PointFace: Point Cloud Encoder-Based Feature Embedding for 3-D Face Recognition

Changyuan Jiang, Shisong Lin, Wei Chen, Feng Liu, *Member, IEEE*, and Linlin Shen, *Senior Member, IEEE*

***Abstract*—The accuracy of 2D face recognition (FR) has progressed significantly due to the availability of large-scale training data. However, the research of deep learning based 3D FR is still in the early stage. Most of available 3D FR generate 2D maps from 3D data and apply existing 2D CNNs to the generated 2D maps for feature extraction. We propose in this paper a light-weight framework, named PointFace, to directly process point set data for 3D FR. In this framework, two weight-shared encoders are designed to directly extract features from a pair of 3D faces and the distances between embeddings of the same person and different person are minimized and maximized, respectively. The framework also use a feature similarity loss to guide the encoders to obtain discriminative face representations. A pair selection strategy is proposed to generate positive and negative face pairs to further improve the FR performance. Extensive experiments on Lock3DFace and Bosphorus show that the proposed PointFace outperforms state-of-the-art 2D CNN based FR methods.**

***Index Terms*—3D face recognition, point cloud processing, deep learning, CNN.**

I. Introduction

FACE recognition has been one of the most commonly used biometric technologies for personal identification. During past years, deep learning based 2D face recognition (FR) has achieved great success and has been claimed to surpass human performance [1], [2], [3] due to the development of 2D convolution neural networks (CNNs). Despite of massive accessible training data and carefully designed 2D CNNs, 2D FR is still challenged by the intrinsic limitations of 2D images, such as significant change of pixel values caused by various illumination conditions and self-occlusions caused by different head poses [4], [5]. Compared with 2D images, 3D face data, e.g., point cloud, can provide richer geometric information intrinsically invariant to pose and illumination changes [5], [6]. The characteristic of 3D face data can help face recognition systems overcome the inherent drawbacks of 2D face recognition. Besides, previous works [6], [7], [8] proved that 3D face recognition has the potential to address aforementioned challenges in the 2D domain. Thus, 3D face recognition has become an active research topic in recent years.

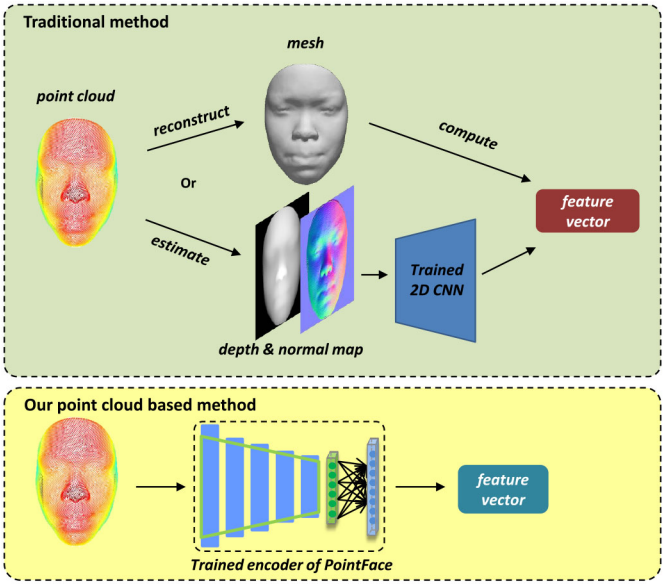


Fig. 1. Difference between our point cloud based method and traditional method (including hand-crafted based method and deep learning based method). Traditional pipelines either reconstruct and extract features from 3D polygon surfaces or utilize 2D CNNs to extract features from 2D geometric maps estimated from point clouds. Compared with traditional method, our proposed pipeline is much more efficient and simple. We directly extract features from point clouds for 3D face recognition.

However, in 3D FR, most traditional solutions including hand-crafted feature based methods and deep learning based methods have several defects. Firstly, they both require data conversions before feature extraction, which requires large computational costs. Specifically, many hand-crafted feature based methods [6], [8], [9], [10] need to reconstruct 3D mesh (also called 3D polygon surfaces) from point clouds or depth images and then compute feature descriptors from key points, curvatures, shape indexes and curves of the reconstructed mesh (shown in Figure 1). In addition, compared with deep learning based methods, many hand-crafted feature based methods do not generalize well on face data with noise or with pose

Manuscript received 22 February 2022; revised 21 June 2022; accepted 31 July 2022. Date of publication 10 August 2022; date of current version 5 December 2022. This work was supported by the National Natural Science Foundation of China under Grant 91959108. This article was recommended for publication by Associate Editor P. Yuen upon evaluation of the reviewers' comments. *(Corresponding author: Linlin Shen.)*

Changyuan Jiang, Shisong Lin, and Feng Liu are with the Computer Vision Institute, College of Computer Science and Software Engineering, and the Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen 518060, China (e-mail: 1900271054@email.szu.edu.cn; linshisong2018@email.szu.edu.cn; feng.liu@szu.edu.cn).

Wei Chen is with the School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K. (e-mail: wxc795@cs.bham.ac.uk).

Linlin Shen is with the Computer Vision Institute, College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China, and also with the SZU Branch, Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen 518172, China (e-mail: llshen@szu.edu.cn).

Digital Object Identifier 10.1109/TBIOM.2022.3197437

changes [11]. As for deep learning based methods, many researchers consider directly utilizing 2D CNNs for 3D FR task. However, point cloud, the most common data format of 3D face, are unordered and unstructured [12] in contrast to 2D grid data (such as images). Thus, researchers need to convert 3D face to 2D maps. Specifically, researchers convert depth images to point clouds for preprocessing (e.g., cropping and dense alignment) and then estimate 2D geometric maps from the point clouds (also shown in Figure 1) as the input of 2D CNNs for feature extraction. Although deep learning based methods achieve satisfying performance on 3D FR, data conversions process will inevitably lead to geometric information loss due to data resampling.

We argue that a deep learning model that can directly extract features from point clouds of 3D faces is more concise and desirable (shown in lower part of Figure 1). Point clouds, which are independent on reconstruction algorithms, can also provide shape information [12]. To the best of our knowledge, our work is the first deep neural network that extracts features directly from point clouds for 3D FR.

Since different orders of a point cloud can represent a specific 3D shape, a deep learning model should satisfy the property of permutation invariance. Qi *et al.* [13] was the first that proposed an efficient network, PointNet, to directly process point clouds with permutation invariance, which shows satisfactory performance on regular 3D shape classification. To overcome some disadvantages of PointNet, Qi *et al.* [14] proposed a hierarchical network, PointNet++, to further improve the performance on regular 3D shape recognition. Based on PointNet and PointNet++, many works [15], [16], [17] design various convolution networks to learn better local features of regular 3D shape, based point clouds.

Although PointNet and PointNet++ based methods achieve impressive performance on regular 3D object recognition, they obtain unsatisfying result on 3D FR task. The reason is that the geometric differences among different categories of regular 3D shapes are significant, while the geometry structure of 3D faces from different individuals look very similar. Discriminative features, that can distinguish the small geometric differences among 3D faces, shall be explored.

To address issues mentioned above, inspired by contrastive learning [18], [19], we propose a light-weight but effective framework, called PointFace (shown in Figure 2), to directly learn fine-grained representations from 3D point clouds for 3D FR. PointFace is composed of two weight-shared encoders that extract embeddings (feature vectors) from a pair of point cloud faces and two weight-shared fully-connected layer that map the embedding to the identities (ground-truth) during training. As shown in Figure 3, we construct the encoder with *set abstraction module* [14] and fully connected layer. Since PointFace takes point cloud pair as input to evaluate their similarity in the training stage, we present a pair selection strategy to generate positive and negative pairs to boost training and design feature similarity loss to supervise our encoder to learn more discriminative features. Feature similarity loss aims to separate the positive samples from negative samples by a distance margin. Softmax loss evaluating the identity classification accuracy is also designed to

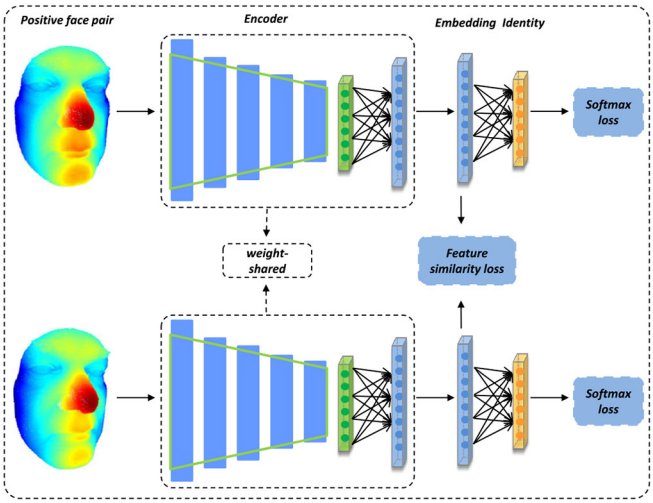


Fig. 2. Overview of our PointFace. PointFace directly encodes a pair of facial point clouds into a pair of embeddings for feature similarity measuring and identity prediction. The parameters of upper and bottom branch of encoder are shared.The fully-connected layers in the end are weight-shared as well. The blue blocks in the encoder denote set abstraction module.

