Probability Density Function based Fusion of CNN Models for Face Recognition from Video

Thesis submitted in partial fulfillment of the requirements for the degree of

Master of Engineering
in
Computer Science and Engineering

by

Saikat Mazumdar
Registration No. - 163569 of 2022 - 2023
Exam Roll No. - M4CSE24013
Class Roll No. - 002210502018
Session - 2022 - 2024

Under the supervision of

Prof. (Dr.) Jamuna Kanta Sing

Department of Computer Science and Engineering
Faculty Council of Engineering and Technology
Jadavpur University
188, Raja S. C. Mallick Road,
Kolkata - 700032, West Bengal, India

Department of Computer Science and Engineering
Faculty Council of Engineering and Technology
Jadavpur University, Kolkata - 700032

CERTIFICATE OF RECOMMENDATION

This is to certify that the work embodied in this thesis entitled **"Probability Density Function based Fusion of CNN Models for Face Recognition from Video"** has been satisfactorily completed by **Saikat Mazumdar (Registration Number 163569 of 2022 - 2023; Class Roll No. 002210502018; Examination Roll No. M4CSE24013)**. It is a bonafide piece of work carried out under my supervision and guidance at Jadavpur University, Kolkata, for partial fulfillment of the requirements for the awarding of the **Master of Engineering** in **Computer Science and Engineering** degree of the Department of Computer Science and Engineering, Faculty Council of Engineering and Technology, Jadavpur University, during the academic year 2022 - 2024.

Prof. (Dr.) Jamuna Kanta Sing,
(Supervisor)
Department of Computer Science and Engineering,
Jadavpur University,
Kolkata - 700032

Forwarded By:

Prof. (Dr.) Nandini Mukherjee,
Head of the Department,
Department of Computer Science and Engineering,
Jadavpur University,
Kolkata - 700032

Prof. (Dr.) Dipak Laha,
DEAN,
Faculty Council of Engineering and Technology,
Jadavpur University,
Kolkata - 700032

Department of Computer Science and Engineering
Faculty Council of Engineering and Technology
Jadavpur University, Kolkata - 700032

CERTIFICATE OF APPROVAL

This is to certify that the thesis entitled **"Probability Density Function based Fusion of CNN Models for Face Recognition from Video"** is a bonafide record of work carried out by **Saikat Mazumdar (Registration Number 163569 of 2022-2023; Class Roll No. 002210502018; Examination Roll No. M4CSE24013)** in partial fulfillment of the requirements for the award of the degree of **Master of Engineering** in **Computer Science and Engineering** degree in the Department of Computer Science and Engineering, during the period of September 2022 to May 2024. It is understood that by this approval, the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the thesis only for the purpose of which it has been submitted.

_____
Signature of Examiner 1
Date:

_____
Signature of Examiner 2
Date:

Department of Computer Science and Engineering
Faculty Council of Engineering and Technology
Jadavpur University, Kolkata - 700032

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that the thesis entitled **"Probability Density Function based Fusion of CNN Models for Face Recognition from Video"** contains literature survey and original research work by the undersigned candidate, as a part of his degree of **Master of Engineering** in **Computer Science and Engineering, Jadavpur University**. All the information have been obtained and presented in accordance with academic rules and ethical conduct.
I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name: Saikat Mazumdar

Examination Roll No.: M4CSE24013

Registration No.: 163569 of 2022-2023

Thesis Title: Probability Density Function based Fusion of CNN Models for Face Recognition from Video

Signature of the Candidate:

Acknowledgements

I am grateful to the Almighty God for giving me the necessary strength, knowledge and understanding of the topic to complete this thesis. His love has been more than sufficient to keep and sustain me.

I dedicate this thesis to my prestigious university, **"Jadavpur University"** for providing me with everything that was required in order to complete this thesis paper.

I convey my honest and sincere thanks to my respected professor and my guide ***Prof. (Dr.) Jamuna Kanta Sing***, Professor of Department of Computer Science & Engineering, Jadavpur University, for his exclusive guidance and undivided support in completing and presenting this thesis successfully. I am indebted to him for the constant encouragement and continuous inspiration that he has given me. The above words are only a token of my deep respect towards him for all that he has done to take my thesis to its present shape.

I would also like to thank ***Mr. Akash Halder*** for his valuable suggestions and key insights that he has provided while working on this thesis.

Finally, I express my sense of gratitude and thanks to my family members, especially my parents for their unconditional love and support. Last but not the least I am grateful to all my friends for staying beside me and being an endless source of optimism and positive thoughts.

Saikat Mazumdar

Examination Roll No.: M4CSE24013

Registration No.: 163569 of 2022-2023

Department of Computer Science and Engineering,

Jadavpur University.

Contents

Table of Contents i

List of Figures ii

List of Tables iv

List of Abbreviations v

Abstract vii

1 Introduction 1
- 1.1 Literature Survey . . . 1
  - 1.1.1 Fusion Before Matching . . . 1
  - 1.1.2 Fusion After Matching . . . 2
- 1.2 Face recognition frameworks . . . 2
  - 1.2.1 Classical Approaches . . . 2
  - 1.2.2 Modern Approaches . . . 3
- 1.3 CNN Based Approaches . . . 3
  - 1.3.1 VGG16 . . . 4
  - 1.3.2 VGG19 . . . 4
  - 1.3.3 AlexNet . . . 4
  - 1.3.4 Inception-v3 . . . 4
  - 1.3.5 Resnet-50 . . . 4
  - 1.3.6 InceptionResNetV2 . . . 4
  - 1.3.7 Xception . . . 5
- 1.4 Activation Functions . . . 5
  - 1.4.1 ReLU . . . 5
  - 1.4.2 Leaky ReLU . . . 5
  - 1.4.3 tanh . . . 5
  - 1.4.4 Sigmoid . . . 5
  - 1.4.5 SoftMax . . . 6
- 1.5 Rank-level Fusion Approaches . . . 6
  - 1.5.1 Highest Rank method . . . 6
  - 1.5.2 Modified Highest Rank method . . . 6
  - 1.5.3 Borda Count . . . 7
  - 1.5.4 Weighted Borda Count . . . 7
  - 1.5.5 Nonlinear weighted rank fusion method . . . 7

2 Proposed Method **8**
2.1 Face Cropping using MTCNN & OpenCV 9
2.1.1 MTCNN 9
2.2 CNN Based Classifiers 10
2.2.1 VGG16 10
2.2.2 InceptionResnetV2 11
2.2.3 Xception 12
2.3 Mathematical Formulation 13
2.3.1 Gaussian Density Function 15
2.3.2 Triangle Density Function 16
2.3.3 Trapezoidal Density Function 17

3 Experiment and Results **19**
3.1 Dataset 19
3.1.1 Chokepoint Dataset 19
3.1.2 Custom Dataset 21
3.2 Evaluation Metrics 22
3.3 Implementation Details 23
3.4 Results 23
3.4.1 Chokepoint Dataset 23
3.4.2 Custom Dataset 31

4 Conclusions **37**
4.1 Conclusions of the Present Work 37
4.2 Scope for Future Works 37

Bibliography **39**

List of Figures

2.1 Cropping Face from a Video using MTCNN 10
2.2 VGG16 Architecture 10
2.3 InceptionResnetV2 Architecture 12
2.4 Xception Architecture 12
2.5 Workflow of Proposed Method 14
2.6 Gaussian Density Function 15
2.7 Triangle Density Function 16
2.8 Trapezoidal Density Function 17

3.1 Capturing Face Image using 3 Different Cameras 20
3.2 p-Value Comparison for S_1 28
3.3 p-Value Comparison for S_2 28
3.4 p-Value Comparison for S_3 29
3.5 p-Value Comparison for S_4 29
3.6 p-Value Comparison for S_5 30
3.7 p-Value Comparison for S_6 34
3.8 p-Value Comparison for S_7 35
3.9 p-Value Comparison for S_8 35
3.10 p-Value Comparison for S_9 36
3.11 p-Value Comparison for S_{10} 36

List of Tables

3.1 Train Set, Validation Set & Test Set Distribution (Chokepoint Dataset) 21
3.2 Train Set, Validation Set & Test Set Distribution (Custom Dataset) 22
3.3 Average Recognition Rate using Gaussian Function (Chokepoint Dataset) 24
3.4 Macro Average using Gaussian Function (Chokepoint Dataset) . . . 24
3.5 p-Value for Gaussian Function (Chokepoint Dataset) 25
3.6 Average Recognition Rate using Triangle Function (Chokepoint Dataset) 25
3.7 Macro Average using Triangle Function (Chokepoint Dataset) . . . 25
3.8 p-Value for Triangle Function (Chokepoint Dataset) 26
3.9 Average Recognition Rate using Trapezoidal Function (Chokepoint Dataset) 26
3.10 Macro Average using Trapezoidal Function (Chokepoint Dataset) . 27
3.11 p-Value for Trapezoidal Function (Chokepoint Dataset) 27
3.12 Average Recognition using Gaussian Function (Custom Dataset) . . 31
3.13 Macro Average using Gaussian Function (Custom Dataset) 31
3.14 p-Value for Gaussian Function based Proposed Method (Custom Dataset) 32
3.15 Average Recognition using Triangle Function (Custom Dataset) . . 32
3.16 Macro Average using Triangle Function (Custom Dataset) 32
3.17 p-Value for Triangle Function based Proposed Method (Custom Dataset) 33
3.18 Average Recognition using Trapezoidal Function (Custom Dataset) 33
3.19 Macro Average using Trapezoidal Function (Custom Dataset) . . . 34
3.20 p-Value for Trapezoidal Function based Proposed Method 34

List of Abbreviations

C

CNN	**C**onvolutional **N**eural **N**etwork

D

DCT	**D**iscrete **C**osine **T**ransform
DL	**D**eep **L**earning

F

FN	**F**alse **N**egative
FP	**F**alse **P**ositive

I

ICA	**I**ndependent **C**omponent **A**nalysis

L

LDA	**L**inear **D**iscriminant **A**nalysis
LTFT	**L**ong-**T**erm **F**ace **T**racking for **C**rowded **V**ideo-**S**urveillance **S**cenarios

M

ML	**M**achine **L**earning
MTCNN	**M**ulti-**T**ask **C**ascaded **C**onvolutional **N**eural **N**etworks

P

PCA	**P**rincipal **C**omponent **A**nalysis

R

ReLU	**R**ectified **L**inear **U**nit

S

SVM	**S**upport **V**ector **M**achine
T	
TN	**T**rue **N**egative
TP	**T**rue **P**ositive

Abstract

This thesis describes Face recognition using CNN classifiers and final ranking for class predicted by the proposed method is done using rank-level fusion. This thesis works on multi instance of individual faces, multi algorithms based on probability density approaches and multiple classifiers. Fuzzy ranking is generated by outputs (confidence factors) of classifier using the density functions based formulation. VGG16, InceptionResnetV2 and Xception are used as CNN classifiers. For mathematical formulation used three different probability density functions are Gaussian Density Function, Triangle Density function and Trapezoidal Density function. Rank-level fusion is done to generate final ranking. Later, comparison between three density functions are implemented. The proposed method is successfully tested two datasets. One is Chokepoint dataset and another Custom dataset. The datasets are created by cropping faces from videos using MTCNN and OpenCV. The rank-level fusion is performing better than used three CNN models according to accuracy. The accuracy scores for Chokepoint dataset corresponding to Gaussian density function, Triangle Density function and Trapezoidal Density Function are 99.802%, 99.904% and 99.896% respectively and accuracy scores for Custom dataset are 99.762%, 99.716% and 99.75% respectively. It shows a supremacy regarding to face recognition.

