ATTENDANCE SYSTEM USING ONE-SHOT LEARNING

PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE OF

MASTER OF TECHNOLOGY IN
COMPUTER SCIENCE & ENGINEERING

Submitted By
ANAND SINGH
2K21/CSE/05
under the supervision of

Dr. ARUNA BHAT
(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

May-2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, **Anand Singh**, Roll No. **2K21/CSE/05** student of M.Tech (Computer Science and Engineering), hereby declare that the Project Dissertation titled **"Attendance system using one-shot learning"** which is being submitted by me to Delhi Technological University, Delhi, in partial fulfillment of requirements for the degree of Master of Technology in Computer Science and Engineering is a legitimate record of my work and is not copied from any source. The work contained in this report has not been submitted at any other University/Institution for the award of any degree.

Place: Delhi

Date: 29 May 2023

Anand Singh

(2K21/CSE/05)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CERTIFICATE

I, hereby certify that the Project titled **"Attendance system using One-shot learning",** submitted by Anand Singh, Roll No. 2K21/CSE/05, Department of Computer Science & Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of degree of M.Tech in Computer Science and Engineering is a genuine record of the project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree to this University or elsewhere.

Place: Delhi **Dr. Aruna Bhat**

20 May 2022 **AssociateProfessor**

ACKNOWLEDGEMENT

I am extremely grateful to my project guide, **Dr. Aruna Bhat,** Associate Professor, Department of Computer Science and Engineering, Delhi Technological University, Delhi for providing invaluable guidance and being a constant source of inspiration throughout my research. I will always be indebted to her for the extensive support and encouragement she provided.

I am highly indebted to the panel faculties during all the progress evaluations for their guidance, constant supervision and for motivating me to complete my work. They helped me throughout by giving new ideas, providing necessary information and pushing me forward to complete the work.

Anand Singh

(2K21/CSE/05)

CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ST	Spatiotemporal
ML	Machine Learning
DL	Deep Learning
CNN	Convolutional Neural Network
GAN	Generative Adversial Network

turnitin

Similarity Report ID: oid:27535:36408847

PAPER NAME

Theses_plag.pdf

AUTHOR

Aruna Bhat

WORD COUNT CHARACTER COUNT 10601 Words 62259 Characters

PAGE COUNT FILE SIZE 49 Pages 984.9KB

SUBMISSION DATE

May 29, 2023 11:03 AM GMT+5:30

REPORT DATE

May 29, 2023 11:04 AM GMT+5:30

8% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 4% Internet database3% Publications
- Crossref databaseCrossref Posted Content
- 6% Submitted Works database
- database
- database

Excluded from Similarity Report

- Bibliographic materialQuoted material
- Cited materialSmall Matches (Less then10
-
- words)

ABSTRACT

The conventional methods for attendance tracking often involve manual processes, such as paper-based sign-in sheets or biometric systems, which are time-consuming, error-prone, and cumbersome. With the advancements in computer vision and machine learning, the use of one-shot learning techniques in attendance systems has gained significant attention. This thesis presents a comprehensive study on the design, development, and evaluation of an attendance system based on one-shot learning, aiming to provide an efficient and reliable solution for attendance management.

In any organization, attendance is one of the most important things whether it can be an educational institute, or any corporate company. In most of the educational institutes, attendance of the students are marked manually, which is a very time consuming process if the number of students is huge. Maintaining attendance manually in the form of paper sheets is a very difficult task. There can be an error in filling records while entering data manually. These records can be easily manipulated. To solve this problem, we should have some computer aided solution to automate the attendance taking process which should be more reliable. Machine learning techniques are almost being used to solve most of the real world problems. Convolutional neural networks have become extensively employed in image processing tasks such as image classification, image segmentation, and object recognition and detection. These networks possess the capability to extracting characteristics or attributes from the input data. Here, we are using one shot learning, where only one sample of each class is fed to the model for training and later on this model is able to predict the unseen samples.

Attendance marking systems play a crucial role in institutions, colleges, and corporations of all sizes. Through remarkable advancements in image processing, we have developed a system that capitalizes on this technology to simplify the process of managing attendance. Compared to other biometric authentication methods, face recognition has gained widespread popularity due to its convenient, non-intrusive, and contactless approach. The primary goal of this system is to identify faces and instantly recognize them by comparing them with the data stored in the database, ultimately recording attendance. By doing so, the system aims to streamline the arduous manual attendance process and improve its overall efficiency. Additionally, this system effectively tackles authentication and proxy-related challenges, as biometrics, including facial features used in face recognition, are unique and cannot be replicated. The system has been designed using OpenCV,

dlib, Face Recognition libraries, and One-Shot Learning techniques for precise face detection and recognition. Notably, only one image per individual in the database is required, resulting in a space-optimized solution compared to traditional training-testing models.

Attendance assessment is considered essential within the classroom environment in many institutions. Every institute, college, and organization has an attendance marking system. The evaluation of a student's performance is significantly influenced by attending lectures. As a result, our educational system places such a high priority on attendance. It is a valuable metric for assessing a student's performance. Therefore, we suggest a real-time attendance system that detects students by their faces after first identifying them. This model utilizes image processing, identifying the person and facial feature detection. Face Recognition is becoming more preferred over other biometric identification methods due to its simple, non-intrusive, and contactless methodology. The main objective of the system is to detect faces in real-time, recognize them, compare them with the database information, and record attendance. This serves to enhance and simplify the cumbersome manual attendance process. By leveraging the uniqueness of biometrics, specifically facial traits used in Face Recognition, the system effectively addresses authentication and proxy-related concerns. Unlike conventional training-testing models, to achieve accurate face detection and recognition, the system was created by incorporating OpenCV, dlib, Face Recognition libraries, and employing One-Shot Learning techniques.

Thesis Components

This thesis is organized into five chapters, each addressing specific aspects of the attendance system utilizes the principle of one-shot learning. The following is an overview of the chapters:

Chapter 1: Introduction

This chapter provides an introduction to the research topic, presents the background, discusses the motivation behind the study, states the problem, outlines the objectives, and highlights the scope and limitations of the research.

1.1 Attendance Systems Overview

This section provides an overview of attendance systems and their importance in various domains. The literature review explores the traditional approaches to attendance tracking, such as manual sign-in sheets and biometric systems, highlighting their limitations in terms of accuracy, efficiency, scalability, and privacy concerns.

1.2 Background and motivation

1.3 Statement of problem

1.4 Objectives of problem

Chapter 2 : Methdology

2.1 Proposed system

2.2 Process description

2.3 System Architecture

2.4 Data Collection and Preprocessing

2.5 One-shot algorithm

2.6 FaceNet

2.7 Haar cascade

2.8 Adam optimizer

Chapter 1 : Introduction

1.1 Attendance System Overview

In today's world, advances in science and technology have made our lives easier by automating many mundane jobs, and approaches are used in many fields with extreme precision in providing outcomes. Traditional attendance systems that are still in use today necessitate a significant amount of manual labor. Automated attendance systems are a recent construct developed in the field of automation to replace manually noted attendance. Automated Attendance Systems are commonly employed in a variety of organizations and locations. The majority of these authentication solutions are designed with biometric information, smart tags, and keys. The labor-intensive nature of manual attendance systems makes them inefficient in situations where there are many people present. The repetitive tasks can be automated using an automated attendance system. The development of an automated attendance system is a difficult procedure that includes:

- Data acquisition,
- Database setup,
- Biometric identification,
- and Data pre-processing.

The next sections go into additional details about the model, its components, an evaluation, and potential future enhancements. A little infant can naturally grasp the distinction between a flower and a plant by seeing a few examples, demonstrating how humans can learn from knowledge with naturally less information. However, in order for machines to recognize patterns and learn from training examples, they need a sufficient volume of data. Deep learning models, which are trained using TBs of data demonstrate this. Recently, algorithms that can learn from less examples than conventional systems have been developed. One method for learning patterns from data is called few shot learning, which requires only a small sample size. For issues relating with object recognition, computer vision employs this strategy.

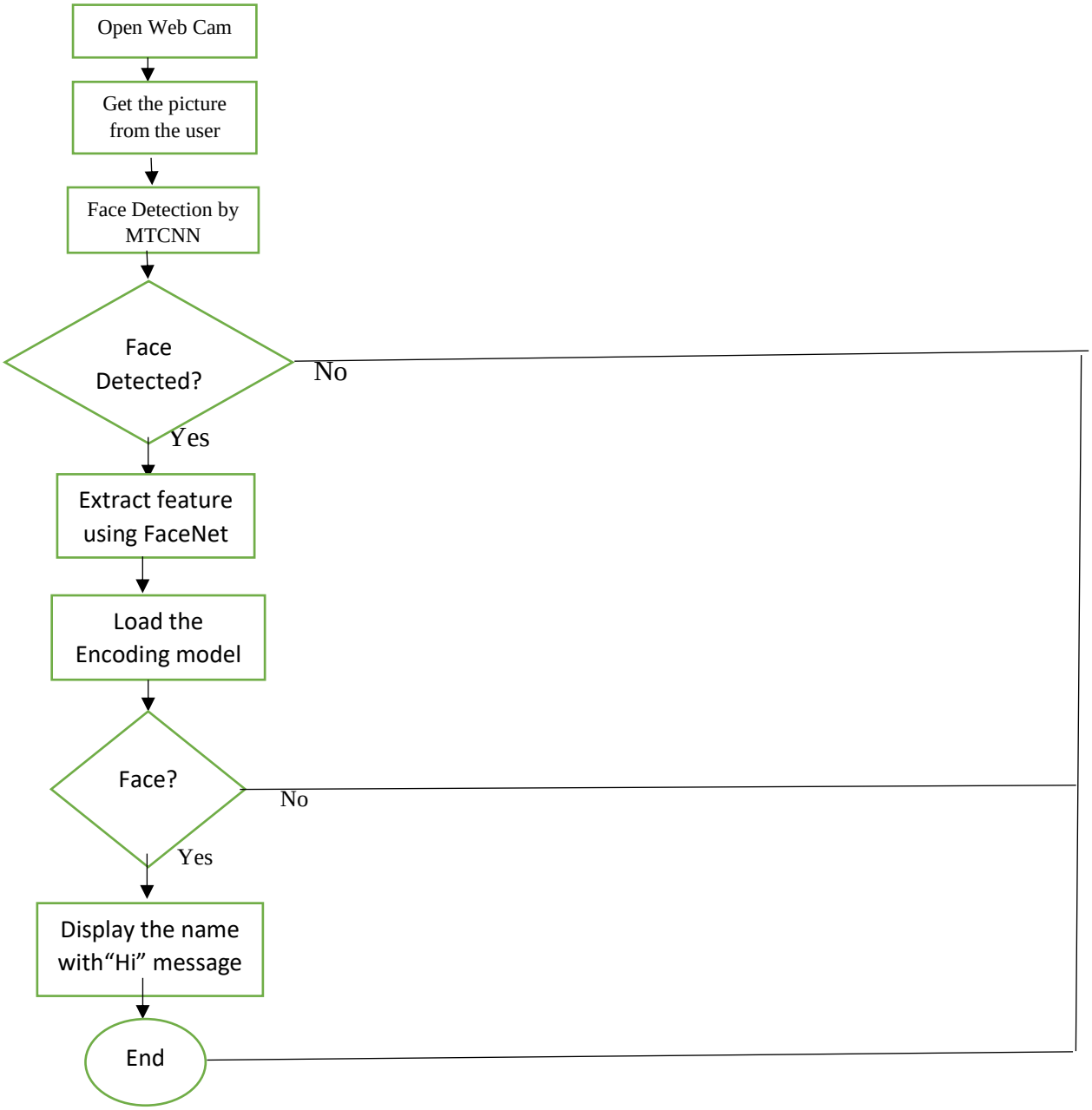


Fig1: Architecture of Attendance System

1.2 Background and Motivation

The efficient management of attendance has always been a critical aspect in various domains, including educational institutions, workplaces, and events. Conventional methods of tracking attendance, such as paper-based sign-in sheets that require manual entry or biometric systems, are frequently associated with laborious processes, significant time consumption, and susceptibility to errors. With the advent of computer vision and machine learning techniques, there has been a growing interest in leveraging these technologies to develop more efficient and reliable attendance systems. The management of attendance plays a crucial role in various domains, including educational institutions, workplaces, and events. Traditional methods of attendance tracking, such as manual sign-in sheets or biometric systems, often suffer from limitations such as inefficiency, errors, and lack of scalability. These conventional approaches are time-consuming, require significant manual effort, and may be susceptible to fraudulent practices. Moreover, they may not be able to handle variations in lighting conditions, pose, or appearance changes, resulting in reduced accuracy. To overcome these challenges, there is a need for an attendance system that can leverage the advancements in machine learning and computer vision techniques.

