3D FACE IDENTIFICATION AND RECONSTRUCTION WITH RANGE SENSOR

[1] *Tsun-An Hsieh* (謝尊安), [2] *Chiou-Shann Fuh* (傅楸善)

[1] Graduate Institute of Networking and Multimedia,
National Taiwan University, Taipei, Taiwan,
[2] Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan
E-mail: r05944036@.ntu.edu.tw; fuh@ntu.edu.tw;

ABSTRACT

A method of data augmentation for 3D face model and using it for 3D face identification is proposed in our work. In the past few years, researchers had achieved significant progress on 2D face identification and verification through neural network approaches, such as VGG (Visual Geometry Group) Face, GoogleNet Inception, and ResNet (Residual Network). Since there are so many hyper parameters that need to be optimized in neural networks, large data must be provided for training. In 2017, FaceID was proposed by Apple Inc. Face identification has been scaled up from 2D to 3D. However, training a 3D face classifier is difficult. 3D face datasets nowadays are so small that even a large set of 3D faces contains only 4,666 faces of 105 identities. In order to solve the lack of data, we use transfer learning [13], and several data augmentation methods by generating face mesh from different views to make the classifier more robust and discriminative.

Keywords: *3D Face Identification, 3D Face Generation, Convolutional Neural Networks.*

1. INTRODUCTION

1.1. Overview

Our approach consists of two parts, feature extraction and data augmentation. They will be introduced in the following sections. Base on VGG Face descriptor, and 3D face reconstruction, we provide a revised 3D face identification workflow, in order to solve small amount of training data, which is the main contribution of this paper.

1.2. Feature Extraction

Descriptors play an important role in face identification because the performance of a classifier is greatly affected by the representation of features extracted by descriptors. Descriptors play an important role in face identification because the performance of a classifier is greatly affected by the representation of features extracted by descriptors.

Recently, neural network approaches are getting popular with the successful pre-trained models such as VGG, ResNet, and GoogleNet Inception. Inheriting the benefit of SIFT descriptor, CNNs are also scale-invariant and shift-invariant. Furthermore, CNNs use relatively less preprocessing than conventional hand-crafted descriptors, that is, CNN learns filters independent of prior knowledge and effort, which is the major advantage.

Our approach to feature extraction is to fine-tune a VGG Face descriptor with augmented data. We use representative features of VGG Face descriptor extracted for ensuing 3D face identification.

1.3. Data Augmentation through Synthesized Faces

The past works have focused on exploring robust features and descriptors on the image geometry of a 3D face by hand-crafted methods. Although these researches achieve good performance on recognition, some of them still suffer from the relatively complex mathematical operations that may cause more computation. Recently, researchers use neural networks to learn better representations. However, many parameters need to be optimized in a model, but due to the lack of data, data augmentation is an important process to solve this problem. The two main benefits of data augmentation are: (1) to generate more data for training and (2) to prevent model from overfitting. D. Kim et al. [6] proposed a training workflow for 3D face identification with relatively small dataset.

There are two key points in this workflow: (1) Using one morphable face model to generate more faces with different poses (pitch), expressions, add-on noises for each identity. With this process on the original dataset, there are approximately infinite data for the training phase. (2) Use pre-trained models instead of training from scratch, which is called transfer learning. The intuition of transfer learning is to use the knowledge from other tasks with the similar structure. The process of training a model using different data is called fine-tuning.

For the part of face model generation, we use the method that A. Tran et al. [24] proposed, which generates 3D face models using CNNs for foundation model and bump maps for detailed texture, and the second part is to fine-tune the pre-trained VGG Face descriptor for face identification.

2. BACKGROUND

2.1. Reconstruct Human Face with Deep Convolutional Neural Network and 3D Morphable Model (3DMM)

There are many works on 3D face reconstruction with multiple images, such as S. Liang et al. [9] reconstruct heads from Internet photos and O. M. Parkhi et al. [12] uses deep convolutional neural work for face reconstruction. Although these methods achieve great accuracy on reconstruction, they still suffer from the difficulty of data acquisition from different identities in different poses.

Recently, researchers prefer using only one image to reconstruct human face. A. Tran et al. [24] proposed a work of regressing 3DMM face in deep neural network described as bellow.

