An Efficient Deep Learning Approach for 3D Face Detection Using Multiple Angular 2D Images

by

Nayeem Rafsan
19101133
Hasibul Hoque Chowdhury
19101433
Hamed Efaz Md. Elahi Dad
19101304
Afzal Hossain Yen
18201131

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
September 25, 2022

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Rafsan

Nayeem Rafsan
19101133

[illegible]

Hasibul Hoque Chowdhury
19101433

Afzal Hossain Yen

Afzal Hossain Yen
18201131

Hamed Efaz Md. Elahi Dad

Hamed Efaz Md. Elahi Dad
19101304

Approval

The thesis/project titled “An Efficient Deep Learning Approach for 3D Face Detection Using Multiple Angular 2D Images” submitted by

1. Nayeem Rafsan (19101133)
2. Hasibul Hoque Chowdhury (19101433)
3. Afzal Hossain Yen (18201131)
4. Hamed Efaz Md. Elahi Dad (19101304)

Examining Committee:

Supervisor:
(Member)

[illegible]

Md. Ashraful Alam, PhD
Assistant Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Md. Golam Rabiul, PhD
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

3D face reconstruction is a useful computer vision technique for facial recognition. Accuracy decreases drastically while extracting features from 2D and moving images. To overcome this problem, we are proposing reconstruction of 3D models generated by multiple 2D angular images. Our primary approach consists of the following steps: rebuilding 3D mesh from 2D image, feature extraction, deep learning algorithm for recognition. We will be taking images of 0°, +10°, +20°, +30°, +40°,+50°, +60°, +70°, +80°, +90°, -10°, -20°,-30°, -40°, -50°,-60°, -70°,-80°, -90° angular deviations. We have compared the results from 2D architectures and 3D architectures and showed that 3D deep learning models perform better on angular images and in motion. The proposed method is time efficient and robust in nature, and it overcomes the previous limitations.

Keywords: 3D Point Cloud, Face Detection & Recognition, CNN, NextFace, PointNet, Haar Cascade, VGG16

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.
Secondly, to our advisor Dr. Md. Ashraful Alam sir and his research assistant Shakib Mahmud Dipto for their kind support and advice in our work. They have helped us whenever we needed help.
And Thirdly, to our parents without their throughout support it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

Chapter 1

Introduction

1.1 Introduction

Face recognition is becoming very popular day by day around the world as it is an incredibly secure and dependable security tool. As it's security level is very high and it is very reliable it is gaining more attention from many entities. Furthermore, when we compare face recognition to other biometric security systems such as fingerprints and palm prints, face recognition provides more advantages. Biometric measurements of a person are taken by the system from a set distance without having to interact with them. The process can serve in criminology sector for identifying someone with a previous record. As a result, this technology is becoming increasingly important for many residential structures and businesses. The idea is to first detect a face, point out the prominent features and compare these traits to faces from existing database. It is designed with functions and operations to extract main characteristics of a human face. The FR system uses these characteristics to quantify important features such as the length of the jawline, the distance between the eyes, the depth of the eyes, and the curve of the cheekbones. These nodal points are calculated by generating Faceprint, which represents the identification of that face in a computer database. With the evergoing advancement of technology, 2D graphics based systems are now available in 3D graphics, which will improve the accuracy and dependability of the system.

1.2 Problem Statement

Biometric security systems like fingerprint, voice, retinal, face recognition from 2D are massively used by the offices for security and attendance of the employees. But these solutions are inefficient because people need to set their thumb, face, eyes in a certain position in front of a device or camera which can be time consuming if the number of people are large. For instance, a factory consisting of hundreds of workers have to go through the system to provide their attendance. During rush hour, we can often see a queue of employees at the entrance trying to register their fingerprints or ID's. This wastes a lot of valuable time in factories. On the other hand, using only a front facing 2D face recognition model raises concerns about accuracy when the employees are not looking at the camera directly. Nevertheless it drops accuracy when a person is in motion. To solve this problem, we are proposing a 3D face recognition system, which can recognise faces even when they're in motion and not looking straight at the camera.

1.3 Research Objectives

We now have the sufficient technologies to train 3D models in deep neural network. 3D data is being used vastly in different sector of the industry. Therefore, we want to utilize it to overcome security issues. Our aim is to prepare a deep learning model to solve the above mentioned problem. We want to reduce the face recognition time in large factories. It will save a great amount of time from which industries can be benefited largely. We believe our solution will recognize faces even when a person is in motion. Furthermore, the model will also be able to recognize face from -60° to +60° angle with high accuracy and provide a prominent result. We will further extend our work and analyze result for larger angles such as -90° to +90°. We believe using computer vision knowledge on 3D point clouds to provide such a solution will also open doors for further research and development which is the ultimate purpose of our paper.

Chapter 2

Literature Review

Early research work acknowledged the problem of using a 2D representation of a face for facial recognition. Different techniques were used in previous times. An approach to replicate a 3D facial model using one 2D image was proposed [3]. The idea of this paper was to use affine transformation on a 3D statistical face model.A different approach was taken in [2] which used 3D eigenface for face recognition. The paper takes advantage of principal component analysis (PCA) to construct the eigenvector of the faces. Afterward, it uses K-nearest neighbor (KNN) to classify the faces. Another approach for face recognition was proposed [4] which suggests a point cloud kernel correlation method. Kernel correlation is a measure of similarity for point clouds. Comparing two-point clouds if they're close to each other the kernel correlation provides a greater value. The first step is to register the face by assuming a static point set and try to find the transformation matrix which maps the point clouds to the static point set with the help of a cost function. Afterward, it tries to find the kernel correlation between one target model and the rest of the registered models and sorts the similarity values in descending order to find the best match. The technique was tested on 427 3D laser scan facial images after the images were normalized by keeping the tip of the nose at the origin. The performance metric in this paper was the total similar rate which was obtained by a total similar face divided by a total number of faces in the database. It can successfully recognize 387 images out of 427 images, which is about 90%. The advantage of this process is no pre-processing is required. We can consider the kernel correlation as similarity metric in our paper.

In the research paper [6] addresses two fascinating ways we can capture 3D data. First option is to reconstruct 3D by combining multiple 2D images with image-based 3D face reconstruction techniques while the second option is using dedicated 3D scanners. As 3D scanners cannot operate in real time and they are very expensive, reconstructing 3D from 2D images is the go to move for researchers. To reconstruct 3D from 2D images, they tried various methods. Exercising using example-based methods necessitates a 3D face database which is huge and diversified, and they don't work well in different lighting conditions. In most situations, datasets utilized in conjunction with example-based 3D reconstruction methods are insufficiently diverse to allow for the reconstruction of faces belonging to people of various ethnic backgrounds and/or ages. Instead of producing geometrically exact 3D faces, the fundamental goal of such technologies is to make genuinely looking 3D faces.Stereo orthogonal image tries to rebuild the geometry more accurately, but they typically necessitate a lot of human work. The lack of a 3D face database, good result to any lighting conditions and ease of implementation are the key advantages of stereo orthogonal approaches. Despite the fact that the goal is to rebuild mathematically correct images, the resulting facial geometry is frequently smoothed out because the generic face model is highly smooth and only common features (across all faces) are chosen, eliminating face characteristics specific to a subject's face. Stereo non-orthogonal approaches, on the other hand, have demonstrated geometric precision and run mostly automatically, with the only drawback being the need for organized illumination.When using video approaches for the first time, it's easy to think they're better because more data is accessible to aid reconstruction. Given that the face is caught from a variety of angles, we can assume that it can create more accurate 3D faces. However, tracking all (or part) of the video frames takes more time, and the results aren't significantly better than stereo orthogonal image-based approaches.So, by analyzing this research paper we can find that using Stereo non-orthogonal approaches is the best option for us.

Another interesting technique of matching features is by extracting features from multiple views. The research work [5] proposes an interesting process for face recognition from 3D models. It extracts multiple 2.5D images after rotating the 3D point cloud incrementally. For this, it computes the extrema of the point cloud along the three axes X,Y and Z, then creates a 2D image and after scaling the points it projects onto the 2D image generating a 2.5D image. After projecting multiple 2.5D images the paper utilizes the Scale Invariant Feature Transform (SIFT) algorithm. This algorithm tries to find similar points in two different images of the same object, if it can match at least three points it assumes that the object is present across the two images. Furthermore, it combines the key points from all the views into a set. For face recognition, it performs the same task and matches the key points with the trained dataset. For the matching purpose, it uses a keypoint voting algorithm. This algorithm finds the closest Euclidean distance from the training dataset and assigns a weight to it, and closed the distance the weight is greater and thus finally classifies the image to the nearest subject. The proposed idea was tested on the GavaDB dataset which consists of 61 different individuals with a total of 549 images and yields an accuracy of 95.08%.

This particular paper [1] presents an object recognition strategy based on machine learning that is able to process pictures fast while attaining a good detection number. In order to do this, they developed a frontal face detection system that generates detection as well as false positive numbers which are comparable to the best reported findings. By only utilizing the information contained in a single grey scale picture, this technique achieves high frame rates. Three significant approaches differentiate this study. First comes the implementation of novel representation of photographs known as the "Integral Image". This increases the calculation of feature detection fast. Next is a learning method inspired by AdaBoost that chooses key visual characteristics from a huge collection to generate efficient yet exceptional classifiers. Third criterion is a "cascade" strategy which combines classifiers discarding background of an image quickly while more processing is focused on probable regions which might present objects. By simple tweaking of AdaBoost, selection of features is performed, inspired by Viola-Tieu. AdaBoost offers a powerful learning algorithm as well as tight generalization constraints. The idea underlying focus of attention techniques is to predict where an object could appear in a picture as fast as possible. These potential sections are then processed via advanced calculations. To recognize frontal upright faces, classifier training of 38-layer was done. Out of 6091 samples 10 features were analyzed on an average by persub-window on the test set of MIT+CMU. The Schneiderman-Kanade detector is around 600 times quicker than the detector by Rowley-Baluja-Kanade, which is about 15 times faster. This research combines unique methodologies, representations, and concepts that are reasonably generic and might be applied to a wider variety of computer vision and machine learning applications.

The authors had proposed [10] a human face dataset called CASIA-WebFace for enhancement in face recognition. For face recognition, two things are most important one is the algorithm and another is the dataset. But many research groups can't give the best approach because of a lack of proper data. As creating a dataset is a very costly so the groups achieving above 97% accuracy keeps their data private including Facebook. That's why this paper became very noteworthy. CASIA-WebFace is a semi-automatic way to collect face images from the Internet and the dataset contains about 500,000 images of 10,000 subjects. Moreover, using an 11 layer CNN this approach gives state of the art accuracy on YTF and LFW. This method collects the face images from the internet and annotates their identity with the help of the IMDb website. Then authors correct the false annotation manually so that it is called a semi-automatic technique. Furthermore, they augment the dataset by mirroring 493456 images and then train the network with 986912 images which is the second-highest number in the present world after Facebook. In all experiments, the network gives better results than previous methods and testing on YouTube Faces(YTF) the accuracy can be reached to 90.6% which is very high. Therefore, this paper can help us to enlarge our data collection.

The paper [21] proposes to perform facial recognition by re-creating a 3D face model from multiple 2D images. Initially, it takes a few 2D images from multiple angles with a 30-degree deviation of the same person. Before constructing the 3D model it used the HaarCascade algorithm [1] to detect the face in the image. Then it uses Structure from motion algorithm which is a combination of scale-invariant feature transform (SIFT), approximate nearest neighbor (ANN), and RANSAC algorithm. Here, SIFT is a feature extractor as mentioned earlier [5] and the rest of the algorithms work to remove outliers and map the same point in the face across different images. All of these work together to rebuild the face in 3D space and afterward it takes snapshots of the 3D images and trains them on a classical machine learning technique AdaBoost. The model could provide a 100% accuracy for images taken from 0 degrees and 30 degrees both left and right. But for 60-degree angle images, the accuracy would drop to 80%.

In this paper [19], the authors created and tested a real time face recognition system based on the Convolutional Neural Network. To evaluate the performance of the proposed system and CNN architecture, they tuned various CNN parameters and improved accuracy. Using ATT and real-time inputs, the suggested system achieves 98.75 percent and 98.00 percent respectively, which is the maximum by far. The performance of the mentioned system is evaluated by changing the number of filters in the convolution layer and the window size of the convolution filter. Face detection-based home automation, attendance system,device control, intruder detection, and other consumer applications can all be benefited from the suggested study.

In this paper [12] the authors described a unified face verification, recognition and clustering method. The authors introduced FaceNet, a system that extracts features from direct mapping of facial images to a Euclidean confined space in which distances correlate to measurements of similarity of faces. Instead of using a layer containing a bottleneck as in prior deep system learning, this method employed DCNN which was developed to optimize the embedding directly. They were able to gain state-of-the-art facial recognition results by allocating 128bytes memory per face. FaceNet uses a triplet-based loss formula based on LMNN to directly train its result to be a concise 128-D embedding. In all of their trials, they used Stochastic Gradient Descent (SGD) with AdaGrad and conventional backprop to train the CNN. They employed two different architectural styles. The first category adds 1x1xd convolutional layers to the ZeilerFergus architecture's regular convolutional layers. In the second category, they employed Inception models based on GoogLeNet. On the LFW (Labeled Faces in the Wild) dataset, their technique performed with a record 99.63 percent accuracy. It received 95.12 percent on YouTube Faces DB. On both datasets, the approach reduced the error rate by 30% when compared to the best reported result. The training of end-to-end streamlines needing small alignments while also utilizing a loss that is relevant to current job produces performance increase. FaceNet distinguishes this paper [12] from other approaches that rely on the CNN layer requiring further post-processing such multiple model concatenation and PCA, along with SVM classification.

