Article

Point CNN: 3D Face Recognition with Local Feature Descriptor and Feature Enhancement Mechanism

Qi Wang, Hang Lei and Weizhong Qian *

School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
* Correspondence: wzq_uestc2023@163.com

Abstract: Three-dimensional face recognition is an important part of the field of computer vision. Point clouds are widely used in the field of 3D vision due to the simple mathematical expression. However, the disorder of the points makes it difficult for them to have ordered indexes in convolutional neural networks. In addition, the point clouds lack detailed textures, which makes the facial features easily affected by expression or head pose changes. To solve the above problems, this paper constructs a new face recognition network, which mainly consists of two parts. The first part is a novel operator based on a local feature descriptor to realize the fine-grained features extraction and the permutation invariance of point clouds. The second part is a feature enhancement mechanism to enhance the discrimination of facial features. In order to verify the performance of our method, we conducted experiments on three public datasets: CASIA-3D, Bosphorus, and Lock3Dface. The results show that the accuracy of our method is improved by 0.7%, 0.4%, and 0.8% compared with the latest methods on these three datasets, respectively.

Keywords: face recognition; convolutional neural network; point clouds; local feature descriptor

1. Introduction

Face recognition, as an important part of the field of computer vision, is widely used in daily life. However, most related studies are based on common RGB images and it is difficult for common digital cameras to obtain effective RGB images under the condition of large illumination changes [1]. The devices of point clouds often do not rely on visible light, such as lidar and Kinect (based on infrared), which makes this kind of data stable to illumination changes and can be applied to some special scenes. The mathematical expression of the point clouds is simple (a group of points in 3D space). However, the disorder of the point clouds makes it difficult for them to have an ordered index such as ordinary 2D images, so it is difficult to use deep learning networks for feature extraction [2]. Deep learning is widely used in various research fields due to its powerful perception. Ref. [3,4] applied deep learning to real engineering technology and achieved outstanding performances. As the pioneers Qi et al. [5] used the symmetric function to construct PointNet that solved the disorder of point clouds in deep learning, many networks based on PointNet have been proposed, such as PointNet++ [6], ppfnet [7], pointcnn [8], etc. Subsequently, point clouds are also widely used in face analysis tasks, such as face detection, pose estimation, face recognition and verification, etc. Particularly in the field of face recognition, a large number of methods have been proposed. However, due to the lack of detailed textures in the point clouds, the fine-grained expression of facial features is still the focus of research in this field. Relying on powerful perception capabilities, convolutional neural networks (CNNs) have made breakthroughs in the field of 2D images. In order to make the point clouds effectively utilize the perceptual power of CNNs, Li et al. [8] constructed a convolution operator, which realizes the permutation invariance of the disordered points through a permutation matrix. Based on [7,8], we utilize the...
convolution operator with a local feature descriptor to construct a new operator, $\psi - conv$, to extract fine-grained features of point faces. Furthermore, we propose a novel feature enhancement mechanism to further enhance the discrimination of facial features and introduce a triplet loss function based on the feature enhancement mechanism for efficient 3D face recognition.

In order to verify the effectiveness of our method, we conduct experiments on three public datasets: CASIA-3D, Bosphorus, and Lock3Dface.

The main novelty and contribution of this paper are summarized as follows:

1. We construct a new operator based on local feature descriptors to achieve fine-grained feature extraction from disordered point clouds;
2. A new feature enhancement mechanism is introduced, which effectively improves the accuracy of the point face recognition;
3. The experimental results on public datasets prove that the accuracy of our proposed method outperforms current advanced algorithms. Additionally, our method can better deal with the interference of facial expressions, partial occlusions, and head pose changes.

2. Related Works

In this section, we briefly review some typical and relevant works in the field of 2D face recognition and 3D face recognition.

2.1. Two-Dimensional Face Recognition

In recent years, the most widely used face recognition methods have mainly been proposed on 2D RGB images. Schroff et al. [9] used a convolutional neural network to extract features and introduced the triplet loss function to build the famous FaceNet for RGB face recognition, which outperforms humans in accuracy. In order to deal with the occlusion and illumination variations, Yang et al. [10] presented a 2D image matrix-based error model (NMR) for face representation and classification. Focusing on the illumination change challenge, Guo et al. [11] proposed a deep network model that takes both visible light images and near-infrared images into account to perform face recognition. Unlike conventional feature descriptors, Lu et al. [12] proposed a new joint feature learning (JFL) approach to automatically learn feature representation from raw pixels for face recognition. Deng et al. [13] proposed an additive angular margin loss to obtain highly discriminative features for face recognition. Aiming at inferring genuine emotions from micro-expression recognition, Zong et al. [14] designed a hierarchical spatial division scheme for spatiotemporal descriptor extraction. Wenhui et al. [15] studied the combination of 2D discriminant analysis and 1D discriminant analysis and proposed a stable framework MMC + LDA for face recognition. Zhang et al. [16] proposed a high-order local pattern descriptor (LDP) for face recognition, which achieves good performance under various conditions.

