FACES OF THE FUTURE: EXPLORING ADVANCED AUTOMATED FACE RECOGNITION IN THE DIGITAL ERA

By
Dr. SHAHINA ANWARUL
Assistant Professor (SG)
School of Computer Science, UPES, Dehradun

NSP Noble Science Press International Publishing

Published By:
Noble Science Press (International Publishing)
Aggarwal Plaza, LSC-1, Mayur Vihar Phase 3, Delhi- 110096
Email Id.: noblesciencepress@gmail.com , submission@noblesciencepress.org

Type setting by: Ms. Shama
NSP ID: NSP/BP/A-057-24
Printed By: KAAV® Media Pvt. Ltd., Delhi
Marketing By: KAAV® Publications, Delhi
Edition: 2024
Print Copy: INR 591/- US$30
E-ISBN- 978-81-982406-0-6
P-ISBN- 978-81-982406-1-3
DOI: https://doi.org/10.52458/9788198240606.nsp.2024.tb

PREFACE

An all-encompassing automated system for video surveillance, dedicated to recognizing faces, consists of various elements: face detection, face alignment, face recognition, and alert generation. In today's world, face recognition has become a powerful technology utilized in numerous applications, particularly in criminal identification. The ongoing manual examination of surveillance videos is an arduous process that demands significant visual focus but lacks mental engagement, making it prone to mistakes. Therefore, this book presents an automated facial recognition system as a solution to tackle these obstacles.

The work consisted of three distinct phases. Initially, the author conducted an evaluation of multiple existing face detection algorithms. After careful analysis, it is determined that the Single-Shot Multibox Detector (SSD) is the most optimal method due to its superior speed and accuracy compared to other alternatives. In the following phase, a new model for face recognition based on ensemble learning is introduced. Recognizing faces has proven challenging due to factors such as pose variations, changes in lighting, aging effects, partial occlusion, and low resolution. Contemporary approaches to face recognition have limitations when dealing with these unconstrained conditions. Therefore, improving face recognition requires incorporating diverse deep learning architectures. Despite advancements in traditional deep learning techniques for face recognition systems, there is still a need for a robust and efficient solution. To address this gap, the work given in this book used Hybrid Ensemble Convolutional Neural Network (HE-CNN) model. This model is established through ensemble transfer learning from modified pre-trained models and contributes to achieving higher accuracy in face recognition tasks.

The model undergoes a two-phase training approach, incorporating a differential learning rate based on a one-cycle policy. This method greatly improves the model's ability to recognize faces. It should be noted that these enhancements result in State-of-the-Art performance. To achieve this, the concatenation of Global Max

Pooling (GMP) and Global Average Pooling (GAP), Batch Normalization (BN), a Fully Connected (FC) layer, and dropout are integrated into the classification layers of pre-trained models. The incorporation of these suggested modifications and refining of the training process, the author observed outstanding results with a significant increase in recognition accuracy. The HE-CNN model has been evaluated using a self-curated criminal dataset to demonstrate its real-time applicability in practical scenarios. Through careful parameter selection and customization of layers, the designed model achieved remarkable accuracy of 95% on the self-curated dataset. Lastly, in the presented work given in this book, an automated alert system has been created that identifies crime-prone areas and helps prevent criminal activities. This is done through the analysis of data obtained from the identification of criminals. The system proactively alerts law enforcement personnel about high-risk areas so they can be prepared and vigilant before any crimes occur. Alerts are sent promptly when individuals with criminal records are detected in specified regions.

In a gist, the book, "Faces of the Future: Exploring Advanced Automated Face Recognition in the Digital Era", provided an efficient face recognition system based on the hybrid model. The hybrid model leverages the benefits of deep ensemble transfer learning techniques to construct a fast and highly accurate model.

The structure of the book is organized into six chapters, including the introduction chapter as Chapter 1. In Chapter 2, a comprehensive exploration of existing techniques for Face Recognition is undertaken, starting with traditional algorithms and progressing to advanced deep learning-based approaches, transfer learning-based methods, and ensemble learning-based techniques. The chapter also discusses the standard datasets that can be used to evaluate FR algorithms.

In Chapter 3, techniques to optimize deep learning models are explored. Similarly, Chapter 4 gives the idea about the available deep learning frameworks to implement face recognition algorithms.

Chapter 5 addresses the challenging areas in Face Recognition. In 6, dataset pre-processing techniques are introduced.

An algorithm for data oversampling is introduced in this chapter as part of an effort to ensure that a balanced dataset is used in evaluating the face recognition algorithm. In Chapter 7, metrics such as accuracy, precision, recall, F1-score, ROC curve are discussed to evaluate face recognition algorithms.

Chapter 8 introduces an automated method for FR system. Deep ensemble transfer learning is used in the proposed system to strike a balance between accuracy and computational resources. In the proposed and implemented system, face detection is handled by SSD, while Face Recognition is handled by a hybrid model. The suggested FR system also includes alert generation to reduce human intervention in recognizing individuals.

Finally, in Chapter 9, the book is summarized, conclusions are drawn, and future research directions are explored.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS

Acronym	Meaning of Abbreviation
FR	Face Recognition
S2S	Still-to-Still
S2V	Still-to-Video
V2V	Video-to-Video
ROIs	Regions of Interest
VS	Video Surveillance
PCA	Principal Component Analysis
SVD	Singular Value Decomposition
3D	Three Dimensional
GPUs	Graphical Processing Units
CCTV	Closed-Circuit Television
HE-CNN	Hybrid Ensemble Convolutional Neural Network
CNN	Convolutional Neural Network
GAP	Global Average Pooling
GMP	Global Max Pooling
FC	Fully Connected
BN	Batch Normalization
SOTA	State-of-the-Art
SSD	Single-Shot Multibox Detector
MTCNN	Multitask Cascaded Convolutional Neural Networks
LBP	Local Binary Pattern
LFW	Labelled Faces in the Wild
CPLFW	Cross-Pose LFW
GT	Georgia Tech
YTF	YouTube Faces
ROC	Receiver Operating Characteristic
CV	Computer Vision
AI	Artificial Intelligence
SIFT	Scale Invariant Feature Transform
LDA	Linear Discriminant Analysis

EBGM	Elastic Bunch Graph Matching
FERET	Facial Recognition Technology
PUB-FIG	Public Figures

Acronym	**Meaning of Abbreviation**
EV-SIFT	Entropy-Based Volume SIFT
PaSC	Point and Shoot Face Recognition Challenge
PSCL	Point-to-Set Correlation Learning
ML	Machine Learning
ANNs	Artificial Neural Networks
ILSVRC	ImageNet Large Scale Visual Recognition Competition
NIN	Network in Network
MLP	Multilayer Perceptron
RU	ResNet Unit
DCNN	Deep CNN
RCNN	Recurrent CNN
FCNT	Fully Convolutional Network-based Tracker
DSN	Deeply-Supervised Nets
DeCAF	Deep Convolutional Activation Feature
ASPP	Atrous Spatial Pyramid Pooling
FRNN	Full Resolution Residual Networks
RPNs	Region Proposal Networks
DA	Domain Adaptation
SIP	Single Image of Person
CCM-CCN	Cross-Correlation Matching CNN
CFR-CNN	Canonical Face Representation CNN
GAN	Generative Adversarial Networks
CPUs	Central Processing Units
API	Application Programming Interface
RNN	Recurrent Neural Networks
KDD	Knowledge Discovery in Databases
SL	Super Learner
RF	Random Forest
TBE-CNN	Trunk-Branch Ensemble CNN
MDR-TL	Mean Distance Regularized Triplet Loss
NR	Not Reported
DPI	Dots Per Inch
JPEG	Joint Photographic Experts Group
CLAHE	Contrast Limited Adaptive Histogram Equalization
GPS	Global Positioning System

Acronym	Meaning of Abbreviation
SD	Standard Deviation
ReLU	Rectified Linear Unit
CONV	Convolutional
LRF	Learning Rate Finder
TPR	True Positive Rate
FNR	False Negative Rate
FPR	False Positive Rate
PC	Proposed Classifier
MDC	Minimum Distance Classifier
SGD	Stochastic Gradient Descent

1

CHAPTER

INTRODUCTION

KEY HIGHLIGHTS

- *Motivation*
- *Classification of Biometric Characteristics*
- *The Block Diagram of an Automated FR System*
- *The Classification of the Factors Affecting FR Accuracy*
- *Different Scenarios for the FR System*

Biometric systems aim to authenticate individuals by utilizing one or multiple distinctive biometric characteristics, such as facial features, iris patterns, fingerprints, and other similar traits. The biometric traits can be classified into behavioral and physiological traits, as shown in Figure 1.1. Traditional authentication methods, such as identification cards and passwords, are often lost or stolen, while biometric-based systems improve security over traditional methods. Broadly, biometric applications can be categorized into three primary categories: verification, identification, and screening. Verification involves comparing an individual's biometric data with the stored data to validate their identity (referred to as one-to-one matching). The second category entails comparing an individual's biometric traits with the traits of various individuals stored in the system (known as one-to-many matching). In the last category, a small number of target persons are matched with unknown persons from a large group of people (*i.e.*, many-to-some matching).

There is a growing need for biometric security solutions to protect against fraud, theft, and other risks. Face Recognition (FR) holds a crucial position in biometrics-based security techniques and has proven to be a valuable tool across a diverse range of applications, including disease diagnosis, forensic analysis, secure transactions, age estimation, missing person searches, e-passport identification, mask recognition, and more. Face Recognition has drawn the most attention from researchers among the many biometric applications in recent years since it is more covert, non-intrusive, and requires less human involvement than other biometrics like the iris, fingerprint, or palmprint. The FR process involves analyzing and comparing essential facial features and expressions, aiming to enhance the intelligence and

safety of our world. This technology finds applications in authentication and surveillance, allowing for the identification of individuals and, when needed, the detection of suspicious behavior or suspects. In surveillance applications, Face Recognition is a crucial component for person identification. The automated FR system encompasses fundamental steps such as face detection, face alignment, face recognition, and alert generation, as depicted in Figure 1.2.

- **Face Detection:** *Identify the faces in an image or video using landmarks on the face such as eyes, nose, mouth, etc.*
- **Face Alignment:** *Alignment and normalization of faces for better recognition accuracy.*
- **Face Recognition:** *Recognize a specific person by comparing an image or video with the stored dataset.*
- **Alert Generation:** *Send an alarm message to the concerned person to reduce human intervention to detect people.*

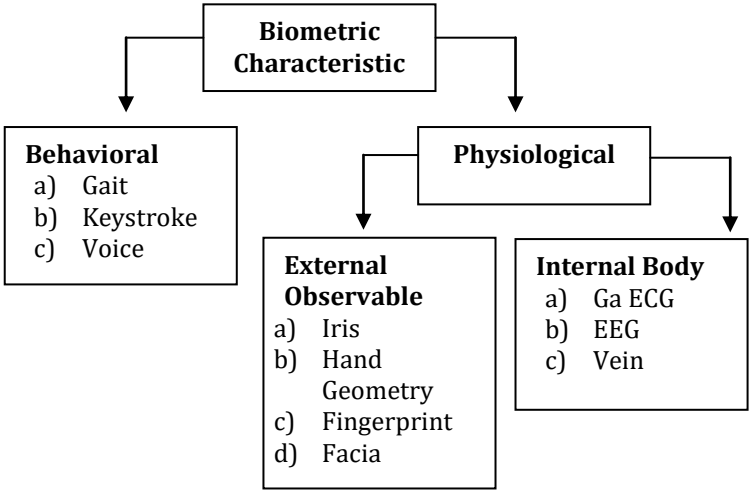

Figure 1.1 Classification of Biometric Characteristics

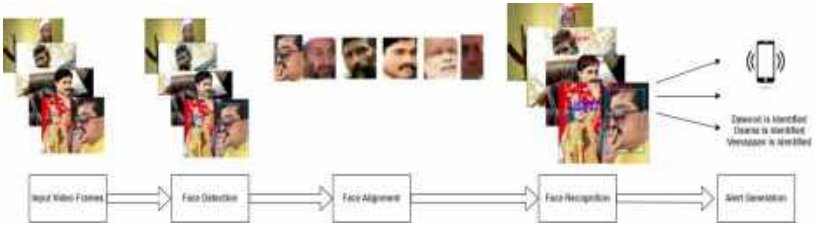

Figure 1.2 The Block Diagram of an Automated FR System

The task of FR in photos and videos is certainly difficult, and reaching 100% accuracy is a constant endeavor due to different factors influencing FR system performance. Despite intensive efforts, sufficient results have yet to be obtained, owing mostly to the numerous factors influencing the accuracy of these systems. Numerous studies have found that occlusion, low resolution, noise, illumination, position change, face expression, aging, and plastic surgery have an impact on recognition accuracy. These components are divided into two categories: internal and extrinsic factors. Intrinsic factors are the physical qualities of the human face that affect recognition accuracy, such as aging, facial expression, and plastic surgery. Extrinsic factors, on the other hand, alter the facial appearance and include occlusion, low resolution, noise, lighting, and position change, as shown in Figure 1.3. Depending on the nature of the training and test data, there are three basic scenarios for creating and evaluating FR systems. These are Still-to-Still (S2S), Still-to-Video (S2V), and Video-to-Video (V2V) FR scenarios, as depicted in Figure 1.4. In the S2S scenario, the FR system utilizes Regions of Interest (ROIs) extracted from still images of specific subjects as reference data to build a face model during the registration phase. Subsequently, the system performs real-time recognition using other still images as operational data. In the S2V scenario, ROIs from reference still images are used to build face models,

but the system operates on video streams for detection purposes. Lastly, the V2V scenario utilizes frames extracted from video streams as dual-purpose data, serving both as reference and operational inputs for Face Recognition. The S2V FR encounters challenges due to environmental differences between the source (registration) and destination (surveillance) domains. The captured images used during registration were obtained under controlled conditions. In contrast, the images captured by surveillance cameras are subject to unconstrained factors, such as low resolution, occlusion, lighting variations, and more. Face Recognition systems tailored for Video Surveillance (VS) purposes strive to precisely detect and recognize individuals of interest across a distributed network of cameras.

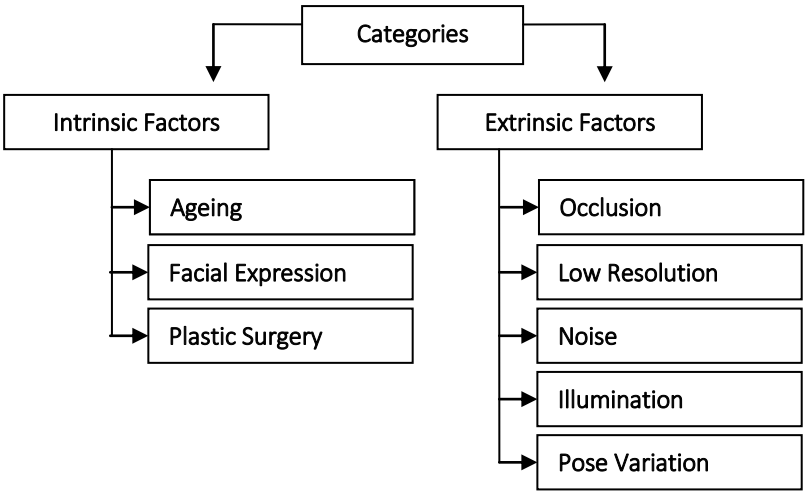

Figure 1.3 The Classification of the Factors Affecting FR Accuracy

Figure 1.4 Different Scenarios for the FR System

Extensive research has been dedicated to various face detection and recognition techniques. Traditional approaches primarily involve the use of Principal Component Analysis (PCA) for Face Recognition, achieving accuracy rates ranging from 69% to 95% in controlled environments. PCA has also been combined with other methods such as Singular Value Decomposition (SVD) and Fisherface techniques, resulting in recognition rates of 93.92% and 99.5% for frontal faces. Furthermore, researchers also investigated non-frontal FR techniques, such as mirroring, fitting, stretching, segmentation, and Three-Dimensional (3D) operations. However, the effectiveness of these methods tends to decline when facial images are captured in challenging environmental conditions, such as inadequate lighting, low-resolution cameras, and occluded facial images. With the recent advent of deep learning, the limitations of traditional methods have come to an end. However, the dependency on the enormous amount of data and systems with high computing power (*e.g.*, parallel processing systems accelerated with Graphical Processing Units (GPUs)) are still the challenges of deep learning techniques. The large amount of annotated facial datasets for the FR tasks is difficult to obtain due to the privacy concerns of the individuals. The recommended solution to address these challenges is deep ensemble transfer learning. It saves our time and resources. Transfer learning is a technique for using the feature representation from a pre-trained

model. Building and training a model from scratch is a tedious procedure. Instead of this lengthy process, transfer learning uses the weights from the pre-trained architectures to train the new model for the desired task. Ensemble learning is employed to enhance recognition accuracy by averaging the weights of multiple deep-learning models. It combines the benefits of deep learning and ensemble learning to achieve improved generalization performance in the final model.