CHAPTER 1

Introduction

Face recognition is one the most successful applications of image analysis, a special part of Computer Vision and Pattern Recognition. So it is very interesting field for researchers. As a result, face recognition has received a magnificent attention over last three decades. Face recognition is a biometric technology that works using feature extraction of human face to identify person's identity. Analyzing unique pattern and structure of face such as eye, nose, lips etc are the main thing in face recognition. Individuals can be uniquely identified and authenticated using biometric systems thanks to their distinct biological characteristics. Unimodal and multimodal biometrics are the two primary categories into which these systems can be divided. In this thesis, the concept of unimodal biometric is used. Working principle of this thesis is based on multi instance of human face, multi algorithm and multi classifier. Three CNN classifiers and three different density functions are used. The three CNN models are VGG16 [1], InceptionResNetV2 [2] and Xception [3]. The used density functions are Gaussian Density function [4], Triangle Density function [5] and Trapezoidal Density function [6]. Rank-level Fusion is done using the density functions on the output of CNN classifiers.

1.1 Literature Survey

This thesis is based on Sing et al. [7]. Fusion can be carried out in two levels - 1) fusion before matching and 2) fusion after matching. [8]–[10]

1.1.1 Fusion Before Matching

For fusion before matching, the process of fusion takes place before any comparison or matching between various datasets or sources. Firstly, information are gathered from multiple sources. In this fusion process, the data from multiple sources are being combined to single cohesive representation. A consolidated

dataset is created by merging data or aggregating data or any other way. The data has been fused and can be given to a classifier or matcher. Example of fusion before matching - i) sensor-level fusion and ii) feature-level fusion.

In the sensor-level fusion approach, raw data from multiple sensors are integrated to generate new data for feature extraction and classifier or matcher. In feature-level fusion, fusion is done between feature vectors on single or multiple dataset to produce a new feature vector.

1.1.2 Fusion After Matching

Fusion after matching means that fusion is done after any comparison, classifying or matching on datasets. Fusion is done on basis of output of classifier or matcher. The example of fusion after matching method are - i) score-level fusion, ii) rank-level fusion and iii) decision-level fusion.

Score-level fusion means that the output or score of multiple classifiers are combined or integrated to generate a single score for final prediction [11]. After comparing of implemented the test sample with the templates according to each matcher, rank level fusion is carried out. Each potential template in the database has a corresponding rank for each testing sample and each unique matcher. The test sample that most closely matches the matching template is represented by the lowest rank [12]. In decision-level fusion, decisions of multiple classifiers or matcher are combined to generate a new decision. Decision-level fusion integrates decisions different modalities of individual for decision. [13]

1.2 Face recognition frameworks

Methods proposed for face recognition fall into a wide range of scientific fields, making it challenging to define a precise boundary that would allow for a standard classification of these techniques. Authors of [14] proposed two categories of face recognition - i) Classical approach and ii) Modern approach.

1.2.1 Classical Approaches

Face recognition techniques classified as classical were created prior to the development of deep learning and contemporary machine learning techniques. These methods mostly rely on image processing and statistical analysis. Holistic methods, Local feature based methods and hybrid models are the part of classical methods. The term holistic method refers to methods that use global features to identify faces by looking at the full representation of the face rather than just

its individual parts, such as the mouth, eyes, nose, etc Sirovich et al. [15] Local feature based approaches were evaluated by distinguishing facial regions like mouth, nose, eyes etc. and finding the geometrical relationship among them. Kanade et al. [16] used local feature based face recognition with linear feature matching scheme[17]. Hybrid version of both feature based face recognition and holistic methods is the third category of classical approaches. Hybrid method is used to take both advantage for local and global methods [18]. Authors of [19] described the comparison between four major classical methods Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Discrete Cosine Transform (DCT) and Independent Component Analysis (ICA). PCA is used for feature extraction and PCA can do dimension reduction [20]. LDA searches the vital vectors which may separate the classes better like SVM (Support Vector Machine) [21]. Energy compaction property is used in DCT. Finite number of data points can be expressed using DCT in term of sum cosine function [22]. ICA is used to decompose the signal observed from the various sources into a linear combination of unknown signals that are independent of each other [23].

1.2.2 Modern Approaches

Deep learning and neural network improvements have completely changed how faces are recognized in modern applications. These approaches perform much better in terms of accuracy, robustness, and scalability than classical methods. Deep learning based methods are more significant and accurate over traditional machine learning methods [24]. Many deep learning model like VGG16, VGG19, Resnet50 etc have achieve better performance than other classical methods. Face recognition systems can benefit from the effective application of fuzzy logic to manage the inherent uncertainties and variations in facial features. Face recognition is the process of recognizing or authenticating a person from a still image or motion picture. Fuzzy logic can improve performance at various points in the process [25].

This thesis is mainly working on the basis of modern approaches. As, it uses CNNs and fuzzy logic.

1.3 CNN Based Approaches

Now a days, deep learning is one the most interesting subject among researchers. Face recognition using deep learning has become more significant. VGG16, InceptionResNetV2 and Xception are used in thesis. In this section some CNN models for recognition is discussed.

1.3.1 VGG16

VGG16 is a CNN architecture which was proposed by Visual Geometry Group (VGG) at the University of Oxford [1]. VGG16 consists of 16 layers - 13 convolutional layers and 3 fully connected dense layers. These layers are organized into blocks, with each block containing multiple convolutional layers followed by a max-pooling layer for downsampling.

1.3.2 VGG19

VGG19 is improved version of VGG16. In VGG19, there are 19 layers. Out of them 16 layers are convolutional layers and 3 are fully connected dense layer [26]. As VGG16, blocks are organized and max-pooling layer for downsampling.

1.3.3 AlexNet

AlexNet is a CNN architecture which has 8 layers out of that 5 layers are convolutional layers and last three are fully connected dense layer. For the first two convolutional layer, there are max pooling layers are present after each of them. Last three convolutional layers are consecutively used and then a max-pooling is done. In every layer, ReLU is used as the activation function [27].

1.3.4 Inception-v3

Inception-v3 is a deep convolutional neural network consisting of 48 layers. Inception-v3 is trained on ImageNet and the error rate for top-5 is 3.5% and also the error rate for top-1 is 17.3% in the time of identification of 1000 classes [28].

1.3.5 Resnet-50

Resnet-50 is a CNN architecture which has 50 layers. In this 50 layers, 48 layers are convolutional layers and 1 layer is for max-pooling and another layer is for average pooling. Lastly, after flattening, fully conected dense layer are there [29].

1.3.6 InceptionResNetV2

Consisting of 164 layers deep convolutional neural network, InceptionResNetV2 is trained using ImageNet dataset over a million of images of 1000 classes. InceptionResNetV2 is a combination of Inception-v3 and Resnet [2].

1.3.7 Xception

Xception is also a deep CNN consisting of 71 layers. Xception is trained over a million of images of ImageNet dataset. Xception is a deep convolutional neural network architecture that involves Depthwise Separable Convolutions. Xception provides an architecture consisting of Maxpooling and Depthwise Separable Convolution blocks [3].

1.4 Activation Functions

Activation functions play a crucial role in the time of introducing non-linearities to the computations of CNNs. They are mainly applied after convolutional layers or fully dense layers. Activation functions introduce non-linear changes that enable CNNs to describe and depict decision boundaries and non-linear data distributions.

1.4.1 ReLU

ReLU stands for Rectified Linear Unit. ReLU is defined as $f(x) = max(0, x)$ where x is the input. It sets negative value as zero and keeps positive as unchanged. ReLU reduces the vanishing gradient issue and is computationally efficient. [30]

1.4.2 Leaky ReLU

Leaky ReLU is modified version of ReLU. It is defined as $f(x) = max(ax, x)$ where x is the input and a is small value (typically positive). It keeps the computational efficiency of ReLU. [31]

1.4.3 tanh

The tanh activation function is defined as $f(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)}$ where x is the input. This function maps the input in the range [-1,1]. This activation function has center at 0, so it is more suitable for zero-centered data. [32]

1.4.4 Sigmoid

The Sigmoid activation function is defined as $f(x) = \frac{\exp(x)}{\exp(x)+1}$ where x is the input. This function maps the input in the range range of 0 to 1. This activation function gives output like probability values. It is suitable for binary classification problems. [33]

1.4.5 SoftMax

The SoftMax activation function is typically used in the output layer of CNNs for multi-class classification problems.SoftMax maps the input to a probability distribution over multiple classes, where the sum of all probabilities is 1. Given an input vector $x = [x_1, x_2, ..., x_N]$ of length N, the SoftMax function is defined as: $SoftMax(x) = \frac{\exp(i)}{\sum_{j=1}^{N} \exp(j)}$ where $i = 1, 2, 3, ..., N$.

For each element x_i in the input vector x, SoftMax computes the exponential of x_i divided by the sum of the exponentials of all elements in x. It allows the network to provide class probabilities for each input sample. [34]

1.5 Rank-level Fusion Approaches

Rank-level fusion is a popular concept for the the field of information retrieval and biometric system. Authors of [35] explained some improved rank level-fusion. Some rank-level fusions are briefly described here.

Let, T classifiers produce T outputs according to highest rank and $r_{i,j}$ is assigned rank for class j, where $i = 1, 2, ..., T$ and $j = 1, 2, ..., P$ (P is number of total classes) and R_j is the final fused ranking.

1.5.1 Highest Rank method

Highest rank method utilizes the highest ranked outputs of different classifiers. Let, T classifiers produce T outputs according to highest rank. This approach is particularly conservative and is used when the aim is to ensure that only those items or individuals that are consistently ranked high across all systems are chosen [36].

$$R_j = \min_{i=1}^{T} r_{i,j} \tag{1.1}$$

1.5.2 Modified Highest Rank method

Modified highest rank is proposed by the authors [35] by using a perturbation factor, ϵ_j imposed for corresponding class j in time final ranking.

$$R_j = \min_{i=1}^{T} r_{i,j} + \epsilon_j \tag{1.2}$$

where,

$$\epsilon_j = \frac{\sum_{i=1}^{T} r_{i,j}}{V} \tag{1.3}$$

A large value is assigned to V.

1.5.3 Borda Count

Borda count method uses all the ranks given by output classifiers. All ranks are summed up to generate final ranking. [37]

$$R_j = \sum_{i=1}^{T} r_{i,j} \tag{1.4}$$

1.5.4 Weighted Borda Count

Weighted Borda Count is improved method of Borda Count. It is calculated by the summation of multiplying a weight w_j corresponding to a class j with $r_{i,j}$. [38]

$$R_j = \sum_{i=1}^{T} w_j \times r_{i,j} \tag{1.5}$$

1.5.5 Nonlinear weighted rank fusion method

Kumar et al. [39] proposed a nonlinear weighted rank fusion by using exponential. The rank is calculated by the weighted sum of class j.

$$R_j = \sum_{i=1}^{T} w_j \times \exp(r_{i,j}) \tag{1.6}$$

CHAPTER 2

Proposed Method

This portion involves the steps in designing the proposed system using CNN rank-level fusion method. This method is induced by three different density functions. They are as follows - **1) Gaussian density function**, **2) Triangle density function** and 3) **Trapezoidal density function**. The goal is to find fused ranking by utilizing different outputs (confidence factors) given by different CNN based model classifiers. The final ranking for classification and recognition is calculated by applying the weighted fused ranks of the confidence factors from multiple classifiers that were trained using distinct feature vectors. Three different CNN classifiers are used to extract features from a face image. The purpose of these feature vectors is to generate a list of k classes based on the "top k" ranking generated by "SoftMax" activation functions in parallel. Ranking is done by taking complement of "top k" classes using the above said three density functions (Gaussian, Triangle and Trapezoidal density functions) that produces rank in range [0,1]. Classifier's outputs regarded as confidence factor (CF) is the base of rank generation. This ranking calculates how much the rank is close to classifier's output. In ideal case the rank is 0, which refers to the highest rank (Rank 1) in conventional ranking. After that the fuzzy ranks and confidence factors corresponding to a particular class from different classifiers are combined. Cost has been imposed for fuzzy rank and confidence factor of particular class k if it does not appear in "top k" classes of a classifier. Lower value of rank sum and confidence factor sum are indicating the probable winner. Confidence factor sum is complemented after taking the average of confidence factors given by different classifiers. Final score corresponding to a class k is calculated by multiplying the rank sum and confidence factor sum against that class. The class which has the smallest score will be defined as winner. The workflow for the proposed method is shown in .Here, 3 different CNN based classifiers are used for classification from face images. The CNN based classifiers are **1) VGG16**, **2) InceptionResnetV2** and **3) Xception**. "SoftMax" activation function is used for every classifier.