2.1.1. Generating Training Data

To generate training data, A. Tran et al. use multi-image 3DMM generation method proposed by M. Piotraschke et al. [17]. The dataset applied to the method is CASIA (Chinese Academy of Sciences, Institute of Automation) Web Face dataset, which is a constrained dataset with faces in different postures. These 3DMM faces trained by CASIA Web Face dataset are then used as ground truth for the CNN 3DMM regressor.

To regress a detailed face, they simply regress a face shape and its texture with two generative models:

$$S' = \hat{s} + W_S\alpha\,, \qquad T' = \hat{t} + W_T\beta.$$

Here, $\hat{s}$ and $\hat{t}$ represent the mean face shape and mean texture of all the aligned 3D facial scans in the 3DMM, respectively. Matrices W_s and W_t are principal components computed from the same facial scans. Vectors α and β are used for the linear combinations with W_S and W_T for face reconstruction.

2.1.2. Pooled 3DMM

Instead of using multiple images for face generation, they use a pooling method for 3DMM parameters with better efficiency. Specifically, 3DMM parameters $\gamma_i = [\alpha_i, \beta_i], i \in 1 \dots N$ for N views belong to the same identity, and the pooling function is

$$\gamma = \sum_{i=1}^{N} \omega_i \cdot \gamma_i\,, \quad \sum_{i=1}^{N} \omega_i = 1,$$

where ω_i are the normalized confidences for each image provided by Constrained Local Neural Fields (CLNF) facial landmark detector [7].

2.1.3. Learning to Regress Pooled 3DMM

To achieve this, they use a very deep CNN which is ResNet with 101 layers trained for face recognition. After substituting the last fully connected output layer for the 198 dimensional feature vector γ, they fine-tune the network on CASIA face images using the pooled 3DMM estimations as target values.

2.2. 3D Face Recognition Methods

H. Patil et al. [14] proposed an overview of face recognition topics in 3D faces. The challenges in 3D face recognition including different expressions, occlusions, noise (such as uneven surface). We will briefly describe some 3D face recognition approaches in the following sections.

2.2.1. Curvature-Based Approaches

Curvature-based methods are based on the curvature information associated with the facial structure. The curvature information includes level curves and radial curves. Level curves are extracted by determining the distance from the surface to a reference point. The radial curves are extracted by the radial lines from the center of a face model (nose-tip), and it is very compact in comparison with level curves approaches.

2.2.2. Morphable Model-Based Approaches

Morphable models, known as deformable models, vary their shapes according to some extracted features.

To compare the probe face and gallery faces, the first is to register the probe with the morphable model so that the coordinate system is aligned, and then calculate the point-to-point feature correspondence of a probe and the morphable model. Next, the morphable model can be fitted by conditioning the deformation parameters obtained.

Generally, a morphable model is built by extracting features of training data. The benefits of this type of approaches are that a morphable model can generate various poses, scales, or even occlusions.

Kakadiaris et al. [4] proposed an expression-invariant 3D face recognition system using morphable model known as AFM (Annotated Face Model). The fitted AFM is sampled into 2D representation then transformed to a norm map.

In the research that Haar and Veltkamp [3] proposed in 2010, they built multi-resolution PCA (Principal Component Analysis) model with small collections of facial landmarks for neutral and other expression scans. A single morphable identity model and 7 isolated expression models are built. For face matching, they applied L_1 distance to calculate similarity, and the expressions are neutralized because of the separated model. Although this approach is expression-invariant, it still needs human-labeled landmarks, and its performance highly depends on how accurate landmarks are localized.

3. METHODOLOGY

3.1. Overview

Our detailed methods for 3D face identification are described in this section. In Section 3.2, data augmentation is demonstrated. It consists of (1) pose variation (Section 3.2.1), (2) add-on noises (Section 3.2.2) and (3) occlusions (Section 3.2.3). After augmentation, a VGG Face pre-trained model is fine-tuned through these data. Section 3.3 shows the detail of fine-tuning process and how we choose the parameters of our model. In Section 3.4, we will describe how the fine-tuned model is used in the identification problem. The whole workflow is shown in Figure 3-1.

