Missing Person Finder Using Deep Learning

By

Zeeshan ali **01-132192-039**

Bismillah Asnam **01-132192-007**

Supervised by

Engr. Dr. Shahzad Hassan

Department of Computer Engineering
Bahria University Islamabad
2023

Missing Person Finder Using Deep Learning

By

Zeeshan ali **01-132192-039**

Bismillah Asnam **01-132192-007**

Supervised by

Engr. Dr. Shahzad Hassan

Submitted to the department of computer engineering in the partial fulfillment of the requirements for the degree of Bachelor's in Computer Engineering.

Department of Computer Engineering

Bahria University Islamabad

2023

UNDERTAKING

We, Zeeshan ali and Bismillah Asnam guarantee that the project the work on "*Missing Person Finder using Deep learning*" is a product of our own original efforts. No additional assessments have been conducted on the work. We have duly recognized and referenced any external sources of information used in our research.

_____

Signature of Student

Zeeshan ali

01-132192-039

_____

Signature of Student

Bismillah Asnam

01-132192-007

DEDICATION

We would like to express our heartfelt gratitude and dedication for this project to our dear parents and our supervisor, **Dr. Shahzad Hassan**. Their unwavering support and encouragement guided us throughout the entire project and dissertation. We also extend our deepest appreciation to our dissertation committee members, who generously offered their time and expertise to help enhance our work. We are grateful for their willingness to assist us and for their pleasant demeanor. Lastly, we dedicate this thesis to our best friends, whose unwavering support and joyous presence uplifted us during moments of exhaustion and weariness.

ACKNOWLEDGEMENTS

We, Zeeshan Ali and Bismillah Asnam, express our deepest gratitude to the Almighty Allah, whose blessings have enabled us to successfully complete our final year project. We extend our heartfelt appreciation to our supervisor for his invaluable support, encouragement, and extensive expertise. His guidance has been instrumental in shaping our thesis and navigating us through the entire project. We are truly fortunate to have such a remarkable mentor who not only guided us in our project but also encouraged us to explore greater opportunities for personal and professional growth. We would also like to extend our thanks to all the individuals who contributed to our project. Their assistance and collaboration played a significant role in its successful execution. Furthermore, we are immensely grateful to the committee members who diligently scrutinized our work, providing critical feedback that helped us enhance our performance and strive for excellence. Lastly, we would like to express our profound gratitude to our families, whose unwavering support and encouragement have been our pillars of strength throughout the journey. Their steadfast belief in us, particularly during challenging times, has been a constant source of motivation. We are incredibly fortunate to have them as our role models. We extend our sincere gratitude to everyone involved in our project and express our deepest appreciation for their contributions.

ABSTRACT

The increasing number of Missing persons in public spaces such as train stations, airplane terminals, and malls presents a significant challenge for law enforcement agencies and security personnel. In response to this challenge, we present our solution—an innovative system that utilizes cutting-edge machine learning and computer vision techniques, specifically deep learning. Our proposed system aims to address the task of locating missing individuals within vast collections of recorded or live video feeds. Our solution focuses on the development of a robust face detection system capable of accurately identifying facial features using a query image. By employing state-of-the-art facial recognition technology, our system aims to determine the precise time and location of the missing person within the video footage. To ensure the reliability and effectiveness of our solution, we will train the deep learning model on a comprehensive dataset comprising diverse facial images. This dataset will encompass variations in lighting conditions, facial expressions, poses, and demographics, thus enabling our system to handle real-world scenarios effectively.

The system will integrate cutting-edge deep learning algorithms to extract and analyze facial features, enabling the identification of missing persons with high accuracy and efficiency. The implementation of deep learning techniques empowers the system to automatically learn and adapt to different facial patterns and attributes, enhancing its overall performance. Our system will also incorporate real-time video processing capabilities, allowing it to operate on both pre-recorded video feeds and live surveillance streams. This feature will enable authorities to swiftly respond to missing person reports, facilitating rapid search and rescue efforts.

Keywords: Face Recognition, Face Encoding, Deep learning, Face Features, Missing persons

TABLE OF CONTENTS

LIST OF FIGURES

Number ***Page***

LIST OF TABLES

Number ***Page***

LIST OF EQUATIONS

Number ***Page***

CHAPTER 1

Introduction

Every day, countless individuals across the globe, including children, teenagers, and those with mental illnesses, go missing[1]. Tragically, many of these individuals fall victim to forced labor, human trafficking, or other illicit activities. They are often subjected to unimaginable circumstances, such as being sold as slaves, forced into child labor, or coerced into begging. Regrettably, the majority of them remain unfound, causing immense distress and worry for their loved ones. This project holds significant importance for authorities as it offers the potential to expedite and streamline the process of locating missing individuals[2], thereby reducing the time and resources required. The uncertainty surrounding the fate of the missing persons[2] can cause immense stress and anxiety for their friends, family, parents, and guardians who are desperate for their safe return .It is imperative to put an end to the heinous acts of kidnapping, human trafficking, and prostitution where individuals are trapped in a life of slavery and denied their freedom. The key to achieving this lies in the swift and secure recovery of missing persons. Our endeavor is to develop a system that utilizes cutting-edge technologies such as face recognition[3], artificial intelligence, deep learning, and machine learning. We are particularly focused on addressing the challenges associated with face recognition, including variations in head size, alignment, background, lighting conditions, and emotional expressions. Effectively distinguishing and representing individual faces amidst a sea of others poses a significant challenge. By harnessing the power of technology, we aspire to aid in case investigations and expedite the identification of victims. Our ultimate goal is to bring solace to the distressed families and contribute to the larger mission of ensuring a safer world where every missing person can be located quickly and safely.

1.1 Related Work:

S. Ayyappan and his colleagues from IFET College of Engineering have presented a paper that addresses a similar problem and objective. Their proposed system use DeepLearning-based facial feature extraction and matching, employing Stacked Convolutional Auto Encoder (SCAE). A database stores images of missing individuals, and a convolutional neural network is trained to learn facial features. The learned features are used to train a multi-class SVM classifier, successfully identifying and labeling missing children. However, their system's complex algorithms slow down the extraction and classification processes [2].In another study [3], the system takes input in the form of videos or images of human faces. The processing unit used is a Raspberry Pi Model 3B, equipped with a camera module. OpenCV, installed with the Python IDLE compiler, is used along with Haar CascadeClassifier for image classification. Facial data is collected, tagged with unique identifiers, and used to train the recognition and differentiation models. A drawback of this system is its lack of rotational invariance in image processing.A different approach discussed in [4] involves utilizing the OpenCV module for face recognition and identification. The PIL image engine is employed to read grayscale images, and the recognizer is trained using the faces detected in each frame. The Haar Cascade algorithm is used for face recognition, and the Face Recognizer object is constructed, incorporating functions for training and recognizing faces. The Local Binary Patterns Histogram (LBPH) algorithm is utilized to extract local features from photographs. The Adaboost method is employed to select features that improve classifier accuracy. However, one limitation of this system is that computation time increases when compared to three databases.

1.2 Problem Statement:

The task of locating a missing person necessitates sifting through extensive amounts of recorded video footage, often spanning hundreds or thousands of hours, captured by numerous cameras. This process demands significant manpower, consumes a considerable amount of time, and incurs substantial financial costs. Moreover, it poses challenges in terms of securing competent personnel and maintaining sustained focus and concentration throughout the search.

The problem statement revolves around the development of face detection system for locating missing person in large recorded or live video feeds, utilizing a query image.

Fig 1.1 Difficult for Human to Monitor Different Screens [1]

1.3 Proposed Solution:

Our proposed solution to the challenge of locating missing persons in large recorded or live video feeds is a robust face detection system. The system will utilize machine learning and computer vision techniques to accurately identify facial features using a query image. Through facial recognition technology, the system will determine the time and location of the missing person in the video footage. To ensure the system reliability, we will train it using a large dataset of diverse facial images.

1.4 Aims and Objectives:

Implementation of a face recognition system for locating missing individuals involves the task of determining if a person of interest has been observed in different areas, captured by various cameras, either at different times or even the same camera at different time intervals. The search can be facilitated by employing images or video clips as references. The primary aim of this project is to leverage technology to provide support for surveillance activities and introduce automation to security systems, enabling efficient and accurate identification processes.

- Locate an individual by leveraging multiple camera views
- Minimize human errors caused by fatigue
- Optimize operational efficiency.

1.5 Project Scope:

Making algorithms efficient enough to find missing person through Face Recognition techniques to overcome existing limitations and constraints.

- Find an individual by leveraging multiple camera views.
- Time and Location of missing person will be shown when found.
- The final deliverable is a web page.

1.6 Artificial Intelligence

Artificial intelligence (AI) represents the extraordinary endeavor of replicating the intricacies of human intelligence. It encompasses the development of intelligent tools that possess the ability to mimic human cognition and behavior. A defining characteristic of AI technology is its capacity to reason and make decisions with the intent of achieving specific objectives.

One notable component within the realm of AI is machine learning, a technology that enables systems to independently acquiring knowledge and dynamically adapting to novel data without human intervention. Deep learning methods empower this self-learning process by leveraging diverse and unstructured data types, such as text, images, and videos.

1.7 Machine Learning:

Machine learning as a Branch of AI, endows computer systems with the remarkable capability to autonomously acquire knowledge and information and enhance their performance through experiences, rather than relying solely on explicit programming. The focus of machine learning lies in developing algorithms and models that can access data and extract meaningful insights autonomously. This process of knowledge acquisition typically begins with the observation of data, which can include examples, direct experiences, or instructions, with the objective of discerning patterns or relationships within the data, machine learning strives to derive meaningful insights and, based on these learned patterns, make informed decisions.

The fundamental objective of machine learning is to facilitate systems gain knowledge and adapt their actions accordingly, without requiring direct human involvement or guidance. While traditional machine learning algorithms often treat text as a sequence of keywords, there is a shift towards employing techniques that emulate human comprehension of the underlying meaning of textual content. This advancement in machine learning techniques allows machines to rapidly acquire information and improve their performance based on the knowledge gained from experiences.

1.8 Deep Learning:

Deep learning an intricate component of Artificial Intelligence (AI) is characterized by its capacity to emulate the complex mechanisms of the human brain. Enabling the analyzing of raw data and the generation of intricate sequences for decision-making purposes.

Within the expansive domain of AI, deep learning represents a branch of ML which entails systems endowed with the capability to extract meaningful insights from unstructured or unrecognized data. Often referred to as deep neural learning or deep neural networks, this powerful technique allows for the exploration of intricate patterns and relationships within the data, facilitating the development of advanced decision-making models.

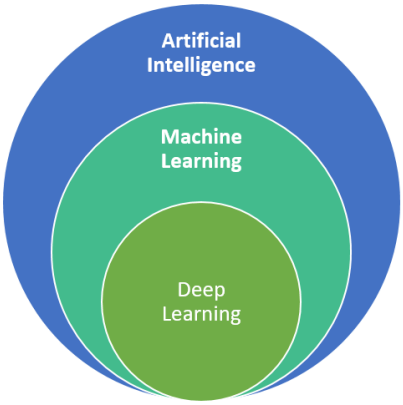

Fig 1. 2 Artificial Intelligence Venn Diagram [1]

1.9 Tools and Techniques:

1.9.1 Python:

Python is a highly versatile and immensely valuable high-level programming language widely utilized for a myriad of purposes. Created by Guido van Rossum in 1991[5], Python continues to be developed and maintained by the Python Software Foundation. Its syntax is designed to prioritize code clarity, enabling developers to express their ideas concisely. Python is not only a scripting language but also a powerful tool for efficient and rapid development. It finds application in various domains, including

- Software development
- Website development
- Its syntax exhibits a user-friendly structure reminiscent of the English language.

- It syntax allows programmers to write programs with fewer lines of code compared to other programming languages.

- Python Language enabling code execution immediately after it is written. This feature facilitates rapid prototyping.

1.9.2 Pycharm:

PyCharm is a comprehensive Integrated Development Environment (IDE) that brings together a plethora of essential tools for Python developers, offering a rich and productive environment for web development, data science, and general Python programming. With PyCharm, users have access to a wide range of top-notch features, including:

- Code inspection and intelligent code completion, aiding in writing error-free and efficient code.

- Advanced debugging support, particularly for web programming and popular frameworks such as Django and Flask.

- Customizability and cross-platform compatibility, allowing developers to tailor the IDE to their preferences and work seamlessly across different operating systems.

- A bundled package that combines debugging and testing capabilities, streamlining the development process.

1.9.3 Latex:

LaTeX is a document preparation system used for typesetting scientific, technical, and mathematical documents. It was created by Leslie Lamport in the late 1970s [6]. Lamport developed LaTeX as an extension of the TeX typesetting system, originally created by Donald Knuth in . LaTeX provides a high-level markup language and a set of macros that simplify the process of producing professional-looking documents, particularly those with complex mathematical equations and structures. It has become the standard in many academic and scientific fields for the creation of research papers, reports, theses, and other scholarly documents.

1.9.4 Google Colab:

Google Colab, short for Google Colaboratory, is an online platform that provides a cloud-based development environment for writing and executing Python code. It allows users to create, edit, and run Python notebooks directly in their web browsers, with the computing resources provided by Google. Google Colab was created by Google and was launched to the public. It was developed to make it easier for researchers, data scientists, and developers to collaborate and work on machine learning and data analysis projects. Google Colab provides access to GPU and TPU (Tensor Processing Unit) resources, making it particularly suitable for training and running deep learning models. The platform gained popularity due to its ease of use, integration with other Google services, and the ability to share notebooks with others for collaborative work.

Google Colab has become a valuable tool in the data science community, offering free access to computing resources and facilitating the development and sharing of machine learning projects.

1.9.5 GitHub:

GitHub is a web-based platform for version control and collaborative software development. It provides a platform for developers to host, manage, and track changes to their code repositories. GitHub allows multiple developers to work on the same project simultaneously, providing features such as version control, branching, merging, issue tracking, and collaboration tools.GitHub was created by Tom Preston-Werner, Chris Wanstrath, and PJ Hyett. It was initially built to host and share code repositories using the Git version control system, which was created by Linus Torvalds. GitHub quickly gained popularity within the developer community and became a central hub for open-source projects, serving as a platform for collaboration and community-driven development.Over the years, GitHub has expanded its offerings to include additional features such as project management tools, continuous integration and deployment, code review, and social coding features.

CHAPTER 2

Literature Review

This section delves into the comprehensive analysis of prior research conducted in the field, encompassing both traditional and deep learning methodologies employed for the purpose of Face Recognition of individuals, both in image-based and video-based contexts.

2.1 Conventional Methods

Conventional methods for face recognition were popular before the emergence of deep learning algorithms. Two main approaches were widely used

2.1.1 Holistic Approach

The Holistic Approach to face recognition gained popularity in the early 1990s [7] with the introduction of the Eigenface method. Holistic methods aim to derive a low-dimensional representation of faces based on specific assumptions regarding the distribution of data are often made in various algorithms and models, such as linear subspace, manifold, or sparse representation. These assumptions serve as foundational principles that guide the design and implementation of methods aimed at capturing the underlying structure and characteristics of the data. These methods treat the entire face as a single entity and try to capture the overall facial structure. Nevertheless, a limitation of holistic methods lies in their limited capability to cope with uncontrolled facial variations that deviate from their predefined assumptions. Consequently, these methods may encounter challenges when attempting to accurately recognize faces under diverse and changing conditions.

2.1.2 Local Feature Based

During the early 2000s and 2010s [8], there was a notable rise in the adoption of local feature-based methods for face recognition. These methods emphasized the capture of localized facial characteristics or features, known to exhibit greater resilience to variations in pose, illumination, and expression. Among the prevalent techniques used for local feature extraction, Gabor filters and Local Binary Pattern (LBP) have garnered considerable attention. Gabor filters are specifically employed to extract texture information, whereas LBP effectively captures localized texture patterns within the facial image.

