Multimedia Tools and Applications
https://doi.org/10.1007/s11042-020-09008-z

Point cloud based deep convolutional neural network for 3D face recognition

Anagha R. Bhople[1] · Akhilesh M. Shrivastava[1] · Surya Prakash[1]

Received: 15 November 2019 / Revised: 6 April 2020 / Accepted: 1 May 2020 /
Published online: 03 June 2020

Abstract

Face recognition is a challenging task as it has to deal with several issues such as illumination, orientation, and variability among the different faces. Previous works have shown that 3D face is a robust biometric trait, and is less sensitive to light and pose variations. Also, due to availability of inexpensive sensors and new 3D data acquisition techniques, it has become easy to capture 3D data. A 3D depth image of a face is found to be rich in information, and biometric recognition performance can be enhanced by using 3D face data along with convolutional neural network. However, the shortcoming of this approach is the conversion of 3D data to lower dimensions (depth image), which suffer from loss of geometric information, and the network becomes computationally expensive. In this work, we endeavor to apply deep learning method for 3D face recognition and propose a deep convolutional neural network based on PointNet architecture which consumes point cloud directly as input and siamese network for similarity learning. Further, we propose a solution to the issue of a limited database by applying data augmentation at the point cloud level. Our proposed technique shows encouraging performance on Bosphorus and IIT Indore 3D face databases.

Keywords 3D face recognition · Point cloud · Deep learning · PointNet · Biometrics

1 Introduction

Biometric methods are responsible for measuring particular human characteristics and determining the identity of a person based on these characteristics. With the outgrowth in

✉ Anagha R. Bhople
ms1804101008@iiti.ac.in

Akhilesh M. Shrivastava
phd1701101001@iiti.ac.in

Surya Prakash
surya@iiti.ac.in

[1] Indian Institute of Technology Indore, Indore 453552, India

technology, biometric techniques have made extraordinary leaps. Among various biometrics, face recognition is found to be a reliable biometric technique for identification and verification. It has become one of the most preferred biometric trait [6] and has several applications in fields like security, surveillance, finance, and many other applications in our day to day life.

The 2D face recognition performance has got highly influenced by the advancement in deep learning and the accessibility of massive training data. It has been seen that a profound neural network trained on numerous labeled training data demonstrates much better outcomes than standard techniques of face recognition like Eigenfaces [48], Fisher faces [39], local binary patterns [4], etc. A deep convolutional neural network (CNN) trained on benchmark databases like Labeled Faces in the Wild (LFW) [20] and Janus [23] has shown remarkable results. Also, the CNN based networks like DeepFace [45] and FaceNet [41] are milestones in computer vision. The recognition accuracy of DeepFace on the LFW database is found to be 97.35%, whereas for the FaceNet, it is found to be 99.63%. In Google FaceNet model, researchers have delivered a new technique for training the deep network and have introduced a triplet loss. This network not only performs classification but can even learn dissimilarity among two faces. Deep learning is proven to be one of the best tools for a 2D face recognition task. Motivated by this fact, we have extended this approach for 3D face recognition.

Though 2D face recognition has achieved phenomenal performance in the field of biometrics, there are still many obstacles with 2D face recognition. For example, the performance of 2D face recognition is affected due to the presence of illumination, pose, occlusion, and facial expression variations. Moreover, the texture, skin color, image resolution are not always the same in 2D images and suppresses the performance [18]. The acquired images are of poor quality when they are captured in an unconstrained environment, and 2D face recognition fails to achieve reasonable accuracy in such a scenario.

3D face recognition has the ability to handle drawbacks of 2D face recognition and also performs superior than its 2D counterpart. It deals with the 3-dimensional data, which is capable of representing all the facial features. Further, 3D model [3] preserves anatomical structure rather than texture or color of the face and provides an absolute representation of 3D data, which in turn improves the discriminating power of face recognition. 3D data has the potential to retain all the geometric and topological information of facial structure and is found to be invariant to lighting conditions, geometric transformations, and expression changes. Surveys presented in [1, 3, 52] shows that the 3D face data is robust against changes in pose and illumination, and it is found to be less sensitive to isometric deformations and facial orientation changes. With the advancement in the 3D data acquisition technique, open 3D databases are increasing day by day.

The recent studies in 3D face recognition demonstrate the use of depth or range images for 3D face recognition task. In [21], Kim *et al.* have proposed deep learning based 3D face recognition, and comparable outcomes have been accomplished against state-of-the-art techniques. In [53], Zulqarnain *et al.* show a method of generating hundreds of synthetic images for large scale training. Though these deep learning methods outperform the conventional approaches of 3D face recognition, they possess certain limitations. Most of the 3D face recognition techniques rely upon 3D point cloud transformation to 2.5D information, known as depth or range image. Though depth images have the potential to capture prominent facial features, still it does not make full use of the spatio-temporal features available in the 3D data.

The work in [15] shows an application of 3D-CNN for object recognition, where the input to the network is a voxel grid. The 3D data representation using a voxel grid divides the object volume into a 3D grid. Though this representation uses complete geometric information of an object, there are some disadvantages of this approach. For example, conversion of non-euclidean point cloud data to volumetric form makes the data unnecessarily voluminous and increases the cost of storage. Further, volumetric data may be immensely vast due to which it may take a lot of processing time. Moreover, this approach cannot be applied for face recognition as grid resolution fails to represent detailed facial features, and usage of high-resolution grid structure costs a massive amount of computational resources and memory. Hence, we need an appropriate data format for effective 3D face recognition. We explore the use of direct point cloud for this.

A point cloud is a set of 3D data points $\{Pi|i = 1, 2, .., N\}$ defined in 3D space, where each point Pi is represented as (x,y,z) by using its position in terms of Cartesian coordinates. It represents the exterior surface of an object in the form of discrete points lying on the surface of the object and is generally produced by 3D scanners. The point cloud is a non-euclidean 3D data representation [3]. It expresses explicit geometry in the form of 3D coordinates from which shape and size of the object can be determined, unlike 2D data, which gives limited information about the shape and size of the object. It is a straightforward yet effective way of representing 3D data. It preserves all the geometric information of the face. Nowadays, it is easily possible to create a large scale 3D database as a number of 3D scanners [3] are available in the market at reasonable rate. Integrating mobile phones such as iPhone with new miniature scan devices has opened new dimensions for the 3D data acquisition. The point cloud is considered as the geometric data structure for any 3D object. One of the main advantages of the point cloud is that each sensor produces a basic set of (x,y,z) points for any object.

Usage of direct point cloud provides numerous benefits in face recognition. With recent advances in deep learning, there are a few networks like PointNet [37] and PointNet++ [38] which provide an architecture for direct use of point cloud. These architectures consume point cloud data directly as an input and also considers permutation, scale, and transformation invariance of the point cloud. The purpose behind these networks is to develop a deep neural network based architecture, which can segment and classify objects based on their 3D point clouds. PointNet is an unified architecture inspired by the VoxNet [30] and ShapeNet [50]. The results achieved by the PointNet on ModelNet40 [50] database are pathbreaking and sets a new research area for the usage of the point cloud.

It is computationally inexpensive to access raw point cloud as compared to the voxel grids or occupancy grids [15]. Due to this, we use a direct point cloud as an input for the task of 3D face recognition. We propose a PointNet based on a deep convolutional neural network. Our proposed architecture is also inspired by the siamese network. Originally, siamese network has been used to regress the similarity between two signatures by Bromley *et al* [9]. Much of the current work on siamese network also focuses on using it with lower-dimensional data representation, for example, 1D, 2D data, to produce a similarity score for classification problems. We extend the siamese network for multiclass classification and use it for 3D face recognition.

In summary, the following are our major contributions.

- We raise a challenging problem of 3D face recognition using point cloud and propose a new architecture using PointNet [37] and deep convolutional neural network. According to our survey, we are the first who have worked on the direct use of point cloud for 3D face recognition to ensure no loss of geometric information.
- We propose an architecture that is made up of PointNet and siamese network. We first improve PointNet and propose a new architecture and name it as PointNet-CNN. In

order to propose a computationally efficient architecture, we have implemented PointNet with CNN, unlike original PointNet, which is based on the MLP (multi layer perceptron). Further, to discriminate against the highly similar biometric face data, it is important to learn the similarity and dissimilarity in input samples. Hence, in the proposed network, PointNet-CNN is used for feature extraction, and those features are used in a strong discriminative siamese network to accomplish 3D face recognition.