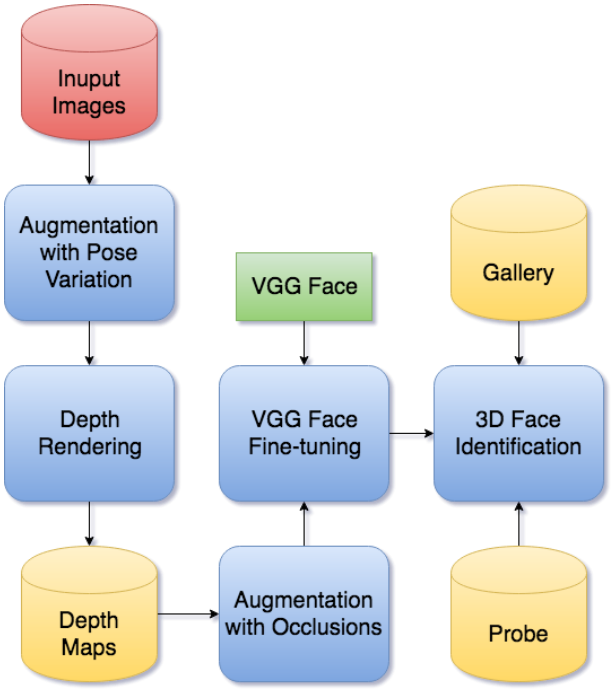


Fig. 3-1: Workflow for 3D face identification. 2D images collected from the subset of VGGFace2 Dataset [2] and CASIA-Face V5 [23] are used for augmentation. After fine-tuning VGG Face, the model will be applied to identification.

3.2. Data Augmentation

Basically, our augmentation method is simplified from [24]. For each 3D face model, we only use single 2D image as input. The output 3D face model is controlled by a foundation face model s and an expression e. A linear 3DMM, is presented as:

$$s = \hat{s} + \sum_{i=1}^{k} \alpha_i W_i$$

where, $\hat{s}$ is the mean face of 3D face models; α_i is coefficient of W_i; and W_i is the i^{th} principal component of face s, where $W \in \mathbb{R}^{3n \times k}$; $3n$ is the 3D coordinate for the n points; and k is number of elements in a principal component. Given a 2D image, s generated by regressing α. Expression model is also represented as a linear equation as:

$$e = \sum_{j=1}^{m} \beta_j M_j$$

where e is expression combined by m principal components of expression M_j; $M \in \mathbb{R}^{3n \times m}$; $3n$ is the coordinates of n points; and m is the number of elements in M. After we estimate s and e, the foundation face F is derived by adding expression on the regressed face model:

$$F = \hat{s} + \sum_{i=1}^{s} \alpha_i W_i + \sum_{j=1}^{m} \beta_j M_j = s + e.$$

Since the foundation face F consists of a general shape and expression, the next step is to add detailed texture to it. To achieve this, we need a bump map derived from 2D image. Given the coordinate of an input image $c(x, y)$, the depth map of a detailed face model $d'(c)$, the depth map of foundation face $d(c)$, and the bump map $B(c)$, we get $d'(c) = d(c) + B(c)$.

After several transforms and noises, occlusions are then added to these synthesized face models. For rigid transform, we apply rotation and translation on a face mesh as follows:

$$F' = F\big[\, R(\theta_x, \theta_y) \big|\, t(x, y, z) \,\big]$$

where R is the rotation matrix, and t is translation vector, $-10.8° < \theta_x, \theta_y < +10.8°$. Translation scale x, y, and z are in the range of $\pm 10\%$ of $\sqrt{P}$ where P is the resolution of depth map. The type of noise we add on face models is salt-and-pepper noise [6]. We generate random occlusions to prevent overfitting on

specific patterns. Furthermore, the occlusion on training data increases the robustness of our model when accessories are on face.

We select 1,740 subjects from CASIA-Face V5 (476 subjects) and VGG Face 2 dataset (1264 subjects) by the size of image and pass them into the augmentation pipeline for face model generation. We render the depth map of each face model from 25 different viewpoints and split them into training/validation set with ratio of 80% and 20%.

The augmented data vary from different views; hence our model is pose-invariant.

