

Human beings perform face recognition automatically every day and practically with no effort.

Although it sounds like a very simple task for us, it has proven to be a complex task for a computer, as it has many variables that can impair the accuracy of the methods, for example: illumination variation, low resolution, occlusion, amongst other.

In computer science, face recognition is basically the task of recognizing a person based on its facial image. It has become very popular in the last two decades, mainly because of the new methods developed and the high quality of the current videos/cameras.

Note that **face recognition** is different of **face detection**:

•**Face Detection**: it has the objective of finding the faces (location and size) in an image and probably extract them to be used by the face recognition algorithm.

•**Face Recognition**: with the facial images already extracted, cropped, resized and usually converted to grayscale, the face recognition algorithm is responsible for finding characteristics which best describe the image.

The face recognition systems can operate basically in two modes:

•**Verification or authentication of a facial image**: it basically compares the input facial image with the facial image related to the user which is requiring the authentication. It is basically a 1x1 comparison.

•**Identification or facial recognition**: it basically compares the input facial image with all facial images from a dataset with the aim to find the user that matches that face. It is basically a 1xN comparison.

There are different types of face recognition algorithms, for example:

- **Eigenfaces(1991)**
- **Local Binary Patterns Histograms (LBPH)(1996)**
- **Fisherfaces(1997)**
- **Scale Invariant Feature Transform (SIFT)(1999)**
- **Speed Up Robust Features (SURF)(2006)**

Woodrow Wilson Bledsoe

Father of facial-Recognition

The implementation of facial recognition has seen many iterations, which saw roots in the 1960s when facial recognition was manually implemented by Woodrow Wilson Bledsoe. Bledsoe is largely considered the father of facial recognition for developing a system that classified photos of faces through a RAND tablet, which was a graphical computer input device. With this device, Bledsoe manually recorded the coordinate locations of facial features such as a person’s mouth, nose, eyes, and even their hairline.

Equipped with a manual log of various faces, facial recognition then plotted new photographs against the database to identify individuals with the closest numerical resemblance based on the plotted information. While this served as a foundational basis, and proof that facial recognition was a viable biometric, it was severely hindered by the technology of the period, proving to have inadequate processing power to meet demanding computing rigors required to scale and refine the technology.

Facial recognition was incrementally refined throughout the 1970s by the likes of Goldstein, Harmon, and Lesk, but largely remained a manually computed process.

The leap from manual computation for facial recognition to a computer-assisted approach using Eigenfaces:

It wasn't until the late '80s to early '90s that significant developments would be made in the field of facial recognition in the form of an application of linear algebra. In what was to become known as the Eigenface approach, Sirovich and Kirby implemented an approach that started as a low-dimensional representation of facial images. Through this, they demonstrated that an analysis of features on a set of images could form a set of basic features. They also established that less than a hundred values were needed to accurately code a normalized image of a face.

The goal of this example is to show how an unsupervised method and a supervised one can be chained for better prediction. It starts with a didactic but lengthy way of doing things, and finishes with the idiomatic approach to pipelining in scikit-learn.

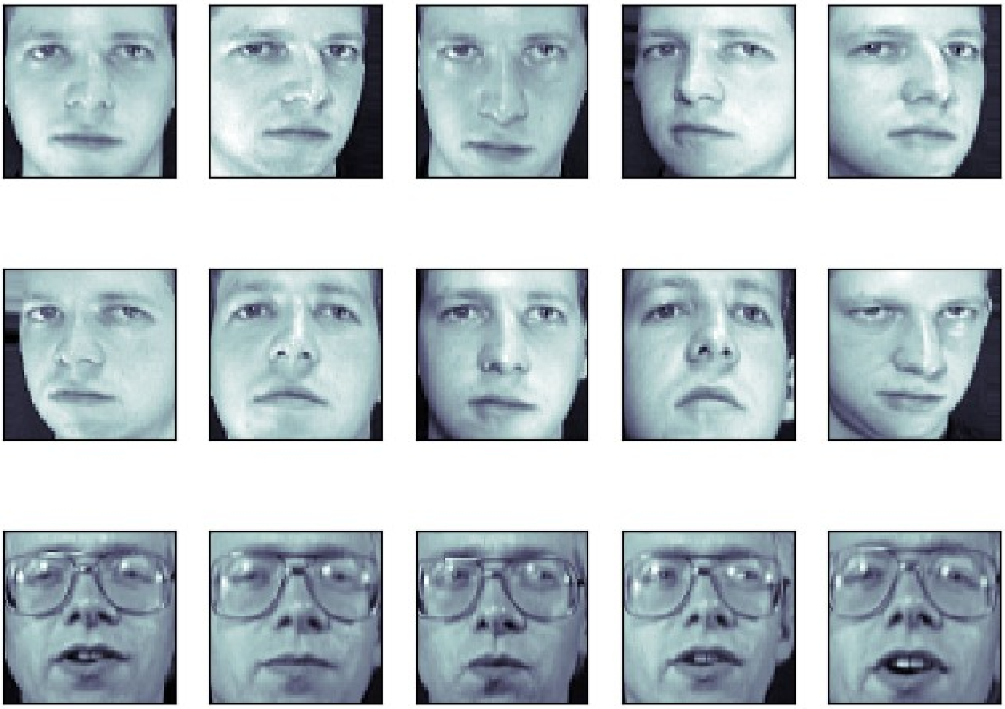
Here we'll take a look at a simple facial recognition example. Ideally, we would use a dataset consisting of a subset of the Labeled Faces in the Wild data that is available with

`sklearn.datasets.fetch_lfw_people()`

```
from sklearn import datasets

faces = datasets.fetch_olivetti_faces()

faces.data.shape
```

Let's visualize these faces to see what we're working with

```
from mathplotlib import pyplot as plt

fig = plt.figure(figsize=(8, 6))

# plot several images

for i in range(15):

    ax = fig.add_subplot(3, 5, i + 1, xticks=[], yticks=[])

    ax.imshow(faces.images[i], cmap=plt.cm.bone)
```



We’ll perform a Support Vector classification of the images. We’ll do a typical train-test split on the images:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(faces.data,
        faces.target, random_state=0)


print(X_train.shape, X_test.shape)
```

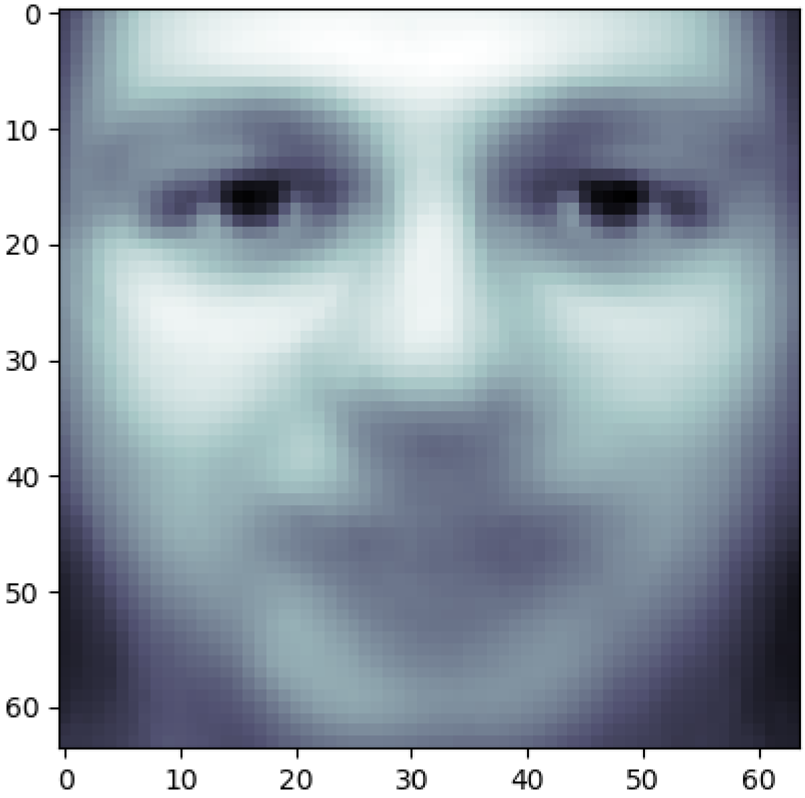
Preprocessing: Principal Component Analysis

1850 dimensions is a lot for SVM. We can use PCA to reduce these 1850 features to a manageable size, while maintaining most of the information in the dataset.

```
from sklearn import decomposition
pca = decomposition.PCA(n_components=150, whiten=True)
pca.fit(X_train)
```

One interesting part of PCA is that it computes the “mean” face, which can be interesting to examine:

```
opplt.imshow(pca.mean_.reshape(faces.images[0].shape),
          cmap=plt.cm.bone)
```



With this projection computed, we can now project our original training and test data onto the PCA basis:

```
X_train_pca = pca.transform(X_train)
X_test_pca = pca.transform(X_test)
print(X_train_pca.shape)
print(X_test_pca.shape)
```

These projected components correspond to factors in a linear combination of component images such that the combination approaches the original face.

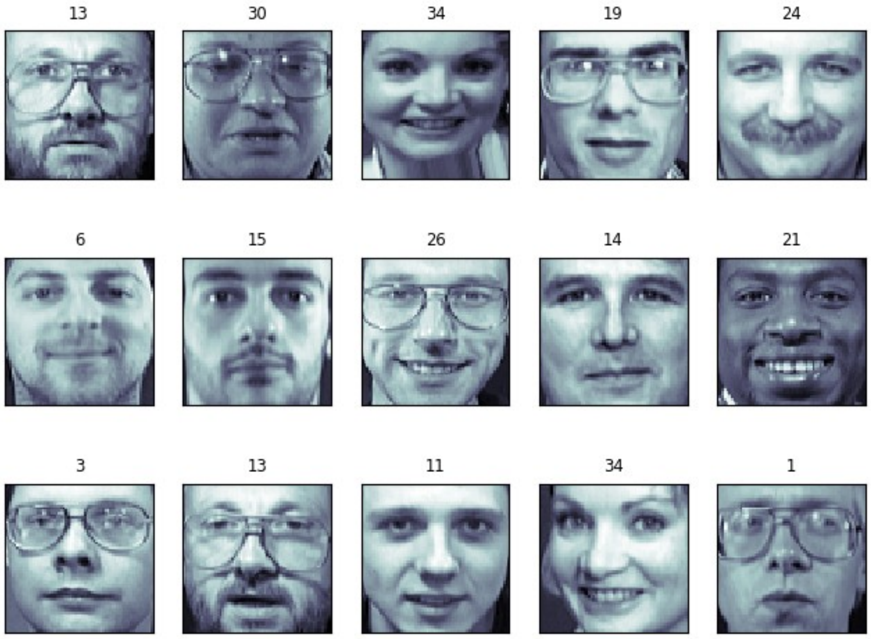
Doing the Learning: Support Vector Machines

Now we’ll perform support-vector-machine classification on this reduced dataset:

```
from sklearn import svm
clf = svm.SVC(C=5., gamma=0.001)
clf.fit(X_train_pca, y_train)
```

Finally, we can evaluate how well this classification did. First, we might plot a few of the test-cases with the labels learned from the training set:

```
import numpy as np
fig = plt.figure(figsize=(8, 6))
for i in range(15):
    ax = fig.add_subplot(3, 5, i + 1, xticks=[], yticks=[])
    ax.imshow(X_test[i].reshape(faces.images[0].shape),
              cmap=plt.cm.bone)
    y_pred = clf.predict(X_test_pca[i, np.newaxis])[0]
    color = ('black' if y_pred == y_test[i] else 'red')
    ax.set_title(y_pred, fontsize='small', color=color)
```



The classifier is correct on an impressive number of images given the simplicity of its learning model! Using a linear classifier on 150 features derived from the pixel-level data, the algorithm correctly identifies a large number of the people in the images.

Again, we can quantify this effectiveness using one of several measures from `sklearn.metrics`. First we can do the classification report, which shows the precision, recall and other measures of the “goodness” of the classification:

```
from sklearn import metrics

y_pred = clf.predict(X_test_pca)

print(metrics.classification_report(y_test, y_pred))
```

Out:

```
precision    recall  f1-score   support

          0       1.00      0.67      0.80         6
          1       1.00      1.00      1.00         4
          2       0.50      1.00      0.67         2
          3       1.00      1.00      1.00         1
          4       0.50      1.00      0.67         1
          5       1.00      1.00      1.00         5
          6       1.00      1.00      1.00         4
          7       1.00      0.67      0.80         3
          9       1.00      1.00      1.00         1
         10       1.00      1.00      1.00         4
         11       1.00      1.00      1.00         1
         12       1.00      1.00      1.00         2
         13       1.00      1.00      1.00         3
         14       1.00      1.00      1.00         5
         15       0.75      1.00      0.86         3
         17       1.00      1.00      1.00         6
         19       1.00      1.00      1.00         4
         20       1.00      1.00      1.00         1
         21       1.00      1.00      1.00         1
         22       1.00      1.00      1.00         2
         23       1.00      1.00      1.00         1
         24       1.00      1.00      1.00         2
         25       1.00      0.50      0.67         2
         26       1.00      0.75      0.86         4
         27       1.00      1.00      1.00         1
         28       0.67      1.00      0.80         2
         29       1.00      1.00      1.00         3
         30       1.00      1.00      1.00         4
         31       1.00      1.00      1.00         3
         32       1.00      1.00      1.00         3
         33       1.00      1.00      1.00         2
         34       1.00      1.00      1.00         3
         35       1.00      1.00      1.00         1
         36       1.00      1.00      1.00         3
```

```
          37       1.00      1.00      1.00         3
          38       1.00      1.00      1.00         1
          39       1.00      1.00      1.00         3
avg / total        0.97      0.95      0.95       100
```

Pipelining

Above we used PCA as a pre-processing step before applying our support vector machine classifier. Plugging the output of one estimator directly into the input of a second estimator is a commonly used pattern; for this reason scikit-learn provides a `Pipeline` object which automates this process. The above problem can be re-expressed as a pipeline as follows:

```
from sklearn.pipeline import Pipeline
clf = Pipeline([('pca', decomposition.PCA(n_components=150, whiten=True)),
               ('svm', svm.LinearSVC(C=1.0))])

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)
print(metrics.confusion_matrix(y_pred, y_test))
plt.show()
```

Out:

```
[[2 0 0 ... 0 0 0]
 [0 4 0 ... 0 0 0]
 [0 0 1 ... 0 0 0]
 ...
 [0 0 0 ... 3 0 0]
 [0 0 0 ... 0 1 0]
 [0 0 0 ... 0 0 3]]
```

The Eigenface method is today used as a basis of many deep learning algorithms, paving way for modern facial recognition solutions .

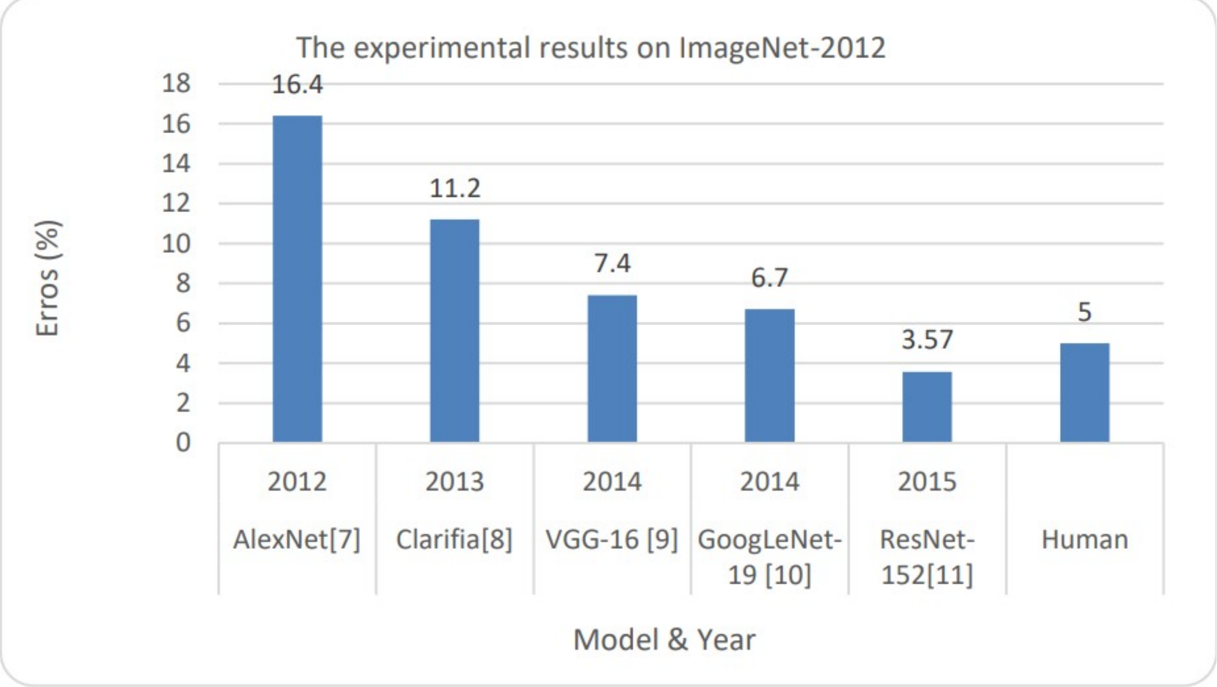
The modern-day game-changers spurred on by the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

ImageNet is essentially a democratized dataset that can be used for machine learning research. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a yearly challenge that exists to evaluate the ability of algorithms to correctly classify images within its repository to increasing degrees of accuracy.

In the 2010s, a good classification error rate was around 25%. In 2012, AlexNet, which was a deep convolutional neural net (CNN) bested that result by getting an error rate of 15.3%. This was a game-changer because it was the first time that such results were achieved, beating competing algorithms in that year by over 10.8%.

AlexNet went on to become the winning entry in the ILSVRC in that year.

Subsequent image processing solutions in the following years improved on results of AlexNet. In 2013, ZFNet, also a CNN, achieved an error rate of 14.8%. In 2014, GooLeNet/Inception achieved an error rate of 6.67%. In 2015, ResNet further brought the error rate down to 3.6%.



Graph showing the progression of the accuracy of image recognition algorithms. Image from ARXIV

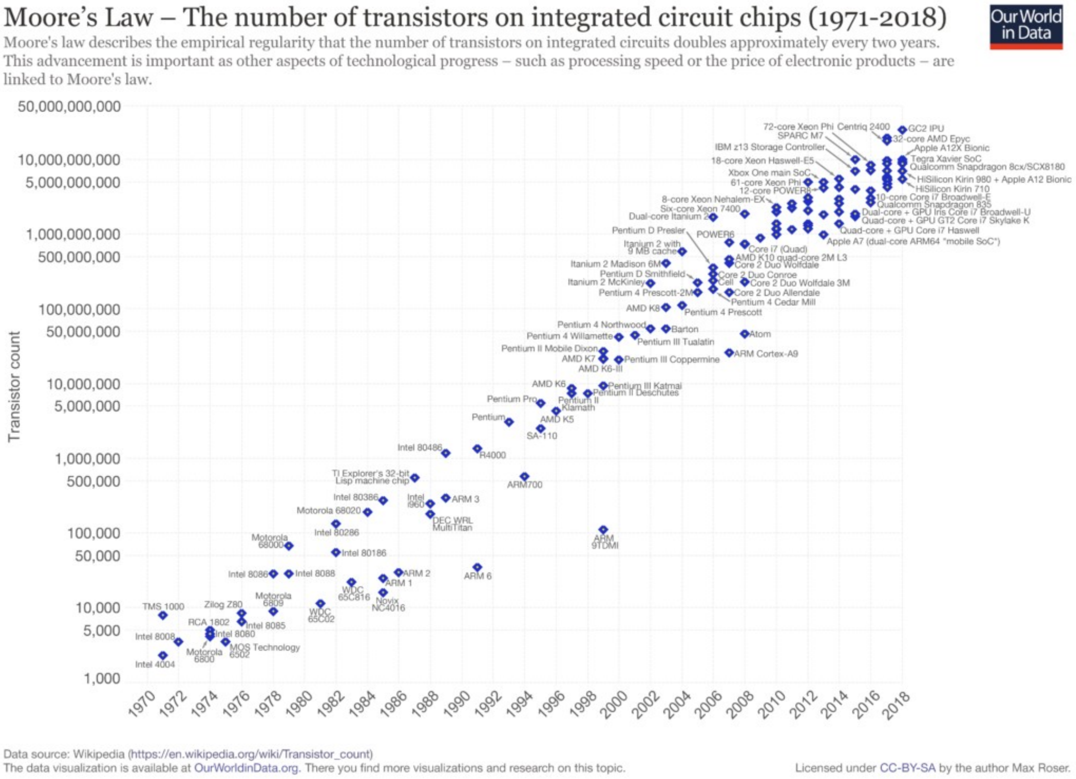
With this, machines could theoretically detect and classify images—albeit based on a set image database, and without the ability to contextualize the image—as good as, or better than human beings.

> Computer processing of images has become so progressively powerful, in no small part thanks to AlexNet. Today, machines can technically identify images to a higher degree of accuracy than a human can.

The strides we've made in recent history can be attributed to the changing approach in image processing. Researchers gradually moved away from human coding techniques and presets, moving onto the utility of deep neural networks and machine learning. This meant that image processing, identification, and classification has led to unparalleled levels of accuracy we have today, that we can apply to facial recognition.

The role of Moore's Law in image processing

In the year that AlexNet won ILSVRC, it used two Nvidia Geforce GTX 580 GPUs to process and achieve those results. While the approach of how images are processed differently keeps getting refined, the improving efficiency and power of graphical processing units (GPUs) also increase the efficacy of how images are processed—and in extension, how faces are detected.



Moore's Law – The number of transistors on integrated circuit chips (1971-2018)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are linked to Moore's law.

Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Licensed under CC-BY-SA by the author Max Roser.

According to Moore's Law, the number of transistors per unit of surface doubles every 2 years. Image from Wikimedia foundation.

With Moore’s law, which has rightly predicted since 1975, that the number of transistors we can cram into the same amount of space doubles. This means double the raw computing power than the years prior, which follows an exponential curve.

Advanced facial recognition technology is spurred on by breakthroughs in image processing

Turns out, how we achieved the advanced telematics capabilities today—where machines can identify images and faces better than humans—essentially boils down to two key factors:

•The exponential increase of computing resources at the same cost.

•Incremental strides from research labs with how we process images.

To surmise the points above, we have the following explainer video that will give you an overview of how rapidly facial recognition is evolving:

Face recognition is among the most productive image processing applications and has a pivotal role in the technical field. Recognition of the human face is an active issue for authentication purposes specifically in the context of attendance of students. Attendance system using face recognition is a procedure of recognizing students by using face biostatistics based on the high definition monitoring and other computer technologies. The development of this system is aimed to accomplish digitization of the traditional system of taking attendance by calling names and maintaining pen-paper records. Present strategies for taking attendance are tedious and time-consuming. Attendance records can be easily manipulated by manual recording. The traditional process of making attendance and present biometric systems are vulnerable to proxies. This paper is therefore proposed to tackle all these problems. The proposed system makes the use of Haar classifiers, KNN, CNN, SVM, Generative adversarial networks, and Gabor filters. After face recognition attendance reports will be generated and stored in excel format. The system is tested under various conditions like illumination, head movements, the variation of distance between the student and cameras. After vigorous testing overall complexity and accuracy are calculated. The Proposed system proved to be an efficient and robust device for taking attendance in a classroom without any time consumption and manual work. The system developed is cost-efficient and need less installation.

Each method has a different approach to extract the image information and perform the matching with the input image. However, the methods Eigenfaces and Fisherfaces have a similar approach as well as the SIFT and SURF methods.

Today we gonna talk about one of the oldest (not the oldest one) and more popular face recognition algorithms:**Local Binary Patterns Histograms (LBPH)**.

Objective

The objective of this post is to explain the **LBPH** as simple as possible, showing the method step-by-step.

As it is one of the easier face recognition algorithms I think everyone can understand it without major difficulties.

Introduction

> ***Local Binary Pattern*** *(LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.*

It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

Using the LBP combined with histograms we can represent the face images with a simple data vector.

As LBP is a visual descriptor it can also be used for face recognition tasks, as can be seen in the following step-by-step explanation.

The LBP feature vector, in its simplest form, is created in the following manner:

- Divide the examined window into cells (e.g. 16x16 pixels for each cell).
- For each pixel in a cell, compare the pixel to each of its 8 neighbors (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise.
- Where the center pixel's value is greater than the neighbor's value, write "0". Otherwise, write "1". This gives an 8-digit binary number (which is usually converted to decimal for convenience).
- Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are greater than the center). This histogram can be seen as a 256-dimensional feature vector.

•Optionally normalize the histogram.

•Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window.

The feature vector can now be processed using the Support vector machine, extreme learning machines, or some other machine learning algorithm to classify images. Such classifiers can be used for face recognition or texture analysis.

A useful extension to the original operator is the so-called uniform pattern,which can be used to reduce the length of the feature vector and implement a simple rotation invariant descriptor. This idea is motivated by the fact that some binary patterns occur more commonly in texture images than others. A local binary pattern is called uniform if the binary pattern contains at most two 0-1 or 1-0 transitions. For example, 00010000 (2 transitions) is a uniform pattern, but 01010100 (6 transitions) is not. In the computation of the LBP histogram, the histogram has a separate bin for every uniform pattern, and all non-uniform patterns are assigned to a single bin. Using uniform patterns, the length of the feature vector for a single cell reduces from 256 to 59. The 58 uniform binary patterns correspond to the integers 0, 1, 2, 3, 4, 6, 7, 8, 12, 14, 15, 16, 24, 28, 30, 31, 32, 48, 56, 60, 62, 63, 64, 96, 112, 120, 124, 126, 127, 128, 129, 131, 135, 143, 159, 191, 192, 193, 195, 199, 207, 223, 224, 225, 227, 231, 239, 240, 241, 243, 247, 248, 249, 251, 252, 253, 254 and 255.

Note: you can read more about the LBPH

here:http://www.scholarpedia.org/article/Local_Binary_Patterns

Step-by-Step

Now that we know a little more about face recognition and the LBPH, let's go further and see the steps of the algorithm:

1. **Parameters**: the LBPH uses 4 parameters:

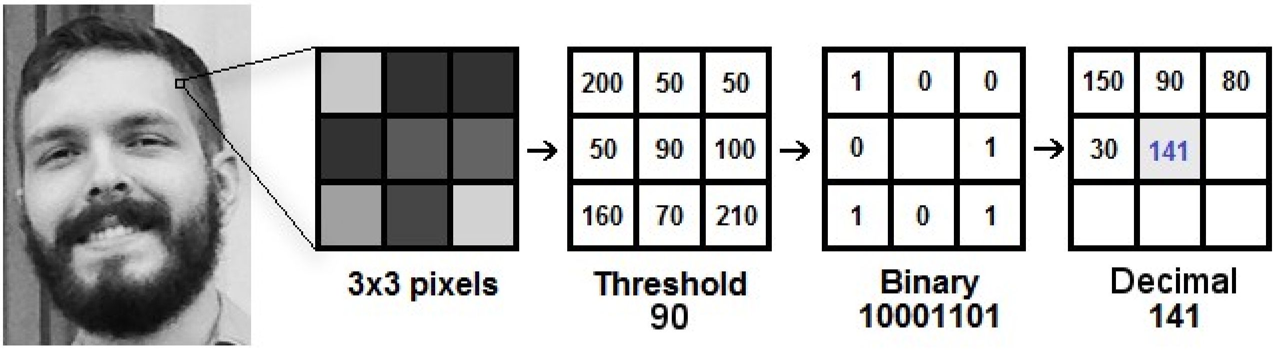
- **Radius**: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.
- **Neighbors**: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- **Grid X**: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- **Grid Y**: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

Don’t worry about the parameters right now, you will understand them after reading the next steps.

2.Training the Algorithm: First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let’s see the LBPH computational steps.

3. Applying the LBP operation: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters **radius** and **neighbors**.

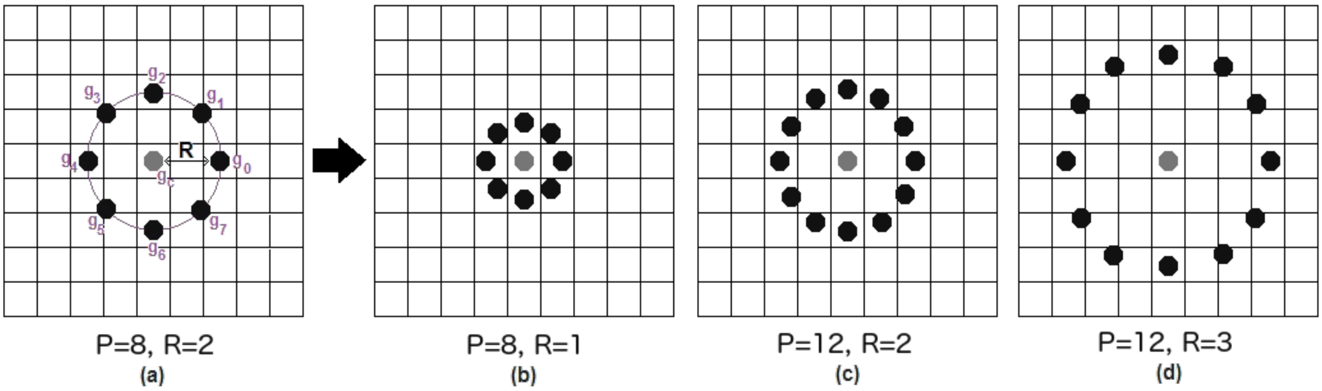
The image below shows this procedure:



Based on the image above, let’s break it into several small steps so we can understand it easily:

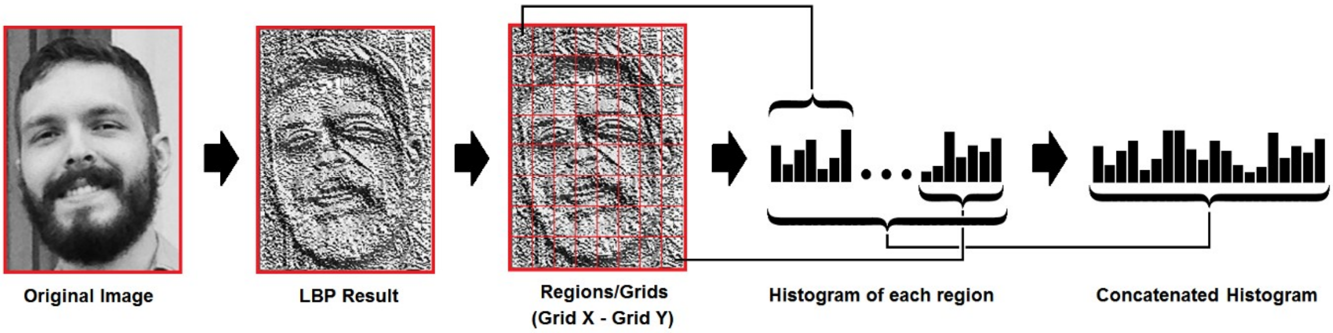
- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
- Then, we need to take the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors.
- For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same.
- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.

- **Note**: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.



It can be done by using **bilinear interpolation**. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

4. Extracting the Histograms: Now, using the image generated in the last step, we can use the **Grid X** and **Grid Y** parameters to divide the image into multiple grids, as can be seen in the following image:



Based on the image above, we can extract the histogram of each region as follows:

- As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.
- Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final histogram. The final histogram represents the characteristics of the image original image.

The LBPH algorithm is pretty much it.

5. Performing the face recognition: In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image.

  - So to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.

- We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: **euclidean distance**, **chi-square**, **absolute value**, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula:

$$D = \sqrt{\sum_{i=1}^{n} (hist1_i - hist2_i)^2}$$

- So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a '**confidence**' measurement.**Note**: don't be fooled about the 'confidence' name, as lower confidences are better because it means the distance between the two histograms is closer.

- We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

Conclusions

- LBPH is one of the easiest face recognition algorithms.
- It can represent local features in the images.
- It is possible to get great results (mainly in a controlled environment).
- It is robust against monotonic gray scale transformations.
- It is provided by the OpenCV library (Open Source Computer Vision Library).

LBPH algorithm

I am implementing the LBPH algorithm in Go programming language. The project is available on Github and is distributed under the MIT license, so feel free to contribute to the project (any contributions are welcome).
Link to the project: https://github.com/kelvins/lbph

Note: as mentioned in the conclusions, the **LBPH** is also provided by the **OpenCV library**. The OpenCV library can be used by many programming languages (e.g. C++, Python, Ruby, Matlab).

Fisherfaces

One way to represent the input data is by finding a subspace which represents most of the data variance. This can be obtained with the use of Principal Components Analysis (PCA). When applied to face images, PCA yields a set of eigenfaces. These eigenfaces are the eigenvectors associated to the largest eigenvalues of the covariance matrix of the training data. The eigenvectors thus found correspond to the least-squares (LS) solution. This is indeed a powerful way to represent the data because it ensures the data variance is maintained while eliminating unnecessary existing correlations among the original features (dimensions) in the sample vectors.

When the goal is classification rather than representation, the LS solution may not yield the most desirable results. In such cases, one wishes to find a subspace that maps the sample vectors of the same class in a single spot of the feature representation and those of different classes as far apart from each other as possible. The techniques derived to achieve this goal are known as discriminant analysis (DA).

The most known DA is Linear Discriminant Analysis (LDA), which can be derived from an idea suggested by R.A. Fisher in 1936. When LDA is used to find the subspace representation of a set of face images, the resulting basis vectors defining that space are known as Fisherfaces.

Computing the Fisherfaces:

The theoretical argument given in the preceding section shows how to obtain the Bayes optimal solution for the 2-class homoscedastic case. In general, we will have more than 2-classes. In such a case, we reformulate the above stated problem as that of minimizing within-class differences and maximizing between-class distances.

Within class differences can be estimated using the within-class scatter matrix, given by

$$\mathbf{S}_w = \sum_{j=1}^{C} \sum_{i=1}^{n_j} (\mathbf{x}_{ij} - \mu_j)(\mathbf{x}_{ij} - \mu_j)^T,$$

where *xij* is the *i* th sample of class *j* , *μj* is the mean of class *j* ,and *nj* the number of samples in class *j* .

Likewise, the between class differences are computed using the between-class scatter matrix,

$$\mathbf{S}_b = \sum_{j=1}^{C} (\mu_j - \mu)(\mu_j - \mu)^T,$$

where *μ* represents the mean of all classes.

We now want to find those basis vectors V where Sw is minimized and Sb is maximized, where V is a matrix whose columns vi are the basis vectors defining the subspace. These are given by,

$$\frac{|\mathbf{V}^T\mathbf{S}_b\mathbf{V}|}{|\mathbf{V}^T\mathbf{S}_w\mathbf{V}|}.$$

The solution to this problem is given by the generalized eigenvalue decomposition

$$\mathbf{S}_b\mathbf{V} = \mathbf{S}_w\mathbf{V}\mathbf{\Lambda},$$

where **V** is (as above) the matrix of eigenvectors and **Λ** is a diagonal matrix of corresponding eigenvalues.

The eigenvectors of **V** associated to non-zero eigenvalues are the Fisherfaces. There is a maximum of $C-1$ Fisherfaces. This can be readily seen from the definition of **S***b*. Note that in our definition, **S***b* is a combination of C feature vectors. Any C vectors define a subspace of $C-1$ or less dimensions. The equality holds when these vectors are linearly independent from one another. the first four Fisherfaces obtained when using the defined algorithm on a set of frontal face image of 100 different subjects. Images were selected to have a neutral expression.

Technicalities

To obtain the Fisherfaces, we need to compute the inverse of **S***w* ,i.e.,**S**−1*w*.If the sample feature vectors are defined in a *p*-dimensional space and *p* is larger than the total number of samples *n* , then **S***w* is singular. There are two typically used solutions to this problem. In the first solution, we project the sample vectors (or, equivalently, the between- and within-class scatter matrices) onto the PCA space of *r* dimensions, with $r \leq rank$(**S***w*) and compute the Fisherfaces in this PCA space. The second solution is to add a regularising term to **S***w* . That is, **S***w*+ϵ **I** , where **I** is the identity matrix and ϵ is a small constant.

One can also substitute the between- and within-class scatter matrices for other measurements that do a similar job. First, note that these two matrices are symmetric and positive semi-definite. Hence, each defines a metric. This means we can substitute these matrices with others as far as they define metrics whose goals are to minimize within-class variance and maximize between-class distances. For example, we can substitute the within-class scatter matrix for the sample covariance matrix.

The Fisherfaces obtained with the approach described thus far are based on the linear assumption mentioned above. This assumption holds when the classes are homoscedastic

Normals. In general, this assumption is violated. To resolve this problem, one can add another metric into the equation. The goal of this new metric is to map the original heteroscedastic (meaning with different covariances) problem into a homoscedastic one. This mapping function can be converted into a kernel mapping thanks to the Representer's Theorem. And, the metric is given by the Gram (or Kernel) matrix. For this reason, this alternative method is called Kernel LDA (or, KLDA for short). Several authors have also defined heteroscedastic measures of within- and between-class differences. Yet, another alternative to lessen the Normal assumption is to represent the samples in each class as a mixture of Normal distributions. In this approach, the trick is how to determine the number of mixtures per class. A popular solution is given by the algorithm Subclass Discriminant Analysis (SDA) and its kernel extension KSDA. Kernel methods are generally preferred when the number of training samples is sufficiently large to facilitate the learning of the non-linear mapping.

SIFT (Scale Invariant Feature Transform) is a feature detection algorithm in computer vision to detect and describe local features in images. It was created by David Lowe from the University British Columbia in 1999. David Lowe presents the SIFT algorithm in his original paper titled Distinctive Image Features from Scale-Invariant Keypoints.

Image features extracted by SIFT are reasonably invariant to various changes such as their llumination image noise, rotation,scaling, and small changes in viewpoint.
There are four main stages involved in SIFT algorithm :
1.Scale-space extrema detection
2.Keypoint localization
3.Orientation Assignment
4.Keypoint descriptor
We will now examine these stages in detail.

SIFT Algorithm : Explained

1. Scale-space extrema detection

Before going into this, we will visit the idea of scale space theory and then, see how it has been used in SIFT.

Scale-space

Scale-space theory is a framework for multiscale image representation, which has been developed by the computer vision community with complementary motivations from physics and biologic vision. The idea is to handle the multiscale nature of real-world objects, which implies that objects may be perceived in different ways depending on the scale of observation.

Scale-space in SIFT

The first stage is to identify locations and scales that can be repeatably assigned under differing views of the same object. Detecting locations that are invariant to scale change of the image can be accomplished by searching for stable features across all possible scales, using a continuous function of scale known as scale space.
The scale space is defined by the function:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

Where:

●L is the blurred image
●G is a Gaussian blur operator
●I is the input image
●σ acts as a scale parameter (Higher value results in more blur)

So, we first take the original image and blur it using a Gaussian convolution. What follows is a sequence of further convolutions with increasing standard deviation(σ). Images of same size (with different blur levels) are called an Octave. Then, we downsize the original image by a factor of 2. This starts another row of convolutions. We repeat this process until the pictures are too small to proceed.

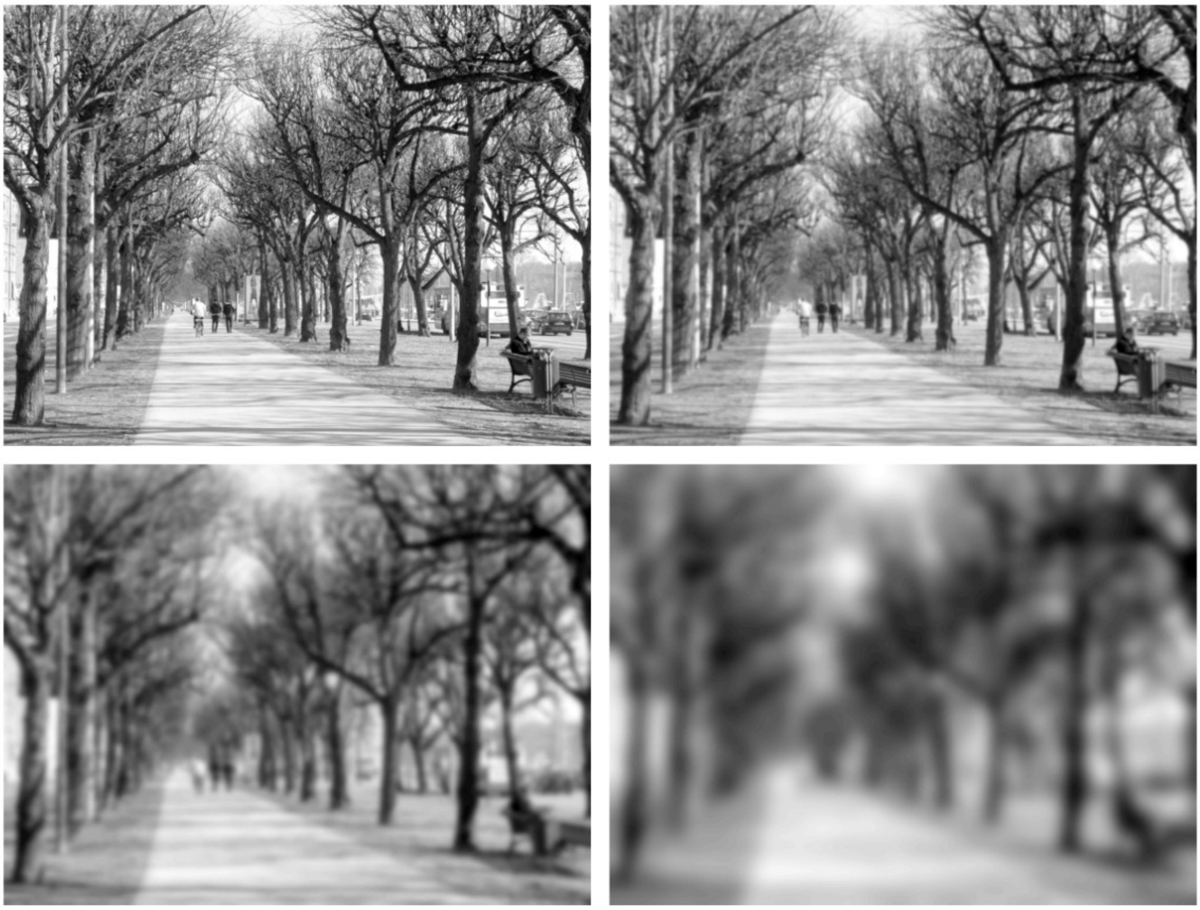


Figure 1 : (top left) A grey-level image and the scale-space representations computed at scale levels t = 1,8 and 64.

Image from:
Encyclopedia of Computer Science and Engineering (Benjamin Wah, ed), John Wiley and Sons, Volume IV, pages 2495–2504, Hoboken, New Jersey, 2009.
Now we have consructed a scale space. We do this to handel the multiscale nature of real-world objects.

Laplacian of Gaussian (LoG) approximations

Since we are finding the most stable image features we consider Lapcian of Gaussian. In detailed experimental comparisons, Mikolajczyk (2002) found that maxima and minima of Laplacian of Gaussian produce the most stable image features compared to a range of other possible image functions, such as the gradient, Hessian, or Harris corner function.
The problem that occurs here is that calculating all those second order derivatives is computationally intensive so we use Difference of Gaussians which is an approximation of LoG. Difference of Gaussian is obtained as the difference of Gaussian blurring of an image with two different σ and is given by
$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)$
$= L(x, y, k\sigma) - L(x, y, \sigma)$.

It is represented in below image:

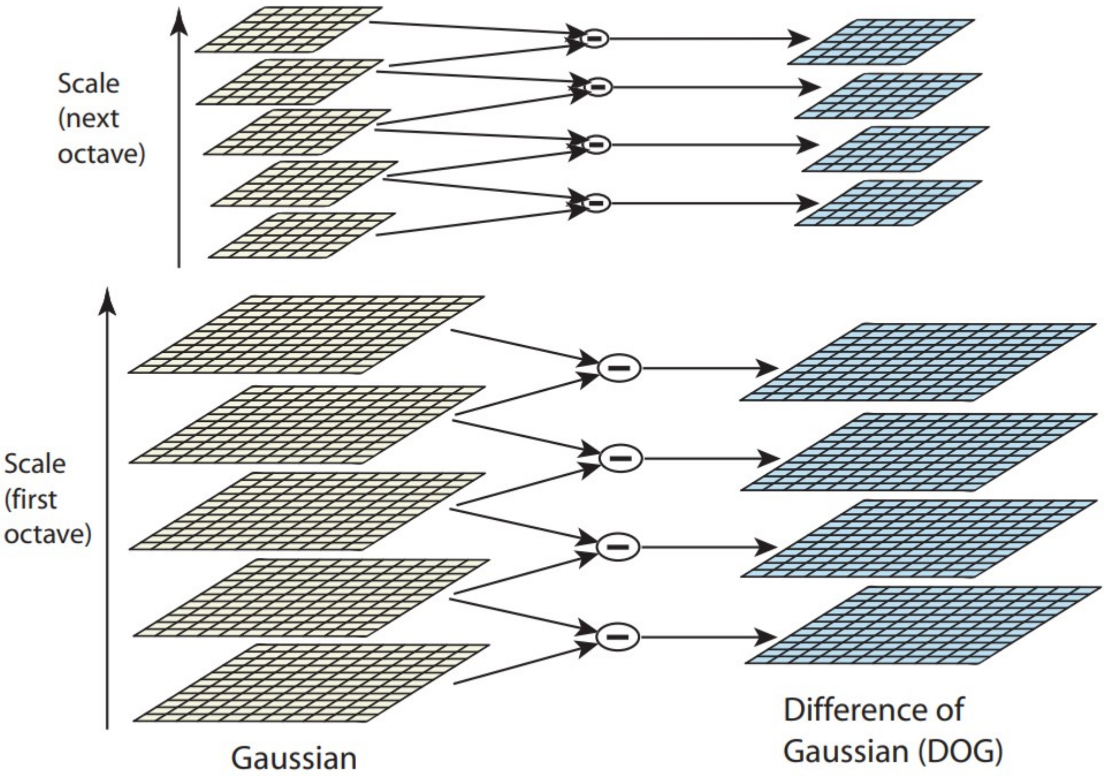


Figure 2 : For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

Image from:
Lowe, David G. "Distinctive image features from scale-invariant keypoints." International journal of computer vision 60.2 (2004): 91-110.
This is done for all octaves. The resulting images are an approximation of scale invariant laplacian of gaussian (which produces stable image keypoints).

2. Keypoint Localization

Now that we have found potential keypoints, we have to refine it further for more accurate results.

Local maxima/minima detection

The first step is to locate the maxima and minima of Difference of Gaussian(DoG) images. Each pixel in the DoG images is compared to its 8 neighbors at the same scale, plus the 9 corresponding neighbors at neighboring scales. If the pixel is a local maximum or minimum, it is selected as a candidate keypoint.

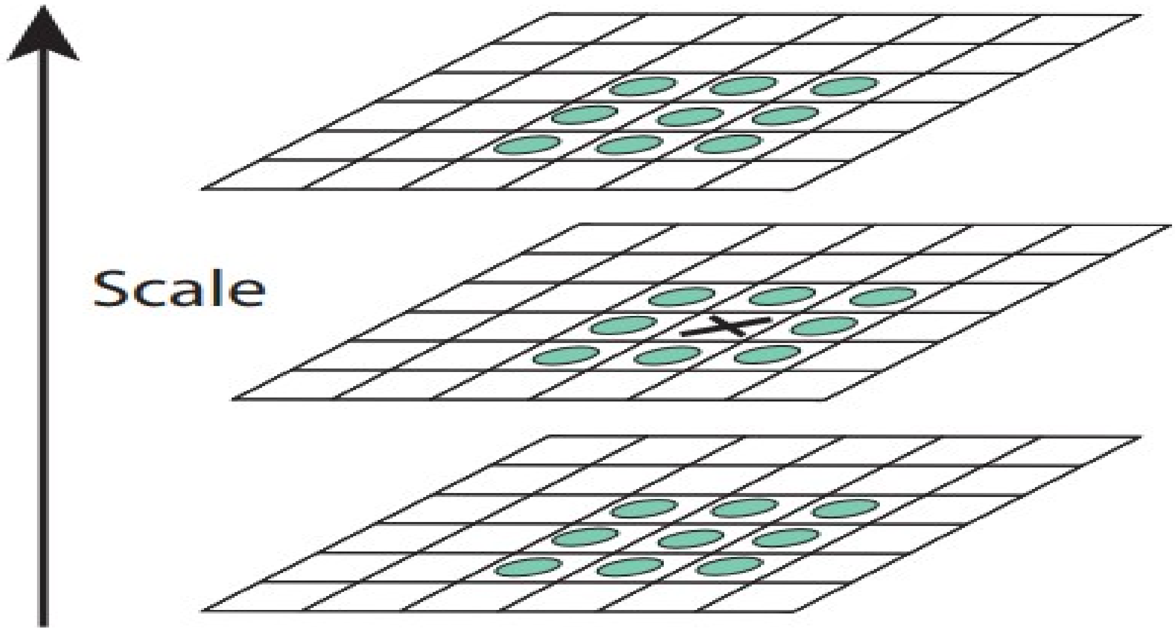


Figure 3 : Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26
neighbors in 3 × 3 regions at the current and adjacent scales (marked with circles).

Image from:
Lowe, David G. "Distinctive image features from scale-invariant keypoints." International journal of computer vision 60.2 (2004): 91-110.

Once a keypoint candidate has been found by comparing a pixel to its neighbors, the next step is to refine the location of these feature points to sub-pixel accuracy whilst simultaneously removing any poor features.

Sub-Pixel Refinement

The sub-pixel localization proceeds by fitting a 3D quadratic function to the local sample points to determine the interpolated location of the maximum. This approach uses the Taylor expansion (up to the quadratic terms) of the scale-space function, D(x, y, σ), shifted so that the origin is at the sample point:

$$D(\mathbf{x}) = D + \frac{\partial D}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^\mathbf{T} \frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

The location of the extremum, x̂ , is determined by taking the derivative of this function with respect to x and setting it to zero, giving:

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1} \frac{\partial D}{\partial \mathbf{x}}.$$

On solving, we'll get subpixel key point locations. Now we need to remove keypoints which have low contranst or lie along the edge as they are not useful to us.

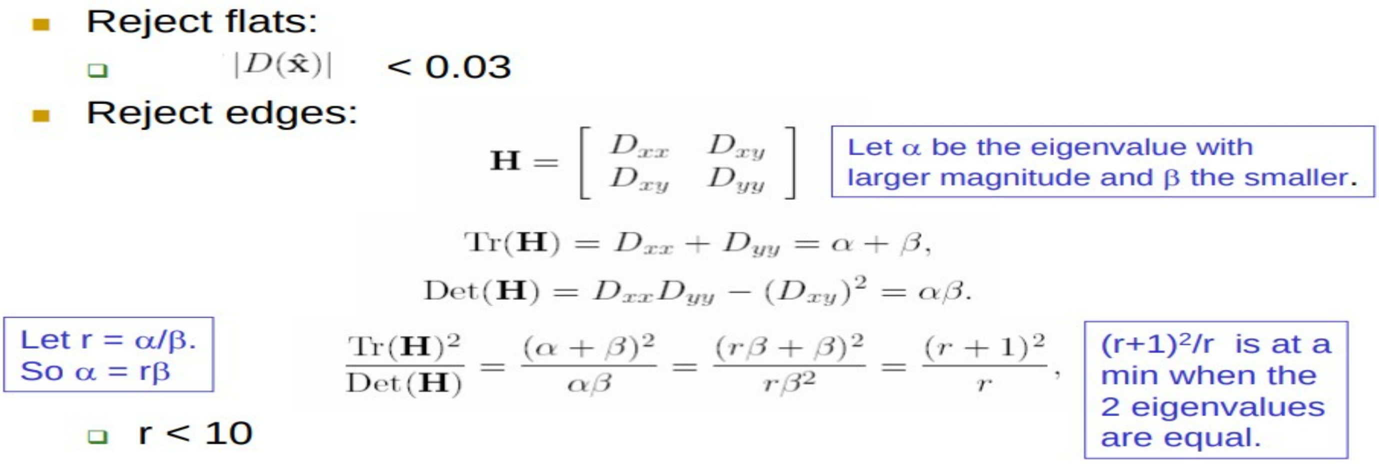
Removing Low Contrast Keypoints

The function value at the extremum, D(x̂), is useful for rejecting unstable extrema with low contrast. This can be obtained by substituting extremum x̂ into the Taylor Expansion (upto quadratic terms) as given above, giving:

$$D(\hat{\mathbf{x}}) = D + \frac{1}{2} \frac{\partial D}{\partial \mathbf{x}}^T \hat{\mathbf{x}}.$$

Removing Edge Responses

This is achieved by using a 2x2 Hessian matrix (H) to compute the principal curvature. A poorly defined peak in the difference-of-Gaussian function will have a large principal curvature across the edge but a small one in the perpendicular direction.



- Reject flats:
  - $|D(\hat{\mathbf{x}})|$ < 0.03
- Reject edges:

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

Let α be the eigenvalue with larger magnitude and β the smaller.

$$\mathrm{Tr}(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta,$$
$$\mathrm{Det}(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta.$$

Let r = α/β. So α = rβ

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r},$$

$(r+1)^2/r$ is at a min when the 2 eigenvalues are equal.

  - r < 10

Image source: https://courses.cs.washington.edu/courses/cse455/10au/notes/SIFT.ppt
So from the calculation from hessian matrix we reject the flats and edges and keep the corner keypoints.

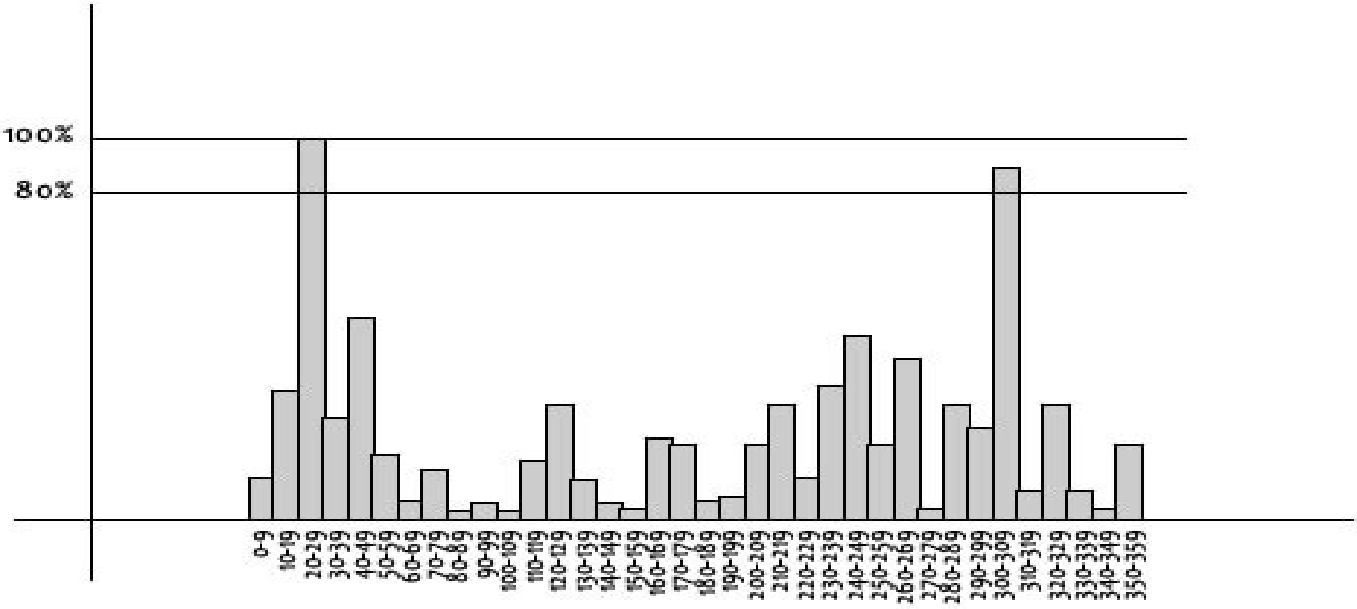
3. Keypoint Orientation Assignment

To determine the keypoint orientation, a gradient orientation histogram is computed in the neighborhood of the keypoint.
The magnitude and orientation is calculated for all pixels around the keypoint. Then, a histogram with 36 bins covering 360 degrees is created.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Magnitude m(x,y) and oreintation θ(x,y) of the pixel at x,y location.
Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ that is 1.5 times that of the scale of the keypoint.



Histogram covering 360 degrees (36 bins).

Image Source:
http://aishack.in/tutorials/sift-scale-invariant-feature-transform-keypoint-orientation.
When it is done for all the pixels around the keypoint, the histogram will have a peak at some point. And any peaks above 80% of the highest peak are converted into a new keypoint. This new keypoint has the same location and scale as the original. But it's orientation is equal to the other peak.

4. Keypoint Descriptor

Once a keypoint orientation has been selected, the feature descriptor is computed as a set of orientation histograms.
To do this, a 16x16 window around the keypoint is taken. It is divided into 16 sub-blocks of 4x4 size.
Within each 4x4 window, gradient magnitudes and orientations are calculated. These orientations are put into an 8 bin histogram.
Histograms contain 8 bins each, and each descriptor contains an array of 4 histograms around the keypoint. This leads to a SIFT feature vector with 4 × 4 × 8 = 128 elements.

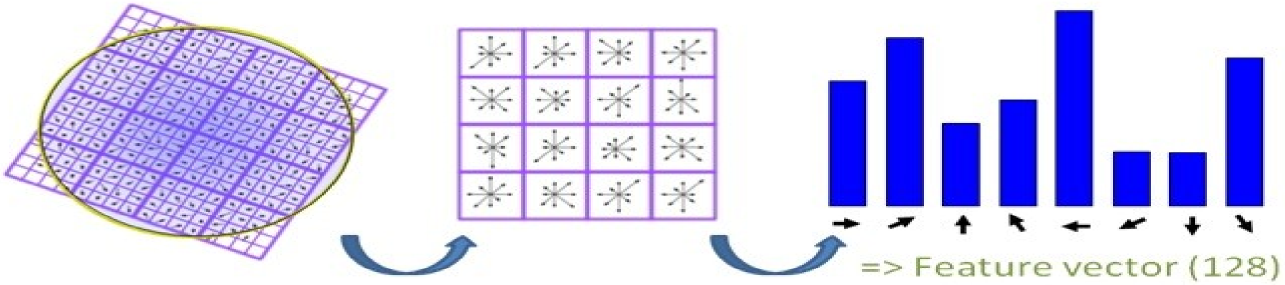


Image Source:
https://gilscvblog.com/2013/08/18/a-short-introduction-to-descriptors/
This feature vector introduces a few complications.
●Since we use gradient orientations, if you rotate the image, all gradient orientations also change. To solve this we subtract the keypoint's rotation from each orientation. Thus each gradient orientation is relative to the keypoint's orientation.
●We also normalize the vector to enhance invariance to changes in illumination. So we threshold the values in the feature vector to each be no larger than 0.2 (i.e. if value larger than 0.2 then it is set to 0.2).

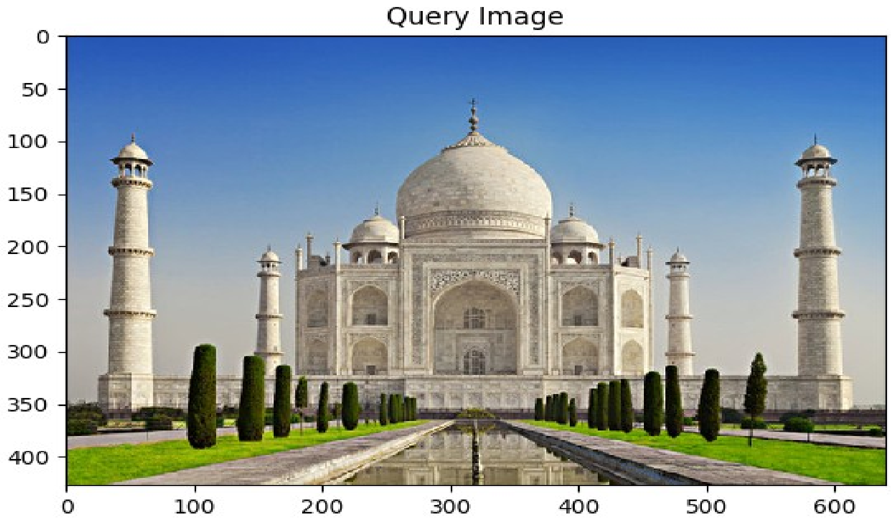
SIFT Implementation

In this section we will be performing object detection using SIFT with the help of opencv library in python.
Now before starting object detection let's first see the keypoint detection.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

train_img = cv2.imread('train.jpg')   # train image
query_img = cv2.imread('query.jpg')   # query/test image
```

```
# Turn Images to grayscale
def to_gray(color_img):
    gray = cv2.cvtColor(color_img, cv2.COLOR_BGR2GRAY)
    return gray

train_img_gray = to_gray(train_img)
query_img_gray = to_gray(query_img)

# Initialise SIFT detector
sift = cv2.xfeatures2d.SIFT_create()

# Generate SIFT keypoints and descriptors
train_kp, train_desc = sift.detectAndCompute(train_img_gray, None)
query_kp, query_desc = sift.detectAndCompute(query_img_gray, None)

plt.figure(1)
plt.imshow((cv2.drawKeypoints(train_img_gray, train_kp, train_img.copy())))
plt.title('Train Image Keypoints')

plt.figure(2)
plt.imshow((cv2.drawKeypoints(query_img_gray, query_kp, query_img.copy())))
plt.title('Query Image Keypoints')

plt.show()
```