1.1 MOTIVATION

Face detection and recognition play a crucial role in authentication systems based on biometric data, serving purposes in both authentication processes and surveillance. As scams and fraudulent activities continue to rise, facial recognition has become an essential system for ensuring security. Extensive research has been conducted globally to advance this field; however, despite continuous efforts, there is still a lack of robust and effective automated systems capable of performing well in both controlled and uncontrolled environments. Face Recognition has always been a highly intricate and demanding task, as it strives to replicate the human ability to perceive and identify faces. Nonetheless, human capabilities have limitations when dealing with various ambiguous phenomena. Hence, there is a need for an automated electronic system with high recognition accuracy and fast processing capabilities. The demand for biometric security systems has witnessed a substantial surge in recent times, driven by the need for enhanced protection and security against fraud, theft, and other related threats. Among the various biometric-based systems, Face Recognition has emerged as a prominent and effective solution. It serves various applications, including forensics, criminal identification,

surveillance, and fraud prevention, as it can authenticate an individual's identity and recognize individuals in different scenarios. Face Recognition system is used in banks, railway stations, airports, and other public places as a security control system where Closed-Circuit Television (CCTV) cameras are leased to identify individuals. It is also used in other sectors such as education, healthcare, media and entertainment, *etc.*, as illustrated in Figure 1.5. Video Surveillance recordings can be used to identify the suspect at the crime scene. Monitoring surveillance videos continuously is a very tiring task that requires visual attention and is also boring, leading to more opportunities for error. Automated surveillance that uses an intelligent system to monitor activities and raise an alarm, when necessary, can form an effective security system. In these real-time scenarios, there is a high possibility that the captured image has a large pose variation, faces are obscured by glasses, clothes, *etc.*, the lighting effect of the image might be dark, the facial expression might be different, *etc*. These are the factors that contribute to the deterioration in facial recognition accuracy. Therefore, an effective automated facial recognition system that offers high accuracy with minimal computational cost is the need of the hour.

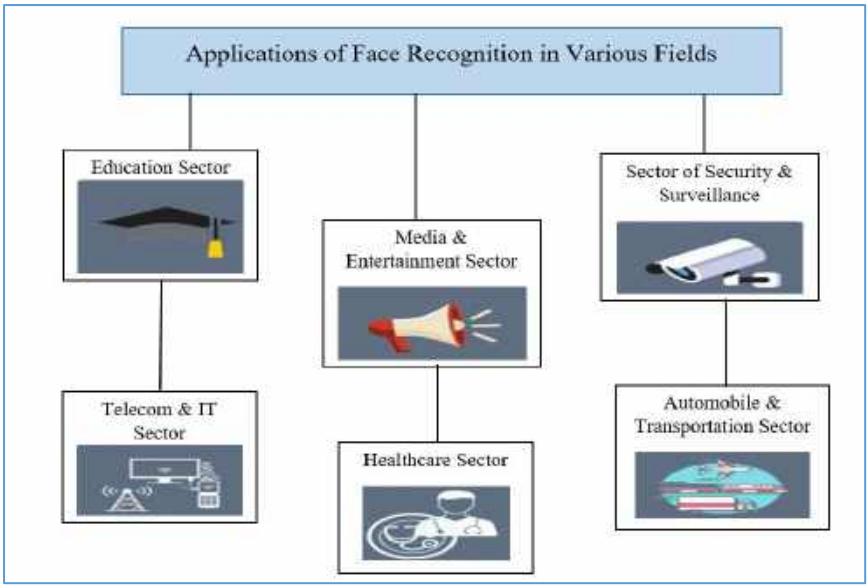

Figure 1.5 Applications of FR in Various Sectors

2

CHAPTER

EXISTING TECHNIQUES FOR FACE RECOGNITION

KEY HIGHLIGHTS

- *Traditional Algorithms for Face Recognition*
- *Deep Learning-Based Approaches for Face Recognition*
- *Transfer Learning or Domain Adaptation-Based Techniques for Face Recognition*
- *Ensemble Learning-Based Techniques for Face Recognition*
- *Available Datasets for Face Recognition*

Computer Vision (CV), a specialized domain within Artificial Intelligence (AI), empowers computers and systems to extract insights from digital images, videos, and other visual inputs. Subsequently, these insights are utilized to execute actions or formulate predictions. This interdisciplinary field bridges diverse areas of study, including computer science (theory, architecture, systems, algorithms), engineering (image processing, natural language processing, speech processing, robotics), biology (neuroscience), mathematics (machine learning, information retrieval), and physics (optics). The key elements of Computer Vision include visual recognition tasks such as image classification, object detection, localization, and segmentation. As the adoption of AI technologies reshapes numerous industries since the inception of machine learning, computer vision emerges as a pivotal player. Particularly in the realm of face recognition, Computer Vision provides algorithms and methodologies to scrutinize and process visual data embedded in images or videos containing human faces. This chapter delves into an intricate exploration of face recognition techniques, encompassing conventional, deep learning, transfer learning, and ensemble learning approaches.

2.1 TRADITIONAL ALGORITHMS FOR FACE RECOGNITION

Numerous investigations have been undertaken to explore diverse methodologies for face detection, identification, and matching. Traditional algorithms for face recognition include Scale Invariant Feature Transform (SIFT), Principal Component Analysis (PCA), AdaBoost, Linear Discriminant Analysis (LDA), Elastic Bunch Graph Matching (EBGM), Fisherface, and Singular Value Decomposition (SVD). However, these approaches are susceptible to limitations stemming from variations in

illumination, pose, and expression. Furthermore, their efficacy in recognizing faces is diminished in uncontrolled settings. These traditional face recognition algorithms have limitations when environmental conditions are not controlled, such as poor lighting, non-occluded images, and low-resolution cameras. The emergence of deep learning has overcome some of the constraints associated with traditional approaches.

2.2 DEEP LEARNING-BASED APPROACHES FOR FACE RECOGNITION

This section discusses deep learning-based algorithms developed for face recognition. Before that, the next subsections introduce deep learning and explore various architectures developed from early to advance deep CNNs.

2.2.1 Deep Learning

The concept of deep learning helps to provide higher recognition accuracy for the classification models in comparison to traditional approaches. Deep learning is a subfield of Artificial Intelligence and Machine Learning (ML) that includes statistical analysis techniques that train data recursively in order to provide predictions, as depicted in Figure 2.1. The distinguishing feature of deep learning models is their ability to learn and improve automatically through experience, enabling them to make predictions on unfamiliar data. Within the framework of Machine Learning, the identification of significant features that capture anomalies or patterns in the data holds utmost importance. These features were traditionally primarily created through human expertise. Nevertheless, models may now learn these features on their own though the advancement of machine learning techniques. Artificial Neural Networks (ANNs) serve as a widely embraced computational model in machine learning, aiming to

mimic the learning process of the human brain. Neural networks, also known as perceptrons, have been in existence since the 1940s but have gained prominence in the field of artificial intelligence over the past few decades. The development of a technique called backpropagation is a key factor propelling their prominence in the field of Machine Learning. Backpropagation facilitates the ability of neural networks to modify the weights within the hidden layer of neurons in accordance with the intended output.

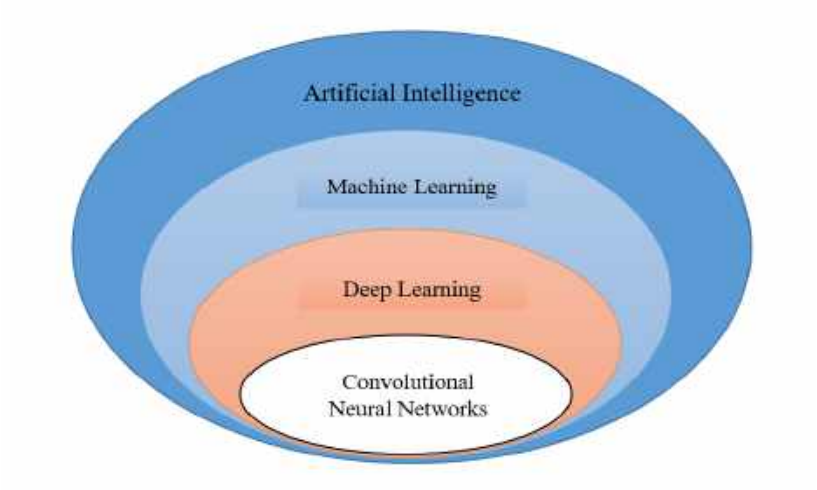

Figure 2.1 Classification of Artificial Intelligence

Deep learning represents the advancement of Artificial Neural Networks, characterized by the incorporation of multiple hidden layers that enable higher levels of abstraction. The introduction of deep layers into the model has significantly enhanced the accuracy of task predictions by enabling the system to learn complex data. A pivotal role in implementing deep learning-based approaches is played by CNNs, which consist of convolutional layers, subsampling layers, and fully connected layers. The feature learning process involves the convolutional and

subsampling layers, whereas the fully connected layer is used for classification, as illustrated in Figure 2.2. The emergence of CNNs has revolutionized feature learning techniques, as they have the ability to learn features automatically instead of relying on manual construction. CNNs have witnessed remarkable success in various computer vision tasks and are considered a significant breakthrough in machine learning. One notable model, AlexNet, brought about a paradigm shift in computer vision in 2012. AlexNet's architecture is similar to LeNet-5, however it was first created to compete in the ImageNet competition. Its triumph in the ImageNet competition effectively demonstrated its efficacy, leading to widespread adoption within the computer vision community. Effective regularization parameters, data propagation techniques, rectified linear units, and the use of Graphics Processing Units (GPUs) to meet computing demands were all credited with this success. One of the top ten deep learning achievements in 2013 was AlexNet. The greatest strength of a CNN lies in its deep architecture which enables the extraction of sophisticated features at various levels of abstraction.

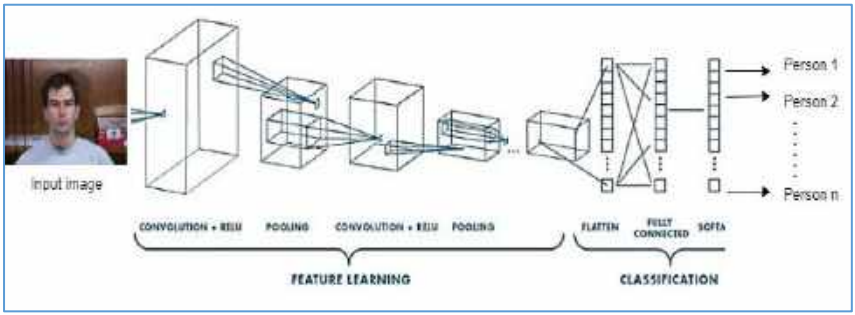

Figure 2.2 The Flow of the Working of CNN

2.2.1.1 Early Deep CNNs

The early deep CNNs first emerged in the late 1990s, starting around 1998. A CNN, also known as a ConvNet, stands as a distinctive and multi-tiered neural network deliberately designed for the task of pattern recognition. Its specialization lies in the capability to directly discern visual patterns from pixelated images, often requiring minimal to no preliminary data preprocessing. A sizable visual dataset created for use in image classification and object detection was made available by the ImageNet project. In order to promote the development and assessment of cutting-edge algorithms, this project also ran the ImageNet Large Scale Visual Recognition Competition (ILSVRC), an annual software competition. The revolutionary CNN architecture LeNet-5 is presented in this section, followed by discussions of the leading CNN architectures of the ILSVRC: AlexNet, Network in Network (NIN), VGGNet, GoogLeNet, ResNet, and DenseNet. In this book, the collection of specified CNN architectures is referred to as L-A-N-V-G-R-D.

a) **LeNet-5 (1998):** Comparing conventional architecture to traditional neural networks has resulted in a series of advancements in image classification. LeNet-5, the first CNN model released in 1998, had seven layers, only three of which were convolutional (C) and one of which was Fully Connected (FC), with a total of 60,000 parameters. In Figure 2.3, this network is displayed. The output of this network is a digit between 0 and 9, which is used to classify and identify 32 x 32-pixel grayscale handwritten numerals [1].

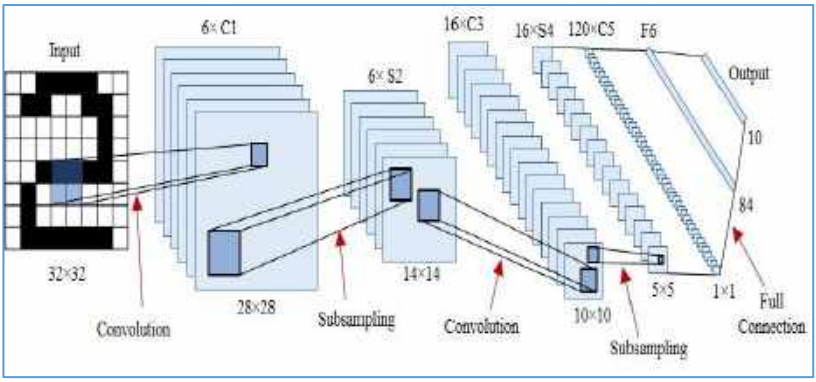

Figure 2.3 LeNet-5 Architecture consisting of 7 Layers [1]

b) **AlexNet (2012):** Higher-resolution images need to be processed using larger convolutional layers. Thus, AlexNet, which had 60 million characteristics in five convolution layers and three fully connected layers, is credited with starting the background of deep learning. Figure 2.4 depicts the AlexNet architecture. The reasonably quick and simple AlexNet is slightly changed into ZF-Net. This network performed substantially better than its predecessors. In a conventional classification network, AlexNet has been applied after downsizing the input image and applying convolutional and FC layers. The output would then be the expected class label for the input image [2].

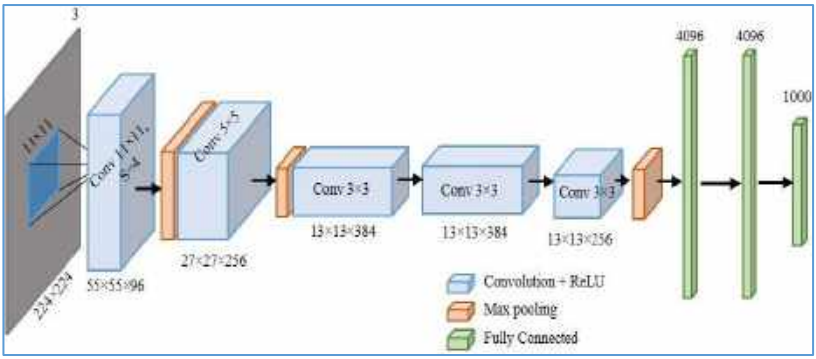

Figure 2.4 AlexNet Architecture [2]

c) NIN (2013): The capacity to distinguish between local patches within the input patch was improved by a Network in Network (NIN) design. Three micro neural networks, essentially nonlinear function approximators, are stacked to generate this model. The Multilayer Perceptron (MLP) is used to create the tiny neural networks. As shown in Figure 2.5, the filter size for each layer of the MLP structure is 1x1, except for the first layer. Like CNN, the micro-networks are slid over the input to produce the feature maps, which are then supplied into the following layer. Multiple MLP structures are stacked to provide deep NIN, while the classification layer uses Global Average Pooling [3].

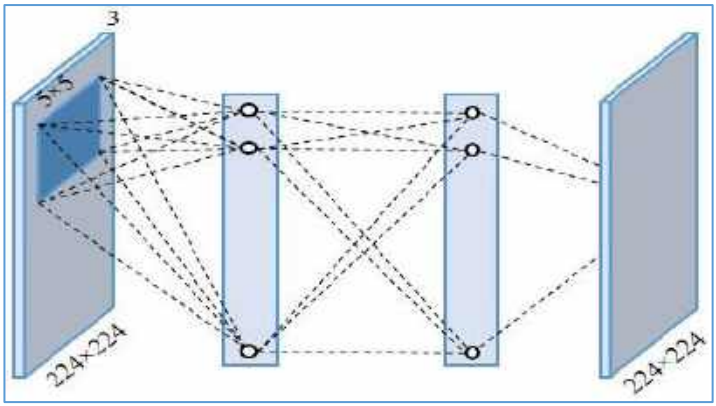

Figure 2.5 MLP Structure [3]

d) **VGGNet (2014):** This network's primary contribution is to assess correctness through deepening the network. This network, which had up to 19 layers and 138 million parameters, was made more accurate at classifying by using mini batch gradient descent with speed and dropout. Six VGGNet configurations have been proposed, ranging from 11 weight layers (eight convolution and three fully linked layers) to 19 weight layers (with 16 convolution and three fully connected layers). The count of filters (depth) in each layer accumulates to 512, originating from an initial count of 64 in the first layer and progressively doubling after each max-pooling layer. Figure 2.6 depicts the VGGNet-16 design. Due to its extremely homogeneous design, VGGNet placed first in the single-object localization test at ILSVRC 2014 [4].

e) **GoogLeNet (2015):** The first section of the GoogLeNet design is similar to LeNet in Figure 2.3 and AlexNet in Figure 2.4, as shown in Figure 2.7, while the block's stack is derived from VGGNet in Figure 2.6 [5]. LeNet, AlexNet,

and VGGNet's stack of FC layers are swapped out for GoogLeNet's worldwide mean pooling at the network's end. Google's top-5 error rate was 6.67%, which is quite near the level of human performance. It won first place in the ILSVRC 2014's classification and detection task. The subsequent adoption of Batch Normalization (BN) speeds up the training process for GoogLeNet. Figure 2.8 shows the GoogLeNet model with 22 layers [6].