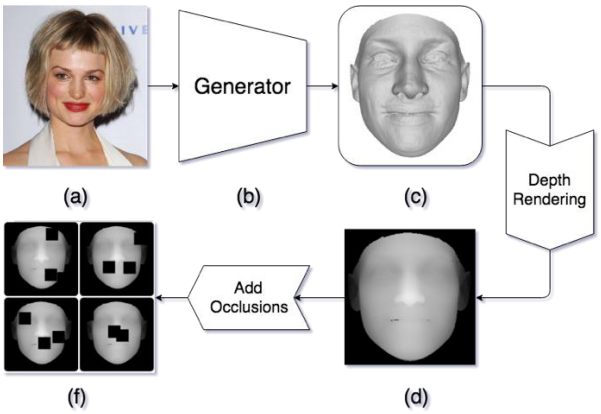


Fig. 3-1: Face generation pipeline of our method. (a) 2D input image passed to the face generator (b). (c) Synthesized face mesh. For each face mesh, we render it into a depth map (d), and add random occlusions on it to enhance the robustness of our recognition model.

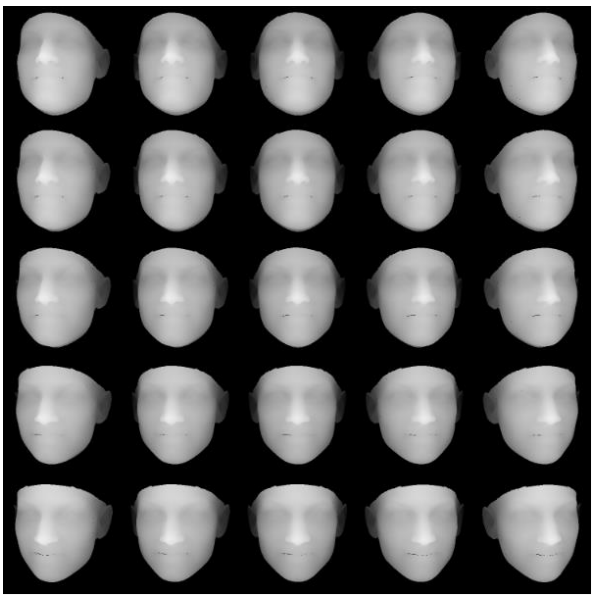


Fig. 3-2: Depth maps render from single face mesh with 25 different views. For a depth map, the adjacent (four-neighbor) depth maps are $\pm 5.4°$ on $x-$axis and $y-$axis.

3.3. Fine-Tuning

To construct a 3D face recognition model, we use the pre-trained VGG Face model [12]. All layers in VGG Face except the output layer (a fully connected layer with softmax activation function, 2,622-dimensional feature vector) are transferred into our model. In other words, we replace the last layer of VGG Face with a fully connected layer in 1,740 dimensions (number of subjects we want to classify) with a softmax activation function.

The softmax function, or normalized exponential function, is a generalization of logistic function that normalizes the summation of a K-dimensional vector to 1. It is used in multi-class classification, and represented as:

$$\sigma(v)$$

All input images (depth maps) are linearly resized to 224 pixels $\times$ 224 pixels $\times$ 3 channels , and rescaled from $[0, 255]$ to the range of $[0, 1]$.

While fine-tuning the VGG Face network, we first fix the weights transferred from VGG Face. Specifically, we set the gradients to zero to fix weights. The main reason is that the weights of the last fully connected layer are randomly initialized with normal distribution, and they are not tuned, yet. If the transferred weights are not fixed at first while training, they will be destroyed by the meaningless gradients back-propagated from the last layer.

After all the weights of last layer are converged (by observing the loss), we unlock the entire model but set different learning rates to transferred layers and last layer. Here, refer to [6], we set learning rate for the transferred layers to 0.001 and 0.01 for the last layer.

For optimization, we use the Stochastic Gradient Decent (SGD) with momentum set to 0.9 and learning rate decay over each updated. Our loss function is cross entropy:

$$H(p, q) = -\sum_x p(x)\log(q(x)),$$

where $p(x)$ and $q(x)$ are two probability distributions. Notice that $p(x)$ is the true distribution of data, and $q(x)$ is the predicted distribution. The intuition [27] of cross entropy is to determine the expected message length per datum under $q(x)$ while the real distribution is $p(x)$.

3.4. Identification

For identification, fine-tuned model is used for feature extraction. Here, instead of using the entire model, we remove the last layer and use the 4096-dimensional vector as the representation for a face.