Here I took pictures of Taj Mahal from different viewpoints for train and query image.



As you can see from the above code that the following function :

```
# Initialise SIFT detector
sift = cv2.xfeatures2d.SIFT_create()

# Generate SIFT keypoints and descriptors
train_kp, train_desc = sift.detectAndCompute(train_img_gray, None)
query_kp, query_desc = sift.detectAndCompute(query_img_gray, None)
```

is the one that does the computations for the SIFT algorithm and returns keypoints and descriptors of the image.

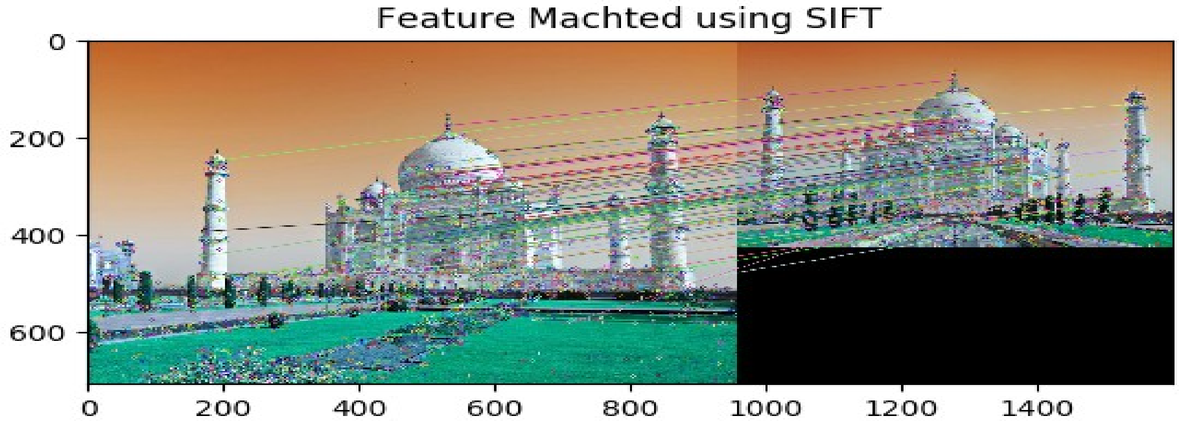
We can use the keypoints and descriptors for feature matching between two objects and finally find object in the query image.

Now we move onto feature maching part. We will match features in one image with others. For feature matching we are using Brute-Force matcher provided by OpenCV. You can also use FLANN Matcher in OpenCV as I wil use in further section of the tutorial.

Brute-Force matcher takes the descriptor of one feature in first set and is matched with all other features in second set using some distance calculation. And the closest one is returned.

```
# create a BFMatcher object which will match up the SIFT features
bf = cv2.BFMatcher(cv2.NORM_L2, crossCheck=True)

matches = bf.match(train_desc, query_desc)

# Sort the matches in the order of their distance.
matches = sorted(matches, key = lambda x:x.distance)

# draw the top N matches
N_MATCHES = 100

match_img = cv2.drawMatches(
    train_img, train_kp,
    query_img, query_kp,
    matches[:N_MATCHES], query_img.copy(), flags=0)

plt.figure(3)
plt.imshow(match_img)
plt.show()
```

Vsualization of how the SIFT features match up each other across the two images is as follow:

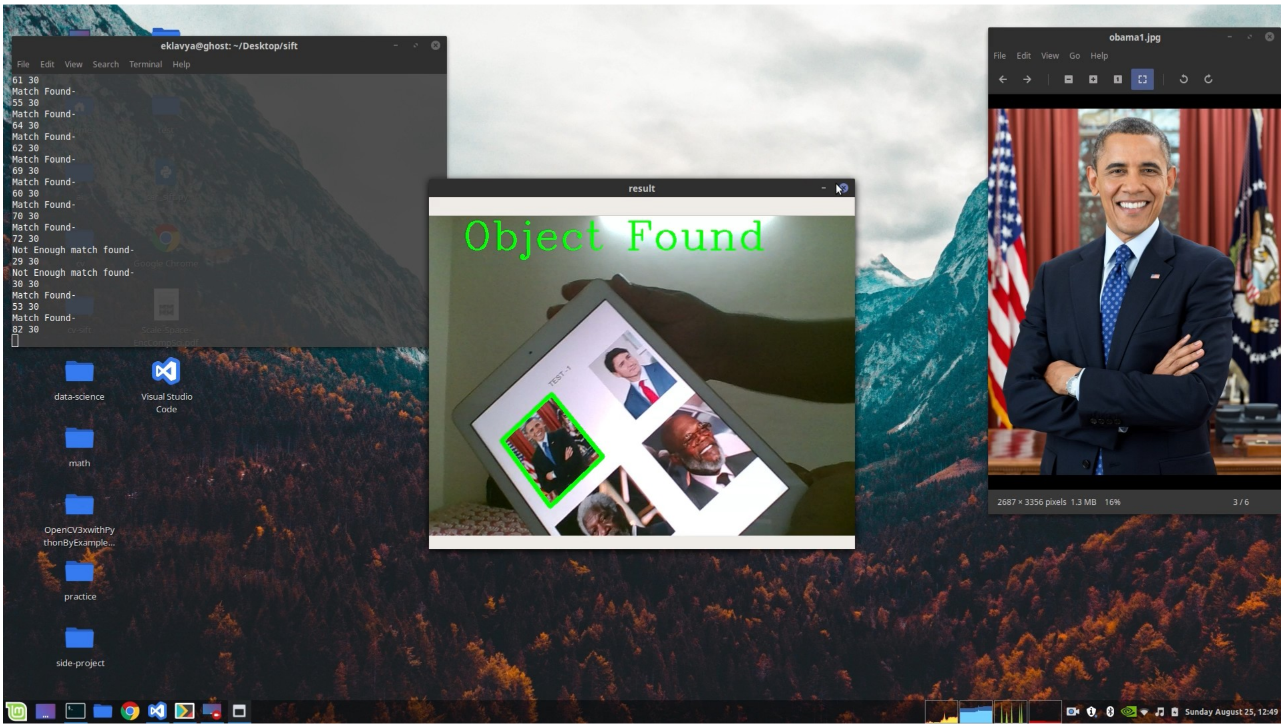


So till now we have found the keypoints and descriptors for the train and query images and then matched top keypoints and visualized it. But this is still not sufficient to find the object. For that, we can use a function cv2.findHomography(). If we pass the set of points from both the images, it will find the perpective transformation of that object. Then we can use cv2.perspectiveTransform() to find the object. It needs atleast four correct points to find the transformation.
So now we will use a train image and then try to detect it in the real-time.

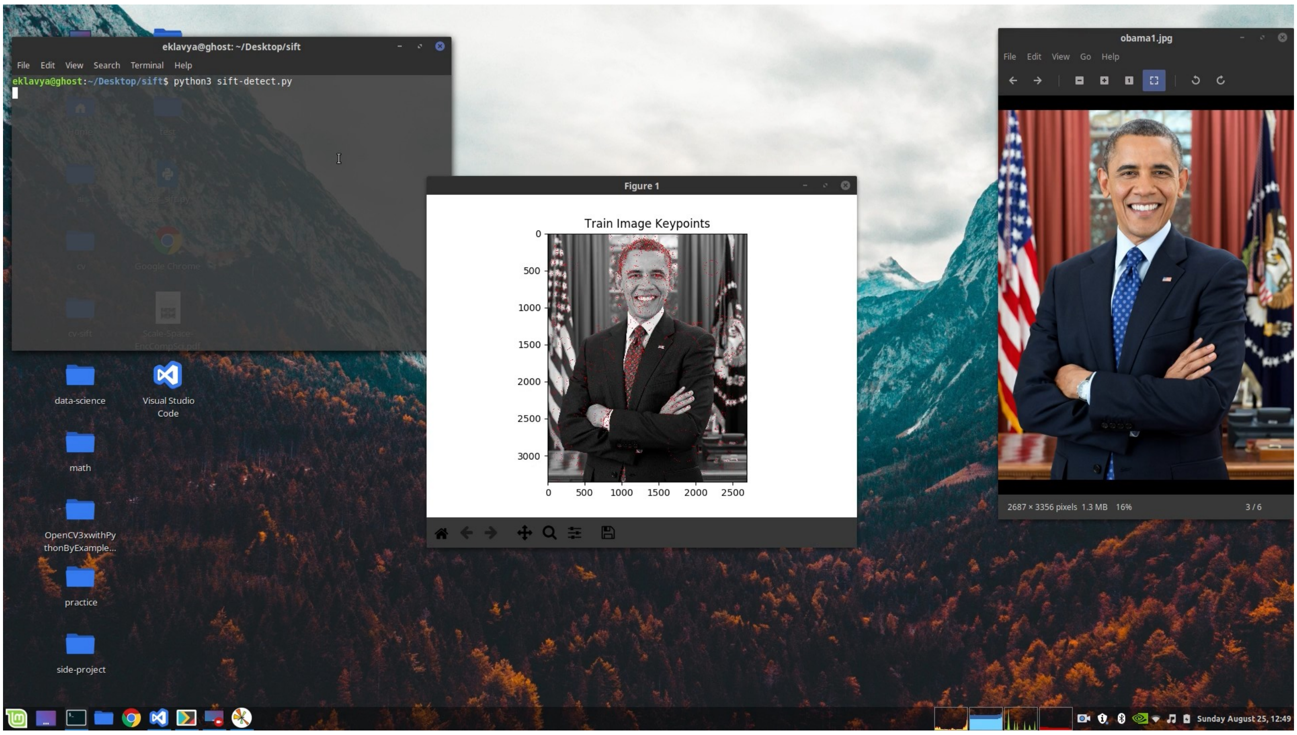
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Threshold
MIN_MATCH_COUNT=30

# Initiate SIFT detector
sift=cv2.xfeatures2d.SIFT_create()


# Create the Flann Matcher object
```

```
FLANN_INDEX_KDITREE=0
flannParam=dict(algorithm=FLANN_INDEX_KDITREE,tree=5)
flann=cv2.FlannBasedMatcher(flannParam,{})


train_img = cv2.imread("obama1.jpg",0)  # train image
# find the keypoints and descriptors with SIFT
kp1,desc1 = sift.detectAndCompute(train_img,None)
# draw keypoints of the train image
train_img_kp= cv2.drawKeypoints(train_img,kp1,None,(255,0,0),4)
# show the train image keypoints
plt.imshow(train_img_kp)
plt.show()

# start capturing video
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    # turn the frame captured into grayscale.
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    # find the keypoints and descriptors with SIFT  of the frame captured.
    kp2, desc2 = sift.detectAndCompute(gray,None)

    # Obtain matches using K-Nearest Neighbor Method.
    #'matches' is the number of similar matches found in both images.
    matches=flann.knnMatch(desc2,desc1,k=2)

    # store all the good matches as per Lowe's ratio test.
    goodMatch=[]
    for m,n in matches:
        if(m.distance<0.75*n.distance):
            goodMatch.append(m)

    # If enough matches are found, we extract the locations of matched 
keypoints in both the images.
    # They are passed to find the perpective transformation.
    # Then we are able to locate our object.
    if(len(goodMatch)>MIN_MATCH_COUNT):
        tp=[]  # src_pts
        qp=[]  # dst_pts
        for m in goodMatch:
            tp.append(kp1[m.trainIdx].pt)
            qp.append(kp2[m.queryIdx].pt)
        tp,qp=np.float32((tp,qp))

        H,status=cv2.findHomography(tp,qp,cv2.RANSAC,3.0)


        h,w = train_img.shape
        train_outline= np.float32([[[0,0],[0,h-1],[w-1,h-1],[w-1,0]]])
        query_outline = cv2.perspectiveTransform(train_outline,H)

        cv2.polylines(frame,[np.int32(query_outline)],True,(0,255,0),5)
        cv2.putText(frame,'Object Found',(50,50), cv2.FONT_HERSHEY_COMPLEX, 
2 ,(0,255,0), 2)
        print("Match Found-")
        print(len(goodMatch),MIN_MATCH_COUNT)
    else:
        print("Not Enough match found-")
        print(len(goodMatch),MIN_MATCH_COUNT)
    cv2.imshow('result',frame)

    if cv2.waitKey(1) == 13:
        break
cap.release()
cv2.destroyAllWindows()
```

Result:



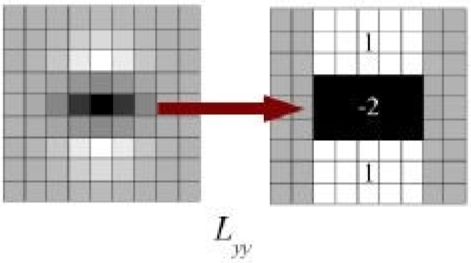
```
61 30
Match Found-
55 30
Match Found-
64 30
Match Found-
62 30
Match Found-
69 30
Match Found-
60 30
Match Found-
70 30
Match Found-
72 30
Not Enough match found-
29 30
Not Enough match found-
30 30
Match Found-
53 30
Match Found-
82 30
```

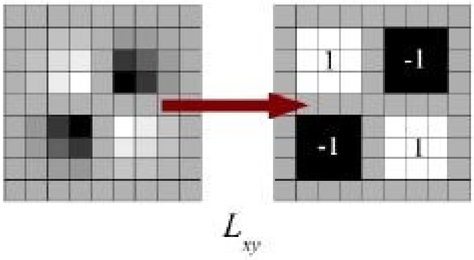