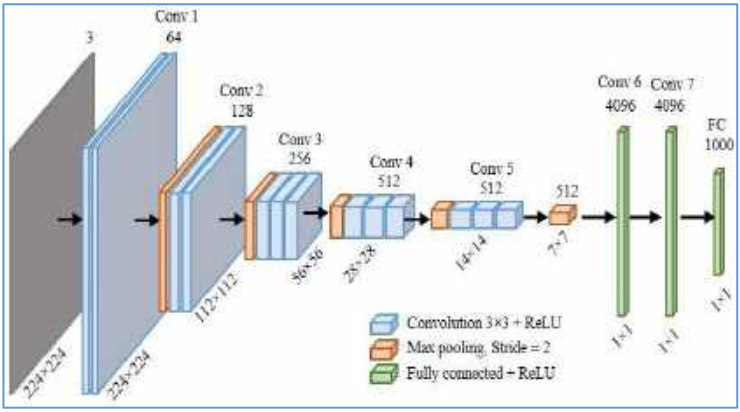

Figure 2.6 VGGNet-16 Architecture [4]

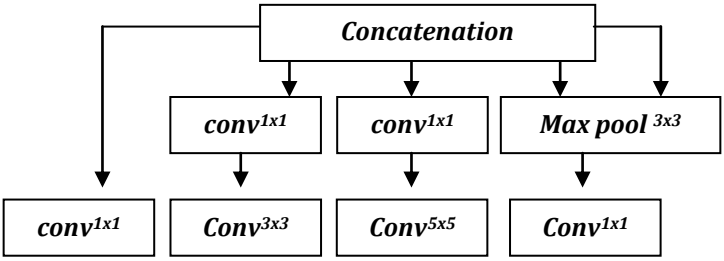

Figure 2.7 Inception Module Architecture [5]

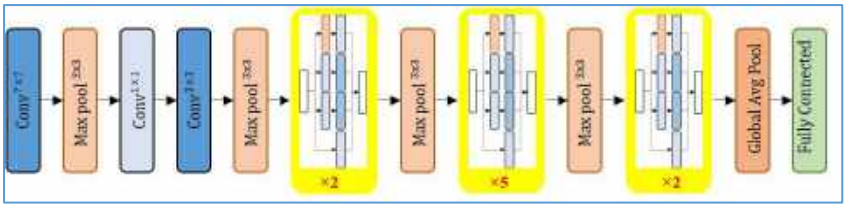

Figure 2.81 22-layer GoogLeNet Architecture [6]

f) **ResNet (2016):** Since it is more difficult to train deeper neural networks than shallower ones, the development of ResNet marked the start of a new phase in deep neural network training efficiency. In order to facilitate training and optimize the significantly deeper networks, which produced greater accuracy, a residual learning system was developed. Instead of learning unsourced functions, the layers were deliberately reformed to learn residual operations concerning the layer inputs. The introduction of the ResNet Unit (RU), shown in Figure 2.9, was made to address the critical issue. This occurs when adding more layers to a powerful deep model causes the training error to increase. By creating the shortcut interconnection as an identity mapping, ResNet solved this issue. The depth of the residual networks might range from 18, 34, 50, 101, or 152 layers. The most complex ResNet is less complex while being eight times larger than VGGNet. This network demonstrated easier optimization than VGGNet while achieving an increase in object accuracy rate of 28%. In Figure 2.10, the ResNet with a 34-layer residual is displayed. This network has four building blocks, and each has a stack of RU building blocks [7].

ResNet-34 consists of 18 RU building components in total [7]. Comparing the VGGNet to AlexNet, which has nearly three

times as few parameters, involves much processing. Compared to AlexNet, which has over 60 million parameters, GoogleLeNet's Inception architecture has about 7 million parameters, which is a 9-times reduction. The ability to transport gradients back across all levels in an efficient manner is a worry, though, considering the relatively enormous depth of Google Net's 22 layers. Because shorter networks did so well at this task, we can conclude that the features generated by the middle layers of the network should be very differentiable. This could be used by connecting additional classifiers to the intermediate levels. A deeper system would produce the same classification error as its shallower counterpart using ResNet's shortcut identity mappings. By employing this method, networks containing the Inception module can achieve comparable accuracy while being less expensive.

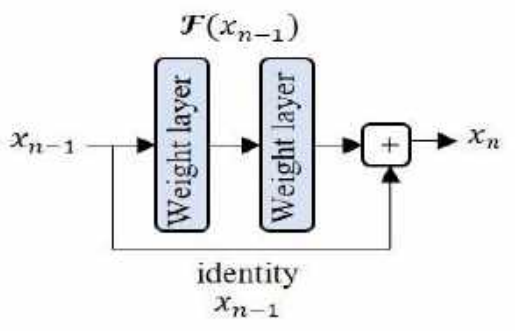

Figure 2.9 A ResNet Unit (RU) [7]

g) **DenseNet (2017):** The deep learning architecture named DenseNet, or "Densely Connected Convolutional Networks," was developed for image classification and other computer vision problems. "Densely Connected Convolutional Networks,", first discussed it in their 2017 publication. DenseNet introduces a special connectivity

design among layers to address the issue of disappearing gradients and information flow in deep neural networks. Each layer in a dense network is directly connected to every layer above it, facilitating information flow and gradients across the network. An architecture with excellent parameter efficiency is produced by this dense connectivity. DenseNet is broken up into a number of dense blocks. The primary innovation within each dense block is that every layer obtains feature maps from all the preceding layers within the same block. Each dense block is made up of several convolutional layers. This encourages feature reuse, enabling the network to learn more condensed and representative features, improving the performance of the network as a whole. The architecture shown in Figure 2.11 has a variety of advantages, such as enhanced gradient flow, feature reuse, a decrease in the number of parameters, and overfitting mitigation. DenseNet models are a popular option in the fields of deep learning and computer vision because of their State-of-the-Art performance on numerous benchmark datasets [8].

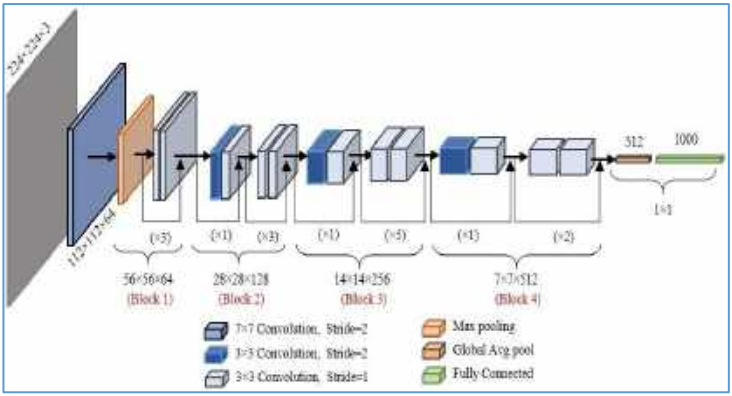

Figure 2.10 A 32-layer ResNet Architecture [7]

The detailed discussion of the pre-trained models is done to show the significance of their use in different applications. The next sub-section provides a detailed overview of the various applications of these pre-trained models.

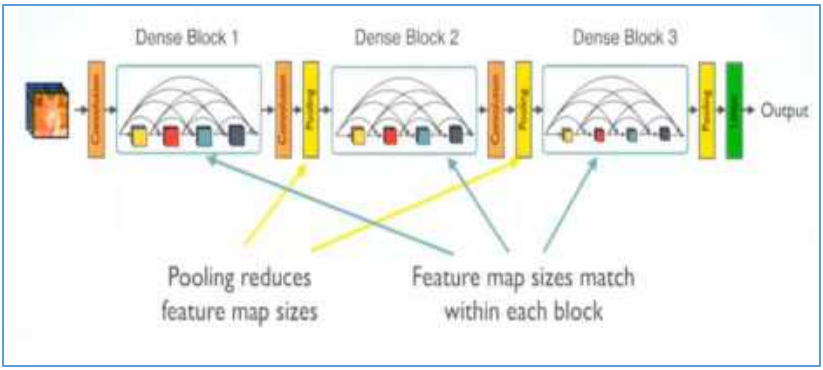

Figure 2.11 The Architecture of DenseNet [8]

2.2.1.2 Advanced Deep CNNs

More sophisticated Deep CNN (DCNN) architectures have adapted the basic L-A-N-V-G-R-D networks for various purposes. Following is a list of advanced DCNNs used in various tasks such as object detection, classification, and segmentation. An illustration of the discussed tasks is given in Figure 2.12.

a) Object Detection

In the field of Computer Vision, object detection refers to the task of simultaneously identifying and precisely locating objects within an image or video frame. Its core objective is not only finding the object but also identifying the correct position of the object, often visualized through bounding boxes encircling the detected objects. It is used in various computer vision tasks such as image and video analysis, surveillance infrastructure,

facial recognition, *etc.* Many researchers have proposed various deep learning techniques for object detection.

b) Classification

Classification is a fundamental task in both the fields of Machine Learning and data analysis. It involves the systematic arrangement of data points or objects into predefined groups or categories, primarily determined by the assessment of their inherent attributes or characteristics. Deeply-Supervised Nets (DSN) are suggested to give a close, integrated look at the hidden layers instead of only supervising the output nodes and sending this information back to earlier levels. Although introducing a residual learning framework with 152 levels made it easier to train deeper networks, the high computing cost of deeper neural networks still makes them difficult to deploy. At that point, the two main issues that need to be handled are the disappearing gradient and model size.

c) Pixel Classification

Pixel classification, also known as segmentation, is the process of assigning labels to each pixel of an image that helps to segment the image into regions. An object recognition and semantic segmentation network is proposed in the research, by fusing several low-level image data points with high-level context. This network uses bottom-up region recommendations in conjunction with CNNs to localize objects and segment them. Another deep neural network, dubbed DeepLab, which enhances the localization of object boundaries, also addresses semantic segmentation. This model incorporates two new elements: the Atrous Spatial Pyramid Pooling (ASPP) module to partition the objects at various scales, and the atrous convolution, a potent tool for controlling the resolution in dense prediction. The Full

Resolution Residual Networks (FRNN) model, another DCNN-based model for semantic segmentation, improves localization accuracy while offering remarkable recognition performance.

Most DCNNs have excessive parameters and need millions or even billions of starting point operations. Therefore, deep network designers' primary concerns are storage and computing capacity. One of the primary drivers for reducing the number of these networks' parameters is to increase the effectiveness of their deployment on mobile apps like MobileNet or their training in Internet-scale clusters, which results in lower computing costs and storage requirements. An overview of dimension reduction methods used with deep networks is provided in the next sub-section.

Figure 2.12 Applications of Advanced Deep CNNs

2.2.1.3 Existing State-of-the-Art Approaches for Face Recognition using Deep Learning

In recent times, Deep Convolutional Neural Networks (CNNs) have demonstrated remarkable achievements across a range of computer vision tasks, particularly in the realm of object detection. These deep CNN models have proven their ability to

effectively capture diverse variations present in large datasets and learn discriminative nonlinear feature representations. Consequently, they have emerged as powerful tools for face recognition (FR) applications by directly learning effective feature representations from face images. For instance, DeepID, DeepID2, and DeepID2+ were introduced to acquire a set of high-level discriminative feature representations. DeepID is trained by employing a collection of small CNNs and achieves an impressive recognition accuracy of 97.45%. Each CNN is individually fed with specific facial image regions such as the eyes, nose, and mouth, and the learned features are combined to form a powerful model. Expanding on this research, subsequent studies amplified the feature dimension of the last hidden layer and leveraged the hierarchical and non-linear characteristics of the convolutional layers. This approach facilitated the acquisition of hierarchical and nonlinear feature representations, intended to better differentiate between different individuals by extracting unique traits from each identity while minimizing variations within the same individual. In contrast to the DeepID series, Microsoft DeepFace integrates precise facial alignment to extract a resilient facial representation through a 9-layer deep CNN. Developed by Facebook, DeepFace achieved a remarkable recognition accuracy of 97.35% in face recognition, comparable to human-level performance. The model consists of over 120 million parameters and is trained on a dataset of 4.4 million images belonging to 4000 identities. Training such a model requires several days and highly computational systems. Likewise, in the context of Single Image of Person (SIP) challenges, recent studies adopted a loss function based on triplets to acquire robust facial embedding. The goal of this loss function is to distinguish between pairs that are positive and match the same facial regions

of interest (ROIs) and pairs that are negative and match distinct face ROIs. Autoencoder neural networks offer another avenue to extract deterministic nonlinear feature maps that are resilient to various factors affecting facial images, such as lighting, expression, and poses. The autoencoder architecture comprises encoder and decoder components, wherein the encoder converts input data into latent nodes, while the decoder reconstitutes these latent nodes back into the initial input data domain. The goal is to minimize the reconstruction error. In addition, a supervised deep architecture known as FlowNet tackles the estimation of optical flow by precisely predicting flows through the correlation of feature vectors derived from pairs of images located at different positions. In the context of Single Image of Person (SIP) scenarios, a deep supervised autoencoder is proposed that maps non-frontal faces with various complicating circumstances to the canonical face, a frontal face with neutral expression and normal lighting, of the same person in order to learn a robust face representation. However, due to their computational intricacies and the distinctions between static images and video frames, these methods may not be optimally suited for S2V FR tasks. To overcome these challenges and address the limitations of domain matching, researchers proposed a solution called the supervised autoencoder-based Canonical Face Representation CNN (CFR-CNN). This methodology forms the foundational framework for a S2V FR system that centers around domain alignment by the reconstruction of frontal faces from specific video Regions of Interest (ROIs). To facilitate the matching of input probes, a separate, fully connected network was trained as a classifier. The development of an accurate depth model necessitates the simultaneous consideration of both static images and videos during network training and optimization. Furthermore, to

address the variations inherent in the Single Image of Person (SIP) context, a supervised autoencoder was introduced. This autoencoder maps diverse facial variations to a canonical representation of a single individual's face. Advances in frontal view synthesis and pose-invariant representation acquisition through an adversarial process have led to the development of Generative Adversarial Networks (GANs). For example, a two-path GAN simultaneously manages the overall facial structure and the transformation of local intricacies. However, these methodologies require landmark detection and may not comprehensively accommodate variations such as blurring and scale changes (due to subjects' distance from surveillance cameras), thereby making them less suited for video-oriented face recognition applications. Other face recognition algorithms rooted in deep learning, like Deep Face Recognition, have demonstrated impressive achievements. For instance, they achieved a recognition accuracy of 98.95% on the Labeled Faces in the Wild (LFW) dataset and 97.3% on the YouTube Faces (YTF) dataset.

Training a DCNN from scratch requires a substantial amount of training data, making it challenging to achieve proper model convergence, especially in scenarios where privacy is a primary concern. To address data scarcity and overfitting issues, a technique called data augmentation is employed, entails generating new data by applying small adjustments to the current dataset, including flips, rotations, mirroring, translations, *etc.* Data augmentation helps mitigate the shortage of data and improves the generalization capability of the model. Retraining a CNN from another network with pre-trained settings is a promising additional strategy to address data scarcity. Modern image classification networks that have been trained on millions

of photos from a particular domain are called pre-trained CNNs. They can be used for many domains of interest after undergoing several weeks of training across several servers. This approach has proven to be highly valuable for researchers facing resource constraints, as it allows them to leverage the knowledge and features extracted by these large pre-trained models for their specific area of interest. Researchers can achieve optimal performance for their application based on the available data, providing a practical and effective solution by fine-tuning an existing model.

2.3 TRANSFER LEARNING OR DOMAIN ADAPTATION-BASED TECHNIQUES FOR FACE RECOGNITION

Transfer learning can be employed to enhance classification performance by transferring knowledge from a domain that has ample unlabeled data to a domain with limited labeled data. This approach is useful when there are differences in data distribution or feature spaces between the training and test datasets. In scenarios where collecting and labeling new data can be costly and time-consuming, transfer learning offers an attractive strategy compared to traditional Machine Learning approaches. Transfer learning involves utilizing a pre-trained model as a foundation for a new machine learning task, leveraging the knowledge gained during the initial training to improve learning and performance on a related task. This approach is particularly beneficial when the new task has concise annotations or the data distribution differs from the original task. By reusing the pre-trained model, transfer learning saves time and computational resources while enhancing accuracy and generalization. Essentially, the investigation of various domains, tasks, and distributions between the training and testing stages is

made possible by transfer learning. In transfer learning, the importance of the target task takes precedence over the source task since the model is fine-tuned for the target task. Transfer learning can be classified into two settings: (1) inductive and (2) transductive, as defined by equations (2.1) and (2.2). Inductive transfer learning uses distinct target and source tasks, with some labeled data available in the target domain. Conversely, transductive transfer learning addresses distinct source and destination domains while preserving the same task. In transductive transfer learning, labeled data from the source domain and unlabeled data from the target domain must be used to adjust the learning function, as is the case in Domain Adaptation (DA). The scarcity of datasets, especially in scenarios where privacy is a significant concern, has driven the application of transfer learning techniques by researchers. For the Single Image of Person (SIP) problem, a discriminative transfer learning approach has been proposed. In this approach, a generic training set (source domain) is used to learn a feature projection that is then transferred to a single-sample gallery set (target domain) through discriminant analysis. This approach aims to minimize the differences between the source and target domains and incorporates sparsity regularization to enhance robustness against outliers and noise.

Inductive transfer learning

$$if\ S_D \neq T_D\ or\ L_s \neq L_t, \tag{2.1}$$

It improves the learning of $f(.)$ in T_D by applying the knowledge in $S_D\ and\ L_s$, where $L_s \neq L_t$

Transductive transfer learning

$$if\ S_D \neq T_D\ or\ L_s = L_t, \tag{2.2}$$

It improves the learning of $f(.)$ in T_D by applying the knowledge in S_D *and* L_s, where $L_s = L_t$

Here, S_D is the source domain, and L_s is the learning task of the source domain, T_D represents the target domain, and L_t signifies the learning task within the target domain, the functions $f(.)$ serves as the predictive function.