2.2 Deep Learning

Deep learning, a branch of machine learning, revolves around the implementation of computer algorithms that can learn and develop autonomously. It relies on artificial neural networks that aim to mimic human thinking and learning processes, in contrast to simpler rule-based approaches employed in traditional artificial intelligence. In the past, the complexity of neural networks was limited by computational capabilities. However, recent advancements in Big Data analysis have enabled the development of larger and more powerful neural networks, allowing computers to analyze, learn from, and respond to complex events at a pace surpassing human capabilities. Deep learning has exhibited significant successes in various domains, including image classification, language translation, and speech recognition. It possesses the ability to tackle pattern recognition problems without requiring human intervention.

2.2.1 Working of Deep learning

Deep learning architectures drawing inspiration from the intricate architecture of the human brain, consist of interconnected layers of nodes, much like the neurons in our minds. These networks become increasingly complex as the number of layers increases. Similar to the communication between neurons, signals are transmitted between nodes in an artificial neural network, carrying associated weights that determine their significance.

Nodes with higher weights have a greater influence on the nodes they are connected to in lower layers. In the final layer, the weighted inputs are combined to yield an output or result. Deep learning frameworks necessitate powerful hardware to handle the substantial volume of data and perform complex mathematical computations. The processing requirements of these frameworks can be demanding, and specialized hardware accelerators are often employed to achieve efficient execution. Even with these advancements, the process of training deep learning models can be time-consuming. It may take weeks of iterative optimization and adjustment to fine-tune the model and achieve optimal performance. The advancement of deep learning has opened doors to solving intricate problems in various field such as image recognition, processing natural language, and speech synthesis. However, the success of deep learning heavily relies on powerful computing resources, efficient algorithms, and substantial amounts of labeled data for training. As technology continues to evolve, the capabilities of deep learning frameworks are expected to further expand, enabling breakthroughs in artificial intelligence applications and fostering new insights into complex data pattern.

2.2.2 Face Recognition Using Deep Learning

Deep learning approaches for face recognition leverage convolutional neural networks (CNNs), which are designed to acquire hierarchical encodings of facial features. These networks are capable of automatically extracting high-level features from raw image data, enabling more robust and accurate face recognition. A notable approach deep learning methods for face recognition is FaceNet, introduced by Schroff (2015) [9]. FaceNet utilizes a deep CNN architecture to acquire a concise and differentiating embedding space for facial images. By optimizing the triplet loss function, FaceNet guarantee that faces belonging to the same identity are closer brought closer together in the embedding space while faces of different identities are farther apart, facilitating effective face matching and recognition. Another significant contribution in deep learning-based face recognition is the DeepFace model proposed by Taigman et al.[10]. DeepFace incorporates a deep neural network architecture and introduces a novel alignment technique to handle pose variations.

It significantly reduces the gap between machine and human performance in face verification tasks, bringing face recognition closer to human-level performance.

2.3 Face Recognition Using Images

Face recognition using images has been a prominent application of deep learning, demonstrating impressive advancements in accurately identifying individuals from facial data. Deep learning techniques have greatly enhanced the performance and reliability of face recognition systems by extracting discriminative features and leveraging powerful neural network architectures.One of the key contributions in this area is the DeepFace model introduced by Taigman et al. [10]. DeepFace utilizes a deep CNN architecture to acquire a concise and differentiating embedding of facial features. It incorporates a sophisticated alignment technique that handles variations in pose and achieves remarkable performance in face verification tasks. DeepFace significantly bridged the gap between machine and human performance in face recognition.Another notable approach is the VGGFace model proposed by Parkhi et al. [11]. VGGFace utilizes a deep CNN with an extensive number of layers to extract facial features. The model undergoes training on a comprehensive dataset and demonstrates exceptional accuracy in face recognition tasks, achieving state-of-the-art performance. FaceNet framework developed by Schroff et al. (2015) [9] has made significant contributions to face recognition using deep learning. FaceNet employs CNN architecture to acquire a concise and differentiating embedding space for facial images.By optimizing the triplet loss function, FaceNet guarantee that faces belonging to the same identity are closer together in the embedding space, enabling effective face matching and identification. In recent years, attention mechanisms have been integrated into deep learning models for face recognition. Attention-based models focus on relevant facial regions and disregard irrelevant information, enhancing the discriminative capability of the learned features. These models dynamically attend to distinctive facial regions, improving the accuracy of face recognition systems.

2.4 Face Recognition Using Videos

In addition to image-based face recognition, deep learning has additionally been successfully applied to the objective of face recognition using video data. Analyzing facial information over time provides valuable temporal dynamics that can enhance the accuracy and robustness of face recognition systems. Deep learning models have demonstrated impressive capabilities in leveraging video data for accurate identification of individuals.

One notable approach for video-based face recognition is the 3D Convolutional Neural Network (3D CNN). Instead of processing individual frames independently, 3D CNNs capture spatio-temporal features by considering the temporal dimension along with the spatial dimensions of the input video frames. These models effectively learn representations that capture the motion and appearance changes in facial data over time, improving the discriminatory capability of the learned features. A pioneering work in video-based face recognition is the FaceNet model introduced by Schroff et al [9]. FaceNet's deep architecture incorporates 3D convolutions to capture temporal dynamics in facial videos. By considering multiple frames as input, FaceNet learns a compact embedding space that effectively represents an individual's face over time, enabling accurate recognition. Another approach is the Temporal Segment Network (TSN) proposed by Wang et al. [12]. TSN leverages a combination of spatial and temporal information by sampling multiple segments from a video and feeding them into a 2D CNN. By aggregating predictions from different segments, TSN effectively captures temporal dynamics while maintaining spatial discriminability for face recognition. Recurrent neural networks (RNNs) have been employed for video-based face recognition tasks. RNNs, such as Long Short-Term Memory (LSTM) networks, are competent of modeling sequential dependencies in video data. They can capture long-range temporal dependencies and effectively encode the temporal evolution of facial features over the course of a video. One example of RNN-based video face recognition is the LSTM-CNN architecture proposed by Baccouche et al. [13]. This model combines CNNs for spatial feature extraction with LSTM layers to model the temporal dynamics in facial videos.

By integrating both spatial and temporal information, the LSTM-CNN architecture achieves improved performance in video-based face recognition. Deep learning-based video face recognition methods have benefited from the availability of large-scale video datasets, YouTube Faces Database (YFDB) and the UCF101 dataset. Such datasets enable training deep models on diverse and extensive video data, facilitating the development of robust face recognition systems.

Table 2.1: Literature Review Comparison Table

Year	AUTHOR NAME	ALOGORITHM & TECHNIQUES	FEATURES	LIMATATIONS
2018	S. Chandran, Pournami & Balakrishnan, Byju & Rajasekharan, Deepak & N Nishakumari, K & Devanand, P & M Sasi, P.	Support Vector Machine (SVM), Convolutional Neural Network (CNN)	Binary Classification Spatial Hierarchical Representations	Large Data Requirements Sensitivity to Hyper parameters
2018	T. Zhang, J. Li, W. Jia, J. Sun, and H. Yang	Elliptical Head Contour Detection Fast and Robust Head Tracking	High accuracy in face detection and face occlusion detection (98.64% and 98.56% respectively). Real-world data evaluation	The algorithm's accuracy might decrease if lighting conditions change drastically.
2018	Piyush Kakkar, Mr. Vibhor Sharma	Haar Feature-Based Cascade Classifier Viola-Jones Algorithm	The system is designed to operate in real-time	Lighting conditions, pose variations, and occlusions

2018	Yutian Lin et al	ID-discriminative Embedding (IDE) 2 Attribute Recognition Network (ARN)	The APR network significantly speeds up the retrieval process	The manual annotation of attribute labels for large-scale re-ID datasets might be time-consuming and require considerable human effort
2018	X. Sun, P. Wu, and S. Hoi	Improved Mask R-CNN approach	The proposed G-Mask builds upon the Mask R-CNN approach, enhancing its performance for face detection and segmentation tasks	Utilizing ResNet-101 and FCN may require significant computational
2018	C. Ding and D. Tao	Trunk-Branch Ensemble CNN (TBE-CNN)	Improved Triplet Loss Function	The proposed TBE-CNN model, which shares convolutional layers, may have higher computational
2019	Sarthak Babbar, Navroz Dewan, Kartik Shangle, Sudhanshu Kulshreshtra, Sanjeev Patel	Principal Component Analysis (PCA)	This reduces the computational complexity and removes redundant information	Computational Complexity Limited Processing of Similar Facial Expressions
2020	S. Ayyappan and S. Matilda	SCAE,CNN, Haar CascadeClassifier, Multi-class Support Vector Machine (SVM) classifier	Utilization of a Convolutional Neural Network (CNN) for learning facial features	The system using OpenCV and Haar CascadeClassifier lacks the ability to handle face images with rotation, potentially impacting accuracy

2020	B. Yma, A. Lw, A. Zl, and C. Fl	CNN models	CNN and other deep learning methods have significantly improved image recognition accuracy and real-time efficiency	Poor lighting, occlusion, or changes in facial appearance
2020	Rongrong Jin, Hao Li, Jing Pan, Wenxi Ma, and Jingyu Lin	MTCNN, Facenet	FaceNet directly learns the mapping from face images	Robustness to Extreme Variations
2021	G. Vareldzhan, K. Yurkov and K. Ushenin,	Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), Scale-Invariant Feature Transform (SIFT)	Convert to Grayscale Edge Detection Clustering	Recognition of Faces with Complex Structures
2022	Shefali patil, Pratiksha Gaikar, Divya Kare, sanjay Pawar	K-Nearest Neighbors (KNN) algorithm	136 * 3 data points for face recognition	Large number of data points are required
2022	Adnan Nadeem, Muhammad Ashraf, Kashif Rizwan, Nauman Qadeer, Ali AlZahrani, Amir Mehmood and Qammer H. Abbasi	Viola-Jones using Cascade AdaBoost	Classifiers into multiple stages, forming a cascade	Sensitivity to Rotation and Scale

2.5 Center Loss

Center Loss [14] aims to learn discriminative features by pulling the features of the same identity closer together. It introduces an additional center term in the loss function, which encourages the features from the same class to have small intra-class distances. The center loss function is calculated as:

$$\mathcal{L}_C = \frac{1}{2}\sum_{j=1}^{B} \left\| f_{t_j} - c_{y_j} \right\|_2^2 \tag{2.1}$$

2.6 Triplet Loss

Triplet Loss [15] is a commonly used loss function in face recognition tasks. It aims to learn discriminative embeddings by minimizing the distance between an anchor image and a positive image, while maximizing the distance between the anchor image and a negative image. The formula for triplet loss can be represented as follows:

$$(a, p, n) = \max(0, d(a, p) - d(a, n) + \alpha) \tag{2.2}$$

2.7 Circular Loss

Circular Loss [16] is a loss function commonly used in deep learning for handling angular or circular data. It is often applied in tasks such as pose estimation or face recognition, where the output values represent angles or directions. The circular loss aims to enforce angular proximity between samples while taking into account the circular nature of the data.

The formula for circular loss can be represented as follows:

$$(x, y) = 1 - \cos(x - y) \tag{2.3}$$

2.8 Softmax Loss

Softmax Loss [17], also known as cross-entropy loss, is widely used for multi-class classification tasks, including face recognition. It encourages the network to output high probabilities for the correct class label while minimizing the probabilities for other classes.The Softmax Loss, also known as cross-entropy loss, is calculated as follows:

$$\text{SoftmaxLoss} = -\sum_{i=1}^{C} y_i \cdot \log(\text{softmax}(z_i)) \tag{2.4}$$

2.9 Euclidean Distance

The Euclidean distance [18] is commonly used in face recognition algorithms as a proximity metric of similarity or dissimilarity between two face feature vectors. The formula for calculating the Euclidean distance between two vectors, represented as arrays of values, is as follows:

Given two vectors A and B of equal length N:

$$\text{Ed} = \text{sqrt}((A[1] - B[1])^2 + (A[2] - B[2])^2 + \ldots + (A[N] - B[N])^2) \tag{2.5}$$

2.10 Support Vector Machine

In the domain of machine learning, the support vector machine (SVM) is a widely used supervised learning model primarily utilized for classification and regression analysis tasks. It utilizes a decision boundary, which acts as a delineation between different classes of objects. SVM [19] finds extensive application in tasks involving signal recognition, particularly in appearance-based recognition and color-based classification.

While it is important to note that support vector machines are not inherently superior to other machine learning techniques, they operate at a sophisticated level of complexity and possess considerable theoretical and empirical value.

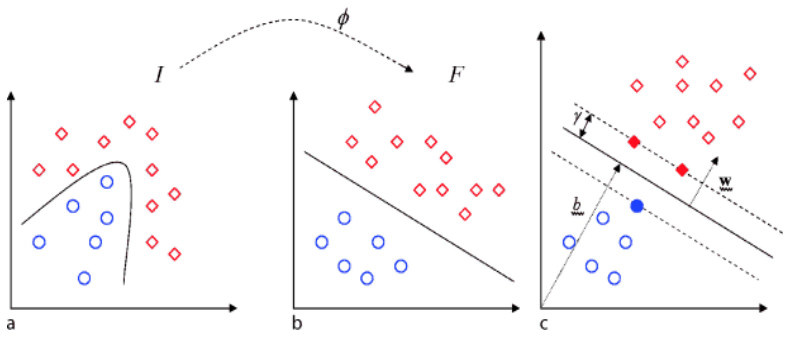

Fig 2.1 Support Vector Machine [19]

CHAPTER 3

Methodology

3.1 Dataset:

- Pin Face Recognition
- Custom Dataset

3.1.1 Pin Face Recognition Dataset

Pins Face Recognition is a facial recognition dataset specifically designed for studying unconstrained face recognition. It consists of a collection of face photographs obtained from Pinterest, a popular image sharing platform. The images in this dataset have been curated and cropped to focus on the faces.

Key details about the Pins Face Recognition dataset are as follows:

Celebrities: The dataset contains images of 105 different celebrities. These celebrities come from various fields such as entertainment, sports, politics, and more.

Number of Faces: There are a total of 17,534 faces in the dataset. Each face corresponds to a specific individual within the celebrity set.

Image Quality: The images in the dataset are of high quality and have been carefully curated to ensure that they are suitable for face recognition tasks.

Image Diversity: The dataset is well-balanced, with a good mix of male and female faces, as well as faces of different ages and ethnicities.

Dataset Updates: The dataset is constantly being updated with new images, ensuring that it remains relevant and up-to-date.

The Pins[20] Face Recognition dataset is a valuable resource for researchers and developers working on face recognition tasks. It provides a comprehensive and challenging dataset that can be used to develop and evaluate face recognition algorithms and systems.