```
eklavya@ghost:~/Desktop/sift$ python3 sift-detect.py
```

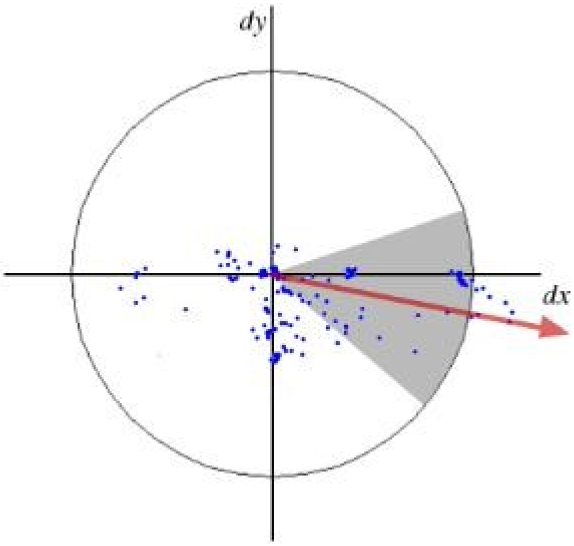
Introduction to SURF (Speeded-Up Robust Features)

In SIFT, Lowe approximated Laplacian of Gaussian with Difference of Gaussian for finding scale-space. SURF goes a little further and approximates LoG with Box Filter. Below image shows a demonstration of such an approximation. One big advantage of this approximation is that, convolution with box filter can be easily calculated with the help of integral images. And it can be done in parallel for different scales. Also the SURF rely on determinant of Hessian matrix for both scale and location





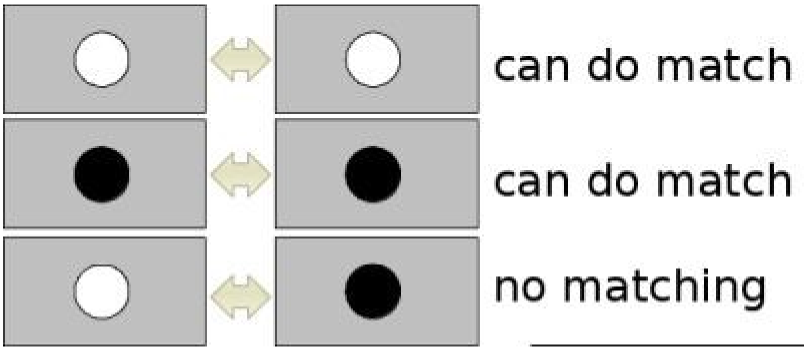
For orientation assignment, SURF uses wavelet responses in horizontal and vertical direction for a neighbourhood of size 6s. Adequate gaussian weights are also applied to it. Then they are plotted in a space as given in below image. The dominant orientation is estimated by calculating the sum of all responses within a sliding orientation window of angle 60 degrees. Interesting thing is that, wavelet response can be found out using integral images very easily at any scale. For many applications, rotation invariance is not required, so no need of finding this orientation, which speeds up the process. SURF provides such a functionality called Upright-SURF or U-SURF. It improves speed and is robust upto ±15◦. OpenCV supports both, depending upon the flag, upright. If it is 0, orientation is calculated. If it is 1, orientation is not calculated and it is faster.



For feature description, SURF uses Wavelet responses in horizontal and vertical direction (again, use of integral images makes things easier). A neighbourhood of size 20sX20s is taken around the keypoint where s is the size. It is divided into 4x4 subregions. For each subregion, horizontal and vertical wavelet responses are taken and a vector is formed like this,

$$v = \left(\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right)$$

This when represented as a vector gives SURF feature descriptor with total 64 dimensions. Lower the dimension, higher the speed of computation and matching, but provide better distinctiveness of features.
For more distinctiveness, SURF feature descriptor has an extended 128 dimension version. The sums of dx nand |dx|are computed separately for dy<0 and dy≥0. Similarly, the sums of dy and |dy|are split up according to the sign of dx , thereby doubling the number of features. It doesn't add much computation complexity. OpenCV supports both by setting the value of flag extended with 0 and 1 for 64-dim and 128-dim respectively (default is 128-dim)
Another important improvement is the use of sign of Laplacian (trace of Hessian Matrix) for underlying interest point. It adds no computation cost since it is already computed during detection. The sign of the Laplacian distinguishes bright blobs on dark backgrounds from the reverse situation. In the matching stage, we only compare features if they have the same type of contrast (as shown in image below). This minimal information allows for faster matching, without reducing the descriptor's performance.



In short, SURF adds a lot of features to improve the speed in every step. Analysis shows it is 3 times faster than SIFT while performance is comparable to SIFT. SURF is good at handling images with blurring and rotation, but not good at handling viewpoint change and illumination change.

SURF in OpenCV

OpenCV provides SURF functionalities just like SIFT. You initiate a SURF object with some optional conditions like 64/128-dim descriptors, Upright/Normal SURF etc. All the details are well explained in docs. Then as we did in SIFT, we can use SURF.detect(), SURF.compute() etc for finding keypoints and descriptors.

First we will see a simple demo on how to find SURF keypoints and descriptors and draw it. All examples are shown in Python terminal since it is just same as SIFT only.

```
>>> img = cv.imread('fly.png',0)
# Create SURF object. You can specify params here or later.
# Here I set Hessian Threshold to 400
>>> surf = cv.xfeatures2d.SURF_create(400)
# Find keypoints and descriptors directly
>>> kp, des = surf.detectAndCompute(img,None)
>>> len(kp)
699
```

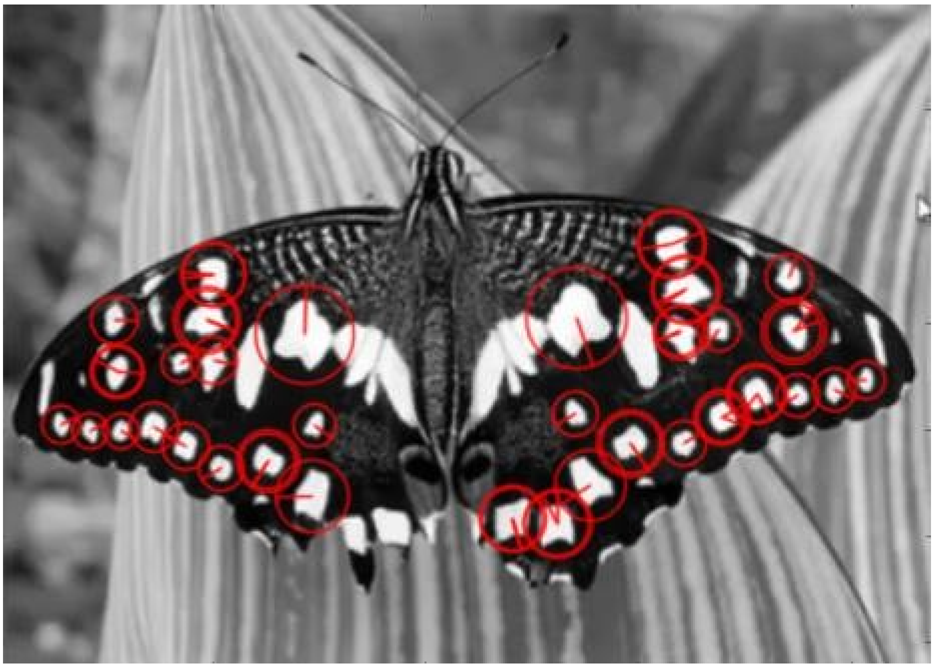
1199 keypoints is too much to show in a picture. We reduce it to some 50 to draw it on an image. While matching, we may need all those features, but not now. So we increase the Hessian Threshold.

```
# Check present Hessian threshold
>>> print( surf.getHessianThreshold() )
400.0
# We set it to some 50000. Remember, it is just for representing in picture.
# In actual cases, it is better to have a value 300-500
>>> surf.setHessianThreshold(50000)
# Again compute keypoints and check its number.
>>> kp, des = surf.detectAndCompute(img,None)
>>> print( len(kp) )
47
```

It is less than 50. Let's draw it on the image.

```
>>> img2 = cv.drawKeypoints(img,kp,None,(255,0,0),4)
>>> plt.imshow(img2),plt.show()
```

See the result below. You can see that SURF is more like a blob detector. It detects the white blobs on wings of butterfly. You can test it with other images.

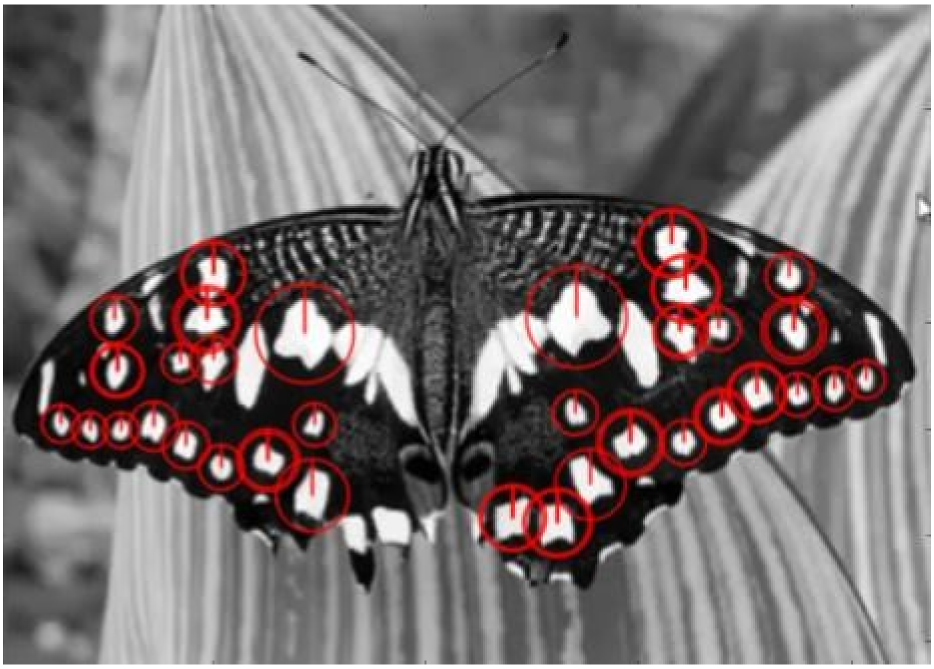


image

Now I want to apply U-SURF, so that it won't find the orientation.

```
# Check upright flag, if it False, set it to True
>>> print( surf.getUpright() )
False
>>> surf.setUpright(True)
# Recompute the feature points and draw it
>>> kp = surf.detect(img,None)
>>> img2 = cv.drawKeypoints(img,kp,None,(255,0,0),4)
>>> plt.imshow(img2),plt.show()
```

See the results below. All the orientations are shown in same direction. It is faster than previous. If you are working on cases where orientation is not a problem (like panorama stitching) etc, this is better.



image

Finally we check the descriptor size and change it to 128 if it is only 64-dim.

```
# Find size of descriptor
>>> print( surf.descriptorSize() )
64
# That means flag, "extended" is False.
>>> surf.getExtended()
False
# So we make it to True to get 128-dim descriptors.
>>> surf.setExtended(True)
>>> kp, des = surf.detectAndCompute(img,None)
>>> print( surf.descriptorSize() )
128
>>> print( des.shape )
(47, 128)
```

Platform used (Hardware/Software)

Hardware:

Brand	Dell
Colour	Black
Form Factor	Laptop
Standing screen display size	14 Inches
Package Dimensions	38.1 x 33.02 x 5.08 cm; 1.6 Kilograms
Processor Brand	Intel
Processor Type	Core i5
Processor Count	4
RAM Size	4 GB
Hard Disk Description	Solid_state_drive
Graphics Chipset Brand	Intel
Connectivity Type	Wi-Fi
Operating System	Windows 10 Pro
Item Weight	1 kg 600 g

System Requirements :

4 GB RAM (Minimum)

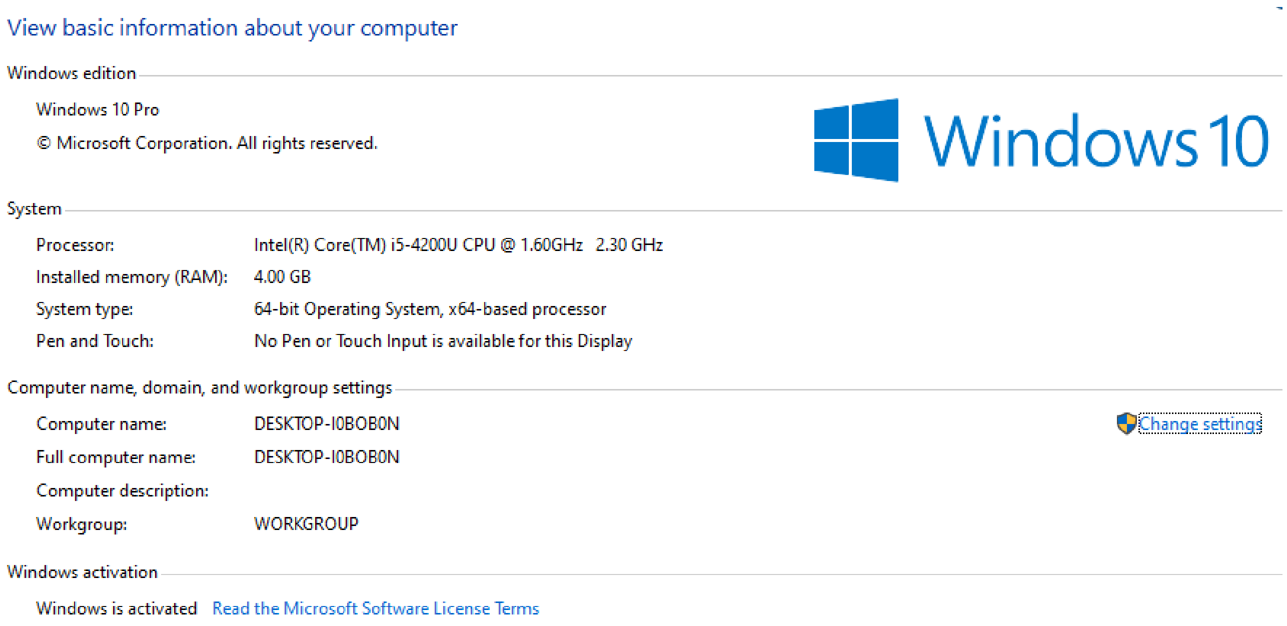
80 GB HDD

Dual Core processor

CDROM (installation only). VGA resolution monitor

Windows 7,Windows 8 and 8.1,Windows 10,Windows 11

SQL Server 2008 R2 or greater



View basic information about your computer

Windows edition

Windows 10 Pro

© Microsoft Corporation. All rights reserved.

System

Processor:	Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz 2.30 GHz
Installed memory (RAM):	4.00 GB
System type:	64-bit Operating System, x64-based processor
Pen and Touch:	No Pen or Touch Input is available for this Display

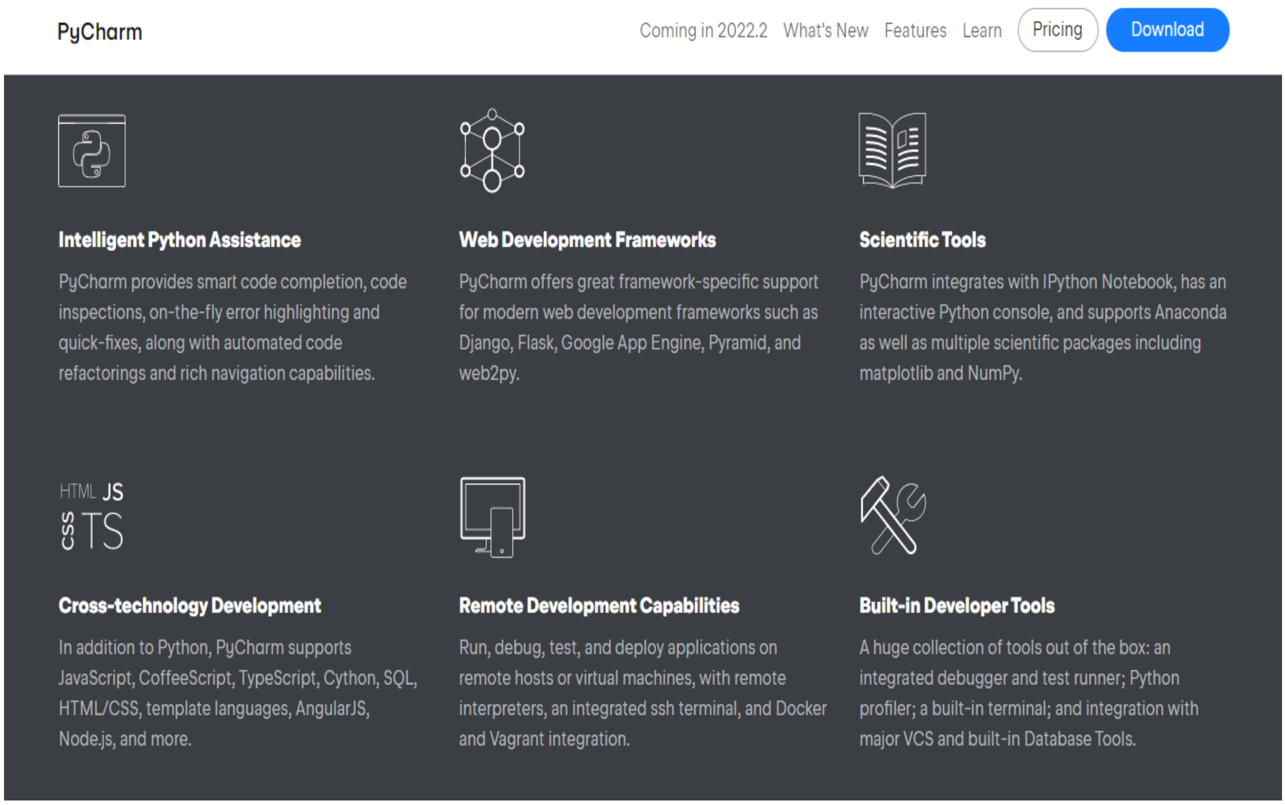
Computer name, domain, and workgroup settings

Computer name:	DESKTOP-I0BOB0N
Full computer name:	DESKTOP-I0BOB0N
Computer description:	
Workgroup:	WORKGROUP

Windows activation

Windows is activated Read the Microsoft Software License Terms

Pycharm: the Python IDE



Intelligent Python Assistance

PyCharm provides smart code completion, code inspections, on-the-fly error highlighting and quick-fixes, along with automated code refactorings and rich navigation capabilities.

Web Development Frameworks

PyCharm offers great framework-specific support for modern web development frameworks such as Django, Flask, Google App Engine, Pyramid, and web2py.

Scientific Tools

PyCharm integrates with IPython Notebook, has an interactive Python console, and supports Anaconda as well as multiple scientific packages including matplotlib and NumPy.

Cross-technology Development

In addition to Python, PyCharm supports JavaScript, CoffeeScript, TypeScript, Cython, SQL, HTML/CSS, template languages, AngularJS, Node.js, and more.

Remote Development Capabilities

Run, debug, test, and deploy applications on remote hosts or virtual machines, with remote interpreters, an integrated ssh terminal, and Docker and Vagrant integration.

Built-in Developer Tools

A huge collection of tools out of the box: an integrated debugger and test runner; Python profiler; a built-in terminal; and integration with major VCS and built-in Database Tools.

PyCharm is one of the most popular Python IDEs. There is a multitude of reasons for this, including the fact that it is developed by JetBrains, the developer behind the popular IntelliJ IDEA IDE that is one of the big 3 of Java IDEs and the "smartest JavaScript IDE" WebStorm. Having the support for web development by leveraging Django is yet another credible reason.

There are a galore of factors that make PyCharm one of the most complete and comprehensive integrated development environments for working with the Python programming language.

Before proceeding further into exploring the know-how of PyCharm i.e., features, installation, and pros & cons, let's first get a brief introduction to PyCharm.

What is PyCharm?

Available as a cross-platform application, PyCharm is compatible with Linux, macOS, and Windows platforms. Sitting gracefully among the best Python IDEs, PyCharm provides support for both Python 2 (2.7) and Python 3 (3.5 and above) versions.

PyCharm comes with a plethora of modules, packages, and tools to hasten Python development while cutting-down the effort required to do the same to a great extent, simultaneously. Further, PyCharm can be customized as per the development requirements, and personal preferences call for. It was released to the public for the very first time back in February of 2010. In addition to offering code analysis, PyCharm features:

- A graphical debugger
- An integrated unit tester
- Integration support for version control systems (VCSs)
- Support for data science with Anaconda

What is PyCharm used for?

The main reason Pycharm for the creation of this IDE was for Python programming, and to operate across multiple platforms like Windows, Linux, and macOS. The IDE comprises code analysis tools, debugger, testing tools, and also version control options. It also assists developers in building Python plugins with the help of various APIs available. The IDE allows us to work with several databases directly without getting it integrated with other tools. Although it is specially designed for Python, HTML, CSS, and Javascript files can also be created with this IDE. It also comes with a beautiful user interface that can be customized according to the needs using plugins.

What is an IDE?

PyCharm is an extremely popular Python IDE. An Integrated Development Environment or IDE features a code editor and a compiler for writing and compiling programs in one or many programming languages.

Furthermore, an IDE comes with a galore of features that facilitate comprehensive software development. As an IDE allocates different colors to different programming entities, typically known as syntax highlighting, it becomes more accessible to:

- Differentiate between various programming entities, such as a class and a function, and to spot them.
- Look for the wrong keywords.
- Read and comprehend the code.

Most IDEs feature an auto-complete feature that produces suggestions when writing code. This makes writing code more efficient, quick, and less prone to errors and typos. Other standard features offered by a modern IDE are:

- Project editor window for efficiently managing and organizing files necessary for a program/project.
- Inspecting the output of the code written using the output window
- Suggestions for resolving errors and warnings
- A range of modules and packages readily available at a single place

PyCharm Pricing Model

PyCharm is offered in three variants:

- A freemium version dubbed The Community Edition available under the Apache License.
- A commercial version labeled Professional Edition available under a proprietary license.
- A free-to-use educational version dubbed the Edu Edition, aimed for students and professionals interested in learning Python, available under the Apache License.

Features of PyCharm

1. Intelligent Code Editor

PyCharm comes with a smart code editor that facilitates writing high-quality Python code. It offers an enhanced level of code comprehension and readability by means of distinct color schemes for keywords, classes, and functions, i.e., syntax and error highlighting.

In addition to offering the smart code completion feature, the code editor generates instructions for completing the current code. Identifying errors and issues is much more comfortable, along with linter integration and quick fixes.

2. Availability of Integration Tools

PyCharm provides support for integrating a range of tools. These tools vary from helping in enhancing the code productivity to facilitate dealing with data science projects. Some of the most essential integration tools available for PyCharm include:

- Anaconda - A free and open-source Python distribution geared towards scientific computing with simplified package management and deployment.
- IPython - A robust command shell for interactive computing.
- Kite - An AI-powered autocomplete plugin.
- Pylint - A source-code, bug, and quality checker.
- pytest - A framework for writing small tests for Python code.
- WakaTime - A developer dashboard with productivity metrics and automatic time tracking

3. Data Science and Machine Learning [Professional Edition Only]

PyCharm comes with support for scientific libraries, such as Matplotlib and SciPy, to help Python developers accomplish data science and machine learning projects.

4.Google App Engine [Professional Edition Only]

Google App Engine, or directly App Engine, is a PaaS and cloud computing platform targeted towards developing and hosting web applications. It offers automatic scaling for web applications. The professional edition of PyCharm provides support for Google App Engine.

5. Integrated Debugging and Testing

An IDE comes with support for debugging and testing programs. To accomplish the same, PyCharm features an integrated Python debugger and integrated unit testing with line-by-line code coverage.

6. Multi-technology Development [Professional Edition Only]

Python developers can also use PyCharm for creating web applications. As such, the Python IDE provides support for popular web technologies, including CoffeeScript, CSS, HTML, JavaScript, TypeScript. Additionally, it also includes support for Cython, template languages, and SQL.

Live editing is also available in PyCharm, i.e., developers can create/modify a web page while pushing it live simultaneously. Hence, changes can be followed directly on a web browser. Building web applications using AngularJS or NodeJS is also available.

7. Project and Code Navigation

The code navigation feature makes it much easier for developers to navigate to a class, function, or file. It also helps in significantly cutting-down effort and time required to edit and enhance the Python code. File structure views and specialized project views are readily available.

The lens mode allows a developer to inspect and debug the entire Python source code thoroughly. With code navigation, locating an element, variable, etc. is done in almost no time. Developers can quickly jump between classes, files, and methods.

8. Refactoring

The refactoring feature in PyCharm helps in improving the internal structure of a Python program without affecting the external performance of the same. Making changes to both local and global variables is efficient and fast.

The extract method is also there to split up extended classes and functions. Other useful code refactoring features include:

- Introduce constant
- Introduce variable
- Pull up
- Push down
- Rename

9. Remote Development

PyCharm allows running, debugging, testing, and deploying applications on remote hosts or virtual machines. For the purpose, the Python IDE offers:

- An integrated SSH terminal
- Docker and Vagrant integration
- Remote interpreters

10. Support for Popular Python Web Frameworks [Professional Edition Only]

PyCharm lets developers leverage Django in their Python development projects. The Python IDE offers the autocomplete feature and generates suggestions for Django.

Debugging code written using Django is also available. PyCharm also provides support for other popular Python frameworks, namely Flask, Pyramid, and web2py.

11. Version Control Systems (VCSs) Integration

In its simplicity, a version control system (VCS) keeps track of the changes made to files, applications, and other sources of information. It can be considered as a database of changes.

PyCharm provides a unified user interface for CVS, Git, Mercurial, Perforce, and Subversion.

Other Pycharm Features

Other than the aforementioned features, PyCharm offers the following capabilities:

- Code generation for generating language-specific code constructs.
- Code reference information for instantly accessing API documentation, hints on using various programming entities, etc
- File templates for creating scripts, stub classes, etc
- Import assistance for importing missing libraries
- Intention actions and quick fixes for optimizing code
- Language-specific tools for developing, running, testing, and deploying applications
- Language injections to work with supported languages inside attributes, tags, or string literals
- Live templates for expanding abbreviations into complicated code constructs

Installing and Setting Up PyCharm

Minimum System Requirements

Memory - 4GB

Storage Space - 2.5GB (main) + 1GB (caches)

Resolution - 1024x768

OS - 64-bit version of macOS 10.11/Microsoft Windows 7 SP1/any Linux distribution supporting Gnome, KDE, or Unity DE

Recommended System Requirements

Memory - 8GB

Storage Space - 5GB of SSD

Resolution - 1920x1080

OS - Any latest 64-bit version of macOS/Microsoft Windows/Linux

One of the significant advantages is that PyCharm brings to the Python development table is easy installation and less setup time. There are three modes of installation available for PyCharm:

1. Standalone Installation

Here is how to install the popular Python IDE in the traditional way:

•Step #01 - Go to https://www.jetbrains.com/pycharm/download.

•Step #02 - Choose a platform among Windows, Mac, and Linux.

For Windows

•Step #03 - Choose the PyCharm edition; Professional (paid), or Community (free).

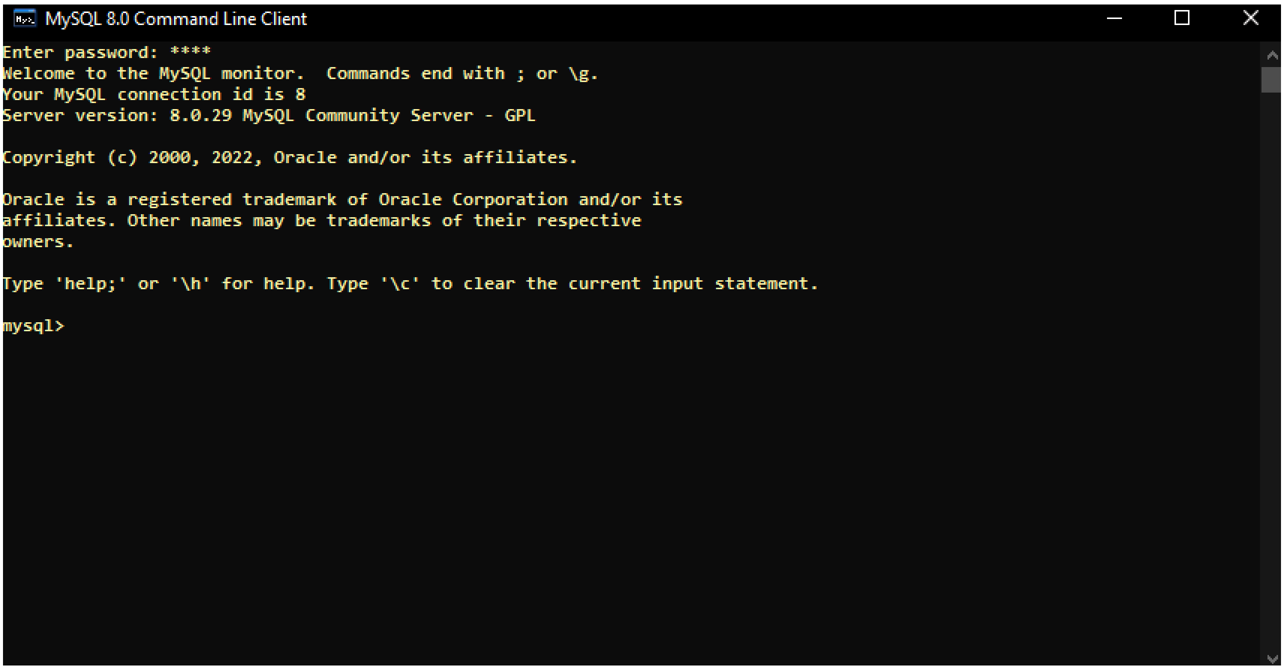
•Step #04 - Hit the Download button to start downloading the .exe installer.

•Step #05 - Once downloaded, successfully run the installer and follow the setup wizard steps.

Conclusion

PyCharm is, undoubtedly, one of the most efficient and powerful IDEs for tackling Python development. The continuously rising popularity of Python ensures that the Python IDE will continue to be developed for the better over time and will attract new Python developers.

Mysql