2.4 ENSEMBLE LEARNING-BASED TECHNIQUES FOR FACE RECOGNITION

Ensemble techniques within the realm of machine learning involve the amalgamation of multiple models or classifiers to create an optimal composite model that delivers accurate predictions for the intended outcomes. The main idea behind ensemble models is to use the best parts of different learning algorithms at the same time. Compared to single models, this lets ensemble models make better predictions. The precision of a classifier is inherently linked to the quality of features extracted or learned from input data, such as images. Nevertheless, through the fusion of numerous classifiers and the amalgamation of their outcomes, further refinement of accuracy becomes feasible. Ensemble classification systems have garnered considerable attention across various domains, encompassing fields like face recognition, geospatial land classification, video-based face recognition systems, medical image segmentation, and wind power forecasting. These models exhibit heightened accuracy by effectively mitigating overfitting concerns and curtailing bias and variance errors in comparison to individual classifiers. The value of ensemble models is underscored by their triumphant application in renowned machine learning competitions, exemplified by the likes of the Netflix challenge, the Knowledge Discovery in Databases (KDD) Cup 2009, and

Kaggle, where ensemble-based models clinched top-ranking accuracy scores.

The performance of the classifier is greatly increased by the introduction of multi-classifier-based systems, in which the output of separate base classifiers is combined. Pattern recognition tasks with sparse and uneven training data are especially well-suited for ensemble approaches. Ensemble techniques' main concept is to create a variety of classifiers from the original data and combine them to get predictions that are better than those of any one basic classifier. Numerous studies have demonstrated that ensemble methods bolster the resilience and accuracy of classification systems. Key considerations in ensemble-based systems include the accuracy and diversity of the classifiers within the ensemble. While accurate classifiers are desirable, it is also crucial for the classifiers to be distinct from each other. Selecting the best classifier from the ensemble should not solely rely on training data accuracy. It is essential to incorporate diversity among the classifiers in the ensemble to ensure effectiveness.

This can be achieved through various approaches, as outlined below:

- Using same classification algorithm with different instantiation or different hyper-parameter settings.
- Using different classification algorithms for ensemble system.
- Using different feature sets:
  - *Random selection*
  - *Feature selection*

- Using different training sets:
  - *Bagging*
  - Boosting
  - Stacking

a) Bagging or Bootstrap Aggregating: An ensemble technique widely acknowledged in the field involves using a non-hybrid classifier, applying the same classification algorithm with various instantiations or hyperparameter settings to create an ensemble model. Bagging, another name for bootstrap aggregation, is an early ensemble-based method that is simple to understand. It operates by training multiple models using subsets of randomly chosen datasets from the original training set with replacement. A majority decision among the individual classifiers determines the ensemble's prediction. Figure 2.13 illustrates the process flow of the bagging technique. There are several variations of this algorithm aimed at enhancing the model's performance.

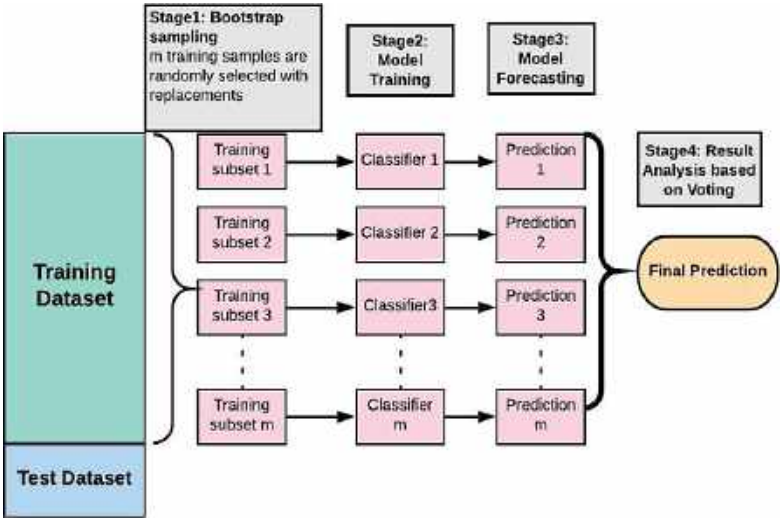

Figure 2.13 Parallel Execution of Bagging Process

b) Boosting: A variant of the bagging technique known as boosting is employed to enhance the classification model by sequentially transforming weak learners into strong learners, with each learner attempting to correct its predecessor. The fundamental distinction between bagging and boosting lies in their training approaches. In boosting, the architecture of the current model is dependent on the performance of earlier classifiers, but in bagging, each model is built independently during a parallel training period. Boosting is a sequential procedure in which the data is first given similar weights, and these weights are then redistributed following each training phase. This redistribution allows subsequent learners to place greater emphasis on misclassified cases, which are now assigned higher weights. Figure 2.14 illustrates this process.

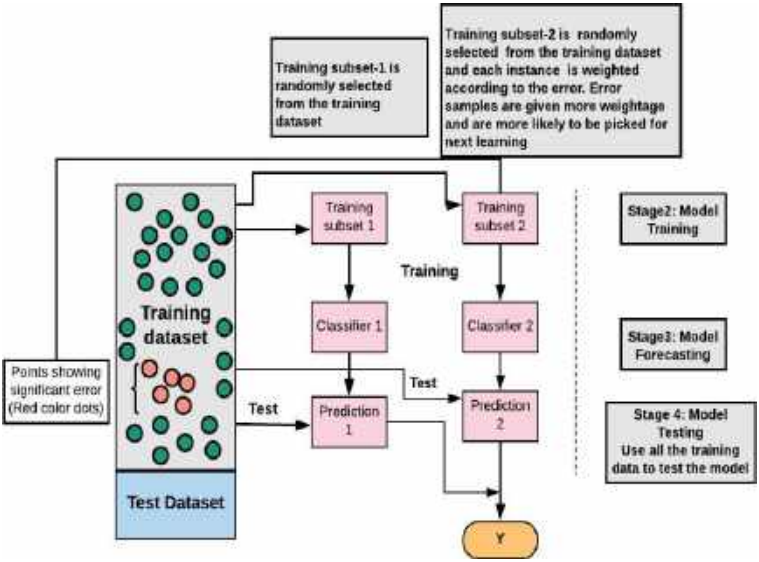

Figure 2.14 The Flow of the Execution of the Boosting Process

c) **Stacked Ensembles**: Stacking is a multi-layer learning technique in which base learners make up the first layer while lower-level meta-learners then use the base learners' outputs to figure out the optimal set of first-level models. The concept of the Super Learner (SL) was initially introduced in 1992, and its implementation with enhanced performance was demonstrated in 2007, highlighting the effectiveness of stacked ensembles in creating an optimal learning model. Random Forest (RF) is a well-known machine learning algorithm that uses the bagging technique. As Figure 2.15 illustrates, RF combines a collection of weak learners, such decision trees, to build a single powerful learner.

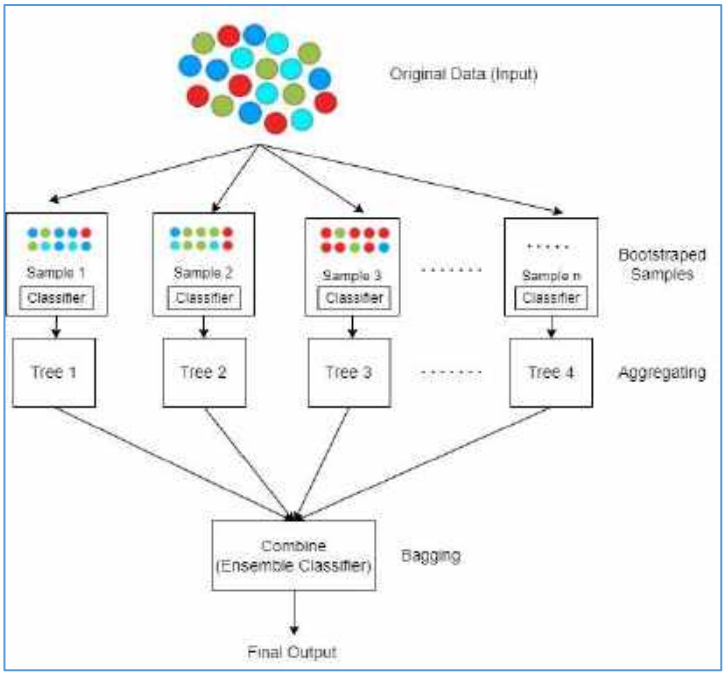

Figure 2.15 The Flow of the Execution of the Stacking Process

2.4.1 Existing State-of-the-Art Approaches for Face Recognition Using Ensemble Learning

A Convolutional Neural Networks (CNN)-based framework proposed by Ding *et al.* [9] addressed the challenges in video-based facial recognition. They introduced the Trunk-Branch Ensemble CNN (TBE-CNN) model to handle pose and occlusion variations. The TBE-CNN was trained using the Mean Distance Regularized Triplet Loss (MDR-TL) function. The proposed method was evaluated on multiple video datasets, including COX Face, PaSC, and YouTube Faces. Impressive recognition accuracies were achieved, such as approximately 95% on the YouTube Faces dataset, 96% on the PaSC dataset, and 99.33% accuracy for V2V, 98.96% for V2S, and 95.74% for S2V on

the COX dataset. Their approach secured first place in the BTAS 2016 Video Person Recognition Evaluation. The proposed approach effectively addressed challenges such as blur, partial occlusion, and pose variations. Tang *et al.* [10] proposed an ensemble model combining CNN and Local Binary Pattern (LBP) for face recognition. LBP was used to extract texture-related features from the face, and ten convolutional neural networks with five different network structures were employed to extract features and obtain classification results in the fully connected layer. The face recognition result was obtained using parallel ensemble learning with majority voting. However, specialized CNN models like TBE-CNN and HaarNet can enhance robustness to facial appearance variations at the expense of increased computational complexity. In these models, complicated and asymmetric face traits are captured by branch networks, and the root network captures the overall facial look (holistic representation). For example, TBE-CNN uses face landmarks, and HaarNet uses three branching networks based on Haar-like features. However, these complex CNN models may not be suitable for real-time face recognition applications [11]. Therefore, there is a need for a simple ensemble model that can provide high accuracy with fewer computations.

2.5 AVAILABLE DATASETS FOR FACE RECOGNITION

In the course of the last three decades, numerous face datasets have been created, reflecting a clear trend towards larger scales, diverse sources, and real-world unconstrained conditions. As simpler datasets such as LFW reached performance saturation, the development of increasingly complex datasets became essential to facilitate further research in face recognition. It is fair to say that the evolution of face datasets played a

significant role in shaping the direction of face recognition research. In this section, Table 2.1 includes freely available training datasets, and Table 2.2, which lists the testing datasets specifically designed for deep face recognition tasks.

Table 2.1 Publicly Available Training Datasets for Face Recognition

S. No.	Datasets	Publication Year	No. of Images	No. of Classes	No. of Images per Class (Min/Average/ Max)
1.	MillionCelebs [12]	2020	18.8 million	6,36,000	29.5
2.	MS-Celeb-1M (Challenge 3) [13]	2018	4 million	80,000	NR
3.	IMDB-Face [14]	2018	1.7 million	59,000	28.8
4.	VGGFace 2 [15]	2017	3.31 million	9,131	87/362.6/843
5.	UMDFaces-Videos [16]	2017	22,075	3,107	NR
6.	MS-Celeb-1M (Challenge 1) [17]	2016	10 million	100,000	100
7.	MS-Celeb-1M (Challenge 2) [17]	2016	1.5 million	20,000	1/NR/100
8.	MegaFace	2016	4.7	6,72,057	3/7/2469

	[18]		million		
9.	VGGFace [19]	2015	2.6 million	2,622	1,000
10.	CASIA WebFace [20]	2014	4,94,414	10,575	2/46.8/804

(*NR is Not Reported).

Table 2.2 Publicly Available Testing Datasets for Face Recognition

S. No.	Datasets	Publication Year	No. of Images	No. of Classes	No. of Images per Class (Min/Average/Max)
1.	IJB-C [21]	2018	3,13,000 images 11,779 videos	3531	42.1
2.	RFW [22]	2018	40,607	11,429	3.6
3.	IJB-B [23]	2017	11,754 images 7,011 videos	1.845	36.2
4.	CPLFW [24]	2017	11,652	3,968	2/2.9/3
5.	CALFW [25]	2017	12,174	4,025	2/3/4
6.	CFP [26]	2016	7,000	500	14
7.	UMDFaces [27]	2016	3,67,920	8,501	43.3

8.	IJB-A [28]	2015	25,809	500	11.4
9.	COX-S2V [29]	2015	NR	1,000	1 image, 4 video clips
10.	PaSC [30]	2013	2,802	265	NR
11.	YTF [31]	2011	3,425	1,595	48/181.3/6,070
12.	Choke point [32]	2011	64,204 images 54 videos	54	NR
13.	FG-NET [33]	2010	1,002	82	12.2
14.	YTC [34]	2008	1,910	47	NR
15.	LFW [35]	2007	13,000	5000	1/2.3/530

(*NR is Not Reported).

3

CHAPTER

TECHNIQUES TO OPTIMIZE DEEP LEARNING MODELS

KEY HIGHLIGHTS

- *Pros and Cons of Optimization Techniques*

To enhance the efficiency of deep learning-based algorithms, the subsequent approaches can be implemented to mitigate the model's training time.

a) Backpropagation: Utilizing backpropagation techniques is an effective way to compute the gradient function during each iteration. This approach within deep learning employs gradient-based methods to address optimization challenges.

b) Stochastic Gradient Descent (SGD): It efficiently locates the optimal minimum through the utilization of convex functions by disregarding local minima. The determination of the optimal minimum across diverse trajectories is influenced by parameters such as step size, learning rate, and activation function values. The mathematical equation for SGD to update the model's parameter is given in equation (3.1).

$$\theta_{i+1} = \theta_i - \eta \nabla J\,(\theta_i;\, x^{(i)}, y^{(i)}) \qquad (3.1)$$

Here, θ_i represents the model's parameters at iteration$i$, η is the learning rate, and $\nabla J\,(\theta_i;\, x^{(i)}, y^{(i)})$ is the gradient of the loss function J.

c) Learning Rate Decay: Modifying the learning rate leads to a reduction in the training time of gradient descent algorithms while concurrently enhancing the model's overall performance. This approach finds extensive application due to its capacity to effect substantial changes during the initial training stages, subsequently gradually diminishing the learning rate. Moreover, this technique enables fine-tuning of weights in subsequent iterations and is mathematically represented by equation (3.2).

$$k = i \times \frac{1}{1+d\times\frac{k}{step\ size}} \quad (3.2)$$

Here, k is the learning rate, i is the initial learning rate at the beginning of training the mode, d is the decay rate at which the learning rate decreases, and step size is the number of epochs before each decay.

d) Max-Pooling: Across non-overlapping segments of the input layer, a pre-configured filter is employed to extract maximum values and generate the resulting output. The application of the max-pooling technique also brings about a reduction in computational expenses associated with learning multiple parameters and is mathematically represented as given in equation (3.3).

$$P = O_{max}^{n,n}(F) \quad (3.3)$$

Here, F is the input feature map of size $n \times n$ obtained from the previous convolutional layer.

e) Dropout: Tailored for the challenge of neural network overfitting, the dropout technique employs a strategy of randomly omitting units and their connections throughout the training phase. For a single neuron in a neural network layer, the output ρ after dropout is applied can be calculated using equation (3.4). This technique serves as an improved regularization approach, effectively curbing overfitting within neural networks and ameliorating generalization error. In the realm of deep learning, this method garners superior results for supervised learning tasks.

$$\rho = \frac{1}{1-p}.x.d \quad (3.4)$$

Here, x is the output of the neuron before applying dropout, d is the binary dropout mask obtained from the Bernoulli distribution with probability p.

f) Batch Normalization: Batch normalization reduces covariate shift, which increases the learning rate of deep neural networks. During the training process, for each small batch, this method normalizes the input layer as the weights are adjusted. Enhanced network stability is achieved through the normalization of output from the final activation layer. Furthermore, batch normalization methodologies contribute to improved learning rates and a reduction in the required training epochs.

g) Transfer Learning: In transfer learning, a model initially trained for a particular task is adapted to undergo training for a comparable task. The knowledge acquired from addressing one challenge can be efficiently utilized to tackle another related issue. This process expedites advancements and enhances performance when addressing the second related task.

h) Ensemble Learning: In Machine Learning, ensemble techniques combine several models or classifiers to generate an ideal model that produces precise predictions for the intended result. Ensemble learning is employed to enhance recognition accuracy by averaging the weights of multiple deep learning models.

Based upon the aforementioned techniques for optimizing deep learning models, a comparison of their respective advantages and disadvantages is presented in Table 3.1.