To upgrade the accuracy of face recognition the authors proposed [20] a novel type of face recognition architecture which is called GroupFace. The main motive for the authors was a model which needed to learn differentiation of millions of face data by using a little amount of dimensional embedding features and these huge data cannot be encoded in a single branch. So, to improve the quality of embedding features GroupFace can use multiple group aware representations. Another feature of GroupFace is it gives self distributed labels that distribute samples in different groups without human annotation so it decreases the search space of desired identity. In many good models there is a difficulty to enable effective training to recognize face identities with a huge quantity because it is very difficult to recognize new faces which are not included in the training set. So They came up with a new technique which is grouping. It can efficiently embed a huge number of people and provide a description of unknown identities. Because every person can be classified in a group by their facial appearance. By adding a few convolutions we can improve any existing methods through the concept of grouping and can show a perfect visual description. Finally, in the datasets such as LFW, YTF, MegaFace this method gets the state of the art result with brilliant accuracy. As this is a unique technique of face recognition, this paper gives us motivation to analyse it.

The authors introduced a new 3D face recognition method that incorporates a

DCNN algorithm (Deep Convolutional Neural Network) and a three dimensional facial augmentation technique in this [15] study. They presented a 3D augmentation technology that uses a single 3D face scan to synthesize a variety of facial emotions. To accomplish 3D to 3D surface matching, the authors recommended using existing networks that have been trained for 2D facial recognition software and fine-tuning them using a limited number of 3D scans. Secondly, they propose to supplement their 3D face dataset with synthetic 3D face data that considers facial expressions to overcome this issue. From the Basel Face Model researchers employed the cross 3D morphable models in conjunction with the FaceWarehouse expression to supplement training data. After processing the initial 3D scans they implement conventional CNNs for two dimensional face recognition and perfect them for 3D face identification using additional datasets followed by the change of expression of each 3D scan within the training data and addition of a generated point cloud to it via 3DMM. After sorting out the random patches and fine-tuning the system by combining weights from VGG Face, identification happens in the 4096 vector dimension via feature extraction using Principal Component Analysis by matching features of the nearest gallery. In accordance with their experiment the average performance rate was 96.3% where the rate was 95.0%, 94.8% and 99.2% on BU-3DFE, 3D-TEC and Bosphorus respectively. Even with a little dataset, the authors offered the very first 3D face recognition system using DCNN.

In this paper [17] authors surveyed some of the most cutting-edge methods for 3D face identification in the face of expressions, occlusions, and position changes. Local approaches and holistic methods are the two basic groups of recognition methods. Despite the fact that many trials are based on the holistic method, we feel the local method is better for 3D face recognition. This research looked at different strategies for pose-invariant face identification on publicly available databases that can handle a wide range of positions. The majority of the time, the recognition mechanism was local. Half face matching, for example, can be used to create a whole face model. To overcome the data missing problem, a statistical model in the radial curve's shape space was used. For expression-invariant 3d face recognition, there are two methods: local approaches based on expression invariant approaches and holistic approaches.

In this research [13] the authors proposed a network called PointNet which is a novel deep neural network and takes input data as point clouds and gives output as 3D recognition in three categories, first one is object classification, second one is part segmentation and last one is scene segmentation. Due to irregular format of point clouds the data was transformed by researchers into 3D voxel grids before training to the network so the data became unnecessarily large and made the natural invariances of the data unclear. So the network PointNet gave the best solution. This was a unified architecture and it represented every point by a vector with 3 coordinates denoted by x, y, z. This architecture used a symmetric function called max pooling. This network also learned the function of selecting interesting and important points from point clouds and provided the reason for selection points. For 3D object detection, they split 12311 CAD models into 9843 for training and 2468 for testing. By fully connected layers and max-pooling function, this network gave the maximum accuracy compared to previously used methods as this network is the first to take raw point clouds as input. This network also gave 80% accuracy when 20% of the points are excluded. So the motivation behind this paper was to train our face recognition network by the .ply files which were also the point clouds of 3D facial mesh.

The authors updated the PointNet network to PointNet++ in this [16] research paper. It is difficult to capture detailed local structures which are persuaded by metric space because of the failings of the PointNet network so it is difficult for the PointNet network to generalize the complexity and recognize the fine-grained patterns. To solve these issues they came up with a hierarchical neural network called PointNet++. It partitioned the input point clouds and recursively uses PointNet. This network made the neurons of higher levels in order to get larger receptive fields so that the power of generalizability to rare cases could be increased. The main two tasks of this network are partitioning the point set and abstracting local features through local feature learners. Here PointNet can be used as a local feature learner. Thus PointNet is used on the point clouds recursively which would be partitioned in nested structure. To achieve detail capture PointNet++ leverages neighborhoods at multiple scales. There is another issue with the non-uniform point sample. To solve this issue two novel set abstraction layers which can aggregate multi-scale information according to local point densities. Finally, PointNet++ gave better accuracy than PointNet based on performance on 3D point clouds.

The research [18] paper highlighted the usage of 3D data points in modern-day technologies like autonomous cars, robotics, augmented reality but these point clouds are unstructured data, unlike 2D grid pixels. Because of this, it is quite difficult to apply the convolutional operation on these data. For working with 3D data the paper [18] proposed a new approach for classifying 3D point clouds. It suggested a convolution operator, called PointConv. The main goal of this operator is to approximate the 3D continuous convolution by taking points cloud as input. It tried to view the convolution operations as a discrete approximation of a continuous convolution operator and in 3D it treated the weight to be a continuous function of the local 3D point with respect to a reference 3D point. The other big achievement of this paper is to figure out an efficient approach to implement the PointConv, as naive implementation is not memory efficient. It used the summation order method and the approach can step up to the level of CNN. The final contribution of this paper was to build a deconvolutional version of this same operator for achieving good segmentation results. Combining all these techniques, the proposed approach could achieve the performance of modern-day CNN on 3D data points. The PointConv operator mentioned in this paper can help us to work with 3D point cloud.

Chapter 3

Methodology

3.1 Architecture of the Proposed System

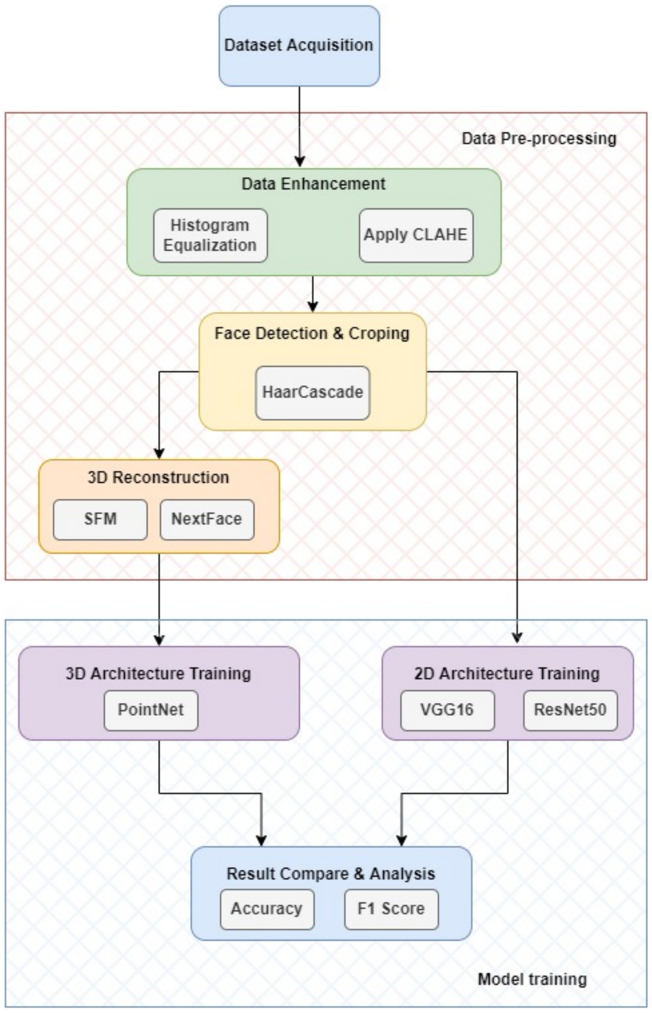


Figure 3.1: Workflow Diagram of Proposed System

Firstly, we acquired angular images of 100 person to use in our system. We took pictures using Canon EOS 750D. We took 1900 facial images of 100 different person. Each person has about 19 images and each image was taken every 10° angular deviation from -90° to +90°. 10 images from 0° to +90° and 9 images from -10°

to -90°. In the pre-processing segment, we used CLAHE, Histogram Equalization to enhance the quality of our images. We used HaarCascade to detect faces from the image and crop out the unnecessary elements of the image. Then we used SFM and NextFace to reconstruct 3D face and mesh to complete our 3D dataset. We used VGG16 and ResNet50 for 2D Face Recognition. We used PointNet for 3D face recognition. We used different analysis metrics such as Precision, Recall, F_1, Accuracy and Confusion Metrics for result analysis.

3.2 Used Architectures for 3D Face Reconstruction

In our research, we use a pipeline called SFM and will implement three architectures in total, which are PointNet, PointNet++ and PointConv. The details of them are given below:

3.2.1 Structure From Motion (SFM)

SFM stands for Structure From Motion [22] which is a algorithm that uses multiple 2D images of a scene to construct the 3D structure of that scene. IN 3D scanning and Augmented Reality, SFM is very useful application. Because of the number of cameras and order of images we can't construct the actual scale of the object. We need more information such as the size of an object and information from other sensors to compute the actual scale of the structure. For simple case, we can use SFM by two views using 2 cameras or moving a single camera. The algorithm works by assuming camera 1 is at origin. The overview of the algorithm is given below:

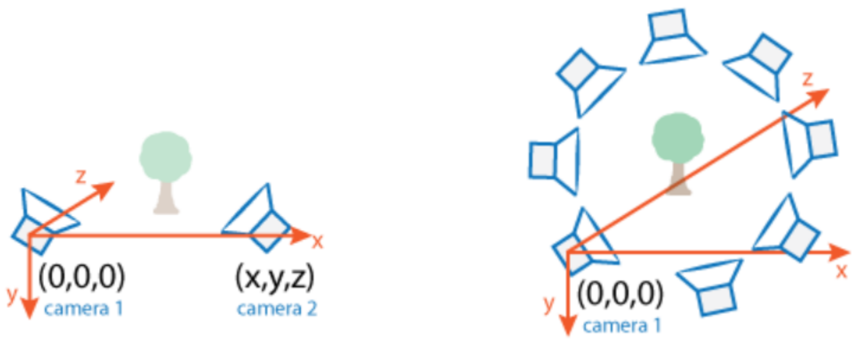


Figure 3.2: SFM from two views, SFM from multiple views

First of all, between multiple images SFM requires point correspondences by the matching features or tracking points from multiple images. Then fundamental matrix must be calculated to find the position of the second camera relative to the first one. Basically the fundamental matrix calculates the epipolar geometry of the cameras. Then by using the fundamental matrix we get output from relativeCameraPose that gives orientation and location of the second camera by assuming origin is the first camera. Then SFM determines the 3D locations of the points which are matched using triangulate. The triangulate function requires two camera matrices which can be calculated by cameraMatrix function. Then SFM uses pcshow function to display the 3D structure. Moreover, we can use the multiple views technique with the same approach of two views technique. The main algorithm is pretty similar but in multiple view the pose of camera 3 is calculated relative to the camera 2 and so on. Then all relative poses must be formed into a single coordinate system so tha all camera poses can be relative to camera 1.

3.2.2 NextFace

For modeling and rebuilding face features, NextFace is an effective formulation. With the use of NextFace, a 3D face can be generated from a single 2D photograph. Utilizing picture consistency loss and 3DMM, NextFace models geometry. Cook-Torrance model and statistical priors are used in the parametric reflectance model to model specular albedo and dilute it. In order to represent scene lighting with differentiable ray tracing, it employs a unique parameterized virtual light stage. These parametric qualities are then framed as an optimization problem that is resolved using a two-stage approach. In NextFace, a 3D Face may be created from a single 2D face picture in three steps.

The facial expression along with the position of the head are estimated in the first stage, referred to as the Coarse Stage, by reducing the geometrical loss between both the 2D landmarks and corresponding face vertices. As a result, the following round of optimization gets off to a solid start. Albedo priors space and statistical geometry both play a role in this phase. Due to the limitations of the original model's low dimensional space, this initial reconstruction does not contain geometry and critical diffused or specular albedo characteristics.

In the second step, referred to as the Medium Stage, the position of the head, statistical diffuse and specular albedos, the identity of the expression, and the lighting of the surroundings are approximated by reducing the photo consistency deficit between the real picture and ray traced image. Without being constrained by the statistical prior, this stage enhances the albedos which were originally computed.

In the third step, the method achieves the originally estimated albedos on a per-pixel basis and seeks to capture additional albedo features, which is called the Fine Stage. Regularizers for consistency, symmetry, and smoothness are employed to prevent overfitting and increase resilience to lighting conditions. Additional diffuse albedo and fine geometric features are added in this stage.