The feature vectors are normalized by applying square root element-wise. After transform all images in the gallery and probe set into feature vectors, we use PCA to reduce the dimension of feature vector from 4096 dimensions to 99 dimensions because the 4096-dimensional feature vector is very sparse. The reduced features are then matched by cosine distance between probe and gallery sets, and we take the closest identity as our prediction result for each probe subject.

4. EXPERIMENTAL RESULT

4.1. Overview

In this section, we show the experimental results of our algorithm under different circumstances. In Section 4.2, we briefly introduce the datasets used for training and testing and also show how we evaluate the performance for our identification model in Section 4.3. The results are shown in Section 4.4. The system information of our device is as follow.

System Environment	
OS	Ubuntu 16.04 LTS
CPU	Intel® Core™ i7-7700K 4.2 GHz
RAM	32 Gigabytes
Language	Python, Matlab
GPU	Nvidia Geforce GTX1080
Frameworks	Keras, TensorFlow

4.2. Datasets

4.2.1. VGG Face2 Dataset [2]
VGGFace2 Dataset is a large-scale face dataset containing 3.31 million images over 9131 identities, with average 362.6 images/identity. The images are in large variations of pose, age, illumination, ethnicity, and profession. We use a subset of VGGFace2 Dataset (1,264 identities) for training data augmentation by selecting images with the largest size.

4.2.2. CASIA-Face V5 [23]
CASIA-Face V5 is Version 5 for the CASIA Face Image Database. It contains 2,500 color images over 500 subjects in pose variations. All face images are 16-bit BMP (BitMaP) at resolution $640x480$ pixels. We use the subset of CASIA-Face V5 with frontal faces for training data augmentation, 476 identities totally.

4.2.3. Bosphorus Database [18, 19]
The Bosphorus Database is a 3D face database for testing. It contains 4,666 color images and depth maps over 105 identities. Each identity has 9 classes:

1) Neutral Pose & Expression
2) Lower Face Action Unit
3) Upper Face Action Unit
4) Action Unit Combination
5) Emotional Expression
6) Yaw Rotation
7) Pitch Rotation
8) Cross Rotation
9) Occlusion

For our experiment, we use 105 neutral faces as gallery set and the remaining 3,098 faces in various expressions and occlusions as probe set.

4.3 Evaluation

Here we use receiver operating characteristic (ROC) [29] as our evaluation metric. A receiver operating characteristic is a plot for diagnosing a binary classifier. The ROC curve plots the true positive rate against the false positive rate at different settings of thresholds.

For instance, in a problem of face verification, the true positive rate is the rate that the number of faces correctly predicted over all predictions, and the false positive rate means the ratio of the number of faces wrongly predicted over all predictions.

4.4. Analysis on Validation Set and Bosphorus Dataset

In this section, we will discuss how our face identification model performs under various circumstances on Bosphorus face dataset. After comparison, we can see how expressions affect the performance of our model.

Figure 4-1 shows ROC curve in the neutral to neutral case. In this case, it has the best performance on Bosphorus dataset using our model. In Figure 4-2 (neutral to non-neutral) and Figure 4-3 (neutral-all), we can observe that if the variation of expression increases, the performance of our model will degrade.

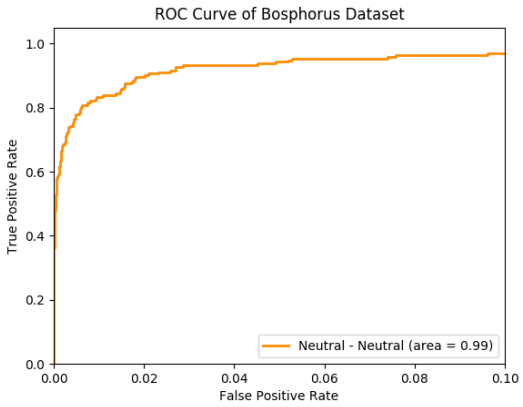


Fig. 4-1: ROC curve of neutral to neutral set. Here we use 105 neutral faces as the gallery, and the remaining 191 neutral faces as the probes.

