Multimedia Tools and Applications (2022) 81:12795–12814
https://doi.org/10.1007/s11042-022-12211-9

Few-data guided learning upon end-to-end point cloud network for 3D face recognition

Yi Yu[1,2] · Feipeng Da[1,2] · Ziyu Zhang[1,2]

Received: 13 May 2021 / Revised: 30 August 2021 / Accepted: 10 January 2022 /
Published online: 21 February 2022

Abstract
Deep-learning-based 3D face recognition methods have developed vigorously in recent years, while the potential of these methods is being exploited in more and more scenarios. In this paper, an end-to-end deep learning network entitled Sur3dNet-Face for point-cloud-based 3D face recognition is proposed. The method uses PointNet, which is a successful point cloud classification solution but performs unexpectedly in face recognition, as the backbone. To adapt the backbone to 3D face recognition, modifications in network architecture and a few-data guided learning framework based on Gaussian process morphable model is supplemented. Instead of mass data in multiple datasets for training, our method takes only Spring2003 subset of FRGC v2.0 for training which contains 943 facial scans and the network is well trained with such a small amount of real data. The processing time to generate face representation is less than 0.15 s. Without fine-tuning on the test set, the Rank-1 Recognition Rate (RR1) is achieved as follows: 98.85% on FRGC v2.0 dataset and 99.33% on Bosphorus dataset, which proves the effectiveness and the potentiality of our method. When facing scenarios with limited resource, the proposed method is expected to give a competitive performance.

Keywords Deep learning · 3D face recognition · Point cloud network

1 Introduction

Biometrics has always been a research hotspot, among which face recognition is considered a reliable biometric technology for identification and verification. It has been extensively

This work is supported by Shenzhen Science and Technology Innovation Commission (STIC) (Grant Nos. JCYJ20180306174455080), Natural Science Foundation of Jiangsu Province of China (Grant Nos. BK20181269), and Special Project on Basic Research of Frontier Leading Technology of Jiangsu Province of China (Grant Nos. BK20192004C).

✉ Feipeng Da
dafp@seu.edu.cn

[1] School of Automation, Southeast University, Nanjing 210096, China

[2] Key Laboratory of Measurement and Control of Complex Systems of Engineering, Ministry of Education, Southeast University, Nanjing 210096, China

studied for years [4, 53] and is widely used in applications including security, surveillance, and finance.

In the beginning, 2D face recognition usually uses artificially designed feature extractors. Among them, there are some successful classic solutions such as EigenFace [56], FisherFace [6], and LPB Face [1]. In recent years, the performance of 2D face recognition has zoomed up following the advancement in deep learning and the accessibility of massive training data such as LFW [25] and CASIA-WebFace [58]. Training upon these massive data, deep learning methods comfortably outperform classic ones.

Among many different network architectures, end-to-end networks, or networks that directly input raw data and output recognition results, are the trend of network design. Without artificially designed feature extractor, end-to-end networks automatically learn how to extract features during the learning process. The majority of well-known 2D face recognition methods are end-to-end networks, including FaceNet [48], SphereFace [36], and ArcFace [16], which have respectively achieved accuracy rates of 99.63%, 99.76%, and 99.83% on LFW dataset. Motivated by these successful methods, many researchers hope that end-to-end deep learning networks can also be applied to 3D face recognition.

Although 2D face recognition has achieved decent accuracy on datasets, it still faces many problems in practical applications. For example, 2D face images vary greatly under different poses and illumination. More importantly, images can be easily forged [28], bringing into question the security of 2D face recognition. 3D face recognition has the potential to overcome these drawbacks. Specifically, the 3D model represents the absolute anatomical structure rather than the texture or color of the face [15], thereby improving the discriminative ability of face recognition. Also, it is robust to changes in pose and illumination [11].

Numerous 3D face recognition methods have been developed, as reviewed in [44, 50, 52]. Similar to the trend in 2D face recognition, 3D face recognition methods based on deep learning [26, 63] is gradually overtaking classic methods using artificially designed feature extractors. However, in most of these deep learning methods, 3D data are firstly projected into 2D images, named depth images or range images, and then 2D networks are utilized for recognition. These methods are not end-to-end as the process of projecting 3D data into 2D images still uses artificially designed extractors. Though this strategy has the potential to capture prominent facial features, yet it does not make full use of the features available in 3D data.

On the other hand, whether deep learning can achieve good results, to a great extent, depends on training data. As is known, many articles use more and more data to train the network. Although these methods compare the recognition rate on the same dataset, they do not use the same training set, which makes the comparison unfair. Take the state-of-the-art method proposed in [12] as an example, six datasets are used in the training process including about $22K$ scans, and the real data used for training are ten times more than those for testing. Such methods unsurprisingly obtain satisfactory recognition rates, but in large-scale practical applications, the training set is usually smaller than the test set, and thus the recognition rate could be far below expectation. All these facts reveal that how to train 3D face recognition network with a small amount of data, especially an end-to-end one, is still an unsolved problem.

In fact, end-to-end networks have been pervasively used in many 3D scenarios other than face recognition, such as 3D object classification and 3D segmentation. For example, Charles et al. proposed an end-to-end network PointNet [13] to directly handle point clouds, and the enhanced version PointNet++ [46] is established upon PointNet. Similar works such as PointCNN [32], KPConv [54], DGCNN [57], and DPAM [34] are all end-to-end networks for 3D object classification and 3D segmentation.

These end-to-end 3D networks have achieved breakthrough performance on 3D object classification and 3D segmentation. However, not designed for faces, these networks show bad performances on 3D face recognition. Whether these end-to-end 3D networks can, by some modification, achieve good results in 3D face recognition is a topic of interest to many researchers. Some progress has been made [9, 62], but satisfactory results have not yet been achieved.

The motivation and the goal of this paper are as follows:

According to the above analysis, the end-to-end networks predominate in tasks including 2D face recognition, 3D object classification, and 3D segmentation. But in 3D face recognition, existing methods still use artificially designed feature extractors, and the lack of 3D face dataset further limits the training of end-to-end networks. Motivated by these factors, this paper is aimed at designing an end-to-end network for 3D face recognition and a strategy to train our network with a small amount of data. The accuracy is expected to be on a par with the state-of-the-art methods, and the speed faster than existing methods. Our experimental results prove that our proposed method successfully achieves this goal.

The main features of our method are as follows:

1) It is an end-to-end deep learning network for 3D face recognition. Different from those methods entitled ”3D” but actually based on 2D images projected from 3D data, our method directly uses point clouds as the input, and the entire processing time from input point cloud to output face representation is less than 0.15s.
2) Taking advantage of a novel training framework upon Gaussian process morphable models (GPMM), the network is well trained with a small amount of real data.

The main contributions of this paper are as follows:

1) An end-to-end deep learning network for 3D face recognition is proposed to get face representations directly from 3D point clouds, and a large number of different parameter combinations are tested to determine the optimal network architecture.
2) A few-data guided learning framework based on GPMM is established, upon that our network can be well trained with a small amount of real data.
3) The ablation study is analyzed to determine the parameters of the proposed network and the comparisons with other methods are conducted to prove the effectiveness and the potentiality of our method.