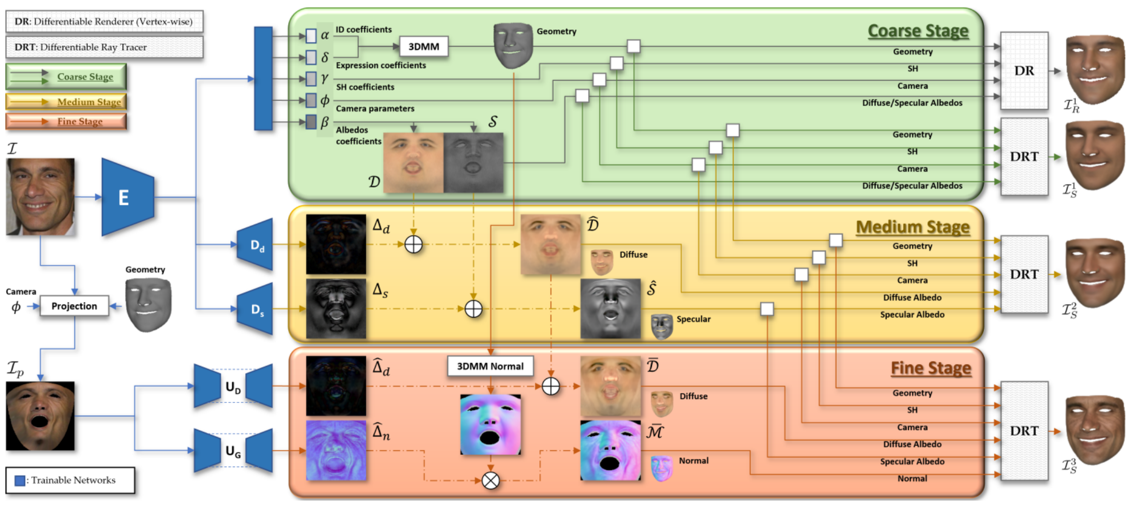


Figure 3.3: NextFace Architecture [23]

3.3 Used Architectures for 2D Face Recognition

3.3.1 VGG16

VGG16 [8] was proposed in 2014 ILSVRC and it is recognized as a best convolutional neural network till today. It has a 16 layer deep network. VGG16 takes input images as dimensions of 224, 224, 3. It gives best accuracy in the imageNet dataset that includes 1400000 images of 1000 classes.

There are 64 channels of a kernel of size 3 x 3 in the first two layers. Then it has a layer of max pool of stride size 2. After that there are 2 convolution layers of 128 kernels with size 3 x 3. Then again it has a similar max pooling layer. After that there are 2 convolution layers of 256 kernels of size 3 x 3. Then it has a max pool and 3 convolution layers of 2 sets. Each set has 512 kernels of size 3 x 3. Then the image is sent to a stack that has a couple of convolution layers.
The kernel we used in these layers are size of 3 x 3 instead of AlexNet's 11 x 11. In few layers we also uses pixel of 1 x 1 and it can manipulate input channel's numbers. VGG16 also includes a mandatory padding after every convolution layer of size 1 pixel because it helps to secure the image's spatial feature. After these convolution and max pool layers we get a feature map of size (7, 7, 512). Then we build a feature vector of size (1, 25088) from the feature map. Then there are three fully connected layers.

The first fully connected layer takes data from the feature vector and also gives output as a vector of size (1, 4096). The second fully connected layer gives the same output as the previous layer. However, the third fully connected layer helps to install the softmax function so that 1000 classes of objects can be classified. All the intermediary hidden layers uses activation function called ReLU.

VGG16 is one of the best architectures in the neural networks world. It gained second position in the ILSVRC challenge. It achieved 7.32% in the top 5 classification error and became first in the localization task with an error of 25.32%.Although,

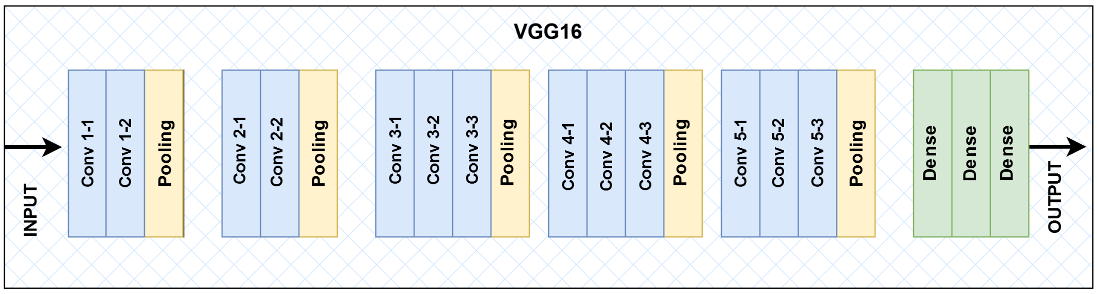


Figure 3.4: VGG16 Architecture Map [8]

VGG16 is quite slow in training and it also takes huge bandwidth and space. Millions of parameters also create explod of gradients problem so researchers proposed ResNet models to prevent the problems of VGG16.

3.3.2 ResNet50

Resnet50 [11] is a new version of ResNet architecture including 48 convolution layers with an Average Pool and a Max Pool Layer. It is a very popular ResNet model because ResNet50 does $3.8 * 10^9$ amounts of Floating point task.

After winning the ILSVRC2012 competition ResNet gains priority in the deep learning world because without ResNet it is impossible to train very deep networks. So ResNet gives the best performance in that network which has thousands of layers. The purpose of ResNet was to decrease the training error. ResNet is a shortcut connection that does identity mappings and the advantage of these identity mapping is no extra parameters and no increase of computational time.

There was some update in the ResNet50 model. In traditional ResNet two layers were skipped in shortcut connection but in Resnet50 three layers were skipped. It also has additional 1 * 1 convolution layer.

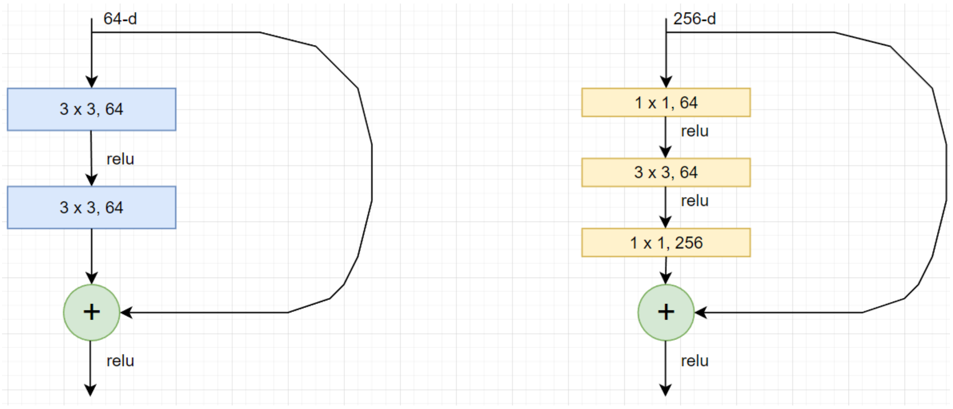


Figure 3.5: Residual structure of ResNet50 [11]

ResNet50 has a 7 * 7 kernel's convolution along with 64 variants of kernels that have 2 units of stride measurement so we get 1 layer. MaxPooling has the same stride size as kernels. In the next convolution there are three layers and they are repeated 3 times so we get 9 layers. The three kernels are: a) 1 * 1, 64, b) 3 * 3, 64 and c) 1 * 1,256 . Next we get 12 layers because here again 3 layers repeated 4 times. They are: a) 1 * 1, 128, b) 3 * 3, 128 and c) 1 * 1, 512. Then again we get 18 layers by kernel 1 * 1, 256; 3 * 3, 256 and 1 * 1, 1024. Then it gives 9 layer by kernel 1 * 1,512; 3 * 3, 512 and 1 * 1, 2048. Then an average pool layer terminates with 1000 nodes and a function of softmax which gives us the fully connected layer. So total 1+9+12+18+9+1 = 50 layers.

In ImageNet the Resnet50's result is satisfactory and it gives 20.47% in top 1 error rate and also 5.25% in top 5 error rate. This result was given by a single model which contains 50 layers.

3.4 Used Architectures for 3D Face Recognition

3.4.1 PointNet

PointNet [14] architecture was proposed in 2016 by researchers of Stanford University. The main goal of this research is to represent the 3D structure of 2D images and classify the images and classify the segment of images. Point cloud data is used to implement this architecture. Basically, a set of points build the Point cloud which is the representation of the 3D shape of any 2D image. But it has some irregularities so many researchers converted the point cloud to other data structures of 3D representations such as voxel. However, this conversion makes the data too large which is very difficult to train the model. So PointNet gives a solution to this problem. A novel method is proposed by PointNet for consuming Point clouds directly and gives the best accuracy in the classification of images and segmentation of images.

The PointNet architecture is instinctive. It takes Point Sets as input from Point Cloud. For object classification input is directly sampled from the 3D shape but input could be single object in semantic segmentation or small segment of 3D shape from object region segmentation.

The classification network mainly does the mapping which maps each n points from 3 dimensions to 64 dimensions with the help of a shared multi-layer network and each of the n points has its own multi-layer network. Similarly, each of the n points is mapped from 64 dimensions to 1024 dimensions in the next layer. In R^{1024}, we now use max-pooling to construct a global feature vector. Finally, with the help of a 3 layer fully connected newtwork the score of k output classification is mapped to global feature vector.

In segmentation networks, segmentation relies on local and global features so each n inputs must be assigned to one classes of segmentation. The global feature space is added to points in the 64-dimensional space, which is the result of a possible feature space of size ($n * R^{1088}$).

3 key modules of PointNet:

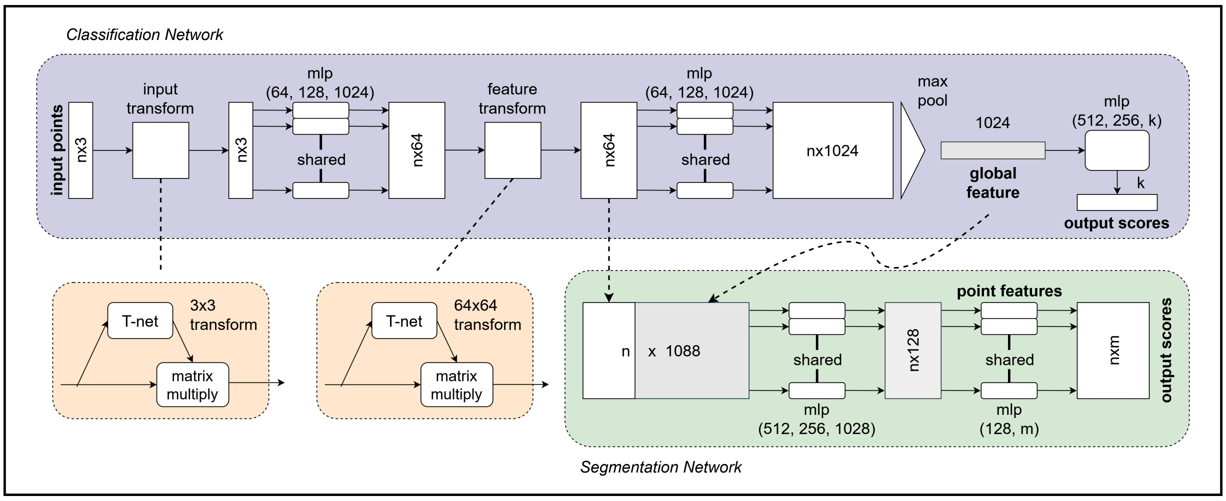


Figure 3.6: PointNet Architecture

1. The Max-pooling layer
2. A local and global structure which is combined
3. 2 joint alignment networks

If a symmetric function: $f(x_1, x_2, ...x_n) = g(h(x_1), h(x_2)...h(x_n))$

where $f : 2^{R^N} \to R; h : R^N \to R^K; g : R^K{*}R^K...R^K(ntimes) \quad is \quad a \quad symmetric \quad function.$

Here, h is multilayer perception, g is a combination of a max pooling function and a single variable function. F is output layer.

The output of the previous part is a form of vector $[f1, f2, ...fn]$, which is the global architecture of input set. This will now function perfectly because the SVM can be simply trained to produce a classifier output. However, we need a combination of local and global information for point segmentation. Authors compute the global feature vector to give input to the point feature by adding per point features with global features to achieve desired outcome. This approach predicts number of per-point by depending on both global semantic and local semantic.

Point cloud's semantic labeling must be invariant to geometric transformations. Here transformation could be rotation and translation. The researcher calculates the transformation matrix using a mini-network and then gives it as input to the coordinates of input point.

The features which depend on input at the T-net's fully connected layer are coupled with weights and biases which are globally trainable to produce a 3 by 3 transformation matrix in the T-net's final stage.

Normalization of pose was advanced to a 64-dimensional space. T-net is virtually identical to prior picture, with the exception of a 25640964096 increase in the dimensionality of weights and biases which are trainable, which gives result in a transformation matrix size of $64 * 64$. Overfitting occurs with trainable's number increases.