2.2. Three-Dimensional Face Recognition

With the development of 3D sensors, more and more methods have been proposed for 3D face analysis. Zhang et al. [17] proposed a general approach to deal with the 3D face recognition problem by making use of multiple key point descriptors (MKD) and the sparse representation-based classification (SRC). Chouchane et al. [18] presented an automatic face recognition system in the presence of illumination, expressions, and pose variations based on 2D and 3D information. In [19], Szegedy et al. explored ways to scale up CNNs that aimed at utilizing the added computation for computer vision. In order to optimize deeper neural networks for image recognition, He et al. [20] presented a residual learning framework to ease the training of networks. Based on local derivative pattern (LDP), Soltanpour et al. [21] proposed a descriptor for 3D face recognition. Focusing on the intrinsic invariance to pose and illumination changes, Mu et al. [22] designed a
lightweight yet powerful CNN with low-quality data to achieve an efficient and accurate deep learning solution. Dutta et al. [23] constructed a sparse principal component analysis network (SpPCANet) to extract 3D face features for recognition.

In the field of 3D vision, as the PointNet [5] solves the disorder of point clouds in deep learning, these kinds of data are widely used with their simple mathematical expression; more algorithms are proposed for 3D face recognition. Bhople et al. [24] combined PointNet and Siamese network for similarity learning of point faces and have achieved encouraging performances in the field of face recognition. Atik et al. [25] mapped point clouds to feature maps and used 2D methods to solve 3D face recognition. In order to enhance the robustness of the 3D point cloud face recognition system for multiple expressions and multiple poses, Gao et al. [26] used point clouds as input and constructed a deep learning feature extraction network, ResPoint. Yu et al. [27] modified PointNet and supplemented a few data-guided learning frameworks based on a Gaussian process morphable model for 3D face recognition. Cao et al. [28] utilized PointNet++ and RoPS local descriptors to extract local features of a 3D face. In order to deal with the lack of large-scale 3D facial data, Zhang et al. [29] established a statistical 3D morphable model-based 3D face synthesizing strategy to generate large-scale unreal facial scans to train the proposed network. Yu et al. [30] proposed a meta learning-based adversarial training (MLAT) algorithm for deep 3D face recognition on point clouds, which consists of two alternate modules: adversarial sample generating for 3D face data augmentation and meta learning-based deep network training. Jiang et al. [31] used two weight-shared encoders and a feature similarity loss to guide the encoders to obtain discriminative face representations and have achieved good performance on different datasets. Apart from face recognition, point clouds are also used for other 3D face analysis tasks such as face verification and head pose estimation [1,2,32].

3. Methods

The convolutional neural network (CNN) is highly invariant to image translation, scaling, and tilting through multi-layer feature extraction and regional weight sharing [8]. However, due to the disorder of the point clouds, a CNN cannot directly perform feature extraction on them. In this section, firstly, we introduce a local feature descriptor for fine-grained feature representation and then introduce the \( \psi - conv \) for the convolution operation of the point clouds. Thirdly, depending on the \( \psi - conv \), we construct a new convolutional neural network for facial feature extraction. Fourthly, a new feature enhancement mechanism is proposed to enhance the discrimination of facial features. Finally, based on the feature enhancement mechanism, we adopt a triplet loss function for training and construct an efficient face recognition network.

3.1. Local Feature Descriptor

In this part, inspired by [7], in order to obtain the fine-grained representation of features, we use a hand-crafted descriptor to describe the local geometric features of the point clouds.

For a points pair \( (p_i, p_j) \), the geometric relationship between two points is represented by a four-dimensional descriptor:

\[
\psi_{ij} = (\|d\|_2, \angle(n_i, d), \angle(n_j, d), \angle(n_i, n_j))
\]  

(1)

where \( \|d\|_2 \) represents the Euclidean distance between two points. The \( n_i \) and \( n_j \) are normal vectors of \( p_i \) and \( p_j \), respectively. The \( \angle \) is the angle between two vectors:

\[
\angle(v_i, v_j) = \tan^{-1} \left( \frac{\|v_i \times v_j\|}{v_i \cdot v_j} \right)
\]  

(2)
where $\angle(v_i, v_j) \in [0, \pi)$, the "X" represents the cross-product and the "." represents the dot-product. As described above, the $\psi_{ij}$ describes in detail the geometric relationship between two points through normal vectors and angles.

For a local region, $\{p_1, p_2, p_3, \ldots, p_n\}$, we choose a center point $p_i$, which has a total of $n$ pairs of points (including $(p_i, p_j)$); the geometric feature of this local region is expressed as follows:

$$F_i = [p_1, n_1, p_2, n_2, \ldots, p_j, n_j, \psi_{i1}, \psi_{i2}, \ldots, \psi_{ij}]$$

where $p_j$ is the point in the local region and $n_j$ is the normal vector of point $p_j$. The $\psi_{ij}$ is the four-dimensional descriptor between $p_j$ and center point $p_i$. As shown in Figure 1, $F_i$ uses all points pairs with the center point $p_i$ to describe the spatial geometric characteristics of the local region.

**Figure 1.** Example of the local feature descriptor with the center point $p_i$.

### 3.2. $\psi$ - conv Operator

As mentioned above, because of the disorder of the point clouds, they cannot directly use the convolution operation. To deal with the problem, Li et al. [8] trained a permutation matrix through a multi-layer perceptron (MLP) to realize the permutation invariance of the point clouds. As shown in Figure 2, the points in Figure 2a,b have the same distribution but the orders are different.