Table 3.1 Pros and Cons of Optimization Techniques

S. No.	Technique	Description	Pros	Cons
1.	Back Propagation	Used in the optimization problems	Used to calculate the gradient	Susceptible to the effects of noisy data
2.	Stochastic Gradient Descent	Locate optimal minima in optimization problems	Prevents getting into local minima	Convergence time is large, demanding substantial computational resources
3.	Learning Rate Decay	Reduce the learning rate gradually	Enhancing the performance of the model helps reduce training time	Demanding significant computational resources
4.	Max-pooling	Downsampling technique for feature extraction	Reduces dimensionality and computational overhead	Considers only the maximum value of the region of an image, which may lead to an unacceptable result

5.	Dropout	Random deactivation of neurons during training	Prevents overfitting	Increases the training time required for the model to converge
6.	Batch Normalization	Nomalizing the activations of a layer within a mini-batch of data	Reduction in covariate shift, stable and faster convergence, and improved generalization	Increases implementation complexity and slows down the training of the model
7.	Transfer learning	Utilization of the knowledge of first model to resolve another problem	Handles data scarcity, improves performance of the model, and prevents overfitting	Limited flexibility (i.e., can work with similar types of problems)
8.	Ensemble learning	Prediction of each model is averaged to get the final prediction	Improves recognition accuracy	Computational overhead during training

4

CHAPTER

AVAILABLE DEEP LEARNING FRAMEWORKS TO IMPLEMENT FACE RECOGNITION ALGORITHMS

KEY HIGHLIGHTS

- *Comparison of Deep Learning Framework*

Deep learning frameworks facilitate the expedited design of neural networks by obviating the need for delving into the intricacies of underlying algorithms. Typically, each framework is tailored to address specific problem statements. The subsequent table, Table 4.1, provides a succinct overview of several deep learning frameworks.

a) **Fast.ai:** Built on PyTorch, the user-friendly, open-source fast.ai deep learning library places a strong emphasis on simplicity and effectiveness. It offers a high-level API that makes data augmentation, transfer learning, and model training simpler. Fastai promotes data comprehension and preprocessing with a focus on organized deep learning. For quicker convergence, it incorporates cutting-edge methods like learning rate annealing and progressive resizing. Through the use of visualization tools, the library facilitates model interpretation. It encourages community cooperation, provides courses, and has a wealth of documentation. Due to its user-friendly design, it is particularly beneficial for beginners exploring the deep learning discipline.

b) **PyTorch:** It serves the dual purpose of constructing deep neural networks and performing tensor computations. As a Python-based package, PyTorch offers tensor computation capabilities and a framework for generating computational graphs.

c) **Keras:** Built atop TensorFlow, the Keras Application Programming Interface (API) is coded in Python. This interface facilitates rapid experimentation and extends support to CNN as well as Recurrent Neural Networks (RNN). It provides the same deep learning model

deployment capabilities on CPUs and GPUs as TensorFlow.

d) TensorFlow: TensorFlow offers compatibility with a range of modern programming languages, including C++, Python, and R. This framework enables the seamless deployment of deep learning models on both Central Processing Units (CPUs) and Graphics Processing Units (GPUs), and was developed by Google Brain.

e) Deeplearning4j: Implemented in Java, Deeplearning4j exhibits superior efficiency compared to Python. Utilizing the ND4J tensor library, it empowers the manipulation of multi-dimensional arrays or tensors. This framework is compatible with both CPUs and GPUs. Deeplearning4j seamlessly handles diverse data formats, including images, CSV, and plaintext.

f) Caffe: Caffe, developed by Yangqing Jia, is an open-source framework. What distinguishes Caffe from other frameworks is its rapid processing speed and proficiency in learning features from images. Pre-trained models are made available through the Caffe Model Zoo framework, which makes the solution of various problems easy.

Table 4.1 Comparison of Deep Learning Framework

S. No	Deep Learning Framework	Release Year	Written in Language	CUDA Supported	Pre-trained Model
1.	Fast.ai	2018	Python	Y	Y
2.	Pytorch	2016	C, Python	Y	Y
3.	Keras	2015	Python	Y	Y
4.	TensorFlow	2015	C++, Python	Y	Y
5.	Deeplearning4j	2014	C++, JAVA	Y	Y
6.	Caffe	2013	C++	Y	Y

(*Y=Yes)

5 CHAPTER

CHALLENGING AREAS OF FACE RECOGNITION

KEY HIGHLIGHTS

- *Factors Affecting Facial Recognition Accuracy*

Face recognition from images and videos presents significant challenges, and extensive research has been conducted to achieve high precision. However, satisfactory results are yet to be attained due to various factors that affect the performance of these systems. These factors include occlusion, low resolution, noise, illumination, pose variation, expressions, aging, and plastic surgery. These can be classified into two main groups: intrinsic and extrinsic factors. Intrinsic factors are tied to the inherent attributes of the human face, including aging, facial expressions, and plastic surgery, directly influencing the system. Conversely, extrinsic factors entail alterations in facial appearance like occlusion, low resolution, noise, illumination, and pose variation, as depicted in Figure 5.1.

a) Occlusion: Partial occlusion emerges as a notable obstacle in the realm of face recognition endeavors. The concealment of specific facial features impedes the precise identification of individuals. For instance, eyeglasses or sunglasses can obscure the eyes; earrings or hair might veil the ears; scarves could shroud a substantial portion of the face; and facial hair like moustaches and beards might obscure significant facial attributes, as portrayed in Figure 5.1 (a). These factors have a detrimental effect on the performance of face recognition systems.

b) Low Resolution: Figure 5.1 (b) illustrates that pictures captured from surveillance video cameras often contain small faces, resulting in low resolution. Comparing a low-resolution query image with a high-resolution gallery image poses a significant challenge. The limited data in a low-resolution image leads to the loss of many important details, which can significantly degrade recognition accuracy.

c) **Noise:** Digital images are susceptible to different types of noise, which can result in poor accuracy in detection and recognition tasks. The introduction of noise into images can occur through various means, depending on how the image is created. Pre-processing plays a crucial role in the overall face detection and recognition system. Figure 5.1 (c) visually depicts the original image along with the presence of salt and pepper noise.

d) **Illumination:** The variations in illumination can have a significant negative impact on the performance of face recognition systems. Various factors, such as background light, shadow, brightness, and contrast, can contribute to these variations. Figure 5.1 (d) illustrates images captured under different lighting conditions.

e) **Pose Variation:** Pose variation poses a significant challenge for face recognition systems. Matching a profile face with a frontal face in the gallery requires frontal face reconstruction. This reconstruction is necessary because dataset images typically contain frontal views, and matching non-frontal profile faces can lead to inaccurate results. Researchers have proposed various approaches to convert non-frontal faces to frontal faces, which can improve recognition accuracy. Figure 5.1 (e) illustrates the different pose distributions of an individual.

f) **Expressions:** Facial expressions play a crucial role in expressing our emotions, as depicted in Figure 5.1 (f). They can alter the facial geometry, and even a slight variation can introduce ambiguity for face recognition systems. Muscle contractions that occur quickly lead to changes in

facial features like the mouth, cheeks, and eyebrows, which are all part of facial expressions.

g) Aging: Aging is a natural factor that significantly impacts face recognition systems, often posing challenges for algorithms. The face comprises various components, including skin tissues, facial muscles, and bones. When muscles contract, they cause distortions in facial features. However, aging brings about substantial changes in facial appearance, such as changes in facial texture (*e.g.*, wrinkles) and face shape over time. Figure 5.1 (g) illustrates the different textures of the faces of the same individual at various ages.

h) Plastic Surgery: Plastic surgery is another significant factor that can impact the accuracy of face recognition. Incidents have occurred where individuals have undergone plastic surgery due to accidents, resulting in their faces becoming unrecognizable to existing face recognition systems. This factor is particularly relevant in cases where criminals attempt to alter their identities through plastic surgery. Therefore, there is a need for an identification system capable of recognizing faces even after reconstructive surgery. The impact of plastic surgery on facial appearance is depicted in Figure 5.1 (h).

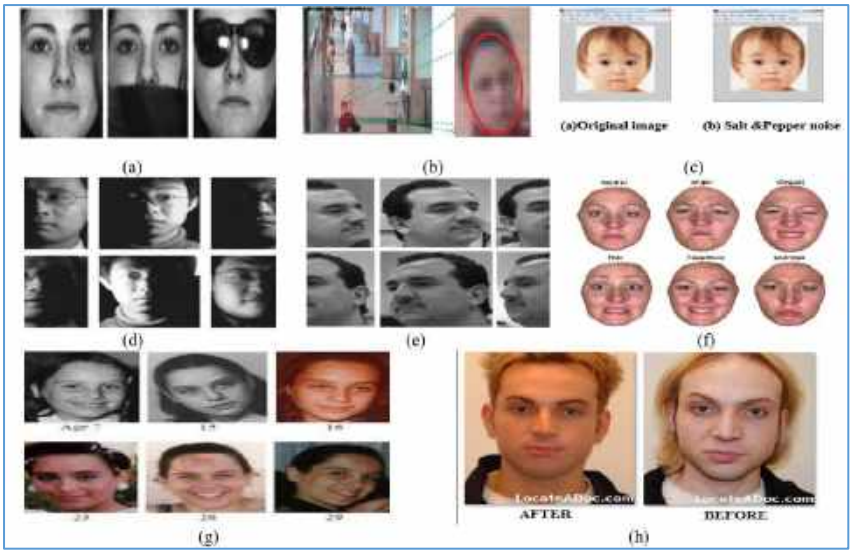

Figure 5.1 Factors Affecting Facial Recognition Accuracy

6 CHAPTER

DATASET PRE-PROCESSING TECHNIQUE

KEY HIGHLIGHTS

- *Data Oversampling*
- *Sample Output of Oversampled Images*
- *Algorithm for the Process of Data Oversampling*

This chapter provides insight into the meticulous preprocessing steps known as data oversampling that can be carried out to enhance the datasets for face recognition.

6.1 DATA OVERSAMPLING

Data augmentation, or data oversampling, serves as a strategy to amplify the dataset by generating virtual iterations of each image using a range of image transformation techniques. This practice is particularly valuable in the context of image classification tasks, as the augmentation process contributes to enhancing model performance by providing a broader and more varied dataset. Data augmentation methodologies are used to rectify the inherent imbalances present in the datasets. This approach encompasses a diverse set of transformations, including but not limited to mirroring, rotation, shearing, cropping, zooming, and alterations to color saturation.

Algorithm 6.1 specifies the complete process of oversampling the dataset in the present research. For every image, one augmentation technique (a_i) is randomly selected within configured thresholds, controlled by a parameter 't', to apply a random magnitude of the transformation. If the dataset contains p samples in each class and the value of p varies within the classes of the dataset, then the proposed algorithm generates n target samples in each class to make it a class-balanced dataset. It selects the random augmentation a_i and applies all the r transformations to the randomly selected image. The output image I_{out} is generated after applying the transformations and added to the class C of the dataset. Figure 6.1 demonstrates a set of randomly generated oversampled images. Through the implementation of oversampling techniques, the datasets are equalized, leading to a uniform distribution of images across all

classes. The quantity of images for oversampling is determined by selecting the maximum image count among all classes within the dataset.

Figure 6.1 Sample Output of Oversampled Images

Algorithm 6.1 Algorithm for the Process of Data Oversampling

Input: *Dataset D contains C different classes where each class consists of less than or equal to n unique samples, n is the maximum number of images in any class of the dataset, and A= [a1, a2, a3, a4, a5], where A is the set of all transformations applied on datasets.*

a1 = [centerCrop + ShiftScaleRotate + CLAHE]

a2 = [randomRotate90 + ShiftScaleRotate]

a3 = [flip + resize + randomBrightness]

a4 = [transpose]

a5 = [strongTransformation]

Output: *Dataset D contains C different classes where each class consists of n unique samples.*

1: **procedure** *Oversampling (D, n, A)*

2: n ⟵ target number of images per class
3: **for all** $C \epsilon D$ **do**
4: p ⟵ number of images in class C
5: **while** $p \leq n$ **do**
6: I ⟵ select one random image of C
7: a_i ⟵ select one random transformation function from A
8: r ⟵ transformations in a_i
9: I ⟵ I_{out}
10: **for all** $r \epsilon a_i$ **do**
11: I_{out} ⟵ $r(I_{out})$
12: **end for**
13: q ⟵ I_{out}
14: C ⟵ C U q
15: p ⟵ $p + 1$
16: **end while**
17: **end for**
18: **end procedure**

Here, a detailed discussion of the used data augmentation techniques and the application of these techniques in existing SOTA is given.

- **Center Crop:** The "CenterCrop" data augmentation approach isolates and extracts the central component of an image while excluding the surrounding areas. This method

frequently improves the dataset's diversity and makes it easier to train machine learning models by concentrating on the image's most noticeable elements. The size of the original image is decreased, resulting in a new image that captures the core content by executing a center crop. This strategy is especially helpful when the principal object of interest is in the center and the surrounding context is less important. Center cropping can also help reduce unwanted noise or pointless details in the dataset, improving the generalization and performance of the model. Object recognition, classification, and segmentation are a few examples of image processing jobs that frequently use the center crop data augmentation technique. Models are exposed to alterations in the main image content as a result of their application, which promotes resilience and adaptability in handling various scenarios and points of view.

- **Shift Scale Rotate:** The technique of "ShiftScaleRotate" serves as a data augmentation method that introduces random affine transformations, including shifting, scaling, and rotating, to augment the training dataset. This approach diversifies the perspectives from which an object is observed within the dataset, enriching its variety. By incorporating these transformations, the dataset's diversity is heightened, bolstering the resilience and adaptability of machine learning models. Importantly, this augmentation strategy achieves these improvements without necessitating the acquisition and annotation of additional data points.
- **Clahe:** Contrast Limited Adaptive Histogram Equalization (CLAHE) data augmentation is a technique that employs

adaptive histogram equalization in selected localized regions to enhance image contrast and detail. Contrary to traditional histogram equalization, which may make the noise worse, CLAHE focuses on smaller image portions, preventing over-enhancing while maintaining the integrity of the entire image. This augmentation technique finds frequent application in enriching image datasets, especially within the realm of computer vision tasks. By introducing fluctuations in contrast and texture, it effectively heightens model resilience and performance.

- **Random Rotate 90:** The data augmentation method known as "RandomRotate90" introduces random rotations in 90-degree steps to images. This method introduces variation and improves the model's performance and robustness by exposing the model to a variety of object orientations within the dataset. This augmentation method works exceptionally well for applications like object recognition, where there are a wide variety of object orientations. By including such rotations, the dataset becomes more inclusive and enables the model to generalize successfully across a variety of orientations without the need for additional labeled data.

- **Flip:** The "Flip" technique is a data augmentation approach that entails mirroring images horizontally or vertically. This strategy enriches dataset diversity by showcasing objects in altered orientations or viewpoints. Horizontal flipping entails a left-to-right reversal, whereas vertical flipping involves an up-to-down reversal. Frequently employed in image processing and computer vision applications, flipping serves to enhance model generalization and overall performance, particularly in

tasks where object orientation is of secondary importance. Through the integration of these mirror-image alterations, the dataset gains breadth, thereby enhancing the model's proficiency in detecting and comprehending objects across diverse vantage points.

- **Resize:** The technique of "Resizing" in data augmentation involves adjusting the dimensions of images while preserving their original aspect ratio. This method modifies image size within a dataset, either increasing or decreasing resolution and it is widely employed. The versatility of resizing enables images to be standardized to meet specific input size prerequisites for machine learning models or to introduce size variations for enhanced generalization. During resizing, images can undergo enlargement or reduction, impacting the level of detail and potentially accentuating distinct attributes. This method proves especially valuable when handling images of disparate dimensions within a dataset or when preparing data to align with precise model input specifications. Employing resizing as a data augmentation approach renders the dataset adaptable to the model's requirements, fostering heightened performance and accuracy during both training and testing stages.

- **Random Brightness:** The data augmentation approach known as "Random Brightness" encompasses the application of random changes to the brightness levels within images. Through this method, fluctuations in illumination are introduced, bolstering the dataset's resilience and capacity to capture distinct lighting situations. By incorporating random brightness adjustments, the augmentation procedure emulates real-life

settings characterized by shifting lighting conditions. Consequently, the model's capacity to generalize effectively and achieve robust performance across a spectrum of environments is heightened. Notably beneficial for tasks like object detection and recognition, where objects can manifest amidst diverse lighting scenarios, this technique proves to be an invaluable asset.

- **Transpose:** It serves as a data augmentation technique involving the exchange of rows and columns within an image matrix. The introduction of transpose augmentation injects diverse spatial layouts of objects and patterns, enriching the dataset's variety. Through the application of transpose, the dataset gains an assortment of alternate perspectives for the same image, permitting the model to glean insights from varying spatial correlations. This augmentation method is especially advantageous in endeavors like image classification and pattern recognition, wherein object orientations may differ. The process of transposition gives the model the ability to handle changes in how objects are aligned and how they are arranged in space. This allows the model to be more general and perform better.
- **Strong Trans for Mation:** In the context of data augmentation, a "Strong Transformation" refers to a more profound and impactful alteration applied to an image. This category of transformation frequently encompasses a fusion of various augmentation techniques, including rotations, flips, adjustments in brightness, contrast, and other modifications. The utilization of strong transformations aims to introduce substantial diversity into the dataset, compelling the model to navigate through a

spectrum of scenarios and amplifying its capacity to withstand shifts in real-world conditions. Such transformations prove especially advantageous during the training of models intended to navigate intricate and heterogeneous environments, exposing the model to an extensive array of potential inputs and circumstances.

These techniques can be used in different ways to create a large dataset that helps to improve the performance of the model.