The rest of this paper is organized as follows:

Section 2 reviews the related work. Section 3 describes the proposed method including the network architecture and the training details. Section 4 presents the ablation study and the recognition results on several datasets. Section 5 discusses other features of our method such as computational complexity. Section 6 concludes the paper.

2 Related work

Since our method consists of a 3D face recognition network and a training framework that generates training data, we discuss the most relevant work on classic 3D face recognition, deep-learning-based 3D face recognition, and 3D face generation.

2.1 Classic 3D face recognition

Numerous 3D face recognition methods have been developed, as reviewed in [44, 50, 52]. At present, methods not using deep learning are generally regarded as classic ones, among which matching-based methods and feature-based methods are the two main branches.

Matching-based methods are quite intuitive. Mian et al. [38] proposed a multimodal face recognition system. For 3D face images, faces are automatically segmented into two regions: the eyes and forehead region and the nose region. These regions are thus matched with their corresponding ones using a modified Iterative Closest Point (ICP) method. Mian et al. used the shape variation in combination with 2D SIFT descriptors and achieved good performance. Also, Iterative Closest Normal Point (ICNP) [40] is used to match face surfaces. After matching facial scan with average facial model, linear discriminant analysis (LDA) is applied to identify faces. This method achieves high accuracy on database, but LDA cannot work without a generic training set when only a single scan is available for each subject. Yu et al. [60] proposed RDSICP, an enhanced version of ICP, to match facial scans for face verification.

These matching-based methods are computationally complex and slow in recognition speed. Generally, they are applicable only for 1:1 face verification.

In 1:N face identification scenario, feature-based methods are more preferable, and many artificially designed feature extractors have been proposed to extract face representations from 3D faces.

Some feature-based methods define several curves or sub-regions to extract features on the face. For example, Elaiwat et al. [18] calculated Curvelet transform to extract features from semi-rigid regions, presenting a multi-modal face identification approach. Emambakhsh and Evans [19] created a set of planes using pairs of landmarks and the features are extracted from the intersection of these planes with the nasal region curves. Li and Da [31] proposed facial curves classifier to handle facial expression and hair occlusion.

Some other methods design local descriptor to generate face representations. For example, Lei et al. [29] proposed a keypoint-based Multiple Triangle Statistics (KMTS) method to handle pose variations. Soltanpour et al. [49] proposed a keypoint-based method using curvature maps for 3D domain combined with 2D keypoints. Al-Osaimi [2] used a set of Rotation-invariant and Adjustable Integral Kernels (RAIKs) computed from the surface patch around a 3D point for keypoint-based matching. Berretti et al. [8] proposed stable keypoints and local descriptors for 3D face identification.

There are also methods that use features other than the above two categories, or combine multiple types of features, as reviewed in [50].

2.2 Deep-learning-based 3D face recognition

As a latecomer, learning-based methods have gradually raised its head nowadays. At present, the existing deep-learning-based 3D face recognition can be roughly divided into two branches.

The first branch still uses 2D face recognition, but the training data is augmented through a 3D model to improve the performance of original 2D face recognition. For example, Neves and Proença [41] utilized 3D models to generate caricatures and trained neural network from caricatures. Song et al. [51] built 3D models to generate 2D images to improve the accuracy of 2D methods.

The other branch uses artificially designed feature extractors to project 3D point cloud into 2D range images. For example, Gilani and Mian [63] and Kim et al. [26] employed existing 2D deep neural networks to solve the 3D face recognition problem by projecting 3D point clouds into 2D space as depth map, azimuth map, and elevation map. Cai et al. [12] proposed a deep learning technique based on several facial component patches cropped from depth map of the 3D face.

To conclude, among these deep-learning-based 3D face recognition methods, the majority are still based on 2D networks. Though entitled ”3D”, the 3D data are firstly projected into 2D images and then the traditional 2D networks are utilized to solve the recognition problem. Some of these methods work properly, but yet they do not make full use of the features available in 3D data.

2.3 3D face generation

Whether deep learning can achieve good results, to a great extent, depends on training data. There are many 2D face generation methods, some of which generate faces of different identities [20], while others generate facial appearances such as rejuvenation, aging, and cosmetics [5]. These 2D face generation methods have played an important role in training 2D networks. 3D face data are even more inadequate. As reviewed in [55], numerous 3D face generation methods have been developed.

Some 3D face generation methods are aimed at visual fidelity. For example, GANFIT [21] and MMFace [59] utilized 2D faces to reconstruct 3D faces. Similarly, Liu et al. [33] generated 3D faces from 2D faces through cascaded regression. These methods are designed for game or display industry rather than training face recognition network.

Some other methods use 3D models, but generate 2D data. For example, Gilani and Mian [63] and Kim et al. [26] proposed data augmentation methods to generate millions of range images. These data are specially generated for training their 3D face recognition networks. Apparently, these augmentation methods are not applicable for end-to-end networks directly using point clouds as the input.

Different from the above two categories, the goal of 3D face morphable model (3DMM) [10] is to generate 3D face data more realistic, and upon that Dou et al. [17] reconstructed 3DMM parameters with a deep neural network. Similarly, Lüthi et al. [37] proposed GPMM face model, a generalization of point distribution models. These methods can generate 3D point clouds and are more preferable for training end-to-end networks.

3 Methodology

In this section, we firstly introduce the architecture of our network, and then the training details are addressed. The overall procedure of the proposed method is shown in Fig. 1.

3.1 Network architecture

Different from the traditional learning-based 3D face recognition methods [26, 63], our method designs an end-to-end network directly inputting the coordinate of point cloud, so as to maintain the advantages of 3D data such as rotation invariance and transformation invariance. The output of our network is a feature vector, and the cosine distance between two feature vectors is calculated to reflect the probability that the two input faces are grabbed from the same subject.

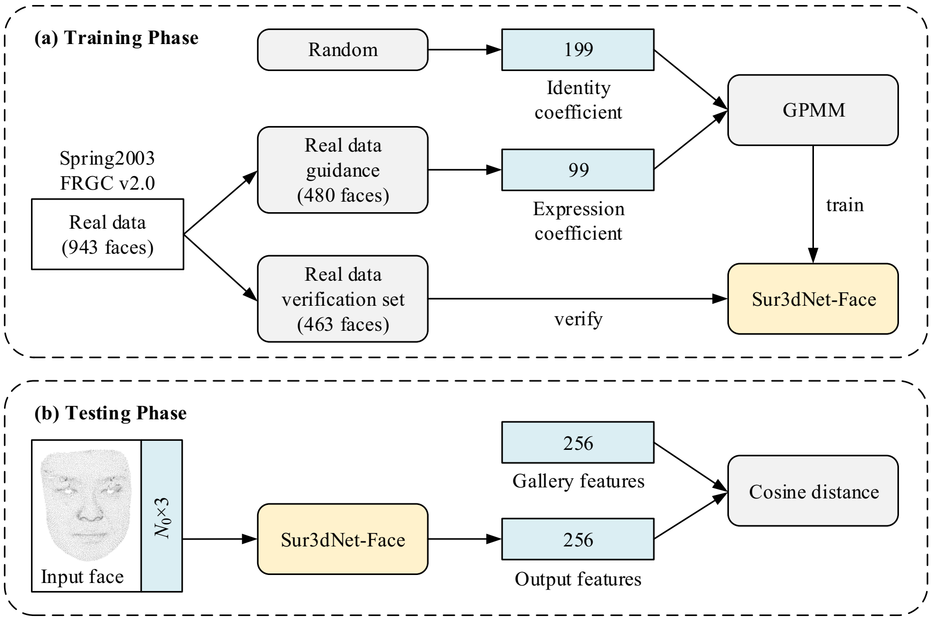

Fig. 1 An overview of the proposed few-data guided learning framework for face recognition. The Random set contains an array of random parameters representing different identities. Random sets of different sizes (500/1000/3000/6000/10000) are used for training and testing accuracy (refer to Section 4.3.4). The Sur3dNet-Face module is introduced in Section 3.1, the GPMM module in Section 3.4, and the Cosine distance module in Section 3.3.2

Specifically, the forward process of our network can be represented as:

$$f = Sur3dNet\left(\Gamma\right) \tag{1}$$

where $\Gamma = \left\{x_1, x_2, \cdots, x_{N_0}\right\} \in \mathbb{R}^{N_0 \times 3}$ is the unordered input point cloud, N_0 denotes the number of points, $f \in \mathbb{R}^{256}$ is the output features.