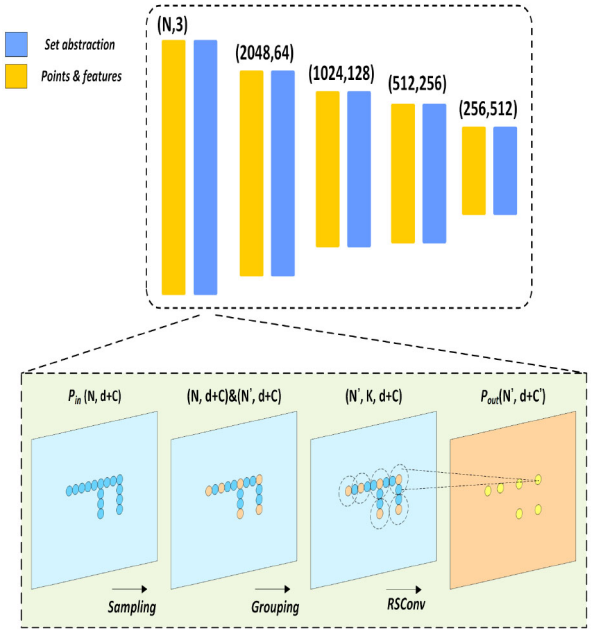


Fig. 3. Details of the set abstraction module. The input of the encoder is point cloud with N points and 3-dimension coordinates. The number in bracket above each layer denotes the number of points and the dimension of corresponding features. The last fully connected layer outputs a 512-dimension embedding. "RSConv" means relation-shape convolution.

supervise the network training. Jointly learning the differences among samples and identity information of each sample itself improves the performance of the encoder. In testing stage, we only need the encoder to extract feature vectors from 3D faces for identification and verification task. Compared with our conference version [20], we perform more comprehensive ablation studies and experiments to demonstrate the effectiveness of different modules and strategies of our PointFace. In addition to identification, we have also added verification test to evaluate the performance of our approach on Lock3DFace [21] and Bosphorus [22]. Moreover, we compare PointFace with the

single encoder (which we use in PointFace) trained with common loss functions used in 2D FR tasks. We have also replaced our encoder with the one used in graph networks [15], [17] to show that our proposed pair selection strategies and loss functions can be generalized to different network architectures. Experimental details will be described in Section IV.

In summary, the main contributions are highlighted as follows:

- We propose a light-weight but effective framework, PointFace, to directly extract features from 3D point cloud for 3D FR. In training stage, PointFace applies two weight-shared encoders to extract features from face pair and then learn the feature differences among samples. In evaluation stage, we utilize the encoder to obtain feature vectors for FR.
- To efficiently measure feature similarity among samples and acquire fine-grained face representations, inspired by contrastive learning, we present a pair selection strategy to boost training and design a feature similarity loss to supervise the training of the encoders.
- Our model achieves state-of-the-art performance on Lock3DFace [21] in terms of rank-one accuracy, which surpasses state-of-the-art Led3D [23] by 2.9%. In addition, we explore cross-quality scenario [23] on Bosphorus [22] and our model surpasses the state-of-the-art Led3D by 1.6%.

II. Related Work

We briefly review some typical and relevant works in the field of 3D face recognition, deep learning based point cloud processing and contrastive learning.

A. 3D Face Recognition

Approaches for 3D FR can be approximately divided into two categories: hand-crafted feature based methods and deep learning based methods. Both of them are evaluated on several publicly available databases, such as FRGC v2 [24] and Bosphorus [22] and achieve impressive performance on such high-quality databases. Recently, Zhang *et al.* [21] propose a low-quality face database and put forward a new challenge: 3D face recognition on low-quality data.

Hand-crafted feature based methods [6], [8], [9], [10] extract features from the key points, curvatures, shape indexes and curves of a 3D face mesh, then identify the face by comparing these features. For instance, Mian *et al.* [9] proposed a repeatable key point detection algorithm for 3D faces and represented the 3D keypoints with 2D Scale Invariant Feature Transform for multimodal face recognition. Gilani *et al.* [10] proposed a key point based dense correspondence model and performed 3D FR by matching the parameters of a 3D deformable model. Samir *et al.* [8] represented surfaces by unions of level curves, called facial curves, and compared shapes of facial curves for face recognition. Nevertheless, hand-crafted feature based methods obtain unsatisfactory performance on noisy data captured by cheap 3D scanner and their generalization ability on large database is limited [11].

TABLE I
Differences Between Our Method and Other 3D FR Methods

Method		Input data
Hand-crafted feature based methods	Mian *et al.* [9]	point cloud & textured map
	Gilani *et al.* [10]	point cloud (need dense correspondence)
	Samir *et al.* [8]	3D mesh
2D CNN based methods	Gilani *et al.* [11]	depth & azimuth & elevation map
	Mu *et al.* [23]	depth map & normal map
Point cloud encoder based	Ours	raw point cloud

As for deep learning based methods, many researchers make an effort to directly utilize 2D CNNs on 3D FR task, since 2D CNNs have made a huge progress [1], [2], [25]. Researchers transform 3D surfaces to 2D geometric maps (such as normal maps) and extract features using 2D CNNs. For example, Kim *et al.* [26] fine-tuned a pretrained VGG-Face network using an augmented dataset with 123,325 depth images. Gilani and Mian [11] expanded training data by generating abundant synthetic 3D faces with rich shape variations and fed three-channel geometry maps to VGG-16 network [27]. Recently Mu *et al.* [23] proposed a light-weight network, Led3D, to perform FR on low-quality data and they trained Led3D with 2D depth images concatenated with normal maps. While [23] achieved state-of-the-art performance on low-quality 3D data, their algorithms convert 3D data to 2D maps as well, which might cause geometric information loss.

Table I summarizes the differences between our method and other traditional 3D FR methods we have mentioned in this section.

B. Deep Learning Based Point Clouds Processing

As the structure of point cloud is irregular and sparse, in the era of deep learning, networks not only need to find a compact representation from point clouds but also need to satisfy the quality of permutation invariance and scale invariance. To apply CNNs to 3D object recognition, some researchers considered voxelizing point clouds and applying 3D CNN to process the volumetric representation [28], [29]. Such kind of methods suffer from the cubic growth of computation complexity and memory costs, hence the resolution of the 3D grid is limited.

As a pioneering work in deep learning based point clouds processing, Qi *et al.* [13] proposed PointNet to directly process point clouds with a symmetric function for permutation invariance. Specifically, it directly takes a point cloud as input and learns point-wise features independently via multi-layer perceptrons (MLPs) and summarizes them via max-pooling (the symmetric function) to obtain global feature. Since features are learned independently for each point in PointNet [13], the local structural information between points is ignored. Therefore, Qi *et al.* [14] proposed a hierarchical network, PointNet++, to capture local geometric information from the neighborhood of each point. Inspired by PointNet and PointNet++, following works [15], [16], [17] design sophisticated convolution kernels on point clouds, resembling 2D convolution on

images, to learn better local features. Liu *et al.* [16] proposed relation-shape convolution among each local region of point clouds to learn shape features. Wang *et al.* [15] proposed *EdgeConv* to incorporate local neighborhood features using a dynamic graph computed from the point cloud. Similar to [15], Zhou *et al.* [17] proposed Adaptive Graph Convolution (AdaptConv) to design adaptive convolution kernels, based on the dynamically learned features and global position for each point.

C. Contrastive Learning

The concept of contrastive learning is inspired by "siamese" neural network [30]. The aim of contrastive learning is to obtain discriminative representations by comparing local or global features between positive and negative samples even when labels of subjects are unknown [18], [19]. Yi *et al.* [31] and Li *et al.* [32] applied contrastive learning to person re-identification tasks. In the field of 2D face verification, works [1], [19], [25] were based on contrastive learning as well.

III. Proposed Approach

In this section, we introduce the overall architecture of our PointFace in Section III-A. Then we describe our encoder by briefly reviewing *set abstraction* [14] and relation-shape convolution [16] in Section III-B. Finally, we detail the proposed pair selection strategy and feature similarity loss in Section III-C.