3.4.2 PointNet++

In PointNet++ [16] architecture researchers propose a layered neural network. PointNet is not sufficient in this field because it can't capture local structural information at multiple scales and also can't recognize the fine-grained task so in PointNet++ researchers apply PointNet architecture recursively to the hierarchical neural network nested partitions.PointNet++ employs spatial distance measurement, and the network can learn local features as the context size grows. Adaptive learning is advocated since the density of the point set is varied. Multi-scale features mixed with stratification.

The local structure as measured by different scales is ignored by PointNet's design. Local structure, on the other hand, is critical to the success of the convolutional network architecture. Using data defined on a regular grid as input, CNN can gradually capture more and larger characteristics.Here measurement of distance of bottom is used to divide a group of points into local areas which are overlapped. We collect local features as a huge number and process them so that we can produce higher level features just like local features are extracted and sharp geometric designs are captured by CNNs of small neighborhoods. We repeat this process until all of the attributes of the point set have been obtained. To achieve detail capture PointNet++ uses neighborhoods at so many scales. PointNet++ demonstrates its ability to efficiently process point sets.

PointNet++ is set up in a three-layer hierarchical structure. From input the sampling layer selects points randomly. Local area's centroid is calculated by this set of points. The layers are then grouped around the center point by finding the neighboring points. Micro PointNet is used to encode the local area pattern as a feature vector in the PointNet layer.

By utilizing FPS, click $(x1, x2, x3, ...xn)$ to obtain different point sets. Farthest point sampling, as opposed to random sample, can cover the complete data point.

The principle of FPS (farthest point sampling) is to randomly select a point, then select the point furthest away from this point to join the point set, and then iterate, selecting the point furthest from all points which are present in the set of points, until the required number is gained from the points.

The input size is $N(d+C)$, here point cloud's number is N, d is dimension of coordinate, and c denotes the point feature. The output size is $N, K(d+c)N, K/times(d+c)N, XKX(d+c)$, where K denotes the number of points around the centroid. It's worth noting that K varies by group, but the PointNet layer after that can turn flexible points into fixed-length local area feature vectors. The radius of a point is used to find all points within that radius (the larger limit of K is fixed in the working process). The K nearest neighbor (kNN) search calculates the neighboring point's number in a predetermined range and kNN is another range query. In comparison

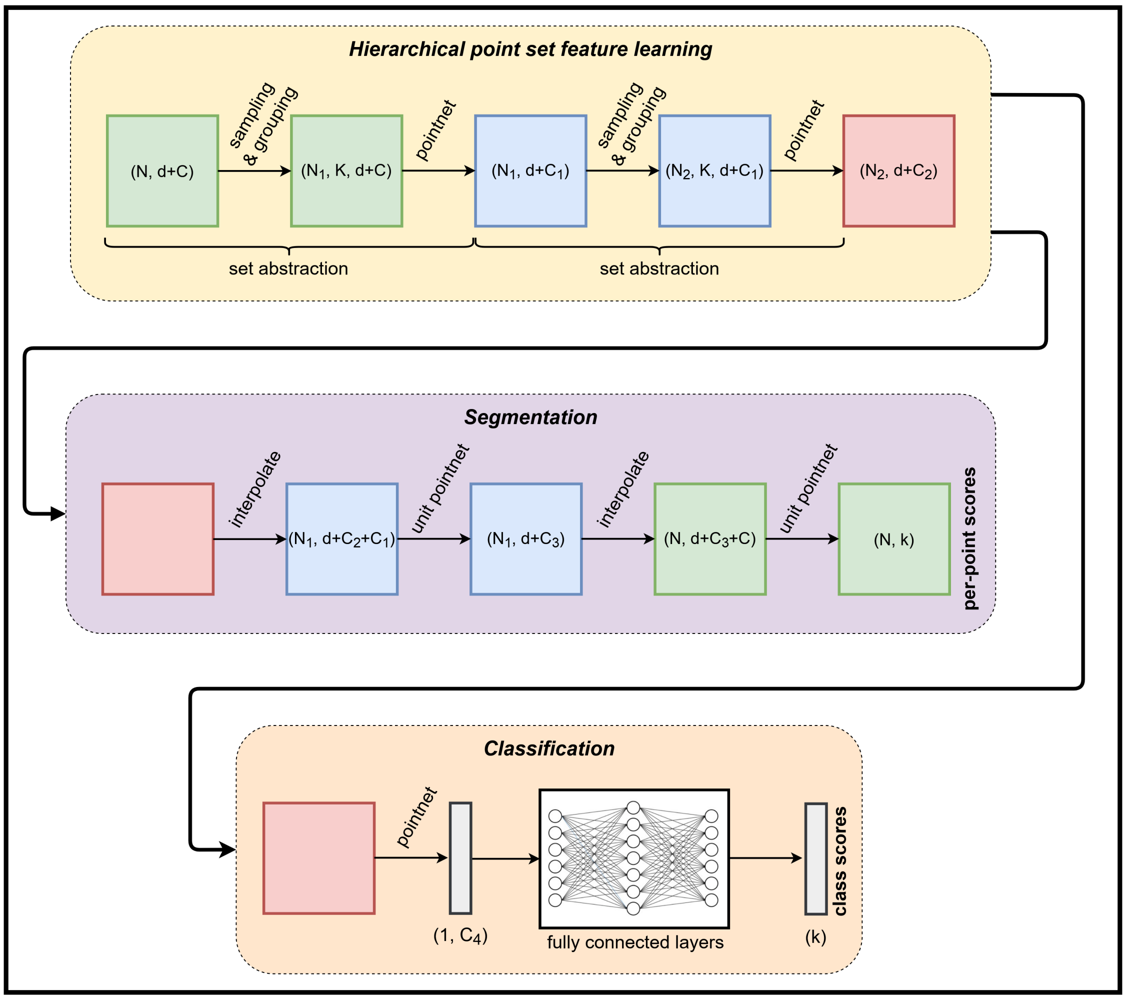


Figure 3.7: PointNet++ Architecture

to kNN, the local neighborhood of ball query fixes a determined scale of area and make the local feature more reliable and universal in space. For processes that use recognition of local pattern, this is the first choice (such as semantic point marking).

This layer takes as input a local area of N points with a data size of $NK(d + C)$. From the result every local region is decoupled from centroid of that region and the encoding centroid neighborhood's local attributes. The size of the output data is $N(d + C)$.

Each point must be classified in semantic segmentation, hence we must acquire the point features of all the original points. How to propagate the points travelling through the PointNet back to the initial point in reverse in the prior multi-scale Grouping PointNet. Using all of the original points as the center point in the Sample stage to integrate the information of the nearby points, which will result in greater computing costs. Return the features from the sub-sampling point to the starting point.

Distance interpolation and cross-level jump links are employed in a hierarchical propagation approach. The weights W and K nearest neighbor points f are used in this article to propagate the features from the sample point back to the original point, and the PointNet points are created using the feature. Based on kNN we use the average inverse distance weighted as one of the various interpolation possibilities. The following is the formula:

$$f^{(i)}(x) = \frac{\sum_{i=1}^{k} w_i(x) f_i^{(j)}}{\sum_{i=1}^{k} w_i(x)} \text{ where } w_i(x) = \frac{1}{d(x, x_i)^p}, j = 1, ..., C$$

PointNet++ proposes a new hierarchical architecture and beats PointNet by a large margin in 2D and 3D dataset. It also gives the better result than mature CNN architectures. Moreover, it improves robustness to the density with adaptive grouping.

3.4.3 PointConv

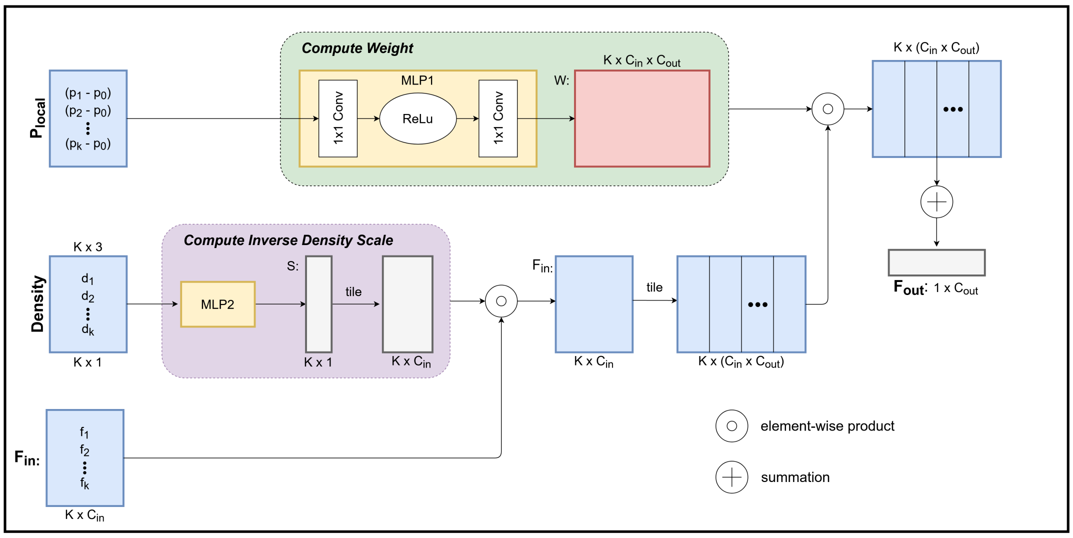


Figure 3.8: PointConv Architecture

PointConv [18] enhances existing image convolution applied over point clouds. It calculates approximate function of weight via MLP and then implements scale of density to re-evaluate the known weight function. PointConv develops a network to replicate continuous convolution weights. On raster pictures, traditional discrete convolution filters cannot be directly applied to the point cloud. To solve this problem, PointConv introduces continuous 3D convolution like below where $F(x + \delta_x, y + \delta_y, z + \delta_z)$ depicts a feature point centered by G around the point $p = (x, y, z)$.

$$Conv(W, F)_{xyz} = \iiint_{(\delta_x, \delta_y, \delta_z) \in G} W(\delta_x, \delta_y, \delta_z) F(x + \delta_x, y + \delta_y, z + \delta_z) d\delta_x \delta_y \delta_z$$

Since $(\delta_x, \delta_y, \delta_z)$ can possibly be any point of the local area the PointConv can be defined as -

$$\sum_{(\delta_x,\delta_y,\delta_z)\in G} S(\delta_x,\delta_y,\delta_z)W(\delta_x,\delta_y,\delta_z)F(x+\delta_x,y+\delta_y,z+\delta_z)$$

Here, $S(\delta_x,\delta_y,\delta_z)$ depicts density inverted at point $(\delta_x,\delta_y,\delta_z)$. The goal is to calculate the approximate weight function W(x, y, z) by 3D multilayers $(\delta_x,\delta_y,\delta_z)$ and the inverse density function $S(\delta_x,\delta_y,\delta_z)$. As the MLP weights are allotted to every point, the permutations invariance gets maintained. Feeding the input where local points $P_{local} \in R^{K\times 3}$ and feature $F_{in} \in R^{K\times C_{in}}$ we get the output as -

$$F_{out} = \sum_{k=1}^{K}\sum_{c_{in}=1}^{C_{in}} S(k)W(k,c_{in})F_{in}(k,c_{in})$$

where C_{in}, C_{out} defines the quantity of channels for input-output feature and a 1x1 convolution has been used for the MLP.

Chapter 4

Implementation

4.1 Dataset

We created our own dataset which contains 1900 facial images of 100 different person. Each person has about 19 images and each image was taken every 10° angular deviation from -90° to +90°. 10 images from 0° to +90° and 9 images from -10° to -90°.

4.1.1 Data Sample

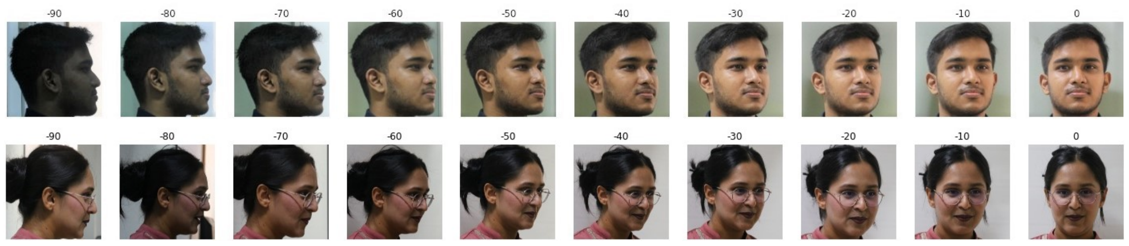


Figure 4.1: -90 degree to 0 degree angular deviation

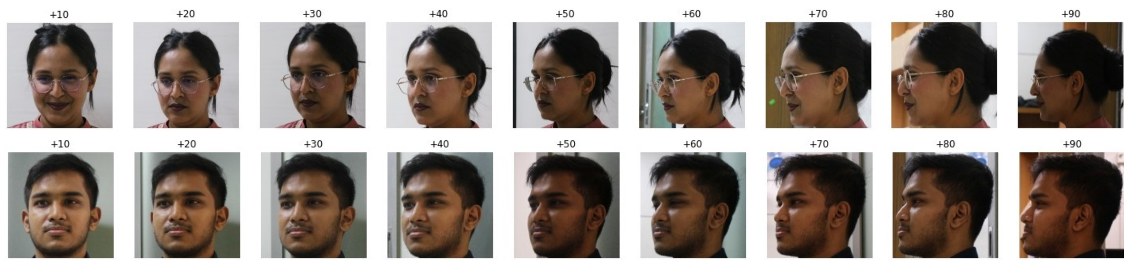


Figure 4.2: +10 degree to +90 degree angular deviation

4.1.2 Why create our own dataset?

As mentioned before for our data training we need to make a 3D mesh of each sample that's why we need images of a person of different angular deviation. Our created dataset consists of -90 degree to +90 degree angular deviation which gives a perfect 3D mesh of a person for our training purpose.