The architecture of our proposed network is shown in Fig. 2 and the submodules in the figure will be introduced in the subsequent subsections.

3.2 Normal estimation

As is known, the normal vector is one of the most important attributes of point clouds. We calculate the normal vectors $n \in \mathbb{R}^{N_0 \times 3}$ of the input point cloud Γ through Principal Component Analysis (PCA), during which the corresponding eigen values $e \in \mathbb{R}^{N_0 \times 1}$ can also be derived, which reflect the curvature of the surface. Therefore, the output of the normal estimation submodule is $\left[\,\Gamma \;\; n \;\; e\,\right] \in \mathbb{R}^{N_0 \times 7}$.

3.3 Modified PointNet

PointNet [13] learns a function that maps a set of points to a feature vector, where multi-layer perceptrons (MLPs) are applied to every point individually before a max-pooling layer that aggregates features of all points to a global vector.

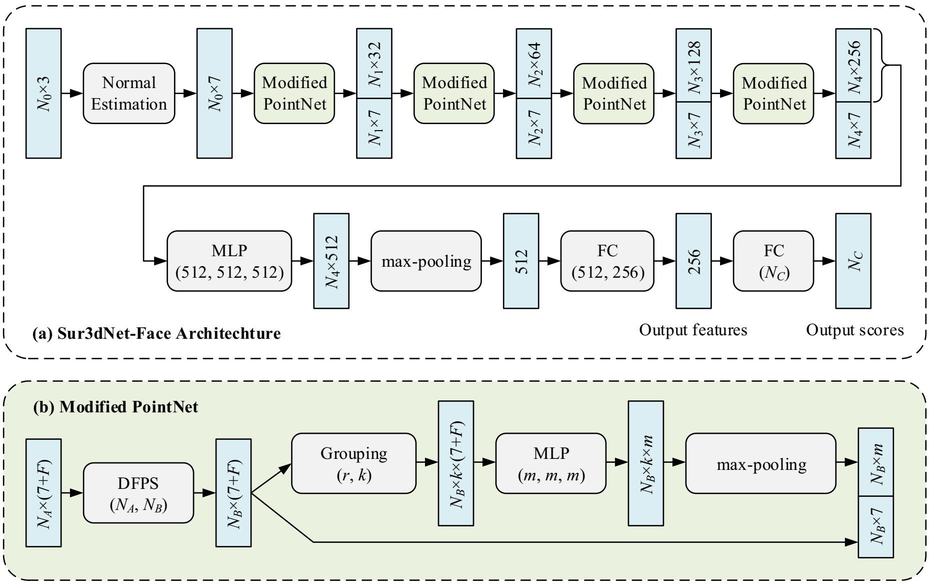

Fig. 2 Architecture of our proposed Sur3dNet-Face, an end-to-end point cloud network for 3D face recognition

Our network uses PointNet as the backbone and the modifications are as follows.

3.3.1 Ball query with physical size

The coordinates of all objects are normalized to the range of $(-1, 1)$ in PointNet. However, size is an important attribute of faces, which will be lost in the normalization process. To avoid this, we discard the normalization process and directly input the point cloud with original physical size with the unit of millimeter, and therefore, the radius of ball query in our network should also be measured in millimeters.

3.3.2 Dithering farthest point sampling (DFPS)

The farthest point sampling (FPS) algorithm contains an iterative process. The input point set is Γ and output set is S, where

$$\begin{aligned} \Gamma &= \left\{x_1, x_2, \cdots, x_{N_A}\right\} \\ S &= \left\{x_1, x_2, \cdots, x_{N_B}\right\} \end{aligned} \tag{2}$$

The FPS process will contain N_B iterations, and the formula for each iteration is as follows:

$$x_j = \arg\max\left(\min d\left(x_i, x_j\right)\right) \tag{3}$$

where $x_i \in S$ is the points already taken out before this iteration, $x_j \in \Gamma$ is the point to be taken out in this iteration, $d\left(x_i, x_j\right)$ is the Euclidean distance between x_i and x_j, so that $x_j \in \Gamma$ is the most distant point relative to the set S.

According to our analysis, the main reason for the poor performance of PointNet in face recognition is that FPS preferentially selects points at the edge region of the point cloud. For the closed point clouds generated from CAD model (e.g. ModelNet dataset), this strategy ensures that more corner points can be taken out. However, as compared in Fig. 3a, in practical scenario, on the ground that the point cloud is grabbed from one direction, barely including one perspective of the object, the edge points are rather unstable. Therefore, the recognition rate of PointNet on the actual measured point cloud is far below expectation, especially on the face with different posture, where the boundary line can be quite different. Meanwhile, for the edge points, the neighbors clustered by ball query are aggregated on one side of the center, leading to high susceptibility of features to pose variation. To solve these problems, we propose dithering farthest point sampling (DFPS) as follows:

$$x_j = \arg\max\left(\min \lambda d\left(x_i, x_j\right)\right) \tag{4}$$

where

$$\lambda_j = \begin{cases} 0, d\left(x_j, x_{NT}\right) > R \\ e_j^p, else \end{cases} \tag{5}$$

where x_{NT} is the coordinate of nose tip, e_j is the eigen value of x_j (as mentioned before in normal estimation subsection), p is the weighting factor, R is the valid radius beyond which the λ_j will be set to 0 and the corresponding x_j will never be selected by DFPS.