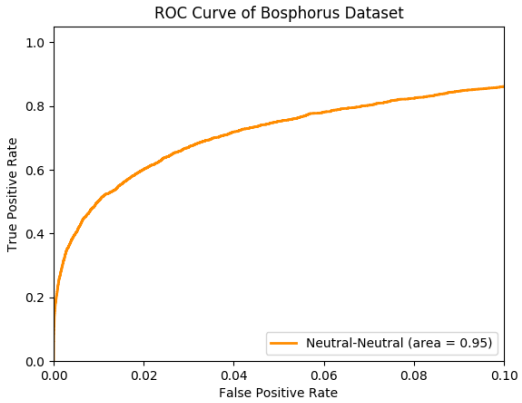


Fig. 4-2: ROC curve of neutral to non-neutral set. Gallery consists of 105 neutral faces, and the probe consists of 2,908 non-neutral faces.

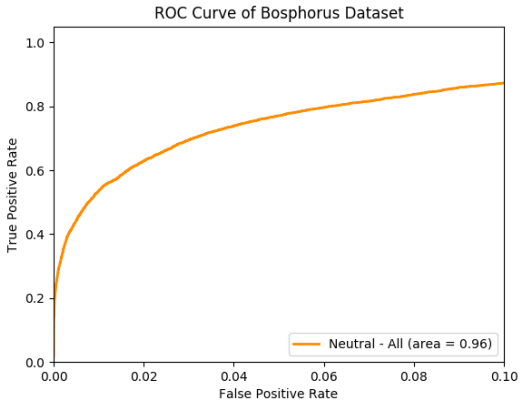


Fig. 4-3: ROC curve of neutral to all set. Gallery consists of 105 neutral faces, and there are 3,204 other faces of the database in the probe set.

Here, by observing the Area Under Curve (AUC), we can easily realize that expression highly affects the performance of our model. Our model is not expression-invariant, because we did not generate different expressions for subjects, and CNN cannot learn how to adapt to variation on expressions.

In spite of the low robustness on adapting expression variation, our model can easily handle transforms such as rotation, translation, and serious occlusions are also tolerated.

To test the robustness of our model, we randomly sample five non-frontal faces as validation set, which is not seen by the model. Here, we have two settings. The first is faces with rotation and translation but without occlusion. The second setting is faces with rotation, translation, and occlusions.

Figure 4-4 is the ROC curve of the first setting, the area under curve score is perfect. Even with occlusions, our model still performs very well, and we cannot even notice the difference between the results.

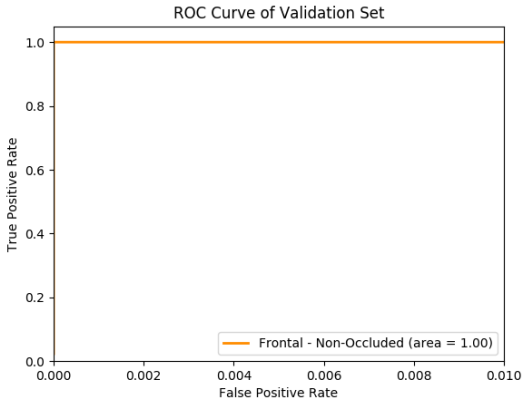


Fig. 4-4: ROC curve of non-occluded validation set. The validation set here contains 8,700 images with five different poses.

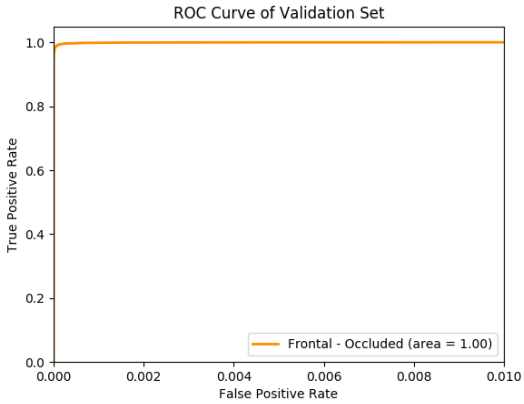


Fig. 4-5: ROC curve of validation set with occlusions. Compared with the ROC curve in Figure 4-4, there is almost no difference.

Accuracy over different cases on Bosphorus dataset and Validation set. Our model is mainly trained in augmented data, so its accuracy is lower when applied in real-world data. Expression variations cause serious degrade on accuracy, but pose variations and occlusions do not affect accuracy much.

By this observation, we can claim that our model has great robustness to rotation and translation. To make our model expression-invariant, we need to change the augmentation method and generate face meshes with different expressions.