In Figure 2, $f_a, f_b, f_c, f_d$ represent the features of the corresponding points and the number represents the order of each point. We use a same convolution kernel $K = [k_\alpha, k_\beta, k_\gamma, k_\delta]^T$ to operate on the above two point clouds:

$$f(a) = \text{Conv}(K, [f_a, f_b, f_c, f_d]^T)$$

$$f(b) = \text{Conv}(K, [f_c, f_a, f_b, f_d]^T)$$

$$f(a) \neq f(b)$$

As shown above, the two sets of point clouds have the same distribution, but the convolution results are different. As shown in Figure 3, in order to make the convolution result only related to the distribution but not to the order, we use a permutation matrix to adjust the order of the points.
Based on the local feature descriptor and the permutation matrix, we construct a new operator $\psi - \text{conv}$, which achieves permutation invariance and fine-grained feature extraction of a local region of the point clouds. The algorithm of the $\psi - \text{conv}$ operator is as follows in Algorithm 1 below:

**Algorithm 1 $\psi - \text{conv}$ operator**

**Input:** $P, p, K$

**Output:** $F_p$

1: $P^* \leftarrow P - p$ \hspace{1em} Local coordinate transformation.
2: $(\psi_1, \psi_2, \ldots, \psi_n) \leftarrow (P^*, p)$ \hspace{1em} Encode point pairs with the descriptor $\psi_{ij}$.
3: $F_i \leftarrow [\psi_1, P^*]$ \hspace{1em} Local feature descriptor.
4: $F_\beta \leftarrow \text{PointNet}(F_i)$ \hspace{1em} PointNet to extract local geometric features.
5: $F_\alpha \leftarrow \text{MLP}_\alpha(P^*)$ \hspace{1em} $\text{MLP}_\alpha$ performs point-by-point feature extraction.
6: $F_x \leftarrow \text{concat}(F_\beta, F_\alpha)$ \hspace{1em} Concatenate $F_\beta, F_\alpha$.
7: $\chi \leftarrow \text{MLP}_\chi(P^*)$ \hspace{1em} Obtain weight matrix $\chi$ through $\text{MLP}_\chi$.
8: $F_x \leftarrow \chi \times F_x$ \hspace{1em} Achieving feature permutation invariance.
9: $F_p \leftarrow \text{Conv}(K, F_x)$ \hspace{1em} Feature extraction using the convolution kernel $K$.

The input of $\psi - \text{conv}$ is the set of feature points in the local region $P = \{p_1, p_2, p_3, \ldots, p_k\}$ and $p$ is the center of $P$ (we take $p$ as the center and use the k-nearest neighbors algorithm (KNN) to sample the nearest $k$ points, $p \in \mathbb{R}^C$). $K$ represents the convolution kernel and the size of $K$ is $k$ (the size of the convolution kernel is the same as the number of points in the local region).

In the first step, the spatial coordinates of $P = \{p_1, p_2, p_3, \ldots, p_k\}$ are transformed into relative coordinates based on the center point $p$ (relative coordinates make local points translation invariant).

The second step is to encode the points pairs in the local region according to Formula (1).

In the third step, according to Formula (3), the local feature descriptor is used to encode the local geometric feature.

The fourth step, the PointNet is used to extract local geometric features. The structure of the PointNet is shown in Figure 4.

**Figure 3.** The permutation matrix to adjust the order of the points.

**Figure 4.** The PointNet for local feature extraction.
The PointNet consists of an MLP and a max pooling layer. The MLP has three layers and the number of nodes in each layer is the same, all of which are $C_\gamma$. After feature extraction, we obtain a local feature $F_\beta \in \mathbb{R}^{C_\gamma}$. In the fifth step, we use the $\text{MLP}_\alpha(P')$ to improve the feature dimension of each point. The structure of the $\text{MLP}_\alpha$ is shown as Figure 5.

![Figure 5. The structure of $\text{MLP}_\alpha$.](image)

In Figure 5, $k$ is the point number of the local region and $C_1$ is the feature dimension of the points. $\text{MLP}_\alpha$ has two convolutional layers. Due to the disorder of the points, only the $1 \times 1$ convolution kernel can be used to increase the dimension of the points (point-by-point). The number of channels of the two convolutional layers is $C_2$ and $C_3$, respectively ($C_3$ is the output dimension, $C_2 = \lfloor (C_1 + C_3)/2 \rfloor$).

In the sixth step, the high-dimensional feature $F_\alpha$ of each point obtained in the fourth step is concatenated with the local geometric feature $F_\beta$ obtained in the fourth step (each point has the same $F_\beta$).

In the seventh step, according to [8], we use $\text{MLP}_\chi$ to train a permutation matrix (as shown in Figure 3, which is only related to the distribution of points; $k$ is the number of points in the local region) that redistributes the weight of each point to eliminate the influence of different orders. The structure of the $\text{MLP}_\chi$ is shown in Figure 6.

![Figure 6. The structure of $\text{MLP}_\chi$.](image)

In Figure 6, a fully connected layer (FC) map $k$ points ($\mathbb{R}^{\text{Dim}}$) to $k \times k$ and reshapes it into a $k \times k$ matrix. Then, we adopt two layers of depth-wise convolution (DC, different from normal convolutional layer, the kernel of depth-wise convolution is responsible for one channel and the feature map has the same number of channels as the input layer) and reshape the feature maps; a $k \times k$ permutation matrix $\chi$ can be obtained:

$$\begin{align*}
\text{DC}(k \times k \rightarrow k \times k) & \rightarrow \text{reshape}(k \times k \rightarrow k \times k) \\
\text{DC}(k \times k \rightarrow k \times k) & \rightarrow \text{reshape}(k \times k \rightarrow k \times k) \rightarrow \chi
\end{align*}$$

Ideally, the permutation matrix is a binary matrix, as shown in Figure 3, but the obtained matrix by $\text{MLP}_\chi$ is a weight matrix, as shown in Figure 7. The weight matrix can approximate the permutation invariance of the local region.
The weight matrix for permutation invariance.