The motivation behind this research is to develop an attendance system that is efficient, reliable, and capable of overcoming the limitations of traditional methods. By harnessing the power of one-shot learning, which enables accurate recognition based on a single instance or a small number of samples, the proposed system aims to streamline the attendance management process, enhance accuracy, and reduce administrative burdens.

1.3 Problem Statement

The existing attendance systems suffer from several limitations, such as the need for physical contact (in the case of biometric systems), high costs, susceptibility to fraud, and scalability issues. Additionally, traditional face recognition approaches often struggle with accuracy when dealing with variations in lighting conditions, pose, and appearance changes. Therefore, there is a need for an attendance system that can overcome these challenges by leveraging the capabilities of one-shot learning techniques.

The existing attendance systems face several challenges that hinder their effectiveness. Manual sign-in sheets are prone to errors, fraudulent entries, and time-consuming data processing.

Biometric systems, although more accurate, often require physical contact and can be costly to implement and maintain. Moreover, they may face privacy concerns due to the collection and storage of biometric data. Another significant challenge is the limited accuracy of traditional face recognition approaches when dealing with variations in lighting conditions, pose, and appearance changes. These factors can result in false acceptances or rejections, impacting the overall reliability of attendance tracking.

To address these problems, the research focuses on the development of an attendance system using one-shot learning, which allows for accurate recognition using minimal training data. By leveraging this approach, the system aims to achieve higher accuracy, efficiency, and scalability while mitigating the limitations of traditional attendance systems.

Attendance recording is an important part of many organizations. For example in an educational institute, attendance and student performance are positively correlated and this an essential metric for assessment of students. The traditional attendance process is tiresome and needs to be done manually which is cumbersome in a large crowd of students and wastes crucial class time. A few negatives of taking attendance manually are:

1. Human Errors

Humans can make errors in filling records while entering data manually. This is evident when in a large batch size and these errors also need to be manually corrected and it becomes difficult to verify the details.

2. Manipulation of Data

Manual attendance system suffers from buddy punching and time-theft. Manually recorded data is prone to manipulation by people and its authenticity can’t be verified. Inaccurate information may be entered into the manual record with no proof for verification which would lead to wasted man hours and decrease in productivity.

3. Time Exhaustion

Taking attendance and then also verifying it is a time consuming task especially when the sample size is large and useful class time is wasted. In case of errors, students will rush in to get it corrected and this would also waste time. Let suppose in a 8 hour day of 1 period per hour if 10 minutes goes

in for taking attendance then out of the 480 minutes of class time, 80 minutes are wasted on a regular basis.

4. Paperwork

Paperwork is not cheap and takes up more space than a digital copy. Lost paperwork is hard to retrieve and this can be a hassle for people who are handling multiple subordinates at the same time.

Based on the problem statement the following problem statements can be identified:

1. How to automate the attendance recording process?
2. How to minimize human interference in recording attendance?
3. How to do real time reporting of attendance?
4. How to make automated systems reliable?
5. What features of a person to use for recording attendance?
6. What algorithms can be employed in this specific domain?
7. What are the recent algorithms used for this domain?

1.4 Objectives

The objective of implementing an attendance system using one-shot learning is to develop a more accurate, efficient, and automated method for tracking and managing attendance in various domains, such as educational institutions, workplaces, or events. The specific objectives of the attendance system using one-shot learning include:

1. Improved Accuracy: The attendance system aims to achieve higher accuracy in identifying and verifying individuals based on their facial features. By leveraging one-shot learning techniques, which enable recognition with minimal training data, the system aims to accurately match individuals even with limited reference images.

2. Enhanced Efficiency: The attendance system seeks to automate the attendance tracking process, reducing manual effort and time required for traditional methods such as manual sign-in

sheets or barcode scanners. By leveraging one-shot learning algorithms, the system aims to quickly process and identify individuals, enabling efficient and real-time attendance management.

3. Scalability: The attendance system aims to handle a large number of individuals efficiently, making it suitable for various settings with a significant number of attendees. The system should be able to scale and adapt to accommodate an increasing number of individuals without sacrificing accuracy or efficiency.

4. Robustness to Variations: The attendance system aims to be robust to variations in lighting conditions, pose, expressions, and appearance of individuals. By employing preprocessing techniques, feature extraction methods, and one-shot learning algorithms that can handle variations, the system seeks to ensure reliable attendance tracking under diverse conditions.

5. User-Friendly Interface: The attendance system aims to provide a user-friendly interface for both administrators and end-users. The system should be intuitive and easy to use, allowing administrators to set up the system, manage the database, and extract attendance records. End-users should have a seamless experience in registering their attendance and receiving verification.

6. Privacy and Security: The attendance system aims to address privacy and security concerns related to biometric data. It should incorporate measures to protect individuals' privacy, ensure secure storage and transmission of attendance data, and comply with relevant privacy regulations.

Overall, the objective of the attendance system using one-shot learning is to provide a more accurate, efficient, and user-friendly solution for attendance management, offering benefits such as reduced administrative workload, improved record-keeping, and enhanced security.

The primary objective of this thesis is to design, create, and evaluate an attendance system using one-shot learning, with the aim of providing an efficient and reliable solution for attendance management. The specific objectives include:

1.4.1 Investigating existing attendance systems and their limitations: This objective involves conducting a comprehensive review of traditional attendance systems, their strengths, weaknesses, and the challenges they face.

1.4.2 Exploring the concepts and principles of one-shot learning: This objective focuses on understanding the fundamental principles of one-shot learning and exploring different algorithms and techniques used in this field.

1.4.3 Designing a system architecture for the attendance system using one-shot learning: This objective involves designing a robust and scalable system architecture that incorporates the principles of one-shot learning for attendance tracking.

1.4.4 Collecting and preprocessing a suitable dataset for training and evaluation: This objective involves acquiring a diverse dataset comprising facial images for training and evaluating the attendance system. Additionally, appropriate preprocessing techniques will be applied to enhance the quality of the dataset.

1.4.5 Implementing one-shot learning algorithms for attendance classification and verification: This objective entails implementing state-of-the-art one-shot learning algorithms to enable accurate attendance classification and verification based on a single or a few instances.

1.4.6 Evaluating the system's accuracy performance, efficiency, and scalability: This objective involves conducting rigorous experiments and evaluations to assess the performance of the developed attendance system. Accuracy metrics, computational efficiency, and scalability will be considered to determine the system's effectiveness.

1.4.7 Identifying and addressing potential challenges and limitations: This objective aims to identify any challenges or limitations encountered during the development and implementation of the attendance system using one-shot learning. Strategies and techniques will be explored to mitigate these challenges and enhance system performance.

1.4.8 Providing recommendations for future enhancements and applications: This objective involves analyzing the research findings.

Chapter 2: Methodology

2.1 The proposed system

one shot learning algorithms that work with extremely few instances were used to apply the suggested model. The proposed model is composed of two stages:

1. Preparation State

a. Image Capture : The Camera is mounted on doors which are the entry point of the class. Pictures of the children are taken with a camera. It is recommended that the children' captured images have a resolution of 640x480. If it is not at that(mention 640x480) size then resizing of the image is necessary but resizing may give poor performance. So it is necessary to capture 640X480 at time of image capturing so that resizing must be avoided.

b. Face Detection : OpenFace is an open-source software with Google's FaceNet or Facebook's DeepFace. It has a free license and allows commercial purposes. It is similar to DeepFace. It is implemented using Python. It is implemented in 9 keras library, so it can be run on CPUs or GPUs. It starts detecting the face using dlib and OpenCV — a AI-based machine learning library.

c. Pre-Processing: Pictures of the childrens are taken with a camera. From before a face that has been discovered is retrieved and subjected to pre-processing. There are various steps in this cleaning stage:

i. Enlarge the face region that was obtained to 100x100.

ii. A method of histogram normalization is capable of approximating. By extending the range of a picture's intensity, it serves to increase contrast and make a picture more lucid.

d. Database Development : The dataset development stage in our multifactor authentication system entails various steps, including: i. Every person's image should be taken. ii. procedures are then used to improve the image. iii. save the file's holdings.

2. Forecast HAAR and dlib algorithms are used for the task of face recognition. It matches with the images stored in the database with the image captured from the camera or image fed as a file. After this, if similarity is found between the captured image and any image from the database then the identity is verified else identity verification fail.

To address this problem, we employ Machine Learning techniques which involve storing data for the recognition process, generating new data based on existing information, and determining whether an image is suitable for attendance marking. These challenges can be overcome by utilizing machine learning algorithms, inputting the data, and obtaining the desired outcome. The process can be outlined as follows:

1. Initially, the system analyzes an image to identify all the faces of individuals attending the class or session.

2. Subsequently, it evaluates each face to determine its positioning and lighting quality, distinguishing between images that are correctly captured and those with unfavorable conditions.

3. Next, the system extracts distinctive facial features such as eyes, face structure, and other relevant characteristics that aid in person recognition.

4. Lastly, the system compares the current image with previously stored data and records attendance directly in an Excel sheet.

2.2 Process description

We are developing a facial recognition system in Python using OpenFace and dlib. The process can be described in the following steps:

Step 1: Face Detection

To begin, we utilize face detection to locate faces in a photograph. By converting the image to black and white, we can analyze each pixel and its surrounding pixels to identify faces in the image.

Step 2: Posing and Projecting Faces

In this step, we address the challenge of faces appearing in different directions. To overcome this, we adjust the image to ensure that the eyes and lips are consistently positioned. This facilitates easier comparison of faces in subsequent steps.

Step 3: Face Encoding

To perform face recognition, we extract specific measurements or features from each face. These measurements include attributes like ear size, eye spacing, and nose length. Using a neural

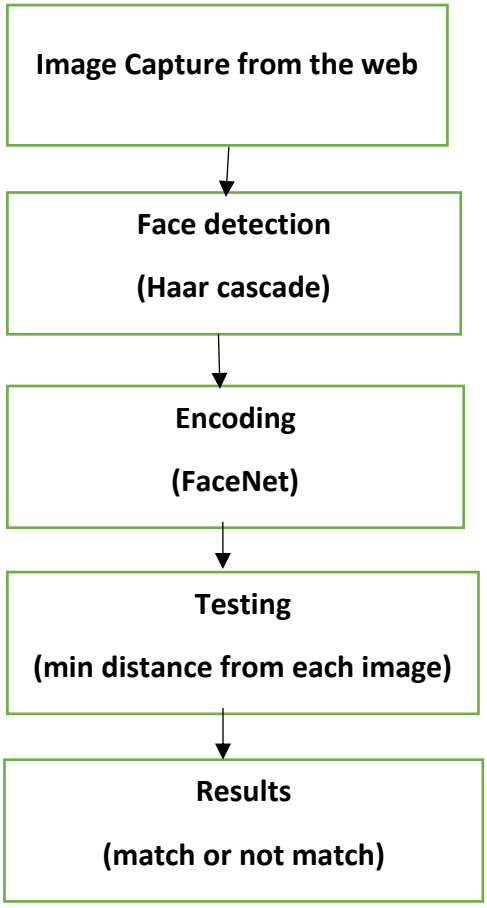
network, we train the system by comparing measurements between different known faces. The neural network learns to generate a set of 128 measurements for each image, ensuring that multiple images of the same person yield similar measurements.

Step 4: Identification and Matching

In the final step, we compare the measurements of a test image with the database of known faces. Using distance measurement, we calculate the similarity between the test image and each image in the database. The image with the lowest distance is considered the closest match. By training a classifier, we can determine which known person from the database closely matches the test image and display their name with a "Hi" message on the screen.

By following this process, we can effectively build a facial recognition system that can identify and match faces using OpenFace and dlib libraries in Python.

Process of Face Attendance system



2.3 System Architecture

This section outlines the proposed system architecture for the attendance system using one-shot learning. The architecture includes the key components and their interactions. It defines the flow of data and the sequence of operations involved in the attendance tracking process. The architecture incorporates the necessary modules for data collection, preprocessing, feature extraction, one-shot learning algorithms, attendance classification, and verification.

2.4 Data Collection and Preprocessing

This subsection discusses the process of data collection for training and evaluation purposes. It outlines the criteria for selecting a diverse dataset that captures variations in lighting conditions, pose, and appearance. The review covers different data acquisition techniques, such as image capture devices or publicly available datasets. Furthermore, preprocessing techniques, such as image resizing, normalization, and noise reduction, are discussed to enhance the quality and consistency of the dataset.