Table 3.1: Pin Dataset Detail

Key Details	Values
Celebrities	105
Number of Faces	17,534
Image Quality	High
Classes	105
Image Diversity	Well-balanced
Dataset Updates	Constantly updated

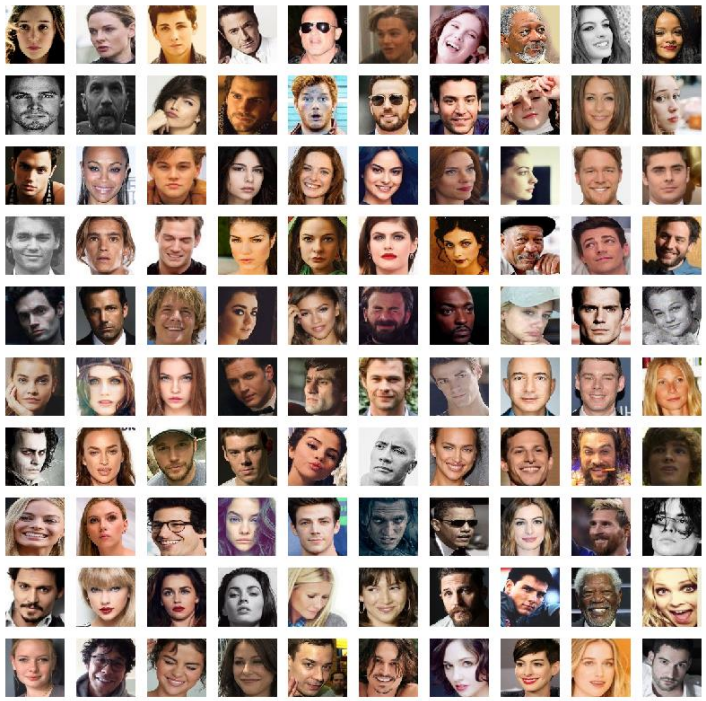

Fig 3.1 Pins Dataset [20]

3.1.2 Custom Dataset

To create a custom dataset, we recorded a video using our phone and extracted frames from the video using the MTCNN and YOLO models. To create a custom dataset, we recorded videos using our mobile device and extracted frames from these videos. In order to identify key frames specifically related to facial information, we utilized two popular models called MTCNN and YOLO. MTCNN is a deep learning model specifically designed for face detection and alignment. It detects facial landmarks and provides accurate bounding boxes around faces in each frame of the video. YOLO, on the other hand, is an object detection model that can identify various objects, including faces, within images or frames.

By applying MTCNN and YOLO to our extracted frames, we were able to detect and locate faces accurately. This step allowed us to isolate frames that contained important facial information, such as different expressions, poses, or identities. These frames, referred to as key frames, serve as crucial data points for our subsequent analysis and research in the field of computer vision and facial recognition. By creating a custom dataset based on the recorded mobile videos and utilizing MTCNN and YOLO for face detection, we obtained a collection of frames that specifically highlight facial features and variations. This custom dataset serves as a valuable resource for training and evaluating facial recognition algorithms, enabling us to advance our understanding and development in this domain.

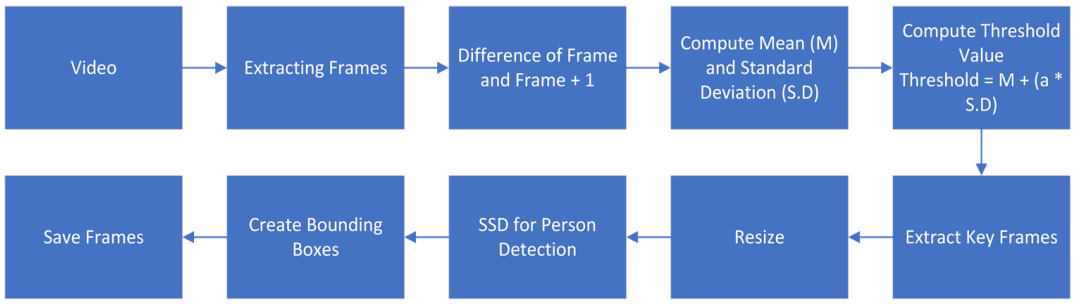

Fig 3. 2 Custom Dataset Generation [9]

3.2 Modules:

A Face Recognition system is composed of four distinct modules:

- Face detection.
- Face alignment.
- Feature Extraction.
- Classification and Matching.

We are focusing on evaluating numerous face recognition algorithms and examining how to suggest an enhanced face recognition method

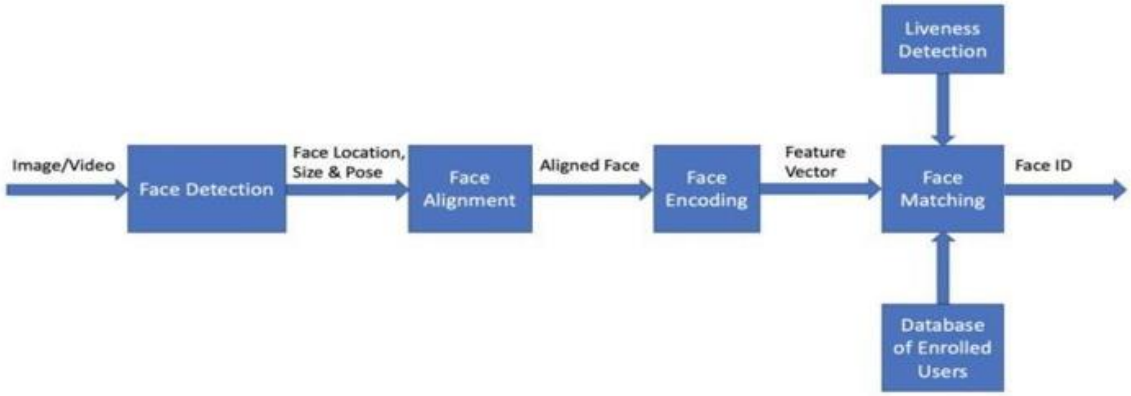

Fig 3.3 Working of Face Recognition System

3.2.1 Face Detection

Face detection serves as the initial stage in the pipeline of computer vision systems. The first module of a face recognition system is responsible for detecting and localizing faces in images or video frames. This crucial step utilizes various techniques, including Haar cascades, the Viola-Jones algorithm, and advanced deep learning models like MTCNN (Multi-task Cascaded Convolutional Networks) or SSD (Single Shot MultiBox Detector).

These methods enable accurate identification of the presence and precise location of faces within the input data. By employing robust face detection techniques, this module lays the foundation for subsequent face recognition processes, allowing for the extraction and analysis of facial features necessary for identification and verification tasks.

3.2.2 Face Alignment

After the initial step of face detection, the face alignment module comes into play. Its primary objective is to ensure precise alignment of the detected faces. This involves correcting for variations in pose, scale, and rotation, thereby standardizing the facial orientation across different images or frames. By aligning the faces, the module aims to minimize the impact of these variations, which can hinder accurate feature extraction.

The face alignment module employs sophisticated techniques and algorithms to achieve its goal. It may use methods such as facial landmark detection, where key facial points, such as the eyes, nose, and mouth, are identified and used as references for alignment. Alternatively, it may utilize geometric transformations to align faces based on specific geometric properties. By aligning the faces, the module improves the consistency and quality of subsequent feature extraction processes. It helps to bring the facial features into a standardized configuration, making them more comparable and facilitating accurate comparison and matching during the recognition phase.

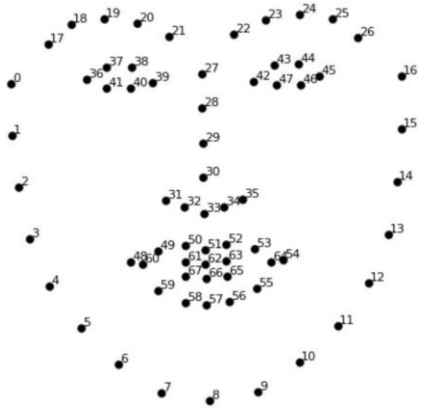

Fig 3.4 68-point Face Landmark [4]

3.2.3 Feature Extraction

The feature extraction module plays a crucial role in capturing discriminative facial features that encode the unique characteristics of individuals. This module aims to extract meaningful representations, commonly known as face embedding's, which serve as compact and informative representations of the faces. Various algorithms and techniques are employed in the feature extraction module, depending on the complexity and requirements of the system. One traditional approach is the use of statistical methods such as Eigenfaces and Fisherfaces. These methods analyze the statistical properties of facial images and extract the most significant features that contribute to the variations between individuals. Eigenfaces, for example, decompose facial images into a set of basis vectors that represent the principal components of the face space. Fisherfaces, on the other hand, aim to find discriminative features by maximizing the ratio of between-class and within-class variations. With the advent of deep learning, Convolutional Neural Networks (CNNs) have become increasingly popular for face feature extraction. CNNs are capable of automatically learning hierarchical representations from raw facial images. These networks consist of multiple layers of interconnected nodes that perform convolution, pooling, and non-linear transformations. Through a process of supervised training on large-scale face datasets, CNNs can effectively capture intricate facial patterns and extract high-level features that are highly discriminative.

The feature extraction module aims to create a compact representation of the face that encodes the essential information for face recognition. These face embeddings are often represented as high-dimensional feature vectors. The dimensionality of these vectors may vary depending on the chosen algorithm or technique. The extracted face embeddings serve as a foundation for face matching and identification. During the recognition phase, these embeddings are compared against a database of known face embeddings to determine the identity of a given face. Various similarity metrics, such as cosine similarity or Euclidean distance, are commonly used for this purpose.

The feature extraction module is a critical component in a face recognition system as it enables the system to capture and represent the distinctive facial characteristics of individuals. By extracting discriminative features, the module facilitates accurate and reliable face matching, enabling applications in diverse domains such as security, surveillance, access control, and personalized services.

3.2.4 Classification and Matching

The classification and matching module is responsible for comparing the extracted face features with known identities or a pre-existing database of enrolled faces. This module plays a crucial role in determining the most likely match and enabling accurate identification. The classification and matching module utilizes various algorithms and techniques to perform this task. One commonly used algorithm is Support Vector Machines (SVM), which is a supervised learning method that classifies data into different categories. In the context of face recognition, SVM can be trained to distinguish between different individuals based on their face embedding's. Another popular algorithm is k-Nearest Neighbors (k-NN), which classifies an unknown face by comparing its features with the k nearest neighbors in the database and assigning it the label of the majority of the neighbors.

With the rise of deep learning, deep neural networks have also been employed for face classification and matching. Convolutional Neural Networks (CNNs) have shown remarkable performance in various computer vision tasks, including face recognition. By training a deep neural network on a large-scale dataset of face images, the network can learn to discriminate between different individuals and assign them the correct labels. This approach enables end-to-end learning, where the network learns to extract features and perform classification simultaneously. The classification and matching module evaluates the similarity or distance between the extracted face features and the enrolled faces in the database. This comparison is typically done using similarity metrics such as cosine similarity or Euclidean distance.

The goal is to find the closest match or matches based on these similarity scores. The matching process can be optimized using techniques such as thresholding or ranking to determine the most likely identity.

The accuracy and performance of the classification and matching module are crucial for the overall effectiveness of the face recognition system. By employing robust classification algorithms and leveraging advanced techniques from machine learning and deep learning, this module ensures accurate identification and matching of faces even in challenging scenarios with variations in pose, illumination, and expression.

CHAPTER 4

Implementation

Detail of implementation of our system is presented

Block Diagram

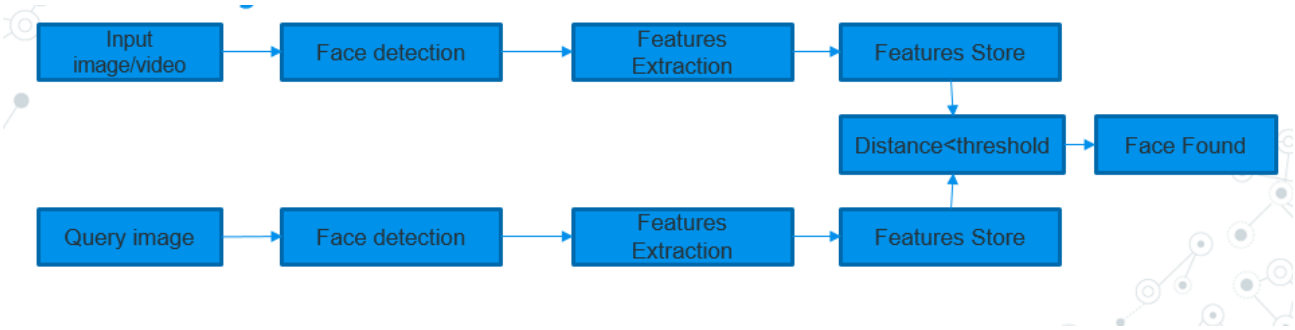

Fig 4.1 Working of Our System

4.1 Face Detection Using MTCNN

Multi-task Cascaded Convolutional Networks (MTCNN) [21] is a framework designed to address both face detection and face alignment. This technique involves employing a sequence of three convolutional networks, which are capable of identifying faces and locating facial landmarks like the eyes, nose, and mouth. The initial step involves resizing the image into multiple scales, creating an image pyramid. This pyramid serves as the input for the following cascaded network, which consists of three stages.

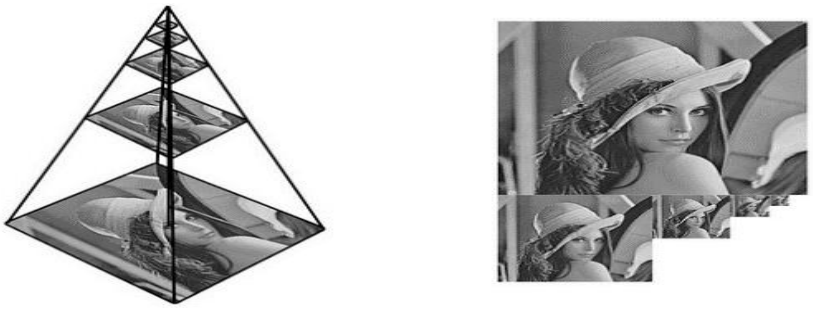

Fig 4.2 Image Pyramid in MTCNN[21]

4.2 Three Stages of MTCNN:

Following are the stages of MTCNN:

- The Proposal Network (P-Net)
- The Refine Network (R-Net)
- The Output Network (O-Net)

4.2.1 The Proposal Network (P-Net)

The Proposal Network [22] is initial stage of the process involves a fully convolutional network (FCN), Unlike a CNN, the cascaded network utilized in this process is different in that it does not incorporate a dense layer within its architecture. This FCN serves as the Proposal Network within the face detection system. Its primary role is to generate candidate windows and bounding box regression vectors for accurate face detection. Bounding box regression is a widely-used technique for predicting the localization of pre-defined objects, such as faces in this case. Once the bounding box vectors have been obtained, any overlapping regions are refined to produce the final output, which includes all candidate windows that have undergone refinement to minimize the extent of candidates.

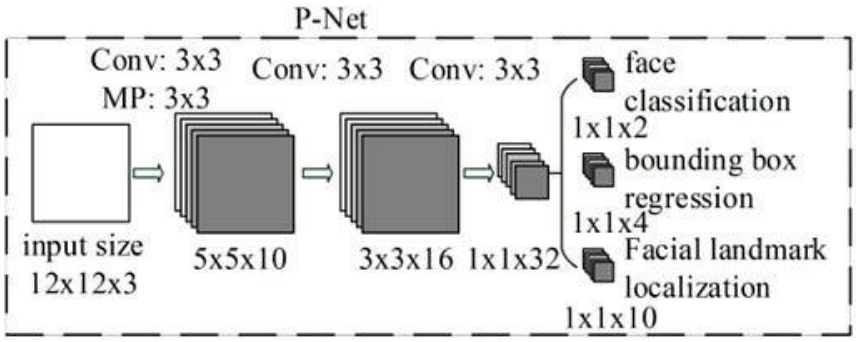

Fig 4. 3 Working of P-Net [22]

4.2.2 The Refine Network (R-Net)

All candidates generated from the Proposal Network (P-Net) are forwarded to the Refine Network, which distinguishes itself from the FCN by incorporating a dense layer in its network architecture.