- Our results indicate that the siamese network can be used to recognize 3D faces, and its performance is improved when integrated with a deep convolutional neural network like PointNet-CNN. The proposed network also delivers the real time solution for 3D face recognition even with the limited data.
- In order to make training robust and to avoid overfitting, various augmentation methods are applied at point cloud level, such as rotating, jittering, and perturbing point cloud data.
- To show robustness of the proposed network, we perform rigorous experimentation on two challenging databases and declare the possibility of achieving high recognition performance with other databases.

Rest of the paper is structured as follows. In the next section, existing work related to 3D face recognition is reviewed. In Section 3, basic PointNet and siamese network architectures are presented. These architectures provide a foundation for our proposed deep learning based network model. In Section 4, proposed network model is described, whereas in the next section, outcomes of the experimental analysis are discussed. Paper is concluded in the last section.

2 Related work

Face recognition is a highly explored topic in the field of biometrics. As a result, many detailed surveys [8, 35, 44, 51] exist in the literature. The fast evaluation of 3D scanners and increasing 3D databases have set an exciting research area for 3D face recognition, which indeed can address the limitations of 2D face recognition. The geometric and temporal information preserved by the 3D data can effectively enhance the recognition performance in an unconstrained environment, which is challenging to handle in case of 2D data. Here, we discuss the most relevant work on 3D face recognition and divide it into classical 3D face recognition and deep learning based 3D face recognition.

2.1 Classical methods for 3D face recognition

A 3D face recognition overview is provided in [35]. The conventional methods of 3D face recognition extract the prominent 3D features and match those characteristics to other database characteristics using different metrics for verification/identification tasks [19, 29]. 3D face recognition can be broadly classified into local and global descriptor based approaches, local region based approaches, model based approaches, facial curve based approaches, and point cloud based approaches [42].

Local descriptor based techniques perform matching of 3D facial keypoints obtained by curvature, shape index, and normals. Mian et al. [32] have suggested a highly reproducible keypoint detection algorithm for 3D face data. They have used 2D Scale-Invariant Function Transform (SIFT) and combined it with 3D keypoints to develop multimodal face recognition techniques. Mian et al. [31] have also proposed a multimodal hybrid method (MMH) that is invariant to facial expressions and is able to extract the local and global features in 2D and 3D face data. This work is based on a variant of the ICP algorithm [19], which is an iterative approach used for shape matching. Gupta et al. have performed face recognition by calculating the Geodesic and Euclidean distances between matched key points as given in [17]. Berretti et al. [5] have proposed geometric histogram descriptors where 3D facial features are represented by 3D keypoints, which are extracted with mesh-DOG algorithm.

Soltanpour et al. [43] have proposed a solution for 3D face recognition based on local normal derivative pattern descriptor. The proposed local derivative pattern gives precise information for local derivatives captured in different directions. The more information about local shape is calculated with surface normals instead of depth, and features are extracted with an extreme learning machine based autoencoder. Abbad et al. [2] have proposed a geometry and local descriptor based approach for 3D face recognition to handle the deformations in 3D faces caused by expressions.

Li et al. [29] have proposed a local-region based approach where local features are extracted from many subregions of the 3D face. 3D keypoint features are detected in the region where local curvature is high, and local shape information is used for creating three keypoint descriptors. In [26], Lei et al. have presented a method for 3D face recognition with a local facial descriptor. 3D facial keypoints are detected with a Hotelling transform, and the descriptor is set by measuring four types of geometric features within a keypoint field. In addition to this, a two-phase classification system is used for 3D face recognition with derived local descriptors.

Morphable model based approaches are described in [47]. In these approaches, a morphable model is fitted over the probe scan, which is subsequently passed to the identification module, where it is checked against the database. The restriction of this strategy is that it takes a lot of time as each probe scan needs to be fitted; therefore, it is an iterative approach. In [16], Gilani et al. have built the model (K3DM) for the dense corresponded faces and have performed 3D face recognition by using the K3DM model to fit unseen faces.

Blanz et al. [7] have introduced a model based technique for textured 3D faces by transforming texture and size of 3D scans in vector form, where new faces can be formed by evaluating the linear combination of existing samples. They have also shown 3D face reconstruction from a single sample. Drira et al. have delivered a technique for face recognition in [12] that is based on curvature analysis of 3D face data. This technique extracts the radial curves of the facial surface around the region of the nose tip, and facial recognition is accomplished by matching those radial curves.

2.2 3D face recognition based on deep learning

A technique has been proposed by Kim et al. in [21] for 3D face recognition using deep learning where the input to the network is a depth map obtained by orthogonally projecting 3D point cloud data on a 2D plane. To augment the 3D data, multi-linear 3D morphable model is used where a shape is generated by the Basel Face Model [36], and facial expressions are generated by the Face-Warehouse [10]. Data is augmented with variations in pose, expressions, and occlusion from the single 3D scan. Transfer learning is used from a pre-trained CNN model on 2D database VGG-Face [34], and CNN is finetuned with 3D facial scans.

In [53], Gilani et al. have proposed a Deep 3D Face Recognition network (FR3DNet) which is a dedicated network for 3D face identification task and is trained on 3.1 Million 3D facial scans. The authors have also proposed a new method of data augmentation where synthetic faces are generated by introducing non-linearity in identities. A new face is created by varying the expression of the original face pair. Another deep learning based research work for 3D face recognition is proposed by Tan et al. [46] where the proposed network is a combination of DRNet (Deep registration network) and FRNet (Face recognition network). In their proposed network, they have designed a deep neural network for 3D point cloud registration, and input to the network is the fused 3D depth map based on the Additive Margin Softmax (AMSoftmax) model [49].

Leo et al. have proposed an expression invariant 3D face recognition system in [27]. The proposed technique is based on the combination of SVM (support vector machine) and PCA (principal component analysis). Dutta et al. [13] have performed a 3D face recognition by using volumetric representation for 3D range images. The 3D landmarks are identified, and classification is performed using SVM and K nearest neighbor classifier. Deng et al. [11] have proposed an Additive Angular Margin Loss function (ArcFace) to improve the discriminating power of face recognition system, where feature embeddings are obtained with deep convolutional neural network.

3 Preliminaries

3.1 PointNet architecture

Recently, the use of deep learning on 3D data has gained momentum. This is due to the fact that nowadays, 3D scanners are inexpensive and are giving rise to a number of 3D databases. Any data acquisition technique generates data in the form of a point cloud. PointNet is an efficient architecture proposed in [37], which can take point cloud as an input. There are various applications of PointNet like classification, segmentation, and semantic scene parsing. PointNet considers the fact that point cloud is only a set of 3D coordinates due to which, resultant PointNet architecture is invariant to permutation and geometric transformation of the point cloud. The important parts of the PointNet architecture are the max pooling layer, which is a single symmetric function [3] that aggregates the features from each input point cloud (these features can be used for classification or segmentation task), local and global features combination module, that is used for segmentation and two input alignment modules, that are Input Transformation Network and Feature Transformation Network. For a classification task, the output of the PointNet is the class label for the entire input point cloud, whereas for segmentation task, the output of the PointNet is per point segment label for each input point cloud. In the proposed work, we are mainly interested in the PointNet classification module.