```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.29 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

MySQL is one of the most recognizable technologies in the modern big data ecosystem. Often called the most popular database and currently enjoying widespread, effective use regardless of industry, it's clear that anyone involved with enterprise data or general IT should at least aim for a basic familiarity of MySQL.

With MySQL, even those new to relational systems can immediately build fast, powerful, and secure data storage systems. MySQL's programmatic syntax and interfaces are also perfect gateways into the wide world of other popular query languages and structured data stores.

What is MySQL?

MySQL is a relational database management system (RDBMS) developed by Oracle that is based on structured query language (SQL).

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or a place to hold the vast amounts of information in a corporate network. In particular, a relational database is a digital store collecting data and organizing it according to the relational model. In this model, tables consist of rows and columns, and relationships between data elements all follow a strict logical structure. An RDBMS is simply the set of software tools used to actually implement, manage, and query such a database.

MySQL is integral to many of the most popular software stacks for building and maintaining everything from customer-facing web applications to powerful, data-driven

B2B services. Its open-source nature, stability, and rich feature set, paired with ongoing development and support from Oracle, have meant that internet-critical organizations such as Facebook, Flickr, Twitter, Wikipedia, and YouTube all employ MySQL backends.

Current Developer	Oracle Corporation
Original Developer	MySQL AB (Then, briefly, Sun Microsystems)
Current Stable Release	8.0.16 (on April 25, 2019)
Original Release	May 23, 1995
License	GPLv2 (or proprietary)
Primary language	C and C++
Website	https://www.mysql.com/
Open-source repository	https://github.com/mysql/mysql-server

4 keys to understanding MySQL

Because MySQL enjoys the most widespread use in many industries, business users from new webmasters to experienced managers should strive to understand its main characteristics. Deciding whether to use this technology, and communicating about it effectively, starts with a review of MySQL's basic availability, structure, philosophy, and usability.

MySQL is widely compatible

Though often associated with internet applications or web services, MySQL was designed to be extensively compatible with other technologies and architectures. The RDBMS runs on all major computing platforms, including Unix-based operating systems, such as the myriad Linux distributions or Mac OS, and Windows.

MySQL's client-server architecture means it can support a variety of backends, as well as different programming interfaces. Data can be directly migrated from MySQL to its forks (e.g. MariaDB), as well as most other RDBMSs thanks to architectural and language similarities.

Established Oracle and third-party migration tools further allow MySQL to move data to and from a vast set of general storage systems, whether these are designed to be on-premises or cloud-based. MySQL can be deployed in virtualized environments, distributed or centralized, and even exists as portable standalone libraries for learning purposes, testing, or small applications.

MySQL's wide compatibility with all these other systems and software makes it a particularly practical choice of RDBMS in most situations.

MySQL databases are relational

The primary factor differentiating relational databases from other digital storage lies in how data is organized at a high level. Databases like MySQL contain records in multiple, separate, and highly codified tables, as opposed to a single all-encompassing repository, or collections of semi- or unstructured documents.

This allows RDBMSs to better optimize actions like data retrieval, updating information, or more complex actions like aggregations. A logical model is defined over all of the contents of the database, describing for example the values allowed in individual columns, characteristics of tables and views, or how indices from two tables are related.

Relational models have remained popular for several reasons. They empower users with intuitive, declarative programming languages — essentially telling the database what result is wanted in language akin to, or at least comprehensible as, written english, instead of meticulously coding up each step of the procedure leading to that result. This moves a lot of the work into the RDBMS and SQL engines, better enforcing logical rules and saving valuable resources and manpower.

MySQL is open-source

Any individual or enterprise may freely use, modify, publish, and expand on Oracle's open-source MySQL code base. The software is released under the GNU General Public License (GPL).

For MySQL code needing to be integrated or included in a commercial application (or if open-source software is not a priority), enterprises can purchase a commercially licensed version from Oracle.

Again, these options provide organizations with additional flexibility if deciding to work with MySQL. The public and community-based nature of open-source releases enriches MySQL's documentation and online support culture, while also ensuring that sustained or newly-developed capabilities never stray too far from current user needs.

MySQL is easy to use

Though MySQL's relational nature and the ensuing rigid storage structures might seem restrictive, the tabular paradigm is perhaps the most intuitive, and ultimately allows for greater usability.

In fact, MySQL makes many concessions to supporting the widest possible variety of data structures, from the standard but rich logical, numeric, alphanumeric, date, and time types, to more advanced JSON or geospatial data. Beyond mere data types and an expansive built-in feature set, the MySQL ecosystem also includes a variety of tools, easing everything from server management to reporting and data analysis.

Regardless of the RDBMS's overarching architecture, users can invariably find a MySQL feature allowing them to model and codify data how they wish. MySQL remains one of the most straightforward database technologies to learn and use.

The relational model was first delineated in a 1970 paper by Edgar F. Codd. One of the first commercial programming languages related to the model, SQL, was developed shortly after at IBM. For some time, SQL was the most widely used database language, adopted as an ANSI standard in 1986 and in ISO a year later.

SQL is composed of four sublanguages, each with a different scope.

- **DQL:** The data query language (DQL) is the most familiar and is used to run queries on databases and extract information from stored data. For example, selecting and returning the maximum value in a column.
- **DDL:** A data definition language (DDL) is used to codify a database's particular structures and schemas. Creating a table or defining data types is an example.
- **DCL:** A data control language (DCL) defines access, authorizations, and permissions for users and processes accessing the database, including granting administrator privileges, or restricting users to read-only privileges only.
- **DML:** And finally, a data manipulation language (DML) is used to make modifications on existing components of a database, like inserting records, updating values in cells, or deleting data.

Swedish company MySQL AB first released MySQL in 1995. Like much of the database software which followed the initial rise of relational systems, MySQL is simply an extension of the original SQL standard, adding more features, support, procedural programming, control-flow mechanisms, and more.

MySQL is a popular, time-tested, but also modern and fully-featured relational database management software. Businesses everywhere use it for mission-critical enterprise data storage, processing, as a backend to major customer-facing applications, and as part of powerful, established web software stacks.

Whether your business already uses MySQL or is planning new systems or migrations to this RDBMS, the importance of data integration cannot be overstated. Talend provides a comprehensive suite of apps for managing data ecosystems from end to end, allowing businesses to collect, transform, govern, and share fast and trusted data from any system.Try Talend Data Fabric today for a seamless data ecosystem.

Module of the project:

The module of this project is divided into six distinct sections, starting with gathering data from users and ending with user recognition.

Dependencies are a key contributing reason to this success.

Each module in this project has classes and a variety of functions to carry out different operations.

The folder project houses the whole project, and within it are many pictures, python programmes, and xml files that make up the project's dependencies.

The project folder must include the files. To complete this project, the code must continue the directory location of the file if it is not.

Lateron, the directory might be altered to suit the requirements of the users.

Module 1:

Fileame – Startup.py

main()	Open the frame
Login_window	A framework for choosing the operation form user.
register_window_open()	To open the window where teacher can register
register_student()	To open the window where student can register.
After_fact()	A Dashboard after teacher login successfully.
Stu_login()	A Framework to Login Sudent.
login()	The login fuction for administrator.
Forgot_pass()	A Toplevel Framework if administrator forgots password.
Forgot_thing()	Function to retrieve the password of administrator.
main_app()	Function used as a recogniser of the student.
Register	An framework for registering the Teacher.
des()	To destroy the current frame
register_button()	To register new administrator.
Student	A framework for registering new student.
Stu_register()	A function to register new student successfully.

Note:The () symbol denotes a function, whereas a class just has the name of the class.

Module 2:

Filename -homePage.py

Affact	A Dashboard for administrator for operations.

ClearAll()	A function for resetting all text fields.
CheckData()	A function for searching for student's information.
changeToShowAt()	A function to open a window for today's attendence.
DeleteSudentAt()	A function for intializing the framework for delte student.
ShowbyHand()	Function for viewing today's Attendence.
DelByTeacher	Class to Initialise delete selected student.
Classifier_after_Delete()	Re-train images after deleting any student.

Module 3:

LoginPage	A framework for student login.
Show()/hide()	Function for show or hide password
login_for_student()	Function for student login()
forgot_Pass()	Function for retrieval of password
After_student_login	Student dashboard.
Average()	Calculating the percentage
Auto()	Auto calling function for another functions
Update()	Updating new data to the database
clear()	Clearing all the text fields in the frame

Module 4:

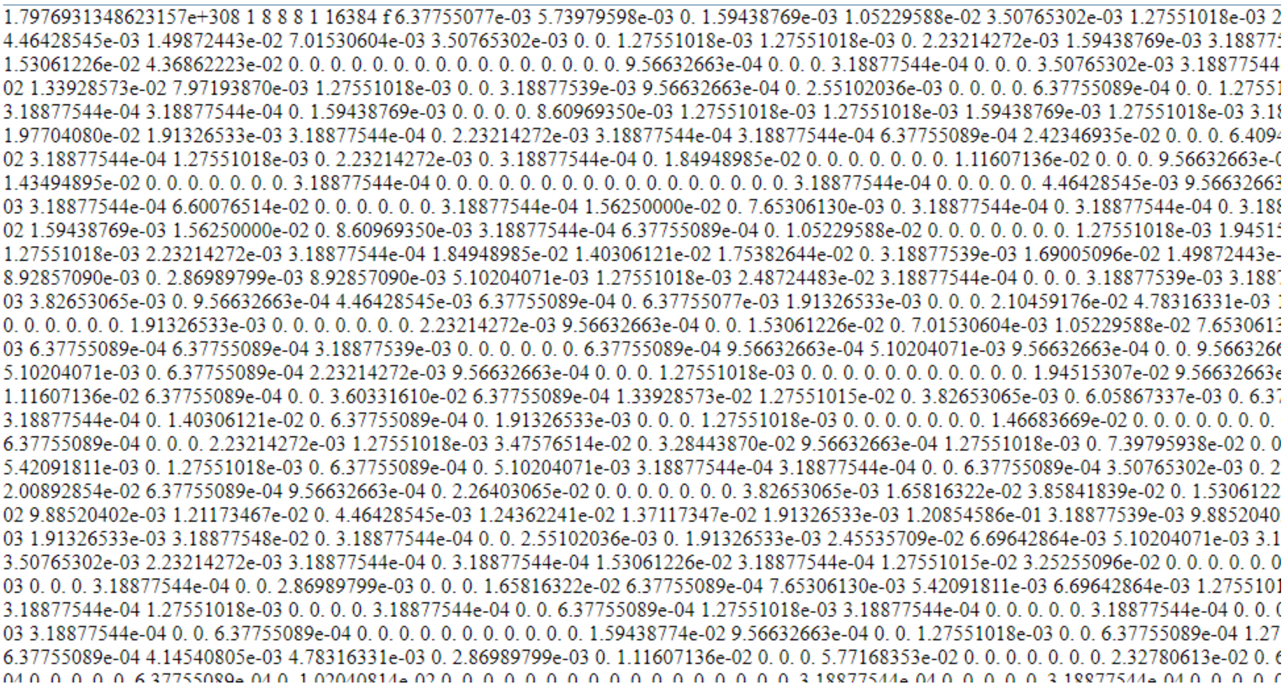
Classifier.xml

Training images are a bunch of images for which the required outcome is known. You input them to a program that analyzers their features and passes the features through a classification routine that determines the appropriate weights to use on the features in order to best achieve the required result.

Once you have used the training images to determine the best weights and procedures to follow, you then pass through a set of "test" images, for which you also know the outcome. The images are extracted and the procedures and weights determined above are applied to make a prediction of how the image should classify. Then the predicted classification is compared to the actual known classification to determine how well the procedures and weights do on analyzing data that was not originally used to determine the weights. If you got a high score on the training images but a low score on the test images then the implication is that the routine "overtrained" and has become too specific to the training images and not able to predict for images whose results are not known.

Once you are getting good scores on the training images and good scores on the test images as well, you can start using the procedures and weights to analyze data for which the outcome is *not* known.

After training images the xml file looks like this.



```
1.7976931348623157e+308 1 8 8 8 1 16384 f 6.37755077e-03 5.73979598e-03 0. 1.59438769e-03 1.05229588e-02 3.50765302e-03 1.27551018e-03 2
4.46428545e-03 1.49872443e-02 7.01530604e-03 3.50765302e-03 0. 0. 1.27551018e-03 1.27551018e-03 0. 2.23214272e-03 1.59438769e-03 3.18877
1.53061226e-02 4.36862223e-02 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 9.56632663e-04 0. 0. 0. 3.18877544e-04 0. 0. 0. 3.50765302e-03 3.18877544
02 1.33928573e-02 7.97193870e-03 1.27551018e-03 0. 0. 3.18877539e-03 9.56632663e-04 0. 2.55102036e-03 0. 0. 0. 0. 6.37755089e-04 0. 0. 1.27551
3.18877544e-04 3.18877544e-04 0. 1.59438769e-03 0. 0. 0. 0. 8.60969350e-03 1.27551018e-03 1.27551018e-03 1.59438769e-03 1.27551018e-03 3.1
1.97704080e-02 1.91326533e-03 3.18877544e-04 0. 2.23214272e-03 3.18877544e-04 3.18877544e-04 6.37755089e-04 2.42346935e-02 0. 0. 0. 6.409
02 3.18877544e-04 1.27551018e-03 0. 2.23214272e-03 0. 3.18877544e-04 0. 1.84948985e-02 0. 0. 0. 0. 0. 0. 0. 1.11607136e-02 0. 0. 0. 9.56632663e-
1.43494895e-02 0. 0. 0. 0. 0. 0. 0. 3.18877544e-04 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 3.18877544e-04 0. 0. 0. 0. 4.46428545e-03 9.56632663
03 3.18877544e-04 6.60076514e-02 0. 0. 0. 0. 0. 0. 3.18877544e-04 1.56250000e-02 0. 7.65306130e-03 0. 3.18877544e-04 0. 3.18877544e-04 0. 3.188
02 1.59438769e-03 1.56250000e-02 0. 8.60969350e-03 3.18877544e-04 6.37755089e-04 0. 1.05229588e-02 0. 0. 0. 0. 0. 0. 0. 1.27551018e-03 1.94515
1.27551018e-03 2.23214272e-03 3.18877544e-04 1.84948985e-02 1.40306121e-02 1.75382644e-02 0. 3.18877539e-03 1.69005096e-02 1.49872443e-
8.92857090e-03 0. 2.86989799e-03 8.92857090e-03 5.10204071e-03 1.27551018e-03 2.48724483e-02 3.18877544e-04 0. 0. 0. 3.18877539e-03 3.188
03 3.82653065e-03 0. 9.56632663e-04 4.46428545e-03 6.37755089e-04 0. 6.37755077e-03 1.91326533e-03 0. 0. 0. 2.10459176e-02 4.78316331e-03 1
0. 0. 0. 0. 0. 0. 1.91326533e-03 0. 0. 0. 0. 0. 0. 0. 2.23214272e-03 9.56632663e-04 0. 0. 1.53061226e-02 0. 7.01530604e-03 1.05229588e-02 7.653061
03 6.37755089e-04 6.37755089e-04 3.18877539e-03 0. 0. 0. 0. 0. 0. 6.37755089e-04 9.56632663e-04 5.10204071e-03 9.56632663e-04 0. 0. 9.566326
5.10204071e-03 0. 6.37755089e-04 2.23214272e-03 9.56632663e-04 0. 0. 0. 1.27551018e-03 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1.94515307e-02 9.56632663e
1.11607136e-02 6.37755089e-04 0. 0. 3.60331610e-02 6.37755089e-04 1.33928573e-02 1.27551015e-02 0. 3.82653065e-03 0. 6.05867337e-03 0. 6.37
3.18877544e-04 0. 1.40306121e-02 0. 6.37755089e-04 0. 1.91326533e-03 0. 0. 0. 1.27551018e-03 0. 0. 0. 0. 0. 0. 0. 1.46683669e-02 0. 0. 0. 0. 0. 0. 0.
6.37755089e-04 0. 0. 0. 2.23214272e-03 1.27551018e-03 3.47576514e-02 0. 3.28443870e-02 9.56632663e-04 1.27551018e-03 0. 7.39795938e-02 0. 0
5.42091811e-03 0. 1.27551018e-03 0. 6.37755089e-04 0. 5.10204071e-03 3.18877544e-04 3.18877544e-04 0. 0. 6.37755089e-04 3.50765302e-03 0. 2
2.00892854e-02 6.37755089e-04 9.56632663e-04 0. 2.26403065e-02 0. 0. 0. 0. 0. 0. 0. 3.82653065e-03 1.65816322e-02 3.85841839e-02 0. 1.5306122
02 9.88520402e-03 1.21173467e-02 0. 4.46428545e-03 1.24362241e-02 1.37117347e-02 1.91326533e-03 1.20854586e-01 3.18877539e-03 9.8852040
03 1.91326533e-03 3.18877548e-02 0. 3.18877544e-04 0. 0. 2.55102036e-03 0. 1.91326533e-03 2.45535709e-02 6.69642864e-03 5.10204071e-03 3.1
3.50765302e-03 2.23214272e-03 3.18877544e-04 0. 3.18877544e-04 1.53061226e-02 3.18877544e-04 1.27551015e-02 3.25255096e-02 0. 0. 0. 0. 0. 0
03 0. 0. 0. 3.18877544e-04 0. 0. 2.86989799e-03 0. 0. 0. 1.65816322e-02 6.37755089e-04 7.65306130e-03 5.42091811e-03 6.69642864e-03 1.2755101
3.18877544e-04 1.27551018e-03 0. 0. 0. 0. 3.18877544e-04 0. 0. 6.37755089e-04 1.27551018e-03 3.18877544e-04 0. 0. 0. 0. 0. 3.18877544e-04 0. 0. 0
03 3.18877544e-04 0. 0. 6.37755089e-04 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1.59438774e-02 9.56632663e-04 0. 0. 1.27551018e-03 0. 0. 6.37755089e-04 1.27
6.37755089e-04 4.14540805e-03 4.78316331e-03 0. 2.86989799e-03 0. 1.11607136e-02 0. 0. 0. 5.77168353e-02 0. 0. 0. 0. 0. 0. 0. 2.32780613e-02 0. 6
04 0. 0. 0. 0. 0. 6.37755089e-04 0. 1.02040814e-02 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 3.18877544e-04 0. 0. 0. 0. 0. 3.18877544e-04 0. 0. 0. 0. 0
```

This is the inside view of Classifier.xml

for us it may looks like bunch of decimal numbers but for computer they are faces.

Module 5:

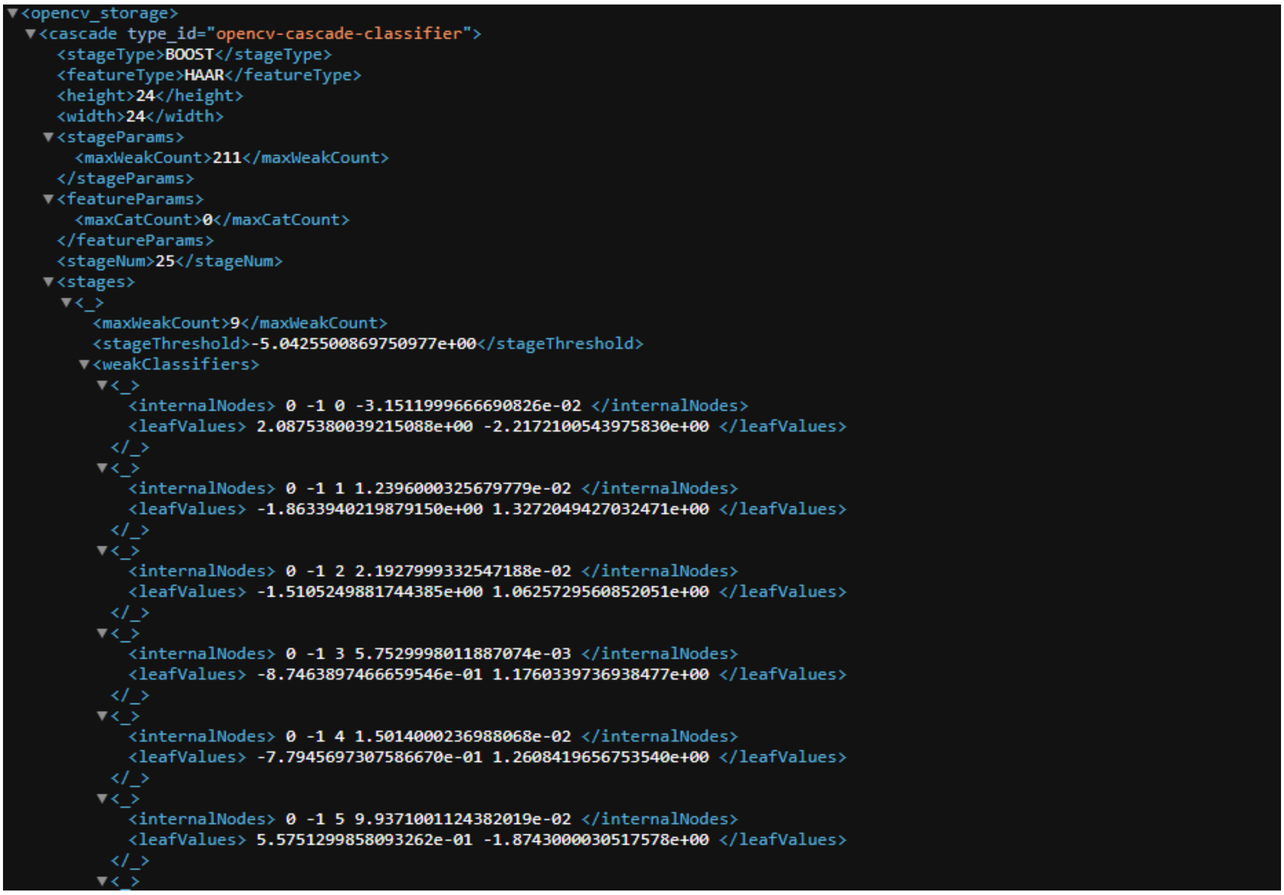
haarcascade_frontalface_default.xml

Haar Cascade classifiers are an effective way of object detection. This method was proposed by Paul Viola and Michael Jones in their paper Rapid Object Detection using a Boosted Cascade of Simple Features. Haar Cascade is a machine learning-based approach where a lot of positive and negative images are used to train the classifier.