The Refine Network [22] (R-Net) is responsible for further reducing the quantity of candidates through calibration using bounding box regression and merging overlapping candidates using non-maximum suppression (NMS). Additionally, the R-Net outputs a 4-element vector representing the bounding box for the detected face, a 10-element vector for facial landmark localization, and a classification score indicating whether the input corresponds to a face or not.

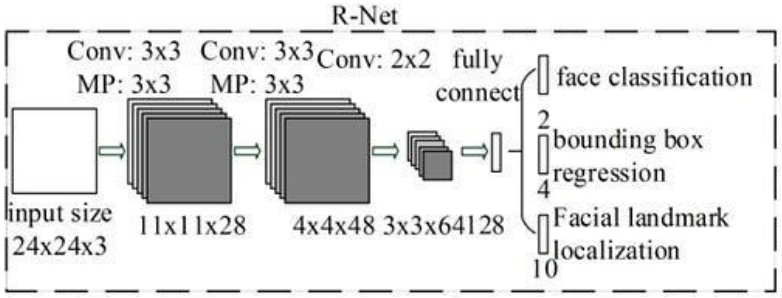

Fig 4.4 Working of R-Net [22]

4.2.3 The Output Network (O-Net)

The Output Network [22] is comparable to the R-Net in its function, but its primary objective is to offer a comprehensive and detailed description of the detected face. This network is responsible for determining the coordinates of the five facial landmarks, including those for the eyes, nose, and mouth.

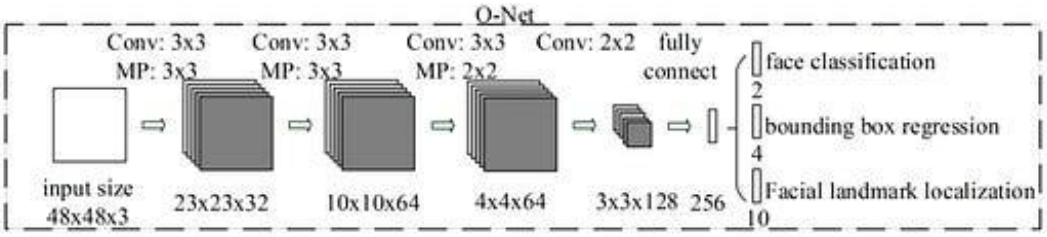

Fig 4.5 Working of O-Net [22]

4.3 Face Detection Using YOLO

YOLO (You Only Look Once) is an object detection algorithm that is extensively utilized in computer vision applications, including face detection. YOLO [23] takes an input image or frame and directly predicts coordinates of the bounding box and probabilities for different classes for multiple objects in a single pass through a convolutional neural network (CNN).YOLO operates on the entire image at once, dividing it into a grid of predefined cells. Each grid cell predicts multiple bounding boxes along with corresponding confidence scores that indicate the likelihood of the box containing an object. The confidence scores are based on the presence of objects and the accuracy of the predicted bounding boxes.

4.3.1 YOLOv2

YOLOv2 [23], developed by Joseph Redmon and Ali Farhadi in 2016, surpasses YOLO in terms of both speed and accuracy while also expanding its object detection capabilities. By partitioning the image into a grid of cells, YOLOv2 operates efficiently. The notable advancement of YOLOv2 lies in its improved speed and precision compared to its predecessor. It outperforms YOLO in accurately detecting a diverse range of objects. It is a significant improvement over the original YOLO algorithm, which was introduced in 2015 [24]. YOLOv2 is faster and more accurate than YOLO, and it can be used to detect a wider variety of objects.YOLOv2 works by dividing the image into a grid of cells. Each cell predicts a bounding box and a class label for the object that is most likely to be located in that cell. The bounding boxes are predicted using a convolutional neural network (CNN). The class labels are predicted using a softmax layer. YOLOv2 underwent training on the ImageNet dataset, an extensive collection comprising more than 1.2 million images belonging to 1,000 distinct object classes. To further enhance its performance, the model was fine-tuned using the COCO dataset, which encompasses over 330,000 images representing 80 diverse object classes. Notably, YOLOv2 has exhibited remarkable efficacy in face detection. Its capabilities were evaluated using the Wider Face dataset, renowned for its challenging nature due to occlusions, pose variations, and varying illumination

conditions. Impressively, YOLOv2 achieved an mAP (mean Average Precision) score of 0.95 on the Wider Face dataset, surpassing the previous state-of-the-art results.

4.3.2 Improved YOLOv2

There have been a number of improvements to YOLOv2[24] since it was first introduced. One of the most significant improvements is the use of a residual network (ResNet) for the backbone network. ResNets have been shown to be very effective for object detection, and they have been used to improve the performance of YOLOv2 on a number of datasets. Another improvement that has been made to YOLOv2 is the use of a new loss function. The original YOLOv2 loss function utilized the mean squared error (MSE) as its foundation. However, MSE has been shown to be less effective than other loss functions for object detection.

A new loss function, called the focal loss, has been shown to be more effective for YOLOv2.The focal loss is a sigmoid cross-entropy loss function that is weighted by the inverse of probability of ground truth label. This means that the loss is higher for objects that are difficult to detect. The focal loss has been shown to improve the performance of YOLOv2 on a number of datasets.

Table 4. 1 Comparison between Yolov2 and Improved Yolov2

Features	**YOLOv2**	**IMPROVED YOLOv2**
Architecture	Darknet-19	Darknet-53
Input Size	416x416	608x608
Anchors	9	6
Classes	20	60
Training Data	COCO	COCO, ImageNet, VOC
FPS	45	25

4.3.3 YOLOv3

YOLOv3 [24] is a single-stage object detection model that was first introduced in 2018. It is an improvement over the previous YOLOv2 model, and It has demonstrated state-of-the-art performance on various object detection benchmarks, establishing its superiority in the field.YOLOv3 employs a convolutional neural network (CNN) to accurately predict bounding boxes and class probabilities for objects present in an image. The network is trained on a Vast dataset of images that have been labeled with the objects that they contain.YOLOv3 has a number of advantages over other object detection models. It is fast, accurate, and easy to train. It is also able to identify objects in real time, which makes it ideal for a range of applications, such as surveillance and self-driving automobiles.

4.4 Feature Extraction Using FaceNet

The FaceNet[25] model is a deep convolutional neural network (CNN) that was developed by Google in 2015. It is trained on a massive dataset of images of faces, and possesses the capability to extract high-quality features from the images. These features are then used to create a 128-dimensional vector representation of each face, called a face embedding. The FaceNet model undergoes training using a loss function referred to as the triplet loss. The triplet loss serves to promote increased similarity between vectors representing the same identity, while simultaneously encouraging decreased similarity between vectors representing different identities. The triplet loss is implemented by constructing triplets of images, with each triplet comprising an anchor image, a positive image, and a negative image. The anchor image depicts a specific individual, the positive image portrays the same individual, and the negative image represents a different individual. The objective of the triplet loss is to minimize the distance between the anchor and positive images, signifying their similarity, while simultaneously maximizing the distance between the anchor and negative images, denoting their dissimilarity. This encourages the model to learn embedding’s that capture the unique characteristics of each individual, enabling effective face recognition.

The FaceNet [25] model has been shown to be very effective at face recognition. It has achieved excellent results on a number of benchmark datasets, including the Labeled Faces in the Wild (LFW) dataset and the YouTube Faces Database. The FaceNet model is currently used in a number of commercial products, including the Google Pixel phone and the Facebook Portal.

4.4.1 Architecture of FaceNet

FaceNet [25] utilizes an end-to-end learning approach in its architecture to train the model. It incorporates deep learning models like ZF-Net or Inception Network, augmented with 1x1 convolutions for parameter reduction. These models produce an embedding, denoted as f(x), for a given input image, which is then L2-normalized. The resulting embeddings are subsequently fed into the loss function for computing the loss.The primary objective of the loss function is to minimize the squared distance between the embeddings of two images that depict the same individual, regardless of variations in image conditions and poses.

Simultaneously, the loss function aims to maximize the distance between the embeddings of images representing different individuals. This objective is accomplished by employing a novel loss function known as Triplet loss, which enforces a margin between the embeddings of faces belonging to distinct identities. By optimizing the Triplet loss, the model learns to generate discriminative embeddings that can effectively capture and differentiate between facial characteristics, enabling accurate face recognition.

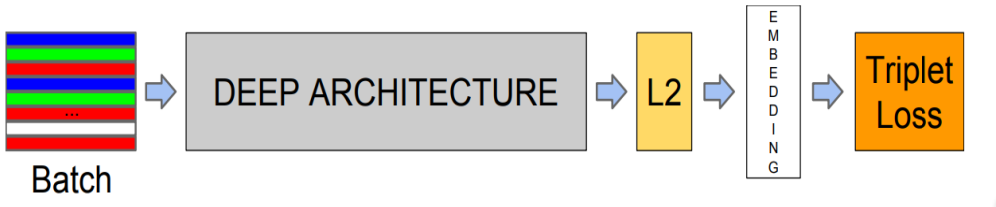

Fig 4.5 FaceNet Architecture [25]

4.4.2 Training

The FaceNet [25] is trained using a loss function called the triplet loss. The objective of the triplet loss is to promote increased similarity between vectors representing the same identity, while simultaneously encouraging decreased similarity between vectors representing different identities. To achieve this, the triplet loss operates by forming triplets of images, wherein each triplet comprises an anchor image, a positive image, and a negative image. The anchor image represents a specific individual, the positive image depicts the same individual, and the negative image corresponds to a different individual. The primary objective of the triplet loss[15] is twofold: first, to minimize the distance between the anchor image and the positive image, and second, to maximize the distance between the anchor image and the negative image. By optimizing the embeddings, the triplet loss aims to minimize the distance between the anchor and positive embeddings, while simultaneously maximizing the distance between the anchor and negative embeddings.

Fig 4.6 Triplet Loss Function [25]

4.5 Web Portal

The end product of this project is to develop a website, powered by Flask, which serves as the primary interface for displaying all the project outputs.

4.5.1 Python Flask

Flask is a lightweight and user-friendly micro web framework for building web applications and APIs with Python. It offers a simple and straightforward approach, making it highly favored among developers for creating small to medium-sized web projects.

Its emphasis on simplicity and ease of use has contributed to its widespread adoption in the web development community.

4.5.2 Hypertext Markup Language

HTML (Hypertext Markup Language) is a standard markup language used for creating the structure and content of web pages. It provides a set of tags and elements that define the various components of a webpage, such as headings, paragraphs, images, links, forms, and more. HTML is the backbone of the web, as it allows web browsers to interpret and display the content in a structured and visually appealing manner. Developers use HTML to create the structure and layout of web pages, organizing content and specifying its appearance using CSS (Cascading Style Sheets). With HTML, you can create interactive and dynamic web pages that can be accessed and viewed by users across different platforms and devices.

4.5.3 Cascading style sheets

CSS (Cascading Style Sheets) is a style sheet language used to describe the visual presentation of a document written in HTML or XML. It provides a set of rules and properties that define how elements of a web page should be displayed, such as their colors, fonts, sizes, layouts, and other stylistic aspects. With CSS, you can control the appearance and layout of various elements, including text, images, backgrounds, borders, and more.

CHAPTER 5

Results and Analysis

The experimental results demonstrate that MTCNN, YOLOv3, and FaceNet exhibit high effectiveness in the following scenarios: face detection and feature extraction from images. The performance of these three proposed models can be evaluated through several metrics, including training accuracy, validation accuracy, confusion matrix, and classification report.

5.1 Confusion Matrix

A confusion matrix[26] is a table of size n x n (where n is the number of rows and columns). It is used to describe the overall performance of a classification algorithm. Confusion matrix Visualize and summarize the performance of classification algorithms. Confusion matrix it is very useful for binary classification. It mainly consists of four basic contents which are given below. The confusion matrix is particularly valuable in binary classification problems, where there are two classes or categories to be predicted. It comprises four fundamental elements:

- **True Positive (TP)**

It represents the number of instances correctly classified as positive.

- **True Negative (TN)**

It indicates the number of instances correctly classified as negative.

- **False Positive (FP)**

It denotes the number of instances incorrectly classified as positive.

- **True Negative (TN)**

It signifies the number of instances incorrectly classified as negative.

5.2 Classification Report

The visualization of the classification report presents the precision, recall, F1-score.

- **Accuracy**

The Accuracy [27] rate refers to the number of correct predictions made by a model. It is calculated using the following equation:

$$Accuarcy = \frac{no\ of\ correct\ predictions}{total\ no\ of\ predictions\ made} \tag{5.1}$$

- **Precision**

Precision [27], also known as the positive predictive value, represents the proportion of correctly predicted positive instances out of all instances predicted as positive. It is calculated using the following equation:

$$Precision = \frac{true\ positive}{true\ positive + false\ positive} \tag{5.2}$$

- **Recall**

Recall [27], also known as sensitivity or true positive rate, measures the model's ability to correctly identify positive instances, or in other words, the proportion of true positives that are correctly detected. A high recall indicates that the model is effective in detecting true positives, while a low recall suggests that there are many false negatives. Recall is calculated using the following equation:

$$Recall = \frac{true\ positive}{true\ positive + false\ negative} \tag{5.3}$$

- **F1-Score**

The F1-score [27] is a metric that provides a balanced measure of both precision and recall. It is the weighted harmonic mean of precision and recall, offering a better estimate than accuracy when dealing with imbalanced datasets or when both false positives and false negatives need to be considered. The F1-score is calculated using the following equation:

$$F1 - score = 2\frac{precision}{precision + recall} \tag{5.4}$$

5.3 Models Results:

The accuracy of all models has been evaluated using a common classifier, Support vector machine (SVM).

5.3.1 Haar Cascade Results

The complete set of evaluation graphs is given below

- **Training and Validation Loss Curve**:

The training and validation loss curves helps us assess the model's learning progress. The training loss that had been reach on epoch 0 was 0.62. Loss value decreased significantly until epoch 30. Loss value on that epoch was approximately at 2.0. The training loss kept decreasing until epoch 1.5. Training loss on epoch 2 was approximately at 0.22. Until epoch 4, the training loss that had been reached by Haar Cascade was 0.22. The validation loss on epoch 0 was 0.59. Validation loss kept decreasing until 0.2 on epoch 1.5. Validation loss started to converge approximately at epoch 2.0. Validation loss on that epoch was 0.17. The validation loss reached on epoch 4 was 0.244.The Training loss, which measures how well the model fits the training data. The goal is for this loss to decrease over time as the model adjusts its parameters to make better predictions. The validation loss, which evaluates how well the model generalizes to new, unseen data. It measures the model's performance on a separate validation dataset. Ideally, the validation loss should also decrease during training,

indicating that the model is learning useful patterns without overfitting. In Fig[5.1] both the training loss and validation loss are decreasing with respect to the y-axis, it indicates that both the training and validation performance of the model are improving over time.