4.2 Data Visualisation and Pre-processing

4.2.1 Histogram of an image

Histogram of an image is a graphical plot which shows the frequency of pixels in a Gray Scale image. It has range of pixel values from 0 to 255. Here we need Grayscale image because it has only information about intensity where pixel value's range is between 0 to 255. Pure black is the weakest intensity and pure white is the strongest intensity in a Grayscale image. Histogram consists of 2 axises. X axis represents pixel values from 0-255 and on the other hand, Y axis represents frequency of the specific pixel value in the image. Our histogram graph shows from the left side of the Y axis, we can see pixel count and from the right side we can see percentage of pixel count.

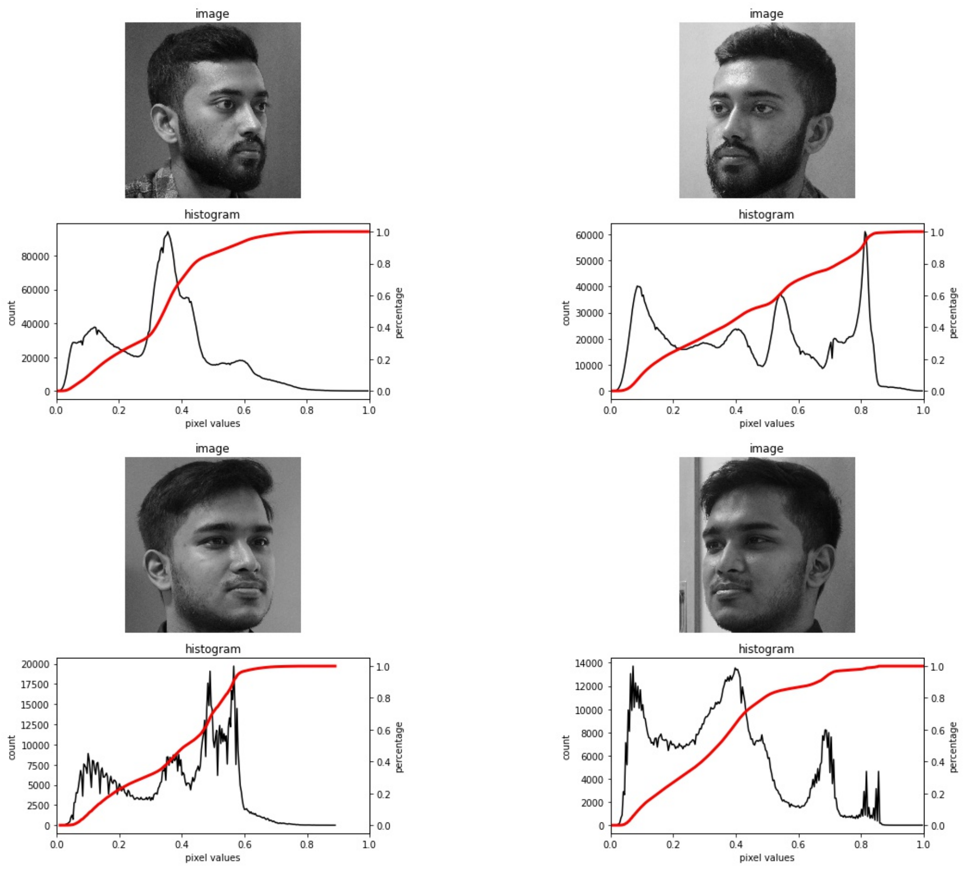


Figure 4.3: Histogram of sample dataset

4.2.2 Histogram Equalization

The process of creating a higher contrast of an image by adjust the intensity values of the image using image's histogram is called Histogram equalization [7]. If the image is low contrasted we can use histogram equalization to increase the global contrast of images. With the help of histogram equalization we can distribute intensities in a better way in the image's histogram. It increases the lower contrast area to higher contrast. It is useful for our those images which has both bright or both dark in backgrounds and foregrounds.

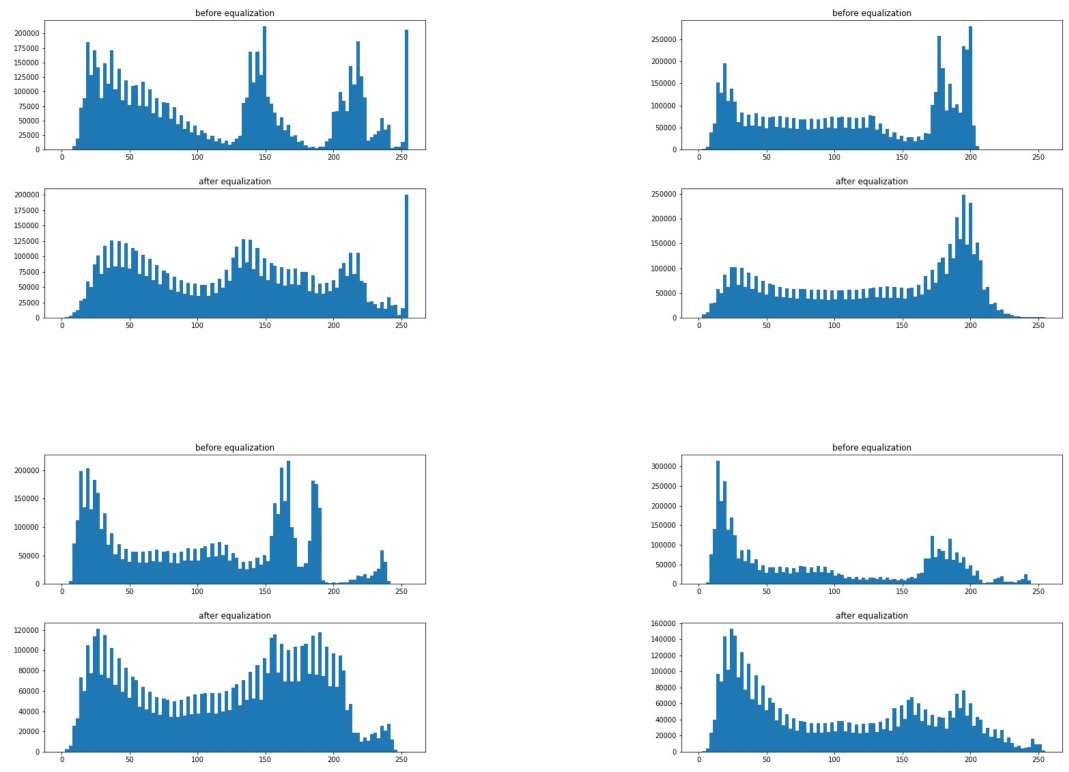


Figure 4.4: Before and after applying Histogram Equalization

We can see from the histograms and second one is more equalized. One important thing is Histogram equalization is better than histogram stretching because histogram equalization can changes the overall shape of histograms.

4.2.3 Applying CLAHE

CLAHE [9] stands for Contrast Limited Adaptive Histogram Equalization which is one of the variants of Adaptive histogram equalization which is called AHE. It is important to improve the situation of over amplification of the contrast. Rather than operating in the entire image CLAHE operates in the small region of the image which is called tiles. To remove the artificial boundaries neighboring tiles are combined by bilinear interpolation. It improves the image's contrast than histogram equalization. In CLAHE 2 parameters are important. One is clipLimit which sets the threshold for contrast limiting. Another is tileGridSize which sets the tile's number in row and column. Tiles division is done before applying CLAHE.

We use CLAHE bcause CLAHE is more advance than other AHE variants. If our images have over amplification it prevents that problem. CLAHE also forms a transformation function to reduce the noise problem that is not taken care of by histogram equalization.

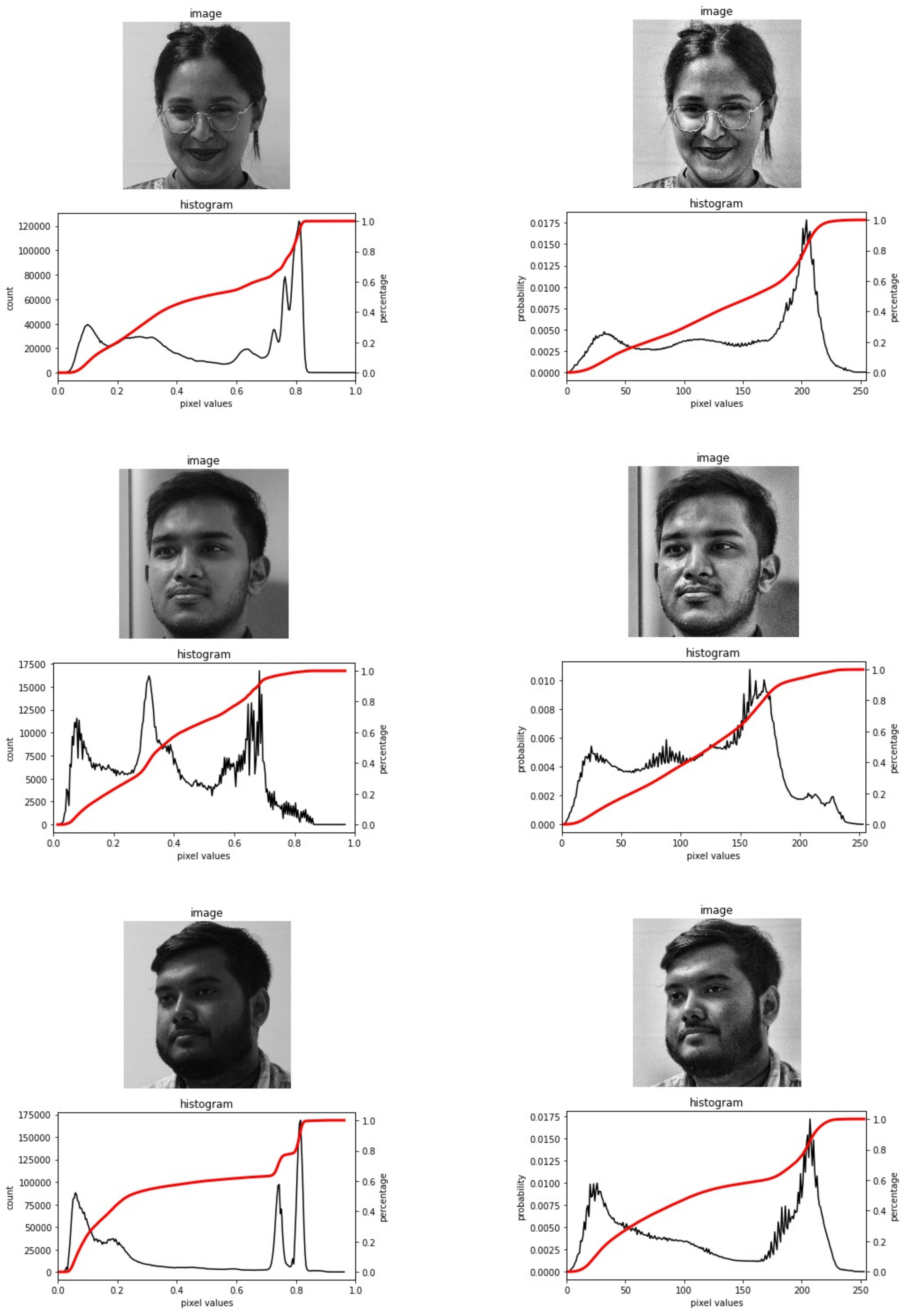


Figure 4.5: Before and after applying CLAHE

4.3 Face Detection & Cropping

The first step towards facial recognition of a person is detecting the face from an image. Different face detection algorithms have been discussed previously in chapter 2. We want to draw a bounding box around the face in an image. And calculate the accuracy of the predicted box using different metrics. For any face detection algorithm, the process is to run a filter on the image and search for the pattern. In the initial stage, it tries to detect small patterns and then tries to combine those patterns and looks for complex portions of the face: eyes, lips, nose, etc. In our work, we would like to apply the HaarCascade face detecting algorithm. The algorithm uses a cascading window and tries to compute features like eyes, and nose in an image and classifies it. The operation has been shown below:

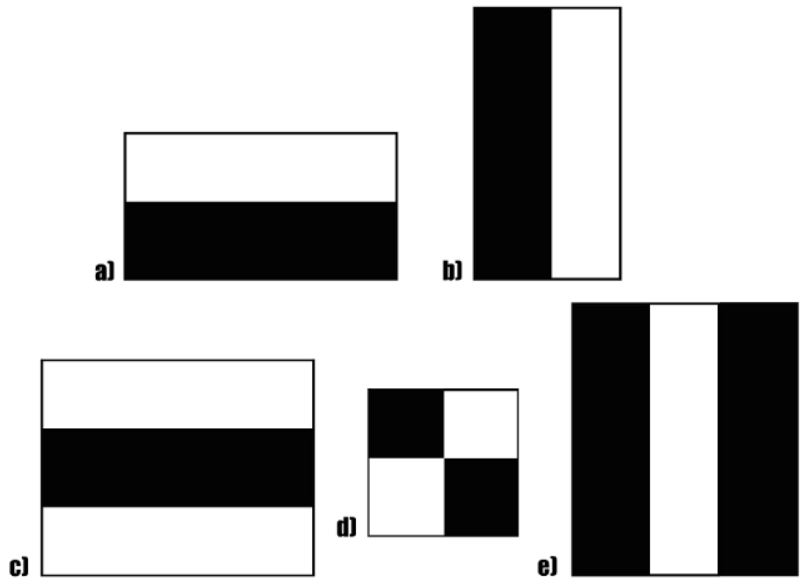


Figure 4.6: Haar Features that were mentioned in the original paper of HaarCascade

The features will detect different portions separately and then identify the face and draw a square box around the face. The reason for choosing this algorithm instead of a deep learning-based algorithm is that it works fast because of its simplicity. And since detecting the face is only one portion of the whole process we need it to be efficient as possible. Although the algorithm is prone to false-positive when the image contains multiple faces, in our work that isn't an issue.