The eighth step, \( F_x \gets \chi \times F_{\ast} \), where \( \chi \) is the weight matrix obtained in the seventh step, \( F_{\ast} \) is the concatenated feature of each point in the sixth step, and the “\( \times \)” represents matrix multiplication. During this step, as shown in Figure 7, the point clouds achieve permutation invariance through the weight matrix \( \chi \) and obtain the weighted features \( F_x \) of the local region.

The ninth step, we can directly perform convolution operation on \( F_x \) to obtain \( F_p \) (feature map of this local region).

The above steps can be represented as follows:

\[
F_p = \psi - \text{Conv}(K, p, P)
\]

\[
= \text{Conv}(K, \text{MLP}_a(P - p) \times [\text{MLP}_a(P - p), \text{PointNet}(P - p)])
\]

where \( K, p, \) and \( P \) represent the input of \( \psi - \text{conv}, \) \( \text{Conv}(. , .) \) is the convolution operation, and \( \text{PointNet}, \text{MLP}_a, \) and \( \text{MLP}_x \) are shown in Figures 4, 5, 6, respectively.

3.3. CNN for Feature Extraction

In Section 3.2, we use the local feature descriptor to describe the local fine-grained feature of a local region of the point clouds and adopt \( \psi - \text{conv} \) to weight the disordered point clouds to achieve the permutation invariance. In this section, based on \( \psi - \text{conv} \), we construct a convolutional neural network (CNN) for facial feature extraction. The structure of our network is shown in Figure 8.

The network consists of 5 convolutional layers; the parameters of each layer are shown in Figure 8, where \( K \) is the number of points in a local region in this layer, \( C \) is the output feature dimension, \( N \) is the number of feature points in the next layer, and \( D \) is the dilation rate, which determines the receptive filed of the convolutional layer: \( (K \times D)/N_p \).
\( N_p \) is the number of feature points in the previous layer. For each layer, we also list the dimension of \( F_\beta, F_\alpha \), which presents the size of the PointNet in Figure 4 and MLP\(_s\) in Figure 5.

Take the first layer as an example. The input point cloud has 1024 points (in our method, according to [32] we use farthest point sampling (FPS) algorithm sample 1024 points for each face). We use k-nearest neighbors algorithm (KNN) to sample 8 nearest points for each point (each local region has 8 points), then we adopt \( \psi - \text{conv} \) to extract the \( F_\beta \) (convolution result, feature map) of each local region, where \( F_\beta \in \mathbb{R}^8, F_\alpha \in \mathbb{R}^8 \) which represent the \( C_\alpha \) in the PointNet of this layer is 8 (as shown in Figure 4) and \( C_\beta \) in the MLP\(_s\) of this layer is 8 (as shown in Figure 5). After the \( \psi - \text{conv} \) operation, each local region becomes a feature map \( F_\beta \in \mathbb{R}^{32} \) and is regarded as a new point in \( \mathbb{R}^{32} \) for the next convolution layer.

After 5 convolution layers, the number of feature points changes as follows: \( 1024 \rightarrow 1024 \rightarrow 512 \rightarrow 256 \rightarrow 128 \rightarrow 32 \). The feature dimension changes as follows: \( 3 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow 256 \rightarrow 512 \). In the last convolutional layer, the receptive field \((K \times D)/N_p = 1\), which means the last 32 feature points “see” the whole region of the previous layer. Then, we use a global average pooling to extract the global feature \( F_g \in \mathbb{R}^{512} \) from 32 feature points. According to [9], in order to avoid large differences between facial features, we normalize the features by 2-norm (\( L_2 \)):

\[
L_2(F_g) \rightarrow F_L \in \mathbb{R}^{512}
\]  

(9)

3.4. Feature Enhancement Mechanism

In Section 3.3, we obtain normalized facial features \( F_L \in \mathbb{R}^{512} \) (the value of each dimension is between \((-1\) and \(1\)). However, not every dimension plays the same role in the recognition task. For example, the larger the eigenvalue of a certain dimension, the higher the recognition contribution of this dimension provides; on the contrary, the smaller the eigenvalue of a certain dimension is, the lower the recognition contribution of this dimension provides. Based on the above phenomenon, we propose a new feature enhancement mechanism to enhance the discrimination of features.

First, take the absolute value of the eigenvalues of each dimension according to Formula (10). Then, use softmax to map \( |F_i| \) to the probability distribution between \((0\) and \(1\)). In this step, according to Formula (11), the numerator of eigenvalue with a large absolute value grows fast and the numerator of eigenvalue with a small absolute value grows slowly (because \((e^x)' = e^x\)). The stretched eigenvalues can improve the discrimination of features. Finally, as shown in Formula (12), we restore the eigenvalues to their original positive and negative distributions.

\[
|F_i| = (|x_1|, |x_2|, |x_3|, \ldots, |x_{512}|)
\]  

(10)

\[
F_s = (f_1, f_2, f_3, \ldots, f_i) = \text{softmax}(|F_i|), f_i = \frac{e^{\text{softmax}(|x_i|)}}{\sum_{k=1}^{n} e^{\text{softmax}(|x_k|)}}
\]  