The unified, high-level architecture for PointNet is shown in Fig. 1, which consists of the classification network of the PointNet. Initially, input to the PointNet is a point cloud with (N×3) dimensions. In the Input Transformation Network, these scattered points are aligned to a canonical space to make it invariant to any rigid geometric transformation such as rotation, translation, scaling, etc. Further features are extracted with MLP where layer sizes are (64, 64) and generate local features with output dimension as (N×64). Output features are then transformed in the Feature Transformation Network. Feature matrix in the Feature Transformation Network has much higher dimensions as compared to (N×3) dimension matrix in Input Transformation Network; that is why it becomes difficult to optimize the network. Hence, in the Feature Transformation Network, regularization is used in the softmax training loss function as follows.

$$L_{reg} = \|I - AA^T\|_F^2 \tag{1}$$

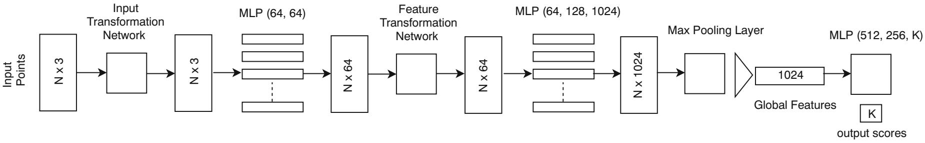


Fig. 1 The original PointNet architecture for classification, where the input to the network is N set of 3-dimensional points and it passes through the series of Input Transformation Network, Feature Transformation Network, and max pooling layer. The network consist of shared multi layer perceptron layers i.e. MLP(64, 128, 1024) and values in the bracket indicate the layer sizes for the MLP layers. Also, the fully connected layer is applied where output size is (512, 256) and the output of PointNet is the classification score for K classes

where A is the feature matrix predicted by the Feature Transformation Network, I is the identity matrix, and feature transformation matrix is aligned to be close to orthogonal. In the next set of MLP (64, 128, 1024) layers, the output feature vector size is (N×1024). Further, max pooling is used as a symmetric function, which makes point cloud invariant to permutations by destroying ordering information and gather the global features. The network produces point features for each input point cloud. Each point is processed independently and identically, and point cloud features are aggregated by applying symmetric functions as follows.

$$f(\{x_1, \ldots, x_n\}) \approx g(h(x_1), \ldots, h(x_n)) \tag{2}$$

where f indicates the function that needs to be approximated. h is the pointwise transformation, and g denotes symmetric function. In the above equation, h is calculated using MLP, whereas g is the combination of single variable functions and max pooling operation. For all the shared layers, batch normalization (BN) is used along with a rectified linear unit (ReLU). The softmax activation function is used in the last layer, and the output of the PointNet classification module is the K classification scores.

3.2 Siamese architecture

Bromley et al. is the first to implement the siamese network for signature verification [9], and the network has successfully detected the 95.5% genuine signature pairs. Siamese network can have many architectures; hence, this approach can be extended to other tasks like image recognition, image verification, multi-class classification, etc [24]. The main advantage of the siamese network over a traditional deep neural network for multi-class classification is that siamese network requires comparatively less training data. As a baseline, it needs only two input images per subject, one as a reference image that is stored in the database and another one as a testing image. The network calculates the matching score showing the resemblance of the two input images. Whenever any new subject is introduced to the network, we need only one image to predict the class of the given image successfully (known as one-shot classification). On the other hand, in the traditional classification problem, we need a lot of labeled training data, and if the new identity comes in, it requires data collection and retraining.

Siamese network consists of two identical architectures based on deep learning, and a pair of images as input to the siamese network. The overall concept of the siamese network is given in Fig. 2. There are two possibilities for input pair, one is genuine pair which is labeled as 1, and the other one is imposter pair which is labeled as 0 where, genuine pair indicates that both images belong to the same class, and imposter pair implies that the two images belong to distinct classes. The network is trained with a database consisting of both

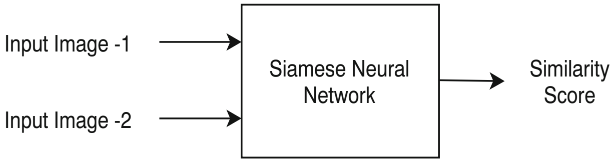


Fig. 2 High level architecture of siamese network

genuine and imposter pairs. Siamese network computes the output in the form of a matching score, which shows the similarity between the two input images. The output similarity score is calculated according to the difference between the feature spaces of two input images. Here, features are produced by two similar deep neural networks. The score is normalized between 0 and 1, where 1 indicates full similarity (that is genuine pair) whereas 0 indicates no similarity (that is imposter pair).

4 Proposed approach

Existing work in 3D face recognition based on deep learning believes in the conversion of 3D data to a lower dimensional space such as 2.5D depth images or 2D images [1]. Another common approach while applying deep learning to 3D face recognition is to convert the original point cloud data into the volumetric representations like voxel grids or occupancy grids. As there is no common 3D representation for 3D face images, most of the researchers prefer lower resolution images like depth maps or multiview images. However, this may cause quantization errors, and essential facial information may be lost in this process. To overcome this limitation, we have used point cloud representation for 3D face data and have designed a deep network which can take point cloud as input directly. However, even if the points are scattered over the surface of a 3D face, they are not isolated, and there is a relation among neighboring points that form the subsets defining the local features. The model should be able to capture the local features and interactions among the points. To accomplish this, we propose a PointNet-CNN architecture for 3D face recognition. An overview of the proposed architecture is as shown in Fig. 3.

In our approach, input to the network is a point cloud data, and we successfully integrate a PointNet approach with a convolutional neural network. Also, we propose to solve the problem of 3D face recognition with limited data with the use of a siamese network by performing similarity learning. The combined proposed architecture is shown in Fig. 6,

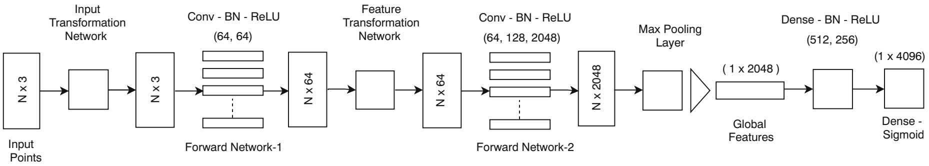


Fig. 3 The proposed PointNet-CNN network takes (N×3) set of points as an input, and it passes through a series of convolutional layers along with batch normalization and ReLU to learn local and global features. Global features are extracted through max pooling, then a fully connected layer along with sigmoid function is applied to get the output feature vector of size (1×4096). In the figure, N denotes number of 3D points in the input, whereas Conv and BN stand for convolutional and batch normalization layer. Moreover, rectified linear unit (ReLU) is the activation function

where PointNet-CNN is the core feature extraction module, and we utilize these features in the siamese network for 3D face recognition. Though the vital idea is inspired by the PointNet, our network is different from the original PointNet [37]. In the proposed approach, modified PointNet-CNN is used for feature extraction, whereas classification is done by the siamese network. We also perform data preprocessing and augmentation to make data suitable for the network. Further, we optimize the network to reduce the computational and memory requirements. We are also able to avoid the extra cost of data conversion by using direct point data as an input. Various steps involved in the proposed architecture are explained below.

4.1 Preprocessing

The preprocessing is performed to minimize the impact of data quality on the network. Input to our proposed network is the point cloud that represents the data on the external surface of a 3D face into the three-dimensional structure system, that is in the form (x,y,z) coordinates. First, we remove the off-face region from the point cloud and fix the dimensionality of the partial point clouds to N points for each sample by sub-sampling uniform points on 3D facial surfaces [22]. Data is normalized in the scale between 0 to 1 to achieve uniformity in the point cloud [38].

4.2 Augmentation

The effectiveness of the neural network can be improved by using more training data. The data augmentation techniques are a proven way to improve the generalization of a network. In data augmentation, more samples are created per class by applying some transformations without changing the number of classes. It is performed to make training robust and to avoid overfitting. For augmentation, 3D face is rotated by 90° with respect to the up-axis. The data is augmented by randomly perturbing the points by small rotations with angle-sigma=0.06, angle-clip=0.18, and jittering the position of each point cloud by Gaussian noise with mean and standard deviation values as 0.02 and 0.06 respectively [25, 37, 38]. We also use some other augmentation techniques like shuffling the order of points in each point cloud and randomly shifting and scaling of point cloud data. We apply all the data augmentation techniques mentioned here at point cloud level.