Using HaarCascade, we detected faces in images and cropped the unnecessary elements out of the photo. After cropping the unnecessary elements, only the face remains in the image.

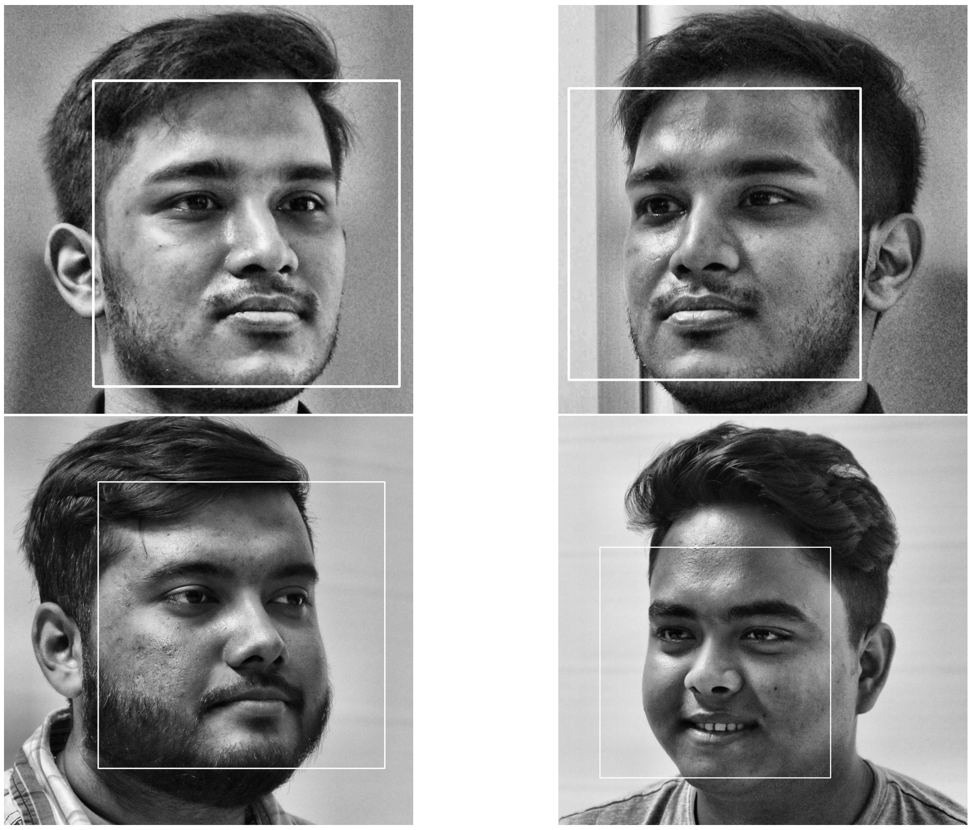


Figure 4.7: Face detection using HaarCascade

4.4 3D Face Reconstruction

4.4.1 SFM

After extracting the face using the above-mentioned algorithm we try to construct a 3D model of the face. For this, we use Structure From Motion pipeline which includes algorithms such as SIFT, DOG, ANN, and RANSAC. First, we try to find the prominent feature in each image and try to match them across different images. This is done using the Scale-invariant feature transform algorithm. Afterward, we try to construct a 3D point cloud from those features. The algorithm maps the points to the correct location and scaling factor using triangulate. The steps have been show below:

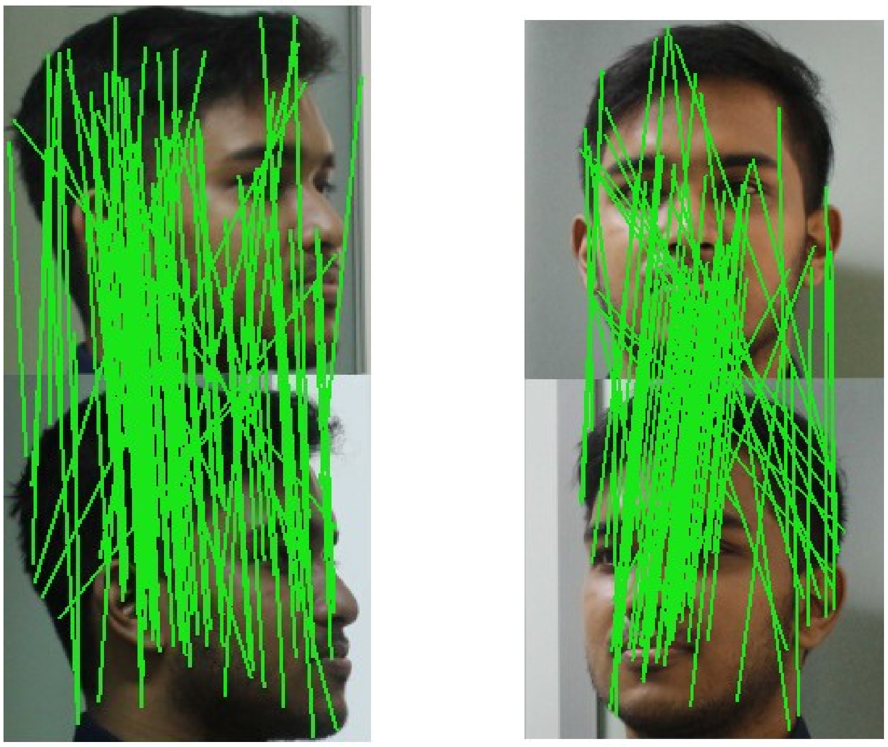


Figure 4.8: Face detection using HaarCascade

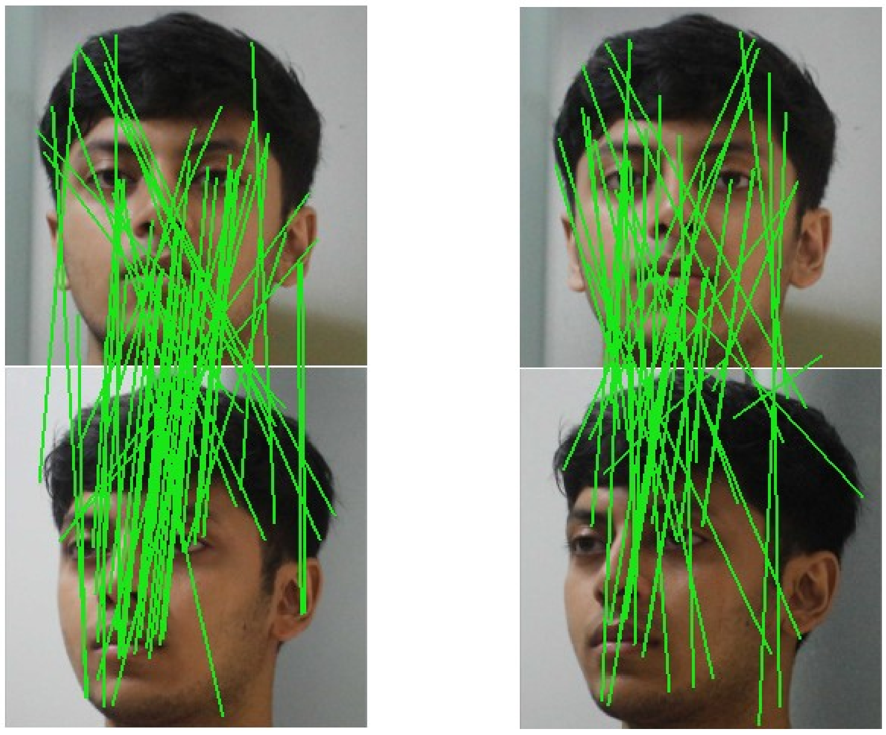


Figure 4.9: Face detection using HaarCascade

4.4.2 NextFace

We implemented NextFace using PyTorch with a PC which had NVIDIA GeForce GTX 1080ti GPU and Intel i7 7700 . NexFace uses Ray tracing to trace features of the face. For optimization we used Adam with default $\beta1 = 0.9, \beta2 = 0.999$ and

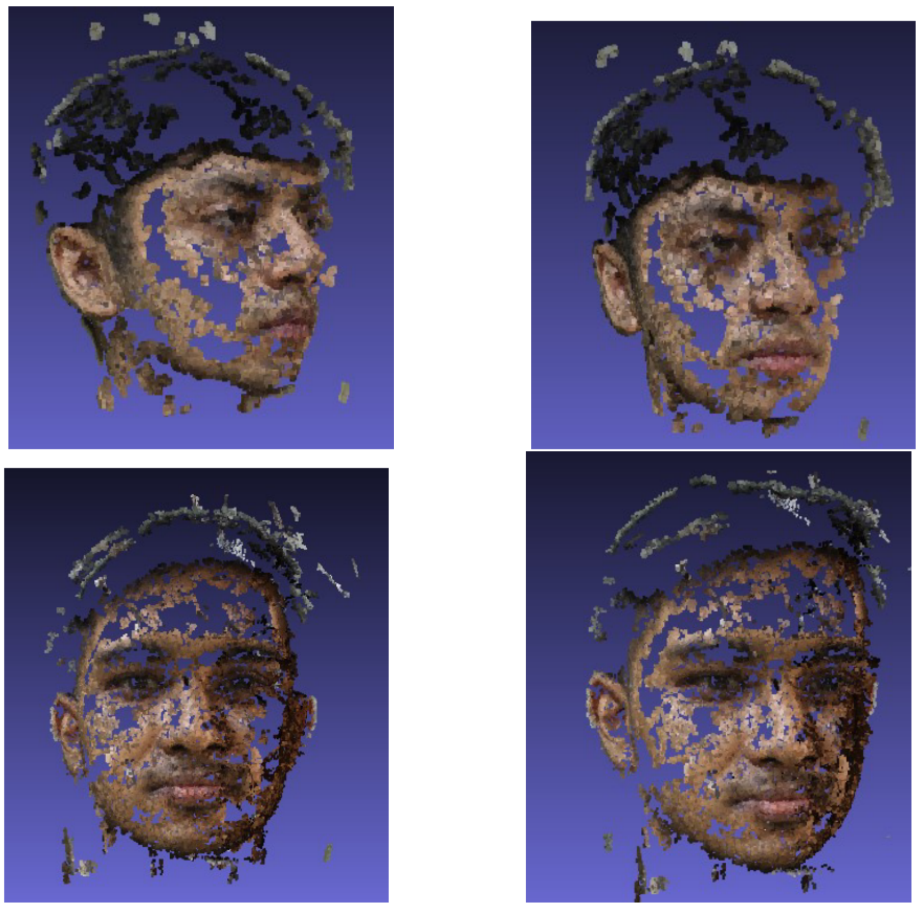


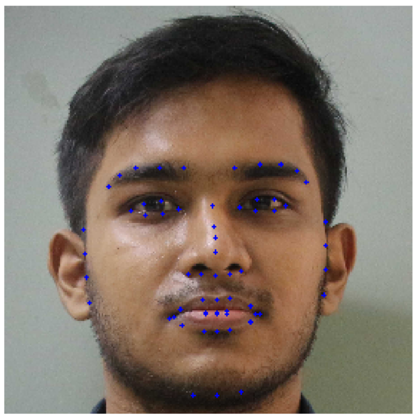
Figure 4.10: Face reconstruction using SFM

$\alpha 1 = 1$. For each parameter, we used different learning rates so that all parameters are weighted equally. The values of Learning Rates for different cases used by us are given below -

Learning Rates	**Value**
Light Stage Parameters	0.001
Statistical Albedo	0.02
Shape Identity	0.01
Camera Rotation, Translation and Blendshapes	0.001
Diffuse, Specular and Roughness	0.005

To reconstruct a 3D mesh from a single image with a PC of the above mentioned configuration, it took about 6.7 minutes for the full optimization, where Stage I takes 0.3 minutes, Stage II takes 5.1 minutes and Stage III takes 1.3 minutes. As this architecture is dependant on GPU performance, we could've got a better result by using latest GPU models and it can finish one reconstruction within two minutes.

We reconstructed 1900 3D meshes from 1900 facial 2D images of our dataset. In the 3D dataset, each subject from our 100 subjects has 19 3D meshes.