2.5 One-Shot Learning Algorithms

This section explores various one-shot learning algorithms suitable for the attendance system. It provides an in-depth analysis of different approaches, such as metric learning, siamese networks, and prototypical networks, highlighting their strengths and weaknesses. The review also discusses the training procedures and the process of generating embeddings or prototypes from the available training instances. The selected one-shot learning algorithm(s) for the attendance system are justified based on their effectiveness and compatibility with the system requirements.

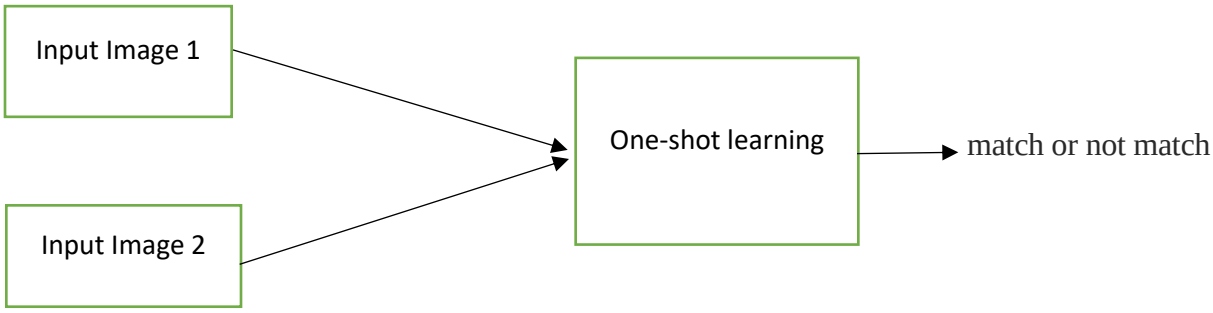


Fig3 : Algorithm

One-shot learning is a machine learning-based (ML) algorithm that compares the similarities and differences between two images. The goal is simple: verify or reject the data being scanned. In human terms, it's a computer vision approach for answering one question, is this person who they claim to be?

One-shot learning is a machine learning(ML) technique that rely on the ability to recognize and classify objects or patterns with limited training examples, typically only a single example per class. Traditional machine learning algorithms require a large amount of labeled training data to generalize well to unseen data. However, in many real-world scenarios, obtaining such a large dataset for each class may be impractical or unfeasible. Unlike traditional computer vision projects, where models are trained on thousands of images or videos or detect objects from one frame to the next, one-short learning algorithms are given limited data to compare images against. In many ways, one-shot algorithms are simpler than most computer vision models. However, other computer visions and ML models are given access to vast databases to improve accuracy and training outputs in comparison to one-shot or few-shot learning algorithms.

One-Shot vs. Few-Shot Learning

Few-shot learning is similar in almost every way. The only difference is that a computer vision model has a few more data points to compare an object against. Instead of one image, the database might have three or four. Few-shot learning works the same way as a one-shot. It still operates as a comparison-based approach, except the CV model making the comparison has access to slightly more data.

Examples of One-Shot and Few-Shot Learning in Computer Vision

One-shot and few-shot computer vision algorithms and models are mainly used in image classification, object detection and comparison, localization, and speech recognition. Real-world use cases involve face and signature recognition and verification. Such as airport passport scanners or law enforcement surveillance cameras, scanning for potential terrorists in crowded public places. Banks and other institutions use this approach to verify ID against the copy stored in their databases. As do hotels, to ensure keycards only open the door to the assigned room.

One-shot and few-shot learning is also used for signature verification. Plus, one-shot learning is integrated into drone and self-driving car technology, helping more complex algorithms detect

objects in the environment. It's also helpful when using neural networks during automated translations.

The objective of one-shot learning is to enable models to learn from minimal training examples, mimicking the human ability to recognize and generalize from limited experiences. This is particularly useful in domains where acquiring a large number of labeled samples is challenging or costly, such as rare diseases diagnosis, fine-grained image recognition, or personalized user preferences. One-shot learning approaches typically leverage techniques such as metric learning, transfer learning, or generative models to address the limited data problem. Some common methods used in one-shot learning include:

1. Siamese Networks: Siamese networks consist of twin neural networks that share the same architecture and weights. They take pairs of input samples and map them to their respective embeddings. Siamese networks are trained using a contrastive loss function that encourages similar examples to have embeddings that are close together and dissimilar examples to have embeddings that are far apart.

2. Triplet Networks: Triplet networks are designed to learn how to map input samples to embeddings. During training, triplet networks optimize the network by utilizing triplets of examples. Each triplet consists of an anchor example, which serves as a reference, a positive example that belongs to the same class as the anchor, and a negative example from a different class. The network is trained to minimize the distance between embeddings of the anchor and positive examples while maximizing the distance between embeddings of the anchor and negative examples. This optimization process enables the network to learn effective embeddings that facilitate accurate similarity measurements between different examples.

3. Metric Learning: The goal of this is to learn a suitable distance metric or similarity measure in the embedding space. The goal is to ensure that examples from the same class have small distances or high similarities, while examples from different classes have large distances or low similarities. Different distance metrics, such as euclidean distance or cosine similarity, can be employed to quantify the similarity between embeddings. These metrics provide a measure of the proximity or dissimilarity between embeddings, aiding in the comparison and evaluation of their similarity.

4. Transfer Learning: Transfer learning involves leveraging pre-trained models or feature extractors to benefit from knowledge learned on a large-scale dataset. By pre-training a deep neural network on a vast dataset, such as ImageNet, the network learns generic features that can be fine-tuned or used as feature extractors for the one-shot learning task. Transfer learning helps in improving the generalization ability of the model even with limited training examples.

5. Generative Models: Generative models, including generative adversarial networks (GANs) or variational autoencoders (VAEs), can be used in one-shot learning to generate new synthetic examples or augment the limited training data. By generating additional samples, the model can learn more robust and discriminative features.

Overall, one-shot learning is a valuable approach that enables models to learn from limited training examples, expanding the applicability of machine learning in scenarios where acquiring large labeled datasets is challenging or infeasible. It exhibits promising potential across diverse domains, including CV (computer vision), NLP, and personalized recommendation systems.

2.6 FaceNet

FaceNet is a deep learning model and framework developed by researchers at Google that focuses on face recognition and facial feature extraction. It utilizes a CNN architecture to generate high-dimensional embeddings that represent the unique characteristics of each individual's face. FaceNet is a deep learning model and framework developed by Google researchers for face recognition and facial feature extraction. It uses a deep convolutional neural network (CNN) architecture to generate high-dimensional embeddings that represent the unique characteristics of each individual's face.

The main concept underlying FaceNet is to train a mapping from the image space to a condensed Euclidean space, where the distances between embeddings directly correspond to the similarities between faces. This allows for efficient and accurate face recognition by comparing the distances or similarities between face embeddings.

Here is a detailed explanation of the FaceNet model and its components:

1. Convolutional Neural Network (CNN) Architecture: FaceNet utilizes a convolutional neural network (CNN) to extract features from input face images. This CNN consists of several layers of

convolutional and pooling operations, which facilitate the learning of hierarchical representations of the face images. The convolutional layers are responsible for capturing local patterns and features, while the pooling layers down sample the feature maps to retain essential information. By employing this architecture, FaceNet can effectively extract discriminative features from face images for various applications.

2. Triplet Loss Function: The training of the FaceNet model involves the use of a triplet loss function, which specifically evaluates the distances between embeddings of anchor, positive, and negative examples. The anchor example corresponds to an image of a specific individual, the positive example corresponds to another image of the same individual, and the negative example corresponds to an image of a different individual. The primary goal is to decreases the distance between the anchor and positive embeddings, while simultaneously increases the distance between the anchor and negative embeddings. By adopting this training approach, the model is encouraged to acquire discriminative features that facilitate accurate discrimination between distinct individuals.

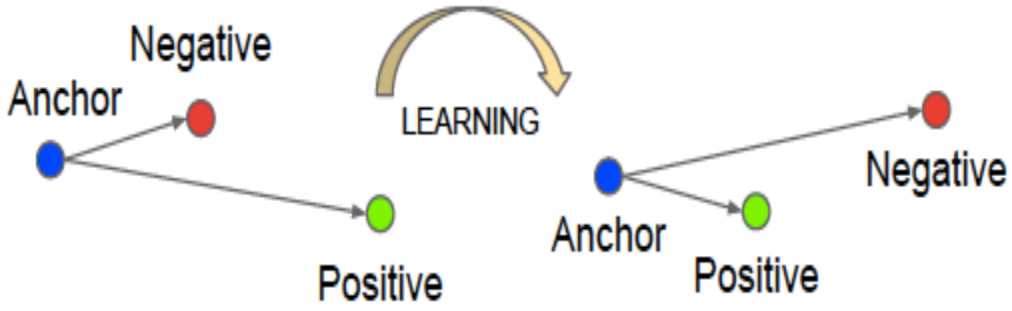


Fig2: Triplet Loss

The rationale behind using the triplet loss function is to ensure that the anchor image, which represents a specific person A, is closer in distance to positive images (all images of person A) than to negative images (all other images). In simpler terms, the goal is to minimize the distances between the embedding of the anchor image and the embeddings of the positive images, while maximizing the distances between the embedding of the anchor image and the embeddings of the negative images. This can be formally expressed and defined as the triplet loss function.

$$\sum_{i}^{n}[\left|\left|f(x_i^a) - f\left(x_i^p\right)\right|^2\right| - ||f(x_i^a) - f(x_i^n)||^2 + \propto]$$

Fig: Triplet Loss Formula

$X_i - It\ represents\ an\ image$

$f(X_{i)} - It\ represents\ the\ embedding\ of\ an\ image$

$\propto - It\ represents\ the\ margin\ between\ positive\ and\ negative\ pairs$

Fig 4 : Terms of triplet loss

In the triplet loss function, the superscripts "a," "p," and "n" represent the anchor, positive, and negative images, respectively. The parameter "alpha" denotes the margin between positive and negative pairs, which sets a threshold for the difference between image pairs. For example, if alpha is set to 0.5, we aim for the difference between the anchor-positive and anchor-negative image pairs to be at least 0.5.

The selection of appropriate image triplets is crucial for effective learning. It is important to choose triplets that violate the triplet constraint to ensure faster convergence and meaningful representation learning. Specifically, we aim to select "hard positives" and "hard negatives" that satisfy the following conditions:

- Equation (1) implies that given an anchor image of person A, we want to find a positive image of A with the largest distance between them.

- Equation (2) implies that given an anchor image of person A, we want to find a negative image with the smallest distance between them.

This approach helps accelerate convergence by focusing on informative image pairs. However, computing hard positives and hard negatives across the entire dataset can be computationally infeasible.

Triplet loss, introduced in the 2015 paper "FaceNet: A Unified Embedding for Face Recognition and Clustering," has become a popular loss function for supervised similarity or metric learning. It encourages dissimilar pairs to be distant from any similar pairs by at least a specified margin. Compared to contrastive loss, triplet loss is less greedy. Triplet loss ensures a margin between the distances of negative pairs and positive pairs, making different samples easily distinguishable from similar ones. Contrastive loss, on the other hand, considers the margin value only when comparing dissimilar pairs and does not prioritize the positioning of similar pairs. As a result, triplet loss can lead to better organization of the vector space while avoiding early convergence to local minima.

In the formulation of triplet loss, it operates on three objects simultaneously: the anchor, positive (a sample with the same label as the anchor), and negative (a sample with a different label from the anchor and positive).

```
def triplet_loss(y_true, y_pred, alpha=0.2):
    """
    Arguments:
    y_true -- true labels, required when you define a loss in Keras, you don't need it in this function.
    y_pred -- python list containing three objects:
            anchor -- the encodings for the anchor images, of shape (None, 128)
            positive -- the encodings for the positive images, of shape (None, 128)
            negative -- the encodings for the negative images, of shape (None, 128)

    Returns:
    loss -- real number, value of the loss
    """
    anchor, positive, negative = y_pred[0], y_pred[1], y_pred[2]
    pos_dist = tf.reduce_sum(np.square(tf.subtract(anchor, positive)), axis=-1)
    neg_dist = tf.reduce_sum(np.square(tf.subtract(anchor, negative)), axis=-1)
    basic_loss = tf.add(tf.subtract(pos_dist, neg_dist), alpha)
    loss = tf.reduce_sum(tf.maximum(0.0, basic_loss))
    return loss
```

Fig : Triplet Loss function used in project

3. Embedding Space: The output of the FaceNet model is a high-dimensional embedding vector that represents the unique characteristics of a face. The embedding space is designed to have desirable properties, such as having small intra-class distances (distance between embeddings of the same person) and large inter-class distances (distance between embeddings of different people). By learning embeddings in this space, it becomes easier to perform face recognition and similarity comparisons.