- **Positive images:** These images contain the images that we want our classifier to identify.
- **Negative Images:** Images of everything else, which do not contain the object we want to detect.

Requirements:

The inside view of haarcascade_frontalface_default.xml file



```
▼<opencv_storage>
  ▼<cascade type_id="opencv-cascade-classifier">
     <stageType>BOOST</stageType>
     <featureType>HAAR</featureType>
     <height>24</height>
     <width>24</width>
    ▼<stageParams>
       <maxWeakCount>211</maxWeakCount>
     </stageParams>
    ▼<featureParams>
       <maxCatCount>0</maxCatCount>
     </featureParams>
     <stageNum>25</stageNum>
    ▼<stages>
      ▼<_>
         <maxWeakCount>9</maxWeakCount>
         <stageThreshold>-5.0425500869750977e+00</stageThreshold>
        ▼<weakClassifiers>
          ▼<_>
             <internalNodes> 0 -1 0 -3.1511999666690826e-02 </internalNodes>
             <leafValues> 2.0875380039215088e+00 -2.2172100543975830e+00 </leafValues>
           </_>
          ▼<_>
             <internalNodes> 0 -1 1 1.2396000325679779e-02 </internalNodes>
             <leafValues> -1.8633940219879150e+00 1.3272049427032471e+00 </leafValues>
           </_>
          ▼<_>
             <internalNodes> 0 -1 2 2.1927999332547188e-02 </internalNodes>
             <leafValues> -1.5105249881744385e+00 1.0625729560852051e+00 </leafValues>
           </_>
          ▼<_>
             <internalNodes> 0 -1 3 5.7529998011887074e-03 </internalNodes>
             <leafValues> -8.7463897466659546e-01 1.1760339736938477e+00 </leafValues>
           </_>
          ▼<_>
             <internalNodes> 0 -1 4 1.5014000236988068e-02 </internalNodes>
             <leafValues> -7.7945697307586670e-01 1.2608419656753540e+00 </leafValues>
           </_>
          ▼<_>
             <internalNodes> 0 -1 5 9.9371001124382019e-02 </internalNodes>
             <leafValues> 5.5751299858093262e-01 -1.8743000030517578e+00 </leafValues>
           </_>
          ▼<_>
```

System Design:

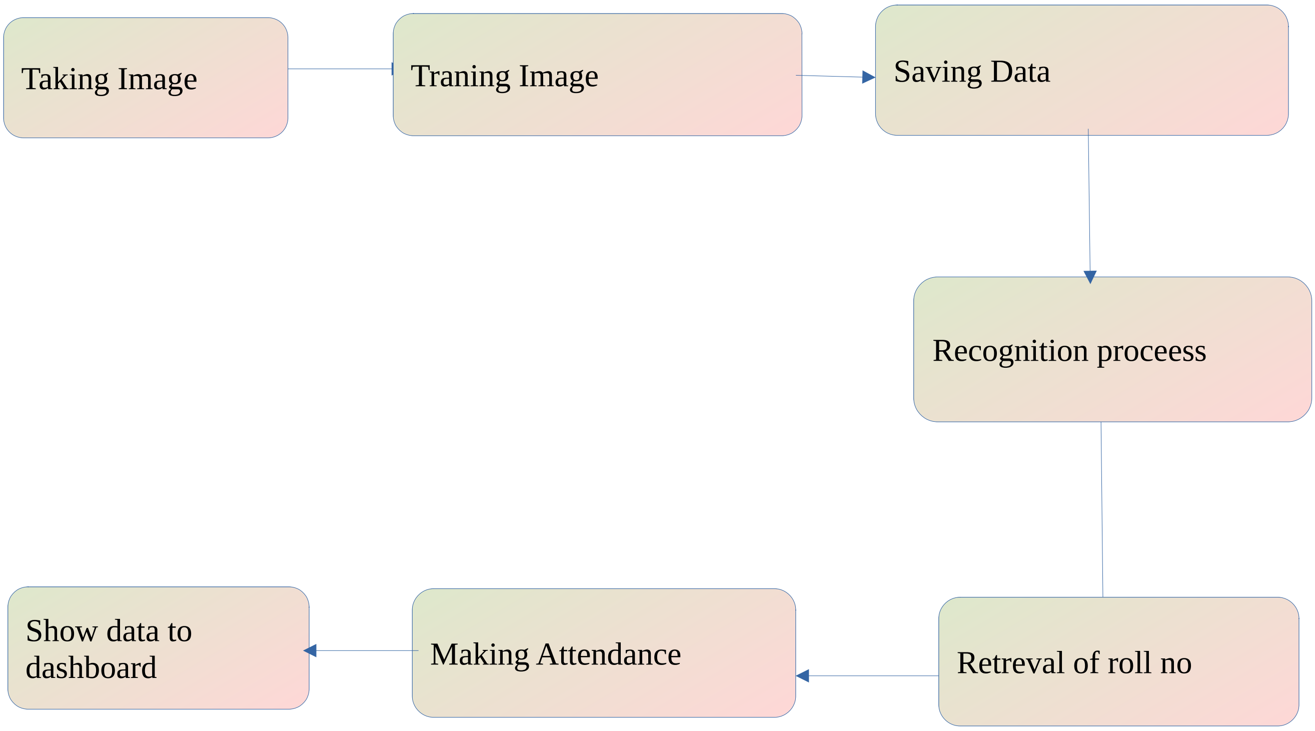
Installing Dependencies;

- User left click on”Start (button)”
- Click on pycharm icon
- User mousehover to file on left corner
- User left click on “Settings(window)” in settins
- Next Click on Python Interpreter
- Search for these listed Packages and install it

1. Babel
2. Django
3. Pillow
4. PyMySQL
5. TatSu
6. asgiref
7. cmake
8. cmake-analyzer
9. colorame
10. decorator
11. face-recognition-models
12. image
13. joblib
14. mysql-connector-python
15. mysqlpy
16. opencv-contrib-python
17. opencv-python
18. pandas
19. pip
20. protobuf
21. python-dateutil
22. pytz
23. self

24. setuptools
25. six
26. sqlparse
27. tkcalendar
28. tqdm
29. tzdata

A throughout survey has revealed that various methods and combination of these methods can be applied in development of a new face recognition system. Among the many possible approaches, we have decided to use a combination of knowledge-based methods for face detection part and neural network approach for face recognition part. The main reason in this selection is their smooth applicability and reliability issues. Our face recognition system approach



Input Part :

Input part is prerequisite for face recognition system. Image acquisition operation is performed in this part. Live captured images are converted to digital data for performing image-processing computations. These captured images are sent to face detection algorithm.

```
def Stu_register(self):
    if self.SeqA.get()=="" or self.SeqQ.get()=="Select" or self.Sturoll.get() == "" or
self.Stumail.get() == "" or self.Stuname.get() == "" or self.Stuph.get() == "" or
self.StuComDept.get() == "Select" or self.StuGen.get() == "Select" or
self.Stupass.get() == "" or self.StuConPass.get() == "" or self.Stucal.get() ==
"1/1/22":
        messagebox.showerror("Error", "All Fields Must be Filled!!", parent=self.root)
    elif self.Stupass.get() != self.StuConPass.get():
        messagebox.showerror("Error", "Entered Password are Diffrent",
parent=self.root)
    else:
        try:
            con = pymysql.connect(host="localhost", user="root", password="0000",
database="project")
            cur = con.cursor()
            cur.execute("select * from student where EMAIL=%s", self.Stumail.get())
            row = cur.fetchone()
            cur1 = con.cursor()
            cur1.execute("select EROLL from student where EROLL=%s",
self.Sturoll.get())
            row1 = cur1.fetchone()
            if row != None:
                messagebox.showerror("Error", "Email Address is Already Registred!!",
parent=self.root)
                self.Stumail.delete(0, END)
                self.Stumail.focus()
            elif row1 != None:
                messagebox.showerror("Error", "Enter RollNumber is Already
Registered!!", parent=self.root)
                self.Sturoll.delete(0, END)
                self.Sturoll.focus()
            else:
                cur2 = con.cursor()
                cur2.execute(
                    " insert into student
(EROLL,EMAIL,NAME,PHONE,DEPT,SEM,BIRTH,GEN,PASS,SEQQ,SEQA)values(%s,%s,%s,%s,%s,%s,%s,
%s,%s,%s,%s)",
                    (
                        self.Sturoll.get(), self.Stumail.get(), self.Stuname.get(),
self.Stuph.get(),
                        self.StuComDept.get(), self.StuSem.get(),
self.Stucal.get_date(), self.StuGen.get(),
                        self.Stupass.get(),self.SeqQ.get(),self.SeqA.get()))
                con.commit()
                con.close()
                messagebox.showinfo("Opening Camera for Taking Sample Images!!",
parent=self.root)
```

```
                # self.root.destroy()
                face_classifier =
cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
                # FACE CAPTURE
                try:
                    def face_cropped(img):
                        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
                        faces = face_classifier.detectMultiScale(gray, 1.3, 5)
                        for (x, y, w, h) in faces:
                            face_cropped = img[y:y + h, x:x + h]
                            return face_cropped

                    cap = cv2.VideoCapture(0)
                    img_id = 0
                    while True:
                        ret, my_frame = cap.read()
                        if face_cropped(my_frame) is not None:
                            img_id += 1
                        face = cv2.resize(face_cropped(my_frame), (450, 450))
                        face = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)
                        file_name_path = "Data/user." + self.Sturoll.get() + "." +
str(img_id) + ".jpg"
                        cv2.imwrite(file_name_path, face)
                        cv2.putText(face, str(img_id), (50, 50),
cv2.FONT_HERSHEY_COMPLEX_SMALL, 2, (0, 0, 255), 2)
                        cv2.imshow("Crooped Face", face)
                        if cv2.waitKey(1) == 12 or int(img_id) == 100:
                            break;
                    cap.release()
                    cv2.destroyAllWindows()
```

Training the Algorithm:

First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let’s see the LBPH computational steps.

```
 # Training Images
  data_dir = ("Data")
  path = [os.path.join(data_dir, file) for file in os.listdir(data_dir)]
  faces = []
  ids = []
  for image in path:
      img = Image.open(image).convert('L')
      imageNP = np.array(img, 'uint8')
      id = int(os.path.split(image)[1].split('.')[1])
      faces.append(imageNP)
      ids.append(id)
      # cv2.imshow("Traning Images", imageNP)
      cv2.waitKey(1) == 13
  ids = np.array(ids)
  clf = cv2.face.LBPHFaceRecognizer_create()
  clf.train(faces, ids)
  clf.write("Classifier.xml")
  # messagebox.showinfo("Sure", "Training Images completed!!!", parent=self.root)
  self.root.destroy()
cept Exception as es:
  messagebox.showinfo("Faulty", "Traning Imgae failed Try Again!!!")
  con = pymysql.connect(host="localhost", user="root", password="0000",
database="project")
  cur = con.cursor()
  cur.execute("delete from student where EROLL=%s", self.Sturoll.get())
  con.commit()
  con.close()
```

In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image.

```
def main_app(self):
    if not os.path.isfile("Classifier.xml"):
        messagebox.showinfo("INFO", "Please train the data first!!", parent=self.root)
    else:
        face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
        recognizer = cv2.face.LBPHFaceRecognizer_create()
        recognizer.read(f"Classifier.xml")
        cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)
        while True:
            ret, frame = cap.read()
            # default_img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            faces = face_cascade.detectMultiScale(gray, 1.3, 5)
            for (x, y, w, h) in faces:
                roi_gray = gray[y:y + h, x:x + w]
                id, predict = recognizer.predict(roi_gray)
                confidence=((100*(1-predict/300)))
                if confidence>80:
                    con =
pymysql.connect(host="localhost",user="root",password="0000", database="project")
                    cur = con.cursor()
                    cur.execute("select * from attendence where EROLL=%s AND
DATE=CURDATE();", (id,))
```

```
                    result = cur.fetchone()
                    if result is None:
                        con1 = pymysql.connect(host="localhost", user="root",
password="0000", database="project")
                        cur1 = con1.cursor()
                        sql = "insert into
attendence(eroll,name,phone,mail,gen,attend,date,dept,sem,time)select
eroll,name,phone,email,gen,\"P\",curdate(),dept,sem,curtime() from student where eroll=
%s"
                        val = (id)
                        cur1.execute(sql, val)
                        con1.commit()
                        con1.close()
                    else:
                        cur.execute("Select NAME from student where EROLL=%s", (id,))
                        i = cur.fetchone()
                        i = "+".join(i)
                        cur.close()
                        font = cv2.FONT_HERSHEY_PLAIN
                        frame = cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255,
0), 2)
                        frame = cv2.putText(frame, i, (x, y - 4), font, 1, (0, 255,
0), 1, cv2.LINE_8)
                else:
                    text = "UnknownFace"
                    font = cv2.FONT_HERSHEY_PLAIN
                    frame = cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0,
255),2)
                    frame = cv2.putText(frame, text, (x, y - 4), font, 1, (0, 0, 255),
1, cv2.LINE_AA)
            cv2.imshow("image", frame)
            if cv2.waitKey(1) == 13:
                break
        cap.release()
        cv2.destroyAllWindows()
```

So to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.

We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: euclidean distance, chi-square, absolute value, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula:

$$D = \sqrt{\sum_{i=1}^{n}(hist1_i - hist2_i)^2}$$

So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a 'confidence' measurement. Note: don't be fooled about the 'confidence' name, as lower confidences are better because it means the distance between the two histograms is closer.

We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

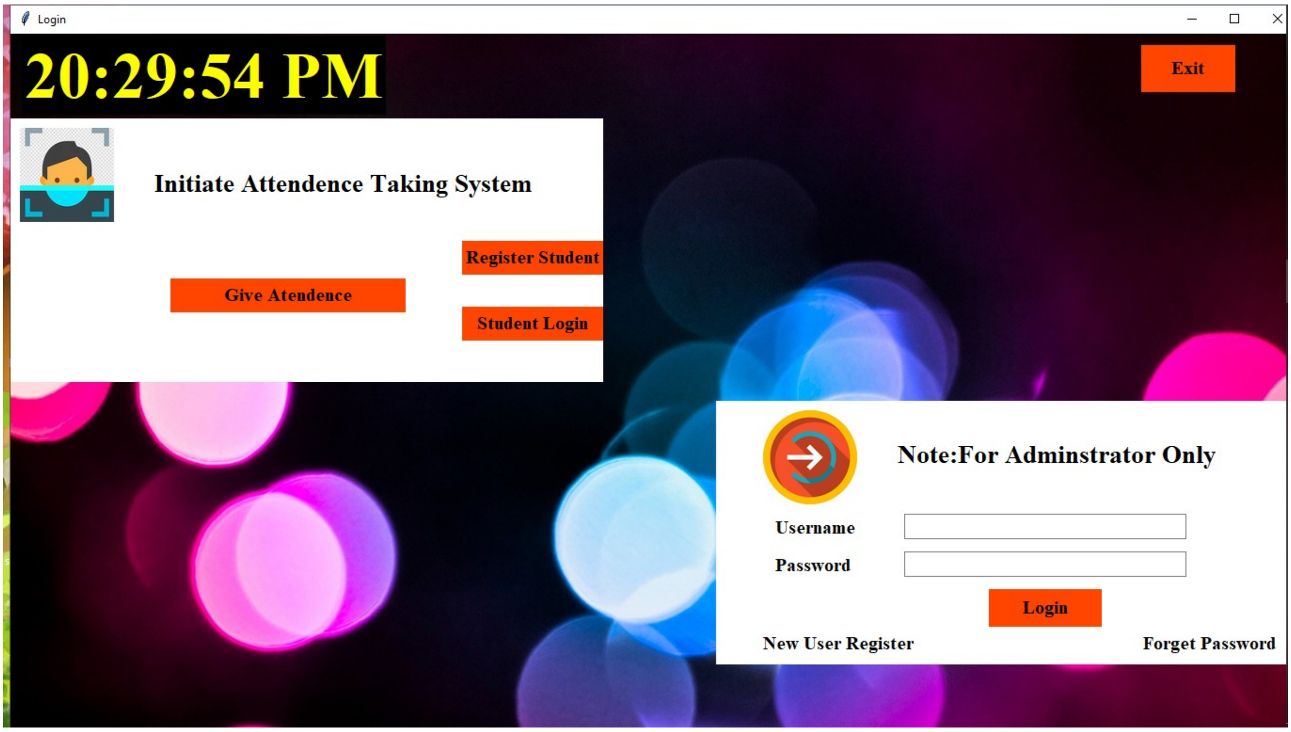
Showing Data in the Dashboard:

The marked attendance is fetched from database and showed into both faculty and student's dashboard.

```
#table Building
scrool_x = ttk.Scrollbar(UpperLabel, orient=HORIZONTAL)
scrool_y = ttk.Scrollbar(UpperLabel, orient=VERTICAL)
self.Student_Table = ttk.Treeview(UpperLabel, columns=("roll", "Na", "dept", "Sem",'time'),
                      xscrollcommand=scrool_x.set, yscrollcommand=scrool_y.set)
scrool_x.pack(side=BOTTOM, fill=X)
scrool_y.pack(side=RIGHT, fill=Y)
scrool_y.config(command=self.Student_Table.yview)
scrool_x.config(command=self.Student_Table.xview)
self.Student_Table.heading("roll", text="Exam Roll No")
self.Student_Table.heading("Na", text="Name")
self.Student_Table.heading("dept", text="Departmet")
self.Student_Table.heading("Sem", text="Semester")
self.Student_Table.heading("time",text="Time")
self.Student_Table['show'] = "headings"
 self.Student_Table.column("roll", width=100, minwidth=100, )
self.Student_Table.column("Na", width=100, minwidth=100)
self.Student_Table.column("dept", width=100, minwidth=100)
self.Student_Table.column("Sem", width=100, minwidth=100)
self.Student_Table.column("time",width=100,minwidth=100)
self.Student_Table.pack(fill=BOTH, expand=1)
self.autoShow()
####################################        FUCNTION FOR AUTO SHOW TODAY'S ATTENDE
def  autoShow(self):
try:
  self.Student_Table.delete(*self.Student_Table.get_children())
  conn = pymysql.connect(host="localhost", user="root", password="0000", database="project")
  curr = conn.cursor()
```

```
    curr.execute(
      "select EROLL,NAME,DEPT,SEM,TIME from Attendence where date=curdate()")
    values = curr.fetchall()
    if len(values) == 0:
      messagebox.showerror("Error", "Attendence For Today not marked!!", parent=self.root)
    else:
      for i in values:
        self.Student_Table.insert("", END, values=i)
    conn.commit()
    conn.close()
  except Exception as es:
    messagebox.showerror("Error", f"error due to:{str(es)}", parent=self.root)
```

Snapshot of Running Project:

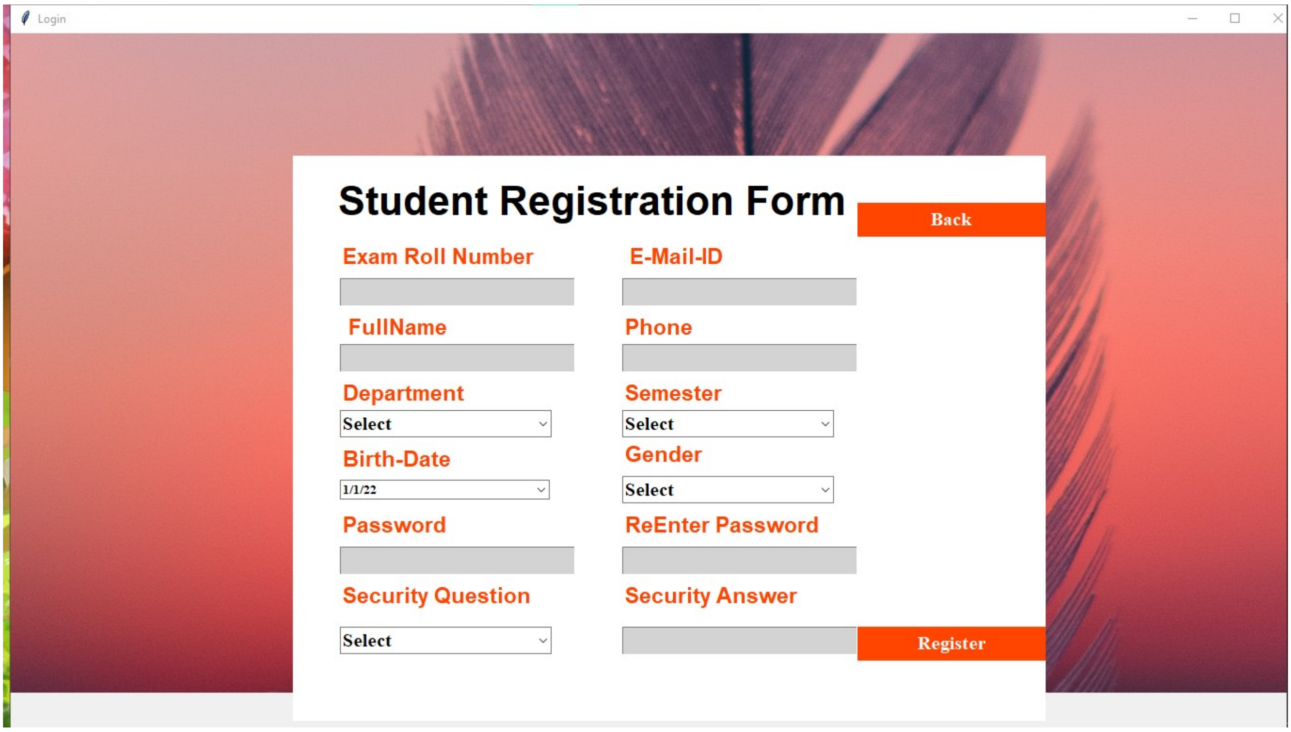


The figure shows a snapshot of the running of an actual project. The first interface will look like this after you run the project.

Upon describing this frame interface , A digital running time clock, an attendance making button, and a login for both the student and the administrator.

As stated, different buttons perform different functions.