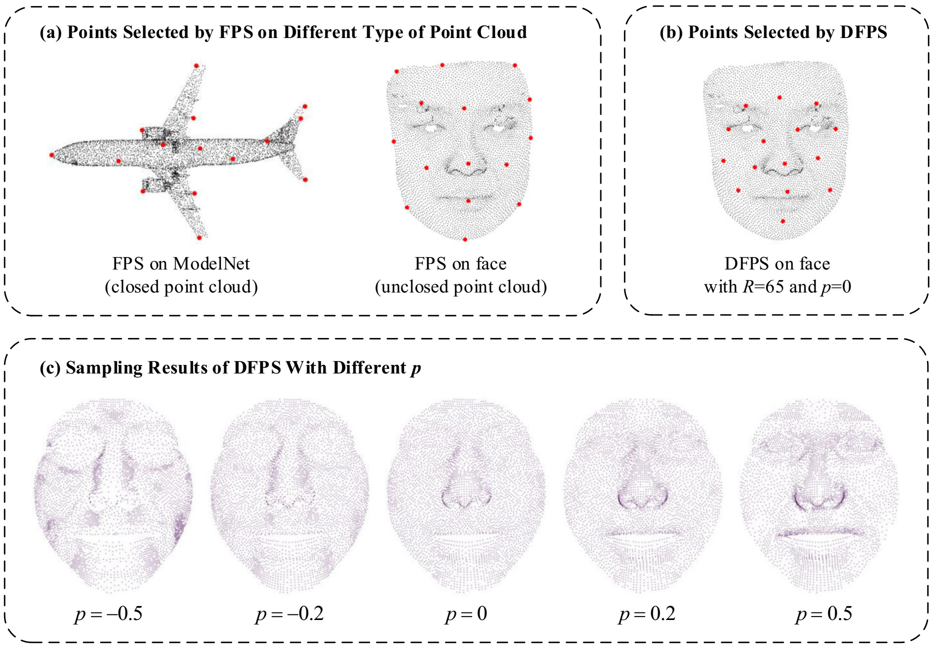
(a) Points Selected by FPS on Different Type of Point Cloud
FPS on ModelNet
(closed point cloud)
FPS on face
(unclosed point cloud)
(b) Points Selected by DFPS
DFPS on face
with R=65 and p=0
(c) Sampling Results of DFPS With Different p
p=−0.5
p=−0.2
p=0
p=0.2
p=0.5

Fig. 3 Analysis on the dithering farthest point sampling (DFPS). In (**a**) and (**b**), the black points are the input and the first 15 points selected by the algorithm are marked in red. (**c**) shows the sampling results of DFPS with different p, where the corner points are more likely to be selected as p increases

DFPS has different behavior in training phase and testing phase. During training, random variations are added to R and p to improve the adaptability of the network. Specifically, in the testing phase $R = 65$ and $p = 0$, while in the training phase, R and p are random numbers in range of (50, 80) and (−0.2, 0.2) respectively.

The first row of (5) ensures that the points output by FPS are mostly selected in the central area that contains abundant facial features, avoiding the influence of unstable edge points, as is shown in Fig. 3b.

Additionally, it is necessary to explain the second row of (5) to analyze its principle. As a rule, there is a lot of noise on the real captured point clouds. Owing to the tendency for FPS to select distant points, corner points and noise floating outside the surface of point cloud are prone to be selected. As the eigen value e_j reflects the smoothness near the point x_j, by adjusting the value of p, DFPS will tend to select corner points (when p increases) or points on the smooth surface (when p decreases), as is shown in Fig. 3c. For different measurement system, the smoothness of the output point cloud can be quite different, leading to difficulty in determining p, so we adopt a more concise strategy that randomly generates p in training phase. With different p, the coordinates of the sampled points will dither slightly, so that the trained network can adapt to different types of noise. Our strategy actually adds more randomness to the traditional FPS, and the experiments show that when supplemented with the proposed dithering strategy, the network gets better results.

As is shown in Fig. 2a, there are four modified PointNet layers in our network, and the parameters of each layer are shown in Table 1.

These modifications are seemingly simple, but for the task of face recognition, they are of great significance. In the experiments section, the impacts of these modifications on the recognition rate are compared intuitively through the ablation study.

3.4 Other details

Our network has no limitation for the number of input points, but in the training phase, point clouds with different sizes cannot be stacked into a batch. In order to use a larger batch size, the input point clouds are firstly downsampled to $N_0 = 24576$ through random choice. Note that there is no such limitation in the test phase or in practical use, where the N_0 denotes the actual number of input points.

Batch normalization layer and dropout layer can significantly improve the performance of the network, so we add both on the FC layers (except the last one), and the dropout rate is 0.5. We use Adam [27] optimizer to train the network with the initial learning rate of 1e-3 multiplied by a factor of 0.1 every 10 epochs. Also, the weight decay is set to 1e-4, batch size is 32, and the total number of epochs is 35.

Table 1 Parameters of the four modified PointNet layers

Parameter	Layer 1	Layer 2	Layer 3	Layer 4	Comments
N_A	24576	4096	1024	256	Input point number
N_B	4096	1024	256	64	Output point number
r	4	8	16	32	Radius of ball query (mm)
k	24	32	48	64	Ball query point number
F	0	32	64	128	Input feature number
m	32	64	128	256	Output feature number

3.5 Identification

Similar to traditional solutions, we take the 256 dimensional vector from the output of the penultimate FC layer as a face representation. After acquiring features of both gallery and probe, we calculate the cosine distance between them, and afterwards, identity of the probe is determined by the gallery with minimum distance. The cosine distance is defined as:

$$d\left(f_1, f_2\right) = 1 - \frac{f_1 \cdot f_2}{|f_1|\,|f_2|} \tag{6}$$

where f_1 and f_2 are the features (256 dimensional vectors representing the face) of gallery and probe.

3.6 Training data

3.6.1 Gaussian process morphable models

In order to generate sufficient training data, we use the GPMM face model [37], a generalization of point distribution models, which assumes that any shape Γ can be represented as a discrete set of points:

$$\Gamma = \{x_1, x_2, \cdots, x_N\} \tag{7}$$

where N denotes the number of points.

Shape Γ is represented as a vector $s \in \mathbb{R}^{3N}$:

$$s = \left[\, x_{1x} \;\; x_{1y} \;\; x_{1z} \;\; \cdots \;\; x_{Nx} \;\; x_{Ny} \;\; x_{Nz} \,\right]^T \tag{8}$$

GPMM model assumes that the shape variation can be modeled using a normal distribution:

$$s \sim \mathcal{N}(\mu, \Sigma) \tag{9}$$

According to the theory proposed by [37], any face $s \in \mathbb{R}^{3N}$ can be expressed as:

$$s = \bar{s} + B_S\sqrt{\Lambda_S}\alpha + B_E\sqrt{\Lambda_E}\beta \tag{10}$$

where the parameter α determines the shape of human face, β determines the expression of human face, and $\bar{s}$ is the mean face model. B_S and B_E are the basis respectively for shape and expression, and similarly, Λ_S and Λ_E are the variance. For $\bar{s}$, B_S, B_E, Λ_S, and Λ_E, we directly adopt the values provided by [37].

3.6.2 GPMM-based data generation

According to formula (10), the face generated by GPMM depends on two parameters, α and β, where α determines the identity and β determines the expression.

As GPMM is not specially designed for training neural networks, when using faces generated by formula (10) for training, the trained network can recognize the frontal faces. However, in practical application with posture and noise, in order to improve the variety of the data, the formula should be modified as:

$$s = f\left(\bar{s} + B_S\sqrt{\Lambda_S}\alpha + B_E\sqrt{\Lambda_E}\beta + \delta\right) \tag{11}$$

where $f(\bullet)$ is the random rotation function, $\delta \sim \mathcal{N}(0, \sigma_\delta^2)$ is the Gaussian noise, $\alpha \sim \mathcal{N}(0, \sigma_\alpha^2)$ is the shape coefficient, $\beta \sim \mathcal{N}(0, \sigma_\beta^2)$ is the expression coefficient.