2.1 Face Cropping using MTCNN & OpenCV

MTCNN and OpenCV are used to crop faces from videos. A video is a collection of thousand images. An instance of the MTCNN "face_detector" has been created. Video has been read frame by frame using OpenCV. The instance of MTCNN face detector detects the faces for a particular frame in a certain time.

2.1.1 MTCNN

"MTCNN" stands for Multi-Task Cascaded Convolutional Networks. Zhang et al. [40] proposed MTCNN which is a combination of three stages of convolutional networks. The main usage of MTCNN is face detection, bounding box regression and facial landmark localization. The three stages of convolutional networks are described below -

1. **Proposal Network (P-Net) :**
   The first stage of MTCNN is P-Net which is responsible for generating candidate windows of potential faces. It applies a window sliding over the input image and uses convolutional neural networks to predicate potential face. It also predicts bounding box and proposes facial landmark networks.

2. **Refinement Network (R-Net) :**
   The potential faces which are generated by P-Net are refined under R-Net. It takes the candidate windows containing potential faces and applies more filters for refinement. It reduces False Negative windows.

3. **Output Network (O-Net) :**
   The candidate windows generated by R-Net are refined for the last time under O-Net. It generates the final windows containing faces by further refining the facial landmark locations and bounding box predictions.

The detected portion of face is cropped using OpenCV and resized the cropped image to the dimension (100, 100). Figure 2.1 shows an example of cropping faces from a frame of video. The cropped face images are stored locally. After that dataset is created with a set of classes containing face images with faces of particular person in each class. This dataset is distributed in train set, validation set and test set in future.



Figure 2.1: Cropping Face from a Video using MTCNN

2.2 CNN Based Classifiers

As said before three different CNNs are used for feature extraction. After that "SoftMax" activation function is used on all of that feature vectors. Brief description of these feature extraction methods are presented below -

2.2.1 VGG16

VGG-16 consists of 16 layers total — 13 Convolutional layers and 3 fully dense layers. These layers are organized into blocks, with each block containing multiple convolutional layers followed by a max-pooling layer for downsampling. At last "SoftMax" is used in output layer. VGG-16 is well known for its efficiency and simplicity, as well as for its ability to perform well on a variety of computer vision tasks, such as object recognition and image categorization.

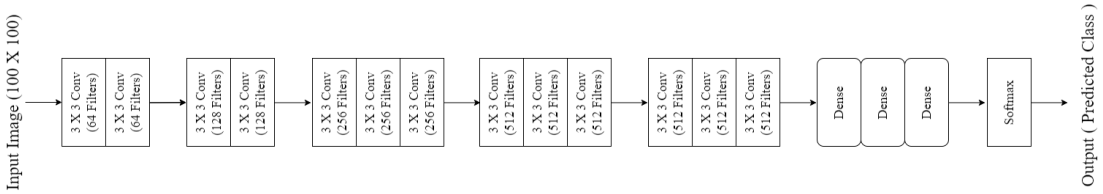


Figure 2.2: VGG16 Architecture

VGG-16 takes images of size $100 \times 100 \times 3$ for this process. The image is

100×100 with 3 channels - RGB. After that images are gone through two consecutive convolutional layers with 64 filters each, using 3x3 filters. Padding is applied for both convolutions to maintain spatial dimensions. The output shape after these two convolutional is $100 \times 100 \times 64$. Max pooling layer is used after that which kernel size is 2×2 with stride 2. This produces a output with shape $50 \times 50 \times 64$. Now, another two consecutive convolutional layers are applied with 128 filters each using 3×3 filters. Once again, max pooling is done with 2×2 kernel having stride 2 and new shape of output is $25 \times 25 \times 128$. Instead of two, three consecutive convolutional layers are used with 256 filters and kernel size of 3×3 and padding is given to maintain dimension. After these 3 convolutional layers, max pooling is applied with a 2×2 kernel, having stride as 2. The shape of new output is $12 \times 12 \times 256$. Another three consecutive convolutional layers are applied on new output with 3×3 kernel having same padding with 512 filters. After max pooling with same kernel size and stride, the shape of output becomes $6 \times 6 \times 512$. Last three convolutional layers having 512 filters are applied consecutively and max pooling is done after that with same kernel size 2×2 and stride 2. At last, the output shape will be $3 \times 3 \times 512$. Global Average pooling is done and new shape is 512. Now, a fully connected dense layer with 1024 units and ReLU as activation function. Another fully connected dense layer having number of classes for images and "SoftMax" activation function.

2.2.2 InceptionResnetV2

InceptionResNetV2 consists of 164 layers deep convolutional neural networks. InceptionResNetV2 is a combination of Inception-v3 and Resnet.

The input for InceptionResnetV2 is same as for VGG16 with size $100 \times 100 \times 3$. Initially, the image is gone through under stem module for initial feature extraction and it maintains the spatial dimension of image. The inception modules and residual connections are combined. This is based on conception of full convolutional layers. Reduction modules are used to improve the feature maps' depth while downsampling their spatial dimensions. ReLU is used as activation function. At last output shape of the train image will be $1 \times 1 \times 1536$. After that Global Average pooling is done and the shape will be 1536. Now, a fully connected dense layer with 1024 units and ReLU as activation function. Another fully connected dense layer having number of classes for images and "SoftMax" activation function.

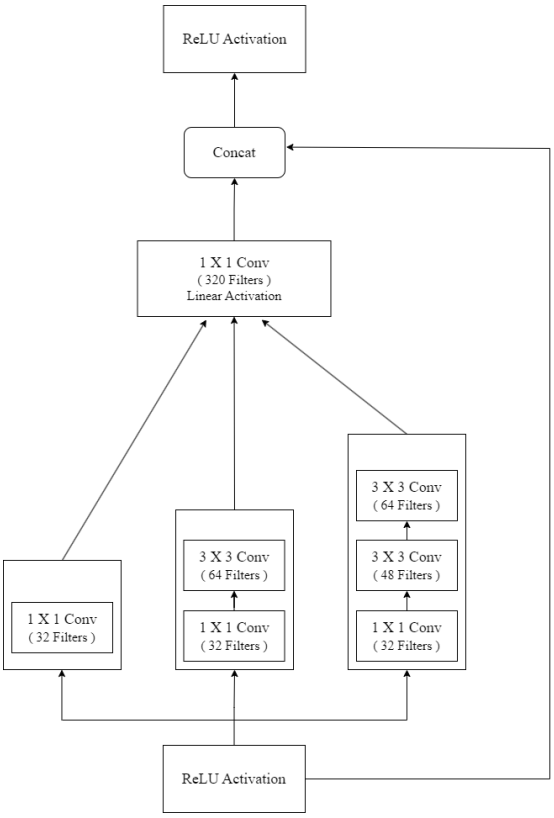


Figure 2.3: InceptionResnetV2 Architecture

2.2.3 Xception

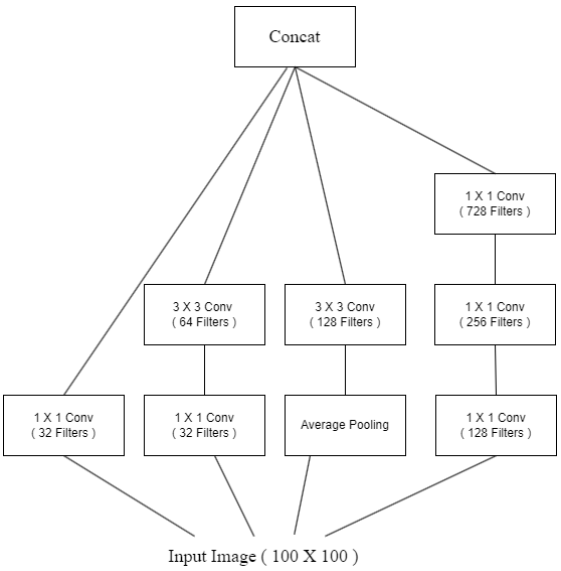


Figure 2.4: Xception Architecture

Xception is a deep convolutional neural network architecture that involves Depthwise Separable Convolutions. Xception provides an architecture consisting of Maxpooling and Depthwise Separable convolutional blocks. Xception consists

of two different levels. Three convolutional layers are present in each of the two levels of this model. The output is divided into three segments by this layer before being sent to the following set of filters.

A image which has dimension of (100 X 100) is given as the input of Xception. ReLU is used as activation function. Atlast, after global average pooling the output shape will be 2048. In that time flatten is done and "SoftMax" is used as activation function.

2.3 Mathematical Formulation

This section refers to a mathematical formulation to the proposed method. The proposed fusion method has been applied for face recognition using feature vector and classifier. Let, N be the number of different CNN models. Let, generated "SoftMax" output vector from these N models are (SPV_1, SPV_2, SPV_3, ... , SPV_N) corresponding to face image X. In this case, the value of N is 3. As "top k" classes are selected and here K=5. The maximum number of predicted classes in output vector in all combined N models is N*K (here, 3*5=15). This will happen when all N models predicted different top K classes. If all N models predict the same top K classes then number of predicted classes after combining is k. Let, predicted classes be ($CL_1^{N_1}$, $CL_2^{N_1}$, $CL_3^{N_1}$, ..., $CL_K^{N_1}$) with confidence factor be (${CF_1}^{N_1}$, $CF_2^{N_1}$, ${CF_3}^{N_1}$, ..., $CF_k^{N_1}$) where K denotes top selected distinct classes respectively. Where, $CL_i^{N_j}$ and $CF_i^{N_j}$ denotes predicted class i of j-th model and its confidence factor. Similarly we have ($CL_1^{N_2}$, $CL_2^{N_2}$, $CL_3^{N_2}$, ..., $CL_K^{N_2}$) and ($CF_1^{N_2}$, $CF_2^{N_2}$, $CF_3^{N_2}$, ..., $CF_K^{N_2}$) etc.
Confidence factors are being normalized with this following function -

$$\sum_{k=1}^{K} CF_k^{N_i} = 1 \ \forall \quad i, i = 1, 2, 3, ..., N \tag{2.1}$$

The fuzzy rank corresponding to a class k, $RV_k^{N_i}$ using confidence factor is calculated. $RV_k^{N_i}$ is calculated using three different probability density functions. They are **1) Gaussian Density Function**, **2) Triangle Density Function** and **3) Trapezoidal Density Function**. The detailed formulation is explained in subsection 2.3.1, subsection 2.3.2 and subsection 2.3.3.

Let, K^{N_i} be the "top k" selected classes generated by N_i model. The complement of average of the cumulative sum of confidence factors of any class k is denoted as CFS_k and cumulative rank sum of any class k is denoted as FRS_k.