A. Overall Architecture

As shown in Figure 2, given point cloud data for training, we firstly pair them as input of PointFace via our proposed strategy (detailed in Section III-C). Then, two weight-shared encoders consisting of a stack of set abstraction modules (detailed in Section III-B) encode point cloud pair into two 512-dimension embeddings for identity prediction and feature similarity measurement.

In training phase, we design feature similarity loss to evaluate the similarity between embeddings of two samples so that the encoder is able to distinguish 3D faces from different individuals and compactly cluster features of faces from the same individual. We employ softmax classification loss to supervise the training of encoder as well so that the encoder can learn identity information from each sample itself. Jointly supervised by feature similarity loss and softmax loss, our encoder can obtain more fine-grained representations. In testing phase, we only need embeddings produced by the encoder for identity recognition.

B. Encoder

As shown in Figure 3, the encoder of PointFace uses five set abstraction modules to produce features with dimension of 64, 128, 256, 512, 1024 and one fully connected layer to generate features with dimension of 512.

Set abstraction module [14], a module resembles convolution for 2D images, abstracts a set of points to produce a new set with fewer elements and high-level features. Set abstraction module consists of three key layers: *Sampling layer*, *Grouping layer* and *Feature learning layer*.

Given an input 3D point cloud with N points

$$P_{\text{in}} = \left\{ p_i \in \mathbb{R}^3; i = 1, 2, \ldots, N \right\} \tag{1}$$

and its corresponding features

$$F_{\text{in}} = \left\{ f_{p_i} \in \mathbb{R}^C; i = 1, 2, \ldots, N \right\}, \tag{2}$$

we can use a tensor M_{in} of size $N \times (3 + C)$ to represent the input point cloud:

$$M_{\text{in}} = \left\{ (p_i, f_{p_i}) \in \mathbb{R}^{3+C}; i = 1, 2, \ldots, N \right\}. \tag{3}$$

Sampling layer selects N' points to define the centroids of local regions of input point cloud M_{in}. Let M_{cen} be the output of *Sampling layer*. The process can be formulated as:

$$\begin{aligned} M_{\text{cen}} &= \textit{Sampling layer}(M_{\text{in}}) \\ M_{\text{cen}} &= \left\{ \left(p_j^{\text{cen}}, f_{p_j^{\text{cen}}} \right) \in \mathbb{R}^{3+C}; j = 1, 2, \ldots, N' \right\} \end{aligned} \tag{4}$$

where M_{cen} is a tensor composed of centroid points and their corresponding features. *Grouping layer* constructs local regions around the centroid points by finding K neighbors of each centroid point within a pherical radius r. The process can be formulated as:

$$M_{\text{group}} = \textit{Grouping layer}(M_{\text{in}}, M_{\text{cen}}) \tag{5}$$

where M_{group} is the tensor of size $N' \times K \times (3 + C)$. *Feature learning layer* encodes and aggregates each local region pattern into a more condensed representation. The process can be formulated as:

$$\begin{aligned} M_{\text{out}} &= \textit{Feature learning layer}(M_{\text{group}}) \\ M_{\text{out}} &= \left\{ \left(p_j^{\text{cen}}, f'_{p_j^{\text{cen}}} \right) \in \mathbb{R}^{3+C'}; j = 1, 2, \ldots, N' \right\}. \end{aligned} \tag{6}$$

See Fig. 3 for better understanding. Here, we utilize relation-shape convolution [16] to act as *Feature learning layer* for better local feature learning.

Relation-shape convolution [16] aims to learn weights for each point in a local region around a certain centroid and aggregate them to obtain condensed representation for the local region. The weights are learned by mapping a low-level relation prior (such as Euclidean distance and relative position) to high-level relation vector using MLPs.

Specifically, let p_j^{cen} be one of the centroid points sampled by *Sampling layer*, $\mathcal{N}(p_j^{\text{cen}})$ be its neighbor area within a spherical radius r, relation-shape convolution can be formulated as:

$$f'_{p_j^{\text{cen}}} = \mathcal{A}\left(\left\{ \mathcal{E}(f_{p_k}), \forall p_k \in \mathcal{N}\left(p_j^{\text{cen}} \right) \right\} \right), \tag{7}$$

where $\mathcal{A}$ is a symmetric aggregation function (we use max-pooling here) that helps the convolution achieve permutation invariance and aggregate local features, $\mathcal{E}$ is a function that learns features for all points in $\mathcal{N}(p_j^{\text{cen}})$. As defined in [16], $\mathcal{E}$ is formulated as:

$$\mathcal{E}(f_{p_k}) = \mathcal{M}\left(h\left(p_j^{\text{cen}}, p_k \right) \right) \otimes f_{p_k}, \tag{8}$$

where $\otimes$ denotes Hadamard product, $\mathcal{M}$ is multi-layer perceptrons (MLPs) that learns convolution kernels (or weights) from low-level relation prior vector $h(p_j^{\text{cen}}, p_k)$ and $h(p_j^{\text{cen}}, p_k)$ is a prior vector that can be defined flexibly. For instance, we define $h(p_j^{\text{cen}}, p_k)$ as 10-dim vector $(p_j^{\text{cen}}, p_k, p_j^{\text{cen}} - p_k, ||p_j^{\text{cen}} - p_k||_2)$ for our tasks. As a consequence, Eq. (7) becomes:

$$f'_{p_j^{\text{cen}}} = \mathcal{A}\Big(\Big\{\mathcal{M}\Big(h\Big(p_j^{\text{cen}}, p_k\Big)\Big) \otimes f_{p_k}, \forall p_k \in \mathcal{N}\Big(p_j^{\text{cen}}\Big)\Big\}\Big). \quad (9)$$

Note that, *feature learning layer* based on relation-shape convolution is different from MLPs. If MLPs are simply used as *feature learning layer*, the process can be formulated as:

$$f'_{p_j^{\text{cen}}} = \mathcal{A}\Big(MLPS\big(f_{p_k}\big), \forall p_k \in \mathcal{N}\Big(p_j^{\text{cen}}\Big)\Big). \quad (10)$$

The main difference between Eq. (9) and Eq. (10) is that Eq. (10) does not introduce low-level relation prior vector $h(p_j^{\text{cen}}, p_k)$ which offers geometric priors for $\mathcal{M}$ to learn convolution weights. Besides, we have conducted ablation study (in Section IV-F) on the *feature learning layer* to prove the effectiveness of relation-shape convolution.

C. Pair Selection and Feature Similarity Loss

Pair Selection: To fully explore the advantages of contrastive learning and boost training, we present a novel pair selection strategy to generate positive pairs and negative pairs. Specifically, for each sample called "anchor" in the training data, we randomly select one positive sample from the training data and pair them as input of the network. During the process, we ensure that each positive counterpart is selected only once and every pair is unique. The result is that all samples belonging to an individual are linked together with minimum number of pairs. After the encoder of PointFace encodes a mini-batch of anchor-positive pairs into a mini-batch of paired embeddings, for each anchor, we pick the hardest negative sample (whose embedding have the closest distance from anchor) from counterpart of other anchors within the mini-batch to group anchor-negative pair. Triplets that contains anchors and their corresponding positive and negative samples within the mini-batch are then used to compute feature similarity loss. In practice, we use all anchor-positive pairs (if they exist), instead of only selecting the hardest positive, to boost convergence. Detailed implementation is shown in Algorithm 1.

Analysis of our Strategy: Generally speaking, in terms of generating pair data, the performance of model can be influenced by two factors [1], [19]: (1) ratio of positive and negative pairs, and (2) pair diversity (anchor shall compare with as many samples as possible).

It is important to balance the positive and negative pairs during training. Empirically, too much negative samples for each anchor will lead to poor performance of model. In our strategy, offline positive-anchor pair grouping guarantees at least one positive sample for each anchor and hardest negative mining ensures that only one hardest negative sample is compared with each anchor in the mini-batch. The number of positive and negative pairs can thus be balanced. In addition, we compare hardest negative mining with semi-hard negative mining [1], and find that hardest negative mining enable our network to perform better on 3D point clouds. i.e., we do not need to pick more negative samples for each anchor in our task. In practice, we find that using all positive samples within a mini-batch can accelerate convergence without degrading performance.