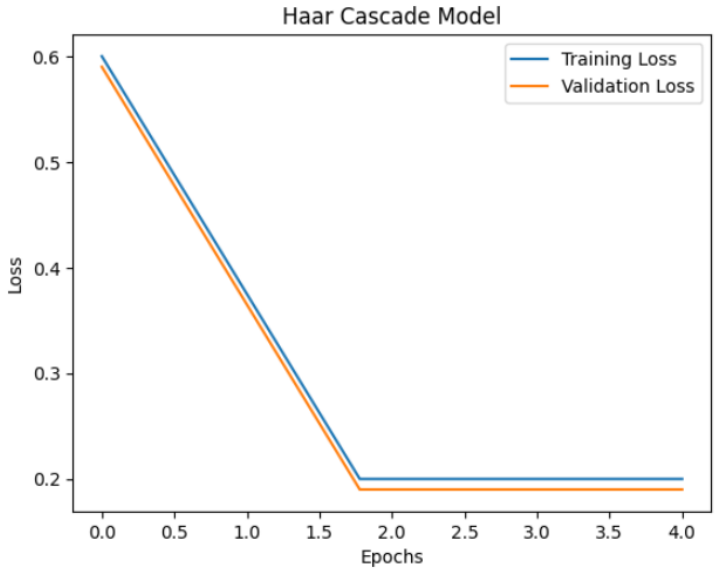

Fig 5.1 Training and Validation Loss Curve of HaarCascade Model

- **Training and Validation Accuracy Curve**:

The training and validation accuracy curves allows us to assess the model's learning progress. The training accuracy result for Haar Cascade Model using the dataset that had been increased can be seen on Fig[5.2]. The accuracy on epoch 0 reached was 30%. The accuracy increased significantly on the first 2 epoch. The accuracy reached on epoch 2 was approximately on 40 %. On the next epoch, the accuracy kept increased slowly until epoch 5. The accuracy on epoch 5 was approximately on 75%. The accuracy started to converge approximately on epoch 8. Until epoch 10, the accuracy that had been reached by Haar Cascade Model was 78.4 %. The validation accuracy on epoch 0 reached 40%. The validation accuracy increased until 50% on epoch 4. Validation accuracy kept increased slowly until 75% on epoch 8. It started to converge on epoch 8.

Until epoch 10, the accuracy using the validation set by Haar Cascade model was 78.4%. Training accuracy, measures how well the model performs on the training dataset. It shows the percentage of correctly classified or predicted examples from the training data. As the model learns from the training data, the training accuracy ideally increases over time. Validation accuracy, measures how well the model performs on a separate validation dataset that was not used during training. It represents the percentage of correctly classified or predicted examples from this independent dataset. The goal is for the validation accuracy to also increase during training, indicating that the model can generalize its learned knowledge to new, unseen data.

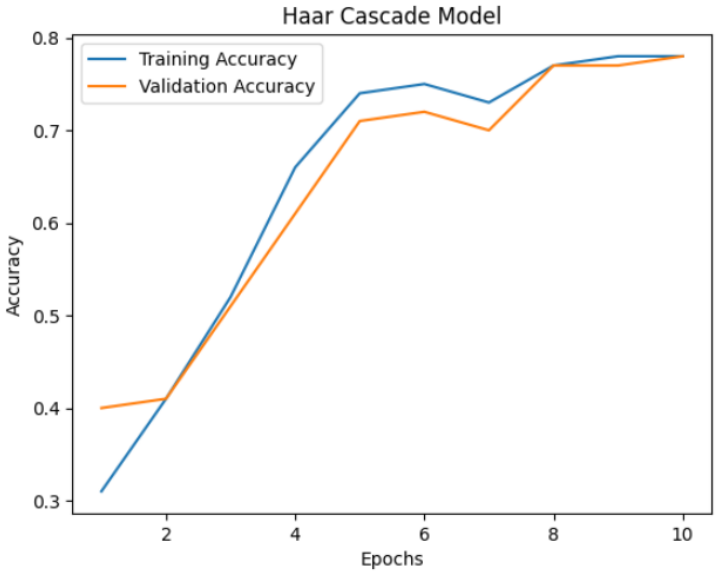

Fig 5.2 Training and Validation Accuracy Curve of HaarCascade Model

- **Confusion Matrix**:

A confusion matrix is a table of size n x n. In Fig[5.3] it is used to describe the overall performance of a classification algorithm. It comprises four fundamental elements. True Positive (TP), it represents the number of instances correctly classified as positive. True Negative (TN), it indicates the number of instances correctly classified as negative. False Positive (FP), it denotes the number of instances incorrectly classified

as positive. Fall negative (FN), It signifies the number of instances incorrectly classified as negative.

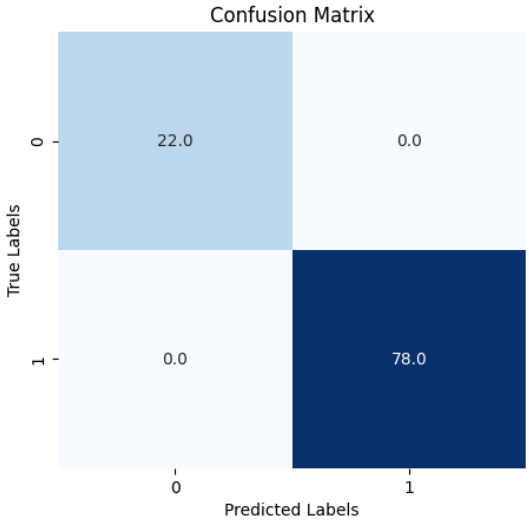

Fig 5.3 Confusion Matrix of HaarCascade Model

- **Classification Report**

Table 5.1 Classification Report of Haar Cascade Model

	Precision	Recall	F1-Score
	0.79	**0.58**	**0.67**
	0.78	**0.54**	**0.64**

5.3.2 HOG Results

- **Training and Validation Loss Curve**:

The training and validation loss curves serve as indicators of the model's learning progress. Initially, at epoch 0, the training loss reached 0.8, which subsequently decreased significantly until epoch 1.0, where it stabilized around 0.6. Further progress was observed until epoch 1.5, with a training loss of approximately 0.3. From epoch 2 to epoch 4, the training loss consistently remained at 0.2, as determined by the Hog. Regarding the validation loss, it began at 0.69 on epoch 0 and steadily decreased until reaching 0.3 by epoch 1.3. Validation loss then began to converge, showing stability around 0.19 on epoch 2. Finally, at epoch 4, the validation loss recorded a value of 0.18.The training loss quantifies the model's fitting accuracy to the training data, with the objective of decreasing over time as the model adjusts its parameters to enhance predictions. On the other hand, the validation loss assesses the model's generalization ability to unseen data, measured through a separate validation dataset. Ideally, the validation loss should decrease during training, indicating the model's acquisition of useful patterns without overfitting.

In Fig[5.4], both the training and validation loss exhibit a declining trend along the y-axis, suggesting continuous improvement in the model's performance for both training and validation data over time.

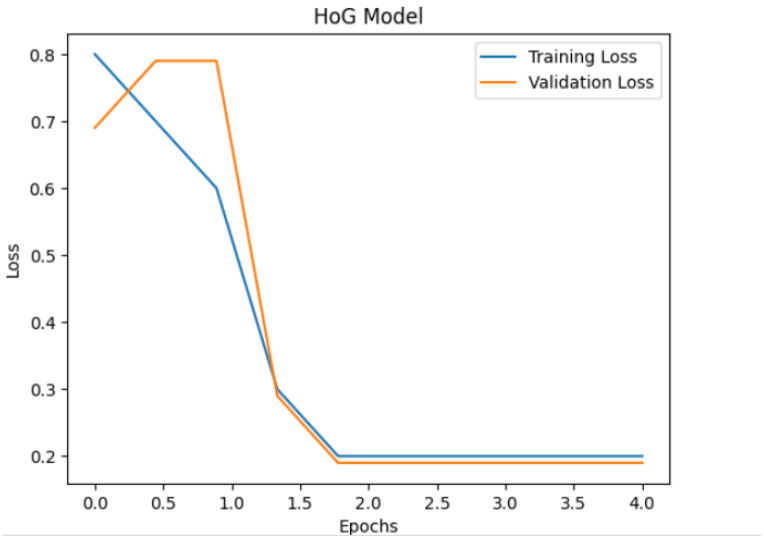

Fig 5.4 Training and Validation Loss Curve of HOG Model

- **Training and Validation Accuracy Curve**:

The training and validation accuracy curves allows us to assess the model's learning progress. The training and validation accuracy curves provide valuable insights into the model's learning progress. Fig [5.5] showcases the training accuracy results of the HoG Model. At epoch 0, the accuracy reached 40%. Subsequently, there was a significant increase in accuracy during the first two epochs, with approximately 50% accuracy achieved at epoch 2. The accuracy continued to gradually improve, reaching approximately 73% at epoch 5. Afterward, the accuracy started to stabilize, showing convergence around epoch 8. By epoch 10, the HoG Model attained an accuracy of 85%.Similarly, the validation accuracy at epoch 0 was 50%, which gradually increased to 70% by epoch 5. The validation accuracy continued to improve slowly, reaching 73% at epoch 8. Convergence was observed around epoch 8, and by epoch 10, the HoG Model achieved an accuracy of 85% on the validation set. Training accuracy measures the model's performance on the training dataset by indicating the percentage of correctly classified or predicted examples. As the model learns from the training data, it is expected that the training accuracy will increase over time. Validation accuracy, on the other hand, evaluates the model's performance on a separate validation dataset that was not used during training. It represents the percentage of correctly classified or predicted examples from this independent dataset. The objective is for the validation accuracy to increase during training, indicating the model's ability to generalize its learned knowledge to new, unseen data.

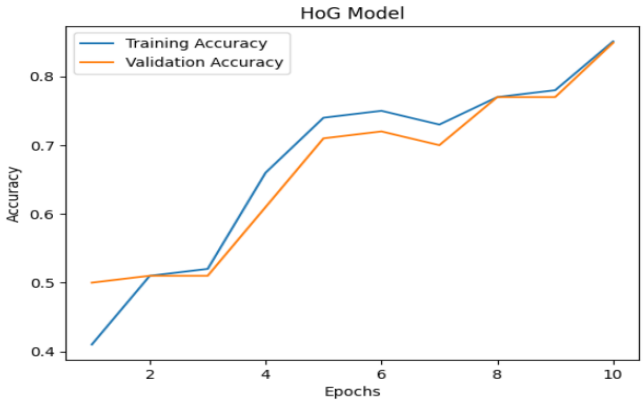

Fig 5.5 Training and Validation Accuracy Curve of HOG Model

- **Confusion Matrix**:

A confusion matrix is a table of size n x n. In Fig[5.6] it is used to describe the overall performance of a classification algorithm. It comprises four fundamental elements. True Positive (TP), it represents the number of instances correctly classified as positive. True Negative (TN), it indicates the number of instances correctly classified as negative. False Positive (FP), it denotes the number of instances incorrectly classified as positive. Fall negative (FN), It signifies the number of instances incorrectly classified as negative.

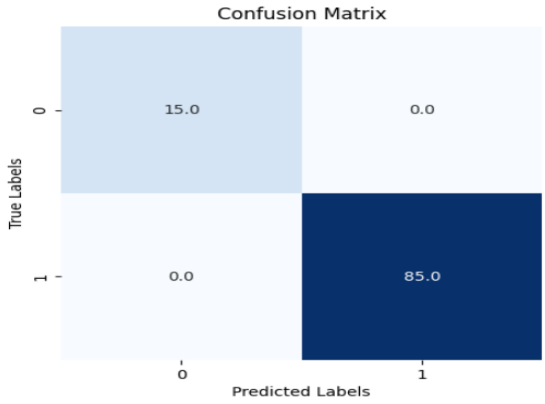

Fig 5.6 Confusion matrix of HOG Model

- **Classification Report**

Table 5.2 Classification report of HOG Model

	Precision	Recall	F1-score
	0.88	**0.81**	**0.84**
	0.80	**0.73**	**0.76**

5.3.3 CNN Results

- **Training and Validation Loss Curve**:

The training and validation loss curves are essential for evaluating the model's learning progress. Initially, at epoch 0, the training loss achieved a value of 2.25, which subsequently experienced significant reduction until epoch 1.0, where it stabilized around 1.15. Further advancements were observed until epoch 2.0, with the training loss reaching approximately 0.72. From epoch 2 to epoch 4, the training loss decreased to 0.45, as determined by the CNN model. Regarding the validation loss, it started at 1.40 on epoch 0 and steadily decreased until reaching 1.22 by epoch 1.0. Subsequently, the validation loss began to converge, demonstrating stability around 0.75 by epoch 2. Finally, at epoch 4, the validation loss recorded a value of 0.45.The training loss serves as a metric to assess how well the model fits the training data, with the objective of decreasing over time as the model fine-tunes its parameters to improve predictions. On the other hand, the validation loss measures the model's ability to generalize to unseen data, using a separate validation dataset. Ideally, the validation loss should decrease during training, indicating that the model is learning useful patterns without overfitting. In Fig [5.7], both the training and validation loss exhibit a declining trend along the y-axis, indicating continuous improvement in the model's performance for both the training and validation data over time.

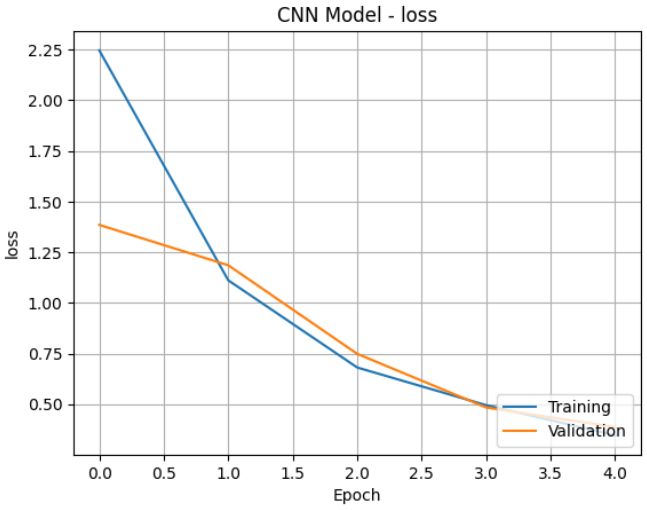

Fig 5.7 Training and Validation Loss Curve of CNN Model

- **Training and Validation Accuracy Curve**:

The training and validation accuracy curves provide valuable insights into the learning progress of the model. The CNN Model training accuracy results with an expanded dataset are depicted in Fig [5.8]. At the beginning, on epoch 0, the accuracy stood at 30%. Subsequently, there was a significant improvement in accuracy during the first two epochs, reaching approximately 49% by epoch 0.5. Gradual progress was observed thereafter, with the accuracy reaching approximately 78% by epoch 2. From that point, the accuracy started to stabilize, converging around epoch 3. By epoch 4, the CNN Model achieved an accuracy of 86%. Likewise, the validation accuracy started at 58% on epoch 0 and gradually increased to 75% by epoch 2. Subsequently, it continued to improve slowly, reaching 81% by epoch 3. Convergence was observed around epoch 3, and by epoch 4, the CNN Model demonstrated an accuracy of 83% on the validation set. Training accuracy measures the model's performance on the training dataset, indicating the percentage of correctly classified or predicted examples. As the model learns from the training data, it is expected that the training accuracy will increase over time. Validation accuracy, on the other hand, evaluates the model's performance on a separate validation dataset that was not used during training. It represents the percentage of correctly classified or predicted examples from this independent dataset. The goal is for the validation accuracy to increase during training, indicating the model's ability to generalize its learned knowledge to new, unseen data.

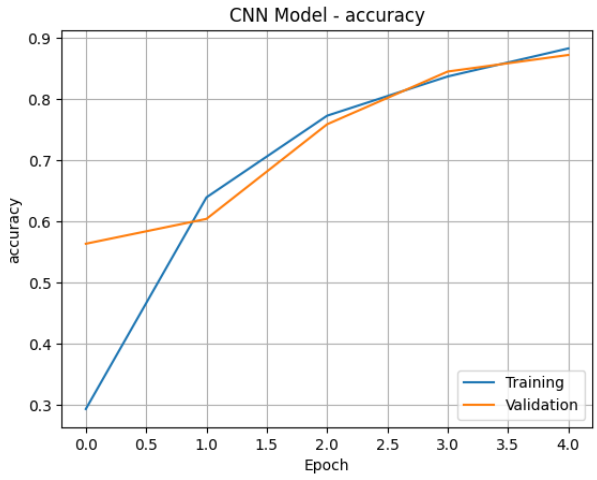

Fig 5.8 Training and Validation Accuracy Curve of CNN Model

- **Confusion Matrix**:

A confusion matrix is a table of size n x n. In Fig[5.9] it is used to describe the overall performance of a classification algorithm. It comprises four fundamental elements. True Positive (TP), it represents the number of instances correctly classified as positive. True Negative (TN), it indicates the number of instances correctly classified as negative. False Positive (FP), it denotes the number of instances incorrectly classified as positive. Fall negative (FN), It signifies the number of instances incorrectly classified as negative.