7 CHAPTER

METRICS TO EVALUATE FACE RECOGNITION ALGORITHMS

KEY HIGHLIGHTS

- *Confusion Matrix for Face Detection*

The various evaluation parameters for the assessment of face detection algorithms are True Positive Rate (TPR), False Negative Rate (FNR), and False Positive Rate (FPR). TPR can also be called recall or sensitivity. It is the ability of the classification model to identify all the significant instances. FPR is the total count of false-negative assessments divided by the number of all negative evaluations. FNR shows the proportion of correct results that were missed and classified as incorrect. The formulae to estimate the values of TPR or recall, FPR, and FNR are given in equations (7.1) – (7.3), where TP refers to having both the actual and predicted label the same. For example, an image contains a face, and the algorithm also detects it as a face. FP is defined as the true label not being a face, but the predicted label being a face. FN has the true label as a face, but the predicted label does not have a face. The definitions of the mentioned measures are depicted in Figure 7.1.

$$TPR\ or\ Recall = \frac{TP}{TP+FN} \quad (7.1)$$

$$FPR = \frac{FP}{FP+TN} \quad (7.2)$$

$$FNR = \frac{FN}{FN+TP} \quad (7.3)$$

The classification accuracy is the evaluation parameter that can be used to calculate the performance of the face recognition algorithms, which is calculated using the formula given in equation (7.4).

$$Accuracy = \frac{Number\ of\ correctly\ recognized\ images}{Total\ number\ of\ images} \quad (7.4)$$

The other two evaluation measures, such as precision and recall, can be used to assess the classification model because accuracy alone is insufficient to choose the best classifier due to

the accuracy paradox. Precision defines the number of true positives out of the predicted positives. Recall and precision can be derived from the formulas given in equations (7.1) and (7.5). The Receiver Operating Characteristic (ROC) curve is also used to evaluate the performance of the classification models. The ROC curve gives an estimation of the rate of true positives relative to the rate of false positives for the classifier. In other words, it highlights the sensitivity of the classifier. In addition, the total number of inaccurate predictions on the test set divided by all of the test set predictions can be used to compute the error rate given in equation (7.6). We can always determine accuracy from the error rate since they are complementary quantities.

$$Precision = \frac{Truly\ Positives}{Predicted\ Positives} \tag{7.5}$$

$$Error\ Rate = \frac{Number\ of\ incorrect\ predictions}{Total\ number\ of\ images} \tag{7.6}$$

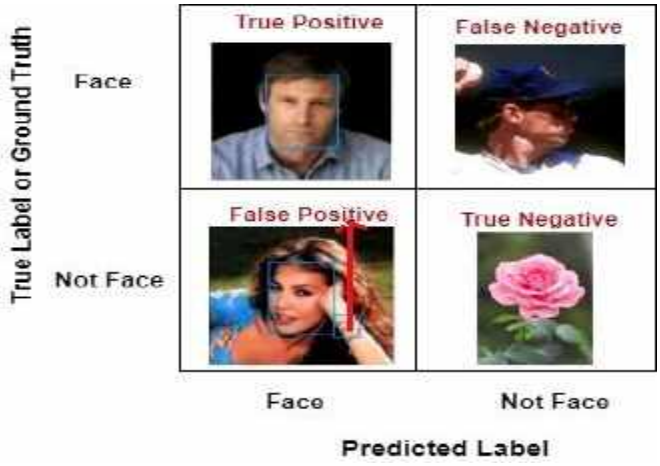

Figure 7.1 Confusion Matrix for Face Detection

8

CHAPTER

APPLICATION OF AN AUTOMATED FACE RECOGNITION SYSTEM IN CRIMINAL RECOGNITION

KEY HIGHLIGHTS

- *An Automated Face Recognition System*
- *The Modified Architecture of Baseline Models*
- *Hybrid Ensemble Cnn (He-Cnn) Model*
- *Various Modules of the Automated Face Recognition System*

In this chapter, a detailed overview of an automated face recognition system and HE-CNN model has been discussed. The face detection in an automated system is done using SSD as it is faster and accurate in comparison to other existing face detection algorithms.

8.1 AN AUTOMATED FACE RECOGNITION SYSTEM

This section proposes an automated system for criminal face identification and also helps police officials identify crime-prone areas. Figure 8.1 depicts the graphical representation of an automated recognition system that comprises face capture, face detection, face recognition, alert generation, and prediction of crime-prone regions. Algorithms 8.1, 8.2, 8.3, and 8.4 demonstrate the workings of an automated recognition system. The approach is divided into four modules: database creation, criminal recognition, alert generation, and prediction of crime-prone areas.

In the provided algorithms, the video is captured from the Global Positioning System (GPS)-enabled camera that attaches the location coordinates $L = \{lat, long\}$ with the video frames $D = \{F_i, L\}_{i=1}^{n}$, where F_i is the i^{th} frame, and L is the location coordinates of the camera. The GPS is used in the presented solution to track the current location of the static cameras that are deployed in different locations of the city. Then, the number of detected and aligned faces $\{\{A_1, L\}, \{A_2, L\}, \ldots, \{A_y, L\}\}$ from the captured video frames using SSD is stored in A along with the location coordinate L. The detected and aligned faces with the location coordinates are transferred to the recognition module for the recognition of criminals using the HE-CNN model. The recognized faces $\{\{R_1, L\}, \{R_2, L\}, \ldots, \{R_y, L\}$ and the location coordinates are sent to the alert generation phase.

The system finds out the distance between the police stations $P = \{\{lat_1, long_1\}, \{lat_2, long_2\}, \ldots, \{lat_z, long_z\}\}$ and the location of the criminal $L = \{lat, long\}$ using the Haversine formula. After calculating the distance, $Ð = min(H_j)\ _{j=1}^{z}$ stored the distance from the nearest police station. Then, the implemented system generated an alert via message $\{I_i, L\}_{i=1}^{y}$ and e-mail $\{R_i, I_i, L\}_{i=1}^{y}$ to the registered contact number and e-mail ID of the nearest police station, where I_i is the information (name, age, gender, crime, date of crime, identity mark) of the i^{th} criminal, L is the location of the criminal, and R_i is the image of the criminal. Parallelly, the location coordinates of the identified criminal L= $\{lat, long\}$ is stored in a separate file to form clusters of crime-prone regions. $L' = \{\{lat', long'\}, \{lat'', long''\}, \ldots., \{lat^{y'}, long^{y'}\}\}$ contains the location coordinates of identified criminals collected through the cameras installed in different locations. The clusters $C=\{c_i\}_{i=1}^{p}$ are formed using the K-means clustering technique. These clusters are then visualized on a Google Map, as the traffic flow in different areas is represented. This information serves as a valuable tool for police officials to identify regions that are prone to criminal activity. The input and output of the implemented system are given below. The results produced by Algorithm 4.1 serve as the input for Algorithm 4.2. Likewise, Algorithm 4.2's output is utilized as input for Algorithm 4.3. Algorithm 4.4, on the other hand, receives input in the form of location coordinates for all identified criminals, which are then used to create clusters in crime-prone regions.

Input: Video frames from the GPS-enabled camera $D = \{F_i, L\}_{i=1}^{n}$ and location coordinates of registered police stations $P = \{lat_i, long_i\}_{i=1}^{z}$

//where F_i is the i^{th} frame, n is the total number of frames, $L = \{lat, long\}$ is the location coordinates (*i.e.,* latitude and longitude) of the camera that is the same for all the frames, and z is the number of registered police stations.

Output: Message and e-mail containing image and information about the criminal $M = \{m_{i,}\ e_i\}_{i=1}^{y}$ and clusters of the crime-prone regions C=$\{c_i\}_{i=1}^{p}$

//where y is the number of criminals identified, and p is the number of clusters formed.

Algorithm 8.1 Face Detection

Input: *Video frames from the GPS-enabled camera* $D = \{F_i, L\}_{i=1}^{n}$ *and location coordinates of registered police stations* $P = \{lat_i, long_i\}_{i=1}^{z}$
//where F_i *is the* i^{th} *frame,* n *is the total number of frames,* $L = \{lat, long\}$ *is the location coordinates (i.e., latitude and longitude) of the camera that is the same for all the frames, and* z *is the number of registered police stations*
Output: *Detected faces with location coordinates {*A_i*, L} of* y *criminals*

1. **procedure** *FaceDetection (D)*
2. $D = \{\{F_1, L\}, \{F_2, L\}, \ldots., \{F_n, L\}\}$
3. **Repeat**
4. **for all** $\{F_i, L\} \in D$ **do**
5. SSD $\longleftarrow \{F_i\}$
6. $A \longleftarrow \{\{A_1, L\}, \{A_2, L\}, \ldots., \{A_y, L\}\}$
7. **end for**
8. **until** n **times**
9. **return** $\{A_i,\ L\}_{i=1}^{y}$
10. **end procedure**

Algorithm 8.2 Face Recognition

Input: *Detected faces and location coordinates of criminals from algorithm 4.1(i.e., {A_i, L}}*

Output: *Recognized faces with location coordinates {R_i, L} of y criminals*

1. **procedure** *FaceRecognition (A)*
2. $A = \{\{A_1, L\}, \{A_2, L\}, \ldots, \{A_y, L\}\}$
3. **Repeat**
4. **for all** $\{A_i, L\} \epsilon A$ **do**
5. HE-CNN $\longleftarrow$ $\{A_i\}$
6. $R \longleftarrow \{\{R_1, L\}, \{R_2, L\}, \ldots, \{R_y, L\}$
7. **end for**
8. **until** y **times**
9. **return** $\{R_i, L\}_{i=1}^{y}$
10. **end procedure**

Algorithm 8.3 Alert Generation

Input: *Recognized faces and location coordinates of criminals from algorithm 4.2 (i.e., {R_i, L}} and registered police stations records P*

Output: *Message and e-mail containing image and information about the criminal (i.e., $M = \{m_i, e_i\}_{i=1}^{y}$*

1. **procedure** *AlertGeneration (R, P)*
2. *// P is the police station record that contains the contact details, e-mail ID, and latitude and longitude of all registered police stations*

   $R = \{\{R_1, L\}, \{R_2, L\}, \ldots, \{R_y, L\}\}$

3. $L = \{lat, long\}$
4. $P = \{\{lat_1, long_1\}, \{lat_2, long_2\}, \ldots., \{lat_z, long_z\}\}$
5. **Repeat**
6. **for all** $\{R_i, L\} \epsilon R$ **do**
7. **Repeat**
8. **for all** $\{P_j\} \epsilon P$ ***do***
9. $H_j \longleftarrow \{L, P_j\}_{j=1}^{z}$
10. $Ð \longleftarrow Min(H_j)_{j=1}^{z}$
11. **end for**
12. **until** z **times**
13. $m_i \longleftarrow \{I_i, L\}_{i=1}^{y}$
14. $e_i \longleftarrow \{R_i, I_i, L\}_{i=1}^{y}$
15. **end for**
16. **until** y **times**
17. **return** $\{m_i, e_i\}_{i=1}^{y}$
18. **end procedure**

Algorithm 8.4 Clusters of Crime Prone Regions

Input: *Location coordinates of the recognized criminals*

Output: *Clusters of the crime-prone regions C*=$\{c_i\}_{i=1}^{p}$

1. **procedure** *Cluster (L)*
2. // L' stores the location coordinates (L) of all the identified criminals from the cameras installed in different locations

L'= []

3. **Repeat**

4. **for** $k \longleftarrow$ *1 to y* **do**

5. $(lat^{k'}, long^{k'})$= L

6. L'= L'+ $(lat^{k'}, long^{k'})$

7. **end for**

8. **until** y **times**

9. // Therefore, L' = $\{\{ lat', long' \}, \{ lat'', long'' \}, ..., \{lat^{y'}, long^{y'}\}\}$

$\{c_1, c_2, ..., c_p\} \longleftarrow$ *K-means*(L')

10. C=$\{c_i\}_{i=1}^{p}$

11. **return** C

12. **end procedure**

8.2 THE MODIFIED ARCHITECTURE OF BASELINE MODELS

In this section, various pre-trained models such as ResNet50, VGG19, and DenseNet169 have been used. These models were fine-tuned and combined through ensemble transfer learning to create an optimized hybrid model specifically designed for the task at hand. The modification consists of a baseline model and customized classification layer. For the base network, the pre-trained ResNet50, VGG19, and DenseNet169 models, using their initial weight parameters have been utilized. The base architecture of pre-trained models is originally trained on the ImageNet dataset and includes 1000 columns of distinct weight matrices at the end. However, these weight matrices are not significant for the experiments, as the classes in the face

datasets differ from those in the ImageNet dataset. To adapt the model to the task, two new weight matrices in the classification head section with a Leaky ReLU activation function are introduced. Kaiming initialization to initialize these weight matrices is employed. Kaiming initialization has been utilized to prevent the activation outputs of the layers from exploding during the forward pass in a deep neural network. At each layer '*l*', the weight matrix is initialized with random numbers drawn from a standard normal distribution, where each random number is multiplied by the value of '*fan_in*', representing the number of input connections or the number of neurons in the previous layer that connect to the current layer. It indicates the size of the input space for a specific layer '*l*'. Further, '*fan_out*' refers to the number of output connections, or the number of neurons in the current layer that connect to the next layer. It represents the size of the output space for a specific layer. Therefore, in the present application, Xavier initialization is replaced, where the weights of a layer are initialized using random values selected from a uniform distribution with specific bounds, as shown in equation (8.1).

$$SD = \frac{\sqrt{2}}{\sqrt{fan\_in + fan\_out}} \quad (8.1)$$

Here, SD is the Standard Deviation of the random numbers drawn from a standard normal distribution, which are used to initialize the weights of a layer. The initial layers of the CNN model are responsible for extracting features, while the final layers are utilized for classification purposes. The compact representations of the pre-trained models used in this application (VGG, ResNet, and DenseNet) are illustrated in Figure 8.2.

However, based on experimental findings in the field of facial recognition algorithms, relying solely on pre-trained models is insufficient to achieve optimal accuracy. Therefore, certain modifications have been implemented to enhance the recognition accuracy of these models. In this application, a modified architecture for the base models is introduced, which involves the incorporation of global pooling, batch normalization (BN), and dropout in the classification layers. The addition of a pooling layer helps reduce the number of trainable parameters in the model. Typically, two types of pooling techniques have been employed, namely average pooling and max pooling, which can be mathematically described using equations (8.2) and (8.3).

$$P = O_{max}^{n,n}(F) \tag{8.2}$$

$$P = O_{avg}^{n,n}(F) \tag{8.3}$$

In the present application, the input feature map *F* obtained from the previous convolutional layer has been processed using pooling operations. The maximum pooling operation, denoted as $O_{max}^{n,n}(F)$ operates on the input feature map of size $n \times n$, while the average pooling operation, denoted as $O_{avg}^{n,n}(F)$, calculates the average value. The output of the pooling layer, denoted as *P*, is obtained by concatenating the maximum and average values using the concatenate function in the Keras library. Both the maximum and average pooling techniques have their advantages, and their performance can vary depending on the activation map's maximum and average values. To preserve both of these values, the concatenation technique has been employed. Global pooling is used to reduce each channel in a feature map to a single value and serves as an alternative to

densely connected or fully connected layers in a classifier. It helps reduce the model's complexity. Batch normalization has been utilized to normalize the positive and negative features from the previous convolutional layer, addressing the issue of covariate shift. It effectively improves accuracy without any side effects. To prevent overfitting, dropout layers have been added for regularization. Dropout is a regularization technique that randomly drops out a fraction of the neurons during training. The optimal dropout value for the model is determined experimentally, as it can significantly impact the model's accuracy. The non-linearity functions, such as ReLU, PReLU, Leaky ReLU, *etc.*, can be placed before or after the BN layer. In the present application, the use of Leaky ReLU after the BN layer gives better results. Therefore, the Leaky ReLU activation function is used because it alleviates the problem of "dying ReLU". The mathematical expression for calculating the value of Leaky ReLU is provided in equation (8.4).

$$f(x) = \max(0.01 * x, x) \qquad (8.4)$$

The Leaky ReLU activation function is defined as follows: when given a positive input x, it produces a value of x; however, if the input is negative, it outputs a minimum value of 0.01 times x. This modification allows Leaky ReLU to produce an output for negative inputs as well. Unlike the standard ReLU function, this modification results in a non-zero gradient on the left side of the mathematical graph, effectively addressing the issue of "dead neurons" in that region. The modified architecture of the baseline models, incorporating Leaky ReLU and other enhancements, is depicted in Figure 8.3. The reasons for these modifications in the

baseline model architecture are discussed in Table 8.1, outlining the benefits and justifications for each modification.