(11)

\[
F_s = (f_1^*, f_2^*, f_3^*, \ldots, f_{512}^*), f_i^* = \begin{cases} f_i, x_i \geq 0 \\ -f_i, x_i < 0 \end{cases}
\]  

(12)

We use softmax to enhance the eigenvalue in \( F_L \) but, in order to avoid ignoring some original information in \( F_L \), we utilize the enhancement parameter \( \lambda \) to linearly add \( F_L \) and the enhanced feature \( F_s^* \):

\[
F_T = F_L + \lambda F_s^*
\]  

(13)

In Formula (13), the eigenvalues in \( F_L \) and \( F_s^* \) are between \((-1\) and \(1\), but there is still a large gap. Parameter \( \lambda \) determines the degree of coupling of the two features and also determines the contribution of the proposed feature enhancement mechanism to the \( F_T \). The structure of feature enhancement mechanism is shown in Figure 9.
3.5. Triplet Loss Function

In the feature space, the metric distance between objects is related to the similarity and the training purpose of the face recognition network is to make the same object have a closer metric distance, with a far metric distance between different objects.

In the field of 2D face recognition, FaceNet [9] constructed a triplet loss function and has surpassed humans in accuracy. In this section, we construct a triplet loss based on enhancement parameter $\lambda$.

The triplet loss function includes three types of samples: anchor samples (Anchor), positive samples (Positive), and negative samples (Negative). The anchor samples and positive samples come from the same object and the negative samples come from different objects. As shown in Figure 10, the purpose of the network is to make the metric distance between the anchor sample ($F_A$) and the positive sample ($F_P$) with the farthest distance smaller than the anchor sample and the negative sample ($F_N$) with the closest distance.

According to Formula (13), face features $F_T$ are composed of two parts: $F_L$ and $\lambda F_s$. As shown in Formula (12), $F_L$ to $\lambda F_s$ is a non-linear change process. If directly using $F_T$ for measurement, some original details of the features will be ignored. Therefore, in this section, we construct a new triplet loss according to parameter $\lambda$ and the training purpose in Figure 10 can be expressed as follows:

$$\|F_A^A - F_L^A\|_2^2 + \|\lambda F_s^A - \lambda F_s^P\|_2^2 + \beta < \|F_A^A - F_L^N\|_2^2 + \|\lambda F_s^A - \lambda F_s^N\|_2^2$$  (14)

where $F_A^A$, $F_L^P$, and $F_N^N$ represent the $F_L$ feature (as Formula (9)) of Anchor, Positive, and Negative samples, respectively. $F_s^A$, $F_s^P$, and $F_s^N$ represent the $F_s$ feature (as Formula (12)) of Anchor, Positive, and Negative samples, respectively. $\lambda$ is the enhancement parameter (as Formula (13)) and $\beta$ is a margin that is the minimum distance between $\|F_A^A - F_L^A\|_2^2 + \|\lambda F_s^A - \lambda F_s^P\|_2^2$ and $\|F_A^A - F_L^N\|_2^2 + \|\lambda F_s^A - \lambda F_s^N\|_2^2$.

In the training process, only samples that do not satisfy Formula (14) are used to optimize the model (the loss of sample that satisfies the Formula (14) is 0):

$$\|F_A^A - F_L^A\|_2^2 + \|\lambda F_s^A - \lambda F_s^P\|_2^2 + \beta > \|F_A^A - F_L^N\|_2^2 + \|\lambda F_s^A - \lambda F_s^N\|_2^2$$  (15)

The loss function of our model is defined as follows:

$$Loss = \sum [AP - AN + \beta]_+$$  (16)
\[
AP = \|F_{\text{norm}} A - F_{\text{norm}} P\|^2_2 + \|\lambda F_{\text{en}} A - \lambda F_{\text{en}} P\|^2_2
\] (17)
\[
AN = \|F_{\text{norm}} A - F_{\text{norm}} N\|^2_2 + \|\lambda F_{\text{en}} A - \lambda F_{\text{en}} N\|^2_2
\] (18)

where \(N\) represents the total number of triplet samples satisfying Formula (15). During the training process, according to the loss function, Anchor and Positive samples with far distance become closer. Anchor and Negative samples with close distance become farther. The whole structure of our network is shown in Figure 11.

Ideally, we want the farthest pair of same objects (hard positive pair) to have a smaller metric distance than the closest different objects (hard negative pair). However, for a large number of training samples, it is difficult to find the hard positive pair and the hard negative pair. Sample selection is very important for the performance of the model. As described in Section 3.3, each point face samples 1024 points as input. According to \[8,9\], we set the mini-batch in each batch. For a mini-batch, 40 samples are selected from the same subject and we find the hard positive pair in the 40 samples. The hard negative pair is randomly selected from other subjects. The margin \(\beta\) in Formula (14) is computed in each mini-batch. The size of each batch in our network is fixed at 1800. The ADAM optimizer has an initial learning rate of 0.01 for the training of our model.

4. Experiments

In this section, we conduct a series of experiments on public datasets to verify the effectiveness of our proposed method. Firstly, we introduce three public datasets CASIA-3D, Lock3Dface, and Bosphorus. Then, we conduct ablation experiments and explore the influence of enhancement parameter \(\lambda\). Finally, we use our best results for comparison with current advanced methods and analyze the comparison results.

4.1. Datasets

CASIA-3D [33]: This dataset used Minolta vivid910 to scan 123 subjects and each subject collected 37 or 38 face images with the influence of different facial expressions, head poses, and light intensities. The dataset has a total of 4626 face samples.