3.6.3 Real data guided data generation

The randomly generating strategy appears difficult to cover the common expressions in the real faces. Taking the disgust expression as an example, we observe that few of the faces with random coefficients look like the disgust. To make our training data further closer to the real data, we propose an enhanced strategy matching the real face with GPMM model, so as to use the real data as a guidance for the expression coefficient to generate training data.

We denote a pair of real faces in the dataset as (Γ_N, Γ_E), where Γ_N and Γ_E are respectively the neutral face and the expressive face of a certain subject. According to [22], the registration problem can be solved as:

$$\begin{aligned} \alpha_N &= \arg\max_{\alpha} p(\alpha)\, p\left(\Gamma_N \left| \alpha, \bar{S}\right.\right) \\ \beta_E &= \arg\max_{\beta} p(\beta)\, p\left(\Gamma_E \left| \beta, \Gamma_N\right.\right) \end{aligned} \tag{12}$$

where α_N is the shape coefficient of both Γ_N and Γ_E, and β_E is the expression coefficient of Γ_E.

β_E of each expressive face in the Spring2003 subset of FRGC v2.0 are calculated for preparation. When generating a face, one of β_E is randomly selected to obtain β_F as:

$$\beta_F = \lambda\beta_E + (1 - \lambda)\,\beta \tag{13}$$

where λ is a random value between (0, 1), $\beta \sim \mathcal{N}(0, \sigma_\beta^2)$.

Afterwards, β_F is substituted into formula (11) to generate the face, so that the faces generated can cover more expressions similar to real faces in datasets. Experiments show that this strategy can considerably improve the recognition rate of faces with expressions.

3.7 Verification set

To avoid overfitting, a verification set separated from the training set is usually used in deep learning to verify the performance of the network. In our method, all the training samples are generated from GPMM model upon the same formula, leading to similar distribution. So if we directly take a part of the training samples as the verification set, the loss of the verification set will have almost the same trend as that of the training set, and thus cannot play the role of verifying whether the network is overfitted. Existing public datasets contain too few samples for training, but they are quite sufficient as the verification set. Therefore, we put forward a strategy training the network with the generated data and verifying upon the real data.

Since real data in the verification set do not share the identity with our training data, feature vectors extracted by the network from the real data are used to calculate the cosine distance. Through cosine distance, we can evaluate the recognition performance on the real data verification set through Rank-1 Recognition Rate (RR1), Verification Rate (VR) under FAR $= 1e-3$, and Area Under ROC Curve (AUC). Taking these three commonly used indicators into account, we define the loss of the verification set as:

$$loss = 1 - \mathrm{VR} \times \mathrm{RR1} \times \mathrm{AUC} \tag{14}$$

In the experiments, we observe that the loss initially decreases with the training epoch, and then it starts to increase, indicating that the network starts to overfit. The network parameters with the lowest verification loss are saved as the final result of the training process.

4 Experiments

In the experiments, the proposed network is implemented on PyTorch [43] and is training with i7-8700K CPU and two GTX1080TI GPU.

4.1 Datasets used

We use Face Recognition Grand Challenge (FRGC) v2.0 dataset [45] and Bosphorus dataset [47] to evaluate the performance of the proposed face recognition method.

FRGC v2.0 dataset [45] containing 4007 scans of 466 subjects in total is divided into three partitions as Spring2003 subset, Fall2003 subset, and Spring2004 subset. The 3D images consist of both a 3D poind cloud and a 2D texture image, and the data were acquired by a Minolta Vivid 900/910 series sensor. Protocol [45] uses the Spring2003 for training and the remaining for testing. In our experiments, we follow this protocol taking Spring2003 as real data to train the network. Specifically, we randomly select 480 faces in Spring2003 for real data guidance and the other 463 faces for real data verification set.

Bosphorus dataset [47] includes totally 4666 scans collected from 105 subjects (60 men and 45 woman aged between 25 and 35) with poses changes, expression variations, and four types of occlusions, and the yaw rotation of faces is from 10 degrees to 90 degrees. For each subject, 31-54 facial scans are available. The database is collected using the Inspeck Mega Capturor II 3D scanner, which is a structured light-based 3D scanner. In our experiments, the neutral scans with file name containing N_N_0 are used as the gallery.

4.2 Data preprocessing

Our network directly inputs the face point clouds, but the point clouds in datasets contain background and noise, which need to be removed as follows:

Firstly, the nose tip coordinates need to be obtained. Bosphorus dataset provides nose tip coordinates per se. FRGC v2 dataset, though without nose tip coordinates, provides one-to-one correspondence between the 2D images and 3D point clouds. So MTCNN [61] is adopted to detect nose tip on 2D images, and then the coordinates are mapped to 3D point clouds.

After the nose tip coordinates are obtained, points more than 90 mm away from the nose tip are removed, and the remaining points are input into our network for face recognition. Since our method uses an end-to-end design, no other preprocessing is required.

4.3 Ablation study

It takes about 50 hours on our hardware to train the network with the training set of 3000 identities, each with 200 different expressions. In order to make more comparisons in a shorter time, we use training set of 3000 identities for ablation study unless otherwise specified, and the rank-1 recognition rate on Bosphorus dataset excluding occlusion subset and posture subset is the main indicator to evaluate the performance.

4.3.1 Parameters in DFPS

There are two new parameters introduced in DFPS, namely R and p. Figure 4a demonstrates the rank-1 recognition rate on Bosphorus dataset by training the network with different R values under $p = 0 \pm 0.2$. It can be seen from the blue line that when $45 \leqslant R \leqslant 75$, the recognition rate has little difference. Meanwhile, when the random R strategy is adopted, where R is added by a random variation between -15 and 15 during training phase, as shown in orange line, the recognition rate is further improved.

Figure 4b depicts the rank-1 recognition rate on Bosphorus dataset by training the network with different p values under $R = 65 \pm 15$. In fact, even if we use the same parameters for training, the recognition rate will occasionally show a variation of 0.5% in each reproduction. Owing to the relatively small impact of parameter p on the results, it appears difficult to determine which is the best value from these experimental results. However, as can be seen from the figure, the random strategy plays a positive role.

Through this ablation study, the parameters of DFPS is determined as follows: in the testing phase $R = 65$ and $p = 0$, while in the training phase, R and p are random numbers in range of (50, 80) and $(-0.2, 0.2)$ respectively for each mini-batch.

4.3.2 Ball query radius

Ball query radius has a significant impact on the recognition rate. We observed that when r and k match a certain proportion, the recognition rate goes higher. Specifically, a smaller r is more suitable for a smaller k, and vice versa. Among all the tested combinations, as shown in Table 2, the combination of $r = 4, 8, 16, 32$ and $k = 24, 32, 48, 64$ gets the best result, and these parameters are also given in Table 1. Note that the recognition rate in Table 2 is obtained without using real data guided generation and verification.

4.3.3 Real data guided generation and verification

The previous experiments are aimed at network parameters, among which the highest rank-1 recognition rate we achieve is 97.85%. Afterwards, we make further experiments verifying the effect of training with a small amount of real data.