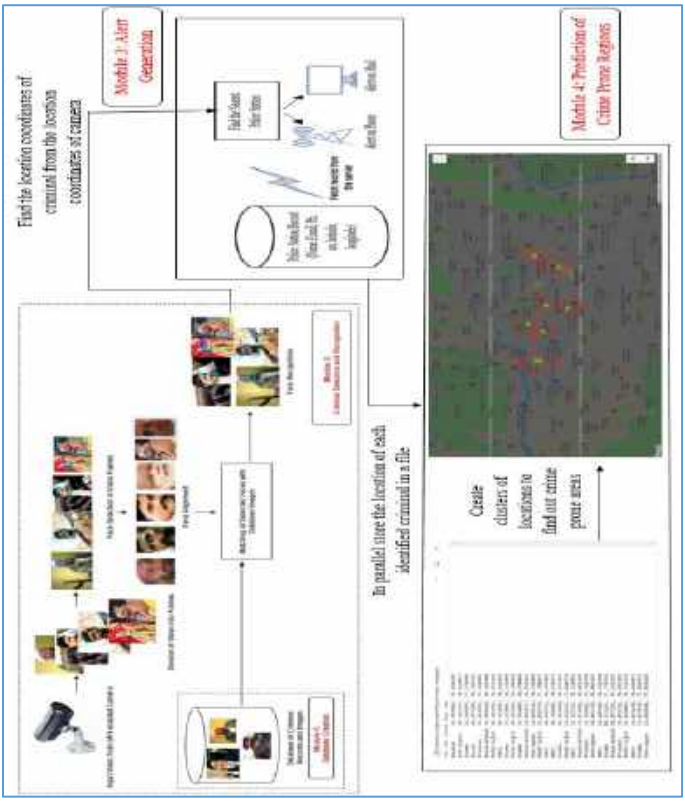

Figure 8.1 The Schematic Flow of an Automated Face Recognition System

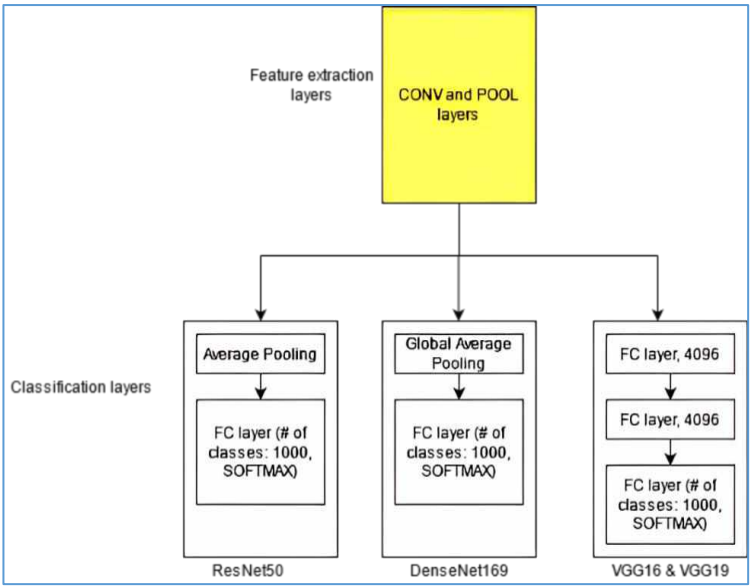

Figure 8.2 The Architecture of Classification Layers of Pre-Trained Models (ResNet50, DenseNet169, VGG16, and VGG19)

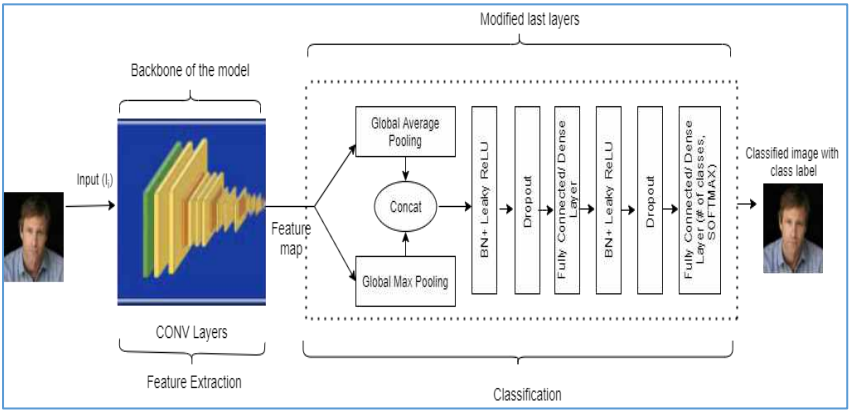

Figure 8.3 The Modified Architecture of the Baseline Model Consisting of GMP, GAP, BN, dropout, and FC layers (The Dotted Line Shows the Modified Part of the Model)

Table 8.1 The Persuasive Reasons for the Rectification of the Classification Layers of Baseline Models

S. No.	Charact-eristics	Standard ResNet50	Standard VGG19	Standard DenseNet169	Modified Architecture	Persuasive Reasons for the Modifications
1.	Pooling layer	Average pooling	No pooling layer	GAP	Concatenation of GAP and GMP	The activation map from the previous layer can outperform its mean value, and vice versa.
2.	No. of FC layer	1	3	1	2	Adding a layer enhances ResNet and DenseNet, while removing one improves VGG.
3.	Linear activation	ReLU	ReLU	ReLU	Leaky ReLU	To address the issue of dying ReLU.
4.	Regularization	No dropout	No dropout	No dropout	The dropout layer is used	To minimize the overfitting issue.

8.3 HYBRID ENSEMBLE CNN (HE-CNN) MODEL

The use of ResNet, VGG, and DenseNet in an ensemble model for face recognition can lead to improved accuracy, robustness, diversity, and flexibility in the model design. ResNet, VGG, and DenseNet are all deep neural network models that

have been widely used for image classification tasks. Each of these models boasts a distinct architecture designed to encapsulate varied features and patterns inherent in an image. Through amalgamating their outputs, the ensemble model attains superior accuracy in contrast to any standalone model. Ensemble models exhibit heightened resilience against overfitting and a greater propensity to generalize well on novel data. By melding disparate models, the ensemble can glean insights from a broader spectrum of features and patterns, cultivating robustness against input variations. ResNet, VGG, and DenseNet each present distinct architectural nuances. Consequently, integrating them within an ensemble introduces diversity to the models, thereby fostering enhanced overall performance. Specific strengths and weaknesses in feature extraction are intrinsic to each model. For instance, ResNet might excel in capturing edge and corner features, whereas DenseNet might prove adept at discerning intricate patterns within textures. We can take advantage of their complementary strengths by combining the models. Using an ensemble allows for more flexibility in the design of the model. By adjusting the weights assigned to each model, we can optimize the ensemble to achieve the desired level of accuracy, efficiency, and resource utilization. Therefore, the ensemble of the modified versions of these three models has been used to get an optimized hybrid model for the face recognition task.

The stacking method, a hallmark of ensemble learning, has employed to devise this hybrid model tailored for the face recognition task. The ultimate prediction of the hybrid model is realized by aggregating outcomes from the fine-tuned baseline models through a weighted sum operation. This operation, commonly known as weighted average, is often employed in ensemble models to accord greater significance to predictions

from better-performing models. The idea behind the weighted sum operation is to give different weights to the predictions of each model in the ensemble based on how well they did on a validation set. Models exhibiting superior performance receive higher weights, while those performing less effectively receive lower weights. As a result, the ensemble's final prediction bears a more pronounced influence from the better-performing models and a diminished influence from the weaker ones. Applying the weighted sum operation to ensemble models allows us to harness the strengths of multiple models while mitigating their shortcomings. This approach fosters superior overall performance and more resilient predictions. The predicted face using the hybrid ensemble model, denoted as PHE−CNN, is defined by equation (8.5). Through experimentation, VGG19 showcases superior accuracy. Consequently, the VGG19 version is chosen over the VGG16 counterpart. Nonetheless, the face recognition stage in the present application incorporated VGG19, DenseNet169, and ResNet50 within the hybrid model. The comprehensive procedure is visually outlined in Figure 8.4.

$$P_{HE-CNN} = \sum_{i=1}^{\S} W_i \sum_{j=1}^{n} \frac{1}{\sum_{k=1}^{C} e^{\theta_k^T f^{(j)}}} \begin{pmatrix} e^{\theta_1^T f^{(j)}} \\ \dots \\ e^{\theta_C^T f^{(j)}} \end{pmatrix} \quad (8.5)$$

Here W_i denotes the weight of modified baseline models, § is 3 because the HE-CNN considered three models, n is the count of image samples in training data, C denotes the number of classes in a dataset, $f^{(j)}$ is the feature of the j^{th} sample, θ is the parameter matrix of the softmax loss function $L(\theta)$, and $\theta_k^T f^{(j)}$ denotes the inner product of θ_k and $f^{(j)}$. The optimal values of

W_1, W_2, and W_3 are selected using the VotingClassifier available in the scikit-learn library of Python (https://rb.gy/p0ig).

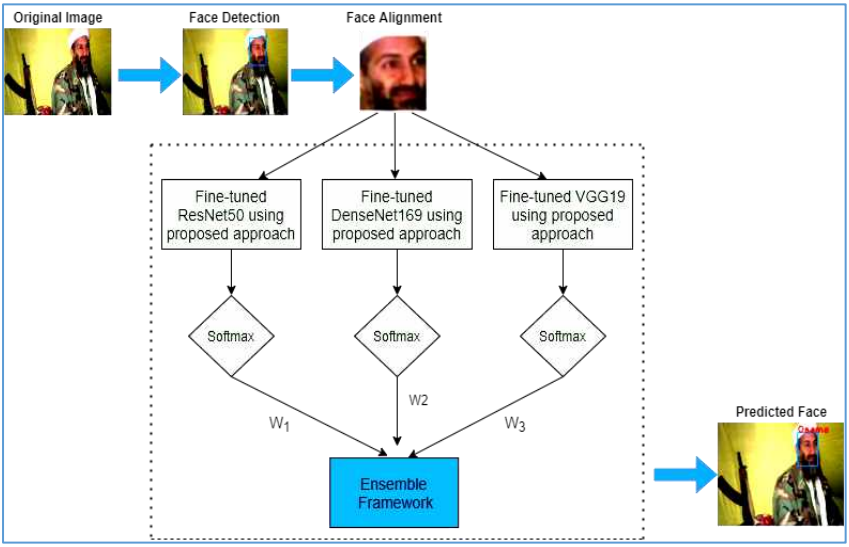

Figure 8.4 The Hybrid Ensemble CNN (HE-CNN) Model

8.4 VARIOUS MODULES OF THE AUTOMATED FACE RECOGNITION SYSTEM

8.4.1 Self-Curated Dataset and Database of Criminals' and Police Officials' Records

In the first module, images of criminals from the Internet (freely available sources) are collected and stored those images in different directories labeled with their names, as given in Figure 8.5. The information about the mugshots, such as the crime date, crime type, age, *etc.*, is stored in the database, as delineated in Figure 8.6 (a). The other table in the database is also created to store the registered email ID, mobile number, and location coordinates of the police stations manifested in Figure 8.6 (b). Personal numbers are used for testing the system; that is why

they are scraped in the image given in Figure 8.6 (b) due to privacy concerns.

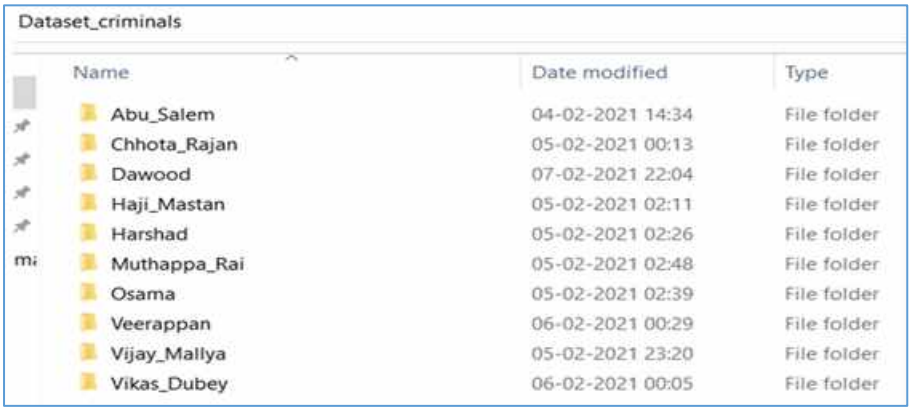

Figure 8.5 Images of Criminals Collected from the Internet

Criminal_ID	Name	Age	Gender	Crime_type	Crime_date	Identity_mark
Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	Abu_Salem	59	M	Robbery;murder;gangster	2019-01-01;2020-05-01;2020-12-03	Cut mark on forehead
2	Chhota_Rajan	62	M	Mobster;smuggling;extortion	2019-02-01;2020-05-01;2020-12-03	Cut mark on left hand
3	Dawood	44	M	Murder;gangster;terrorism	2018-02-01;2018-01-01;2020-12-03	Cut mark on right eye
4	Haji_Mastan	68	M	Kidnapping;murder	2018-02-01;2019-02-01	Cut mark on left eye
5	Harshad	47	M	Scam	2018-02-01	Cut mark on left elbow
6	Muthappa_Rai	42	M	Smuggling, Kidnapping	2018-02-01;2019-02-01	Cut mark on right elbow
7	Osama	55	M	Terrorism;bombblast	2018-02-01;2019-02-01	Cut mark on forehead
8	Veerappan	50	M	Kidnapping;robbery	2018-02-01;2019-02-01	Cut mark on right hand
9	Vijay_Mallya	65	M	Scam	2018-02-01	Cut mark on chest
10	Vikas_Dubey	40	M	Stolen property;kidnapping;murder	2018-02-01;2019-02-01;2020-02-05	Cut mark on chin

(a)

Police_Chowki_ID	Police_chowki_Name	Email	Phone_number	latitude	longitude
1	Laxman Chowk Police Chowki	sanwarul@ddn.upes.ac.in	[redacted]	30.321661	78.021585
2	Police Sahayata Kendre	dgc-police-ua@nic.in	9411112780	30.338329	78.020920
3	Bindaal Police Chowki	dgc-police-ua@nic.in	1352716235	30.329576	78.031539
4	Lakhibagh Police Station	dgc-police-ua@nic.in	9411112809	30.316359	78.032544
5	Kotwali Police Station	dgc-police-ua@nic.in	1352716216	30.320393	78.037553
6	Police Headquarters SSP Office ...	ssp-deh-ua@nic.in	1352716203	30.317734	78.039864
7	Police Chowki Khudura	dgc-police-ua@nic.in	1352616216	30.321719	78.032672
8	Patel Nagar Police Station	dgc-police-ua@nic.in	1352716219	30.292877	78.017799
9	Dhara Chowki	dgc-police-ua@nic.in	1352716216	30.325680	78.042870
10	Uttarakhand Police Headquarters	dgc-police-ua@nic.in	1352712685	30.329997	78.050405
11	Panditwari Police Chowki	sanwarul@ddn.upes.ac.in	[redacted]	30.3324	77.9881
12	Prem nagar Police Chowki	shahinaanwarul@gmail.com	[redacted]	30.333092	77.961016

(b)

Figure 8.6 Record Stored in Database (a) Criminals' Records (b) Police Officials' Records

8.4.2 Detection and Recognition Module

This module contains two steps, namely face detection and face recognition. The first step is to detect the faces and then compare the detected and aligned faces from the gallery images to recognize the mugshot.

8.4.2.1 Face Detection

Face detection is a technology that identifies human faces in an image. Various traditional and deep learning-based methods have been introduced as SOTA for face detection. Traditional approaches like the Haar classifier, also known as the Viola-Jones algorithm, and the LBP classifier for face detection have their benefits and drawbacks, but the major differences are in terms of speed and accuracy. So, a Haar classifier is used in cases where there is a requirement for more accurate detections. But the LBP classifier is faster and, therefore, should be used in

mobile applications or embedded systems. The Haar classifier and the LBP classifier, both conventional algorithms, fail to achieve the desired detection accuracy, as demonstrated by the experimental findings presented in Table 8.2 and Table 8.3. Deep network approaches like MTCNN and SSD for face detection are efficacious in terms of their detection accuracy.

Table 8.2 Detection Accuracy (Number of Detected Faces/Total Faces in an Image) and Time (in Sec) of Face Detection Algorithms on Sample Images

S. No.	Face Detection Algorithm	No. of Faces Found in an Image/ No. of Faces Present in an Image	Time (in Sec)
1.	SSD (Single Shot Multi-Box Detector)	Image 1: 2/2	**0.051**
		Image 2: 5/5	**0.032**
2.	MTCNN (Multi-Task Cascaded Convolutional Networks)	Image 1: 2/2	2.723
		Image 2: 5/5	2.998
3.	Haar Cascade (Viola Jones)	Image 1: ½	0.234
		Image 2: 4/5	0.051
4.	LBP Cascade	Image 1: ½	0.112
		Image 2: 1/5	0.045

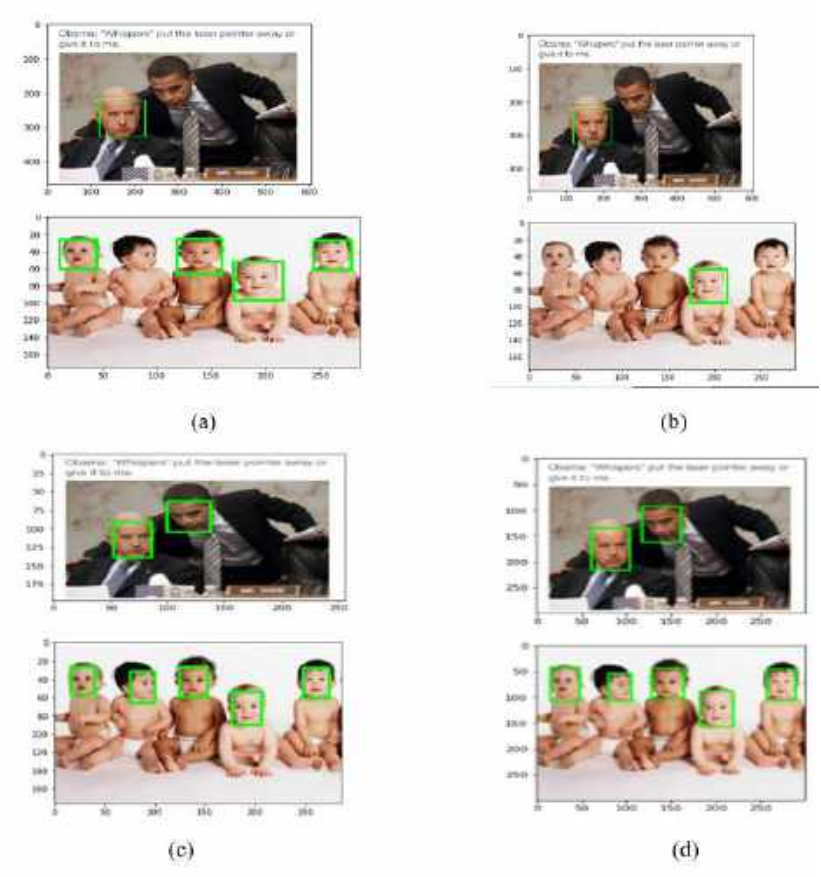

Figure 8.7 Outputs of Different Face Detection Algorithms: (a) Face Detection using Haar Cascade (b) Face Detection using LBP Cascade (c) Face Detection using MTCNN (d) Face Detection using SSD

Images containing multiple faces are considered for the evaluation of the State-of-the-Art face detection algorithms because a video frame can have more than one face at a particular instant in time. From Table 8.2, Figure 8.7, and Table 8.3, it is experimentally proven that Haar cascade is accurate but slower

than LBP cascade, while LBP is faster but less accurate. MTCNN and SSD are used in applications where accuracy has greater importance. But SSD is much faster in comparison to MTCNN and provides equivalent accuracy. The evaluation of the discussed face detection methods is also conducted using standard datasets like LFW, CPLFW, and the self-curated dataset in Table 8.3. After evaluating all the discussed methods, it is concluded that SSD is an efficient approach in terms of accuracy and processing time for face detection. Therefore, the SSD framework is used to detect faces for the face recognition stage in the proposed recognition system.