This frame implements the logic of easy-to-use conditions. Our main goal is to provide the training to the other end person by looking at the first frame and gauging what it's all about

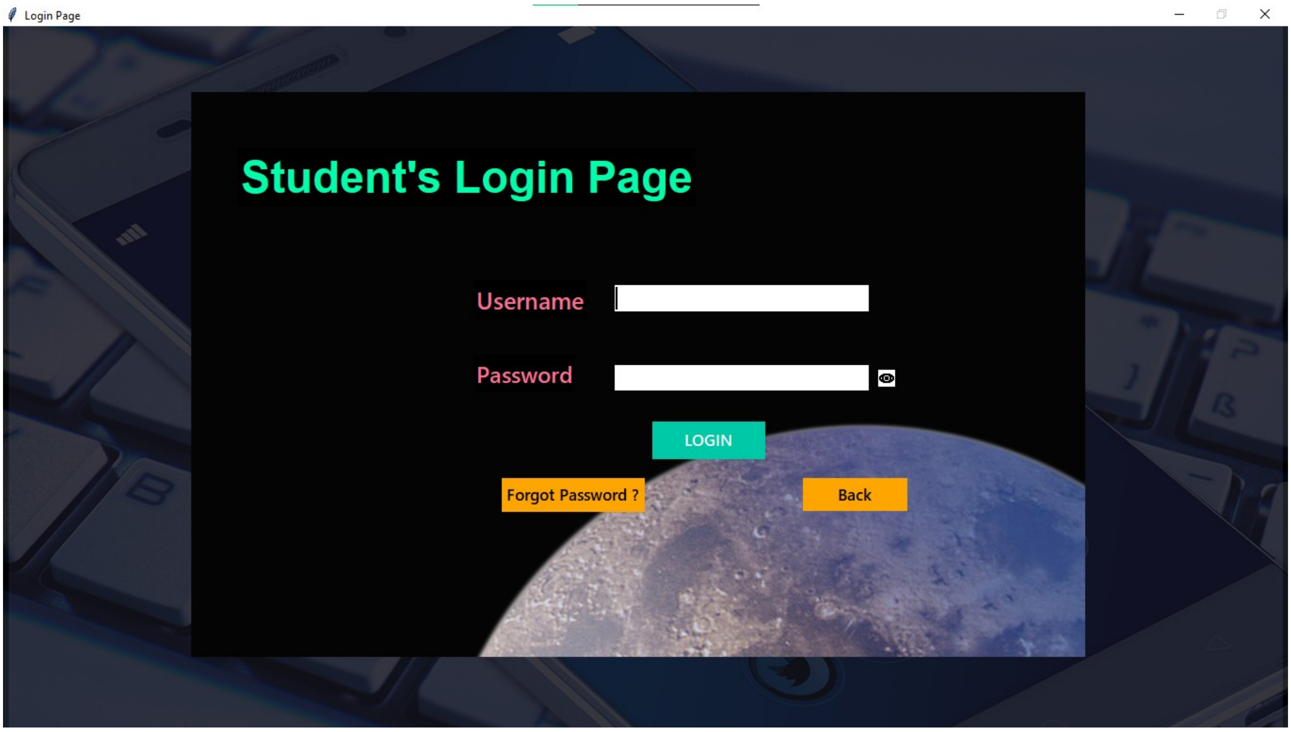


A typical registration form for the student's

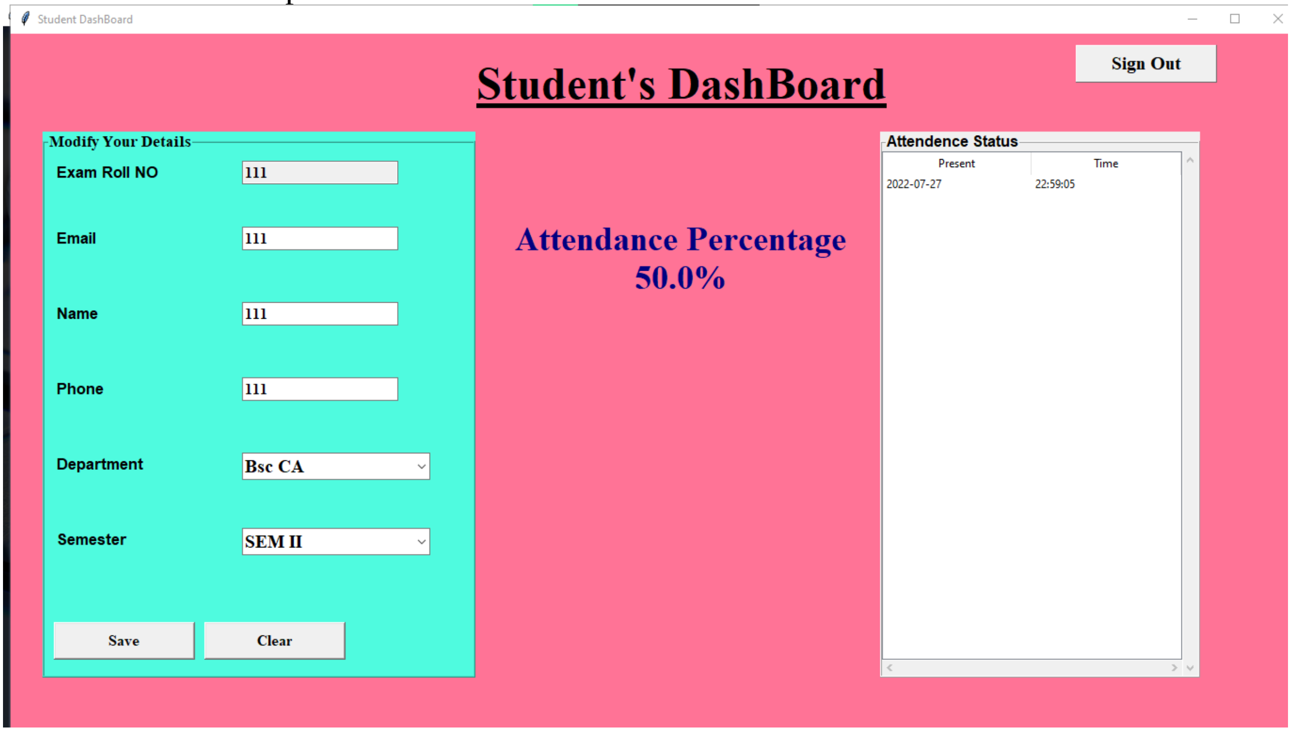
This registration form performs an action of adding the student to the system database. This is an on-time check registration based on no data redundancy.

Along with successful registration, the data is inserted and the camera is triggerd to take the sample images for traning and detecting images later on.

If it fails to detect any images, the system suspends the insertion and flags a popup that the face is not detected.Befor opening the camera the system alerts you that put your face in front of camera at compertable angle to take photo samples.



Student login frame here studnet can login to there respective dashboards . To see the attendence of
of their own they must put right username and password.After getting right they are been redirected to their repective dashboards. Which looks like this.



Student DashBoard

Student's DashBoard

Modify Your Details

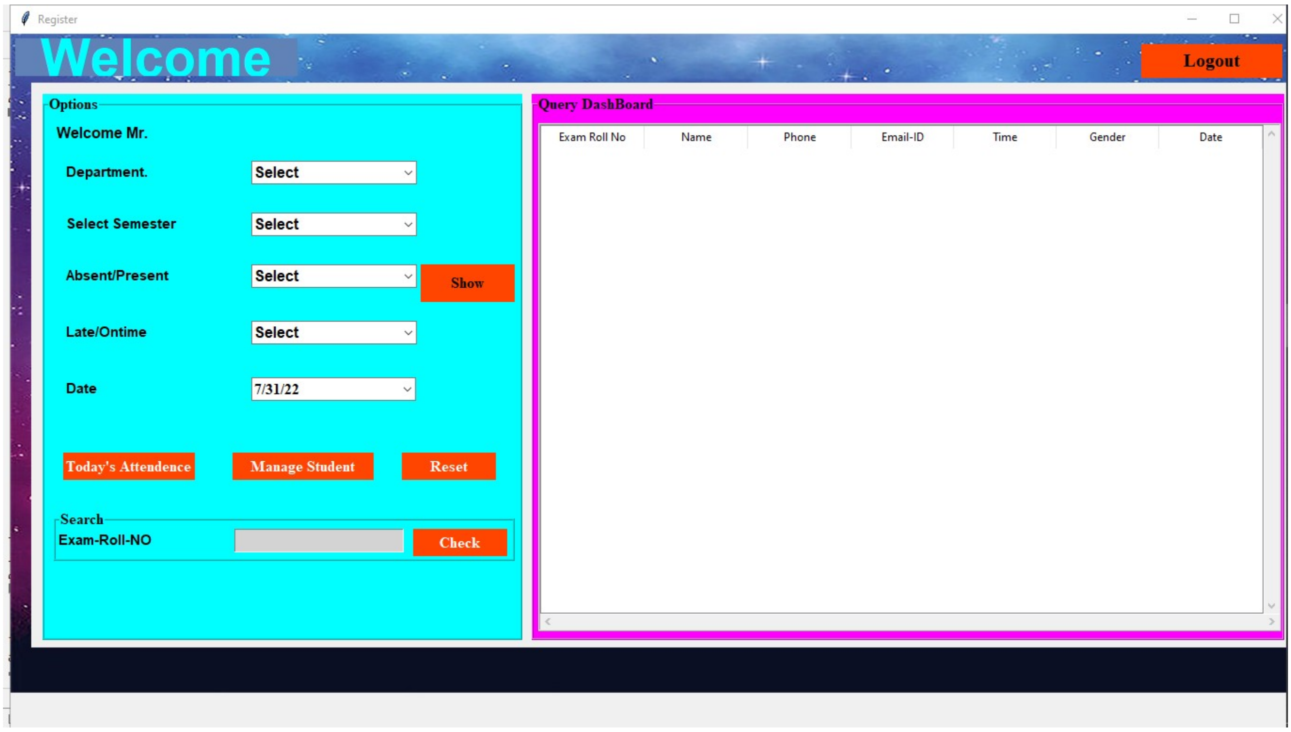
Exam Roll NO	111
Email	111
Name	111
Phone	111
Department	Bsc CA
Semester	SEM II

Attendance Percentage
50.0%

Attendence Status

Present	Time
2022-07-27	22:59:05

A Dashboard of Students Where they can see their date and time of marked attendance, and they can also modify the details of their own.
It is shown that they can also see their overall percentage of attendance for this month.



Register

Welcome

Options

Welcome Mr.

Department.	Select
Select Semester	Select
Absent/Present	Select
Late/Ontime	Select
Date	7/31/22

Search
Exam-Roll-NO

Query DashBoard

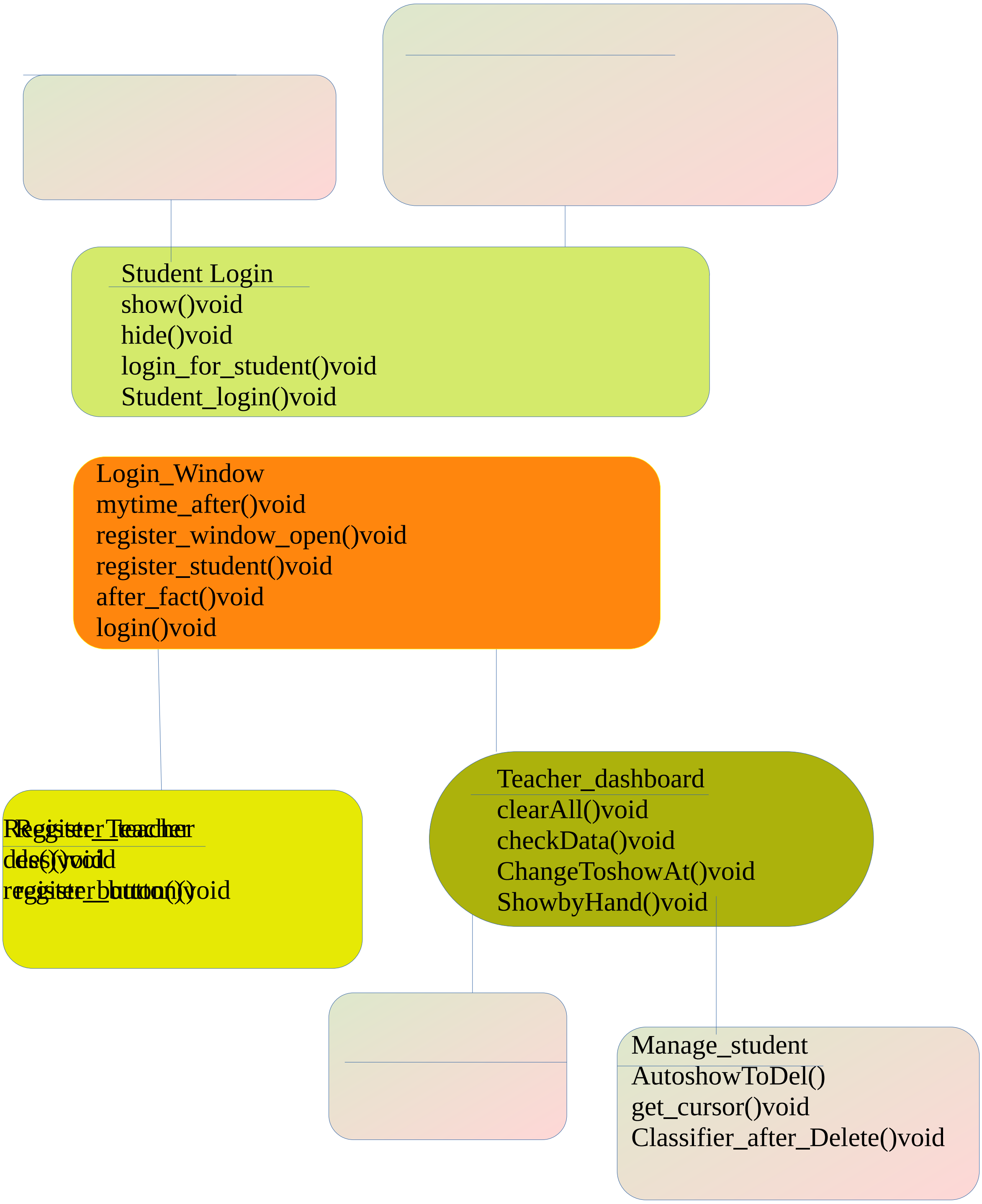
Exam Roll No	Name	Phone	Email-ID	Time	Gender	Date

A dashboard ot Administrator .

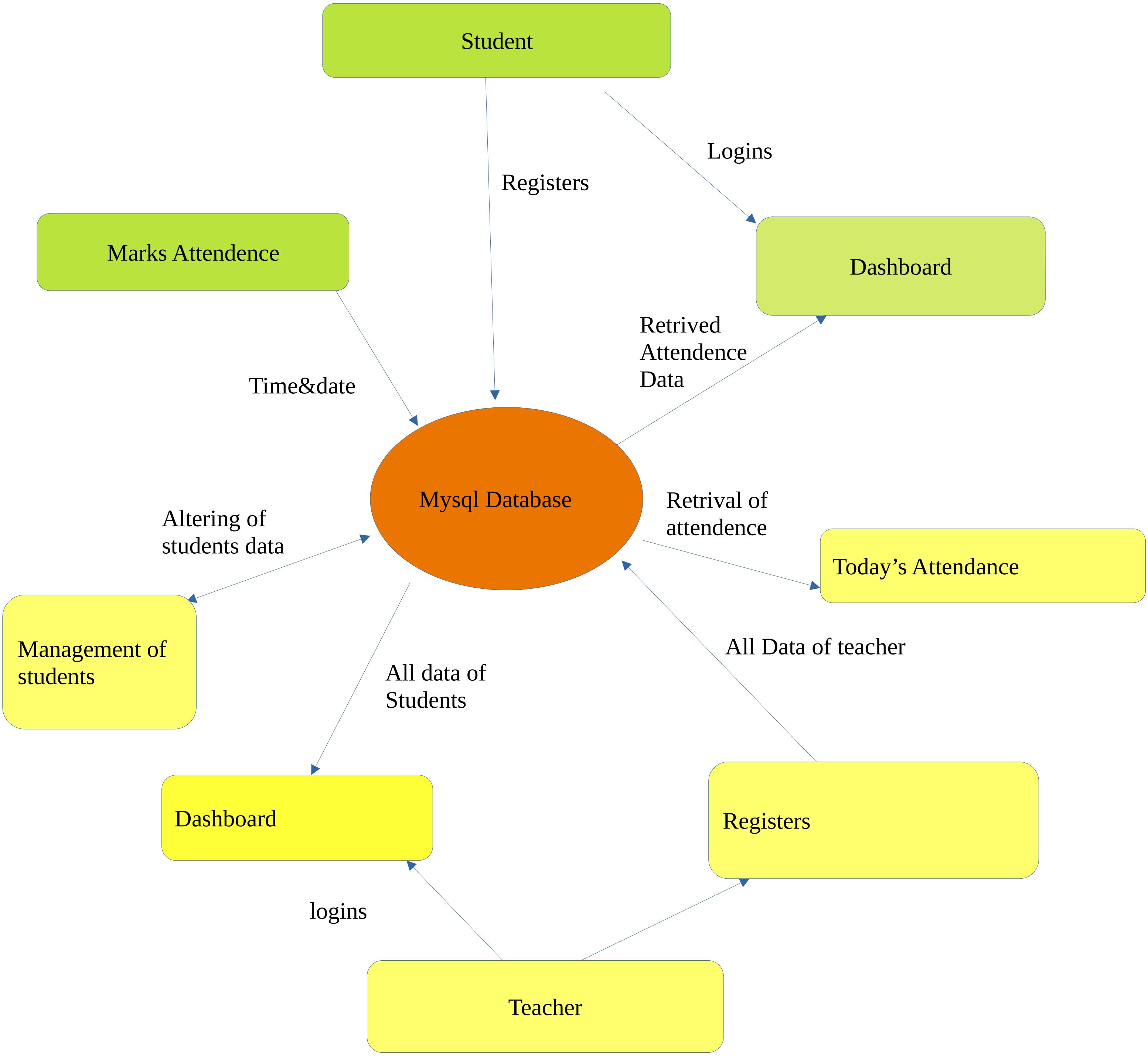
A dashboard where the administrator can see and manipulate students;
There is a special frame for the view of students who are present today.The administrator can only delete students .it can not alter the details of the student. It can see the student who were absent today also..

Face Detection:

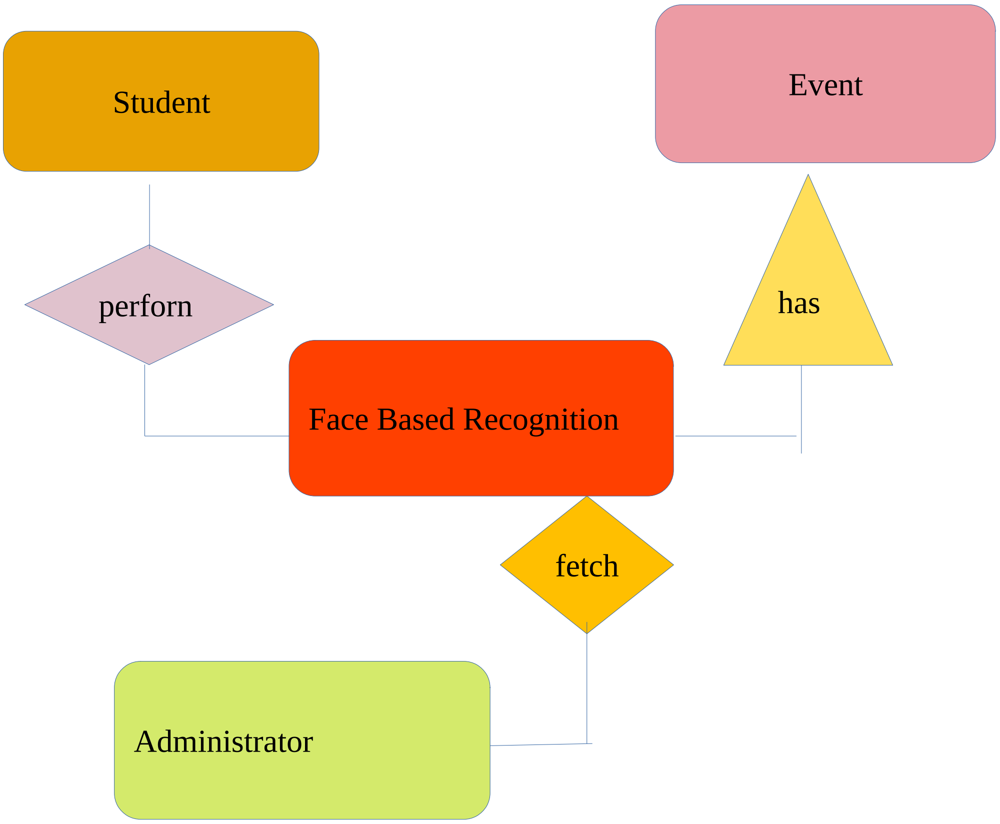
Class Diagram:

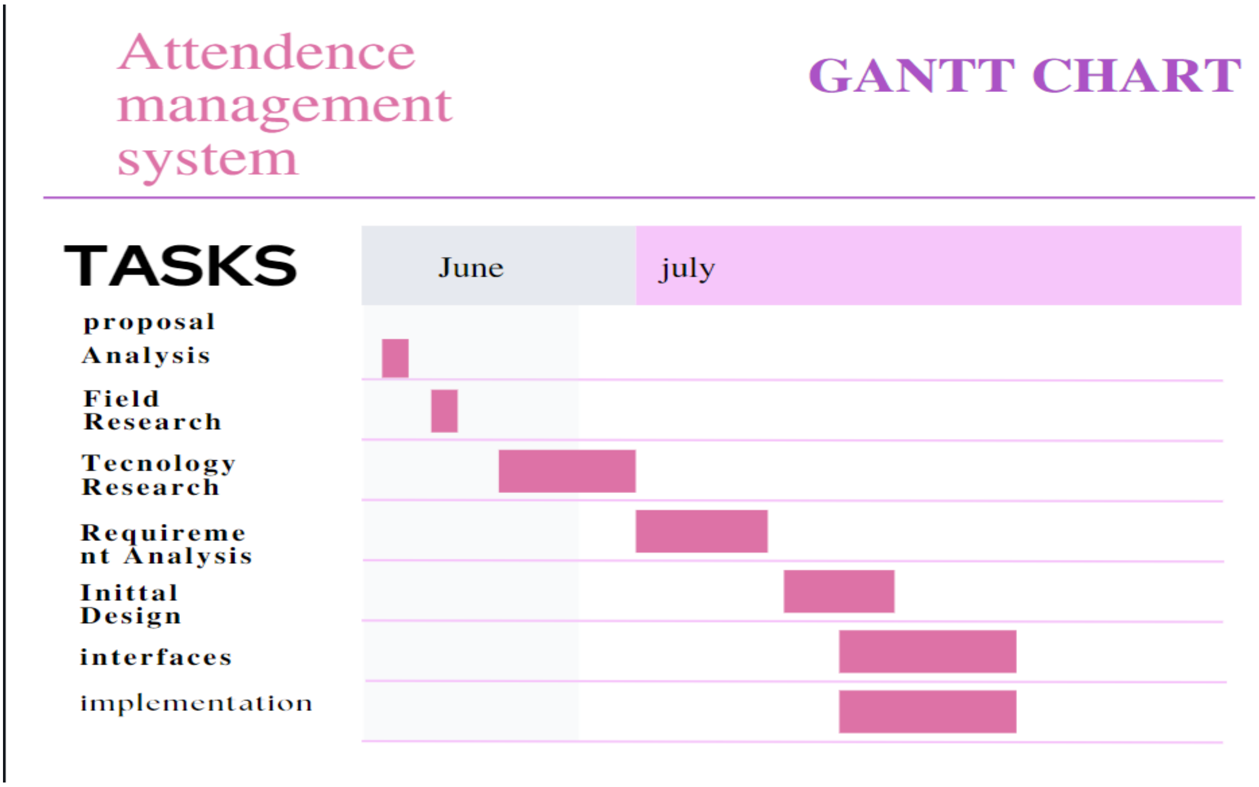


Data Flow Diagram:

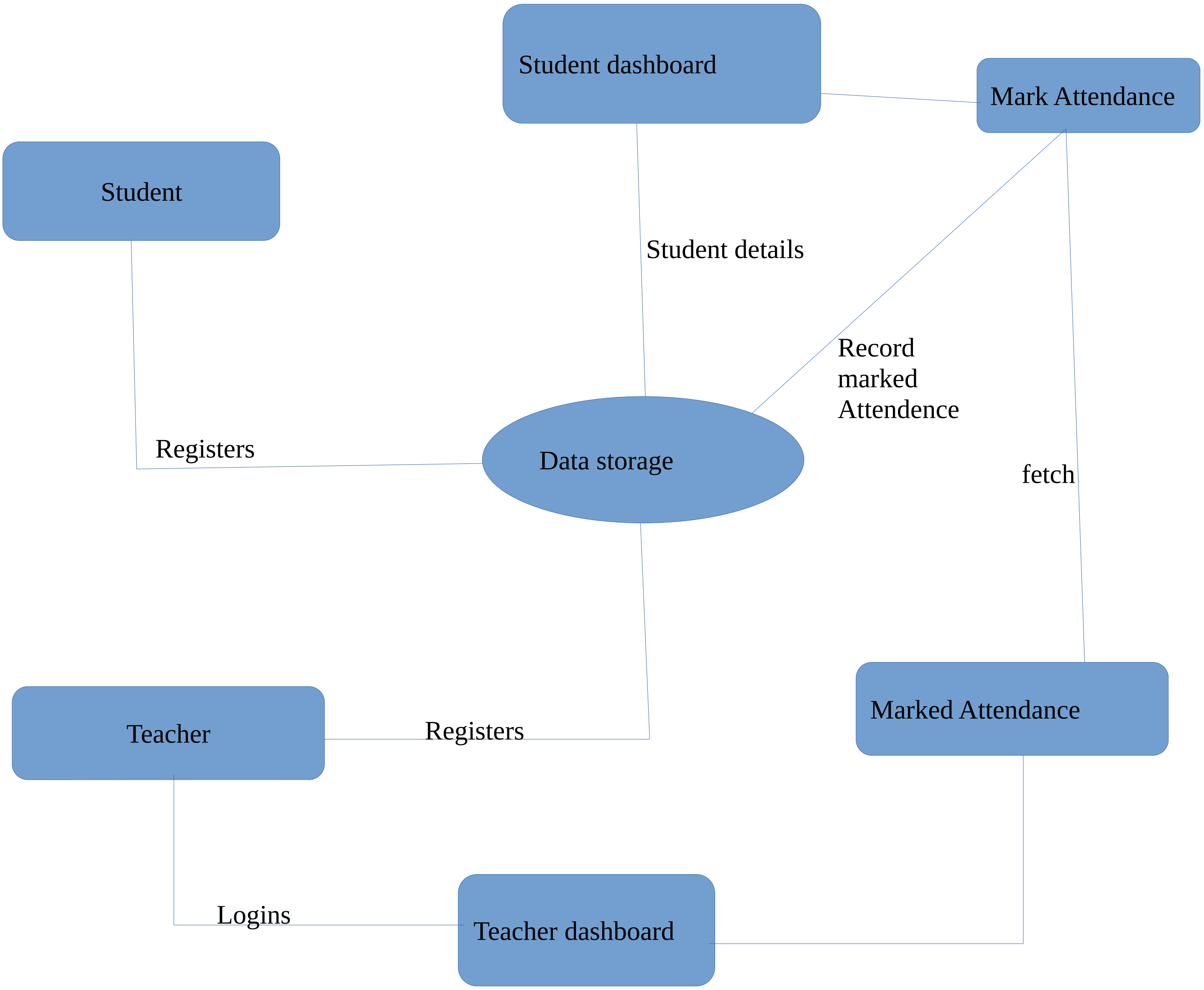


An E-R diagram:





Activity Diagram:



Its relevance and implication in company:

Law Enforcement

As the government adds a layer of artificial intelligence to its surveillance, startups are playing a key role in providing it with the underlying facial recognition technology. Chinese unicorns like SenseTime, Face++, and CloudWalk, for example, are already working with the Chinese government here.
But even in the United States, interest in the tech is surging, according to the CB Insights patent analysis tool.