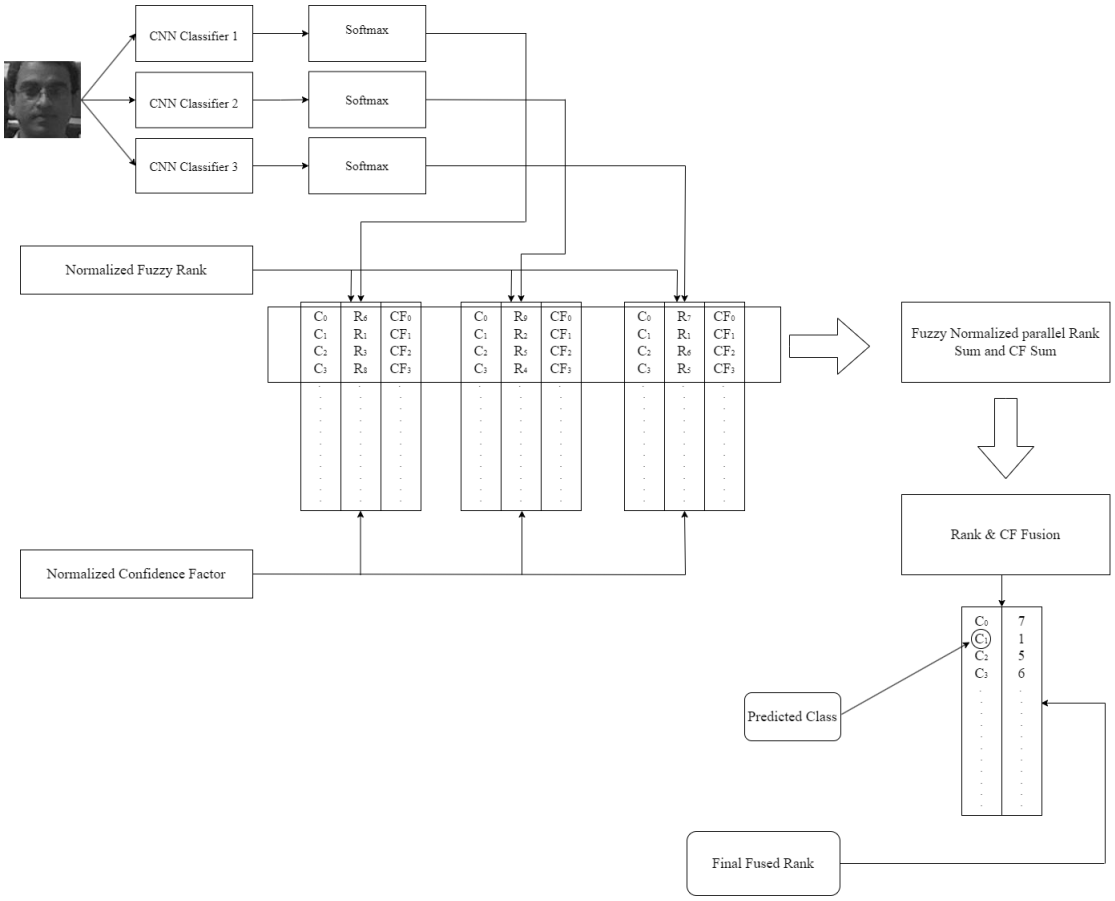


Figure 2.5: Workflow of Proposed Method

Both are initialized to 0 for all instance.

$$FRS_k = \sum_{i=1}^{N} RV_k^{N_i} \tag{2.2}$$

$$CFS_k = 1 - \frac{1}{N} \sum_{i=1}^{N} CF_k^{N_i} \tag{2.3}$$

The final score of any class k, FS_k is calculated by multiplying FRS_k and CFS_k. The formula as follows -

$$FS_k = FRS_k \times CFS_k \tag{2.4}$$

The predicted class of the proposed method will be k for which FS_k will have the smallest value.

Let, predicted class will be PC_X for a test image X.

$$PC_X = \arg \min_{i=1,2,\ldots,k} FS_i \tag{2.5}$$

Figure 2.5 shows the total workflow of the proposed method.

2.3.1 Gaussian Density Function

At first, $RV_k^{N_i}$ is calculated by the complement of Gaussian Density Function Equation 2.6 -

$$RV_k^{N_i} = (1 - e^{\frac{-(CF_k^{N_i} - 1.0)^2}{(2 \times 1.0)}}) \tag{2.6}$$

where, $i = 1, 2, 3, ..., N$ and $k = 1, 2, 3, ..., K$ in Equation 2.6.
$RV_k^{N_i}$ lies between [0,1]. In ideal case $RV_k^{N_i}$ will be 0 and selected as rank 1 (top rank) in conventional ranking.

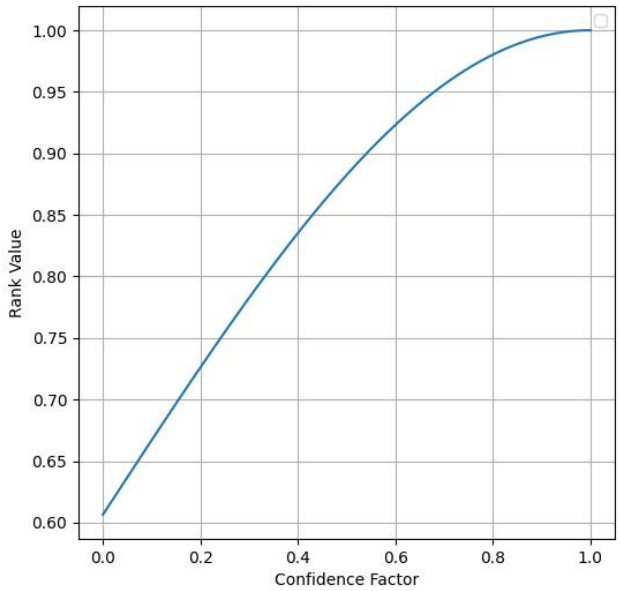


Figure 2.6: Gaussian Density Function

The value of $RV_k^{N_i}$ is calculated using Equation 2.6 if $RV_k^{N_i}$ belongs to K^{N_i} and the value will be 0.394 as putting the value $CF_k^{N_i} = 0$ in the Equation 2.6 if it does not appear in K^{N_i}. The cost value 0.394 is imposed when a class is present in a set, generated from "top k" class, but not present in another set. This costs eliminate the chance that class k will emerge as the unlikely winner.
Afterwards, FRS_k and CFS_k are calculated using Equation 2.2 and Equation 2.3 respectively.

The last step of the proposed method, calculating the final score of any class k, FS_k is calculated using Equation 2.4 and class k is predicted for any image X using Equation 2.5.

2.3.2 Triangle Density Function

The fuzzy rank corresponding to a class k, $RV_k^{N_i}$ using confidence factor is generated by the complement of Triangle Density Function :

$$RV_k^{N_i} = \begin{cases} (1 - \frac{CF_k^{N_i} - a}{b - a}) & \text{if } a \leq CF_k^{N_i} \leq b \\ 1 & \text{otherwise} \end{cases} \tag{2.7}$$

where, $i = 1, 2, 3, ..., N$ and $k = 1, 2, 3, ..., K$ in Equation 2.7.

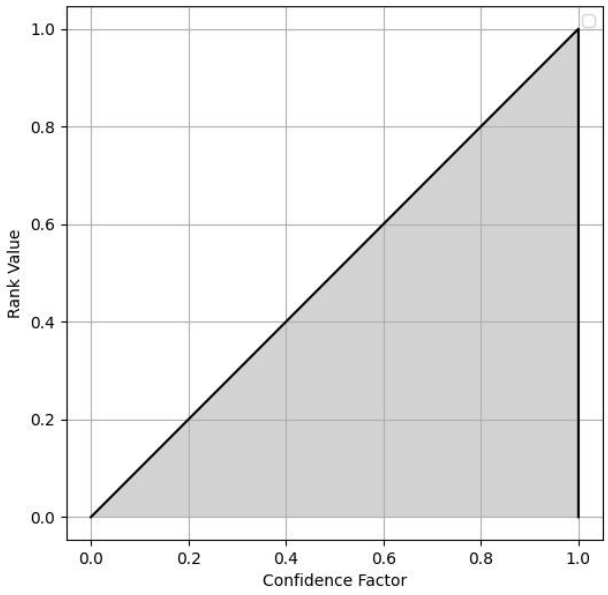


Figure 2.7: Triangle Density Function

a is the minimum value and b is the maximum value of confidence factor of "top k" selected classes by N_i. As same with Gaussian Density Function here also $RV_k^{N_i}$ lies between [0,1]. In ideal case $RV_k^{N_i}$ will be 0 and selected as rank 1 (top rank) in conventional ranking.

In these cases CF_k always lies between a and b. This function is quiet similar to min-max normalization. The values of a and b are as defined :

$$a = \min(CF_k^{N_i}) \tag{2.8}$$

$$b = \max(CF_k^{N_i}) \tag{2.9}$$

where $k = 1, 2, 3, ..., K$ and the values of a and b are calculated for every $N_i \ \forall \ i, \ i = 1, 2, 3, ..., N$

FRS_k and CFS_k are also calculated here as using Equation 2.2 and Equation 2.3. For Triangle Density function while calculating FRS_k the value of $RV_k^{N_i}$ is calculated using the first condition of Equation 2.7 if $RV_k^{N_i}$ belongs to K^{N_i} and the value will be 1 as per the second condition of Equation 2.7 if the that class k does not appear in K^{N_i}.

$CF_k^{N_i}$ will be 0 when the class k does not appear in the "top k" class in Equation 2.3. This cost which is imposed for not present in certain set may lead elimination of class k as the unlikely winner.

The final score of any class k, FS_k is calculated as given in Equation 2.4 and after this class based on proposed fusion method is predicted using Equation 2.5.

2.3.3 Trapezoidal Density Function

$RV_k^{N_i}$ is generated by the complement of Trapezoidal Density Function :

$$RV_k^{N_i} = \begin{cases} (1 - \frac{CF_k^{N_i} - a}{b-a}) & \text{if } a \leq CF_k^{N_i} \leq b \\ 0 & \text{if } b \leq CF_k^{N_i} \leq c \\ 1 & \text{otherwise} \end{cases} \tag{2.10}$$

where, $i = 1, 2, 3, ..., N$ and $k = 1, 2, 3, ..., K$ in Equation 2.10.

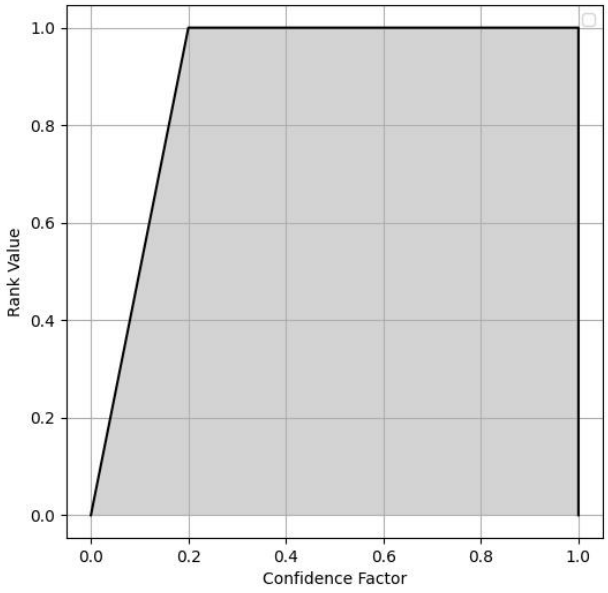


Figure 2.8: Trapezoidal Density Function

a is the minimum value and c is the maximum value of confidence factor of "top k" selected classes by N_i. b is the mean value of these confidence factors of selected classes. As same with Gaussian Density Function and Triangle Density Function here also $RV_k^{N_i}$ lies between [0,1]. In ideal case $RV_k^{N_i}$ will be 0 and selected as rank 1 (top rank) in conventional ranking.