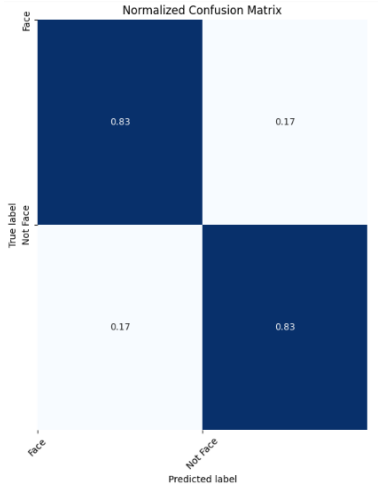

Fig 5.9 Confusion Matrix of CNN Model

- **Classification Report**

Table 5.3 Classification report of CNN Model

	Precision	Recall	F1-score
	0.83	**0.92**	**0.87**
	0.74	**0.89**	**0.81**

5.3.4 MTCNN Results

- **Training and Validation Loss Curve**:

The training and validation loss curves helps us assess the model's learning progress. The training loss that had been reach on epoch 0 was 0.57. Loss value decreased significantly until epoch 5. Loss value on that epoch was approximately at 0.1. The training loss kept decreasing until epoch 12.5. Training loss on epoch 12.5 was approximately at 0.08. Until epoch 20, the training loss that had been reached by MTCNN was 0.05. The validation loss on epoch 0 was 0.65. Validation loss kept decreasing until 0.13 on epoch 5.0. Validation loss started to converge approximately at epoch 10.0. Validation loss on that epoch was 0.1. The validation loss reached on epoch 20.0 was 0.04.The Training loss, which measures how well the model fits the training data. The goal is for this loss to decrease over time as the model adjusts its parameters to make better predictions. The validation loss, which evaluates how well the model generalizes to new, unseen data. It measures the model's performance on a separate validation dataset. Ideally, the validation loss should also decrease during training, indicating that the model is learning useful patterns without overfitting. In Fig[5.10] both the training loss and validation loss are decreasing with respect to the y-axis, it indicates that both the training and validation performance of the model are improving over time.

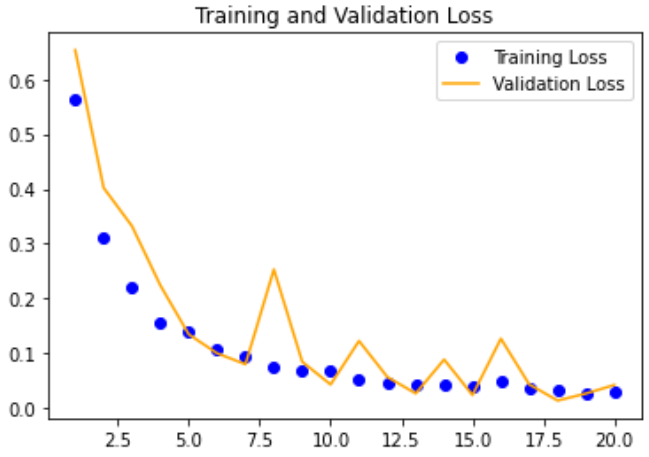

Fig 5.10 Training and Validation Loss Curve of MTCNN Model

- **Training and Validation Accuracy Curve**:

The training and validation accuracy curves allows us to assess the model's learning progress. The training accuracy result for MTCNN using the dataset that had been increased can be seen on Fig 5.11. The accuracy on epoch 0 reached was 74%. The accuracy increased significantly on the first 2 epoch. The accuracy reached on epoch 2.5 was approximately on 84 %. On the next epoch, the accuracy kept increased slowly until epoch 7.5. The accuracy on epoch 5 was approximately on 95%. The accuracy started to converge approximately on epoch 12.5. Until epoch 20.0, the accuracy that had been reached by MTCNN was 97 %. The validation accuracy on epoch 0 reached 60%. The validation accuracy increased until 95% on epoch 7.5. Validation accuracy kept increased slowly until 96% on epoch 17.5. It started to converge on epoch 17.5. Until epoch 20.0, the accuracy using the validation set by MTCNN was 97%. Training accuracy, measures how well the model performs on the training dataset. It shows the percentage of correctly classified or predicted examples from the training data. As the model learns from the training data, the training accuracy ideally increases over time. Validation accuracy, measures how well the model performs on a separate validation dataset that was not used during training. It represents the percentage of correctly classified or predicted examples from this independent dataset. The goal is for the validation accuracy to also increase during training, indicating that the model can generalize its learned knowledge to new, unseen data.

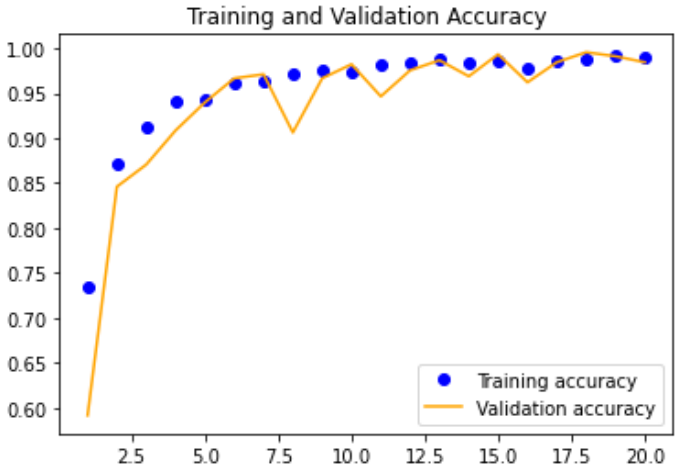

Fig 5.11 Training and Validation Accuracy Curve of MTCNN Model

- **Confusion Matrix**:

A confusion matrix is a table of size n x n. In Fig [5.12] it is used to describe the overall performance of a classification algorithm. It comprises four fundamental elements. True Positive (TP), it represents the number of instances correctly classified as positive. True Negative (TN), it indicates the number of instances correctly classified as negative. False Positive (FP), it denotes the number of instances incorrectly classified as positive. Fall negative (FN), It signifies the number of instances incorrectly classified as negative.

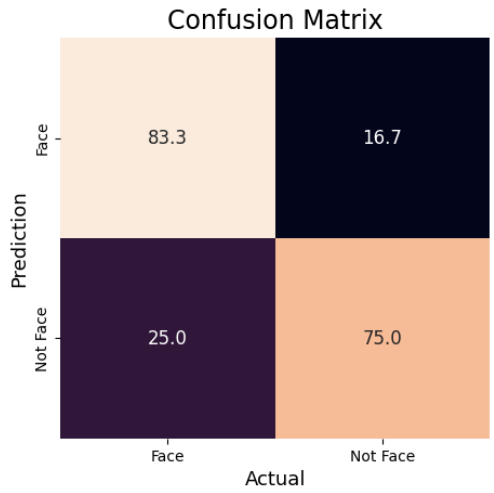

Fig 5.12 Confusion Matrix of MTCNN Model

- **Classification Report**

Table 5.4 Classification report of MTCNN Model

	Precision	Recall	F1-score
	0.95	**1.00**	**0.97**
	1.00	**0.95**	**0.97**

5.3.5 Face Detection from Image Using MTCNN

The implementation of our trained model MTCNN on different images for Face detection.

Fig 5.13 Single Face Detection from Image Using MTCNN

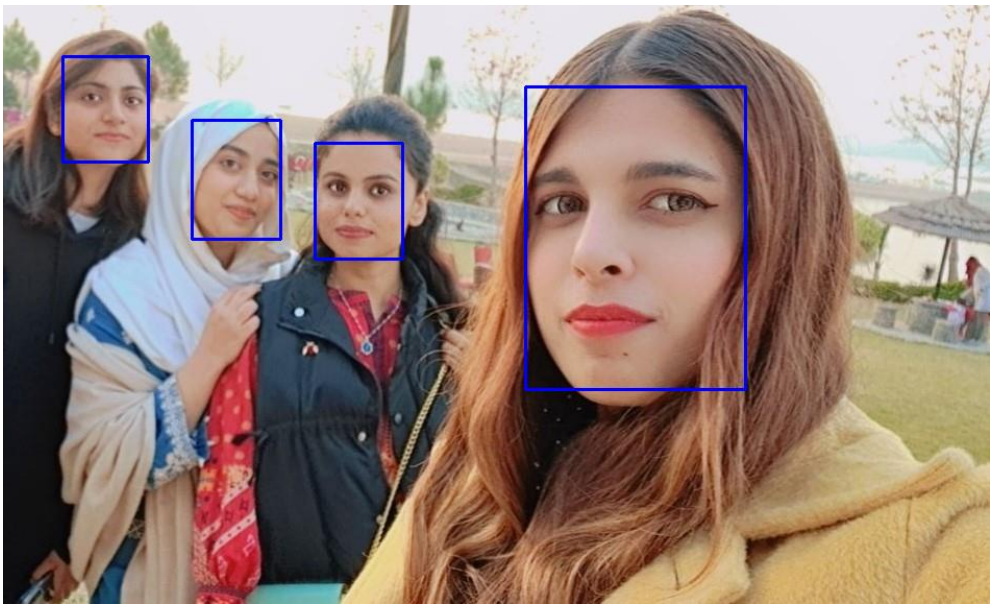

Fig 5.14 Multi Face Detection from Image Using MTCNN

5.3.6 Face Detection from Video Using MTCNN

The implementation of our trained model MTCNN on different videos for Face detection.

Fig 5.15 Multi Face Detection from Video Using MTCNN

Fig 5.16 Multi Face Detection from Video Using MTCNN

5.3.7 YOLOv3 Results

- **Training and Validation Loss Curve**:

The training and validation loss curves serve as indicators of the model's learning progress. Initially, at epoch 0, the training loss reached 0.13, which subsequently decreased significantly until epoch 1.0, where it stabilized around 0.6. Further progress was observed until epoch 2.5, with a training loss of approximately 0.03. From epoch 7.5 to epoch 20.0, the training loss consistently remained at 0.01, as determined by the Yolov3. Regarding the validation loss, it began at 0.40 on epoch 0 and steadily decreased until reaching 0.05 by epoch 2.5. Validation loss then began to converge, showing stability around 0.01 on epoch 7.5. Finally, at epoch 20.0, the validation loss recorded a value of 0.01.The training loss quantifies the model's fitting accuracy to the training data, with the objective of decreasing over time as the model adjusts its parameters to enhance predictions. On the other hand, the validation loss assesses the model's generalization ability to unseen data, measured through a separate validation dataset. Ideally, the validation loss should decrease during training, indicating the model's acquisition of useful patterns without overfitting. In Fig[5.17], both the training and validation loss exhibit a declining trend along the y-axis, suggesting continuous improvement in the model's performance for both training and validation data over time.

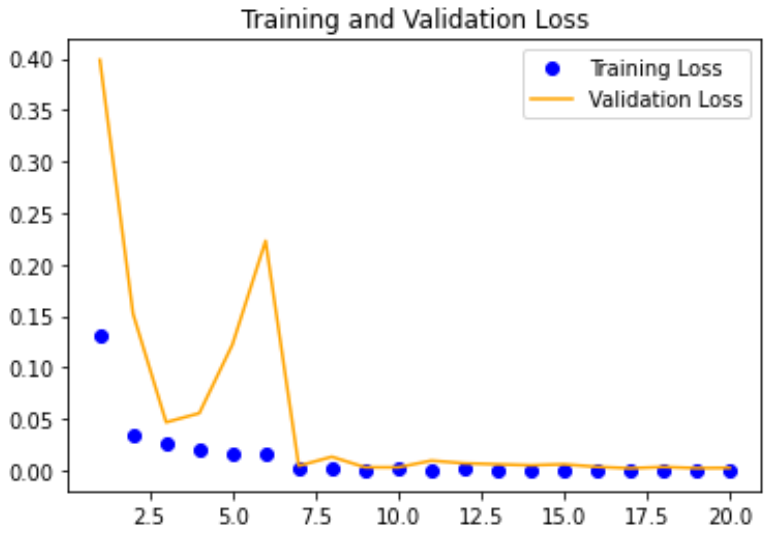

Fig 5.17 Training and Validation Loss Curve of YOLOv3 Model

- **Training and Validation Accuracy Curve**:

The training and validation accuracy curves allows us to assess the model's learning progress. The training and validation accuracy curves provide valuable insights into the model's learning progress. Fig [5.18] showcases the training accuracy results of the Yolov3 Model. At epoch 0, the accuracy reached 94.2%. Subsequently, there was a significant increase in accuracy during the first two epochs, with approximately 98.45% accuracy achieved at epoch 2.5. The accuracy continued to gradually improve, reaching approximately 98.5% at epoch 7.5. Afterward, the accuracy started to stabilize, showing convergence around epoch 7.5. By epoch 20, the Yolov3 Model attained an accuracy of 99%. Similarly, the validation accuracy at epoch 0 was 90%, which gradually increased to 98% by epoch 2.5. The validation accuracy continued to improve slowly, reaching 98.5% at epoch 7.5. Convergence was observed around epoch 7.5, and by epoch 20.0, the Yolov3 Model achieved an accuracy of 99% on the validation set. Training accuracy measures the model's performance on the training dataset by indicating the percentage of correctly classified or predicted examples. As the model learns from the training data, it is expected that the training accuracy will increase over time. Validation accuracy, on the other hand, evaluates the model's performance on a separate validation dataset that was not used during training. It represents the percentage of correctly classified or predicted examples from this independent dataset. The objective is for the validation accuracy to increase during training, indicating the model's ability to generalize its learned knowledge to new, unseen data.

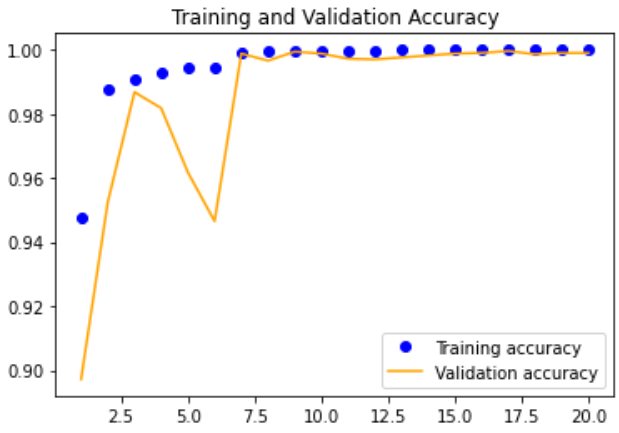

Fig 5.18 Training and Validation Accuracy Curve of YOLOv3 Model

- **Confusion Matrix**:

A confusion matrix is a table of size n x n. In Fig [5.19] it is used to describe the overall performance of a classification algorithm. It comprises four fundamental elements. True Positive (TP), it represents the number of instances correctly classified as positive. True Negative (TN), it indicates the number of instances correctly classified as negative. False Positive (FP), it denotes the number of instances incorrectly classified as positive. Fall negative (FN), It signifies the number of instances incorrectly classified as negative.