Algorithm 1 PointFace Pipeline

Require: B: batch size; N_s: number of subjects; $\mathcal{F}$: encoder; $\{SL_n\}$: subject lists ($n = 1, 2, ..., N_s$);
Ensure: optimal encoder $\mathcal{F}$

Initialize empty set AP
for $n = 1$ to N_s **do**
  let len_n be the length of SL_n
  for $i = 1$ to len_n **do**
    let P_i be the point cloud of $SL_n[i]$
    add $(P_i, P_{(i)mod(len_n)+1})$ to AP
  end for
end for
repeat
  get mini-batch $\{(P_i, P_i^+)\}_{i=1}^B$, $(P_i, P_i^+) \in AP$
  for each (P_i, P_i^+) in $\{(P_i, P_i^+)\}_{i=1}^B$ **do**
    $P_i = data\_augment(P_i)$ ▷ data augmentation
    $P_i^+ = data\_augment(P_i^+)$
    $emb_i = \mathcal{F}(P_i)$
    $emb_i^+ = \mathcal{F}(P_i^+)$
  end for
  Initialize empty set T
  for each emb_i in $\{(emb_i, emb_i^+)\}_{i=1}^B$ **do**
    let emb_i^- be the hardest negative embedding found in $\{emb_j^+\}_{j \neq i}^B$
    add $(emb_i, emb_i^+, emb_i^-)$ to T ▷ we use all positive embeddings in practice
  end for
  compute Eq. (12) using T and softmax loss
  update parameters of encoder $\mathcal{F}$
until encoder $\mathcal{F}$ is converged

Our strategy maintains pair diversity as well. All positive samples are linked together with minimum number of pairs and hardest negative mining ensures that each anchor will be compared with every positive sample and hard negative sample directly or indirectly as training proceeds.

Feature Similarity Loss: For the purpose of separating positive pairs from negative samples by a margin, inspired by triplet loss [1] and contrastive loss [19], we design feature similarity loss to supervise the training of our encoder. Note that we aim to minimize the similarity among samples from the same subjects and maximize the discrepancy among different subjects simultaneously in high-dimension feature space. Specifically, given N_s triplets with emb_i, emb_i^+ and emb_i^- ($i = 1, 2, \ldots, N_s$) as L_2 normalized embeddings of an anchor, positive sample, and negative sample respectively, our feature similarity loss can be described as:

$$L_{sim} = \sum_{i=1}^{N_s} \big[\mathcal{D}\big(emb_i, emb_i^+\big) + m - \mathcal{D}\big(emb_i, emb_i^-\big)\big], \quad (11)$$

where m denotes the margin and $\mathcal{D}(\bullet, \bullet)$ denotes function that calculates distance between two vectors. Different from triplet

loss in [1], we utilize cosine distance, instead of L_2-distance, to evaluate similarity between two vectors. The reasons are: (1) good embeddings of different samples belonging to the same subject should have the same angle in high-dimension feature space [33] and cosine distance as loss function can guide the encoder to achieve this goal, (2) L_2-distance attempts to project different samples from a specific subject to a point in high-dimension feature space, which can not guide network to reach optimal status for our tasks (as our experiments shows). Due to monotonic decrease of cosine function (when angle is in range $[0,\pi/2]$) , we reformulate L_{sim} in Eq. (11) as:

$$L_{sim} = \sum_{i=1}^{N_s} \left[1 - \mathcal{D}_{cos}\left(emb_i, emb_i^+\right) + \mathcal{D}_{cos}\left(emb_i, emb_i^-\right) - m\right], \tag{12}$$

where $\mathcal{D}_{cos}(\bullet, \bullet)$ denotes cosine distance between two feature vectors.

Combining feature similarity loss and softmax classification loss $L_{softmax}$, the total loss for PointFace can be described as:

$$L_{total} = L_{softmax} + \lambda L_{sim}, \tag{13}$$

where λ is a constant to balance each item.

IV. Experiments

In this section, we firstly introduce databases we use in Section IV-A and the implementation of PointFace in Section IV-B. Then, we describe the protocols of 3D FR on Lock3DFace [21] and Bosphorus [22] in Section IV-C. We show results of 3D FR experiments on these two databases in Section IV-D. Extensive ablation studies and analytical experiments are carried out in Section IV-F. In Section IV-F2, we compare PointFace with the single encoder (which we use in PointFace) trained with loss functions (such as CosFace [33] and ArcFace [34]) common in 2D FR tasks. In Section IV-F3, we show the generalization performance of our proposed pair selection strategy and loss functions by replacing the encoder with the network proposed in [17] and the one proposed in [15]. In Section IV-F4, we explore the robustness of our apporach against Gaussian noise, random cropping and random rotation.

A. Datasets

Lock3DFace [21] is a comprehensive database consisting of low-quality 3D faces collected by Kinect V2, which includes 5,671 RGB-D video clips of 509 individuals. The dataset is divided into five subsets covering variations in expression (FE), neutral face (NU) , occlusion (OC), pose (PS) and time lapse (TM). Samples of facial scans in point cloud format are shown in the first row of Figure 4. We can see that there are lots of noises in the face data from Lock3DFace and some facial regions are blurry. Though the dataset contains RGB information, we only use depth information for all models in our experiments.

FRGC v2 [24] consists of 4,007 high-quality 3D faces of 466 individuals with expression variations. Facial data are collected by a high-precision 3D Laser scanner. Some facial scans in point cloud format are visualized in the second row of Figure 4. We can see that face data from FRGC v2 is clean and facial features are distinctive.

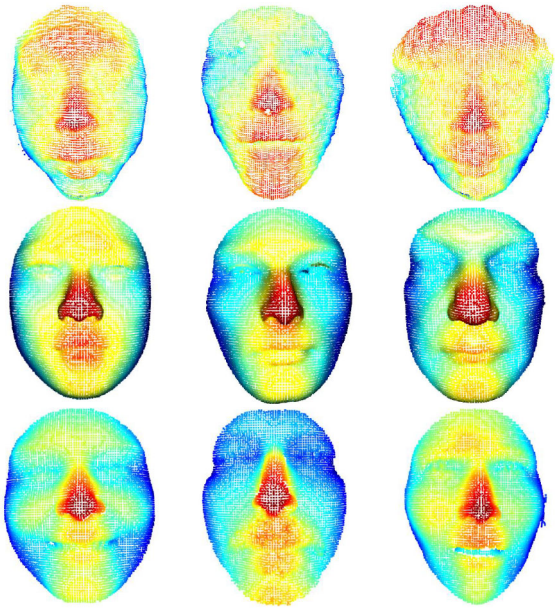


Fig. 4. Facial scans of three datasets. First row: Lock3DFace. Second row: FRGC v2. Third row: Bosphorus.

Bosphorus [22] is a database consisting of 4,666 high-quality faces of 105 individual presenting neutral face (299 facial scans) and variations in expression (2621 facial scans), occlusion (381 facial scans), and pose (1365 facial scans). The third row of Figure 4 shows some of the examples.

B. Implementation Details

We train our PointFace in an end-to-end fashion. Our network is implemented using Pytorch. We employ Adam [35] optimization for training, with a mini-batch size of 2×32. The learning rate begins with 0.001 and decays with a rate of 0.7 every 20 epochs. Dropout rate (for fully connected layer) is set to 0.4. For feature similarity loss in Eq. (12), we set m to 0.35 and 0.4 for Lock3DFace [21] and Bosphorus [22] respectively, and in Eq. (13), we empirically set λ to 1.

C. Protocols and Training Settings

1) Lock3DFace: Protocol: To perform ablation studies, identification and verification on Lock3DFace, we follow the same protocol proposed in [23], [36] for training and testing. Firstly, we randomly select 340 subjects for training and the rest (169) are used for testing. In training set, we sample six frames from each video with an equal interval. In testing set, we also extract six frames for each subject and the first frame from each face video with neutral expression is taken as gallery samples (resulting in 169 gallery samples) and the remaining frames are used as probes. Secondly, for both training set and testing set, we preprocess all samples by nose-tip based alignment for non-facial area removal [23] and face cropping. After preprocessing, all data are in point cloud format, which can be directly fed to our PointFace. As for training of the 2D CNN

counterparts, we follow [23] to obtain the refined depth maps from the preprocessed point clouds. In evaluation phase, we extract embeddings of probes and galleries from the encoder of PointFace and other 2D CNNs. Then, we match the feature of each probe with all identities in the gallery. The identity of the probe is assigned as that of gallery who has the minimum cosine distance.