4.3 Network architecture

This section describes the proposed network for 3D face recognition. In our proposed PointNet-CNN, we show how CNN can comply with the unstructured point cloud. The proposed PointNet-CNN network is divided into three parts *viz.* Input Transformation Network, Feature Transformation Network, and Forward Network. The proposed architecture of the network is displayed in Fig. 3. We propose PointNet with CNN to get an improved network in terms of efficiency and computational cost. On the other hand, original PointNet is implemented with MLP; it is a neural network which is a combination of fully connected layers. In MLP, each node is connected to every other node that results in redundancy, inefficiency, and an increase in a number of parameters. CNN enables weight and parameter sharing such that filters look for a particular pattern no matter where the pattern is located, whereas spatial information may be lost in the case of MLP. In CNN, a number of parameters are less, weights are much smaller, and hence it is easy to train it as compared to MLP. Also, it is computationally efficient to use CNN as compared to MLP. The input to the deep

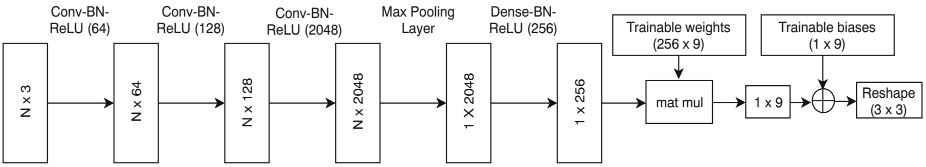


Fig. 4 Input Transformation Network Architecture. In the figure, N denotes number of 3D points in the input, whereas Conv and BN stand for convolutional and batch normalization layer. Moreover, rectified linear unit (ReLU) is the activation function and mat-mul stands for a matrix multiplication operation

network is given as preprocessed and augmented point cloud data, where the point cloud is represented in the form of a matrix of size (N×3).

Initially, the input is passed to the Input Transformation Network as shown in Fig. 4. This network performs input transformation, pose normalization, and aligns unordered point cloud to make it invariant to geometric deformation. Input transformation performs multiplication of each point cloud with a transformation matrix. The architecture of this network resembles high-level PointNet architecture as, it also consists of a series of convolutional, max pooling, and dense layers. Convolutional layers with sizes (64, 128, 2048) are used, which map the input to higher dimensions, and max pooling layer encode the global information. The output of the dense layer (256) is combined with globally trainable weights, biases, and transformation matrix of size (3×3) is generated. Further, this transformation matrix is multiplied to input data of size (N×3) before proceeding to the next layer. Also, batch normalization and ReLU activation functions are used at subsequent layers. We optimize this network and apply less number of dense layers to make network computationally efficient as compared to original PointNet, which may become expensive due to dense layers.

The input to the first Forward Network, as shown in Fig. 3 is an ordered set of points (N×3) where features are learned with convolutional layers of sizes (64, 64), and the output of the network is produced as a set of local features of size (N×64). These learned features are transformed into the Feature Transformation Network. The architecture and working of Feature Transformation Network are given in Fig. 5, which are the same as Input Transformation Network, except dimensions of trainable weights and biases becomes (256×4096) and (1×4096) respectively, which results in transformation matrix of size (64×64). Input to the second Forward network is the transformed features of the size (N×64). Here, three convolutional layers are applied with layer sizes as (64, 128, 2048) where each convolutional layer is associated with a kernel followed by Batch Normalization and ReLU as the activation function. Kernel produces the features for each point and does not include unrelated points in the feature map. The multiple convolutional operations generate the feature

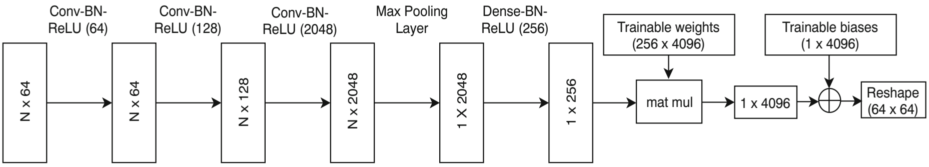


Fig. 5 Feature Transformation Network Architecture. In the figure, N denotes number of 3D points in the input, whereas Conv and BN stand for convolutional and batch normalization layer. Moreover, rectified linear unit (ReLU) is the activation function and mat-mul stands for a matrix multiplication operation

matrix of the size (N×2048). The max pooling layer aggregates each of the local features and produces the global features for each input sample. Further, we apply dense layers with output size (512, 256) along with ReLU activation. Finally, flattening is performed, and a dense layer with a sigmoid activation operation gives an output vector of size (1×4096).

We propose the PointNet-CNN with the siamese network to perform 3D face recognition. Figure 6 shows the architecture of our proposed approach with significant layers and outputs. The proposed architecture consists of the siamese network, where we make full use of 3D facial features produced by the proposed PointNet-CNN. As siamese is a two flow network, it is mandatory to use a pair of inputs for training so that it can produce a similarity score between the inputs. When two distinct input 3D scans are applied to proposed PointNet-CNN architecture for each input 3D face scan, PointNet-CNN gives output feature vector of size (1×4096).

Siamese network calculates the similarity between the two 3D face scans according to the difference in their feature spaces obtained by PointNet. In order to find dissimilarity between the input feature vectors, lambda operations are performed. We use four different types of lambda function to calculate the differences between the input feature vectors such as $x_1 - x_2$, $x_1 + x_2$, $x_1 * x_2$ and $(x_1 - x_2)^2$ [24]. The output of these four operations is concatenated and reshape into a vector of size (4 × 4096), which is further given as an input to the convolutional block where 2D convolution is implemented. The prediction scores are generated by the final fully connected layer where sigmoid activation is applied, which gives the similarity score for the input face pair and acts as the top layer, which joins the twin network.

Once the proposed network (combination of PointNet-CNN and siamese) has been trained, the performance is evaluated on the testing set to determine the discriminating power of the proposed network. The testing is carried out on the unknown 3D faces in a pairwise manner, where the network outputs a probability score for each testing pair. Thus the matching/probability score determines if the input 3D face scans belong to the same class or belong to two different classes. If the two input 3D face scans belong to the same person, then their feature vectors must be similar to each other, whereas if two input 3D face scans belong to the different person, then feature vectors will be different. Hence, the matching

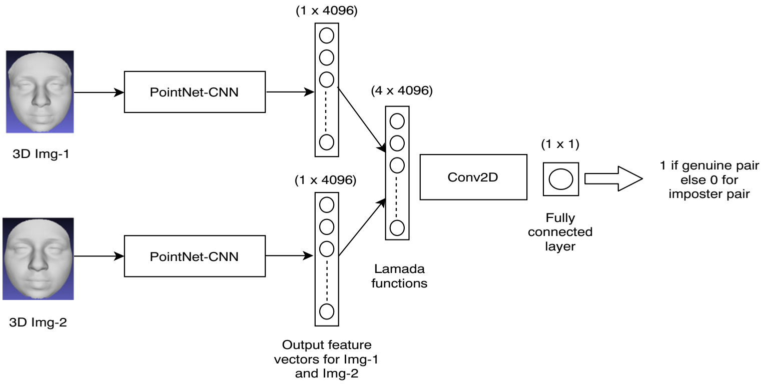


Fig. 6 The proposed siamese network where features are learned with PointNet and output feature vectors are compared, four lambda functions are used for pairwise component comparison of two input feature vector of size (1×4096). The output is passed to the convolutional block, and the final fully connected layer is applied with the sigmoidal unit, which gives (1x1) prediction score by comparing the similarity between two input feature vectors

score generated by the network will also be different for each pair, where the highest score pair shows that both 3D scans belong to the same person. Due to the robust feature learning provided by PointNet-CNN, it is possible to compute the similarity between the two 3D face scans. The following discussion elaborates on the crucial layers that are part of our proposed PointNet-CNN network.

4.3.1 Convolutional layer

Convolution is a linear operation that performs a weighted sum of its input vectors. Mathematically, it is an operation [14] on two functions which can be expressed as follows in 1D:

$$y(t) = (x * w)(t) = \int x(\tau)w(t-\tau)d\tau \tag{3}$$

where $*$ is the convolutional operator, x is input data, and w is the weighting function (also known as the kernel). Here, $t \in R, \tau \in R$, $x\colon R \to R$, and $w\colon R \to R$. The resulting function $y\colon R \to R$ is obtained after convolution. The function x and w are denoted as the integration of the product of both functions where one function is reversed, and the other one is shifted by τ.

CNN requires a highly regular input format; however, point clouds are much irregular and handling such data for simple CNN is very challenging. To overcome this, the input point cloud is processed and ordered in the Input Transformation Network. The convolutional layers of the network capture the interaction among the data points. The series of convolutional layers are used in the Input Transformation Network, Feature Transformation Network, and Forward Network that effectively learns the point features out of each point cloud.