4. Training Data and Preprocessing: FaceNet requires a large dataset of labeled face images for training. The dataset should include images of various individuals under different conditions, such as lighting variations, pose changes, and expressions. Preprocessing techniques, such as face alignment and normalization, are applied to ensure consistent facial landmarks and image representations across different images. This helps in reducing the variations caused by factors other than facial appearance.

5. Face Verification and Identification: Once the FaceNet model is trained and embeddings are generated, face verification and identification tasks can be performed. Face verification involves comparing the similarity or distance between two face embeddings to determine whether they belong to the same person or not. Face identification, on the other hand, involves matching a given face embedding against a database of known embeddings to find the most similar identity.

FaceNet has achieved impressive results in face recognition tasks and has been widely adopted in various applications. Its ability to generate highly discriminative embeddings and its robustness to variations in facial appearance make it a powerful tool for face recognition systems. FaceNet offers a unified embedding solution for face recognition, verification, and clustering purposes. It projects each face image onto a Euclidean space where the distances between embeddings correspond to the similarity of faces. This means that an image of person A will have closer proximity to all other images of person A compared to images of any other individual within the dataset.

The key distinction of FaceNet from other techniques lies in its approach to learning the mapping from images and generating embeddings, rather than relying on a bottleneck layer for recognition or verification tasks. Once the embeddings are generated, various standard techniques can be applied to perform tasks like verification and recognition within the specific domain. These newly generated embeddings serve as the feature vectors for subsequent analyses. For instance, face recognition can be accomplished using k-nearest neighbors (k-NN) by leveraging the embeddings

as feature vectors. Similarly, clustering techniques can be employed to group similar faces together, while verification can be achieved by setting a threshold value to determine if a given pair of embeddings is considered a match.

In summary, FaceNet's innovation lies in its ability to learn mappings and create embeddings directly from images, enabling the use of standard techniques for subsequent tasks in face recognition, verification, and clustering.The main objective of FaceNet is to learn a mapping from the image space to a compact Euclidean space, where the distances between embeddings directly correspond to the similarities between faces. By mapping faces to a continuous embedding space, FaceNet enables efficient and accurate face recognition by comparing the distances or similarities between face embeddings. FaceNet has achieved significant advancements in face recognition tasks and has been widely adopted in various applications. Its ability to generate highly discriminative embeddings and its robustness to variations in facial appearance make it a powerful tool for face recognition systems.

To identify faces from an image, the initial step involves isolating the actual face from the background and distinguishing each face from others present in the image. Face detection algorithms need to handle challenges such as varying lighting conditions, inconsistent backgrounds, and facial positions like tilting or rotation. There are different techniques available for face identification, two of which involve utilizing the dlib library and the Haar cascade classifier from OpenCV.

The dlib library offers a face detection approach that combines the use of Histogram of Oriented Gradient (HOG) features and Support Vector Machine (SVM) classification. This method involves training the algorithm on positive images (containing faces) and negative images (without faces) to learn the distinguishing features of faces.

Alternatively, OpenCV provides the Haar cascade classifier, which is another technique for face detection. It involves training the classifier on positive and negative images, similar to the dlib approach. The Haar cascade classifier uses a set of pre-defined features known as Haar-like features to identify faces in an image.

Both approaches rely on machine learning algorithms to detect faces and require training on a dataset with labeled positive and negative examples. The choice between using dlib or the Haar cascade

classifier depends on the specific requirements of the application and the performance of the algorithms on the given dataset.

Feature used in face attendance system :

2.7 HAAR cascade

The Haar cascade algorithm is a popular method for object detection in computer vision. It is named after the Haar wavelets, which are mathematical functions used to analyze and represent signals. The algorithm was introduced by Viola and Jones in 2001 and has since become widely used for face detection and other object recognition tasks. Here's a detailed explanation of the Haar cascade algorithm:

1. Haar-like Features: The Haar cascade algorithm operates by detecting Haar-like features in an image. Haar-like features are rectangular regions with specific intensity patterns, such as edges or changes in texture. These features are defined by comparing the sum of pixel intensities in adjacent rectangular regions. The algorithm uses a set of predefined Haar-like features, including edge features, line features, and corner features.

2. Integral Image: The Haar-like features in the face detection algorithm are efficiently computed using an integral image. An integral image is a transformed representation of the original image, where each pixel stores the cumulative sum of the pixel intensities to its left and above. This integral image enables rapid computation of the sum of pixel intensities within any rectangular region using only four simple pixel operations. By utilizing the integral image, the algorithm can effectively and quickly evaluate the Haar-like features for face detection.

3. Training Stage: The Haar cascade algorithm requires a training stage to learn the discriminative properties of Haar-like features for the target object. During training, a large dataset of positive and negative examples is used. Positive examples contain images that contain the target object, while negative examples do not. The algorithm goes through a series of stages, with each stage consisting of a set of weak classifiers.

4. AdaBoost: The Haar cascade algorithm employs AdaBoost, a boosting algorithm, during the training stage. AdaBoost combines multiple weak classifiers, each of which focuses on a specific Haar-like feature, into a strong classifier. The algorithm iteratively selects the best weak classifier

at each stage based on its performance in distinguishing between positive and negative examples. The selected weak classifiers are combined to form the strong classifier.

5. Cascade Classifier: The Haar cascade algorithm uses a cascade classifier, which is a series of stages that progressively filter out non-object regions in an image. Each stage consists of multiple weak classifiers, and the stages are ordered from simple to complex features. The cascade classifier is designed to quickly reject non-object regions by using efficient filtering mechanisms, allowing for faster processing and reducing the number of false positives.

6. Sliding Window Approach: During the detection stage, the Haar cascade algorithm employs a sliding window approach. A fixed-size window moves across the image at various scales to search for the target object. At each window position and scale, the Haar-like features within the window are evaluated using the trained cascade classifier. If the features pass a certain threshold, the window is classified as containing the target object.

7. Post-processing: To polish the detection results, post-processing steps are often applied. These steps can include non-maximum suppression, which eliminates overlapping bounding boxes of detected objects, and additional verification steps to reduce false positives.

The Haar cascade algorithm has proven to be effective for object detection tasks, particularly in face detection. Its efficiency and accuracy make it suitable for real-time applications. However, it is worth noting that the algorithm may struggle with complex object variations, occlusions, and changes in lighting conditions, requiring further advancements and adaptations for challenging scenarios.

Steps involves in Training and Testing Phase:

1. Training Phase

Step1: Capture the image from the Web Cam

```
def get_image_from_cam():
    print('Starting Camera. Press "c" to capture')
    cap = cv.VideoCapture(0)
    if not cap.isOpened():
        print('Cannot Open Camera')
    while True:
        # Capture frame-by-frame
        ret, frame = cap.read()

        # if frame is read correctly ret is True
        if not ret:
            output("Can't receive frame. Existing")
            break
        # Display the frame
        cv.imshow('frame', frame)
        if cv.waitKey(1) == ord('c'):
            break
    # When everything done, release the capture
    cap.release()
    cv.destroyAllWindows()
    return frame[..., ::-1]
```

Step 2 : Identify face using HAAR cascade algorithm

```
model = cv.CascadeClassifier(str(Path(cv.data.haarcascades) / 'haarcascade_frontalface_default.xml'))
model = MTCNN()
return model
```

Step 3 : Convert color image to gray color scale

```
def get_faces(image, f=6):
    shape = image.shape
    face_points = detector.detectMultiScale(image)
```

Step 3 : Find co-ordinates of the faces

```
    faces = np.empty((len(face_points), 96, 96, 3), dtype='uint8')
    for i, pt in enumerate(face_points):
        x, y, w, h = pt
        ZX = int(w / f)
        ZY = int(h / f)
        xa = max(x - ZX, 0)
        xb = min(x + w + ZX, shape[1])
        ya = max(y - ZY, 0)
        yb = min(y + h + ZY, shape[0])
        face = image[ya: yb, xa: xb]
```

Step 4: Resize the image

```
        face = cv.resize(face, (96, 96), interpolation=cv.INTER_AREA)
```

Step 5 : Return the co-ordinates of the face

```
    for i, pt in enumerate(face_points):
        x, y, w, h = pt
        ZX = int(w / f)
        ZY = int(h / f)
        xa = max(x - ZX, 0)
        xb = min(x + w + ZX, shape[1])
        ya = max(y - ZY, 0)
        yb = min(y + h + ZY, shape[0])
        face = image[ya: yb, xa: xb]
        face = cv.resize(face, (96, 96), interpolation=cv.INTER_AREA)
        faces[i] = face
    return faces
```

Step 6 : Add to the Database

```python
def add_to_db(name, image):
    global register, db
    try:
        encodings = []
        image_fliped = cv.flip(image, 1)
        encoding, face = get_encoding(image, get_face=True)
        cv.imshow(name, face[:, :, ::-1])
        cv.waitKey(0)
        cv.destroyAllWindows()
        encodings.append(encoding)

        encoding = get_encoding(image_fliped)
        encodings.append(encoding)

        db[name] = encodings
        register.update({name: 0})
        output('Done!')
    except Exception as e:
        print(f'Invalid path! {e}')
```

Step 7: Feed to the inception model

```python
def inception_block_1a(X):
    """
    Implementation of an inception block
    """

    X_3x3 = Conv2D(96, (1, 1), data_format='channels_first', name ='inception_3a_3x3_conv1')(X)
    X_3x3 = BatchNormalization(axis=1, epsilon=0.00001, name = 'inception_3a_3x3_bn1')(X_3x3)
    X_3x3 = Activation('relu')(X_3x3)
    X_3x3 = ZeroPadding2D(padding=(1, 1), data_format='channels_first')(X_3x3)
    X_3x3 = Conv2D(128, (3, 3), data_format='channels_first', name='inception_3a_3x3_conv2')(X_3x3)
    X_3x3 = BatchNormalization(axis=1, epsilon=0.00001, name='inception_3a_3x3_bn2')(X_3x3)
    X_3x3 = Activation('relu')(X_3x3)
```

2. Testing Phase

All the steps from step 1-7 are same

Step 8 : Find distance (Final Phase)

```
        # Todo: Add ProcessPoolExecutor
        dists = np.empty(4)
        pairs = itertools.product(encodings, against)
        for i, (enc1, enc2) in enumerate(pairs):
            dists[i] = compute(enc1, enc2)
        print(name, dists)
        if np.sum(dists < threshold) >= 2:
            candidates[name] = np.sum(dists)

    if len(candidates) == 0:
        output('No such person exist!')
    else:
        candidate = min(candidates, key=candidates.get)
        output(f'Hi {candidate}')
        register[candidate] = 1
```

Find the min distance from each image which are stored in a database, among all the distances which image has minimum distance, show their name with “Hi” message on display.

2.8 Adam Classifier

The Adam optimizer is a widely used optimization algorithm in deep learning, frequently employed in tasks such as face recognition. It extends the concept of stochastic gradient descent (SGD) by combining the advantages of adaptive learning rates and momentum-based optimization techniques. The key feature of the Adam optimizer is its ability to dynamically adjust the learning rate for each parameter, taking into account estimates of the first and second moments of the gradients. By adapting the learning rate in this manner, the Adam optimizer enhances the efficiency of optimization and helps improve the performance of face recognition models.

In the context of face recognition, the Adam optimizer is often employed to train deep neural networks that are used for feature extraction or classification tasks. Here's an explanation of how the Adam optimizer works and its advantages:

1. Adaptive Learning Rate: The Adam optimizer dynamically modifies the learning rate for each parameter independently by considering the gradient history specific to that parameter. This adaptability enables the optimizer to customize the learning rate on a parameter-by-parameter basis, which can be advantageous when different parameters necessitate distinct learning rates.

2. Momentum: The Adam optimizer incorporates the concept of momentum, which helps accelerate convergence and avoid getting stuck in local minima. It keeps track of the past gradients and updates the parameters by adding a fraction of the previous update. This allows the optimizer to continue moving in the direction of previous updates, providing a smoother optimization trajectory.

3. First and Second Moment Estimations: The Adam optimizer calculates and utilizes the first and second moments of the gradients associated with each parameter.. The first moment estimation, known as the mean or average of the gradients, captures the trend of the gradients over time. The second moment estimation, known as the variance or average of the squared gradients, provides information about the scale of the gradients. These estimations are used to update the parameters and adapt the learning rate.

4. Bias Correction: The Adam optimizer incorporates bias correction to address the initial bias caused by the estimates of the first and second moments being initialized as zero. The bias correction ensures that the estimates are accurate, particularly during the early stages of training when the estimates are less reliable.

Advantages of using the Adam optimizer in face recognition tasks:

1. Fast Convergence: The adaptive learning rate and momentum-based updates of the Adam optimizer can help accelerate convergence during training. This is especially beneficial when training deep neural networks for face recognition, as it allows for faster model convergence and reduced training time.