We divide the training set and test set of CASIA-3D according to the method in [26]. Only the frontal face and small pose interference samples are used for experiments, including 1784 samples in the training set and 1783 samples in the test set.

Bosphorus: Savran et al. [34] collected this dataset for studying 2D and 3D face analysis tasks. This dataset, based on the structured light 3D system, collected a total of 4666 samples of facial data from 105 subjects; one-third of the subjects were professional actors and each subject provided 35 types of expressions.

We divide the training set and test set according to the method in [23], in which the training set contains 2403 samples and the test set contains 2263 samples.
Lock3DFace: Zhang et al. [35] collected this dataset by Kinect V2 for 3D face analysis. A total of 5671 samples from 509 subjects were included. According to different scenarios, this dataset is divided into five subsets covering variations in expression (FE), neutral face (NU), occlusion (OC), pose (PS), and time lapse (TM).

We divide the training set and test set of Lock3DFace according to the method in [31], in which the 340 subjects are randomly selected as the training set and the remaining 169 subjects are selected as the test set.

4.2. Ablation Experiments

In this section, we first investigate the effectiveness of the proposed feature enhancement mechanism and explore enhancement parameter $\lambda$ in Formula (13).

In this step, we set $\lambda$ as a fixed value and explore the effect of $\lambda$ on the accuracy of the network. The results on CASIA-3D, Bosphorus, and Lock3DFace are reported in Tables 1–3.

<table>
<thead>
<tr>
<th>$\lambda$</th>
<th>0</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
<th>35</th>
<th>40</th>
<th>45</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc (%)</td>
<td>89.5</td>
<td>94.8</td>
<td>95.9</td>
<td>97.1</td>
<td>97.8</td>
<td>98.3</td>
<td>98.6</td>
<td>98.7</td>
<td>98.8</td>
<td>98.9</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>55</td>
<td>60</td>
<td>65</td>
<td>70</td>
<td>75</td>
<td>80</td>
<td>85</td>
<td>90</td>
<td>95</td>
<td>100</td>
</tr>
<tr>
<td>Acc (%)</td>
<td>98.6</td>
<td>97.9</td>
<td>97.1</td>
<td>96.3</td>
<td>95.7</td>
<td>95.1</td>
<td>94.5</td>
<td>93.7</td>
<td>93.6</td>
<td>93.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\lambda$</th>
<th>0</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
<th>35</th>
<th>40</th>
<th>45</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc (%)</td>
<td>90.1</td>
<td>93.3</td>
<td>94.7</td>
<td>95.6</td>
<td>96.6</td>
<td>97.3</td>
<td>98.0</td>
<td>98.3</td>
<td>98.5</td>
<td>98.8</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>55</td>
<td>60</td>
<td>65</td>
<td>70</td>
<td>75</td>
<td>80</td>
<td>85</td>
<td>90</td>
<td>95</td>
<td>100</td>
</tr>
<tr>
<td>Acc (%)</td>
<td>98.9</td>
<td>98.6</td>
<td>97.8</td>
<td>97.2</td>
<td>96.7</td>
<td>96.2</td>
<td>95.5</td>
<td>94.9</td>
<td>94.0</td>
<td>93.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\lambda$</th>
<th>0</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
<th>35</th>
<th>40</th>
<th>45</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc (%)</td>
<td>82.2</td>
<td>84.5</td>
<td>85.3</td>
<td>86.0</td>
<td>86.5</td>
<td>86.9</td>
<td>87.3</td>
<td>87.5</td>
<td>87.6</td>
<td>87.8</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>55</td>
<td>60</td>
<td>65</td>
<td>70</td>
<td>75</td>
<td>80</td>
<td>85</td>
<td>90</td>
<td>95</td>
<td>100</td>
</tr>
<tr>
<td>Acc (%)</td>
<td>88.0</td>
<td>87.7</td>
<td>86.9</td>
<td>86.2</td>
<td>85.5</td>
<td>84.9</td>
<td>84.5</td>
<td>84.1</td>
<td>83.9</td>
<td>83.8</td>
</tr>
</tbody>
</table>

According to Tables 1–3, when $\lambda = 50$, $\lambda = 55$, and $\lambda = 55$ on CASIA-3D, Lock3DFace, and Bosphorus, our network achieves the best accuracy 98.9%, 98.9%, and 88.0%, respectively. Figure 12 intuitively presents the relationship between $\lambda$ (x-axis) and accuracy (y-axis).

![Figure 12. Accuracy change curves with different $\lambda$ on three datasets.](image-url)
As shown in Figure 12, when $\lambda = 0$, according to Formula (13), the proposed feature enhancement mechanism is not utilized. With the increase in $\lambda$, the feature enhancement mechanism begins to enhance the features and the accuracy of the network begins to increase, which proves that our feature enhancement mechanism can effectively enhance the discrimination of features and improve the recognition accuracy of the network. As $\lambda$ continues to increase, the accuracy begins to decline. This is because the contribution of $F_i$ in Formula (9) becomes small. In this case, features with smaller absolute values will be ignored ($F_s$ is mainly to enhance the features with large absolute value), which will interfere with the accuracy of the network.

Although the best accuracy on the three datasets corresponds to a different $\lambda$, according to Figure 12, when $\lambda \in [40,55]$, the accuracy curves reach a stable peak. In this interval, $F_L$ and $F_s$ have the best coupling degree, which can provide the best discrimination for facial features. According to the evaluation method in [9], Table 4 shows the relationship between $\lambda$ in the peak interval and list the mean accuracy with the standard error of the mean. According to Table 4, the accuracy is relatively stable in this interval for each dataset, which proves that our method has good generalization ability.