(a)

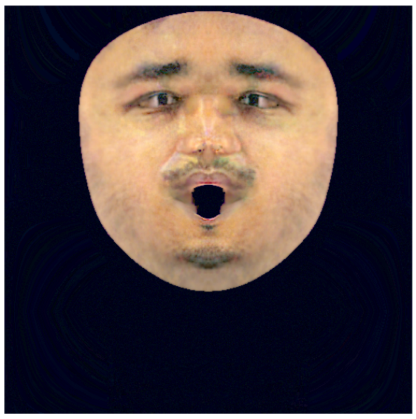
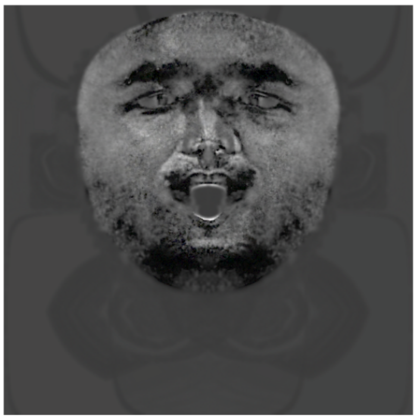

(b)

Figure 4.11: (a) Landmarks of the face (b) Estimated diffuse map in UV space


(c)

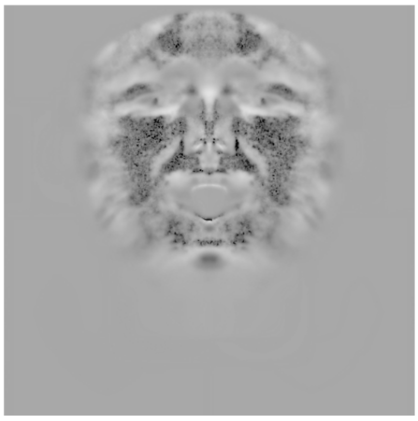

(d)

Figure 4.12: (c) Estimated specular map in UV space (d) Estimated roughness map in UV space

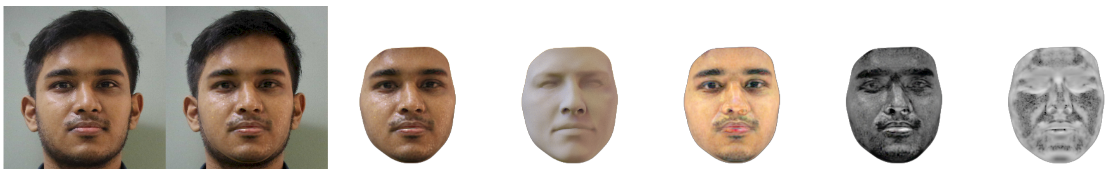
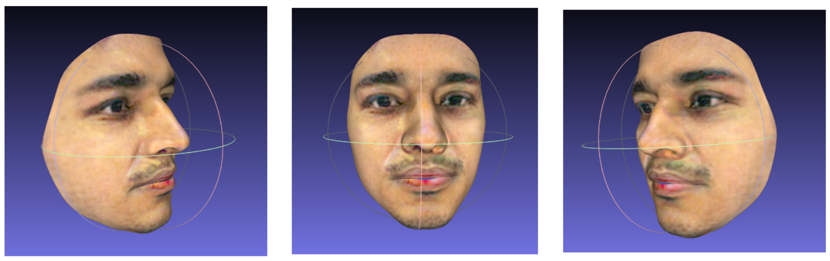
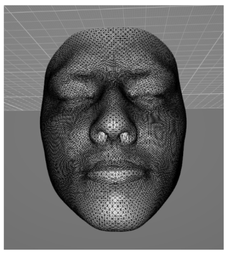


Figure 4.13: From left to right: input image, overlay of the final reconstruction on the input image, the final reconstruction, diffuse, specular and roughness maps projected on the face



(a)



(b)

Figure 4.14: (a) Reconstructed 3D Face (b) Reconstructed 3D Mesh of the Face

4.5 2D Architecture Training

4.5.1 VGG16

We have implemented the VGG16 model for 2D face recognition. With the help of images from ImageNet dataset, we pre-trained this model and retrieved the facial landmarks from the facial images. We intended this architecture to be capable of identifying faces outside the dataset. Because of this, we decided against using a softmax classifier as the last layer. Shape (224,224) is the input for this architecture, which produces a feature vector including the face points from each image. These vectors shapes include (1, 7, 7, 512). These data points are kept in a data frame, and the file is saved locally. We no longer have to train the model each time a brand-new subject is added, which saves us a ton of time. We utilized 1300 images of 100 people for training, or 68.421 percent of our dataset. We utilized 600 photos of 100 people for testing, or 31.579 percent of our dataset. In order to compare the attributes with the test photos, we pull from the previously recorded data frame. Our distance-based matching criteria are as follows:

Subsequently, using feature points from the test photos, we cross-checked them against feature vectors already in use. By computing the minimal distance, we were able to identify the target image's class. We established a threshold of 450. The minimal distance is classified as outside of the dataset if it is bigger than the threshold value.

4.5.2 ResNet50

We implemented the ResNet50 using transfer learning to test on our 2D dataset. We have used the pre-trained weights from ImageNet dataset which was trained with 1000 classes. For the training purpose we have used the architecture as a feature extractor. We currently have 100 classes and if we trained using these classes and afterwards add an additional class we have to train the model again which is costly so we want to extract the facial features and save them in our dataset and for additional class we can simply extract the feature vectors and add them to our dataset. Therefore, this approach saves our time and computation cost. For the face recognition part we calculated the minimum distance of the feature vector and classify them to which has the minimum distance. The model takes (24,24,3) shape images and outputs a vector shape of (1, 7, 7, 512). The other parameters are same as VGG16 model.

4.6 3D Architecture Training

4.6.1 PointNet

We have trained neural networks to predict the persons in our 3D point cloud dataset. The outputs of SFM has certain drawbacks as there are voids in middle of faces so we lose some of the facial landmarks. On the other hand NextFace produces high quality point cloud which we have used in this step. We have divided the dataset into two parts one for training which accounts for 80% of the data and the rest for testing purposes. Also to eliminate overfitting in the model we have divided the training set into 4 folds. Each fold contains an equal number of data points. For cross-validation, we used one of the folds for testing and the rest 3 for training. This method reduces the chance of overfitting the model. Afterward, we tested and evaluated the models using the test dataset. We have used PointNet architecture which is discussed earlier to train our model. The architecture takes a set of point clouds (x_i, y_i, z_i) as input. For the classification task, the model takes the inputs and passes them through a symmetric function to make them invariant to permutation. Afterward, using a single multi-layer perceptron all the points are mapped to a higher dimension and it performs max pooling to get the global feature vector. Finally, it passes to a fully connected layer which outputs the probability of the output class. For our training, we have used softmax as our activation function as it performs better in the multi-class classification tasks. For the loss function, we have used sparse categorical loss entropy. We have used the batch size of 36 and epochs of 40. We have trained the model in various cases, as our target is to maximize the accuracy at an angular deviation such as angles 40°, -40°, 50°, -50°, 60°, -60°, 70°,-70°, and so on. We have replaced the testing dataset with these angles so that we can use our model to recognize faces in such angles.

Chapter 5

Result and Analysis

5.1 Metrics Used

After training all the architectures and taking the best performing models we conclude our result below. To measure the outputs we have used the following metrics:

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{5.1}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{5.2}$$

$$F_1 = \frac{2 \cdot TruePositive}{2 \cdot TruePositive + FalsePositive + FalseNegative} \tag{5.3}$$

$$Accuracy = \frac{CorrectlyPredicted}{TotalTestImages} \tag{5.4}$$

5.2 Performance Evaluation

5.2.1 Comparison between Architectures

Metrics	**VGG16**	**RestNet50**	**PointNet**
Accuracy	89.123%	90.175%	88.349%
Precision	0.872	0.906	0.8967
Recall	0.881	0.9017	0.8620
F1	0.8764	0.9038	0.8602

This table shows the comparison in terms of front images. We can see that in terms of accuracy and F1 score VGG16, ResNet50 performs better than PointNet.

5.2.2 Comparison in Different Angles

Angles	2D Model Accuracy	3D Model Accuracy
-40 to +40	90.175%	88.35%
±50 to ±70	80.4532%	85.221%
±80 to ±90	73.372%	84.312%

The accuracy drops in angular deviation for 2D architectures where as the accuracy of PointNet remains almost consistent.

5.2.3 Graphical Analysis of Models

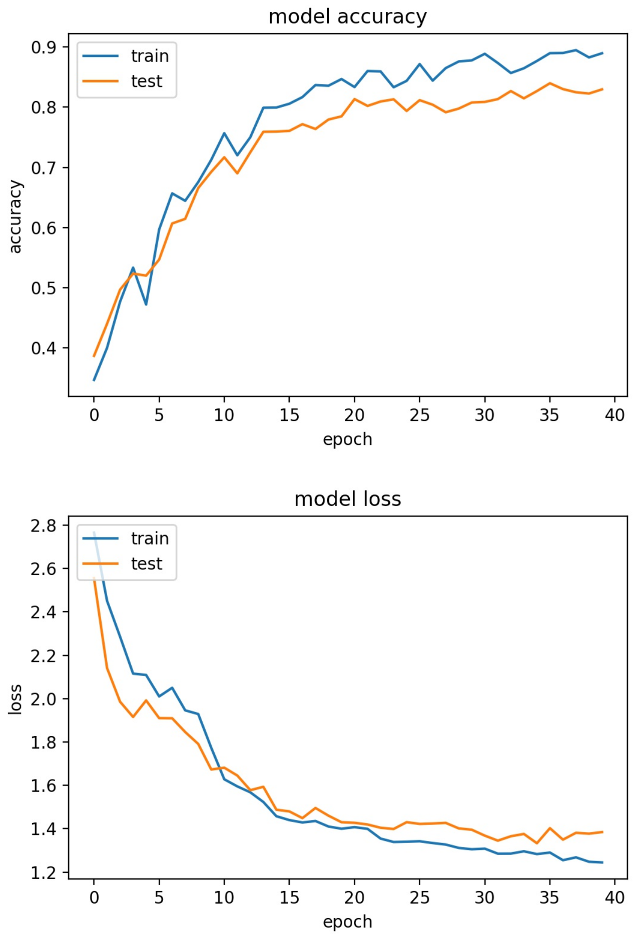


Figure 5.1: PointNet Training and Validation Accuracy and Loss

The first curve represents the accuracy during each epoch for training and validation. We can see the validation curve is almost same as training curve which means the model is not overfitted or underfitted.

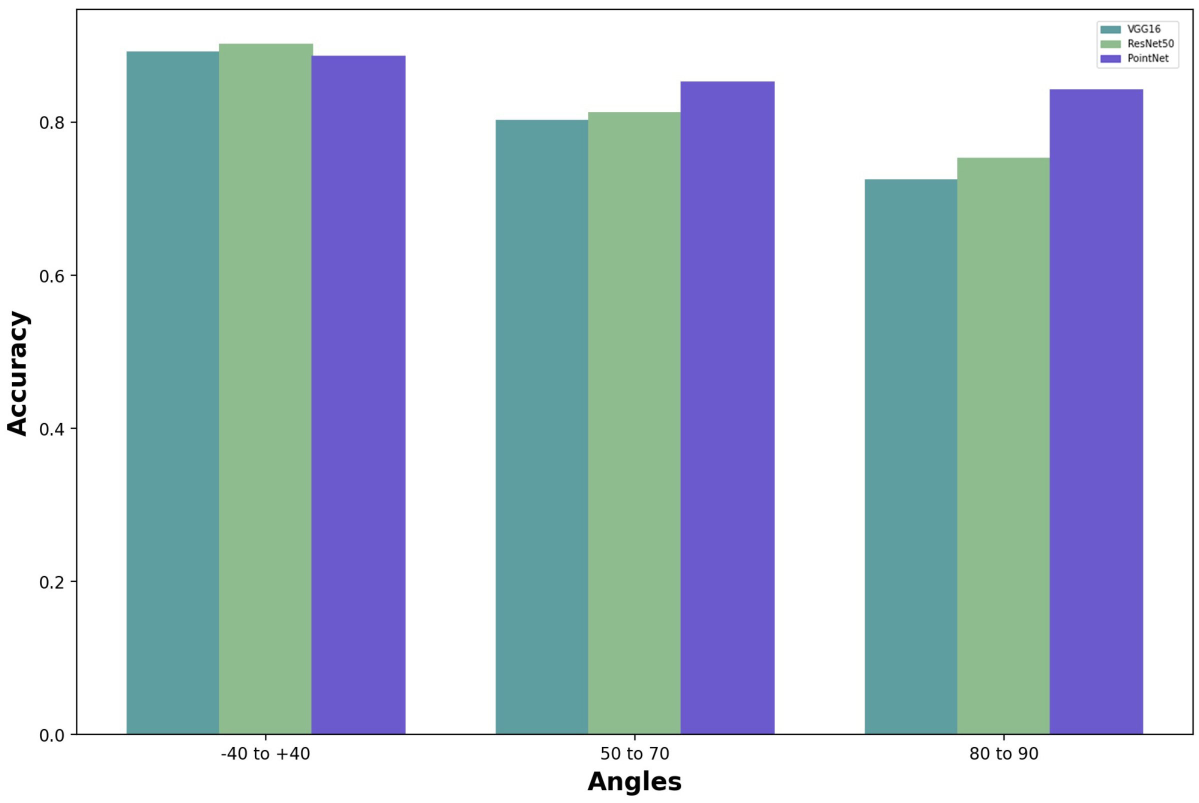


Figure 5.2: Comparison of models in different angles

The above bar chart illustrates the comparison among the models. In front angles 2D models performs significantly well but in case of ±40°, ±50°, ±60°, ±70° angles the 3D architecture performs better and in ±80°, ±90° angles it out performs VGG16, ResNet50 quite significantly.