2. Robustness to Different Learning Rates: The adaptive learning rate mechanism of Adam ensures that different parameters can have different learning rates, which can be advantageous in face recognition tasks where certain facial features or classes may require different rates of learning.

3. Handling Sparse Gradients: The Adam optimizer performs well even when dealing with sparse gradients, which can occur when training deep neural networks on large datasets. This is beneficial in face recognition tasks where the availability and quality of training data may vary across individuals.

Overall, the Adam optimizer is widely used in face recognition tasks due to its adaptive learning rate, momentum-based updates, and robustness to different learning rates. It helps facilitate faster convergence and improved optimization performance, making it a popular choice for training deep neural networks in face recognition systems.

2.9 Inception model

Developed by researchers at Google, the Inception model, also known as GoogLeNet, is a convolutional neural network architecture that operates at a deep level. The Inception model is designed to achieve high accuracy in image classification tasks while minimizing the number of parameters and computational complexity compared to other deep architectures.

Here's a detailed explanation of the Inception model and its key components:

1. Inception Module: The Inception model introduced the concept of the Inception module, which is a key component that allows for efficient and parallel processing of feature maps at different spatial resolutions. The module consists of multiple parallel convolutional branches with different filter sizes, including 1x1, 3x3, and 5x5 convolutions. These parallel branches capture features at different scales and help the network learn diverse representations. Additionally, a pooling branch with 3x3 max pooling is included to capture spatial information.

2. Dimensionality Reduction: To reduce the computational complexity and the number of parameters, the Inception model incorporates 1x1 convolutions for dimensionality reduction. These 1x1 convolutions are applied before the larger convolutions in the Inception module. The

1x1 convolutions effectively reduce the number of input channels, allowing for efficient processing while preserving important spatial information.

3. Bottleneck Layers: In order to further reduce the computational cost, the Inception model utilizes bottleneck layers, which consist of 1x1 convolutions followed by 3x3 or 5x5 convolutions. The bottleneck layers help capture complex features using fewer parameters compared to directly applying larger convolutions.

4. Auxiliary Classifiers: The Inception model includes auxiliary classifiers in intermediate layers to combat the vanishing gradient problem and improve gradient flow during training. These auxiliary classifiers consist of 1x1 convolutions, followed by average pooling, fully connected layers, and softmax activations. The outputs of these auxiliary classifiers are combined with the main classifier's outputs during training to provide additional supervision and regularization.

5. Global Average Pooling: The Inception model find global average pooling as the final layer in fully connected neural network, compared to others traditional methods. Global average pooling reduces overfitting by summarizing the spatial information of the feature maps into a single vector, allowing for efficient inference and avoiding the need for a large number of fully connected parameters.

6. GoogLeNet and Inception Versions: The Inception model architecture has undergone several revisions, resulting in different versions of GoogLeNet. For example, Inception V1 (GoogLeNet) was the original model, while subsequent versions, such as Inception V2, V3, and V4, introduced further optimizations and improvements. These versions incorporated techniques like factorized convolutions, batch normalization, and residual connections to enhance performance and training efficiency.

The Inception model has been widely adopted and serves as a foundation for many state-of-the-art architectures in computer vision tasks. Its design principles of parallel processing, dimensionality reduction, and efficient use of convolutions have influenced subsequent advancements in deep learning architectures. The Inception model's ability to achieve high accuracy while maintaining computational efficiency makes it well-suited for various image classification and recognition tasks.

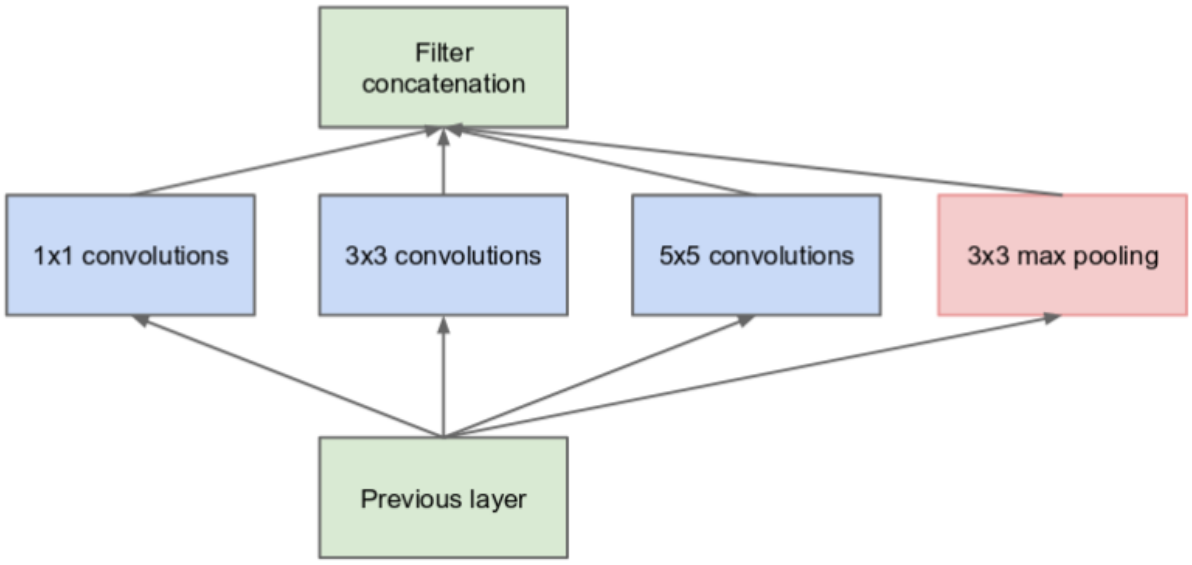


Fig : Naive inception model

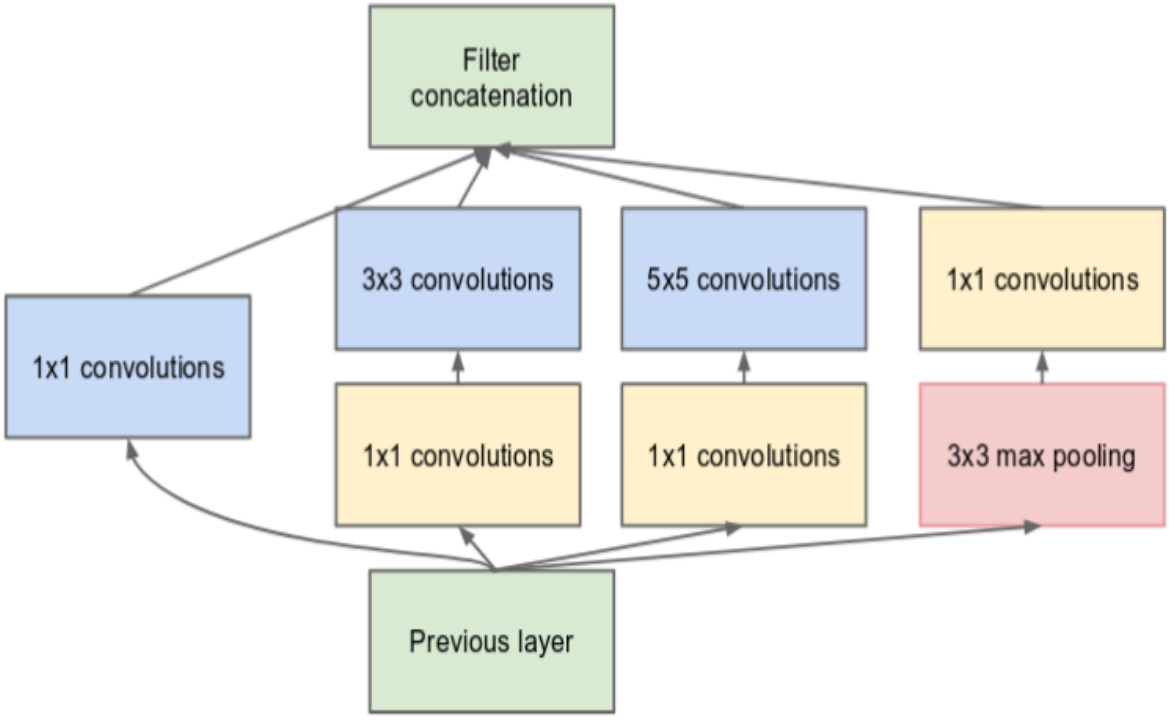


Fig : Advanced Inception model

Terms used in Inception model:

Stride, filter, pooling, flattening, and padding are key concepts and operations commonly used in convolutional neural networks (CNNs) for image processing tasks. Here's an explanation of each term:

1. Stride: In CNNs, stride refers to the step size at which the filter (also known as the kernel) moves across the input image during the convolution operation. When the stride value is set to 1, the filter moves pixel by pixel, whereas a stride value of 2 means the filter moves in increments of two pixels. Increasing the stride value reduces the spatial resolution of the output feature map, leading to a decrease in dimensionality and faster computational processing.

```
X_pool = MaxPooling2D(pool_size=3, strides=2, data_format='channels_first')(X)
```

2. Filter: A filter, also referred to as a kernel, is a small matrix of weights that is applied to the input image during the convolution operation. The primary purpose of the filter is to capture spatial features from the input image by performing element-wise multiplication and combining the results through summation. This process helps in highlighting important patterns and characteristics present in the image. Filters are typically square or rectangular in shape, and they are responsible for capturing different patterns and structures in the image.

```
X_3x3 = Conv2D(96, (1, 1), data_format='channels_first', name ='inception_3a_3x3_conv1')(X)
```

3. Pooling: Pooling is a technique utilized to reduce the spatial dimensions of feature maps obtained from convolutional operations. The most commonly used pooling method is max pooling, which involves partitioning the input image into non-overlapping regions and selecting the maximum value from each region, while disregarding the remaining values. This down sampling process aids in extracting the most prominent features from the input data.. Pooling helps reduce the spatial size, resulting in translation invariance and a reduction in the number of parameters. It also aids in extracting the most salient features.

```
X_pool = MaxPooling2D(pool_size=3, strides=2, data_format='channels_first')(X)
```

4. Flattening: Flattening is the process of converting a multidimensional tensor or matrix into a one-dimensional vector. In CNNs, following multiple convolutional and pooling layers, it is common to flatten the resulting feature maps before inputting them into fully connected layers. This flattening process transforms the multi-dimensional feature maps into a one-dimensional vector, allowing them to be processed by the fully connected layers. Flattening preserves the spatial relationships of the features while converting them into a format that can be processed by traditional dense layers.

5. Padding: Padding is the addition of extra pixels around the input image before applying convolution or pooling operations. It is used to maintain the spatial size of the input and prevent information loss at the borders. Padding helps to maintain the spatial resolution of the feature maps, especially when using smaller filters or when the stride is greater than 1. Common padding techniques include zero-padding, where extra pixels are added with zero values, and reflective padding, where the image is mirrored at the borders.