### Table 4. Accuracy of the different $\lambda$ on three datasets.

<table>
<thead>
<tr>
<th>$\lambda$</th>
<th>CASIA-3D (Acc%)</th>
<th>Bosphorus (Acc%)</th>
<th>Lock3DFace (Acc%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>98.66 ± 0.04</td>
<td>98.23 ± 0.07</td>
<td>87.38 ± 0.12</td>
</tr>
<tr>
<td>45</td>
<td>98.77 ± 0.03</td>
<td>98.47 ± 0.03</td>
<td>87.48 ± 0.12</td>
</tr>
<tr>
<td>50</td>
<td>98.89 ± 0.01</td>
<td>98.78 ± 0.02</td>
<td>87.70 ± 0.10</td>
</tr>
<tr>
<td>55</td>
<td>98.55 ± 0.05</td>
<td>98.88 ± 0.02</td>
<td>87.91 ± 0.09</td>
</tr>
</tbody>
</table>

As the experimental results show above, we explored the relationship between $\lambda$ and accuracy and also demonstrated the effectiveness of the feature enhancement mechanism. Then, in the second step, we continue to explore the effectiveness of the distance metric utilized in the triplet loss function. In Section 3.5, instead of taking the $F_T$ (in Formula (9)) as a whole, we measure the distance by $F_L$ and $F_s$ separately:

$$
\|F_L^\lambda - F_T^\lambda\|_2^2 \neq \|F_L^\lambda - F_s^\lambda\|_2^2 + \|\lambda F_s^\lambda - F_T^\lambda\|_2^2
$$

(19)

According to Formula (19), the right part is not equal to the left part. In order to verify the performance of two measurement methods, we conduct a comparison experiment on three datasets; the results are listed in Table 5.

As shown in Table 5, where $L^*$ represents the left part of Formula (19) to measure the distance between two features, while $L$ represents the right part. Table 5 lists the mean accuracy with the standard error of the mean on three datasets. According to Table 5, the results of the two measurement methods are very close but $L$ is higher. This is because the eigenvalue with a smaller absolute value in $F_T$ will be ignored and $L$ can be regarded as two kinds of features to measure the distance between two samples, which is better to capture more differences.

### Table 5. Accuracy of the different metric distance on CASIA-3D.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$L^*$ Acc (%)</th>
<th>$L$ Acc (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASIA-3D</td>
<td>98.27 ± 0.03</td>
<td>98.89 ± 0.01</td>
</tr>
<tr>
<td>Bosphorus</td>
<td>98.08 ± 0.02</td>
<td>98.88 ± 0.02</td>
</tr>
<tr>
<td>Lock3DFace</td>
<td>87.39 ± 0.11</td>
<td>87.91 ± 0.09</td>
</tr>
</tbody>
</table>

### 4.3. Comparison Experiments

The results of ablation experiments prove the effectiveness of our proposed method. In this section, according to [31], we use our best results to conduct comparison experiments with current advanced methods on three public datasets and analyze the results.

Firstly, in order to verify the effectiveness of the proposed $\psi - conv$ network, we use different point clouds based networks to extract facial features and perform face
recognition under the same setting on CASIA-3D. The accuracy curves in the training process are shown in Figure 13 and the results are listed in Table 6.

![Accuracy curves](image.png)

**Figure 13.** Accuracy change curves during training with different feature extraction network.

**Table 6.** Comparison of accuracy achieved by different feature extraction network on the test set of CASIA-3D.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Acc (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PointNet++ [6]</td>
<td>95.6</td>
</tr>
<tr>
<td>NICP [36]</td>
<td>90.3</td>
</tr>
<tr>
<td>RSCNN [37]</td>
<td>95.9</td>
</tr>
<tr>
<td>Pointcnn [8]</td>
<td>97.5</td>
</tr>
<tr>
<td>Ours</td>
<td>98.9</td>
</tr>
</tbody>
</table>

As shown in Figure 13, during the training process, our accuracy curve is higher than other methods and, as the results listed in Table 6, our method also achieves the best accuracy on the test set, which prove the effectiveness of our $\psi - \text{conv}$ network. Compared with the method in [8], our network has a similar architecture but adds a local feature descriptor. The comparison results with the method in [8] prove that our network based on a local feature descriptor can better obtain facial fine-grained features and is more conducive to improving the accuracy of the model.

Tables 7–9 list the comparison results with the latest face recognition methods on three datasets, respectively.

**Table 7.** Comparison of accuracy achieved by different methods on CASIA-3D.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Acc (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chouchane et al. [18]</td>
<td>96.8</td>
</tr>
<tr>
<td>Dutta et al. [23]</td>
<td>98.2</td>
</tr>
<tr>
<td>Gao et al. [26]</td>
<td>97.6</td>
</tr>
<tr>
<td>Cao et al. [28]</td>
<td>97.9</td>
</tr>
<tr>
<td>Ours</td>
<td>98.9</td>
</tr>
</tbody>
</table>

**Table 8.** Comparison of accuracy achieved by different methods on Bosphorus.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Acc (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zhang et al. [17]</td>
<td>93.0</td>
</tr>
<tr>
<td>Soltanpour et al. [21]</td>
<td>97.3</td>
</tr>
<tr>
<td>Dutta et al. [23]</td>
<td>98.5</td>
</tr>
<tr>
<td>Cao et al. [28]</td>
<td>98.0</td>
</tr>
<tr>
<td>Ours</td>
<td>98.9</td>
</tr>
</tbody>
</table>
Table 9. Comparison of accuracy achieved by different methods on Lock3DFace.