Training settings for pointFace: During training, we sample 5,000 points using *farthest point sampling* from each training sample and normalize them to a unit sphere at each epoch. Moreover, during training, we randomly keep the original data or augment the data with one of following approaches: random anisotropic scaling in the range $[-0.66, 1.5]$, random translation in the range $[-0.2, 0.2]$ and random rotation in the range $[-90°, 90°]$ on yaw-direction and $[-30°, 30°]$ on pitch-direction. We train PointFace from scratch. Two types of input data: coordinates (XYZ) and coordinates with normal (XYZ&Normal) of point clouds are tested. Besides, in evaluation phase, we uniformly sample 5,000 points from each testing sample and normalize them to a unit sphere before feeding to the encoder of PointFace.

Training settings for 2D CNN counterparts: For comparison, we train several representative 2D CNNs [27], [37], [38], [39] using depth images acquired by the preprocessing pipeline [23] following the same protocol. Besides, we use FRGC v2 [24] and Bosphorus [22] to pre-train these 2D CNNs to obtain better performance. Furthermore, we use "random crop" technique for data augmentation in training phase.

2) Bosphorus: Since low-quality and cheap sensors are more affordable than high-precision sensors, a common setting in the real world is that high-quality data and low-quality data are used as gallery samples and probe samples [23] respectively. As the quality of probes and galleries is significantly different, we call such testing as cross-quality evaluation.

To see whether our model has good performance on cross-quality evaluation, we utilize FRGC v2 as training set and evaluates PointFace on Bosphorus. Thanks to Mu *et al.* [23], we acquire the enriched FRGC v2 dataset whose size is increased by virtual ID generation method in [11] for model training. In the training stage, we generate low-quality data from high-quality FRGC v2 dataset by adding Gaussian noise $N(0, 16)$ to the depth values (values along axis Z) of point cloud data, following what presented in [23]. The training set contains both high-quality and low-quality point clouds of 1,000 identities. For testing set, we use the face with neutral expression of each individual in Bosphorus as gallery (resulting in 105 galleries) and the remaining faces are converted into low-quality data as probes by the same operation. In addition to the setting of HL (high-quality galleries and low-quality probes), we have evaluated our model on the setting of LL where both galleries and probes are converted into low-quality as well. Note that we preprocess the point cloud data for both datasets by cropping the facial area, downsampling and estimating normal as well. In testing phase, we extract embeddings of samples from the encoder and the identities of probes are decided based on the minimum cosine distance between embeddings of probes and galleries.

TABLE II
RANK-ONE RECOGNITION RATE (%) ON LOCK3DFACE

Method	Input	Lock3DFace					
		FE	NU	OC	PS	TM	Total
VGG-16 [27]	Depth	94.76	99.57	44.68	49.21	34.50	70.58
ResNet-34 [37]	Depth	96.09	99.29	54.91	61.39	45.00	76.56
Inception-v3 [38]	Depth	93.56	98.97	56.98	54.14	42.17	74.44
MobileNet-v2 [39]	Depth	95.74	98.91	61.44	69.92	43.00	79.49
Led3D [23]	Depth	97.62	99.62	68.93	64.81	64.97	81.02
	Depth&Normal	98.17	99.62	78.10	70.38	65.28	84.22
PointFace (Ours)	XYZ	97.93	99.35	77.06	72.03	66.33	84.78
	XYZ&Normal	**98.52**	99.46	**80.67**	**73.69**	**67.00**	**87.18**

D. FR Results on Lock3DFace

1) Identification (Results): Table II reports the rank-one accuracies of PointFace and other typical 2D CNNs on low-quality Lock3DFace dataset. Note that the rank-one accuracy of state-of-the-art Led3D is reported in [23]. According to Table II, our PointFace outperforms the Led3D by 3.76% in terms of rank-one recognition accuracy when depth information is used as input (PointFace is trained only with XYZ of point clouds and Led3D is trained with depth images). Besides, we can see that our model reach relatively higher rank-one accuracies on subset OC (77.06%) and PS (72.03%) than Led3D. PointFace trained with XYZ from scratch surpasses all typical 2D CNNs on subset FE, OC, PS and TM. Among the 2D CNNs, MobileNet-v2 achieves the best rank-one accuracy, i.e., 79.49%, which is about 5% lower than that of PointFace. It illustrates that PointFace can acquire more geometric information robust to occlusions and pose changes. Surprisingly, we also see that PointFace trained with only XYZ has better performance (84.78%) than Led3D trained with both depth images and normal maps (84.22%).

We now estimate normal for each point in the point cloud data and concatenate it with XYZ as an extra input, i.e., now the input becomes six channels. As extra information is introduced, the performance of both Led3D and our PointFace improves. For example, the total accuracy of Led3D and PointFace increase from 81.02% to 84.22% and 84.78% to 87.18%, respectively. Again, the total accuracy of our approach (87.18%) is about 2.96% higher than that of Led3D, when normal is included as the input. In particular, PointFace achieves much higher accuracies on the subsets of OC (80.67%) and PS (73.69%).

t-SNE visualization: We now use t-SNE to visualize the embeddings of faces learned by different networks, i.e., PointFace, MobileNet-v2 and Led3D in Figure 5. We randomly select six subjects in OC and PS (each subject has around 12 samples) respectively and obtain their embeddings using the three networks. One can observe from Figure 5(a) that most of the samples of the same subjects are closely clustered and the samples from different subjects are distinguishable, i.e., the embeddings learned by our PointFace can well discriminate different subjects. In contrast, the samples of subject marked with red and those marked with purple in (b) are widely separated. Similar samples can also be found in (c) for the subject marked with blue and black in the first column and that marked with red and purple in the second column.

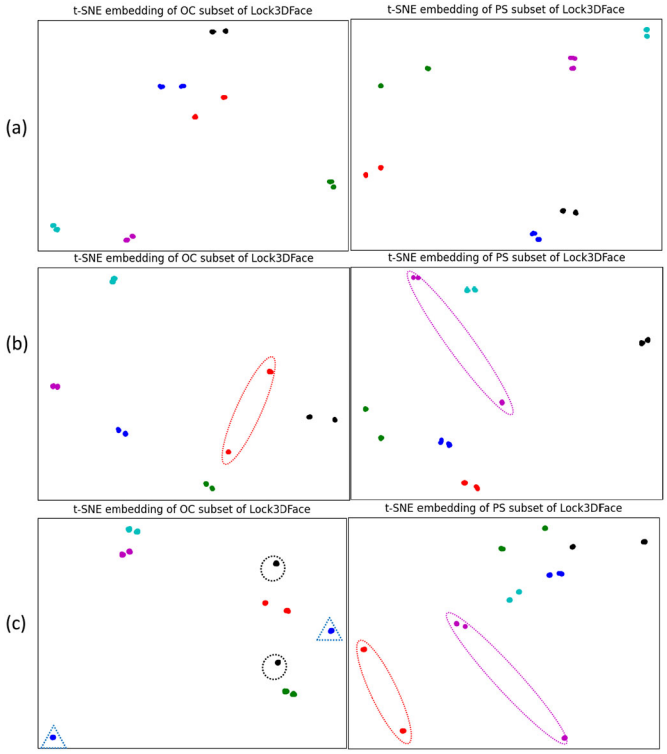


Fig. 5. t-SNE visualization of the features extracted by PointFace(a), MobileNet-v2(b) and Led3D(c). The first column and second column show the faces in OC and PS subsets, respectively. The samples of different subjects are represented with dots of different colors.

The visualization clearly justify that the embeddings extracted by our PointFace can cluster much better than MobileNet-v2 and Led3D.

2) Verification: We now test the verification performance of our approach using Lock3DFace dataset. As there are 169 subjects in the testing set, and each subject consists of one gallery and around 46-60 probe samples, we can generate 8,978 client accesses and about 1,508,000 impostor accesses by paring the probes of a subject with all of the gallery samples. The access is either accepted, or denied, based on the similarity of the pair of 3D faces and a preset threshold. When different values of threshold are set, various FAR (False Accept Rate) and FRR (False Rejection Rate) can be calculated. Figure 6 shows the variation of verification rate (1-FRR) when FAR change from 0.02% to 0.4%, for different networks. In this testing, we implement the Led3D using Pytorch and train it with the same protocol and training setting of 2D CNNs described in Section IV-C. We choose the models with the highest rank-one accuracy to evaluate their verification performance. Note that the input are depth images for 2D CNNs and coordinates for PointFace. One can observe from the figure that the verification rate of PointFace ranks the highest, for all of different FAR. We also list the verification rates of different networks when FAR=0.1% for all different subsets in Table III. One can observe that the overall accuracy of PointFace is about 4.3% higher than that of the best 2D CNNs, MobileNet-v2.