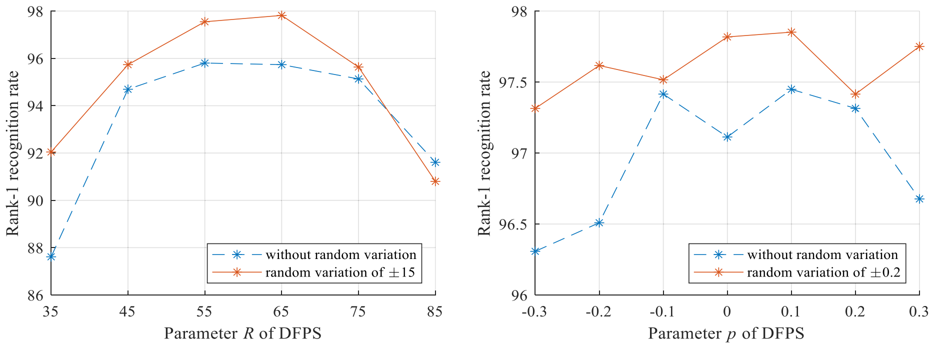

Fig. 4 Comparisons on different R and p. The results in this figure are obtained without using real data guided generation and verification

Table 2 Rank-1 Recognition Rate (RR1) under different combination of r and k on Bosphorus dataset

r	k	RR1
3,6,12,24	16,24,32,48	86.97%
3,6,12,24	24,32,48,64	80.53%
4,8,16,32	16,24,32,48	95.17%
4,8,16,32	24,32,48,64	97.85%
4,8,16,32	40,48,56,64	96.66%
5,10,20,40	24,32,48,64	94.50%
5,10,20,40	40,48,56,64	95.15%

There are two real-data-based techniques proposed in this paper, namely, real data guided generation and real data verification set. In the experiments, we randomly select about half of the faces in Spring2003 subset of FRGC v2.0 for real data guided generation and the other half for real data verification set. The results show that our network achieves rank-1 recognition rate of 98.29% with the former technique, real data guided generation, and when both techniques are applied, the rank-1 recognition rate reaches 98.83% on Bosphorus dataset.

4.3.4 Training data volume

The training data can be arbitrarily generated, but due to the limitation of hardware performance, we test 10000 identities at most, each containing 200 expressions, and the recognition rate on Bosphorus dataset is displayed in Fig. 5.

From the experimental results, the rank-1 recognition rate reaches 99.33% of training with 10000 identities each with 200 expressions. Also, the increasing trend of recognition rate in the figure indicates that if a larger data volume is used, a slightly higher recognition rate can possibly be obtained.

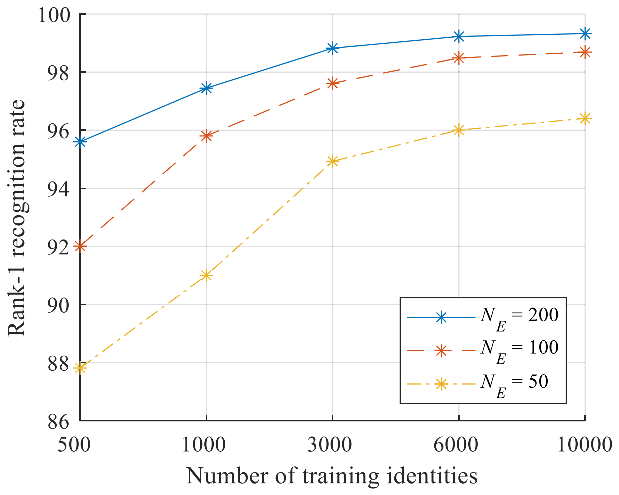

Fig. 5 Comparisons on different training volume. N_E denotes the number of generated expressions for each identity

4.4 Comparisons with other methods

Our method is compared with several representative face recognition methods on FRGC v2.0 [45] dataset and Bosphorus [47] dataset. Judging by the experimental results, though our method does not defeat all the methods, it is on a par with state-of-the-art methods in accuracy. As is known, while many methods have achieved decent accuracy rate, the slight difference in accuracy rate is no longer the main indicator for choosing a method. More importantly, our method has following advantages: 1) The network is end-to-end without complex preprocessing operations. This feature makes our method faster (detailed in discussions section) and easy to deploy on edge computing chips. 2) The training does not require a large number of datasets, nor any publicly unavailable datasets. With accuracy on a par with state-of-the-art methods, these advantages make our method a competitive solution. The detailed experimental data are shown below.

4.4.1 Results on FRGC v2.0

The comparisons between our proposed method and the state-of-the-art methods on FRGC v2.0 dataset are shown in Table 3, where the most representative face recognition methods are compared. Some methods use the corresponding 2D photos of the 3D faces, which we denote as (2D+3D). Also, there are some methods using fine-tuning to further improve the recognition rate, which are marked with (FT).

It can be seen from the table that some deep-learning-based methods have achieved satisfactory recognition rate in recent years, especially those with fine-tuning that can considerably improve the recognition rate. However, in the actual application scenario, due to the dynamic changes of the face gallery, the effect of fine-tuning is much lower than expectation. Among those methods without using fine-tuning, our method is quite competitive, which achieves rank-1 recognition rate of 98.85% and verification rate of 96.75% on FRGC v2.0 dataset.

In the verification experiments on FRGC v2 dataset, with the threshold of FAR $= 1e - 3$, some other criteria are calculated as follows: accuracy 98.76%, specificity 0.10%, sensitivity 97.70%, precision 99.90%.

Table 3 Rank-1 Recognition Rate (RR1) and Verification Rate (VR) under FAR $= 1e - 3$ on FRGC v2.0 dataset

Method	RR1	VR
Huang et al. [24] (2012)	97.60%	98.40%
Liu et al. [35] (2013)	96.94%	90.00%
Elaiwat et al. [18] (2015)	97.10%	99.20%
Lei et al. [30] (2016)	96.30%	98.30%
Al-Osaimi [3] (2016)	96.49%	98.69%
Ouamane et al. [42] (2017)	–	96.65%
Ouamane et al. [42] (2017) (2D+3D)	–	98.32%
Gilani et al. [23] (2018)	98.50%	98.70%
Gilani and Mian [63] (2018)	97.06%	–
Gilani and Mian [63] (2018) (FT)	99.88%	–
Cai et al. [12] (2019) (FT)	100%	100%
Ours	98.85%	96.75%

As is known, whether deep learning can achieve good results, to a great extent, depends on training data. Different from those methods training with more and more data to get a slightly higher recognition rate, the only real data we use in the training process are the 943 faces in Spring2003 subset. With such a small amount of data, we achieve a higher recognition rate than some of the latest methods, which proves the effectiveness and the potentiality of our method.

4.4.2 Results on Bosphorus

The comparisons between our method and the state-of-the-art methods on Bosphorus dataset are shown in Table 4. This paper does not involve occlusion and large posture, so the occlusion subset and the posture subset are excluded. In order to make a fair comparison, the recognition rate under the same subsets of Bosphorus is displayed.

In the verification experiments on Bosphorus dataset, with the threshold of FAR $= 1e - 3$, some other criteria are calculated as follows: accuracy 98.32%, specificity 0.10%, sensitivity 96.75%, precision 99.90%.

5 Discussions

Although some existing methods have achieved satisfactory recognition rate, our method has some notable advantages, upon that we make further discussions.

5.1 Computational complexity

Our method designs an end-to-end network directly inputting the point clouds, which is faster than those methods requiring a complex preprocessing. For example, [30] reports that they need 3.16 s for preprocessing, including detection and alignment, region segmentation, model registration, and other steps; and [26] reports that they need 6.08 s to generate the 2D images including depth map, azimuth map, and elevation map from 3D point clouds.