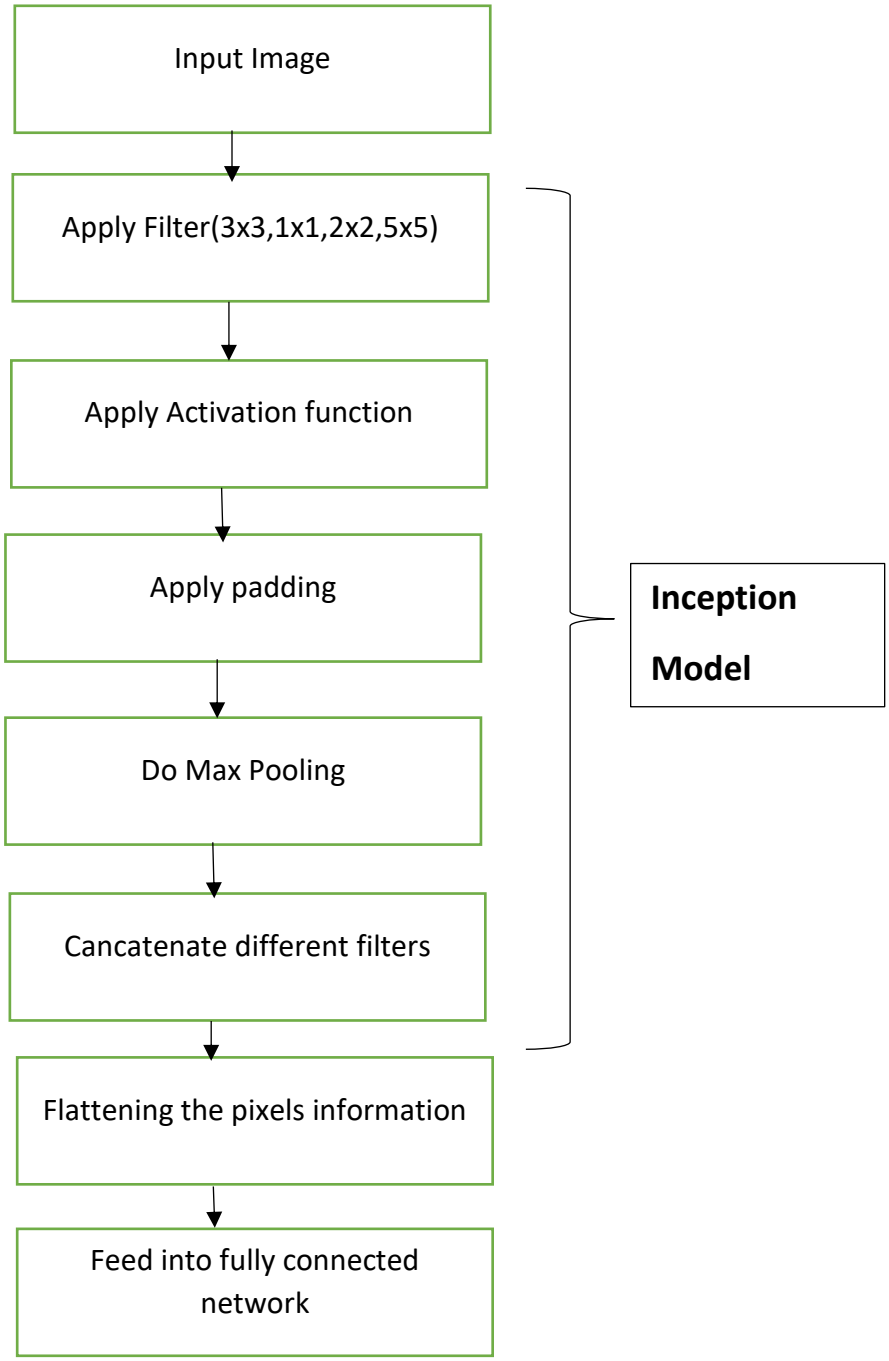
```
X_pool = ZeroPadding2D(padding=((3, 4), (3, 4)), data_format='channels_first')(X_pool)
```

These concepts and operations are fundamental in CNN architectures for image processing tasks. They allow the network to capture and extract meaningful features from input images, reduce dimensionality, handle different spatial resolutions, and facilitate efficient learning and representation of complex visual patterns.

```python
def inception_block_1a(X):
    """
    Implementation of an inception block
    """

    X_3x3 = Conv2D(96, (1, 1), data_format='channels_first', name ='inception_3a_3x3_conv1')(X)
    X_3x3 = BatchNormalization(axis=1, epsilon=0.00001, name = 'inception_3a_3x3_bn1')(X_3x3)
    X_3x3 = Activation('relu')(X_3x3)
    X_3x3 = ZeroPadding2D(padding=(1, 1), data_format='channels_first')(X_3x3)
    X_3x3 = Conv2D(128, (3, 3), data_format='channels_first', name='inception_3a_3x3_conv2')(X_3x3)
    X_3x3 = BatchNormalization(axis=1, epsilon=0.00001, name='inception_3a_3x3_bn2')(X_3x3)
    X_3x3 = Activation('relu')(X_3x3)

    X_5x5 = Conv2D(16, (1, 1), data_format='channels_first', name='inception_3a_5x5_conv1')(X)
    X_5x5 = BatchNormalization(axis=1, epsilon=0.00001, name='inception_3a_5x5_bn1')(X_5x5)
    X_5x5 = Activation('relu')(X_5x5)
    X_5x5 = ZeroPadding2D(padding=(2, 2), data_format='channels_first')(X_5x5)
    X_5x5 = Conv2D(32, (5, 5), data_format='channels_first', name='inception_3a_5x5_conv2')(X_5x5)
    X_5x5 = BatchNormalization(axis=1, epsilon=0.00001, name='inception_3a_5x5_bn2')(X_5x5)
    X_5x5 = Activation('relu')(X_5x5)

    X_pool = MaxPooling2D(pool_size=3, strides=2, data_format='channels_first')(X)
    X_pool = Conv2D(32, (1, 1), data_format='channels_first', name='inception_3a_pool_conv')(X_pool)
    X_pool = BatchNormalization(axis=1, epsilon=0.00001, name='inception_3a_pool_bn')(X_pool)
    X_pool = Activation('relu')(X_pool)
    X_pool = ZeroPadding2D(padding=((3, 4), (3, 4)), data_format='channels_first')(X_pool)

    X_1x1 = Conv2D(64, (1, 1), data_format='channels_first', name='inception_3a_1x1_conv')(X)
    X_1x1 = BatchNormalization(axis=1, epsilon=0.00001, name='inception_3a_1x1_bn')(X_1x1)
    X_1x1 = Activation('relu')(X_1x1)

    # CONCAT
    inception = concatenate([X_3x3, X_5x5, X_pool, X_1x1], axis=1)

    return inception
```

Fig : inception model used in this project

CNN Architecture



2.10 Activation Function

Non-linear activation functions are necessary in neural networks to introduce complexity and enable the learning of intricate patterns and decision boundaries. Linear activation functions result in a network that is essentially a linear combination of inputs, limiting its ability to solve complex problems.

ReLU (Rectified Linear Unit) is a popular non-linear activation function that addresses some of the limitations of earlier activation functions like sigmoid and tanh. Sigmoid and tanh suffer from the issue of vanishing gradients, where the gradients become extremely small over multiple iterations, leading to slow or halted learning. Additionally, computing functions like exponent and tan required by sigmoid and tanh can be computationally expensive.

ReLU overcomes these issues by outputting the input directly if it is positive, and zero otherwise. This avoids the vanishing gradient problem since the gradient is either zero or one. ReLU is computationally efficient as it involves simple thresholding operations. These advantages of ReLU have made it a preferred choice in many neural network architectures.

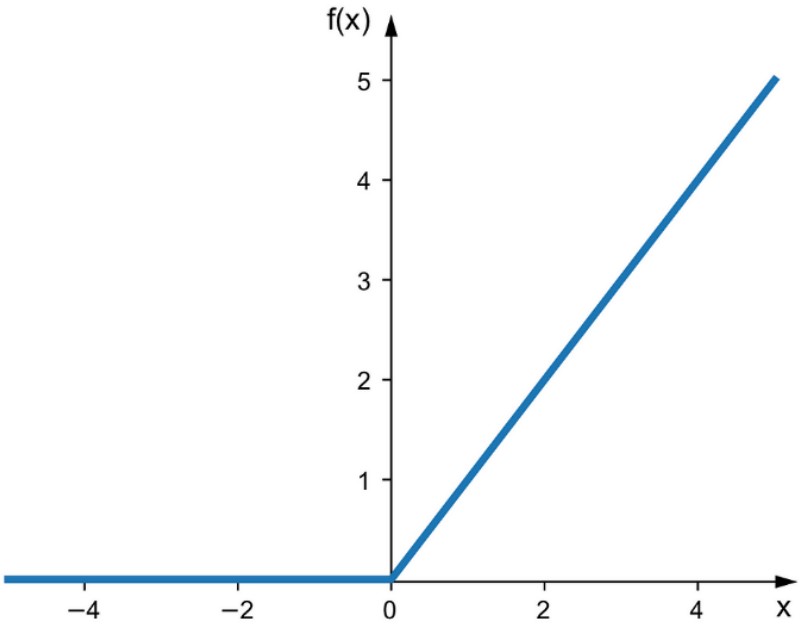


Fig 8: Activation Function

2.10 Python Library used in this Project:

Numpy, Keras, Pandas, Pickle, TensorFlow, OpenCV, and Python ML library are popular libraries and frameworks used in machine learning and data analysis tasks. Here's a brief overview of each library:

1. NumPy : NumPy is a crucial Python library that serves as a cornerstone for scientific computing. Its primary functionality revolves around facilitating the handling of extensive multidimensional arrays and matrices. Additionally, NumPy encompasses a comprehensive set of mathematical functions tailored to efficiently operate on these arrays. As a result, NumPy finds extensive usage in various domains, particularly in machine learning and data analysis, where it empowers numerical computations and facilitates data manipulation.

2. Keras: Keras is a Python-based high-level neural networks API that offers a convenient and user-friendly interface. It is developed on the foundation of TensorFlow, a popular deep learning framework. With Keras, designing, training, and evaluating deep learning models becomes more accessible and streamlined. By providing intuitive and modular building blocks, Keras simplifies the construction of neural networks. As a result, Keras has gained widespread adoption in various domains for deep learning applications.

3. Pandas: Pandas is a robust library that excels in data manipulation and analysis. It introduces specialized data structures like DataFrames, which facilitate the seamless handling and manipulation of structured data. With its extensive repertoire of functions and methods, Pandas empowers users to perform various operations, including data cleaning, filtering, transformation, and aggregation. As a result, Pandas has emerged as a favored tool for tasks related to data preprocessing and analysis.

4. Pickle: Pickle is a Python module utilized for serializing and deserializing objects. It provides a mechanism to convert intricate Python objects into a byte stream, which can be stored in a file or transmitted over a network. Pickle finds frequent application in saving and loading machine learning models, allowing users to store trained models for future utilization or sharing.

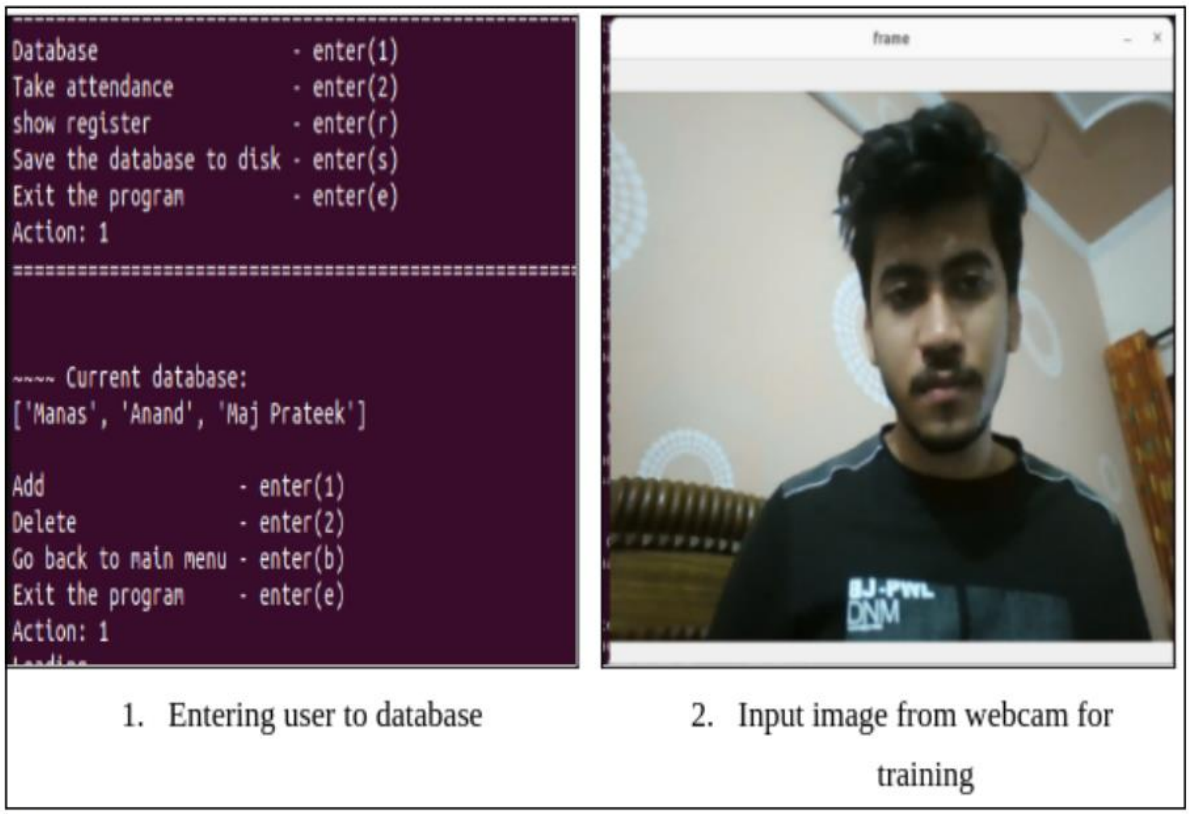
5. TensorFlow: Developed by Google, TensorFlow is an open-source machine learning framework that presents a wide range of tools, libraries, and assets for constructing and deploying machine learning models. TensorFlow provides a flexible and adaptable architecture for designing and training various machine learning algorithms, including deep neural networks. Moreover, it supports both CPU and GPU computations, offering users versatile options for executing their models efficiently.