<table>
<thead>
<tr>
<th>Methods</th>
<th>FE (%)</th>
<th>NU (%)</th>
<th>OC (%)</th>
<th>PS (%)</th>
<th>Total (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K. He et al. [20]</td>
<td>96.1</td>
<td>99.3</td>
<td>54.9</td>
<td>61.4</td>
<td>76.6</td>
</tr>
<tr>
<td>C.S et al. [19]</td>
<td>93.6</td>
<td>99.0</td>
<td>57.0</td>
<td>54.1</td>
<td>74.4</td>
</tr>
<tr>
<td>M.S et al. [38]</td>
<td>95.7</td>
<td>98.9</td>
<td>61.4</td>
<td>69.9</td>
<td>79.5</td>
</tr>
<tr>
<td>Mu et al. [22]</td>
<td>98.1</td>
<td>99.6</td>
<td>78.1</td>
<td>70.4</td>
<td>84.2</td>
</tr>
<tr>
<td>Jiang et al. [31]</td>
<td>98.5</td>
<td>99.5</td>
<td>80.1</td>
<td>73.7</td>
<td>87.2</td>
</tr>
<tr>
<td>Ours</td>
<td>98.5</td>
<td>99.3</td>
<td>82.6</td>
<td>74.9</td>
<td>88.0</td>
</tr>
</tbody>
</table>

The results in Tables 7 and 8 show that, under different datasets, our accuracy is higher than other methods.

As described in Section 4.1, Lock3DFace has five subsets: expression changes (FE), normal face (NU), partial occlusion (OC), head pose changes (PS), and time lapse (TL). In order to intuitively verify the performance of our method in different scenarios, we conduct a comparison experiment on the first four subsets: FE, NU, OC, and PS. The results are shown in Table 9. According to the results, in the NU subset, which has no other interference, Jiang et al. [31] achieves the best accuracy, but in the OC and PS subsets, our method achieves the best accuracy, which proves that our network is better to cope with partial occlusions and head pose interference. Figure 14 shows the t-SN example of our network for face recognition on three datasets (each dataset selects five subjects for classification and each color represents one subject). As shown in Figure 14, the classification results on CASIA-3D and Bosphorus are more convergent, but on Lock3DFace are more discrete. This is because there are fewer samples for each subject in Lock3DFace and there is also more interference for samples. However, according to Figure 14c, our method can still distinguish different objects clearly on Lock3DFace.

![Figure 14. T-SN examples of face recognition](image)

Apart from the accuracy, the time cost is also an important indicator for measuring the efficiency of the network. Table 10 lists the comparison results of different methods in terms of time costs. In Table 10, “Ours *” represents our method without the feature enhancement mechanism. Compared with “Ours **”, our time cost is very close. This is because the feature enhancement adopts the softmax function to stretch the features, the calculation complexity is low, and no additional network parameters are added. The comparison results in Table 10 show that our network also maintains good real-time performance.

Table 10. Comparison of time costs, where “Ours *” represents our method without the feature enhancement mechanism.

<table>
<thead>
<tr>
<th>Methods</th>
<th>fps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dutta et al. [23]</td>
<td>7</td>
</tr>
<tr>
<td>M.S et al. [38]</td>
<td>6</td>
</tr>
<tr>
<td>Xiao et al. [1]</td>
<td>117</td>
</tr>
<tr>
<td>Xiao et al. [32]</td>
<td>125</td>
</tr>
</tbody>
</table>
5. Conclusions

Since point clouds lack detailed textures and since face recognition require fine-grained representation of features, this paper proposes a new operator, $\psi - \text{conv}$, based on the local feature descriptor to realize fine-grained feature extraction of disordered point clouds by a convolutional neural network and constructs the feature enhancement mechanism to improve feature discrimination; meanwhile, the triplet loss function is adopted to optimize the network. In order to verify the performance of our method, we conducted experiments on the CASIA-3D, Lock3Dface, and Bosphorus datasets. The results of the ablation experiments prove that the feature enhancement mechanism and the triplet loss can effectively improve the recognition accuracy of the model. The results of the comparison experiments show that our network outperforms current advanced methods and can better cope with the interference of face expressions, partial occlusions, and head pose changes. Meanwhile, our network also has good real-time performance and can be applied in real scenarios. However, when the pose interference is too large due to the lack of some facial features, the accuracy of our method is still insufficient. We will further explore new methods to improve the accuracy under large pose interference and investigate new algorithms for 3D face analysis, such as head pose estimation, expression recognition, face detection, and other 3D visual tasks in real applications.

Author Contributions: Conceptualization, Q.W.; data curation, Q.W.; formal analysis, Q.W. and W.Q.; investigation, Q.W.; methodology, Q.W.; project administration, H.L.; resources, Q.W.; software, Q.W.; supervision, H.L.; visualization, Q.W.; writing—original draft, Q.W.; writing—review and editing W.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by the National Natural Science Foundation of China (61802052). Innovative Research Foundation of Ship General Performance (26422206). The Sichuan Science and Technology Program (2023YFSY0040).

Institutional Review Board Statement: Not applicable for studies not involving humans or animals.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.