TABLE III
VERIFICATION RATE (%) ON LOCK3DFACE (FAR = 0.001)

Method	Input	Lock3DFace					
		FE	NU	OC	PS	TM	Total
VGG-16	Depth	89.34	98.97	34.24	28.85	15.17	60.64
ResNet-34	Depth	93.63	98.97	39.21	47.09	28.17	68.30
Inception-v3	Depth	78.06	97.51	34.28	23.32	12.50	56.05
MobileNet-v2	Depth	93.56	98.53	49.20	53.99	36.33	72.50
Led3D	Depth	90.79	97.78	45.01	25.81	28.83	63.41
PointFace (Ours)	XYZ	**94.65**	**99.02**	**62.15**	**57.20**	**45.83**	**76.82**

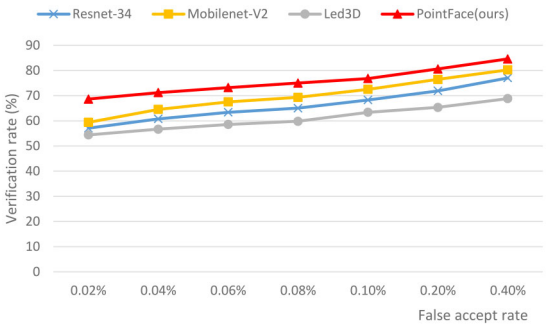


Fig. 6. Verification results for ResNet-34, MobileNet-v2, Led3D and PointFace when setting different FAR.

TABLE IV
RANK-ONE ACCURACY (%) OF CROSS-CATEGORY EVALUATION ON LOCK3DFACE

Test subset	Method	Rank-1 accuracy
OC	PointFace (Ours)	**52.08**
	Led3D [23]	42.82
PS	PointFace (Ours)	**36.87**
	Led3D [23]	15.38
FE	PointFace (Ours)	**94.18**
	Led3D [23]	92.70

3) Cross-Category Evaluation: Here we explore a more challenging scenario, named "cross-category evaluation". We remove the samples of each category from the training set and evaluate the trained model using samples of this category in the test set. The split of the training set and testing set of Lock3DFace is exactly the same as that described in Section IV-C1. For example, if we want to evaluate the cross-category performance on subset OC, we remove samples of OC in the training set and then evaluate the performance of trained model on OC samples of the test set. Note that we keep the same training setting in Section IV-C1 and do not perform extra data augmentation in the training process. The results are shown in Table IV. Note that we implement the Led3D using Pytorch and train it with the same training setting of 2D CNNs described in Section IV-C.

One can observe that the accuracy of both PointFace and Led3D on test subset OC and PS decreases obviously. However, the performance of PointFace on OC (52.08%) and PS (36.97%) is overwhelmingly better than the performance of Led3D on those subsets (42.82% on OC and 15.38% on PS). One can also observe a slight decline in the performance of

TABLE V
RANK-ONE RECOGNITION RATE (%) OF CROSS-QUALITY EVALUATION ON BOSPHORUS (HL: HIGH-QUALITY IN GALLERY AND LOW-QUALITY IN PROBE; LL: LOW-QUALITY IN BOTH GALLERY AND PROBE)

Method	HL	LL
Inception-v2 [40]	78.56	77.23
Led3D [23]	91.27	90.70
PointFace (Ours)	**92.96**	**91.86**

TABLE VI
VERIFICATION RATE (%) AT 0.1% FAR OF CROSS-QUALITY EVALUATION ON BOSPHORUS IN HL SETTING (FE: FACIAL EXPRESSION, OC: OCCLUSION, PS: POSE, ALL: ALL FACIAL SCANS)

Method	N vs. FE	N vs. OC	N vs. PS	N vs. All
Inception-v2 [40]	80.53	66.12	35.69	69.30
Led3D [23]	81.65	56.36	43.07	70.68
PointFace (Ours)	**85.88**	**70.60**	**75.82**	**82.28**

both methods on FE and the accuracy of PointFace (94.18%) is still higher than Led3D (92.70%).

E. Cross-Quality Evaluation on Bosphorus

1) Identification: Table V shows the results of cross-quality 3D FR. Note that the results of Inception-v2 [40] and Led3D [23] are reported in [23]. Since [23] rotates the 3D faces to generate more depth images with different poses, we perform similar data augmentation. Specifically, we rotate point cloud faces in FRGC v2 and utilize *hidden point removal* proposed in [41] to generate the same kind of occlusions caused by pose changes for data augmentation in training phase. Our PointFace achieves 91.86% rank-one accuracy in LL setting and 92.96% in HL setting, which surpass the performance of Led3D in both scenarios. The results suggest that PointFace has relatively good generalization ability and is robust to data quality changes.

2) Verification: We now test the verification performance of our approach using Bosphorus in HL setting. As there are 105 subjects in the dataset, and each subject consists of one gallery and around 30-50 probe samples, we can generate 4,560 client accesses and about 474,350 impostor accesses by paring the probes of a subject with all of the gallery samples. The access is either accepted, or denied, based on the similarity of the pair of embeddings of 3D faces and a preset threshold. Furthermore, we exclude the neutral facial scans in the probes and divided the rest of probes into 3 challenging categories: faces with expressions (FE), faces with occlusions (OC) and faces with poses (PS). Note that in this testing, we implement the Led3D and Inception-v2 using Pytorch and train it with the same training setting of 2D CNNs described in Section IV-C1. We present the results in Table VI. One can observe that PointFace achieves the best verification rate in every situation.

F. Analysis of PointFace

In this section, all analytical experiments are conducted on Lock3DFace using protocol and training setting described in Section IV-C. Note that we only use coordinates of point clouds as input.

TABLE VII
ABLATION STUDIES OF POINTFACE IN TERMS OF RANK-ONE RECOGNITION RATE (%) ON LOCK3DFACE (FSLOSS: FEATURE SIMILARITY LOSS; SLOSS: SOFTMAX LOSS; RSCONV: RELATION-SHAPE CONVOLUTION)

Model	RSConv	Sloss	FSloss	Lock3DFace					
				FE	NU	OC	PS	TM	Total
I		✓		94.45	98.80	64.45	48.40	46.00	75.31
II	✓	✓		96.41	99.13	72.59	59.38	57.17	80.80
III		✓	✓	96.17	98.43	70.93	61.86	47.17	80.10
IV	✓		✓	92.35	97.29	63.05	53.26	23.67	73.66
V	✓	✓	✓	**97.93**	**99.35**	**77.06**	**72.03**	**66.33**	**84.78**

1) Ablation Studies (Ablation Study on Architecture): Table VII summarizes the results of PointFace with different network structures. The baseline (model I) is trained without relation-shape convolution, but with a two-layer MLP as function $\mathcal{E}$ in Eq. (7). Besides, the baseline, similar to PointNet++ [14], is just a single encoder trained with softmax classification loss. Model II is also a single encoder using relation-shape convolution as function $\mathcal{E}$ in Eq. (7). Model III replaces the relation-shape convolution in PointFace with a two-layer MLP. Model IV is PointFace trained without softmax classification loss.

The baseline (model I) only gets rank-one recognition accuracy of 75.31%. Compared with baseline, relation-shape convolution improve the accuracy of model II to 80.80%. The improvement demonstrates that local feature learner based on convolution obtains better local region representations than simple MLPs. Employing two weight-shared encoder trained with feature similarity loss (model III) improves the rank-one score to 80.10% (4.79% higher than the baseline), i.e., our proposed architecture has stronger capability to distinguish point cloud faces from different subjects. Although PointFace trained without softmax loss (model IV) doesn't perform well in challenging subsets (OC, PS and TM), it can still learn features from neutral faces. Moreover, our PointFace (model V) outperforms model II and model IV by 3.97% and 11.12% respectively, proving that the two loss functions jointly help our encoder to learn discriminative feature.