The values of a and c are as defined :

$$a = \min(CF_k^{N_i}) \tag{2.11}$$

$$c = \max(CF_k^{N_i}) \tag{2.12}$$

where $k = 1, 2, 3, ..., K$ and the values of a and c are calculated for every N_i $\forall\ i,\ i = 1, 2, 3, ..., N$

To calculate b (mean of confidence factors) first of all sum of confidence factors is calculated -

$$SCF = \sum_{k=1}^{K}(CF_k^{N_i}) \tag{2.13}$$

where, SCF is the sum of confidence factors and SCF is calculated for every N_i $\forall\ i,\ i = 1, 2, 3, ..., N$.

Now, SCF is divided by number of top k selected classes -

$$b = \frac{SCF}{N^k} \tag{2.14}$$

Let, K^{N_i} be the "top k" selected classes generated by N_i model and N^k is the number of selected "top k" class. FRS_k and CFS_k are calculated using Equation 2.2 and Equation 2.3 respectively.

Here, $RV_k^{N_i}$ is calculated using the Equation 2.10 function. If $RV_k^{N_i}$ belongs K^{N_i} then first conditions are used of Equation 2.10 function and if it does not belong to K^{N_i}, then $RV_k^{N_i} = 1$ and $CF_k^{N_i} = 0$. This costs eliminate the chance that class k will emerge as the unlikely winner.

The final score of any class k, FS_k is calculated by multiplying FRS_k and CFS_k as given in Equation 2.4 and then predicted class is decided using Equation 2.5.

CHAPTER 3

Experiment and Results

A sample of tested video sequences demonstrating the effectiveness of each step in the proposed face recognition system are presented in this section. These video clips function as essential assessment benchmarks, enabling a comprehensive analysis for performance of the model in various contexts. These presentations provide a detailed illustration and analysis of the efficacy of face recognition.

The proposed model may be thoroughly assessed by utilizing a variety of testing settings, guaranteeing its resilience and adaptability in a variety of real-world scenarios. This in-depth evaluation helps to clarify the system's advantages and disadvantages, which in turn aids in improving and optimizing it.

3.1 Dataset

To evaluate the proposed fusion method, two different datasets are used. To validate fusion method we have used a standard dataset which is a publicly available, namely, Chokepoint dataset and another is custom dataset. The Chokepoint dataset is utilized to assess designs for trials in person identification and verification under actual surveillance situations. This custom dataset is formed from a publicly available dataset, namely, LTFT Dataset. MTCNN is used to crop the faces from the available videos.

3.1.1 Chokepoint Dataset

Chokepoint Dataset [41] is used as the standard dataset for training, validation and testing purpose to evaluate the said fusion method. ChokePoint, a video dataset, has been created to study how well computers can recognize people in busy places like hallways or doorways. To form this dataset three cameras were

set up to capture people walking through naturally. Each camera gives a different view, a variety of images were being collected which showing people's faces. Sometimes the lighting might be different, or people might be moving quickly, so their faces might look blurry or not facing directly towards the camera.

The ChokePoint dataset includes videos of people passing through two different portals. There are 25 subjects (19 male and 6 female) in portal 1, and 29 subjects (19 male and 6 female) in portal 2. Every video has a resolution of 800×600 pixels and a frame rate of 30 frames per second. There are 64,204 face images total across 48 video sequences. Every sequence begins with one hundred background-modeling frames devoid of any human presence. Notably, every image or frame in the dataset contains a single person.

The dataset follows a specific naming convention for each video sequence, indicating details like which portal, sequence number, and camera were used, as well as whether people were entering (E) or leaving (L) the portal.For example, "P2L_S1_C3" indicates that camera 3 recorded the first recorded sequence of the video, which was shot at Portal 2 while people were exiting the portal. Two sequences (P2E_S5 and P2L_S5) were recorded in crowded conditions with continuous occlusion, where people might block each other in the footage, to increase the difficulty of the surveillance scenarios. This makes things more difficult, like recognizing people and confirming their faces.



Figure 3.1: Capturing Face Image using 3 Different Cameras

For training, validating and testing purpose, 5263 total number of images of portal 1 is used. These 5263 is divided into 25 subjects (19 male and 6 female). These images distribution mentioned in table below :

Table 3.1: Train Set, Validation Set & Test Set Distribution (Chokepoint Dataset)

Dataset	**Number of Images (%)**		
	Train Set	Validation Set	Test Set
Chokepoint Dataset	1568 (30%)	1830 (35%)	1865 (35%)
	2097 (30%)	1568 (40%)	1598 (40%)
	2624 (50%)	1306 (25%)	1336 (25%)
	3147 (60%)	1044 (20%)	1092 (20%)
	3673 (70%)	778 (15%)	812 (15%)

3.1.2 Custom Dataset

To form the custom dataset an YouTube video is used. A publicly available "LTFT" dataset [42] has an YouTube video known as "Street" which has been used to produce this custom dataset.

The "LTFT dataset" stands for "Long-Term Face Tracking for Crowded Video-Surveillance Scenarios" (LTFT). Face tracking is very challenging task in any crowded video-surveillance scenarios because of various factors including occlusions, changes in lighting conditions, and the presence of multiple individuals in the scene.

The LTFT dataset is a long term tracking on many faces over in busy places where you might only see people's faces. Crowded streets or busy events where lots of people are moving around and sometimes blocking each other. This system is made to work even when faces are the only things you can see clearly, dealing with challenges like lots of movement and people getting in each other's way. This system uses a method called tracking-by-detection. It combines two main things: a smart way to reconnect tracks of faces over time using face recognition, and a fast visual tracker to keep track of faces in the short term. Additionally, it has a module that fixes any mistakes in the tracking without needing extra computing power. So basically, it efficiently follows faces over time, even in busy environments, by recognizing faces and quickly adjusting its tracking as needed.

This "Street" video is 1 min 15 seconds long. MTCNN is used to crop the faces from this video. After cropping the faces 9 different classes of different people are created. Out of that 9 classes, 5 classes are with the faces of men and 4 classes are with the faces of women. As the standard dataset is divided into 5 different parts, this dataset is also divided into 5 different parts each consisting of train, validation and test set. The following table is showing distribution of images for train, validation and test set for the 5 different parts -

Table 3.2: Train Set, Validation Set & Test Set Distribution (Custom Dataset)

Dataset	**Number of Images (%)**		
	Train Set	Validation Set	Test Set
Custom Dataset	340 (30%)	402 (35%)	408 (35%)
	456 (30%)	346 (40%)	348 (40%)
	573 (50%)	286 (25%)	291 (25%)
	687 (60%)	230 (20%)	233 (20%)
	802 (70%)	173 (15%)	175 (15%)

3.2 Evaluation Metrics

In this Experimentation, Accuracy Score and Macro Average of F_1 *Score* are two general commonly used methods. F_1 *Score* is one of the most widely used evaluation metrics. F_1 *Score* provides a more balanced evaluation on imbalanced dataset also. F_1 *Score* not only considers True Positives (TP) but also considers both false positives (FP) and false negatives (FN). Firstly we have to compute Precision and Recall. Precision and Recall can be computed using the equation mentioned below -

$$Precision = \frac{TP}{TP + FP} \tag{3.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.2}$$

In above equations, TP, FP, FN represent True Positive (Model has predicted Positive class whether it is actually positive), False Positive (Model has predicted Positive class whether it is actually negative) and False Negative (Model has predicted Negative class whether it is actually positive) respectively. Accuracy Score and F_1 *Score* can be computed using the equation mentioned below -

$$Accuracy\ Score = \frac{TP}{Total\ Number\ of\ Actual\ Positives} \tag{3.3}$$

$$F_1\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.4}$$

$$Macro\ Average\ of\ F_1\ Score = \frac{\sum_{j \in C} F_1\ Score}{Total\ Number\ of\ Classes} \tag{3.5}$$

where, C is set of all classes in Equation 3.5. Accuracy Score, F_1 *Score* and Macro Average of F_1 *Score* lie between [0,1], higher the values show better Recognition has been done.

3.3 Implementation Details

The proposed fusion method is implemented on a resource, including AMD RYZEN 5536 Processor, Windows 11 as Operating System, 8 GB of RAM and 512 GB of storage space, as well as an AMDA RADEON Garphics Card. Python language has been used for the implementation of this model. Here the latest Tensorflow's version (2.15.0) is used, and it is implemented on platforms like *Thonny* and *Visual Studio Code*. Independent stand alone project which can be accessed offline for model training, testing and output generation. For this model, 1×10^{-4} is set as the learning rate. Due to CPU and RAM memory limitations, an Adam optimizer with a batch size of 32 is selected to train the proposed model. The time taken for training of the model is around 1 and half hour for 30 epochs (10 epochs per CNN model). In the proposed model, there are 94 M total network parameters. For a test image, the proposed model takes about 4 minutes on average to produce the predicted class after training.

3.4 Results

Two distinct datasets are used to train and validate the proposed method. - 1) Chokepoint Dataset and 2) Custom Dataset.

3.4.1 Chokepoint Dataset

As above mentioned, Chokepoint Dataset is used as Standard Dataset. It is divided into 5 different parts Table 3.1. Each of the parts are also divided into 5 different sets. This may help to overcome imbalanced data.

Three different probability density functions are used for this proposed method. The Average Recognition rate and Macro Averages of each density functions are shown in below.

1) Gaussian Density Function :

The above said total of 25 sets (5 sets in each 5 parts) have been implemented on the proposed method using Gaussian Density function. The Average Recognition Rate and Macro Average of F_1 Score for each 5 parts Table 3.1 are shown in Table 3.3 and Table 3.4.

Here 30-35-35 (S_1) denotes 30% train images set, 35% validation images set and 35% test images set of total 5263 images. As well as 40-30-30 (S_2), 50-25-25 (S_3), 60-20-20 (S_4), 70-15-15 (S_5) are denoting the same. The distribution of images shown in Table 3.1.

Table 3.3: Average Recognition Rate using Gaussian Function (Chokepoint Dataset)

	Average Recognition Rate				
Model	S_1	S_2	S_3	S_4	S_5
VGG16	0.9748	0.9543	0.9866	0.9715	0.9899
InceptionResNetV2	0.9526	0.9871	0.9949	0.9170	0.9965
Xception	0.8813	0.9567	0.9974	0.9972	0.9978
Proposed Method	**0.99320**	**0.99740**	**0.99970**	**1.00000**	**0.99980**

Table 3.4: Macro Average using Gaussian Function (Chokepoint Dataset)

	Macro Average				
Model	S_1	S_2	S_3	S_4	S_5
VGG16	0.9719	0.9520	0.9865	0.9826	0.9898
InceptionResNetV2	0.9530	0.9878	0.9946	0.9329	0.9966
Xception	0.8816	0.9542	0.9973	0.9973	0.9977
Proposed Method	**0.99330**	**0.99730**	**0.99970**	**1.00000**	**0.99980**

Average Recognition Rate is calculated by taking average of 5 different sets of each of 5 different parts and Macro Average is also calculated by taking Macro of F_1 Score.

These Table 3.3 and Table 3.4 tables are showing that for each and every case the proposed method has been emerged as winner according to Recognition rate and Macro Average.

For higher number of train image, higher Recognition rate and Macro Average have been observed. It is same not only for the proposed method but also the used CNN models.

To evaluate statistically significance of proposed method over 3 used CNN classifiers i.e. - VGG16, InceptionResnetV2 and Xception, paired t-test between proposed method and other three CNN methods are used. The null hypothesis H_0 is defined as: 'the average recognition rate and macro average of the proposed method is same as the compared method' against the alternative hypothesis H_1 as: 'the average recognition rate and macro average of the proposed method is higher as compared to the related method'. The proposed method will be statistically significant over other methods at $p < 0.05$ (5% level of significance). The p-values for proposed methods over Gaussian density functions are calculated.