4.3.2 Activation functions

As convolutional is a linear operation, a non-linear operation is also required to learn more complex features. In the proposed network, a non-linear operation known as an activation function is also applied. ReLU and sigmoid functions are used in the proposed network where ReLU performs MAX operation that returns the same value if the value is zero or higher, and if the value is negative, it returns zero [14]. It is defined as follows:

$$f(v) = max(0, v) \tag{4}$$

where v is the vectorized input feature. ReLU also reduces the vanishing gradient problem, and it is applied with convolutional and dense layers. Further, the sigmoid function is used in the last fully connected layer of the network, which produces output as the probability values in the range of 0 to 1.

4.3.3 Batch normalization

Batch normalization is used to improve precision and to accelerate the process of training. It normalizes the activations in the intermediate layers of the neural network and improves the gradient flow. Without BN gradient updates, it may cause a diverging loss. BN offers an empirical approach to calculate the feature mean and variance in every immediate convolutional layer before introducing non-linearity. It corrects the activation values to make them with zero-mean and unit gradient decent.

4.3.4 Max pooling layer

The primary purpose of the max pooling layer is to perform dimensionality reduction. It divides the input data points into non-overlapping regions, and the highest value for each region is calculated. Performing a pooling operation has many other advantages. It reduces the output size that, in turn, decreases the number of output parameters to be learned. This reduces the computational cost of the network and removes overfitting. In other words, the pooling layer summarizes the output of convolution. In the proposed network, we use max pooling operation to simplify the feature map. The input to the global feature extraction function is the point features derived from the convolutional layers. Max pooling layer is the building block of global feature extraction function, which aggregates all the local features and outputs global features for each input point cloud with an array of size (1×N).

4.3.5 Dropout

It is applied to avoid the problem of overfitting in multiple non-linear hidden layers. It actually performs regularization. To perform this, some nodes are randomly removed with keep probability of 0.5. A node is removed from the network by simply multiplying its output by zero. This enables the use of a new set of nodes, which makes the training more robust. Dropout is applied before a dense layer and after that, flattening is performed.

4.3.6 Loss function

We have utilized a binary cross-entropy loss function in the proposed network. This loss function is implemented when a model tries to decide whether a given example belongs to a class or not. This function is useful when the model output values are between 0 and 1. In our proposed approach, we are matching two output feature vectors, and the sigmoid activation function is applied in the last fully connected layer, which produces values between 0 and 1. Mathematically, the cross-entropy loss function [24] is defined as follows.

$$L(s1, s2) = -(h(s1, s2)\log(p(s1, s2)) + (1 - h(s1, s2))\log(1 - p(s1, s2))) \tag{5}$$

where $s1$ and $s2$ are the two input 3D face samples, and $h(s1, s2)$ determines the label such that, $h(s1, s2) = 1$ when $s1$ and $s2$ belong to the same class and $h(s1, s2) = 0$ when $s1$ and $s2$ belong to different classes. Further, p is the predicted probability value whereas L is the cross-entropy loss with respect to $s1$ and $s2$.

5 Experimental analysis

The proposed architecture has been analyzed for the purpose of 3D face recognition. Details about the databases used, data preparation, parameter tuning, and performance analysis are provided below.

5.1 Databases used

We use two databases, namely Bosphorus university 3D face database [40] and in-house database (IIT Indore 3D face database), for our experimental analysis. The Bosphorus database [40] contains 3D faces with multiple expressions, poses, and a wide variety of realistic occlusions. For each subject, a maximum 54 facial scans are available, whereas

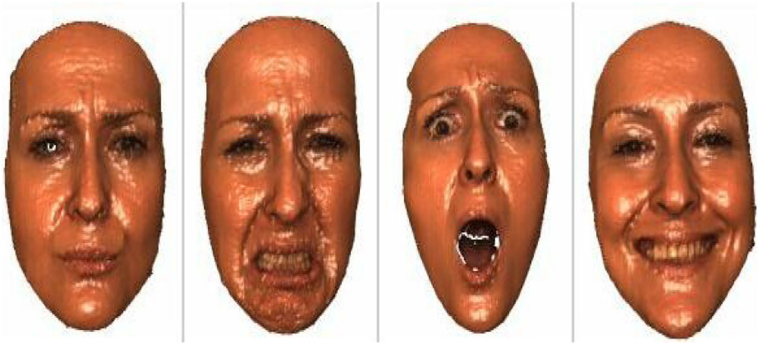


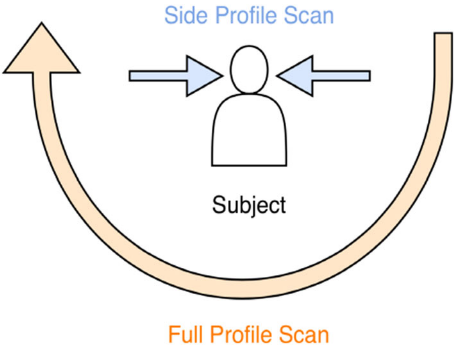
Fig. 7 Some samples from Bosphorus 3D face database with different expressions of a single person. Texture mapping and lighting effects are used for rendering

minimum scans available for a subject are 31. The database contains 4666 3D face scans of 105 subjects. The database is collected using the Inspeck Mega Capturor II 3D scanner, which is a structured light-based 3D scanner. There exist at least 35 expressions for each subject, pose variations with 13 yaws, pitch and cross rotations, and four types of occlusions (hair, eyeglasses, hand, beard). Some sample 3D scans for a subject from the database are shown in Fig. 7.

The IIT Indore 3D face database is comparatively challenging than the Bosphorus database due to the fact that it has been acquired in an uncontrolled environment and has 3D facial scans with noise. The database has been acquired in outdoor conditions and in different environments with improper lightning conditions, which has made the data noisy. Further, the number of 3D scans available in the database per subject are just three, which is very low as compared to the Bosphorus database, which has minimum 31 samples per subject.

3D face scans in IIT Indore database are collected with the help of Artec-Eva® 3D scanner. This scanner is based on structured light technology and is capable of capturing 3D data with 0.5mm 3D resolution and 0.10mm 3D accuracy. The face images are scanned by using the scanner in its standard geometric mode (acquisition setup shown in Fig. 8). The distance between scanner and subject is kept around one meter, and no significant changes are made in the nearby environment for capturing the data. We have only tried to avoid any reflective

Fig. 8 Full profile scanning



object in the vicinity of the subject as a reflection from the object may cause noise in the 3D data. We are only focusing on face part of the human body. Further, we have used a wig cap to cover the hair of a person. Hair may cause unwanted occlusion and refection due to which scanner should not capture hair. In addition, the optical properties of teeth and eyes can be troublesome for the scanner. As a solution, the scanner captures the eyes and teeth (if visible) geometrically inward and generate the holes. The database contains subjects which encompass a diverse age in the range from 18 to 45 years. The different subjects present in the database belong to staff, students, and faculty members of the institute. The histograms representing the age and gender distribution of subjects are given in Fig. 9.

3D face scan is captured with a full profile scan by scanning the subject face in a 360° manner. The experimental setup for 3D face acquisition is depicted in Fig. 8. While capturing 3D face data of a person, a complete 3D view from right to left is scanned for each subject. To capture three samples, the subject is scanned in three sittings. Since we have acquired the complete 3D view from right to left, it contains data for the face from right profile view to left profile view, including frontal view. It is to note that the 3D face scans in IIT Indore database are taken in an unconstrained environment and are neutral ones (without expression and poses). A few samples of two different subjects from IIT Indore 3D face database are shown in Fig. 10. The scanned data in this database contains some noise due to an unnecessary reflection (caused by earring or hair) and an unconstrained environment. Hence, acquired data from the scanner requires preprocessing. For this purpose, different preprocessing techniques such as diffusion methods are used to reduce noise, and the interpolation technique is used to fill the holes in the data. The IIT Indore database has been captured in three phases, where the gap between two subsequent phases is kept at least one year. Considering the quality and number of samples in the database, we use phase-2 and phase-3 data for our experimentation. In the phase-2 database, there are a total of 90 subjects, whereas, in the phase-3 database, there are 99 subjects, each with three samples. The summary of both the databases is given in Table 1.