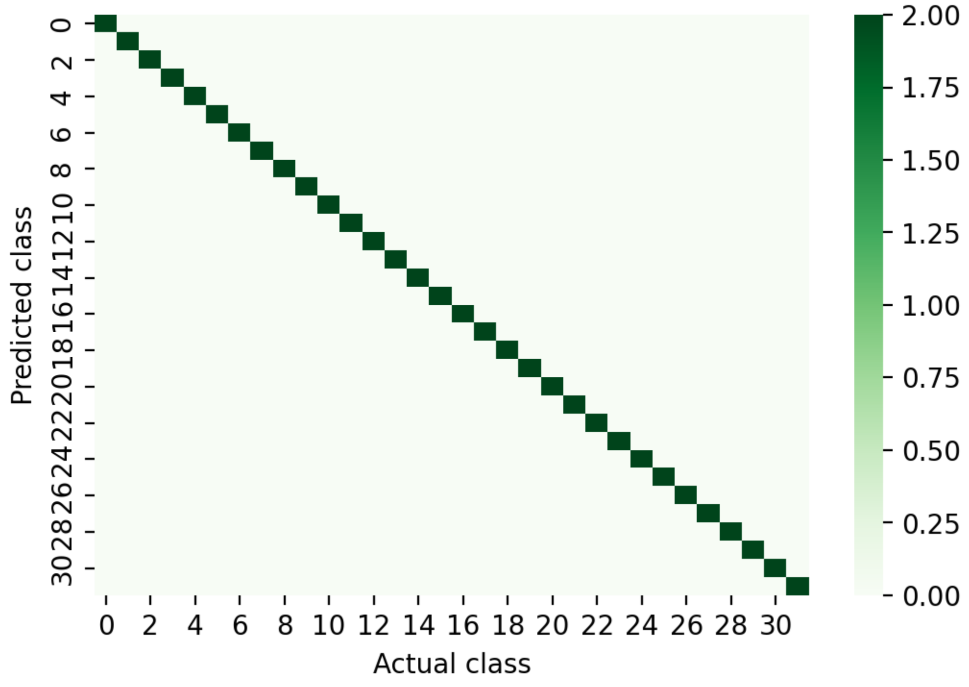


Figure 5.3: Confusion Matrix for VGG16

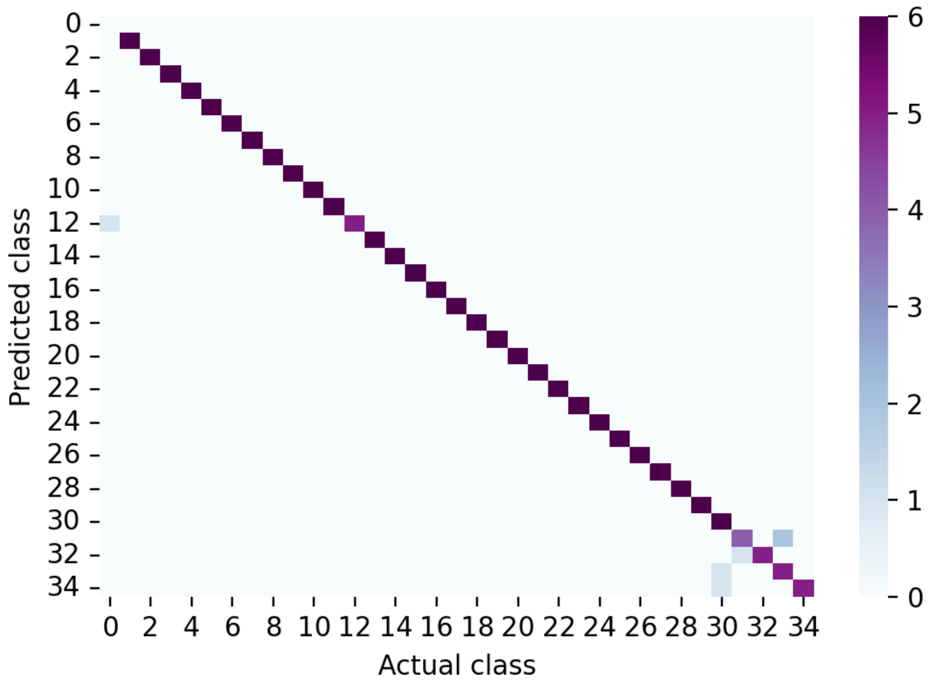


Figure 5.4: Confusion Matrix for ResNet50

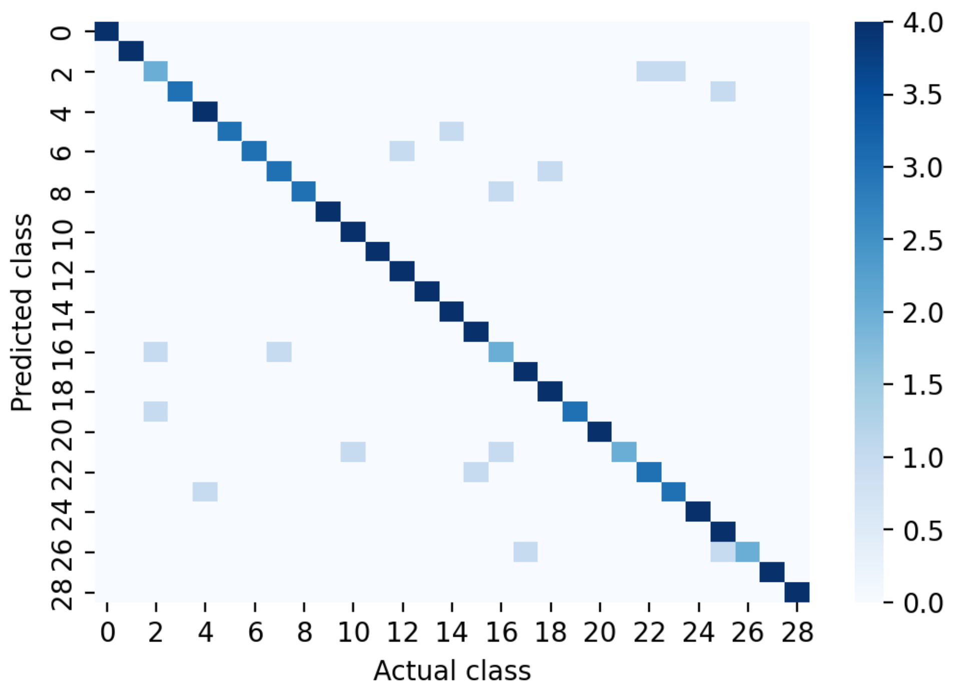


Figure 5.5: Confusion Matrix for PointNet

The confusion metric draws the True Positive, False Positive of different classes from the output.
Furthermore, we can conclude that the outputs from SFM pipeline is not good as that of NextFace. The outputs from SFM misses a lot important facial landmarks. Also it takes a lot of image as input to produce one single 3D mesh. On the other hand, NextFace overcomes these problems. The outputs generated from this pipeline contains most of the facial points and it can provide one point cloud from one single image. Also in the training part we can conclude that 2D models perform better in frontal images whereas it drops accuracy in angular images and when a person is in motion. But the 3D deep learning pipelines works better in this case and overcomes the problems stated above.

Chapter 6

Future Work and Conclusion

6.1 Future Work

Firsty, In our proposed system, for 3D face recognition using PointNet, if a new subject is added to the dataset, the whole model needs to be trained again. We were able to overcome this problem in 2D face recognition with VGG16 and ResNet50 by using feature vectors. In the future, we would like to implement this type of training model in 3D face recognition as well so that if a new subject is added in the dataset, it would not have to train the whole model. We would also like to implement this system using other 3D architectures such as PointNet++ so that we can compare the models and use the best one.

Secondly, it takes almost 5 minutes to reconstruct 3D face from a single 2D image using our PC with the configuration of Intel Core i7 7700 and Nvidia GeForce GTX 1080ti. If we could've used modern GPUs and CPUs like Intel Core i9 11900 and Nvidia GeForce RTX 3090, the time should be less than 2 minutes. But it is still far from optimnized. In the future, we plan to use Nvidia Cuda Computing and Parallel Computing so that we can reconstruct 3D faces in real time.

Finally, we used Canon EOS 750D for data collection. Our system performs better with high quality pictures. We plan to update our system so that it can be implemented using low quality cameras like CCTV camera.

6.2 Conclusion

Nowadays security is a big issue and in the world of machine learning and neural networks, face recognition is still a rare feature in so many fields because of its accuracy and lack of user friendly features. So unlike fingerprint and other recognition techniques face recognition is very fast and automatic. The current ongoing works in face recognition are doing a good job but they mainly focus on 2D images which have a certain downside. They fail to provide an accurate result and cannot recognize people in motion. Our described models already outperforms the 2D architectures in certain angles and with the evolution of graphical processing unit and parallel computing we can detect, reconstruct and recognize face without any buffer. The process can provide even better results in real-time.

Bibliography

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," vol. 1, Feb. 2001, pp. I–511, ISBN: 0-7695-1272-0. DOI: 10.1109/CVPR.2001.990517.

[2] C. Xu, T. Tan, and L. Quan, "A new attempt to face recognition using 3d eigenfaces," *Proceedings of the Asian Conference on Computer Vision*, vol. 2, Jan. 2004.

[3] Y. Guan, "Automatic 3d face reconstruction based on single 2d image," Jan. 2007, pp. 1216–1219. DOI: 10.1109/MUE.2007.95.

[4] T. Fabry, D. Vandermeulen, and P. Suetens, "3d face recognition using point cloud kernel correlation," Nov. 2008, pp. 1–6. DOI: 10.1109/BTAS.2008.4699359.

[5] M. Mayo and E. Zhang, "3d face recognition using multiview keypoint matching," Sep. 2009. DOI: 10.1109/AVSS.2009.11.

[6] G. Stylianou and A. Lanitis, "Image based 3d face reconstruction: A survey," *International Journal of Image and Graphics*, vol. 09, Apr. 2009. DOI: 10.1142/S0219467809003411.

[7] O. Patel, Y. Maravi, and S. Sharma, "A comparative study of histogram equalization based image enhancement techniques for brightness preservation and contrast enhancement," *Signal Image Processing : An International Journal*, vol. 4, Nov. 2013. DOI: 10.5121/sipij.2013.4502.

[8] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014. DOI: 10.48550/ARXIV.1409.1556. [Online]. Available: https://arxiv.org/abs/1409.1556.

[9] G. Yadav, "Contrast limited adaptive histogram equalization based enhancement for real time video system," Sep. 2014. DOI: 10.1109/ICACCI.2014.6968381.

[10] D. Yi, Z. Lei, S. Liao, and S. Li, "Learning face representation from scratch," Nov. 2014.

[11] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. DOI: 10.48550/ARXIV.1512.03385. [Online]. Available: https://arxiv.org/abs/1512.03385.

[12] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," *Proc. CVPR*, Mar. 2015.

[13] C. Ruizhongtai Qi, H. Su, K. Mo, and L. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," Dec. 2016.

[14] R. Charles, H. Su, K. Mo, and L. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” Jul. 2017, pp. 77–85. DOI: 10.1109/CVPR.2017.16.

[15] M. Hernandez, J. Choi, and G. Medioni, “Deep 3d face identification,” Mar. 2017.

[16] C. Ruizhongtai Qi, L. Yi, H. Su, and L. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” Jun. 2017.

[17] S. Zhou and S. Xiao, “3d face recognition: A survey,” *Human-centric Computing and Information Sciences*, vol. 8, Dec. 2018. DOI: 10.1186/s13673-018-0157-2.

[18] W. Wu, Z. Qi, and F. Li, “Pointconv: Deep convolutional networks on 3d point clouds,” Jun. 2019, pp. 9613–9622. DOI: 10.1109/CVPR.2019.00985.

[19] P. B and M. J, “Design and evaluation of a real-time face recognition system using convolutional neural networks,” *Procedia Computer Science*, vol. 171, pp. 1651–1659, Jan. 2020. DOI: 10.1016/j.procs.2020.04.177.

[20] Y. Kim, W. Park, M.-C. Roh, and J. Shin, “Groupface: Learning latent groups and constructing group-based representations for face recognition,” Jun. 2020, pp. 5620–5629. DOI: 10.1109/CVPR42600.2020.00566.

[21] S. Prova, S. Mehzabin, M. Mahmud, and M. Alam, “Machine learning approach for face recognition from 3d models generated by multiple 2d angular images,” Dec. 2020, pp. 1–3. DOI: 10.1109/CSDE50874.2020.9411541.

[22] M. Kholil, I. Ismanto, and M. Fu’ad, “3d reconstruction using structure from motion (sfm) algorithm and multi view stereo (mvs) based on computer vision,” *IOP Conference Series: Materials Science and Engineering*, vol. 1073, p. 012 066, Feb. 2021. DOI: 10.1088/1757-899X/1073/1/012066.

[23] A. Dib, J. Ahn, C. Thebault, P.-H. Gosselin, and L. Chevallier, *S2f2: Self-supervised high fidelity face reconstruction from monocular image*, 2022. DOI: 10.48550/ARXIV.2203.07732. [Online]. Available: https://arxiv.org/abs/2203.07732.