According to Table 3.5, the proposed method is statistically significant over VGG16 when training set is higher (above 50% of total images) but the proposed method is statistically significant over Xception when training set is lower (below

Table 3.5: p-Value for Gaussian Function (Chokepoint Dataset)

Model	**p-Value**				
	S_1	S_2	S_3	S_4	S_5
VGG16	0.068501	0.100207	**0.012425**	**0.048992**	**0.055784**
InceptionResNetV2	0.100574	**0.042921**	**0.029076**	0.182116	**0.031783**
Xception	**0.013976**	**0.026617**	**0.002460**	0.107181	0.153935

50% of total images). It is obvious that any model will perform better for higher number of training images so the proposed method does not support the condition of t-test over both InceptionResnetV2 and Xception. But the proposed method gives better result over Recognition Rate Table 3.3. Among the used CNN models, VGG16 has given better performance with diversity than two other CNN models.

2) Triangle Density Function

The above said total of 25 sets have been tested on another probability density function which is Triangle Density function. The Average Recognition Rate and Macro Average of F_1 *Score* are as follows -

Table 3.6: Average Recognition Rate using Triangle Function (Chokepoint Dataset)

Model	**Average Recognition Rate**				
	$\boldsymbol{S_1}$	$\boldsymbol{S_2}$	$\boldsymbol{S_3}$	$\boldsymbol{S_4}$	$\boldsymbol{S_5}$
VGG16	0.9821	0.9820	0.9806	0.9960	0.9928
InceptionResNetV2	0.9825	0.9961	0.9974	0.9959	0.9985
Xception	0.8653	0.9851	0.9857	0.9680	0.9970
Proposed Method	**0.99620**	**0.99960**	**0.99950**	**0.99990**	**1.00000**

Table 3.7: Macro Average using Triangle Function (Chokepoint Dataset)

Model	**Macro Average**				
	$\boldsymbol{S_1}$	$\boldsymbol{S_2}$	$\boldsymbol{S_3}$	$\boldsymbol{S_4}$	$\boldsymbol{S_5}$
VGG16	0.9817	0.9800	0.9805	0.9952	0.9928
InceptionResNetV2	0.9825	0.9961	0.9974	0.9927	0.9986
Xception	0.8632	0.9844	0.9853	0.9451	0.9969
Proposed Method	**0.99630**	**0.99960**	**0.99960**	**0.99960**	**1.00000**

These Table 3.6 and Table 3.7 are showing that for each and every case the proposed method has been emerged as winner according to Recognition rate and Macro Average.

As usual for higher number of train image, higher Recognition rate and Macro Average have been observed. It is for the proposed method and the used CNN models both. InceptionResnetV2 has emerged as the best among among three CNN models.

The p-values for proposed methods over Triangle density functions are calculated.

Table 3.8: p-Value for Triangle Function (Chokepoint Dataset)

	p-Value				
Model	S_1	S_2	S_3	S_4	S_5
VGG16	**0.007930**	0.067148	0.067786	**0.012221**	**0.034542**
InceptionResNetV2	**0.000762**	**0.035613**	**0.022719**	**0.049526**	**0.017333**
Xception	**0.011344**	**0.027326**	0.139591	0.074359	**0.040212**

It is noted that the proposed method is totally statistically significant over InceptionResnetV2, Table 3.8.The proposed method has been emerged as statistically significant over all three CNN models for both case of lowest and highest percent of train images. It means that the proposed method using Triangle density function may perform well with low level of data as will as high level of data. The proposed method using Triangle density function is approximately statistically significant over all three CNN models.

3) Trapezoidal Density Function :

Last but not the least the 25 sets have been tested using Trapezoidal Density function. The Average Recognition Rate and Macro Average of F_1 *Score* are as follows -

Table 3.9: Average Recognition Rate using Trapezoidal Function (Chokepoint Dataset)

	Average Recognition Rate				
Model	S_1	S_2	S_3	S_4	S_5
VGG16	0.9689	0.9849	0.9910	0.9888	0.9973
InceptionResNetV2	0.9791	0.9912	0.9971	0.9981	0.9973
Xception	0.9238	0.9838	0.9886	0.9985	0.9943
Proposed Method	**0.99560**	**0.99950**	**0.99970**	**1.00000**	**1.00000**

Table 3.10: Macro Average using Trapezoidal Function (Chokepoint Dataset)

	Macro Average				
Model	S_1	S_2	S_3	S_4	S_5
VGG16	0.9674	0.9836	0.9898	0.9889	0.9973
InceptionResNetV2	0.9789	0.9909	0.9970	0.9981	0.9972
Xception	0.9230	0.9834	0.9883	0.9985	0.9945
Proposed Method	**0.99550**	**0.99940**	**0.99970**	**1.00000**	**1.00000**

These Table 3.9 and Table 3.10 are showing that for each and every case the proposed method has been emerged as winner according to Recognition rate and Macro Average.

For higher number of train image, higher Recognition rate and Macro Average have been observed. It is same not only for the proposed method but also the used CNN models.

The p-values for proposed methods over Trapezoidal density functions are calculated.

Table 3.11: p-Value for Trapezoidal Function (Chokepoint Dataset)

	***p*-Value**				
Model	S_1	S_2	S_3	S_4	S_5
VGG16	**0.043594**	0.071958	0.066047	0.105178	0.137275
InceptionResNetV2	**0.017946**	0.082656	**0.046265**	**0.023722**	**0.048432**
Xception	**0.019822**	**0.004042**	0.160715	0.060683	0.136279

It is noted that the proposed method using Trapezoidal density function is almost statistically significant over InceptionResnetV2 for all cases except one, Table 3.11. The proposed method is statistically significant over all three CNN models when number of train images are least. It means that the proposed method using Trapezoidal density function may perform well with low level of data. It is obvious higher number of data leads to higher recognition rate. The proposed method does not satisfy the condition $p < 0.05$ for higher number of train images but the average recognition rate is highest for the proposed method over all other CNN methods shown in Table 3.9. Gaussian, Triangle and Trapezoidal density functions are compared using these p-values. The graphical view is given in Figure 3.2, Figure 3.3, Figure 3.4, Figure 3.5 and Figure 3.6.

The points which are nearer to the X-axis means that p-values of those are less and the proposed method is more significant over them.

In Figure 3.2, p-values over Xception model is always lesser than 0.02 (2% level

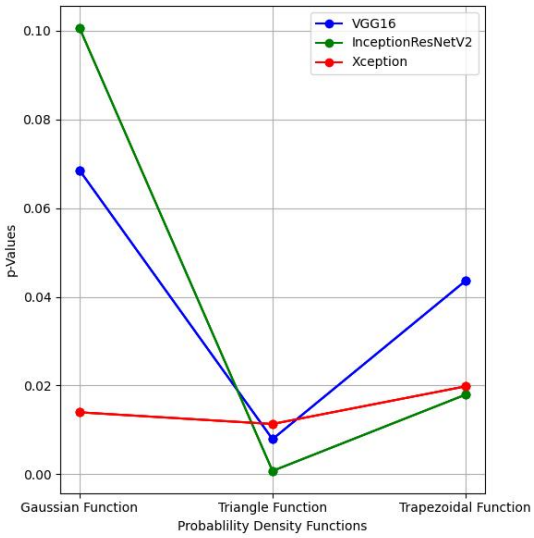


Figure 3.2: p-Value Comparison for S_1

of significance) for all three density functions. That means proposed method is totally statistically significant over Xception in this situation. According to Figure 3.2, proposed method using Triangle density function is statistically significant more than others because the p-values corresponding to all three CNN models are lesser than 0.05.

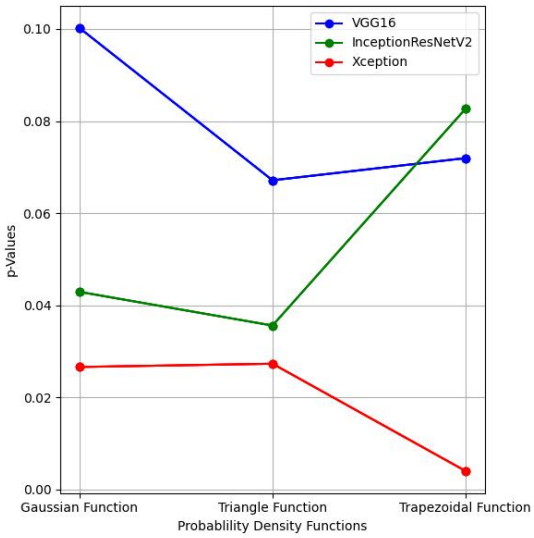


Figure 3.3: p-Value Comparison for S_2

In Figure 3.3, p-values over Xception model is always lesser than 0.03 (3% level of significance) for all three density functions. That means proposed method is totally statistically significant over Xception in this situation. In the time of using Gaussian function and Triangle function, proposed method is statistically significant over InceptionResnetV2 and Xception. But the proposed method is not statistically significant over VGG16, shown in Figure 3.3.

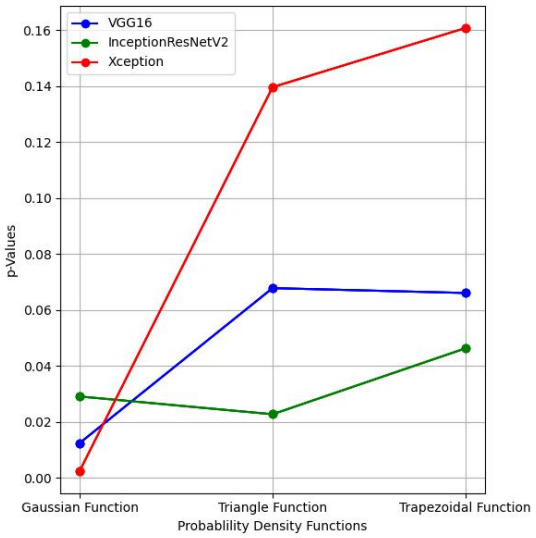


Figure 3.4: p-Value Comparison for S_3

While using Gaussian density function the proposed method is totally statistically significant over all three CNN models and p-value < 0.03 (3% level of significance) which is shown in Figure 3.4. The proposed method is statistically significant over InceptionResnetV2.

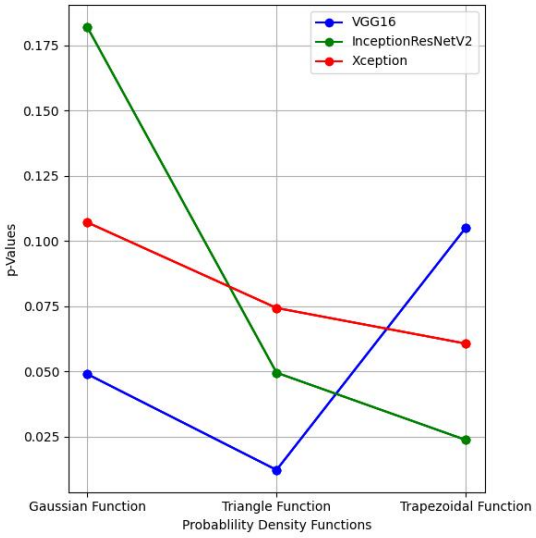


Figure 3.5: p-Value Comparison for S_4

While using Gaussian density function the proposed method is statistically significant over VGG16 only. The proposed method is statistically significant over both VGG16 and InceptionResnetV2 while Triangle density function is used. Trapezoidal function based proposed method is statistically significant over InceptionResnetV2 only which is shown in Figure 3.5.