Since the number of samples for a subject in Bosphorus database varies from 31 to 54, to maintain consistency, we have chosen 30 samples from each subject. Wherever the number of samples in a subject is more than 30, we have selected 30 samples for the subject from the database randomly. For evaluation of our proposed network, we divide the overall database into three parts, *viz.* training, validation, and testing with 70%, 15%, and 15%, respectively. We generate additional 3D face scans with our proposed augmentation methods. Siamese network compares two 3D face scans; hence, it needs two inputs. That is why we have generated genuine and imposter pairs for the siamese

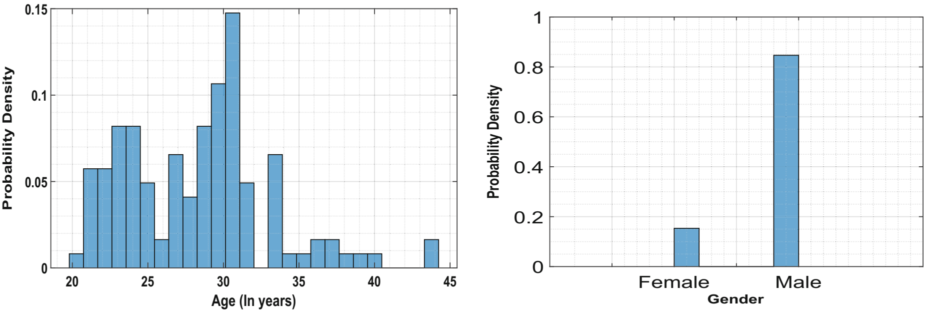


Fig. 9 Age and gender distribution for IIT Indore database

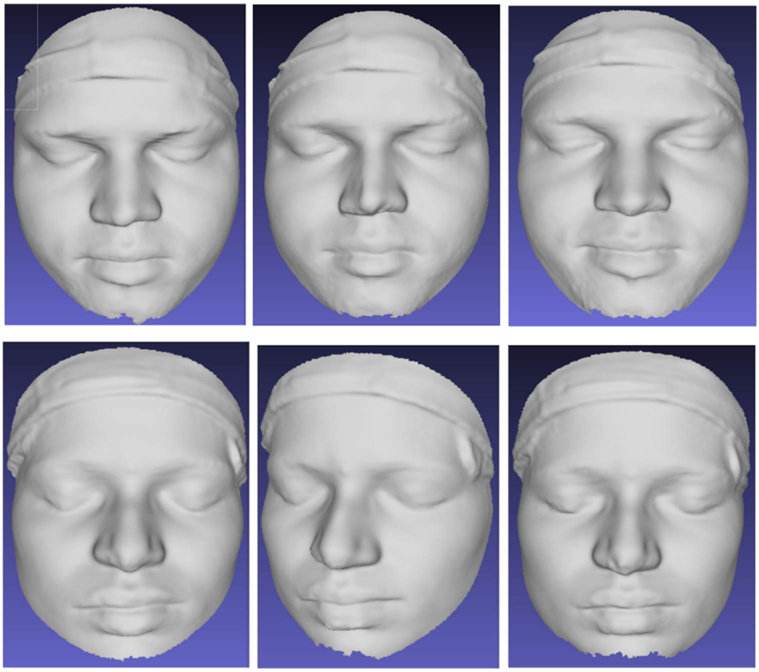


Fig. 10 A few samples from IIT Indore 3D face database where top and bottom rows show samples of two different subjects respectively

network where genuine pair is assigned a label 1 and for an imposter pair, the label is given as 0. We first generate all possible genuine pairs and then generate an equal number of imposter pairs randomly so that half of the database consists of imposter pairs, and the half consists of genuine pairs. We have maintained that there should not be any duplicate pair, and 3D scans of all classes should be covered in both genuine and imposter sets.

5.2 Data preparation

Initially, Bosphorus 3D face data is available in .BNT data format. To make it suitable for our proposed network, we have converted it into .ASC. IIT Indore 3D face data is available

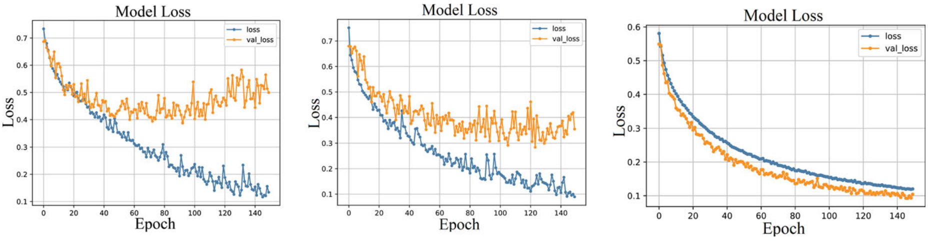
Table 1 Details of databases

Database	IDs	Scans	Expression	Pose	Occlusion	Scanner
IIT Indore phase-2	90	270	-	-	-	Artec-Eva
IIT Indore phase-3	99	297	-	-	-	Artec-Eva
Bosphorus	105	4666	35	±90°	4 types	Inspeck Mega Capturor II 3D

in .ASC file format so, no format conversion is required. Considering the size of the input data, we use .H5 file format for giving training and testing input to the network. In .H5 file format, the data is saved in hierarchical style in the form of a multidimensional array. This file format provides flexibility in data mapping and stores a sample with its class label. To take advantage of this file format, we have converted files of both the databases to .H5 file format. Initially, each face scan may contain millions of points; however, all those points are not needed for face recognition. This also makes input more bulky and difficult to process. Due to this, subsampling becomes essential. In proposed subsampling methods, we fix dimensionality of input data and select 2048 points randomly on the 3D face surface. The points are subsampled in such a way that the resultant point cloud is able to retain the structure of the original face.

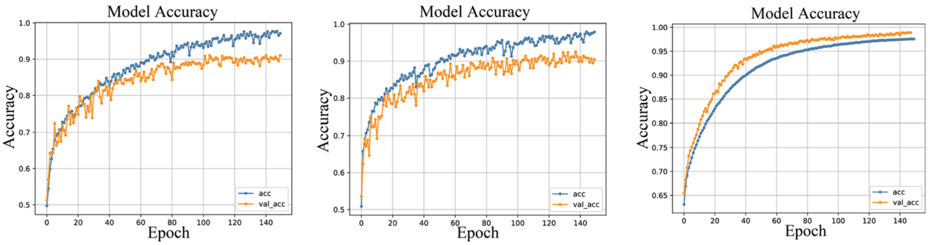
5.3 Parameter analysis

The proposed network is implemented using Keras and Tensorflow frameworks. To train the siamese network with PointNet-CNN as a basic model, we perform hyperparameter tuning as follows. We initialize the weights of the network with a mean of 0.0 and a standard deviation of 0.01 for convolution layers as well as for fully connected layers. In addition, biases are initialized to 0.5 mean and 0.01 standard deviation [24]. Adam (Adaptive momentum estimation) optimizer with β = 0.9 is utilized for optimization during training where it is responsible for evaluating the cost function, based on which all weights are adjusted [37]. We use Adam optimizer with a learning rate of 0.001. This optimizer is based on optimizing stochastic objective features and is based on lower-order moments adaptive estimates. We have used binary cross-entropy as the loss function that minimizes the difference in probability distributions between the true labels and the predicted labels. Since we perform lambda operations on twin feature vectors and sigmoid activation is applied in the final fully connected layer, cross-entropy loss is the right choice for our network [24]. We train our network for 150 epochs with a batch size of 64. Graphs for loss vs. epoch are shown in Fig. 11 for an experiment where it shows the decreasing loss values for training and validation as epochs proceed. Batch size and epochs are decided after extensive experimentation with different hyperparameters. Further to enhance the training, batch normalization is used. Dropout during training is performed with a probability value of 0.5 to avoid overfitting. In Fig. 12, we present the accuracy vs. epoch graph for each database where, we show both training and validation accuracies for different databases.



(a) IIT Indore (phase-2) (b) IIT Indore (phase-3) (c) Bosphorus

Fig. 11 Loss vs. epoch curves for different databases



(a) IIT Indore (phase-2) (b) IIT Indore (phase-3) (c) Bosphorus

Fig. 12 Accuracy vs. epoch curves for different databases

5.4 Performance evaluation

The efficiency of the proposed network is assessed using recognition rate of a 3D face recognition system. Recognition Rate is defined as the percentage of correctly classified genuine and imposter pairs out of all pairs used in the testing.