6. OpenCV: OpenCV, which stands for Open Source Computer Vision Library, is a widely utilized computer vision library. It offers an extensive collection of algorithms and functions for various image and video processing tasks, such as filtering images, detecting features, recognizing objects, and calibrating cameras. OpenCV finds significant application in the field of computer vision, serving purposes like facial recognition, object tracking, and augmented reality.

7. Python ML Library (Scikit-learn): Scikit-learn is a popular Python library for machine learning that is widely utilized. It offers a comprehensive range of tools for performing various machine learning tasks, including classification, regression, clustering, and dimensionality reduction. With its consistent API and extensive implementation of algorithms, scikit-learn caters to both beginners and experienced practitioners in the field of machine learning.

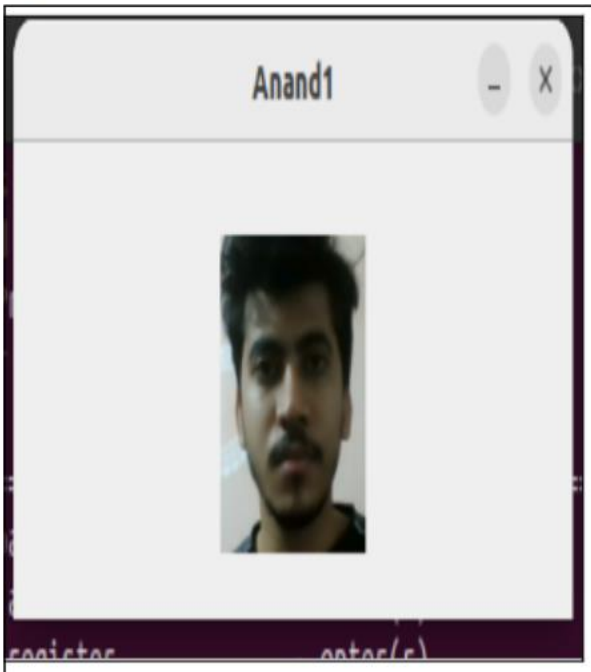
Chapter 3: Evaluation and Results

3.1 Results

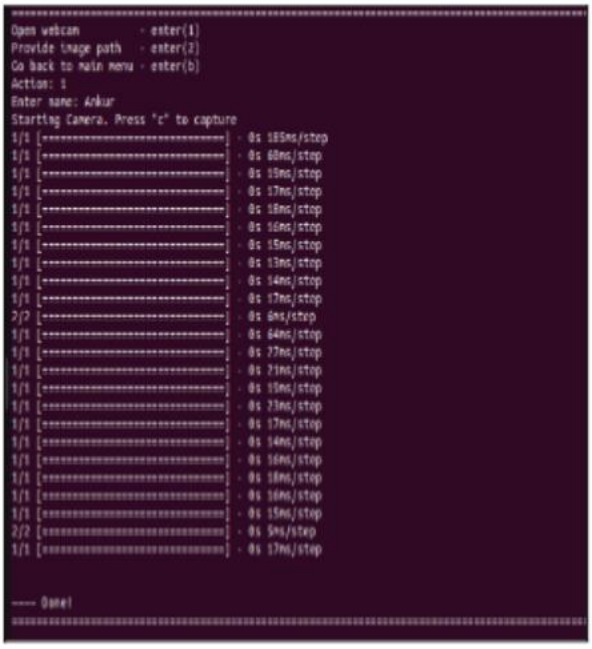


1. Entering user to database
2. Input image from webcam for training

[OBJ]

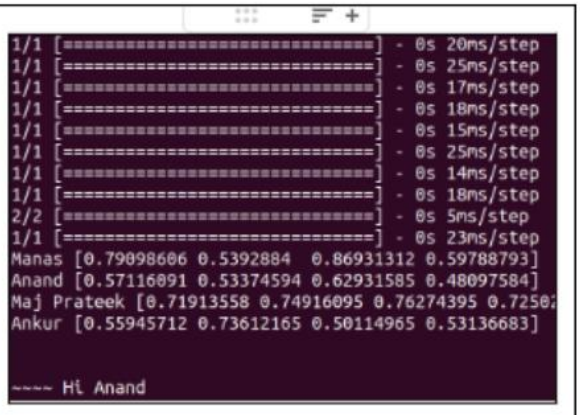


3. Face Detected from input



4. Training

5. Input Image for testing



6. Identity Verified

The working is as follows :

1. All identities can be stored in a database.

2. The face-detection system looks for faces in pictures.

3. The facial recognition model then processes the detected face and generates an embedding for that face.

4. This embedding's Euclidean distance from every other embedding that is stored in the database is determined.

5. If it is also less than a specific threshold, the shortest distance is chosen

3.2 System analysis

HARDWARE REQUIREMENTS:

1. High-resolution camera

2. 4 GB RAM

3. Intel i5 processor or higher

4. 2 GB graphics card

SOFTWARE REQUIREMENTS:

1. Windows 7 or higher

2. Text editor

3. Python 3.9.0

4. OpenCV

5. Firebase

6. Google Chrome

Non-functional Requirements:

1. Accuracy and Precision: Ensure accurate and precise attendance recording to avoid errors.

2. Modifiability: Make the system easy to modify.

3. Security: Ensure the system is secure and protects student privacy.

4. Usability: Design the system to be user-friendly and easy to understand.

5. Maintainability: Enable the maintenance team to fix any issues promptly.

6. Speed and Responsiveness: Ensure fast execution of operations.

USER REQUIREMENTS:

1. The facial recognition system should accurately scan and compare the scanned image with the stored image.

2. Upon finding a match, the system should mark attendance for the respective student.

3. Use a camera with an appropriate megapixel count to ensure proper attendance marking.

4. The user should store the faces of the subjects in the database for comparison.

5. The user should maintain a connection between their devices and the database.

3.3 Comparison with Traditional Attendance Systems

In this section, the performance of the attendance system using one-shot learning is compared with traditional attendance systems. The strengths and limitations of the proposed system are contrasted with those of manual sign-in sheets, barcode scanners, RFID-based systems, and biometric approaches. The comparison includes factors such as accuracy, efficiency, scalability, reliability, and privacy concerns. The experimental results and insights gained from the evaluation of the attendance system are used to support the comparison.

Chapter 4: Conclusion and Future Directions

4.1 Summary of Findings

In this section, a concise summary of the key findings and contributions of the thesis is provided. The achievements of the research in developing an attendance system using one-shot learning are highlighted, emphasizing the improvements over traditional attendance systems. The summary encompasses the system's performance in terms of accuracy, efficiency, and scalability, as well as its potential impact on attendance management in various domains.

4.2 Conclusions

This subsection presents the conclusions drawn from the research conducted in the thesis. It discusses how the attendance system utilizing one-shot learning addresses the limitations of traditional methods and achieves higher accuracy and efficiency. The conclusions highlight the significance of leveraging one-shot learning in attendance tracking and emphasize the potential benefits for educational institutions, workplaces, and events.

This subsection provides a final conclusion, summarizing the overall findings, contributions, limitations, and future directions outlined in the chapter. It reaffirms the significance of the research conducted in developing an attendance system using one-shot learning and its potential impact on attendance management practices. The conclusion also emphasizes the importance of ongoing research in this field and the need for continuous advancements in attendance tracking technologies. The research provided by the analysis of One-Shot learning methods for the facial recognition issue. The performance of features from various facial recognition models using common classifiers according to the investigation, while deep learning-based algorithms can learn efficiently from a small number of example faces rather than the thousands of photographs needed by conventional object identification pipelines, they still need precise parameter settings to function. Currently, the mechanism we have created works well for individual classroom attendance. It can be extensively employed at the collegiate level with the necessary improvements and establishment of a proper database containing all the information about each student in the college or university. Along with managing students, this system can also be used to manage the pupils of staff members and non-staff members. The fully automated system is another enhancement we wish to guarantee. To prevent any irregularities, such as tampering with the

devices, the system now needs supervision. Our aim is to fully automate the procedure using a CCTV camera's real-time live feed capture, which can record students' attendance without any manual supervision and generate accurate and unaltered attendance records.

4.3 Contributions

In this section, the specific contributions of the thesis are enumerated. The original contributions made to the field of attendance systems and one-shot learning are discussed. These contributions may include the development of a novel system architecture, the exploration and implementation of specific one-shot learning algorithms, the evaluation of performance metrics, or the insights gained from the comparison with traditional attendance systems. The unique aspects and innovations of the thesis are emphasized.

4.4 Limitations

This subsection acknowledges the limitations of the attendance system using one-shot learning. The potential challenges or constraints encountered during the implementation, evaluation, or real-world deployment of the system are discussed. These limitations may include the sensitivity of the system to variations in lighting conditions, pose, or appearance, or the potential difficulties in scaling the system to handle a large number of individuals. The implications of these limitations and their impact on the system's effectiveness are addressed.

4.5 Future Directions

In this section, potential avenues for future research and enhancements are presented. This may include addressing the identified limitations, further improving the accuracy and efficiency of the attendance system, or exploring additional applications of one-shot learning in attendance management. Possible research directions could involve investigating advanced one-shot learning algorithms, incorporating multimodal biometric data, or integrating the system with other emerging technologies such as edge computing or blockchain for enhanced security and privacy. The potential impact of these future directions on attendance systems and their broader implications are discussed.

Chapter 5: Recommendations and Implementation Guidelines

5.1 Recommendations

In this section, specific recommendations are provided based on the findings and conclusions of the thesis. These recommendations can serve as guidelines for organizations or institutions interested in implementing an attendance system using one-shot learning. The recommendations may include suggestions for system deployment, data acquisition, preprocessing techniques, choice of one-shot learning algorithms, feature extraction methods, and evaluation metrics. The recommendations aim to facilitate the successful implementation and utilization of the attendance system, considering the unique requirements and constraints of different domains.

5.2 Implementation Guidelines

This subsection offers practical implementation guidelines for deploying the attendance system using one-shot learning. It provides step-by-step instructions and considerations for each stage of implementation, including system setup, dataset acquisition, preprocessing techniques, training of the one-shot learning algorithm, feature extraction and representation methods, attendance classification, and verification. The guidelines encompass both technical aspects, such as hardware and software requirements, as well as operational considerations, such as privacy regulations, data management, and user interface design.

5.3 User Training and Adoption

In this section, recommendations are provided for user training and adoption of the attendance system. It highlights the importance of training system administrators, operators, and end-users on the proper usage and understanding of the system. The recommendations may include training programs, user manuals, and interactive demonstrations to ensure efficient and effective utilization of the attendance system. Considerations for user feedback, support mechanisms, and continuous improvement are also addressed to enhance user satisfaction and system acceptance.

5.4 Maintenance and Upgrades

This subsection focuses on the maintenance and upgrade requirements of the attendance system using one-shot learning. It provides recommendations for ensuring the system's long-term

functionality and reliability. This includes regular system maintenance activities, such as software updates, hardware upgrades, and database management. Additionally, suggestions for incorporating new advancements in one-shot learning algorithms, feature extraction methods, or hardware technologies are discussed to keep the system up-to-date and aligned with the latest developments in the field.

5.5 Security and Privacy Considerations

In this section, recommendations are given to address security and privacy concerns associated with the attendance system. It emphasizes the need for robust data protection mechanisms, user authentication protocols, and secure storage and transmission of attendance data. The recommendations may include encryption techniques, access control policies, and compliance with privacy regulations, like GDPR(General Data Protection Regulation or relevant) local laws. Considerations for data anonymization, consent management, and auditing procedures are also addressed to ensure the system's compliance with ethical and legal requirements.

REFERENCES

1. B. Alsulami, A. Srinivasan, H. Dong and S. Mancoridis, "Lightweight behavioral malware detection for windows platforms," 2017 12th International Conference on Malicious and Unwanted Software (MALWARE), 2017, pp. 75-81,

2. R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran and S. Venkatraman, "Robust Intelligent Malware Detection Using Deep Learning," in IEEE Access, vol. 7, pp. 46717-46738, 2019, doi: 10.1109/ACCESS.2019.2906934.

3. R. Bearden and D. C. Lo, "Automated microsoft office macro malware detection using machine learning," 2017 IEEE International Conference on Big Data (Big Data), 2017, pp. 4448-4452, doi: 10.1109/BigData.2017.8258483.

4. Priyanka Wagh ; Roshani Thakare ; Jagruti Chaudhari ; Shweta Patil,“Attendance system based on face recognition using eigen face and PCA algorithms”, published in 2015 International Conference on Green Computing and Internet of Things (ICGCIoT).

5. Omar Abdul Rhman Salim, Wasiu Adebayo Balogun, Rashidah Funke Olanrewaju, “Class Attendance Management System Using Face Recognition”, 2018 7th International Conference on Computer and Communication Engineering (ICCCE).