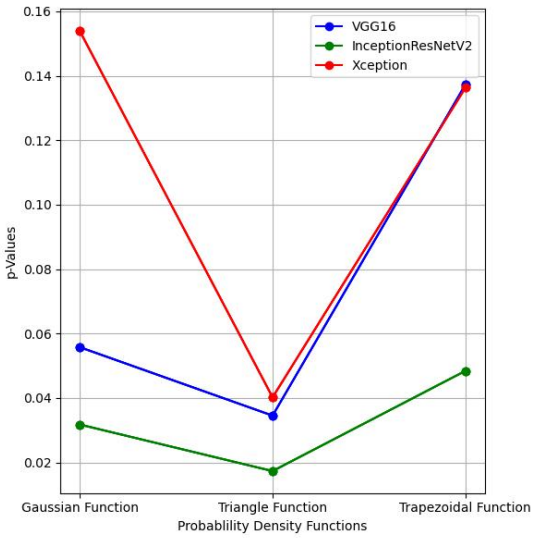


Figure 3.6: p-Value Comparison for S_5

Figure 3.6 describes Gaussian density function based proposed method is statistically significant over VGG16 and InceptionResnetV2 both. The proposed method is statistically significant over all three VGG16, InceptionResnetV2 and Xception when pdf is Triangle density function. Trapezoidal function based proposed method is statistically significant over InceptionResnetV2 only.

According to Figure 3.2, Figure 3.3, Figure 3.4, Figure 3.5 and Figure 3.6, it can be concluded that Triangle density function has emerged as the reason for the proposed method to be the most significant than other density functions used. After triangle density function, Gaussian density function based proposed method is more significant than Trapezoidal density function based proposed method.

3.4.2 Custom Dataset

As said before custom dataset has been produced from "LTFT" dataset using MTCNN and OpenCV subsection 3.4.1. This dataset is also divided into 5 different parts with different percentage of training, validation and test set, shown in Table 3.2. These 5 parts are distributed in 5 different sets. Total of 25 sets are implemented using three density functions - **1) Gaussian Density Function**, **2) Triangle Density Function** and **3) Trapezoidal Density Function**.

1) Gaussian Density Function :

Average Recognition Rate and Macro Average of F_1 *Score* are calculated using Gaussian Density function on the proposed method and those are given below in Table 3.12 and Table 3.13.

Table 3.12: Average Recognition using Gaussian Function (Custom Dataset)

	Average Recognition Rate				
Model	S_6	S_7	S_8	S_9	S_{10}
VGG16	0.9818	0.9845	0.9854	0.9760	0.8057
InceptionResNetV2	0.8118	0.9264	0.9519	0.8884	0.9612
Xception	0.7873	0.7839	0.8725	0.9871	0.9977
Proposed Method	**0.99310**	**0.99710**	**0.99790**	**1.000000**	**1.00000**

Table 3.13: Macro Average using Gaussian Function (Custom Dataset)

	Macro Average				
Model	S_6	S_7	S_8	S_9	S_{10}
VGG16	0.9824	0.9921	0.9871	0.9764	0.7705
InceptionResNetV2	0.8035	0.9242	0.9219	0.8774	0.9622
Xception	0.7324	0.7324	0.8959	0.9840	0.9977
Proposed Method	**0.99340**	**0.99710**	**0.99800**	**1.00000**	**1.00000**

Here 30-35-35 (S_6) denotes 30% train images set, 35% validation images set and 35% test images set of total 5263 images. As well as 40-30-30 (S_7), 50-25-25 (S_8), 60-20-20 (S_9), 70-15-15 (S_{10}) are denoting the same. The distribution of images shown in Table 3.2.

These Table 3.12 and Table 3.13 indicate that the proposed method is performing better than three CNN classifiers, i.e. VGG16, InceptionResnetV2 and Xception not only in case of Average Recognition rate but also Macro Average of F_1 Score.

In maximum cases VGG16 is emerged as winner among the used CNN models. But as the number of train images increase Xception gives superior results, better than VGG16.

Here also to evaluate the statistical significance of proposed method, t-test has been done and p-values are shown in Table 3.14.

Table 3.14: p-Value for Gaussian Function based Proposed Method (Custom Dataset)

	p-Value				
Model	S_6	S_7	S_8	S_9	S_{10}
VGG16	**0.040691**	0.171761	0.129855	**0.029057**	0.186950
InceptionResNetV2	**0.009494**	**0.035847**	0.160517	**0.037377**	**0.027484**
Xception	**0.016866**	0.068749	0.089072	**0.015275**	**0.057525**

The proposed method is approximately statistically significant over Inception-ResNetV2 expect one case. As the number of images of train set increases, the proposed method becomes more statistically significant. It is obvious that higher number train image may lead to higher accuracy and less p-Value but the proposed method works better according to p-Value over higher number train set. So, the model becomes more statistically significant. Table 3.14 shows the results.

2) Triangle Density Function Now, proposed method is tested using Triangle density function on this custom dataset. The Average Recognition rate and Macro Average of F_1 Score on 5 different parts of dataset are calculated and given in Table 3.15 and Table 3.16 below -

Table 3.15: Average Recognition using Triangle Function (Custom Dataset)

	Average Recognition Rate				
Model	S_6	S_7	S_8	S_9	S_{10}
VGG16	0.9851	0.9638	0.9746	0.9906	0.8206
InceptionResNetV2	0.9649	0.9193	0.9278	0.9914	0.8903
Xception	0.9920	0.8607	0.8873	0.9674	0.9817
Proposed Method	**1.00000**	**0.99710**	**0.99420**	**0.99910**	**0.99540**

Table 3.16: Macro Average using Triangle Function (Custom Dataset)

	Macro Average				
Model	S_6	S_7	S_8	S_9	S_{10}
VGG16	0.9856	0.9634	0.9899	0.9908	0.7859
InceptionResNetV2	0.9673	0.9176	0.8208	0.9922	0.8863
Xception	0.9918	0.8253	0.8306	0.9640	0.9827
Proposed Method	**1.00000**	**0.99440**	**0.99360**	**0.99920**	**0.99570**

Table 3.15 and Table 3.16 are showing that the proposed method is giving better Recognition rate and Macro Score for each and every case over the used

CNN classifiers. VGG16 is providing better results among the three CNN models. As train set increases gradually as a result Xception showing better results.

The t-test is done on this Triangle function based proposed method also. The p-values are given in Table 3.17 -

Table 3.17: p-Value for Triangle Function based Proposed Method (Custom Dataset)

	p-Value				
Model	S_6	S_7	S_8	S_9	S_{10}
VGG16	0.154538	**0.050158**	0.089789	0.133014	0.122882
InceptionResNetV2	**0.044239**	**0.028124**	**0.026308**	**0.043822**	**0.032784**
Xception	**0.039909**	**0.017166**	**0.0002663**	**0.053971**	0.141563

The Table 3.17 is showing that Triangle function based the proposed method is totally statistically significant over InceptionResnetV2 and except one case it is also statistically significant over Xception. That one case happens when the train set percentage is highest. Likely, Xception will give better result as training increases so p-Value is greater than 0.05 but Recognition rate is better for the proposed method always which is given in Table 3.15.

3) Trapezoidal Density Function :

Lastly, the third one, Trapezoidal density function based proposed method is tested using this custom dataset. The Average Recognition rate and Macro Average of F_1 *Score* on 5 different parts of dataset are calculated and given in Table 3.18 and Table 3.19 below -

Table 3.18: Average Recognition using Trapezoidal Function (Custom Dataset)

	Average Recognition Rate				
Model	S_6	S_7	S_8	S_9	S_{10}
VGG16	0.9774	0.9896	0.9567	0.9802	0.9989
InceptionResNetV2	0.8554	0.9644	0.9856	0.9159	0.9417
Xception	0.8475	0.9925	0.9402	0.9656	0.9840
Proposed Method	**0.99070**	**1.00000**	**1.00000**	**0.99660**	**1.00000**

Table 3.18 and Table 3.19 are showing that the proposed method is giving better Recognition rate and Macro Score for each and every case over the used CNN classifiers. VGG16 is providing better results overall than other two CNN models.

Table 3.19: Macro Average using Trapezoidal Function (Custom Dataset)

Model	Macro Average				
	S_6	S_7	S_8	S_9	S_{10}
VGG16	0.9770	0.9889	0.9506	0.9798	0.9989
InceptionResNetV2	0.8459	0.9675	0.9855	0.9099	0.9395
Xception	0.8193	0.9924	0.9334	0.9626	0.9833
Proposed Method	**0.99090**	**1.00000**	**1.00000**	**0.99660**	**1.00000**

Table 3.20: p-Value for Trapezoidal Function based Proposed Method

Model	p-Value				
	S_6	S_7	S_8	S_9	S_{10}
VGG16	**0.040691**	0.171761	0.129855	**0.029057**	0.186950
InceptionResNetV2	**0.009704**	**0.035847**	0.160517	**0.037377**	**0.027484**
Xception	**0.017556**	0.068749	0.089072	**0.015275**	**0.057525**

Table 3.20 is showing the proposed method is approximately statistically significant over InceptionResnetV2 and Xception.

Comparing the three density functions is done on the basis of p-values. The points which are nearer to the X-axis means that p-values of those are less and the proposed method is more significant over them.

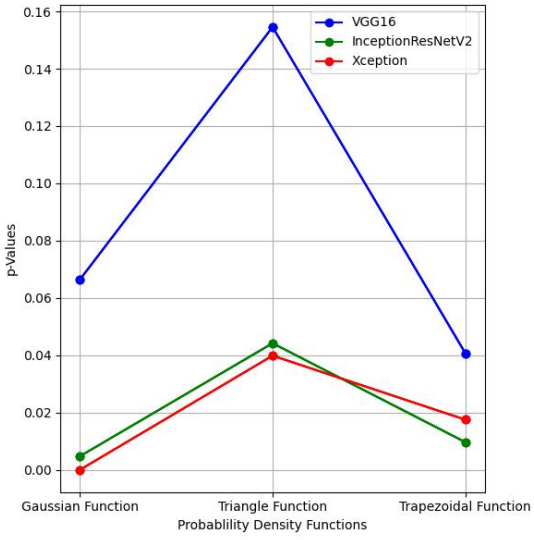


Figure 3.7: p-Value Comparison for S_6

Figure 3.7 shows that the proposed method is totally statistically significant over InceptionResnetV2 and Xception. Only Trpezoidal function based proposed method is statistically significant over VGG16.

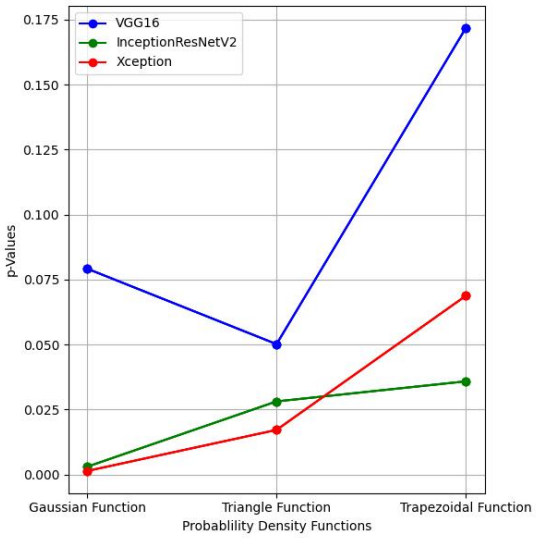


Figure 3.8: p-Value Comparison for S_7

Figure 3.8 shows that the proposed method is totally statistically significant over InceptionResnetV2. Only Triangle function based proposed method is statistically significant over all three CNN models i.e. VGG16, InceptionResnetV2 and Xception.