Ablation study on different pair selection strategies: We test four different pair selection strategies in this evaluation. In strategy I (as baseline), we randomly generate positive and negative pairs for each anchor at each epoch before training and hardest negative mining is not applied. The only difference between strategy II and ours is that we generate all possible positive pairs. Strategy III replaces the hardest negative mining in our strategy with semi-hard negative mining [1]. Note that for each strategy we ensure that the number of batches is the same at each epoch. The rank-one accuracy and the approximate number of epochs for our PointFace to converge are shown in Table VIII. As shown in the table, our proposed strategy achieves the highest accuracy, i.e., 84.78%, among all strategies. The accuracy of our strategy is significantly higher

TABLE VIII
PERFORMANCE OF DIFFERENT PAIR SELECTION STRATEGIES

Strategy	rank-1 accuracy(%)	epochs
I	81.93	≈ 800
II	84.58	≈ 500
III	82.55	≈ 230
Ours	**84.78**	≈ 250

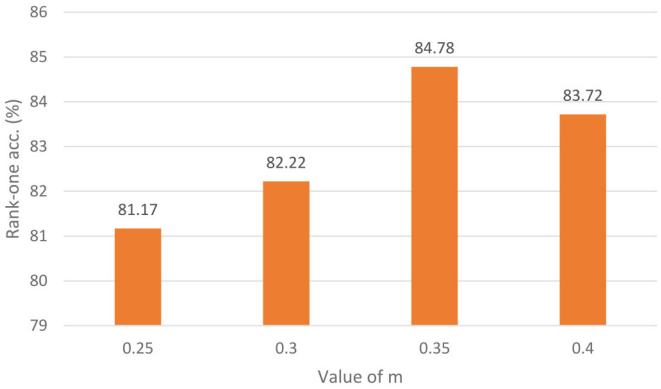


Fig. 7. Ablation study of hyper-parameter m in Eq. (12).

than that of strategy I (+2.85%) and strategy III (+2.23%). For number of epochs required for training, our strategy is slightly higher than strategy III and significantly less than both strategy I and II.

Ablation study on hyper-parameter in feature similarity loss: In this section, we investigate the effect of hyper-parameter m in Eq. (12). We change m from 0.25 to 0.40 and show the FR results in terms of rank-one accuracy in Fig. 7. As shown in the figure, when m is too small, i.e., 0.25, the inter-class discrepancy is too small and the embeddings learned by the model are not discriminative enough, which results in a decrease in recognition performance (81.17%). When the value of m increases, the FR performance improves as well and gets saturated at 0.35.

2) Comparison With Margin-Based Softmax Loss Functions and L_2 Distance: For most of the 2D FR tasks, many works such as CosFace [33] and ArcFace [34] design sophisticated loss functions to guide CNNs to obtain fine-grained features, which surpass contrastive learning based methods [1], [25] in many 2D face datasets. However, the situation is different in point cloud based FR.

We train the single encoder of PointFace with the losses proposed in CosFace [33] and ArcFace [34] separately for comparison on Lock3DFace database. The margins of both ArcFace and CosFace are optimized for the best performance in this testing. As shown in Table IX, the accuracies of both softmax based approaches are significantly lower than that of our similarity loss. Both of ours and margin-based softmax loss aim to maximize the discrepancy among different subjects, but it seems that our loss can guide the encoder to find more discriminative embedding.

We also test the performance of different distance measurements, i.e., L_2 and cosine, for our similarity loss. As shown in the Table IX, cosine distance achieves 2% higher accuracy than L_2 distance. i.e., cosine distance is more suitable for point cloud based face recognition.

TABLE IX
COMPARISON OF DIFFERENT LOSS FUNCTIONS IN TERMS OF RANK-ONE ACCURACY (%)

		FE	NU	OC	PS	TM	Total
Margin-based Softmax	CosFace [33]	90.63	97.30	62.05	46.18	37.67	72.35
	ArcFace [34]	93.83	98.60	59.94	53.11	40.83	74.78
Our similarity loss	L_2	97.58	98.86	74.49	66.81	64.83	82.75
	Cosine	**97.93**	**99.35**	**77.06**	**72.03**	**66.33**	**84.78**

TABLE X
PERFORMANCE WHEN APPLYING DIFFERENT ENCODER IN TERMS OF RANK-ONE ACCURACY (%)

Encoder	FE	NU	OC	PS	TM	Total
DGCNN [15]	94.11	97.67	71.74	42.13	42.33	74.97
DGCNN †	**95.04**	**97.93**	**73.39**	**51.20**	42.33	**77.65**
AdaptConv [17]	94.69	98.42	72.89	43.24	42.83	75.82
AdaptConv †	**96.02**	**98.64**	**74.75**	**55.69**	**43.50**	**79.37**

3) Generalization Performance of Our Pair Selection Strategy and Loss Functions: As mentioned in the survey of 3D point cloud processing network [42], convolutional networks such as PointCNN [43] and RSCNN [16], and graph networks such as DGCNN [15] and Adaptive Graph Convolution [17](denoted as AdaptConv), are the two main categories of deep networks available for point cloud processing. The encoder in our PointFace belongs to convolutional networks and involves the process of subsampling, grouping and convolution. Different with convolutional network, Graph networks consider a point cloud as a graph. Points in the point cloud are considered as the vertices of the graph and the edges of the graph are generated based on the neighbor area of each point. Graph networks usually do not need to subsample points and the graph are constructed in feature space, instead of Euclidean space [12], [15].

To justify the generalization performance of our pair selection strategy and loss functions, we now try to replace the encoder of our PointFace with the one of DGCNN [15] and AdaptConv [17] to test the 3D FR performance. DGCNN constructs the graph in feature space and dynamically updates the graph. At each layer of DGCNN, the features of each points are learned by the EdgeConv [15] operators. Based on DGCNN, a new graph convolution operator is proposed in AdaptConv [17]. The modified PointFace are then trained with the same pair selection strategy and loss functions used in previous FR experiments (denoted with "†"). For baselines, we train the single encoder of DGCNN and AdaptConv only using softmax loss (like the training of model I in Section IV-F1). The results of rank-one accuracy are shown in Table X. One can observe that the overall accuracies of the two graph based encoders trained using our proposed strategy and loss functions are significantly higher than that of baseline DGCNN (+2.68%) and AdaptConv (+3.55%). The results demonstrate that our training strategy and loss functions can help graph networks achieve better performance in 3D FR as well.

TABLE XI
ROBUSTNESS AGAIN NOISE, ROTATION AND CROPPING

Operations	original data	Gaussian noise	cropping	rotation
Total rank-1 acc. (%)	84.78	80.57	80.46	80.69

TABLE XII
TIME AND SPACE COMPLEXITY IN TERMS OF #PARAMS (NUMBER OF TRAINABLE PARAMETERS), #MULT-ADDS (NUMBER OF MULTIPLY-ADD OPERATIONS PER SAMPLE) AND #INFER-TIME (INFERENCE TIME PER SAMPLE)

Model	#Params	#Mult-Adds	#Infer-time (ms)
VGG-16 [27]	138.35M	15.47G	11.56
ResNet-34 [37]	21.80M	3.66G	8.59
Inception-v3 [38]	27.16M	5.72G	25.27
MobileNet-v2 [39]	3.5M	**0.3G**	8.58
Led3D [23]	5.5M	0.5G	**6.67**
PointFace	**1.80M**	1.25G	8.48

4) Robustness Against Noise, Rotation and Cropping: Now we investigate the robustness of our approach against noise, rotation and cropping, using Lock3DFace. Gaussian noises, random rotation and cropping are respectively applied to the face data in probe, and matched against the subjects in gallery for evaluation. A Gaussian noise with 0 mean and 0.04 standard deviation is randomly added to z-axis of normalized faces. We perform random rotations and random cropping around nose tip of 3D faces on the probes, and then identify the noisy/rotated/cropped face against subjects in the gallery. The rotation angle along yaw-direction and pitch-direction ranges from $-60°$ to $60°$. The radius of the cropping on the normalized faces ranges from 0.9 to 0.95. The rank-one accuracies for the faces in probe are shown in Table XI, together with that of original data. One can observe that there is about a 4% decrease of accuracy when noises and operations like cropping and rotation are applied to the probe. It also seems that our approach is more robust against noise and rotation, than random cropping.

G. Model Complexity

Table XII summarizes the memory costs (number of parameters in the network), FLOPs (multiply-add operations/sample) and inference time per sample of PointFace, in comparison with other representative 2D CNN models. Note that PointFace takes 5000 points as input, and 2D CNNs take 2D depth image as input. The last fully connected layer of all models output a 1000-dimension vector. Note that all of the models were tested using a PC equipped with an Intel Xeon CPU of 2.40GHZ and a NVIDIA TITAN X GPU. Among them, PointFace requires the least number of parameters (1.8M). The comparison shows that our approach requires more #Mult-Adds than MobileNet-v2 [39] and Led3D [23], and slightly longer infer-time than Led3D. However, our approach is much more efficient than VGG-16 [27], ResNet-34 [37] and Inception-v3 [38]. Although dynamic kernels in set abstraction module lead to more #Mult-Adds than MobileNet-v2 and Led3D, PointFace achieve higher rank-one accuracy.