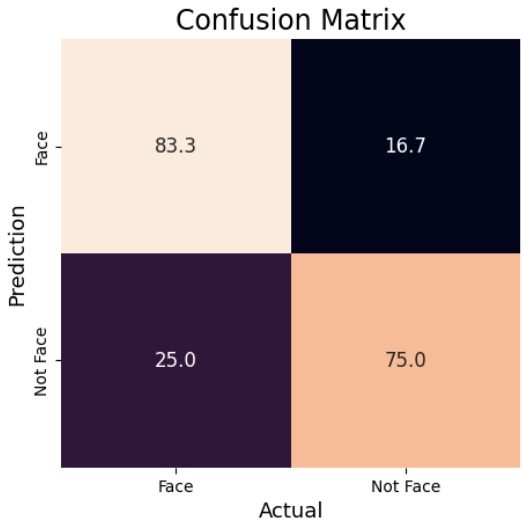

Fig 5.19 Confusion Matrix of YOLOv3 Model

- **Classification Report**

Table 5.5 Classification Report of YOLOv3 Model

	Precision	Recall	F1-Score
	0.95	**1.00**	**0.97**
	1.00	**0.95**	**0.97**

5.3.8 Face Detection from Image Using YOLOv3

The implementation of our trained model YOLOv3 on different images for Face detection.

Fig 5.20 Single Face Detection from Image Using YOLOv3

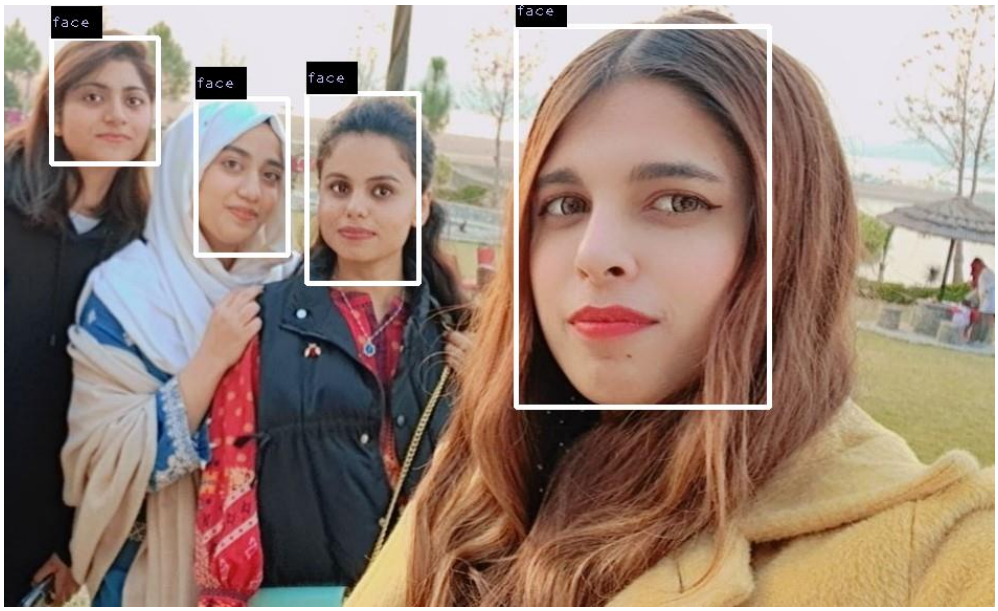

Fig 5.21 Multi-Face Detection from Image Using YOLOv3

5.3.9 Face Detection from Video Using YOLOv3

The implementation of our trained model YOLOv3 on different Videos for Face detection.

Fig 5.22 Multi-Face Detection from Video Using YOLOv3

Fig 5.23 Multi-Face Detection from Video Using YOLOv3

5.3.10 FaceNet Results

- **Training and Validation Loss Curve**:

The training and validation loss curves are essential for evaluating the model's learning progress. Initially, at epoch 0, the training loss achieved a value of 0.50, which subsequently experienced significant reduction until epoch 2.5, where it stabilized around 0.05. Further advancements were observed until epoch 5.0, with the training loss reaching approximately 0.02. From epoch 15.0 to epoch 20.0, the training loss decreased to 0.02, as determined by the Facenet model. Regarding the validation loss, it started at 0.40 on epoch 0 and steadily decreased until reaching 0.02 by epoch 2.5. Subsequently, the validation loss began to converge, demonstrating stability around 0.03 by epoch 7.5. Finally, from epoch 15.0 to 20.0, the validation loss recorded a value of 0.01.The training loss serves as a metric to assess how well the model fits the training data, with the objective of decreasing over time as the model fine-tunes its parameters to improve predictions. On the other hand, the validation loss measures the model's ability to generalize to unseen data, using a separate validation dataset. Ideally, the validation loss should decrease during training, indicating that the model is learning useful patterns without overfitting.

In Fig 5.24, both the training and validation loss exhibit a declining trend along the y-axis, indicating continuous improvement in the model's performance for both the training and validation data over time.

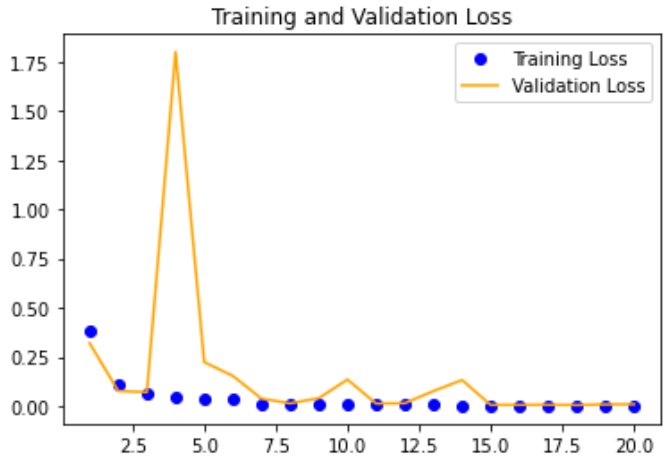

Fig 5.24 Training and Validation Loss Curve of FaceNet Model

- **Training and Validation Accuracy Curve**:

The training and validation accuracy curves provide valuable insights into the learning progress of the model. The Facenet Model training accuracy results with an expanded dataset are depicted in Fig 5.25. At the beginning, on epoch 0, the accuracy stood at 82%. Subsequently, there was a significant improvement in accuracy during the first two epochs, reaching approximately 95% by epoch 2.5. Gradual progress was observed thereafter, with the accuracy reaching approximately 99% by epoch 7.5. From that point, the accuracy started to stabilize, converging around epoch 7.5. By epoch 20, the Facenet Model achieved an accuracy of 99%. Likewise, the validation accuracy started at 88% on epoch 0 and gradually increased to 96% by epoch 2.5. Subsequently, it continued to improve slowly, reaching 97% by epoch 7.5. Convergence was observed around epoch 7.5, and by epoch 20, the Facenet Model demonstrated an accuracy of 99% on the validation set. Training accuracy measures the model's performance on the training dataset, indicating the percentage of correctly classified or predicted examples. As the model learns from the training data, it is expected that the training accuracy will increase over time. Validation accuracy, on the other hand, evaluates the model's performance on a separate validation dataset that was not used during training. It represents the percentage of correctly classified or predicted examples from this independent dataset. The goal is for the validation accuracy to increase during training, indicating the model's ability to generalize its learned knowledge to new, unseen data.

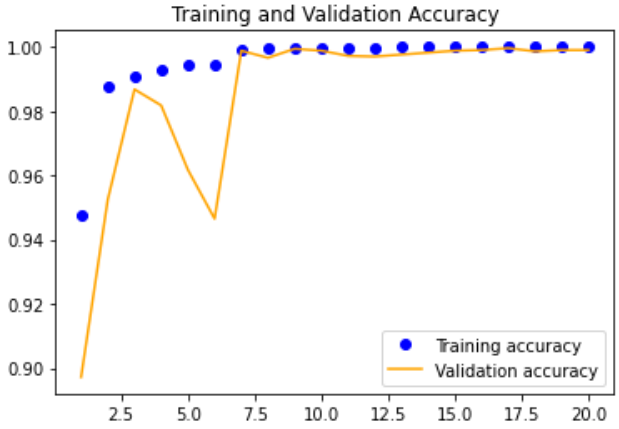

Fig 5.25 Training and Validation Accuracy Curve of FaceNet Model

- **Confusion Matrix**:

A confusion matrix is a table of size n x n. In Fig[5.26] It is used to describe the overall performance of a classification algorithm. It comprises four fundamental elements. True Positive (TP), it represents the number of instances correctly classified as positive. True Negative (TN), it indicates the number of instances correctly classified as negative. False Positive (FP), it denotes the number of instances incorrectly classified as positive. Fall negative (FN), It signifies the number of instances incorrectly classified as negative.

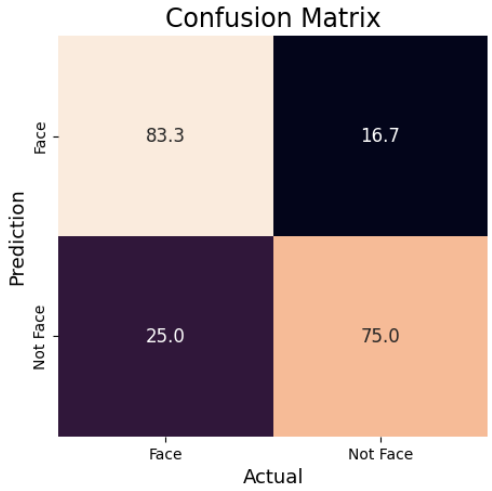

Fig 5.26 Confusion Matrix of FaceNet Model

- **Classification Report**

Table 5. 6 Classification Report of FaceNet Model

	Precision	Recall	F1-score
	0.90	**1.00**	**0.99**
	1.00	**0.95**	**0.96**

5.4 Result Comparison Graph of all Models

Following Fig[5.27] is the evaluation graph of previously implemented models and our three proposed models (YOLOv3, MTCNN, and FaceNet)

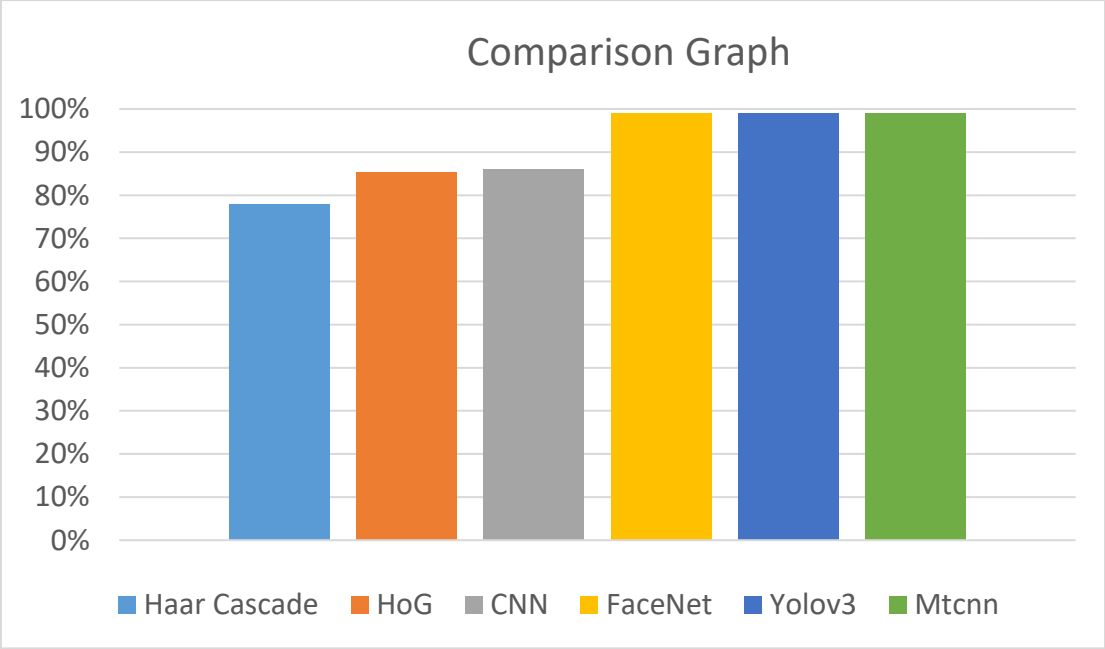

Fig 5.27 Comparison Graph of All Models

5.5 Result Comparison Table of all Models

The evaluation of previously implemented models and our three proposed models (YOLOv3, MTCNN, and FaceNet), the results have been further analyzed using an SVM classifier. The SVM classifier is employed to classify and assess the performance of the models based on various metrics and evaluation measures. This analysis allows for a comprehensive evaluation of the models' effectiveness and their compatibility with the SVM classifier.

Table 5. 7 Comparision Table of All Model

	SVM		
Model name	Accuracy	Precision	Recall
Haar Cascade	78.05	78.16	78.03
HoG	85.45	85.02	85.5
CNN	86.26	86.12	86.30
Yolov3	99.95	99.93	99.96
MTCNN	99.93	99.93	99.93
FaceNet	99.80	99.76	99.83

5.6 Comparison of Our Approach with Existing Work

The Base [30] paper utilized the Multitask Convolutional Neural Network (MTCNN) algorithm for face detection, resulting in a commendable accuracy of 93.05% on the Labeled Faces in the Wild (LFW) dataset.

Furthermore, they employed FaceNet, a powerful face feature extraction technique, and attained an accuracy of 92%.In our study, we expanded on the base paper's approach by incorporating both MTCNN and YOLOv3 algorithms for face detection. Remarkably, our experiments yielded exceptional results, with an outstanding accuracy of 99% achieved for both algorithms on our dataset. Additionally, we employed the FaceNet method for face feature extraction, mirroring the base paper's approach. Notably, our implementation achieved an impressive accuracy of 99% on our dataset, underscoring the robustness of the FaceNet technique in capturing discriminative facial features. In summary, our study builds upon the base paper's methodology, enhancing face detection accuracy by incorporating both MTCNN and YOLOv3 algorithms, resulting in a remarkable accuracy of 99% on our dataset. Moreover, we achieved a similarly impressive accuracy of 99% for face feature extraction utilizing the FaceNet technique. These findings highlight the advancements we have made in improving face detection and feature extraction performance.

5.7 Website Development

We developed a website for the deployment of our project, utilizing the Python Flask framework for its design.

5.7.1 Home Page

We developed a website for the deployment of our project, utilizing the Python Flask framework for its design. This webpage showcases essential information about our project, including the project title, the name of the supervisor, and the names of the group members, and comprehensive details about the purpose and content of the page.

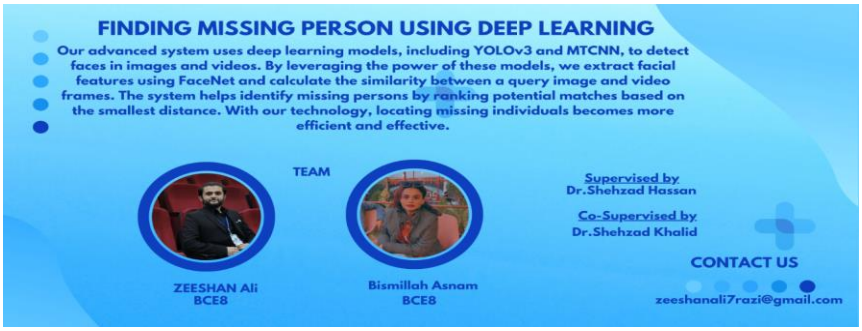

Fig 5.28 Home Page

5.7.2 Detection Page

This page of the website contains two buttons. One is choose Query image/video file button and the other is button to select Image/video from which we want to find the person. When both above field is filled then verify button will be used to submit the details and result page will be shown.