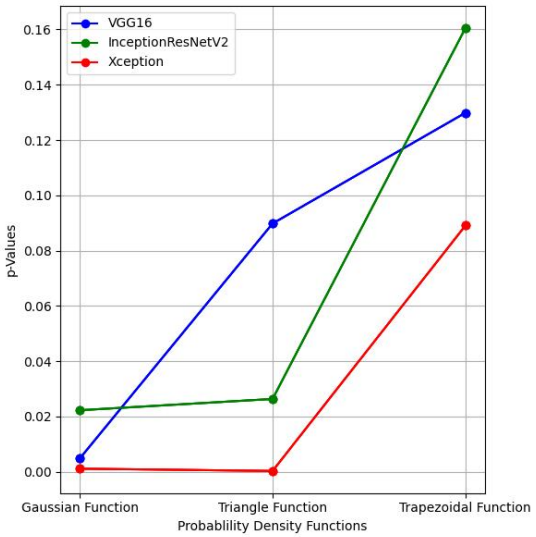


Figure 3.9: p-Value Comparison for S_8

Figure 3.9 is showing that the Gaussian function based proposed method is totally statistically significant over all three CNN models. Triangle function based proposed method is statistically significant over both InceptionResnetV2 and Xception.

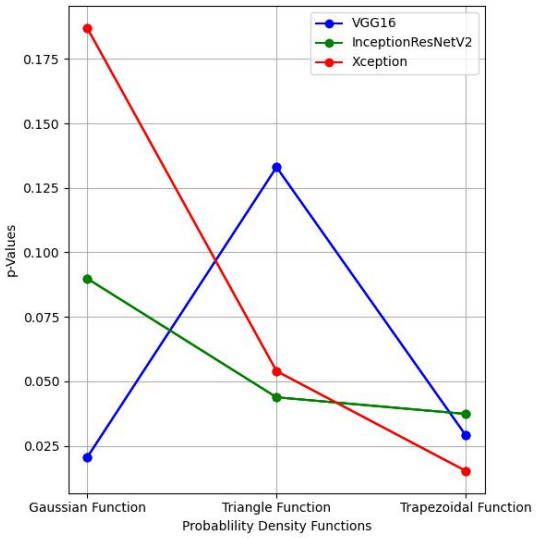


Figure 3.10: p-Value Comparison for S_9

According to Figure 3.10, the proposed method while using Trapezoidal function is totally statistically significant over all three CNN models. Triangle function based proposed method is statistically significant over both InceptionResnetV2 and Gaussian function based proposed method is statistically significant over VGG16.

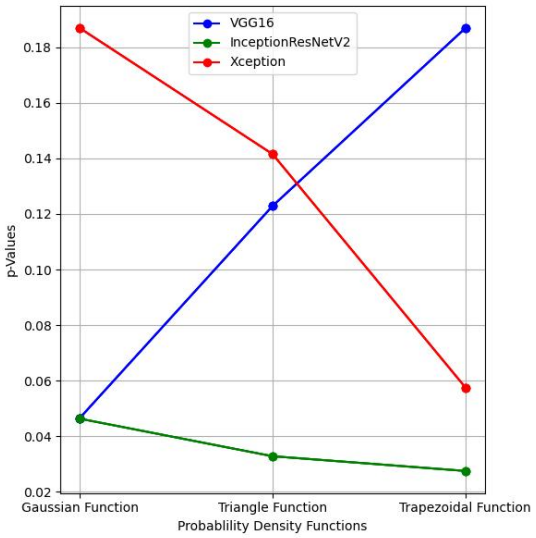


Figure 3.11: p-Value Comparison for S_{10}

According to Figure 3.11, the proposed method is totally statistically significant over InceptionResnetV2 irrespective of any density function used. Only Gaussian function based proposed method is statistically significant over VGG16.

In the figures Figure 3.7, Figure 3.8, Figure 3.9, Figure 3.10 and Figure 3.11, it is noted that Triangle function based proposed method has emerged as the most significant. But Gaussian function based proposed method has better diversity.

CHAPTER 4

Conclusions

4.1 Conclusions of the Present Work

The thesis proposes a unique method for rank level fusion. This method specially focuses on the ranking using confidence factors of different classifiers unlike conventional ranking, fuzzy ranking based fusion is implented. For different classifiers, fuzzy rank is calculated in parallel. The final ranking respect to a class is calculated using weighted rank level fusion technique. The multiplication is done between cummalitive fuzzy ranks and complement of average of commulative confidence factors. The class which has least fused score will be predicted as winner. Three CNN classifiers are used to generate confidence factors 1) VGG16, 2) InceptionResNetv2 and 3) Xception. The results on Chokepoint and custom dataset shows that the proposed method has an improvement performance over the used CNN models. The results of p- value also describes the effectiveness of the proposed method. The proposed method can be used in biometric identification such as finger print, palm print, signature etc.

4.2 Scope for Future Works

Rank-level fusion in face recognition involves combining the outputs of multiple image classifiers at the decision level to improve overall performance. This proposed method has significant potential for enhancing accuracy, robustness, and reliability in face recognition systems. For future work in this area, several key research directions and opportunities can be explored:

- Fine-tuning: Fine-tuned pre-trained deep learning models can be used in rank-level fusion to improve performance on specific datasets or tasks.

- Optimization Techniques Algorithm Selection: Such algorithm may be selected which has more optimization power and as well as simple to implement.

- Real-time Processing Efficiency: Computational efficiency may be improved for real-time face recognition or surveillance.

- Robustness: Designing more robust system for face recognition may help to crucial moments.

- Scalability: Making more scalable fusion method for face recognition is very important and also maintaining the accuracy.

Bibliography

[1] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[2] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017.

[3] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[4] M. I. Ribeiro, "Gaussian probability density functions: Properties and error characterization," *Institute for Systems and Robotics, Lisboa, Portugal*, 2004.

[5] W. T. Scherer, T. A. Pomroy, and D. N. Fuller, "The triangular density to approximate the normal density: Decision rules-of-thumb," *Reliability Engineering & System Safety*, vol. 82, no. 3, pp. 331–341, 2003.

[6] J. Lawrence, R. Kacker, and R. Kessel, "Obtaining a trapezoidal distribution," *Communications in Statistics-Theory and Methods*, vol. 44, no. 21, pp. 4586–4599, 2015.

[7] J. K. Sing, A. Dey, and M. Ghosh, "Confidence factor weighted gaussian function induced parallel fuzzy rank-level fusion for inference and its application to face recognition," *Information Fusion*, vol. 47, pp. 60–71, 2019.

[8] A. K. Jain, P. Flynn, and A. A. Ross, *Handbook of biometrics*. Springer Science & Business Media, 2007.

[9] A. A. Ross, K. Nandakumar, and A. K. Jain, *Handbook of multibiometrics*. Springer Science & Business Media, 2006, vol. 6.

[10] A. Jain, K. Nandakumar, and A. Ross, "Score normalization in multimodal biometric systems," *Pattern recognition*, vol. 38, no. 12, pp. 2270–2285, 2005.

[11] S. C. Dass, K. Nandakumar, and A. K. Jain, "A principled approach to score level fusion in multimodal biometric systems," in *Audio-and Video-Based Biometric Person Authentication: 5th International Conference, AVBPA 2005, Hilton Rye Town, NY, USA, July 20-22, 2005. Proceedings 5*, Springer, 2005, pp. 1049–1058.

[12] S. N. Tumpa and M. L. Gavrilova, "Score and rank level fusion algorithms for social behavioral biometrics," *IEEE Access*, vol. 8, pp. 157 663–157 675, 2020.

[13] V. Chatzis, A. G. Bors, and I. Pitas, "Multimodal decision-level fusion for person authentication," *IEEE transactions on systems, man, and cybernetics-part a: systems and humans*, vol. 29, no. 6, pp. 674–680, 1999.

[14] W. Ali, W. Tian, S. U. Din, D. Iradukunda, and A. A. Khan, "Classical and modern face recognition approaches: A complete review," *Multimedia tools and applications*, vol. 80, pp. 4825–4880, 2021.

[15] L. Sirovich and M. Kirby, "Low-dimensional procedure for the characterization of human faces," *Josa a*, vol. 4, no. 3, pp. 519–524, 1987.

[16] C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition," *IEEE Transactions on Image processing*, vol. 11, no. 4, pp. 467–476, 2002.

[17] A. L. Yuille, P. W. Hallinan, and D. S. Cohen, "Feature extraction from faces using deformable templates," *International journal of computer vision*, vol. 8, pp. 99–111, 1992.

[18] D. Ouyang, Y. Zhang, and J. Shao, "Video-based person re-identification via spatio-temporal attentional and two-stream fusion convolutional networks," *Pattern Recognition Letters*, vol. 117, pp. 153–160, 2019.

[19] F. A. Bhat and M. A. Wani, "Performance comparison of major classical face recognition techniques," in *2014 13th International Conference on Machine Learning and Applications*, IEEE, 2014, pp. 521–528.

[20] A. Daffertshofer, C. J. Lamoth, O. G. Meijer, and P. J. Beek, "Pca in studying coordination and variability: A tutorial," *Clinical biomechanics*, vol. 19, no. 4, pp. 415–428, 2004.

[21] S. Balakrishnama and A. Ganapathiraju, "Linear discriminant analysis-a brief tutorial," *Institute for Signal and information Processing*, vol. 18, no. 1998, pp. 1–8, 1998.

[22] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[23] T.-W. Lee and T.-W. Lee, *Independent component analysis*. Springer, 1998.

[24] M. Coşkun, A. Uçar, Ö. Yildirim, and Y. Demir, "Face recognition based on convolutional neural network," in *2017 international conference on modern electrical and energy systems (MEES)*, IEEE, 2017, pp. 376–379.

[25] K.-M. Lim, Y.-C. Sim, and K.-W. Oh, "A face recognition system using fuzzy logic and artificial neural network," in *[1992 Proceedings] IEEE International Conference on Fuzzy Systems*, IEEE, 1992, pp. 1063–1069.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[28] X. Xia, C. Xu, and B. Nan, "Inception-v3 for flower classification," in *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, 2017, pp. 783–787.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[30] C. K. Chui, X. Li, and H. N. Mhaskar, "Neural networks for localized approximation," *mathematics of computation*, vol. 63, no. 208, pp. 607–623, 1994.

[31] J. Xu, Z. Li, B. Du, M. Zhang, and J. Liu, "Reluplex made more practical: Leaky relu," in *2020 IEEE Symposium on Computers and communications (ISCC)*, IEEE, 2020, pp. 1–7.

[32] H. Chung, S. J. Lee, and J. G. Park, "Deep neural network using trainable activation functions," in *2016 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2016, pp. 348–352.

[33] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.

[34] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[35] A. Abaza and A. Ross, "Quality based rank-level fusion in multibiometric systems," in *2009 IEEE 3rd International Conference on Biometrics: Theory, Applications, and Systems*, IEEE, 2009, pp. 1–6.

[36] M. Lin, R. Ji, Y. Wang, *et al.*, "Hrank: Filter pruning using high-rank feature map," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 1529–1538.

[37] D. G. Saari, "The optimal ranking method is the borda count," Discussion paper, Tech. Rep., 1985.

[38] M. M. Monwar and M. L. Gavrilova, "Multimodal biometric system using rank-level fusion approach," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 4, pp. 867–878, 2009.

[39] A. Kumar and S. Shekhar, "Personal identification using multibiometrics rank-level fusion," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 5, pp. 743–752, 2010.

[40] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE signal processing letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[41] S. Alver and U. Halici, "Attentive deep regression networks for real-time visual face tracking in video surveillance," *arXiv preprint arXiv:1908.03812*, 2019.

[42] G. Barquero, C. Fernández, and I. Hupont, "Long-term face tracking for crowded video-surveillance scenarios," in *2020 IEEE International Joint Conference on Biometrics (IJCB)*, IEEE, 2020, pp. 1–8.