Table 4 Rank-1 Recognition Rate (RR1) and Verification Rate (VR) under FAR $= 1e - 3$ on Bosphorus dataset

Method	RR1	VR
Huang et al. [24] (2012)	97.00%	–
Liu et al. [35] (2013)	95.63%	81.40%
Berretti et al. [7] (2013)	95.67%	–
Elaiwat et al. [18] (2015)	–	91.10%
Lei et al. [30] (2016)	98.90%	–
Al-Osaimi [3] (2016)	92.41%	93.5%
Ouamane et al. [42] (2017) (2D+3D)	–	96.17%
Gilani et al. [23] (2018)	98.5%	–
Gilani and Mian [63] (2018)	96.18%	–
Gilani and Mian [63] (2018) (FT)	100%	–
Cai et al. [12] (2019) (FT)	99.75%	98.39%
Ours	99.33%	97.70%

On the hardware platform described in experiments section, in the training phase with the faces generated in advance, the forward time and backward time of our network for each mini-batch (with batch size of 32) is 0.208 s and 0.095 s, respectively. When being applied to actual applications, it takes about 0.035 s to detect the nose tip and calculate the normal vectors, and 0.105 s to extract the feature of a single face using a single GPU, and there is no additional preprocessing required.

5.2 Data generation and overfitting

There is a common problem in existing deep-learning-based face recognition methods, that is, owing to the lack of data, more faces need to be generated by interpolating existing datasets with each other [26, 63]. Although the generated faces used for training are considered different, they have implicit overlaps with the test set, meaning that the results may be obtained to some extent by overfitting.

Comparably, the training data in our method are absolutely generated from GPMM model, which have little intersection with real datasets. Therefore, what we listed are actually cross-dataset results without any fine-tuning, which are closer to the results in actual application than results of those methods that take part of the datasets to generate training data and use the other part to test.

6 Conclusion

An end-to-end deep learning network entitled Sur3dNet-Face for point-cloud-based 3D face recognition is presented in this paper, along with the concrete approach for training. Coupled with real data guided generation and real data verification set, a few-data guided learning framework based on Gaussian process morphable model is proposed, overcoming the common problem of lacking training data in 3D face deep learning.

Different from existing methods training with more and more data to get a slightly higher recognition rate, the only real data we use in the training process are the 943 faces in Spring2003 subset of FRGC v2.0. With such a small amount of data, we achieve recognition rate on a par with state-of-the-art methods, which proves the effectiveness and the potentiality of our method. Specifically, without any fine-tuning on test set, the Rank-1 Recognition Rate (RR1) and Verification Rate (VR) are achieved as follows: 98.85% (RR1) and 96.75% (VR) on FRGC v2.0 dataset, and 99.33% (RR1) and 97.70% (VR) on Bosphorus dataset.

To conclude, our method has following advantages: 1) The network is end-to-end without complex preprocessing operations, which means it performs efficiently and is easy to be deployed on edge computing chips. 2) The training does not require a large number of datasets, nor any publicly unavailable datasets, which makes our method a really economic solution for cost-sensitive and computing-resource-limited scenarios. With accuracy on a par with state-of-the-art methods, these advantages make our method even more competitive.

In the future research, we are going to 1) generate more occluded faces, angled faces, and other facial appearances such as aging and cosmetics [5] to expand the applicability of our method; 2) adapt the pattern separation network [39] to points cloud data to solve overlapping problems and to obtain better results on 3D face recognition; 3) explore the use of triplet loss function [14] in our training framework so as to further improve the performance of our network.

Declarations

Conflict of Interests The authors confirm that there are no conflicts of interest associated with this publication.

References

1. Ahonen T, Hadid A, Pietikäinen M (2004) Face recognition with local binary patterns. In: European conference on computer vision. Springer, pp 469–481
2. Al-Osaimi FR (2016) A novel multi-purpose matching representation of local 3d surfaces: a rotationally invariant, efficient, and highly discriminative approach with an adjustable sensitivity. IEEE Trans Image Process 25(2):658–672
3. Al-Osaimi FR (2016) A novel multi-purpose matching representation of local 3d surfaces: a rotationally invariant, efficient, and highly discriminative approach with an adjustable sensitivity. IEEE Trans Image Process 25(2):658–672
4. Ali W, Tian W, Din SU, Iradukunda D, Khan AA (2021) Classical and modern face recognition approaches: a complete review. Multimed Tools Appl 80:4825–4880
5. Bastanfard A, Bastanfard O, Takahashi H, Nakajima M (2004) Toward anthropometrics simulation of face rejuvenation and skin cosmetic. Comput Anim Virt Worlds 15(3–4):347–352
6. Belhumeur PN, Hespanha JP, Kriegman DJ (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. IEEE Trans Pattern Anal Mach Intell 19(7):711–720
7. 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
8. Berretti S, Werghi N, del Bimbo A, Pala P (2014) Selecting stable keypoints and local descriptors for person identification using 3d face scans. Vis Comput 30:1275–1292
9. Bhople AR, Shrivastava AM, Prakash S (2020) Point cloud based deep convolutional neural network for 3d face recognition. Multimedia Tools and Applications, 1–23
10. Blanz V, Vetter T (1999) A morphable model for the synthesis of 3d faces. In: Proceedings of the 26th annual conference on computer graphics and interactive techniques. ACM Press/Addison-Wesley Publishing Co., USA, SIGGRAPH '99, pp 187–194
11. Cai Y, Yang M, Li Z (2015) Robust head pose estimation using a 3d morphable model. Math Probl Eng 2015(5):1–10
12. Cai Y, Lei Y, Yang M, You Z, Shan S (2019) A fast and robust 3d face recognition approach based on deeply learned face representation. Neurocomputing 363:375–397
13. Charles RQ, Su H, Kaichun M, Guibas LJ (2017) Pointnet: deep learning on point sets for 3d classification and segmentation. In: 2017 IEEE Conference on computer vision and pattern recognition (CVPR), pp 77–85
14. Cheng D, Gong Y, Zhou S, Wang J, Zheng N (2016) Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1335–1344
15. Creusot C, Pears N, Austin J (2013) A machine-learning approach to keypoint detection and landmarking on 3d meshes. Int J Comput Vis 102(1):146–179
16. Deng J, Guo J, Xue N, Zafeiriou S (2019) Arcface: additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)
17. Dou P, Shah SK, Kakadiaris IA (2017) End-to-end 3d face reconstruction with deep neural networks. In: 2017 IEEE Conference on computer vision and pattern recognition (CVPR), pp 1503–1512
18. Elaiwat S, Bennamoun M, Boussaid F, El-Sallam A (2015) A curvelet-based approach for textured 3d face recognition. Pattern Recogn 48(4):1235–1246
19. Emambakhsh M, Evans A (2017) Nasal patches and curves for expression-robust 3d face recognition. IEEE Trans Pattern Anal Mach Intell 39(5):995–1007
20. Gauthier J (2014) Conditional generative adversarial nets for convolutional face generation. Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, vol 2014
21. Gecer B, Ploumpis S, Kotsia I, Zafeiriou S (2019) Ganfit: generative adversarial network fitting for high fidelity 3d face reconstruction. In: 2019 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), pp 1155–1164
22. Gerig T, Morel-Forster A, Blumer C, Egger B, Luthi M, Schoenborn S, Vetter T (2018) Morphable face models - an open framework. In: 2018 13th IEEE International conference on automatic face & gesture recognition (FG 2018), pp 75–82