Amazon, for example, is selling its tech to law enforcement agencies.
Academic institutions like Carnegie Mellon University are also working on technology to help enhance video surveillance. The university was granted a patent around “hallucinating facial features” — a method to help law enforcement agencies identify masked suspects by reconstructing a full face when only the periocular region of the face is captured. Facial recognition may then be used to compare the “hallucinated face” to images of actual faces to find ones with a strong correlation.
Because this technology is still relatively young, the algorithms haven’t learned enough about the nuances between faces and skin tones, among other things, to provide the total accuracy that law enforcement teams need.
Broadly, facial recognition tech is not free of flaws. Amazon, for example, was in the news for reportedly misidentifying some Congressmen as criminals.
But the tools have the ability to improve. For instance, Amazon was granted a patent that explores additional layers of security, including asking users to perform certain actions like “smile, blink, or tilt his or her head.” These actions can then be combined with “infrared image information, thermal imaging data, or other such information” for more robust authentication.

2. Healthcare

In its most straightforward application, facial recognition has the potential to make it easier for patients to sign in at the doctor or hospital without waiting in line or juggling forms. But there are a number of other ways that the technology can be used to improve services.
Since 2015, Apple has launched two open-source frameworks — ResearchKit and CareKit — to help clinical trials recruit patients and monitor their health remotely. Researchers at Duke University, for example, developed an Autism & Beyond app that uses the iPhone’s front camera and facial recognition algorithms to screen children for autism.
A more futuristic use case is passive monitoring of healthcare biometrics, or extracting changes to facial features over time.

In December 2017, a Google patent was published with an ambitious vision for analyzing cardiovascular function from a person's skin color or skin displacement.

Amazon applied for a similar patent for passive monitoring in 2014, which was later granted in 2017. It combines recognition of facial features (using neural nets or other algorithmic approaches) with heart rate analysis.

For example, algorithms can track color changes in two areas of the face, like regions near the eyes and cheek, using that data to calculate heart rate detection. Tracking these changes could better position hospital staff to more effectively monitor and treat patients.

3. Retail

Linking facial recognition to personalization for shoppers presents a big opportunity in retail.

Facial recognition, for example, could capture what a shopper is looking at, and enable retailers to later serve up related promotions to the shopper via email or online ads. In another example, Walmart has a patent on tech that would capture and analyze facial expressions of people waiting in lines to gauge their satisfaction.

In China, users scan a QR code while entering JD stores. A camera runs facial recognition algorithms to identify the shopper during entry. Every item in JD's physical store carries an RFID tag. While exiting, customers stand at a "stand here" sign marked on the floor, where all the RFID tags are scanned at once, and cameras run facial recognition algorithms again to charge your account.

Facial recognition, along with AR, is also making brands data rich — especially beauty brands.

AR beauty company Modiface's tech collects a variety of data points around facial characteristics, including face shape, skin tone, wrinkles, and more. This can help retailers determine how people with specific facial characteristics may be more likely to purchase certain types of products, thereby potentially predicting inventory with greater accuracy.

Estée Lauder-owned Smashbox partnered with Modiface to use customer eye tracking insights to heat map the areas on a screen receiving more attention by users. This helps Smashbox understand which features are the most interesting and iterate on its website to make the beauty shopping experience more relevant for consumers.

4. Hospitality

Facial recognition is linked to better customer service in the hospitality industry. Tying the technology to a guest's account allows employees to offer them a better, more personalized experience.

Starting with a photo on one's account, for example, this technology could allow a guest to simply walk into a hotel to activate her check-in. The person could use her image to enter her room as well.

This could potentially free up the concierge staff to better serve their guests with suggestions, perks, and services tailored to them.

Currently, technology like this is being used in at least two Marriott hotels in China. Guests check in at a kiosk using their face. The process takes about a minute, and once they check in, their payment is automatically charged to their account.

5. Marketing & Advertising

Ad targeting has become more and more precise. Facial recognition could take that precision bounds forward.

It could be used to gather demographic information about people who stop and look at ads or to enhance the immersive ad experience for viewers. It has surfaced in a number of campaigns to date Virgin Mobile used it in 2013 in an interactive ad, where a viewer's blink would advance the ad's story. Italian company Almax implemented it in mannequins that would scan the faces of passersby who stopped to look at products.

Going forward, with facial recognition built into most phones, tablets, and computers, it could be used to analyze facial expressions and serve up different ads based on mood. It could also put actual weight behind an ad view or impression, which is currently a shaky metric.

6. Banking

Banks are beginning to use facial recognition as secure login tools for customers using their online banking sites and apps. HSBC and Singapore's OCBC have both added facial biometrics as options for logging into mobile banking accounts. The technology is said to offer "less than a one-in-a-million chance of mistaken identity," potentially making online banking more secure.

The tech isn't limited to protecting online data: CaixaBank in Spain is using facial recognition at its ATMs to provide bank customers with an extra layer of security. Face scanning helps to prevent fraud by eliminating the need for a PIN or even a card swipe, both of which are susceptible to being stolen.

As the technology develops and becomes more accessible, it could also make brick-and-mortar banking more secure and personalized. Tellers would know customers' names and account information, offering a better and less clunky customer experience.

7. Events

Last year, a company called Blink Identity received $1.5M from investors including Live Nation (which owns Ticketmaster) and Techstars Ventures. The company uses facial recognition to identify people entering an event without a ticket, reportedly in less than a second.

Other recognition technologies are also digging into this space, offering ways to recognize VIPs like all-access or season ticket holders at events.

Of course, this requires stadiums, arenas, concert halls, and other venues to be equipped with facial recognition cameras or kiosks, and would still require some crowd control, including ways to remove those who aren't authorized to be there.

8. Social Media/Entertainment

Fighting boredom is yet another use case for facial recognition. Apps like CelebsLike.Me, which uses a person's photo to find celebrities that look like them, and Google's Art Selfie, which analyzes a selfie to find art pieces that match facial structures, are making the rounds on social networks.

Snapchat was a pioneer of facial recognition as a personal entertainment tool. Filters and Lenses take a scan of the subject's face, then apply things like makeup or rabbit ears. These tools also know when the user's mouth opens, for example, and can animate dog tongues or rainbows.

Snapchat also gave rise to Bitmoji (which doesn't yet use facial recognition, but could in the future), which inspired Apple's Animoji and Memoji. Within iMessage, users can create 3D emojis or animations of themselves that mirror their facial expressions, thanks to the iPhone's Face ID technology.

9. Air Travel

Similar to the advances possible in hotel hospitality and entertainment, travel could see big changes by implementing facial recognition technology. TSA lines could be expedited for everyone and VIP passengers could have an even more tailored experience.

JetBlue is testing facial recognition instead of boarding passes to streamline the process. Kiosks use passport or ID photos from the US customs database to verify passengers and let them aboard. Last year, Delta also tested face-scanning kiosks to help passengers check their own luggage.

Biometric technology in airports is also a win for security. Last fall, facial recognition stopped a man trying to use a fake passport to enter the US. Customs officials say that these cameras have a 99% accuracy rate, as they compare photos of people's faces on file from official government documents.

10. Automobiles

While self-driving cars are the focus of the facial recognition research in the auto industry currently, there is still potential for other use cases. Face scanning could unlock the car when the owner approaches or function as the key to starting and stopping the car (much like button-start cars function now).

Cars that are have programmable settings for different drivers could make these even simpler with face recognition. Seat distance, car temperature, preferred radio stations could switch instantly when one driver's face is scanned versus another.

Hyundai's self-driving concept car, on display at this year's CES, used facial recognition and AI to "asses the driver's emotional state" and change the lighting to help avoid accidents caused by distracted driving.

11. **Gambling & Casinos**

In casinos, face scanning technology is most associated with security. By scanning and recognizing those who come in and gamble, they can spot regulars who might get VIP treatment, cut down on people known for cheating the house, and reduce incidents resulting from upsetting losses.

Recently, it has also emerged as a data intelligence solution going beyond basic security. It helps provide insight on table games so that companies can optimize their casinos for maximum profit.

12. **Voting**

Biometrics would make voter fraud, complicated voter registration, and other limiting factors that keep people from the polls things of the past.

During last year's midterm elections, West Virginia allowed citizens to use the Voatz app to vote. First, voters had to take a photo of their government ID and a selfie-style video of their face, which was then uploaded to the app to verify. When a person was approved, they could cast their vote through the mobile app.

The app also uses blockchain to safely store information, creating a secure and digital-first solution to an international issue.

13. **Education**

Face scanning is in use at a few schools to make the campuses safer from potential threats. Carnegie Mellon University, for example, is working on technology to help enhance video surveillance.

Additionally, new technology called SAFR from RealNetworks uses facial recognition cameras to help improve safety in schools and streamline guest sign-in & campus security. The technology is being tested in two schools in Seattle, where visitors smile at a camera to

enter the school. It analyzes their face, identifies them, and allows the visitor onto the campus.
In Australia, facial recognition is serving a different purpose: taking attendance. A company called LoopLearn is helping select schools to keep track of students and cut down on the amount of time spent marking attendance reports.

14. Ride-Hailing

Facial recognition technology adds another layer of verification across the sharing economy. Singapore-based ride-hailing company Grab has partnered with Microsoft to incorporate facial recognition into accurately identifying drivers and passengers. Uber has also used Microsoft's technology to identify drivers and reduce safety and security incidents.
In India,Uber drivers use a selfie or face-scanning app on their phone to access their account, verifying that they are who they say they are. This information can then be passed on to the rider, assuring them that they are driving with a verified professional.
Passengers could do the same, using facial identification to open their account and hail a car. When the driver picks them up, they are shown as verified.
In the future, as cars are implemented with face scanning technology, this can go a step further, recognizing the passenger's face as they enter the car to verify that they are the person who called the car.

15. Food & Beverage

A number of restaurants are using face scanning tech to improve customer experience, especially for repeat visitors.
At a California restaurant CaliBurger, facial recognition connects a patron to their loyalty program account. They can see what they've ordered before and receive special discounts, all by approaching a kiosk that scans their face.
Something similar was tested by a KFC in China. Using facial detection and a mobile verification, customers could smile to pay for a meal.
As a person orders, these different kiosks scan their face and take a photo, which is stored with the order data. The next time that person comes in, the kiosk and staff can greet them by name and suggest orders they might enjoy.

16. Consumer Electronics

Already, face scanning is used by companies like Microsoft, Samsung, and Apple to unlock devices like phones, tablets, and computers. In fact, 54% of Americans either already use a device with facial recognition built in or plan to use one to protect their personal data.
From cars to kitchen appliances, other consumer electronics aren't far off. For example, the HiMirror, originally launched in 2016, offers a voice-controlled smart mirror facial

recognition system that provides beauty advice for users. The HiMirror integrates AI to offer a daily skin analysis and personalized recommendations as well as augmented reality to allow users to virtually try on makeup.

Use cases could also be tied to consumer wearables. OrCam, for example, uses facial recognition software to help people who are blind or partially sighted. The device uses a wireless smart camera to read text and identify faces, among other things.

By implementing facial scans into the smart, connected devices that people already use regularly, personalization and privacy can be woven more deeply into the day-to-day.

17. Monitoring employee productivity

The best facial recognition software can potentially replace other forms of biometric tracking like fingerprint scanning in the workplace as they offer several advantages. The greatest case for using facial recognition software to track ‘in’ and ‘out’ times of employees is the sheer convenience. Employees do not have to scan their fingerprints or ID cards to clock hours, the system will automatically identify when they come in and leave and log their hours accordingly.

The second advantage is that facial recognition software virtually eliminates the risk of incorrect entries. With fingerprint scanners, for instance, there are many times when employees leave in groups and do not individually scan their fingerprints. In the case of ID cards, the cards can simply be shared with someone else to scan it at certain times. Facial recognition software, on the other hand, will accurately track an employee’s working hours and can help you manage the productivity of your company as a whole.

18. More advanced security features

Recently, cybercrime has been on the rise, forcing many companies to improve their cybersecurity measures. One of the most common ways of protecting sensitive data from a cyber attack is by using complex passwords that can prevent a hacker from deciphering it and gaining access to a system. But even the strongest of passwords are still vulnerable to threats.

Facial recognition applications, on the other hand, are a much more secure alternative. Not only are they simple to use, in that a user doesn’t have to remember their passwords, but they are almost impossible to hack. Only the user whose face has been authorized to log into a system can gain access to it. This makes facial recognition technology extremely important for businesses which either deal directly with large volumes of sensitive data or are gatekeepers to private data.

This system aims to build an effective class attendance system using face recognition techniques. The proposed system will be able to mark the attendance via face Id. It will detect faces via webcam and then recognize the faces. After recognition, it will mark the attendance of the recognized student and update the attendance record.

Thus, the aim of this paper is to capture the video of the students, convert it into frames, relate it with the database to ensure their presence or absence, mark attendance to the particular student to maintain the record. The Automated Classroom Attendance System helps in increasing the accuracy and speed ultimately achieve the high-precision real-time attendance to meet the need for automatic classroom evaluation.

The proposed method uses face detection and face recognition that helps to maintain the automated attendance system. For detection, Paul–Viola Jones algorithm is used and for face recognition Linear Binary Pattern Histogram (LPBH) algorithm is applied. In the result, the unique ID and name of the student is displayed along with the confidence percentage. Confidence percentage represents the distance between the histogram of the stored image and histogram of the real time image and is calculated by using Euclidean distance. Lower is the distance, higher is the recognition rate.

In this approach a face recognition based automated student attendance system is throughly descrived.The proposded approached provides a method to identify the individuals by comparing their input image obtained from recording video frame with respect to train image. This proposed approach able to detect and localize face from a input facial image, which is obtained from the recording video fram, beside it proves a methodin pre-processing stage to enhace the image contrast and reduce the illluminatio effect. Extracton of features from the facial image is performed by applying both LBP and PCA. The algorithm designed to combine LBP and PCA able to stablize the system by giving consitent results. The accuracy of this proposed approach is 100% for high quality images ,92.31% for low quality images and 95.76% of Yale face database when two imagges per person are trained. As a conclusion for analysis, the extraction of facial featrue could be challengeing especially in diffrent lighting. In pre-processing stage, contranst Limited Adamptive Histogramin Equlizatoin(CLAHE) able to reduce the illumination effect. CLAHE perform better compared to histogram equalization in terms of contrast improved. Enhanced LBP with larget radius size specialfically, radius size two perform better compared to original LBP operator ,with less affected by illumination and more consistent compared to other radius sizes.

In order to maintain this attendance system , this system has been proposed . It replaces the manual system with an automated system which is fast, efficent , cost and time saving as replaces the stationary material and the paper work. Hence this system is expected to give desired results and in future could be improved for logout. Also , The efficiency could be improved by integrating other techniques with it near time future. In this system we have implemented an attendance records to assist faculty. It saves time and effort espeicially if it is a lecure with huge number of students. Automated attendance system has been envisoned for the purposes os reducing the drawbacks in the traditional (manual) system.

Limitation of the project:

1. Poor Image Quality Limits Facial Recognition's Effectiveness

Image quality affects how well facial-recognition algorithms work. The image quality of scanning video is quite low compared with that of a digital camera. Even high-definition video is, at best, 1080p (progressive scan); usually, it is 720p. These values are equivalent to about 2MP and 0.9MP, respectively, while an inexpensive digital camera attains 15MP. The difference is quite noticeable.

2. Small Image Sizes Make Facial Recognition More Difficult

When a face-detection algorithm finds a face in an image or in a still from a video capture, the relative size of that face compared with the enrolled image size affects how well the face will be recognized. An already small image size, coupled with a target distant from the camera, means that the detected face is only 100 to 200 pixels on a side. Further, having to scan an image for varying face sizes is a processor-intensive activity. Most algorithms allow specification of a face-size range to help eliminate false positives on detection and speed up image processing.

3. Different Face Angles Can Throw Off Facial Recognition's Reliability

The relative angle of the target’s face influences the recognition score profoundly. When a face is enrolled in the recognition software, usually multiple angles are used (profile, frontal and 45-degree are common). Anything less than a frontal view affects the algorithm’s capability to generate a template for the face. The more direct the image (both enrolled and probe image) and the higher its resolution, the higher the score of any resulting matches.

4. Data Processing and Storage Can Limit Facial Recognition Tech

Even though high-definition video is quite low in resolution when compared with digital camera images, it still occupies significant amounts of disk space. Processing every frame of video is an enormous undertaking, so usually only a fraction (10 percent to 25 percent) is actually run through a recognition system. To minimize total processing time, agencies can use clusters of computers. However, adding computers involves considerable data transfer over a network, which can be bound by input-output restrictions, further limiting processing speed.

Ironically, humans are vastly superior to technology when it comes to facial recognition. But humans can only look for a few individuals at a time when watching a source video. A computer can compare many individuals against a database of thousands.

MORE FROM FEDTECH: Find out how DHS has embraced cloud-based biometric verification.

How to Overcome Limits on Facial Recognition Tools

As technology improves, higher-definition cameras will become available. Computer networks will be able to move more data, and processors will work faster. Facial-recognition algorithms will be better able to pick out faces from an image and recognize them in a database of enrolled individuals. The simple mechanisms that defeat today's algorithms, such as obscuring parts of the face with sunglasses and masks or changing one's hairstyle, will be easily overcome.

An immediate way to overcome many of these limitations is to change how images are captured. Using checkpoints, for example, requires subjects to line up and funnel through a single point. Cameras can then focus on each person closely, yielding far more useful frontal, higher-resolution probe images. However, wide-scale implementation increases the number of cameras required.

Evolving biometrics applications are promising. They include not only facial recognition but also gestures, expressions, gait and vascular patterns, as well as iris, retina, palm print, ear print, voice recognition and scent signatures. A combination of modalities is superior because it improves a system's capacity to produce results with a higher degree of confidence. Associated efforts focus on improving capabilities to collect information from a distance where the target is passive and often unknowing.

5. Greater threat to individual and societal privacy

The threat to individual privacy is a significant downside of facial recognition technology. People don't like having their faces recorded and stored in a database for unknown future use.

Privacy is such a big issue that some cities, including San Francisco, California and Cambridge, Massachusetts, have banned law enforcement's use of real-time facial recognition surveillance. In these cases, police can use video recordings from personally owned security video devices, but they can't use live facial recognition software.

6. Can infringe of personal freedoms

Being recorded and scanned by facial recognition technology can make people feel like they're always being watched and judged for their behavior. Plus, police can use facial recognition to run everyone in their database through a virtual criminal lineup, which is like treating you as a criminal suspect without probable cause.

7. Violates personal rights

Countries with limited personal freedoms, such as China, UAE, North Korea, Iran and Iraq, commonly use facial recognition to spy on citizens and arrest those deemed troublemakers.

8. Creates data vulnerabilities

There is also concern about the storage of facial recognition data, as these databases have the potential to be breached.

Limitation in the Face Detection

The Facial Recognition System is essential nowadays, and it has come a long way. Its use is essential in quite some applications, for example - Photo retrieval, surveillance, authentication/access, control systems etc. But there are a few challenges that have continuously occurred during image or face recognition system.

These challenges need to be overcome to create more effective face recognition systems. The Following are the challenges which affect the ability of Facial Recognition System to go that extra mile.

9.Illumination

The illumination plays an essential role during image recognition. If there is a slight change in lighting conditions, it will make major impact on its results. It is the lighting to vary, and then the result may be different for the same object cause of low or high illumination.

10.Background

The background of the object also plays a significant role in Face detection. The result might not the same outdoor as compared to what is produces indoors because the factor - affecting its performance-change as soon as the locations change.

11.Pose

The facial recognition system is highly sensitive to pose variations. The movement of head or different camera positions can cause changes of facial texture and it will generate the wrong result.

12.Occlusion

Occlusion means the face as beard, mustache, accessories (goggles, caps, mask, etc.) also interfere with the estimate of a face recognition system.

13.Expressions

Another important factor that should be kept in mind is the different expression of the same individual. Change in facial expressions may produce a different result for the same individual.

Finding:

“Experience is the name everyone gives to their mistakes.” – Oscar Wilde

That's an apt way to put it, to an extent. As a quote it's a funny equivocation, and for the most part there is a lot of truth to it…"Well touching that fire was an experience I won't be doing again" Was it a mistake? We might say yes, but it can't be "wrong" if we learned from it.

However, in a literal sense (I'm sure not the way he implied it) he is saying that mistakes aren't really ever wrong, only part of the learning curve. But you can only use a bad decision once, maybe twice as experience, after that it becomes a mistake, and after 4 times (arbitrary) it's an intentional action….which is what makes his quote all the more funny.

In the past month, I have been unable to aggregate my single thoughts about this project . I doubted whether I would be able to complete my project on time or not. Where do I start with? Where could I find the resources to complete my project? Then I came up with an idea to start with all I learned until now. This gave me the push to start with my project. designing a framework, correcting it again and again . Coping with syntax errors,fixing bugs. making the project function. There are various things I learned with my project . One of the things is, if you are stuck in any part that looks like you are unable to get forward , take a short break before giving up. Wanting to learn something that you have never experienced, in fact, pushed me to the envadour of experience.
I was a little upset because I was a programmer with little knowledge of programming languages. After thinking over and over, I came up with an idea to solve the problem that was building a small part of my project. This helps me a lot.

No matter how organized and productive we are, there is always going to come a project that will give us a really hard time. Now, we do not have the luxury to call the client and say that we are no longer going to do the project, so we have to finish it anyhow.A stuck project can have a lot of adverse effects depending on the organization or project size. If we consider a small size project, then, if the project is stuck, it will result in the delay of project completion, which will then lead to an unsatisfied client. which has many adverse effects surrounding it. The clients are not getting their home on time for which they have invested their hard-earned money, such a huge blockage in a developing country like India can hinder its economic growth and many such effects could be seen.Therefore, a stuck project is like a “Nightmare” for any Project Manager and the Organization. If you are feeling “stuck” on a project and are having a hard time figuring out your next step, try the solutions listed below:

Project management is a field where people skills can make or break a candidate. If you’re less familiar with the technical side of project management, emphasize people skills you have like leadership, communication, and organization. Even if you’re not a formal project manager yet, chances are good that you’ve done some elements of project management in the past. Go through your experiences and find moments when you’ve helped to improve, plan, or execute new processes.

And don’t worry—if you’ve landed an interview, your interviewer probably already knows that you don’t have formal experience but sees potential in you. Convey your enthusiasm for the job and willingness to learn.

Getting stuck on a project is quite a common scenario for anyone who specializes in project-based work. While a stuck project can be solved, a project should be well planned to avoid getting stuck.

FUTURE SCOPE

- The future scope of the project can be integrated with the hardware components for example GSM through which a monthly list of the defaulter students can be sent to the mentor. Additionally, an application can be developed to help students to maintain a track of their attendance. It can also be used in offices where a large group of employees sit in a hall and their attendance will be marked automatically by capturing a video but for this the accuracy of the recognition needs to be improved.

- The project is enhanced for accurate management without much further intervention from the client. By enhancing further by also giving the recognition process to teachers, we can get rid of both the login processes for faculty and students. If it were a student, it would give the option of logging in or marking attendance.

  If it were a faculty, it directly opens it to the respective teacher's dashboard.

- A mail which contains the information about absent as well as attendance percentage is mailed to the respective parents instead of messages using face recognition based attendance management system. Process of taking attendance is done module by module, it is possible to update attendance in one click or touch instead. Parents can also get the information about the internal assessment marks through messages if their children.

- We can make this an attendance system for both teachers and students also, which makes the total an automated attendance system.

Bibliography:

https://en.wikipedia.org/wiki/MySQL
https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html
https://hackr.io/blog/what-is-pycharm
https://www.jetbrains.com/help/pycharm/quick-start-guide.html#meet
https://www.superdatascience.com/blogs/opencv-face-recognition
https://scholar.google.co.in/scholar?q=Scale+Invariant+Feature+Transform+(SIFT)+(1999)&hl=en&as_sdt=0&as_vis=1&oi=scholart
Cahit GÜREL, cgurel@atilim.edu.tr ATILIM University, 06836, Ankara, Turkey Abdulkadir ERDEN, aerden@atilim.edu.tr ATILIM University, 06836, Ankara, Turkey
https://quillbot.com/grammar-check
https://www.researchgate.net/publication/328959960_Image_Training_and_LBPH_Based_Algorithm_for_Face_Tracking_in_Different_Background_Video_Sequence
https://www.ijert.org/attendance-monitoring-system-using-face-recognition
https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b
https://link.springer.com/book/10.1007/b97865
https://www.kaspersky.com/resource-center/definitions/what-is-facial-recognition
Handbook of Face Recognition by Anil Kumar Jain, Stan Z. Li
Face Detection and Recognition: Theory and Practice by Madhura Datta, Pradipta Kumar Banerjee, Asit Kumar Datta
Python and Tkinter Programming by John E. Grayson
Tkinter GUI Application Development Cookbook: A Practical Solution to Your GUI Development Problems with Python and Tkinter