6. Wei-YuChen, Yu-Chiang FrankWang, Yen-Cheng Liu & Zsolt Kira, Jia Bin Huang,” A Closer Look At One-Shot Classification”, published as a conference paper at ICLR 2019.

7. Sujata G. Bhele and V. H. Mankar “A Review Paper on Face Recognition Techniques” published in International Journal of Advanced Research in Computer Engineering & Technology (IJARCET).

8. Mathana Gopala Krishnan,Balaji,ShyamBabu, “Implementation of Automated Attendance System using Face Recognition”, published in International Journal of Scientific & Engineering Research, Volume 6, Issue 3, March-2015.

9. S. Ravi and H. Larochelle, “Optimization as a model for One-Shot learning,” 2016.

10. D. Yi, Z. Lei, S. Liao, and S. Z. Li, “Learning face representation from scratch,” arXiv preprint arXiv:1411.7923, 2016

11. Wang, W., Sun, Q., Shen, J., Zhang, M., Yang, Y., & Wang, Y. (2019). One-shot face recognition with autoencoder-based Siamese networks. Pattern Recognition, 93, 101-111.

12. Li, Y., Sun, Z., Tang, J., Feng, Y., Zhang, W., & Yeung, D. Y. (2019). Sub-center-loss for face recognition with long-tail. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

13. Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., & Fergus, R. (2013). Regularization of neural networks using dropconnect. In Proceedings of the 30th International Conference on Machine Learning (ICML), 2013.

14. Zeng, Y., Zhang, Y., Li, S., He, F., & Liu, Z. (2018). Learning multigrained discriminative features for one-shot face recognition. IEEE Transactions on Information Forensics and Security, 13(5), 1179-1190.

15. Wen, Y., Zhang, K., Li, Z., & Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In European Conference on Computer Vision (ECCV), 2016.

16. Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep learning face attributes in the wild. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.

17. Chen, J., Chen, J., & Ji, Q. (2019). Facial recognition using generative adversarial networks. Pattern Recognition, 90, 128-140.

18. Lee, Y., Choi, J., & Yoon, S. (2018). Dual-path one-shot learning for person re-identification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

19. Yang, L., Zhang, H., & Gao, Y. (2018). Local saliency-aware deep feature embedding for one-shot face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

20. Zhang, X., Luo, H., Fan, X., Zhang, M., Sun, Q., & Wang, Y. (2018). Superpixels guided one-shot face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

21. Mi-Young Bae , Dae-Jea Cho Design and implementation of automatic attendance check system using BLE beacon International Journal of Multimedia and Ubiquitous Engineering , volume 10 , issue 10 , p. 177 - 186 Posted: 2015.

22. C Diwaker , P Tomar , R C Poonia , V Singh Prediction of Software Reliability using Bio Inspired Soft Computing Techniques Journal of medical systems , volume 42 , issue 5 Posted: 2018.

23. Aditya Jogiji , P Ghate WSN Based Automatic Attendance Monitoring System, International Journal of Computer & Mathematical Sciences (IJCMS) , volume 6 , issue 8 Posted: 2017

24. Aziza Ahmedi , Suvarna Nandyal, An Automatic Attendance System Using Image processing, The International Journal Of Engineering And Science , volume 4 , issue 11 , p. 1 - 8 Posted: 2015

25. R Krishnan , R Ramya , C Renuka , R Swetha , Ramakrishnan, Effective Automatic Attendance Marking System Using Face Recognition with RFID, IJSRST , volume 2920 , p. 158 - 162 Posted: 2016

26. K Shengli , Z Jun , S Guang , W Chunhong , Z Wenpei , L Tao, The Design and Implementation of the Attendance Management System based on Radio Frequency Identification Technology, International Conference on Electronic Science and Automation Control Posted: 2015.

27. S Konatham , B S Chalasani , N Kulkarni , T Taeib, Attendance generating system using RFID and GSM, Systems, Applications and Technology Conference (LISAT) , p. 1 - 3 Posted: 2016-04

28. N Kar , M K Debbarma , A Saha , D R Pal, Study of implementing automated attendance system using face recognition technique, International Journal of computer and communication engineering , volume 1 , issue 2 Posted: 2012

29. Naveed Balcoh , M Khan , Waqar Haroon Yousaf , M Iram Ahmad , Baig, Algorithm for efficient attendance management: Face recognition based approach, International Journal of Computer Science Issues (IJCSI) , volume 9 , issue 4 Posted: 2012

30. Chitresh Saraswat , Amit Kumar, An efficient automatic attendance system using fingerprint verification technique, International Journal on Computer Science and Engineering , volume 2 , issue 2 , p. 264 - 269 Posted: 2010

31. Unnati A Patel , S Priya, Development of a student attendance management system using rfid and face recognition: A review, International Journal of Advance Research in Computer Science and Management Studies , volume 2 , issue 8 , p. 109 - 119 Posted: 2014

32. Jomon Joseph , K P Zacharia, Automatic attendance management system using face recognition, International Journal of Science and research , volume 2 , issue 11 , p. 328 - 330 Posted: 2013

33. Ajinkya Patil , Mrudang Shukla, Implementation of classroom attendance system based on face recognition in class, International Journal of Advances in Engineering & Technology , volume 7 , issue 3 Posted: 2014

34. Y L Tian , R M Bolle, System and method for automatically detecting neutral expressionless faces in digital images, U.S. Patent , volume 6 Posted: 2005

35. Ying-Li Tian , Rudolf M Bolle, System and method for automatically detecting neutral expressionless faces in digital images, U.S. Patent , volume 6 Posted: 2005-04-12

36. Nirmalya Kar , Mrinal Kanti Debbarma , Ashim Saha , Dwijen Rudra Pal, Study of implementing automated attendance system using face recognition technique, International Journal of computer and communication engineering , volume 1 , issue 2 Posted: 2012

37. P Wagh , R Thakare , J Chaudhari , S Patil, Attendance system based on face recognition using eigen face and PCA algorithms,Green Computing and Internet of Things (ICGCIoT), 2015 International Conference on , p. 303 - 308 Posted: 2015-10,

38. Matthew Turk , Alex Pentland, Eigenfaces for recognition, Journal of cognitive neuroscience , volume 3 , issue 1 , p. 71 - 86 Posted: 1991

39. Ajinkya Patil , Mrudang Shukla, Implementation of classroom attendance system based on face recognition in class, International Journal of Advances in Engineering & Technology , volume 7 , issue 3 Posted: 2014

40. Jomon Joseph , K P Zacharia, Automatic attendance management system using face recognition, International Journal of Science and research , volume 2 , issue 11 , p. 328 - 330 Posted: 2013

41. D Panwar , P Tomar , V Singh, Hybridization of Cuckoo-ACO algorithm for test case prioritization, Journal of Statistics and Management Systems , volume 21 , issue 4 , p. 539 - 546 Posted: 2018

42. Borra Surekha , Kanchan Jayant Nazare , S Viswanadha Raju , Nilanjan Dey, In Intelligent techniques in signal processing for multimedia security , p. 293 - 319 Posted: 2017

43. Florian Schroff , Dmitry Kalenichenko , James Philbin, Facenet: A unified embedding for face recognition and clustering, Proceedings of the IEEE conference on computer vision and pattern recognition Posted: 2015

44.Narayana Darapaneni , Aruna Kumari Evoori , Vijaya Babu Vemuri , Thangaselvi Arichandrapandian , G Karthikeyan , Anwesh Reddy Paduri , Dhivakar Babu , Jayadev Madhavan, Automatic Face Detection and Recognition for Attendance Maintenance, p. 236 Posted: 2020

45. Vani Perumal, Face Recognition in Video Streams and its Application in Freedom Fighters Discovery - A Machine Learning Approach, p. 1 Posted: 2020

46. Muhammad Abuzar , Amran bin Ahmad , Adi Affandi bin Ahmad, A Survey on Student Attendance System Using Face Recognition, p. 1252 Posted: 2020

47. Nazmun Nessa Moon , Fernaz Narin Nur , Mohd. Saifuzzaman , Farah Sharmin , Protik Hosen , Syeda Farjana Shetu, An Efficient Development of Automated Attendance Management System.

48. Ahmed, E., Tomal, J., & Rahman, M. (2020). 1-The Impact of class attendance on student performance at TRU.

49. Ahmed, N., Gamage, D. T., Suwanabroma, J., Ueyama, T., Hada, S., & Sekikawa, E. (2008). The impact of quality assurance measures on student services at the Japanese and Thai private universities. Quality assurance in Education.

50. Al Sheikh, R., Al-Assami, R., Al-Bahar, M., Al Suhaibani, M., Alsmadi, M., Alshabanah, M., ... & Tayfour, M. (2019). Developing And Implementing A Barcode Based Student Attendance System. International Research Journal of Engineering and Technology (IRJET) Volume, 6.

51. Al_Janabi, S. (2020). Smart system to create an optimal higher education environment using IDA and IOTs. International Journal of Computers and Applications, 42(3), 244-259.

52. Bhattacharya, S., Nainala, G. S., Das, P., & Routray, A. (2018, July). Smart attendance monitoring system (SAMS): a face recognition based attendance system for classroom environment. In 2018 IEEE 18th International Conference on Advanced Learning Technologies (ICALT) (pp. 358-360). IEEE.

53. Charity, A., Okokpujie, K., & Etinosa, N. O. (2017, November). A bimodal biometrie student attendance system. In 2017 IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON) (pp. 464-471). IEEE.

54. Chen, J., & Lin, T. F. (2008). Class attendance and exam performance: A randomized experiment. The Journal of Economic Education, 39(3), 213-227.

55. Chen, W. J., & LI, H. F. (2020). Study on the Intelligent Campus System of the Internet of Things. DEStech Transactions on Social Science, Education and Human Science, (icesd).

56. Chou, P. T., & Kuo, Y. (2012). Examining factors relating to classroom attendance and performance. Journal of Studies in Education, 2(2), 193-204.

57. Dmello, R., Yerremreddy, S., Basu, S., Bhitle, T., Kokate, Y., & Gharpure, P. (2019, January). Automated Facial Recognition Attendance System Leveraging IoT Cameras. In 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 556-561). IEEE.

58. Durden, G. C., & Ellis, L. V. (1995). The effects of attendance on student learning in principles of economics. The American Economic Review, 85(2), 343-346.

59. Gendron, P., & Pieper, P. (2005). Does attendance matter? Evidence from an Ontario ITAL. Unpublished discussion paper, Humber Institute of Technology & Advanced Learning Toronto, Canada, 15.

60. Islam, M. M., Hasan, M. K., Billah, M. M., & Uddin, M. M. (2017, December). Development of smartphone based student attendance system. In 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC) (pp. 230-233). IEEE.

61. Jusoff, K., Samah, S. A. A., & Isa, P. M. (2009). Promoting University Community's Creative Citizenry. International Journal of Human and Social Sciences, 4(1), 25-30.

62. Khan, M. B., Prashanth, N. M., Nomula, N., Pathak, P., & Muralidhar, A. M. (2017). Auto Student Attendance System Using Student ID Card via Wi-Fi.

63. Kumar, M., & Salal, Y. K. (2019). Systematic Review of Predicting Student's Performance in Academics. Int. J. of Engineering and Advanced Technology, 8(3), 54-61.

64. Kumar, V. A., Kumar, V. A., Malathi, S., Vengatesan, K., & Ramakrishnan, M. (2018). Facial recognition system for suspect identification using a surveillance camera. Pattern Recognition and Image Analysis, 28(3), 410-420.

65. Lukas, S., Mitra, A. R., Desanti, R. I., & Krisnadi, D. (2016, October). Student attendance system in classroom using face recognition technique. In 2016 International Conference on Information and Communication Technology Convergence (ICTC) (pp. 1032-1035). IEEE.

66. Mon, S. S., Kham, N. S. M., & Aung, T. H. (2019, February). Application of RFID based Student Attendance System using Cloud Storage and IOT. Seventeenth International Conference on Computer Applications (ICCA 2019). O'Dwyer, A. (2011). Does a link exist between examination performance and lecture attendance for first year engineering students?.

67. Pathan, S. (2019). Student attendance system using facial recognition with AI. Rajeev, A., Rebbi, J., Correya, J. G., Prabhu, V. V., & James, D. (2019). HOG–Neural Network Based Student Attendance System.

68.Salman, H., Uddin, M. N., Acheampong, S., & Xu, H. (2019, March). Design and Implementation of IoT Based Class Attendance Monitoring System Using Computer Vision and Embedded Linux Platform. In Workshops of the International Conference on Advanced Information Networking and Applications (pp. 25-34).