23. Gilani SZ, Mian A, Shafait F, Reid I (2018) Dense 3d face correspondence. IEEE Trans Pattern Anal Mach Intell 40(7):1584–1598
24. Huang D, Ardabilian M, Wang Y, Chen L (2012) 3-d face recognition using elbp-based facial description and local feature hybrid matching. IEEE Trans Inform Forens Secur 7(5):1551–1565
25. Huang GB, Mattar M, Berg T, Learned-Miller E (2008) Labeled faces in the wild: a database for studying face recognition in unconstrained environments. In: Workshop on faces in'Real-Life'Images: detection, alignment, and recognition
26. Kim D, Hernandez M, Choi J, Medioni G (2017) Deep 3d face identification. In: 2017 IEEE International joint conference on biometrics (IJCB), pp 133–142
27. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: Bengio Y, LeCun Y (eds) 3rd International conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings
28. Kollreider K, Fronthaler H, Bigun J (2008) Verifying liveness by multiple experts in face biometrics. In: 2008 IEEE Computer society conference on computer vision and pattern recognition workshops. IEEE, pp 1–6
29. 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(C):218–237
30. 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
31. Li X, Da F (2012) Efficient 3d face recognition handling facial expression and hair occlusion. Image Vis Comput 30(9):668–679
32. Li Y, Bu R, Sun M, Wu W, Di X, Chen B (2018) Pointcnn: convolution on x-transformed points. Advances in Neural Information Processing Systems 31:820–830
33. Liu F, Zeng D, Li J, Zhao Qj (2017) On 3d face reconstruction via cascaded regression in shape space. Frontiers of Information Technology & Electronic Engineering
34. Liu J, Ni B, Li C, Yang J, Tian Q (2019) Dynamic points agglomeration for hierarchical point sets learning. In: 2019 IEEE/CVF International conference on computer vision (ICCV), pp 7545–7554
35. Liu P, Wang Y, Huang D, Zhang Z, Chen L (2013) Learning the spherical harmonic features for 3-d face recognition. IEEE Trans Image Process 22(3):914–925
36. Liu W, Wen Y, Yu Z, Li M, Raj B, Song L (2017) Sphereface: deep hypersphere embedding for face recognition. In: Proceedings of the IEEE Conference on computer vision and pattern recognition (CVPR)
37. Lüthi M, Gerig T, Jud C, Vetter T (2018) Gaussian process morphable models. IEEE Trans Pattern Anal Mac Intell 40(8):1860–1873
38. 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
39. Modhej N, Bastanfard A, Teshnehlab M, Raiesdana S (2020) Pattern separation network based on the hippocampus activity for handwritten recognition. IEEE Access 8:212803–212817
40. Mohammadzade H, Hatzinakos D (2013) Iterative closest normal point for 3d face recognition. IEEE Trans Pattern Anal Mach Intell 35(2):381–397
41. Neves J, Proença H (2018) A leopard cannot change its spots: improving face recognition using 3d-based caricatures. IEEE Transactions on Information Forensics and Security pp 1–1
42. Ouamane A, Chouchane A, Boutellaa E, Belahcene M, Bourennane S, Hadid A (2017) Efficient tensor-based 2d+3d face verification. IEEE Trans Inform Forens Secur 12(11):2751–2762
43. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Kopf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S (2019) Pytorch: an imperative style, high-performance deep learning library. In: Advances in neural information processing systems, vol 32. Curran Associates, Inc., pp 8024–8035
44. Patil H, Kothari A, Bhurchandi K (2015) 3-d face recognition: features, databases, algorithms and challenges. Artif Intell Rev 44(3):393–441
45. Phillips PJ, Flynn PJ, Scruggs T, Bowyer KW, Chang J, Hoffman K, Marques J, Min J, Worek W (2005) Overview of the face recognition grand challenge. In: 2005 IEEE Computer society conference on computer vision and pattern recognition, pp 947–954
46. Qi CR, Yi L, Su H, Guibas LJ (2017) Pointnet++: deep hierarchical feature learning on point sets in a metric space. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R (eds) Advances in neural information processing systems, vol 30. Curran Associates, Inc., pp 5099–5108
47. Savran A, Alyüz N, Dibeklioğlu H, Çeliktutan O, Gökberk B, Sankur B, Akarun L (2008) Bosphorus database for 3d face analysis. In: Schouten B, Juul NC, Drygajlo A, Tistarelli M (eds) Biometrics and identity management. Springer, Berlin, pp 47–56
48. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on computer vision and pattern recognition (CVPR)

49. Soltanpour S, Wu QJ (2017) Multimodal 2d-3d face recognition using local descriptors: pyramidal shape map and structural context. IET Biometrics 6(1):27–35
50. Soltanpour S, Boufama B, Wu QMJ (2017) A survey of local feature methods for 3d face recognition. Pattern Recogn, 72
51. Song X, Feng ZH, Hu G, Kittler J, Wu XJ (2018) Dictionary integration using 3d morphable face models for pose-invariant collaborative-representation-based classification. IEEE Trans Inform Forens Secur 13(11):2734–2745
52. Sun Y, Wang X, Tang X (2014) Deep learning face representation from predicting 10,000 classes. In: 2014 IEEE Conference on computer vision and pattern recognition, pp 1891–1898
53. Taskiran M, Kahraman N, Erdem CE (2020) Face recognition: past, present and future (a review). Digital Signal Process 106:102809
54. Thomas H, Qi CR, Deschaud JE, Marcotegui B, Goulette F, Guibas LJ (2019) Kpconv: flexible and deformable convolution for point clouds. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV)
55. Toshpulatov M, Lee W, Lee S (2021) Generative adversarial networks and their application to 3d face generation: a survey. Image Vis Comput 108:104119
56. Turk MA, Pentland AP (1991) Face recognition using eigenfaces. In: Proceedings 1991 IEEE computer society conference on computer vision and pattern recognition. IEEE Computer Society, pp 586–587
57. Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM (2019) Dynamic graph cnn for learning on point clouds. ACM Trans Graph 38:5
58. Yi D, Lei Z, Liao S, Li SZ (2014) Learning face representation from scratch 1411:7923
59. Yi H, Li C, Cao Q, Shen X, Li S, Wang G, Tai Y (2019) Mmface: a multi-metric regression network for unconstrained face reconstruction. In: 2019 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), pp 7655–7664
60. Yu Y, Da F, Guo Y (2019) Sparse icp with resampling and denoising for 3d face verification. IEEE Trans Inform Forens Secur 14(7):1917–1927
61. Zhang K, Zhang Z, Li Z, Qiao Y (2016) Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Process Lett 23(10):1499–1503
62. Zhang Z, Da F, Yu Y (2019) Data-free point cloud network for 3d face recognition. arXiv:1911.04731
63. Zulqarnain Gilani S, Mian A (2018) Learning from millions of 3d scans for large-scale 3d face recognition. In: The IEEE Conference on computer vision and pattern recognition (CVPR)