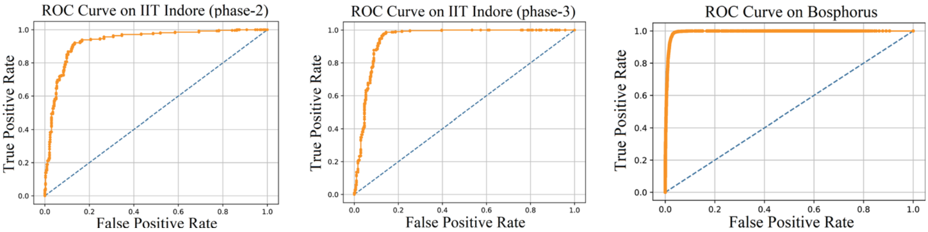
$$\text{Recognition} \quad \text{rate} = \frac{\text{Number of correctly predicted samples}}{\text{Total number of predictions}} \tag{6}$$

We have also shown our results in the form of Receiver operating characteristic (ROC) curve and area under ROC curve (AUC). ROC curve is a probability curve that demonstrates how a biometric recognition system can distinguish between the face models. It shows the relationship between TPR (true positive rate) and FPR (false positive rate), and it shows the performance of the classification model at all thresholds. Where, TPR and FPR can be defined as

$$TPR = \frac{TP}{TP + FN}, \quad FPR = \frac{FP}{FP + TN} \tag{7}$$

AUC is the area under the ROC curve, which aggregates the performance of ROC curve at all possible thresholds. The effectiveness of 3D face recognition can be analyzed with the ROC curve, which summarizes the performance of a recognition system in terms of TPR and FPR values. The ROC curves for different databases are shown in Fig. 13.

In Table 2, we provide recognition rate and values of AUC for each database. The proposed network gives 98.91%, 87.31%, and 92.19% recognition rates for Bosphorus, IIT Indore (phase-2), and IIT Indore (phase-3) databases, respectively. For the Bosphorus



(a) IIT Indore (phase-2) (b) IIT Indore (phase-3) (c) Bosphorus

Fig. 13 ROC curves for different databases

Table 2 Experimental analysis of Bosphorus and IIT Indore 3D face database

Database	Recognition rate (%)	AUC (%)
IIT Indore 3D face (phase-2)	87.31	93.1
IIT Indore 3D face (phase-3)	92.19	94.6
Bosphorus 3D face	98.91	99.4

database, our network achieves AUC as 99.4%, whereas for IIT Indore (phase-2) and IIT Indore (phase-3) databases, it has achieved AUC as 93.1% and 94.6% respectively. The performance in IIT Indore (phase-2) and IIT Indore (phase-3) databases is found to be slightly low as compared to the Bosphorus database. However, this is due to challenging 3D scans in the databases and the availability of less number of samples in the database.

Since not much work is done in the field of 3D face recognition using deep learning, we show a comparison of our proposed network with both conventional as well as deep learning based approaches for 3D face recognition. We compare our method against the existing 3D face recognition techniques, which are evaluated on Bosphorus 3D face data and show comparative performance in Table 3. We can conclude from the presented results that the exploitation of the facial features through our proposed network is capable of producing superior results against the state-of-the-art methods. There is one technique proposed in [53], which has shown perfect performance on Bosphorus 3D face data; however, it is due to the training on millions of 3D facial scans and augmenting data with synthetic images. Our proposed network shows capability of achieving good performance with the use of limited training data. Moreover, it is capable of taking input directly in the form of point cloud format, due to which it decreases the cost of storage, data conversion, preprocessing, and also reduces overall computational time.

We find that our proposed network works better than the original PointNet. The original PointNet (as mentioned in preliminaries) achieves good classification performance for 3D objects, when number of classes are less and quite distinct from each other, and there exists

Table 3 Comparison of recognition rate of proposed network with state-of-the-art techniques on Bosphorus database

Method	Recognition rate in (%)
Mian et al. [31]	96.40
Gilani et al. [16]	98.60
Lei et al. [26]	98.90
Berretti et al. [5]	95.70
Ocegueda et al. [33]	98.60
Li et al. [28]	95.40
Soltanpour et al. [43]	97.75
Leo et al. [27]	96.29
Abbad et al. [2]	93.24
Dutta et al. [13]	96.37
Our Proposed Method	**98.91**

Our proposed method is in bold

Table 4 Time analysis of Bosphorus and IIT Indore 3D face databases

Database	Training time per sample (in seconds)	Inference time per sample (in seconds)
Bosphorus	5.35	0.020
IIT Indore (phase-2)	5.21	0.024
IIT Indore (phase-3)	5.16	0.022

a large number of samples per class. For example, PointNet works well when evaluated for the classification of objects in the ModelNet40 [50] database, which contains objects such as airplanes, tables, chairs, etc. which are fairly distinct from each other. However, in 3D face biometrics, inter subject variations are significantly low (all the faces look geometrically very similar) that is reason original PointNet does not achieve expected performance. Moreover, local feature extraction is not robust in the case of original PointNet, as mentioned in [38]. Further, the original PointNet architecture is completely based on MLP due to which more parameters are required, and a large amount of data is also needed to train the network. As a solution, we have performed 3D face recognition by learning the similarity in 3D faces by proposing PointNet-CNN which is an enhanced version of the original PointNet architecture. The robust feature extraction power of the proposed PointNet-CNN and classification based on similarity learning utility of the siamese network are responsible for achieving high performance in the proposed architecture. We have also utilized the data augmentation to get a sufficient number of samples for each subject for the training. This helps us in better training of the network and in turn, results in higher classification performance.

We also analyze the performance of our proposed network in terms of processing time required for training and testing. We have performed experiments on a machine having Intel Xeon processor and NVIDIA Tesla V100 GPU card with 32GB RAM and Xubuntu operating system with 600GB RAM. We have reported training and inference times per sample for different databases in Table 4.

6 Conclusions

This work has used 3D point cloud data as an input to propose a novel technique for 3D face recognition. The technique utilizes deep learning approaches based on PointNet and siamese network where PointNet is a deep learning based architecture, specially designed to process unorganized point cloud data, and a siamese network is an efficient tool for learning dissimilarities of objects and comparing them. Several methods are existing for face recognition in 3D; however, they mostly use depth images and pre-trained networks, which can be less practical in several instances. We have proposed a deep convolutional neural network based on PointNet for comparing two 3D point clouds. We call the proposed network model as PointNet-CNN. The inputs to the proposed network are facial scans, directly in the form of a point cloud, which reduces the cost of data conversion. Moreover, point cloud has less memory requirement as compared to other data representations such as depth images, voxel grids, etc. Also, spatial and geometric information present in the 3D point cloud data increases the robustness of the network. We have used proposed PointNet-CNN for the extraction of features from the point cloud data. We further make use of a siamese network to compare the features obtained from the proposed PointNet-CNN network. To the best of our knowledge, this is the first work which uses siamese network on PointNet based features obtained from a point cloud. We have demonstrated our proposed network on Bosphorus and IIT Indore 3D face databases. We show the results with different evaluation metrics such as recognition rate, and ROC plot. We have achieved a recognition rate of 98.91% on Bosphorus 3D face database, 87.31% on IIT Indore 3D face database (phase-2), and 92.19% on IIT Indore 3D face database (phase-3). The loss function is a very important component of any network and is responsible for the discrimination of two identities. In our future work, we are going to work to enhance the proposed network by improving the loss function. We would be exploring the use of triplet loss function for the same.