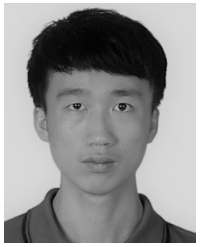
V. CONCLUSION

In this paper, we propose PointFace to directly process point cloud faces for 3D FR. In this framework, two weight-shared encoders are designed to directly extract features from a pair of 3D faces. Together with the identity classification loss, a feature similarity loss is designed to discriminate the distances between embeddings of the same person and different person. Extensive experiments on Lock3DFace and Bosphorus show that the proposed PointFace can achieve state-of-the-art performance. All of the proposed models, i.e., the weight shared encoder architecture, feature similarity loss and pair selection strategy, are verified by the ablation study. We are currently working on high quality point cloud synthesis network, such that the recognition accuracy of 3D facial data captured using low cost 3D hardware like Kinect can be further improved.

REFERENCES

[1] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 815–823.
[2] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1701–1708.
[3] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, Sep. 2015, pp. 1–12.
[4] H. Zhou, A. Mian, L. Wei, D. Creighton, M. Hossny, and S. Nahavandi, "Recent advances on singlemodal and multimodal face recognition: A survey," *IEEE Trans. Human-Mach. Syst.*, vol. 44, no. 6, pp. 701–716, Dec. 2014.
[5] S. Zhou and S. Xiao, "3D face recognition: A survey," *Human-Centric Comput. Inf. Sci.*, vol. 8, pp. 1–27, Nov. 2018.
[6] H. Drira, B. A. Boulbaba, S. Anuj, M. Daoudi, and R. Slama, "3D face recognition under expressions, occlusions, and pose variations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 9, pp. 2270–2283, Sep. 2013.
[7] H. Li, Y. W. Di Huang, J.-M. Morvan, and L. Chen, "Towards 3D face recognition in the real: A registration-free approach using fine-grained matching of 3D keypoint descriptors," *Int. J. Comput. Vis.*, vol. 113, pp. 128–142, Jun. 2015.
[8] C. Samir, A. Srivastava, and M. Daoudi, "Three-dimensional face recognition using shapes of facial curves," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 11, pp. 1858–1863, Nov. 2006.
[9] A. S. Mian, M. Bennamoun, and R. Owens, "Keypoint detection and local feature matching for textured 3D face recognition," *Int. J. Comput. Vis.*, vol. 79, no. 1, pp. 1–12, 2008.
[10] S. Z. Gilani, A. Mian, F. Shafait, and I. Reid, "Dense 3D face correspondence," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, pp. 1584–1598, Jul. 2018.
[11] S. Z. Gilani and A. Mian, "Learning from millions of 3D scans for large-scale 3D face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 1896–1905.
[12] H. Lu and H. Shi, "Deep learning for 3D point cloud understanding: A survey," 2020, *arXiv:2009.08920*.
[13] C. R. Qi, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 77–85.
[14] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, vol. 30. Red Hook, NY, USA: Curran Assoc., Inc., 2017, pp. 5099–5108.
[15] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, Oct. 2019.
[16] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 8887–8896.
[17] H. Zhou, Y. Feng, M. Fang, M. Wei, J. Qin, and T. Lu, "Adaptive graph convolution for point cloud analysis," 2021, *arXiv:2108.08035*.

[18] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. 37th Int. Conf. Mach. Learn.*, vol. 119, Jul. 2020, pp. 1597–1607. [Online]. Available: http://proceedings.mlr.press/v119/chen20j.html
[19] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, 2005, pp. 539–546.
[20] C. Jiang, S. Lin, W. Chen, F. Liu, and L. Shen, "PointFace: Point set based feature learning for 3D face recognition," in *Proc. IEEE Int. Joint Conf. Biometrics (IJCB)*, 2021, pp. 1–8.
[21] J. Zhang, D. Huang, Y. Wang, and J. Sun, "Lock3DFace: A large-scale database of low-cost kinect 3D faces," in *Proc. Int. Conf. Biometrics (ICB)*, 2016, pp. 1–8.
[22] A. Savran *et al.*, "Bosphorus database for 3D face analysis," in *Biometrics and Identity Management*. Berlin, Germany: Springer, 2008, pp. 47–56.
[23] G. Mu, D. Huang, G. Hu, J. Sun, and Y. Wang, "Led3D: A lightweight and efficient deep approach to recognizing low-quality 3D faces," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 5766–5775.
[24] P. J. Phillips *et al.*, "Overview of the face recognition grand challenge," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, 2005, pp. 947–954.
[25] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in Neural Information Processing Systems*, vol. 27, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Assoc., Inc., 2014, pp. 1988–1996.
[26] D. Kim, M. Hernandez, J. Choia, and G. Medioni, "Deep 3D face identification," in *Proc. IEEE Int. Joint Conf. Biometrics (IJCB)*, 2017, pp. 133–142.
[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–14.
[28] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2015, pp. 922–928.
[29] C. R. Qi, H. Su, M. Niessner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 5648–5656.
[30] J. Bromley, I. Guyon, Y. Lecun, E. Säckinger, and R. Shah, "Signature verification using a siamese time delay neural network," in *Proc. 7th NIPS Conf.*, Denver, CO, USA, 1993, pp. 1–8.
[31] D. Yi, Z. Lei, S. Liao, and S. Li, "Deep metric learning for person re-identification," in *Proc. 22nd Int. Conf. Pattern Recognit.*, 2014, pp. 34–39.
[32] W. Li, R. Zhao, T. Xiao, and X. Wang, "DeepReID: Deep filter pairing neural network for person re-identification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 152–159.
[33] H. Wang *et al.*, "CosFace: Large margin cosine loss for deep face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5265–5274.
[34] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 4685–4694.
[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–14.
[36] J. Cui, H. Zhang, H. Han, S. Shan, and X. Chen, "Improving 2D face recognition via discriminative face depth estimation," in *Proc. Int. Conf. Biometrics (ICB)*, 2018, pp. 140–147.
[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778.
[38] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 2818–2826.
[39] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.
[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37, 2015, pp. 448–456.
[41] S. Katz, A. Tal, and R. Basri, "Direct visibility of point sets," in *Proc. ACM SIGGRAPH*, New York, NY, USA, 2007, p. 24. [Online]. Available: https://doi.org/10.1145/1275808.1276407
[42] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021.
[43] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on χ-transformed points," in *Advances in Neural Information Processing Systems*, vol. 31. Red Hook, NY, USA: Curran Assoc., Inc., 2018, pp. 820–830.



Changyuan Jiang received the B.Sc. degree from the College of Computer Science and Software Engineering from Shenzhen University in 2019, where he is currently pursuing the M.Sc. degree with the School of Computer Science and Software Engineering. His current research interests include point cloud based 3-D face recognition and point cloud based 3-D face completion.

Shisong Lin received the B.Sc. degree from the College of Electronic Science and Technology and the M.Sc. degree from the College of Computer Science and Software Engineering, Shenzhen University in 2021. His current research interests include 3-D face recognition and 3-D facial expression recognition.

Wei Chen received the Ph.D. degree from the School of Computer Science, University of Birmingham. He is currently a Research Associate with Defense Innovation Institute and Tianjin Artificial Intelligence Innovation Center. His research interests include, but are not limited to, robot vision, 3-D computer vision, and pose estimation.

Feng Liu (Member, IEEE) received the B.Sc. and M.Sc. degrees from Xidian University, Xi'an, China, in 2006, and the Ph.D. degree in computer science from the Department of Computing, HongKong Polytechnic University in 2014. She is currently an Associate Professor with the School of Computer Science and Software Engineering, Shenzhen University. She has published more than 40 papers in academic journals and conferences, and participated in many research projects either as principal investigators or as primary researchers. Her research interests include pattern recognition and image processing, especially focus on their applications to fingerprints. She is a Reviewer for many renowned field journals and conferences.

Linlin Shen (Senior Member, IEEE) received the B.Sc. and M.Eng. degrees from Shanghai Jiao Tong University, Shanghai, China, in 1997 and 2000, respectively, and the Ph.D. degree from the University of Nottingham, Nottingham, U.K., in 2005. He was a Research Fellow with the University of Nottingham, working on MRI Brain Image Processing. He is currently a Pengcheng Scholar Distinguished Professor with the School of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. He is also the Honorary Professor with the School of Computer Science, University of Nottingham and the Distinguished Visiting Scholar with the University of Macau, Macau, China. He serves as the Director of the Computer Vision Institute, AI Research Center for Medical Image Analysis and Diagnosis, and China–U.K. Joint Research Lab for Visual Information Processing. His research interests include deep learning, facial recognition, analysis/synthesis, and medical image processing. He received the Most Cited Paper Award from the *Journal of Image and Vision Computing*. His cell classification algorithms were the winners of the International Contest on Pattern Recognition Techniques for Indirect Immunofluorescence Images held by ICIP 2013 and ICPR 2016. He is listed as the Most Cited Chinese Researchers by Elsevier.