Table 8.3 Detection Score of Various Face Detection Algorithms (in %)

		Face Detection Algorithm											
		SSD			**MTCNN**			**LBP Cascade**			**Haar Cascade**		
S. No.	**Dataset**	**TPR**	**FPR**	**FNR**	**TPR**	**FPR**	**FNR**	**TPR**	**FPR**	**FNR**	**TPR**	**FPR**	**FNR**
1.	CPLFW	96.9	1.4	1.7	96.3	1.3	2.4	44.2	0.7	55.1	53.9	0.5	45.6
2.	LFW	99.2	0.8	0	99.3	0.7	0	94.3	0.8	4.9	98.4	0.6	1
3.	Criminal Dataset	92.9	1.7	5.4	93.4	2.3	4.3	44	0.6	55.4	53.7	1.3	45

8.4.2.2 Face Recognition

A small criminal dataset has been created containing 25 images of each class of criminals, namely Haji Mastan, Vijay Mallya, Dawood, Harshad, Osama, Veerappan, Chhota Rajan, Muthappa Rai, Abu Salem, and Vikas Dubey, by downloading these images from the Internet to demonstrate the real-time application of face recognition. Mislabeled and vague images from the downloaded images are manually deleted, and 25 images of each class are considered to make a class-balanced dataset. Data augmentation, known as the oversampling technique, has been utilized to expand the count of samples in each class. In order to maintain the balance of the class, images in individual classes have been augmented to generate 50 samples using a set of transformations such as vertical flip, horizontal flip, mirroring, warping, scaling, rotation, zooming, and lighting. The dataset is divided into two sets consisting of 80% and 20% samples (*i.e.*, 400 samples of the dataset are considered for training and 100 samples are taken for testing). Testing on a self-created dataset has been done in two ways. Firstly, the testing has been done using 100 random samples from 500 images. The accuracy of the HE-CNN model on a self-created dataset is 95%, while the precision score, recall score, and error rate are 0.954, 0.952, and 5%, respectively. The confusion matrix to analyze the correct results present in the diagonal of the matrix is delineated in Figure 8.8.

Secondly, another testing dataset contains 50 images with more than one face to demonstrate the real-time surveillance results. The set of 50 images is distinct from the collection of 500 images but consists of the faces of the same 10 criminals and other unknown individuals. As the testing images contain more than one face, the recognition rate of the proposed technique in

the criminal dataset has been calculated by manually analyzing each image, as shown in Figure 8.9, and achieving 87% recognition accuracy.

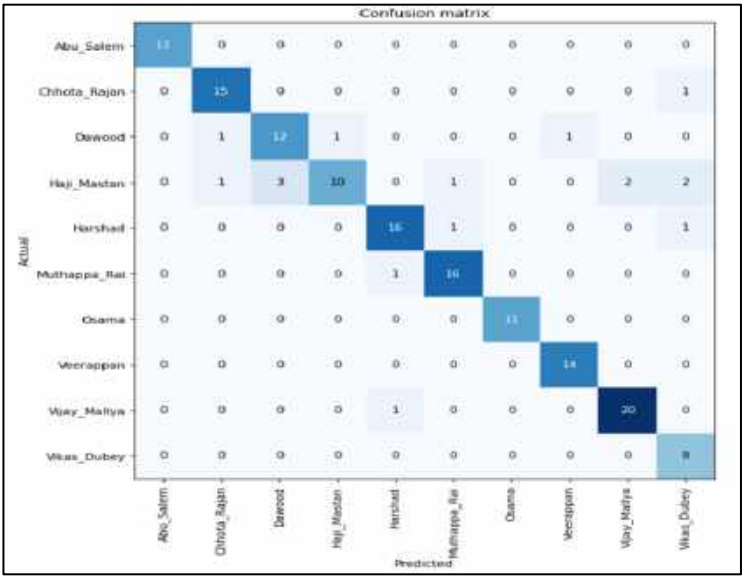

Figure 8.8 Confusion Matrix of Self-Curated Dataset Results

Figure 8.9 Output of the Face Recognition Stage on Self-Curated Dataset

8.4.3 Alert Generation

In this module, the location of the GPS-enabled CCTV camera has been identified after the successful recognition of any suspect, as shown in Figure 8.10. In this application, the camera of the system has been used for experimental purposes. Once the criminal has been identified and recognized, the nearest police station from the current location of the criminal is searched using the Haversine formula given in equation (8.6), and automatically, it sends an alert via mail and message to the registered email ID and contact number of the police official stored in the database given in Figure 8.6 (b). The Haversine formula has been used to calculate the shortest distance between the two locations on the sphere using their latitudes and longitudes. The information, photograph, and current location of the criminal are sent to the registered email ID, and the location is also sent to the registered mobile number, as shown in Figure 8.11.

$$H = 2r\sin^{-1}\left(\sqrt{sin^2\left(\frac{a_2-a_1}{2}\right) + \cos(a_1)\cos(a_2)\,sin^2\left(\frac{\varepsilon_2-\varepsilon_1}{2}\right)}\right) \quad (8.6)$$

Where r is taken as the earth's radius (6371 km), the distance between the two location points on the earth is H; a_1, a_2 are considered the latitudes of the two points, and ε_1, ε_2 are taken as the longitudes of the two points on the earth.

Figure 8.10 Current Location of Criminal

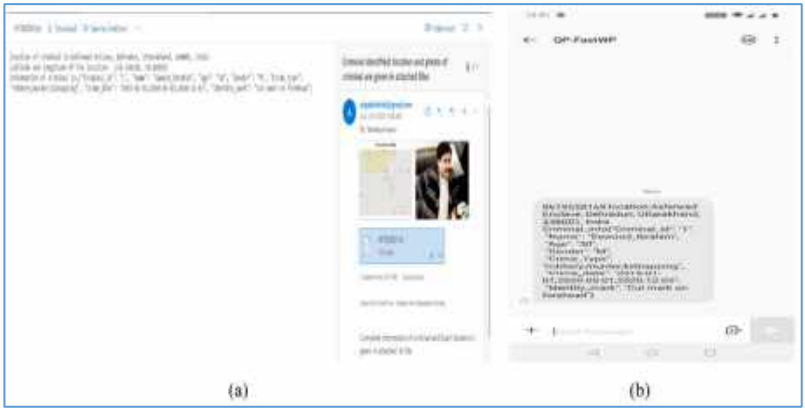

Figure 8.11 Alert Generation via: (a) Mail and (b) Message

8.4.4 Prediction of Crime Prone Areas

All the identified locations (*i.e.*, latitude and longitude) of criminals have been saved in a file given in Figure 8.12 in parallel once the criminal has identified in any location to cluster the crime-prone regions. On a weekly basis, the system generates the

crime regions' clusters and shows those regions on Google Maps. This module of the criminal recognition system helps police officials analyze crime-prone areas. K-means clustering has been utilized to identify the clusters of crime-prone regions (areas where most of the criminals are identified), as highlighted in red in Figure 8.13.

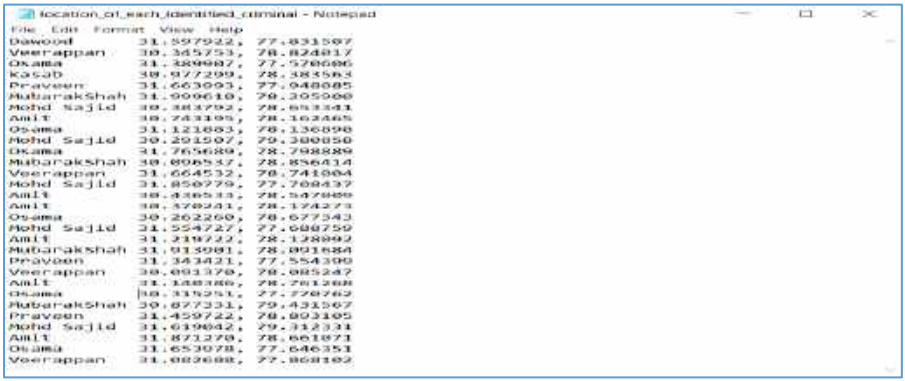

Figure 8.12 Location of Identified Criminals

Figure 8.13 Clusters of the Crime Prone Region

9

CHAPTER

CONCLUSION AND FUTURE DIRECTIONS

Face recognition is a challenging task in video surveillance due to the presence of various unconstrained factors such as pose variation, occlusion, illumination, and low resolution. There is a need for continuous monitoring of CCTV footage to identify the individual in recordings of existing face recognition systems. The automated recognition system helps concerned officials monitor the surveillance area without human intervention. It automatically alerts police officials when there is an identification of criminals in a specified area. It also helps to prevent crimes by providing clusters of crime-prone areas. Due to this, the police officials will become attentive before the crime happens in those areas. The extensive developments in face recognition in recent years have given immense scope to criminal identification and other applications. The emergence of deep learning has made recognition systems accurate, but it requires a large dataset for training the machine. The non-availability of a considerable number of images of criminals limits the accuracy of current systems. Therefore, the advocated solution given in this book takes advantage of transfer learning and extracts features from the modified models trained on the ImageNet dataset that can be executed with less data. In this book, it became evident that the transfer learning approach surpasses conventional methods in terms of performance. The development of the hybrid architecture, known as the HE-CNN model, for face recognition is based on extensive experimentation. Given the challenges associated with collecting a significant volume of face images due to privacy considerations, the present work given in this book offers an optimal solution through the implementation of deep ensemble transfer learning. The primary driving force behind this book was to design and develop an automated face recognition

system based on ensemble deep learning methods with superior accuracy and minimal complexity.

The automated HE-CNN model for face recognition provides better recognition accuracy than the baseline models due to the effectiveness of ensemble learning-based models. However, the topic presented in this book has a wider scope, with several extensions addressing a variety of challenges that require future attention, as is the case with many other academic articles in the same field. The expansion of the self-curated dataset can involve procuring additional image samples for each class through free sources on the Internet. Alternatively, employing various data augmentation techniques, like elastic deformation to replicate distortions and stretches or introducing Gaussian noise, can aid the model in acquiring the ability to discern objects within noisy environments.

It is essential to acknowledge a limitation of the system: it can identify individuals based on images presented in front of the camera, rendering it susceptible to spoofing. Therefore, an extension of the work given in this book could involve identifying spoofed faces to enhance security. Additionally, opportunities for collaboration with government and law enforcement entities could lead to large-scale implementation of the work.

Furthermore, it is worth considering the incorporation of alternative biometric modalities or soft biometric characteristics as supplementary sources of data. This approach can contribute to the development of face recognition systems that are both more dependable and precise, aligning better with the demands of real-world video surveillance applications.

REFERENCES

1 *LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.*

2 *Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.*

3 *Lin, M., Chen, Q., & Yan, S. (2013). Network in network. Computing Research Repository (CoRR), 1312.4400.*

4 *Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. In 3rd International Conference on Learning Representations (ICLR 2015) (pp. 1-14).*

5 *Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826).*

6 *Quinn, J., McEachen, J., Fullan, M., Gardner, M., & Drummy, M. (2019). Dive into deep learning: Tools for engagement. Corwin Press.*

7 *He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).*

8 *Tsang, S. (2018). Review: DenseNet-Dense Convolutional Network (Image Classification). [Online]. Available: https://towardsdatascience.com/review-densenet-image-classification-b6631a8ef803. [Accessed 5 July 2024].*

9 *Ding, C., & Tao, D. (2017). Trunk-branch ensemble convolutional neural networks for video-based face recognition. IEEE transactions on pattern analysis and machine intelligence, 40(4), 1002-1014.*

10 *Tang, J., Su, Q., Su, B., Fong, S., Cao,W., & Gong, X. (2020). Parallel ensemble learning of convolutional neural networks and*

local binary patterns for face recognition. Computer Methods and Programs in Biomedicine, 197, 105622.

11 *Canziani, A., Paszke, A., & Culurciello, E. (2016). An analysis of deep neural network models for practical applications. arXiv preprint arXiv:1605.07678.*

12 *Zhang, Y., Deng, W., Wang, M., Hu, J., Li, X., Zhao, D., & Wen, D. (2020). Global-local gcn: Large-scale label noise cleansing for face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 7731-7740).*

13 *"Ms-celeb-1m challenge 3". [Online] Available: http://trillionpairs.deepglint.com.*

14 *Wang, F., Chen, L., Li, C., Huang, S., Chen, Y., Qian, C., & Loy, C. C. (2018). The devil of face recognition is in the noise. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 765-780).*

15 *Cao, Q., Shen, L., Xie, W., Parkhi, O. M., & Zisserman, A. (2018, May). Vggface2: A dataset for recognising faces across pose and age. In 2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018) (pp. 67-74). IEEE.*

16 *Ba Bansal, A., Castillo, C., Ranjan, R., & Chellappa, R. (2017). The do's and don'ts for cnn-based face verification. In Proceedings of the IEEE international conference on computer vision workshops (pp. 2545-2554).*

17 *Guo, Y., Zhang, L., Hu, Y., He, X., & Gao, J. (2016). Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In ComputerVision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14 (pp. 87-102). Springer International Publishing.*

18 *Nech, A., & Kemelmacher-Shlizerman, I. (2017). Level playing field for million scale face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 7044-7053).*

19 *Parkhi, O. M., Vedaldi, A. & Zisserman, A. (2015). Deep face recognition. BMVC.*

20 *Yi, D., Lei, Z., Liao, S., & Li, S. Z. (2014). Learning face representation from scratch. arXiv preprint arXiv:1411.7923.*

21 *Maze, B., Adams, J., Duncan, J. A., Kalka, N., Miller, T., Otto, C., ... & Grother, P. (2018, February). Iarpa janus benchmark-c: Face dataset and protocol. In 2018 international conference on biometrics (ICB) (pp. 158-165). IEEE.*

22 *Wang, M., Deng, W., Hu, J., Tao, X., & Huang, Y. (2019). Racial faces in the wild: Reducing racial bias by information maximization adaptation network. In Proceedings of the ieee/cvf international conference on computer vision (pp. 692-702).*

23 *Whitelam, C., Taborsky, E., Blanton, A., Maze, B., Adams, J., Miller, T., ... & Grother, P. (2017). Iarpa janus benchmark-b face dataset. In proceedings on the IEEE conference on computer vision and pattern recognition workshops (pp. 90-98).*

24 *Zheng, T., & Deng, W. (2018). Cross-pose lfw: A database for studying cross-pose face recognition in unconstrained environments. Beijing University of Posts and Telecommunications, Tech. Rep, 5(7).*

25 *Zheng, T., Deng, W., & Hu, J. (2017). Cross-age lfw: A database for studying cross-age face recognition in unconstrained environments. arXiv preprint arXiv:1708.08197*

26 *Sengupta, S., Chen, J. C., Castillo, C., Patel, V. M., Chellappa, R., & Jacobs, D. W. (2016, March). Frontal to profile face verification in the wild. In 2016 IEEE winter conference on applications of computer vision (WACV) (pp. 1-9). IEEE.*

27 *Bansal, A., Nanduri, A., Castillo, C. D., Ranjan, R., & Chellappa, R. (2017, October). Umdfaces: An annotated face dataset for training deep networks. In 2017 IEEE international joint conference on biometrics (IJCB) (pp. 464-473). IEEE.*

28 *Klare, B. F., Klein, B., Taborsky, E., Blanton, A., Cheney, J., Allen, K., ... & Jain, A. K. (2015). Pushing the frontiers of unconstrained*

face detection and recognition: Iarpa janus benchmark a. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1931-1939).

29 *Huang, Z., Shan, S., Wang, R., Zhang, H., Lao, S., Kuerban, A., & Chen, X. (2015). A benchmark and comparative study of video-based face recognition on cox face database. IEEE Transactions on Image Processing, 24(12), 5967-5981.*

30 *Beveridge, J. R., Phillips, P. J., Bolme, D. S., Draper, B. A., Givens, G. H., Lui, Y. M., ... & Cheng, S. (2013, September). The challenge of face recognition from digital point-and-shoot cameras. In 2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS) (pp. 1-8). IEEE.*

31 *Wolf, L., Hassner, T., & Maoz, I. (2011, June). Face recognition in unconstrained videos with matched background similarity. In CVPR 2011 (pp. 529-534). IEEE.*

32 *Wong, Y., Chen, S., Mau, S., Sanderson, C., & Lovell, B. C. (2011, June). Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition. In CVPR 2011 WORKSHOPS (pp. 74-81). IEEE.*

33 *"Fg-net aging database". [Online]. Available: http://www.fgnet.rsunit.com.*

34 *Kim, M., Kumar, S., Pavlovic, V., & Rowley, H. (2008, June). Face tracking and recognition with visual constraints in real-world videos. In 2008 IEEE Conference on computer vision and pattern recognition (pp. 1-8). IEEE.*

35 *Huang, G. B., Mattar, M., Berg, T., & Learned-Miller, E. (2008, October). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In Workshop on faces in 'Real-Life' Images: detection, alignment, and recognition.*