References

1. Abate AF, Nappi M, Riccio D, Sabatino G (2007) 2D and 3D face recognition: A survey. Pattern Recogn Lett 28(14):1885–1906
2. Abbad A, Abbad K, Tairi H (2018) 3D face recognition: Multi-scale strategy based on geometric and local descriptors. Comput Electr Eng 70:525–537
3. Ahmed E, Saint A, Shabayek AER, Cherenkova K, Das R, Gusev G, Aouada D, Ottersten B (2018) Deep learning advances on different 3D data representations: A survey. arXiv:180801462
4. Ahonen T, Hadid A, Pietikäinen M (2004) Face recognition with local binary patterns. In: Proceedings of European Conference on Computer Vision, pp 469–481
5. Berretti S, Werghi N, Del Bimbo A, Pala P (2013) Matching 3D face scans using interest points and local histogram descriptors. Comput Graph 37(5):509–525
6. Bhele SG, Mankar V (2012) A review paper on face recognition techniques. Int J Adv Res Comput Eng Technol 1(8):339–346
7. Blanz V, Vetter T et al (1999) A morphable model for the synthesis of 3D faces. In: Proceedings of SIGGRAPH, pp 187–194
8. Bowyer KW, Chang K, Flynn P (2006) A survey of approaches and challenges in 3D and multi-modal 3D + 2D face recognition. Comput Vis Image Underst 101(1):1–15
9. Bromley J, Guyon I, LeCun Y, Säckinger E, Shah R (1994) Signature verification using a "siamese" time delay neural network. In: Proceedings of Advances in Neural Information Processing Systems, pp 737–744
10. Cao C, Weng Y, Zhou S, Tong Y, Zhou K (2013) FaceWarehouse: A 3D facial expression database for visual computing. IEEE Trans Visual Comput Graph 20(3):413–425
11. Deng J, Guo J, Xue N, Zafeiriou S (2019) ArcFace: Additive angular margin loss for deep face recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4690–4699
12. Drira H, Amor BB, Srivastava A, Daoudi M, Slama R (2013) 3D face recognition under expressions, occlusions, and pose variations. IEEE Trans Pattern Anal Mach Intell 35(9):2270–2283
13. Dutta K, Bhattacharjee D, Nasipuri M, Poddar A (2019) 3D Face Recognition Based on Volumetric Representation of Range Image, pp 175–189
14. Garcia-Garcia A (2016) 3D object recognition with convolutional neural networks (phd thesis)
15. Garcia-Garcia A, Gomez-Donoso F, Garcia-Rodriguez J, Orts-Escolano S, Cazorla M, Azorin-Lopez J (2016) PointNet: A 3D convolutional neural network for real-time object class recognition. In: Proceedings of International Joint Conference on Neural Networks, pp 1578–1584
16. Gilani SZ, Mian A, Shafait F, Reid I (2017) Dense 3D face correspondence. IEEE Trans Pattern Anal Mach Intell 40(7):1584–1598
17. Gupta S, Markey MK, Bovik AC (2010) Anthropometric 3D face recognition. Int J Comput Vis 90(3):331–349
18. Hassaballah M, Aly S (2015) Face recognition: challenges, achievements and future directions. IET Comput Vis 9(4):614–626
19. He Y, Liang B, Yang J, Li S, He J (2017) An iterative closest points algorithm for registration of 3D laser scanner point clouds with geometric features. Sensors 17(8):1862
20. Huang GB, Mattar M, Berg T, Learned-Miller E (2007) Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report. University of Massachusetts

21. Kim D, Hernandez M, Choi J, Medioni G (2017) Deep 3D face identification. In: Proc. of IEEE International Joint Conference on Biometrics, pp 133–142
22. Kingkan C, Owoyemi J, Hashimoto K (2018) Point attention network for gesture recognition using point cloud data. In: Proceedings of British Machine Vision Conference, pp 118
23. Klare BF, Klein B, Taborsky E, Blanton A, Cheney J, Allen K, Grother P, Mah A, Jain AK (2015) Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp 1931–1939
24. Koch G, Zemel R, Salakhutdinov R (2015) Siamese neural networks for one-shot image recognition. In: Proceedings of ICML Deep Learning Workshop, vol 2
25. Le T, Duan Y (2018) PointGrid: A deep network for 3D shape understanding. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp 9204–9214
26. Lei Y, Guo Y, Hayat M, Bennamoun M, Zhou X (2016) A two-phase weighted collaborative representation for 3D partial face recognition with single sample. Pattern Recogn 52:218–237
27. Leo MJ, Suchitra S (2018) SVM based expression-invariant 3D face recognition system. Procedia Comput Sci 143:619–625
28. Li H, Huang D, Morvan JM, Chen L, Wang Y (2014a) Expression-robust 3D face recognition via weighted sparse representation of multi-scale and multi-component local normal patterns. Neurocomputing 133:179–193
29. Li H, Huang D, Morvan JM, Wang Y, Chen L (2014b) Towards 3D face recognition in the real: A registration-free approach using fine-grained matching of 3D keypoint descriptors. Int J Comput Vis 113:128–142
30. Maturana D, Scherer S (2015) VoxNet: A 3D convolutional neural network for real-time object recognition. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 922–928
31. Mian A, Bennamoun M, Owens R (2007) An efficient multimodal 2D-3D hybrid approach to automatic face recognition. IEEE Trans Pattern Anal Mach Intell 29(11):1927–1943
32. Mian AS, Bennamoun M, Owens R (2008) Keypoint detection and local feature matching for textured 3D face recognition. Int J Comput Vis 79(1):1–12
33. Ocegueda O, Passalis G, Theoharis T, Shah SK, Kakadiaris IA (2011) UR3D-C: Linear dimensionality reduction for efficient 3D face recognition. In: Proceedings of International Joint Conference on Biometrics, pp 1–6
34. Parkhi OM, Vedaldi A, Zisserman A et al (2015) Deep face recognition. In: Proc. of British Machine Vision Conference, pp 6
35. Patil H, Kothari A, Bhurchandi K (2015) 3D face recognition: features, databases, algorithms and challenges. Artif Intell Rev 44(3):393–441
36. Paysan P, Knothe R, Amberg B, Romdhani S, Vetter T (2009) A 3D face model for pose and illumination invariant face recognition. In: Proceedings of Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, pp 296–301
37. Qi CR, Su H, Mo K, Guibas LJ (2017a) PointNet: Deep learning on point sets for 3D classification and segmentation. In: Proc. of IEEE Conference on Computer Vision and Pattern Recognition, pp 652–660
38. Qi CR, Yi L, Su H, Guibas LJ (2017b) PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Proc. of Advances in Neural Information Processing Systems, pp 5099–5108
39. Rahim R, Afriliansyah T, Winata H, Nofriansyah D, Aryza S et al (2018) Research of face recognition with fisher linear discriminant. In: Proceedings of IOP Conference Series: Materials Science and Engineering, pp 012–037
40. Savran A, Alyüz N, Dibeklioğlu H, Çeliktutan O, Gökberk B, Sankur B, Akarun L (2008) Bosphorus database for 3D face analysis. In: Proc. of European Workshop on Biometrics and Identity Management, pp 47–56
41. Schroff F, Kalenichenko D, Philbin J (2015) FaceNet: A unified embedding for face recognition and clustering. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp 815–823
42. Sharma PB, Goyani MM (2012) 3D face recognition techniques-a review. Int J Eng Res Appl 2(1):787–793
43. Soltanpour S, Wu QMJ (2019) Weighted extreme sparse classifier and local derivative pattern for 3D face recognition. IEEE Trans Image Process 28(6):3020–3033
44. Soltanpour S, Boufama B, Wu QJ (2017) A survey of local feature methods for 3D face recognition. Pattern Recogn 72:391–406
45. Taigman Y, Yang M, Ranzato M, Wolf L (2014) DeepFace: Closing the gap to human-level performance in face verification. In: Proc. of IEEE Conference on Computer Vision and Pattern Recognition, pp 1701–1708

46. Tan Y, Lin H, Xiao Z, Ding S, Chao H (2018) Face recognition from sequential sparse 3D data via deep registration. arXiv:181009658
47. ter Haar FB, Veltkamp RC (2010) Expression modeling for expression-invariant face recognition. Comput Graph 34(3):231–241
48. Turk MA, Pentland AP (1991) Face recognition using eigenfaces. In: Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp 586–591
49. Wang F, Cheng J, Liu W, Liu H (2018) Additive margin softmax for face verification. IEEE Signal Process Lett 25(7):926–930
50. Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, Xiao J (2015) 3D ShapeNets: A deep representation for volumetric shapes. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp 1912–1920
51. Zhou H, Mian A, Wei L, Creighton D, Hossny M, Nahavandi S (2014) Recent advances on singlemodal and multimodal face recognition: a survey. IEEE Trans Human-Mach Syst 44(6):701–716
52. Zhou S, Xiao S (2018) 3D face recognition: a survey. Hum-Centric Comput Inf Sci 8(1):35
53. Zulqarnain Gilani S, Mian A (2018) Learning from millions of 3D scans for large-scale 3D face recognition. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp 1896–1905