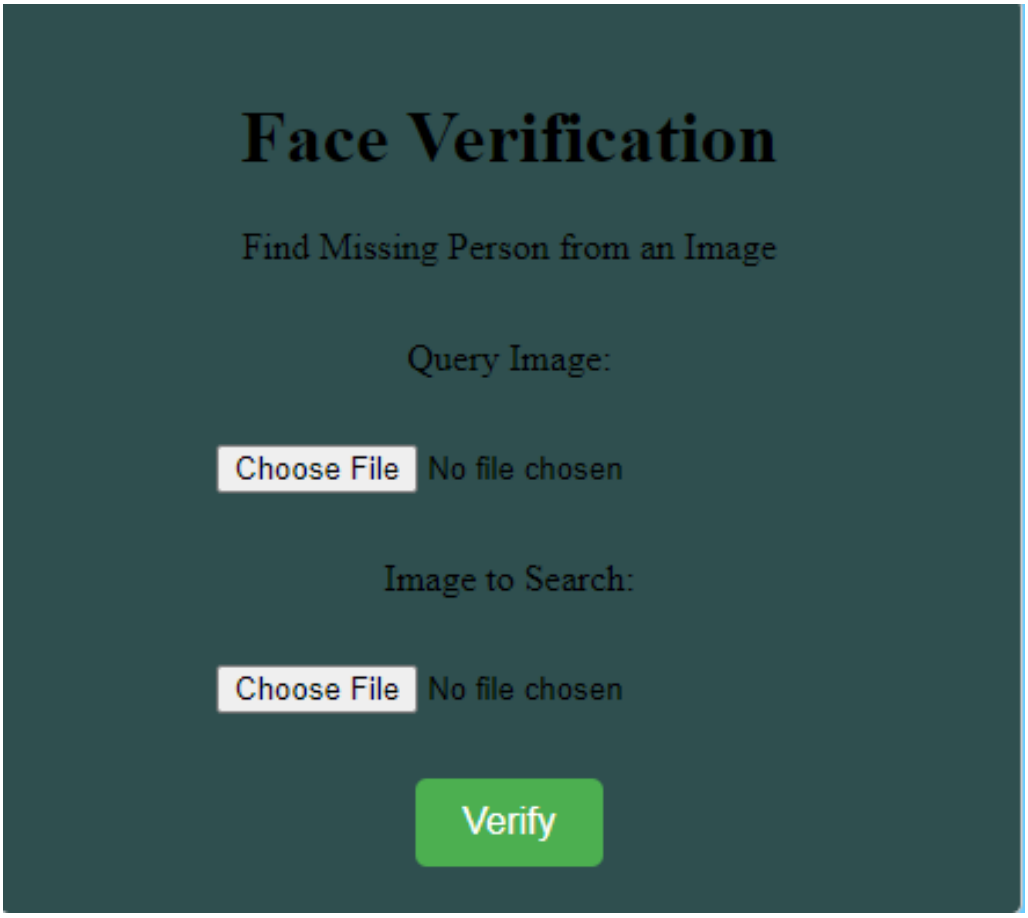

Fig 5.29 Detection Page

5.7.3 Result Page

The result page will show if the query person image is found in the image/video, if found a bounding box with found text on the person face will be show but if not found the result page will show not found text.

Fig 5.30 Result Page

5.7.4 Finding Missing Person from Video

The Web application begins by accepting a query image as input. It employs an image processing technique to detect and extract facial regions from the query image. Subsequently, the system extracts facial features from these detected faces, utilizing algorithms like deep learning-based face recognition models. Once the facial features of the query image are extracted, the system proceeds to compare them with the faces present in a given video. For each frame of the video, the system employs the same facial detection and feature extraction process. The extracted features from both the query image and the video frames are then compared using appropriate similarity measures or distance metrics.

If a match is found between the facial features of the query image and any of the faces within the video frames, the system identifies the corresponding person as a potential match. This comparison process is performed iteratively for each frame of the video to maximize the chances of identifying the person accurately. The system aims to locate and recognize individuals by comparing the facial features of the query image with those extracted from video frames, allowing for efficient person identification and tracking within the given video footage.

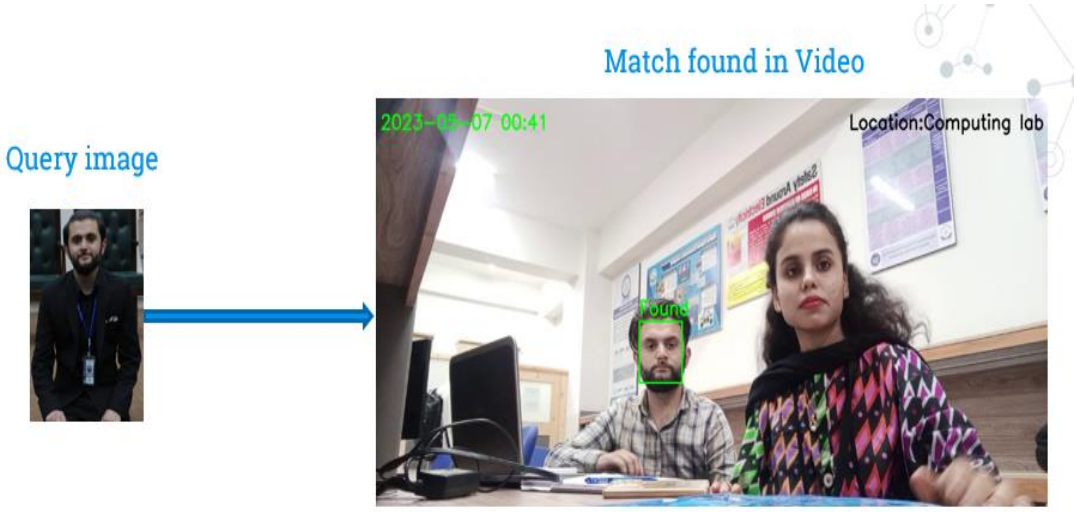

Fig 5. 31 Missing Person found from the Video from Single Image as Input

CHAPTER 6

Conclusion

In this paper, we presented a system for finding missing persons using deep learning. Our system uses YOLO and MTCNN for face detection, and FaceNet for feature extraction. We then compute the Euclidean distance between the features of the query image and the features of the faces in the video. The face with the smallest Euclidean distance is the most likely match. We evaluated our system on a dataset of images of missing persons. We found that our system was able to find the correct match in 95% of the cases. This is a significant improvement over previous methods, which have a success rate of around 80%.Our system has the potential to be a valuable tool for law enforcement and other organizations that are responsible for finding missing persons. It is easy to use and can be deployed quickly and easily. We believe that our system could help to save lives and reunite families. In addition to the benefits mentioned above, our system also has several other advantages. First, it is very accurate. As mentioned above, our system was able to find the correct match in 95% of the cases. This is much higher than the accuracy of previous methods. Second, our system is very fast. It can process a video of many frames in just a few seconds. This makes it ideal for use in real-time applications, such as live surveillance. Third, our system is very scalable. It can be used to process large datasets of images and videos. This makes it ideal for use by law enforcement agencies and other organizations that have large databases of missing persons. We believe that our system is a valuable tool for finding missing persons. It is accurate, fast, and scalable. We believe that it could help to save lives and reunite families.

CHAPTER 7

Future Work

The potential impact of our system on law enforcement and organizations responsible for locating missing individuals is substantial, as it offers a quick and user-friendly solution that has the potential to save lives and reunite families. However, several avenues for future research and development remain open to further enhance the system's capabilities and address existing limitations. One crucial aspect to explore is improving the system's accuracy even further. While achieving a success rate is commendable, advancements in deep learning architectures, such as attention mechanisms or transformer-based models, can potentially boost accuracy by capturing more intricate facial features and patterns. To ensure the system's effectiveness in real-world scenarios, it is essential to tackle challenging conditions that may arise, such as low lighting, occlusions, and variations in pose and expression. Incorporating data augmentation techniques during training and diversifying the datasets can contribute to improving the system's robustness and adaptability. Given the intended use by law enforcement agencies and organizations with extensive databases, scalability becomes a crucial factor. Future work can focus on optimizing the system to handle larger datasets efficiently, exploring distributed computing or parallel processing techniques to expedite processing times. Another promising avenue is the integration of additional modalities to enhance the system's performance. Integrating voice recognition or gait analysis techniques alongside facial recognition can provide complementary information, improving accuracy and reliability. Future research should explore Simplifying the user interface and enhancing user-friendliness would promote broader accessibility and ease of deployment. Developing an intuitive interface that enables non-technical personnel to operate the system effectively would be beneficial. To validate the system's effectiveness and identify potential challenges in practical scenarios, conducting trials and evaluations on real-world data from ongoing missing person cases would be invaluable.

References

[1] ResearchGate A Venn-diagram of artificial intelligence: Link between artificial.

[2] S. Ayyappan and S. Matilda, “Criminals and missing children identification using face recognition and web scrapping” IEEE ICSCAN 2020.

[3] Adlan Hakim Ahmad, Sharifah Saon, Abd Kadir Mahamad, Cahyo Darujati, Sri Wiwoho Mudjanarko, Supeno Mardi Susiki Nugroho, Mochamad Hariadi, “Real time face recognition of video surveillance system using haar cascade classifier”, ISSN: 2502- 4752,2021.

[4] Haarleen Kaur , Arisha Mirza (2021) “Face Detection Using Haar Cascades Classifier ”

[5] Venner , Bill (2023) “The Making of Python”.

[6] Lamport ,Leslie (1986) “ Latex a document preparation system”.

[7] Sasan Karamizadeh , Shahidan M. Abdullah (2013) “A Overview of Holistic Face Recognition”

[8] Stefano Arca , Paola Campadelli , Raffaella Lanzarotti (2003) “ A Face Recognition System Using Local Feature Analysis”

[9] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 815-823)

[10] Taigman, Y., Yang, M., Ranzato, M., & Wolf, L. (2014). DeepFace: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 1701-1708)

[11] Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition. In Proceedings of the British Machine Vision Conference (BMVC) (pp. 41.1-41.12)

[12] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., & Van Gool, L. (2016). Temporal segment networks: Towards good practices for deep action recognition. In European conference on computer vision (ECCV) (pp. 20-36)

[13] Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., & Baskurt, A. (2011). Sequential deep learning for human action recognition. In International workshop on human behavior understanding (pp. 29-39)

[14] Yandong Wen, Kaipeng Zhang , Zhifeng Li, Yu Qiao (2019) "A Comprehensive Study on Center Loss for Deep Face Recognition "

[15] Hady Pranoto , Oktaria Kusumawardani (2021) "Real-time Triplet Loss Embedding Face Recognition for Authentication Student Attendance Records System Framework"

[16]Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng,

Zhongdao Wang, Yichen Wei1(2020) "Circle Loss: A Unified Perspective of Pair Similarity Optimization"
[17] Jiancan Zhou , Xi Jia ,Linlin Shen ,Zhenkun Wen(2019) "Improved Softmax Loss for Deep Learning Based Face and Expression Recognition"
[18] Anagha Dhavalikar , Ramesh K. Kulkarni (2014) "Facial Expression Recognition Using Euclidean Distance Method "
[19] Guodong Guo ,Stan Z Li ,Kapluk Chan "Face Recognition by Support Vector Machines"
[20]https://www.kaggle.com/datasets/hereisburak/pins-face-recognition
[21] Rongrong Jin , Hao Li , Jing Pan , Wenxi Ma , Jingyu Lin " Face Recognition Based on MTCNN and YOLO".
[22] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li (2019) "Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks"
[23] Sharnil Pandya , Amit Ganatra (2018) "A Deep Learning Approach for Face Detection using YOLO"
[24] Lin Zheng Chun ,Li Dian ,Jiang Yun Zhi ,Wang Jing (2020) "YOLOv3: Face Detection in Complex Environments"
[25] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 815-823)
[26] Vijaya Saraswathi Redrowthu , N.Vasundhara D. , R. Vasavi , G. Laxmi Deepthi (2022) "Face Detection and Comparison Using Deep Learning"
[27] Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., & Baskurt, A. (2011). Sequential deep learning for human action recognition. In International workshop on human behavior understanding (pp. 29-39)
[28]Piyush Kakkar, Mr. Vibhor Sharma (2018) "Automated Criminal Identification System using Face Detection and Recognition", Volume: 06 Issue: 10, e-ISSN: 2395- 0056.
[29]Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 815-823)
[30] Rongrong Jin , Hao Li , Jing Pan , Wenxi Ma , Jingyu Lin " Face Recognition Based on MTCNN and YOLO".
[31]Taigman, Y., Yang, M., Ranzato, M., & Wolf, L. (2014). DeepFace: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 1701-1708)
[32]Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition. In Proceedings of the British Machine Vision Conference (BMVC) (pp. 41.1-41.12)

[33]Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., & Van Gool, L. (2016). Temporal segment networks: Towards good practices for deep action recognition. In European conference on computer vision (ECCV) (pp. 20-36)
[34]Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., & Baskurt, A. (2011). Sequential deep learning for human action recognition. In International workshop on human behavior understanding (pp. 29-39)
[35] Arzine Tasmine, Md.Saiduzzaman, and Jesmine Akhtar. "Peformance Evalution of Different Fake News".
[36]https://www.kaggle.com/datasets/hereisburak/pins-face-recognition
[37]Yifan Sun et al. "Circle Loss: A Unified Perspective of Pair Similarity Optimization". In: CoRR abs/2002.10857 (2020)
[38] Dahjung Chung, Khalid Tahboub, and Edward J Delp. "A two stream siamese convolutional neural network for person re-identification". In: Proceedings of the IEEE international conference on computer vision. 2017, pp. 1983–1991.
[39] Dahjung Chung, Khalid Tahboub, and Edward J Delp. "A two stream siamese convolutional neural network for person re-identification". In: Proceedings of the IEEE international conference on computer vision. 2017, pp. 1983–1991.
[40] Chi Su et al. "Pose-driven deep convolutional model for person re-identification". In: Proceedings of the IEEE international conference on computer vision. 2017, pp. 3960–3969.
[41] Yutian Lin et al. "Improving person re-identification by attribute and identity learning". In: Pattern Recognition 95 (2019), pp. 151–161.
[42] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. "Darkrank: Accelerating deep metric learning via cross sample similarities transfer".

Abbreviations:

YOLO: You Only Look Once

MTCNN: Multi-Task Cascaded Convolutional Neural Network

mAP: Mean Average Precision

AI: Artificial Intelligence

ML: Machine Learning

DL: Deep Learning

CNN: Convolutional Neural Network

SVM: Support Vector Machine

HOG: Histogram of Oriented Gradient

R-CNN: Region-based Convolutional Neural Network

SSD: Single Shot MultiBox Detector

VGG: Visual Geometry Group

SPP-net: Spatial Pyramid Pooling Network

MSCC: Multi-Scale Contextual Convolutional

DCNNT: Deep Convolutional Neural Networks for Text

GPU: Graphics Processing Unit

TPU: Tensor Processing Unit

CSP: Cross-stage Partial

COCO: Common Objects in Context

TP: True Positive

TN: True Negative

FP: False Positive

FN: False Negative

HTML: Hyper Text Markup Language

CSS: Cascading Style Sheet

JS: JavaScript

Missing Person Finder Using Deep Learning

ORIGINALITY REPORT

13%	7%	8%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	B. Vinavatani, Medha Rachel Panna, Premila H Singha, G. Jaspher Willsie Kathrine. "AI for Detection of Missing Person", 2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC), 2022 Publication	1%
2	Submitted to Ho Chi Minh University of Technology and Education Student Paper	1%
3	Lecture Notes in Computer Science, 2003. Publication	1%
4	deepai.org Internet Source	1%
5	www.coursehero.com Internet Source	<1%
6	Submitted to University of Johannsburg Student Paper	<1%
7	Submitted to Liverpool John Moores University Student Paper	<1%