	Boshporus Dataset			Validation Set	
Case	Neutral Neutral	Neutral Non-Neutral	Neutral All	Frontal Non-Occluded	Frontal Occluded
Accuracy	80.10%	44.50%	48.53%	100.00%	99.70%

Table 4-1: Accuracy over different cases on Bosphorus dataset and Validation set. Our model is mainly trained in augmented data, so its accuracy is lower when applied in real-world data. Expression variations cause serious degrade on accuracy, but pose variations and occlusions do not affect accuracy much.

4.5. Visualization Views of Convolutional Kernels

In this section, we will visualize kernels in both the original VGG Face descriptor and our 3D face descriptor. In a convolution-pooling block, we take filters in the first convolution layer for visualization.

4.5.1. Gradient Ascending

To achieve this, we activate all kernels in different layers by gradient ascending. Initially, we generate an image with normal distribution as our input image, and feed the image into pre-trained models (VGG Face and our 3d face descriptor) and apply gradient ascending on the image instead of the model itself.

The intuition of this approach is to find an image that activates a specific convolution kernel the most. For an input image x, and a kernel k with m rows and n columns, we define how the kernel activated by the mean value of the kernel as follows:

$$k_{activated} = \frac{1}{mn} \sum_{\substack{0<i<m \\ 0<j<n}} k_{ij}$$

and the image x' that activates the kernel the most:

$$x' = \arg\max_{x}(k_{activated}(x))$$

4.5.2. Visualization

In this section, we will demonstrate what convolutional layers learned. For each layer, we sample 49 images among all kernels in a layer.

First Layer:

For the first convolutional layer in Figs. 4-6 and 4-7, there is only a slight difference between the kernels from VGG Face and our model. Notice that in shallower layers, the kernels are not close to gray scale which means red, green, and blue channels have quite different behavior, weight distribution, or influence.

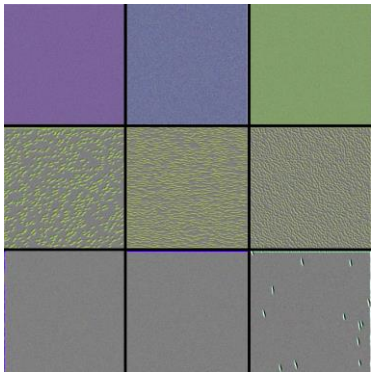


Fig. 4-3: Visualization of the first layer of VGG Face. We can easily observe fine textures in kernels.

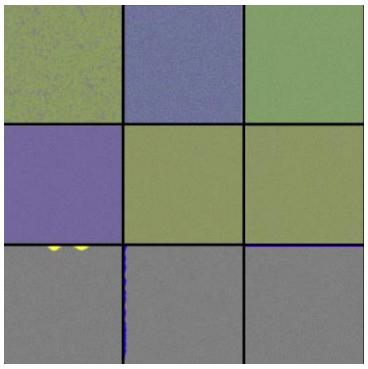


Fig. 4-4: Visualization of our model. Most kernel colors are purple, green, or gray. Compared with the VGG Face, our model is flatter and has less information in the kernel.

Third Layer:

The third layers of two models are shown in Figs. 4-8 and 4-9. Here, textures of kernels are more than the kernels from previous layer. Textures in this layer are similar to the texture of skin and leather in different directions.

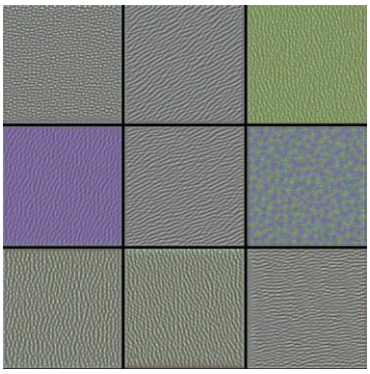


Fig. 4-5: Kernels from third layer of VGG Face. The texture is similar to skin, or materials such as leather.

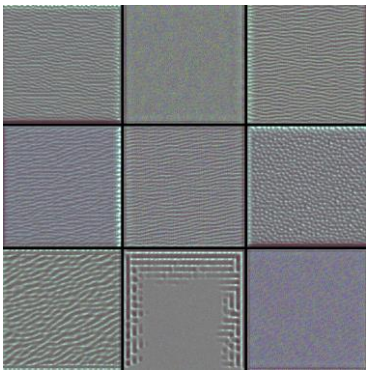
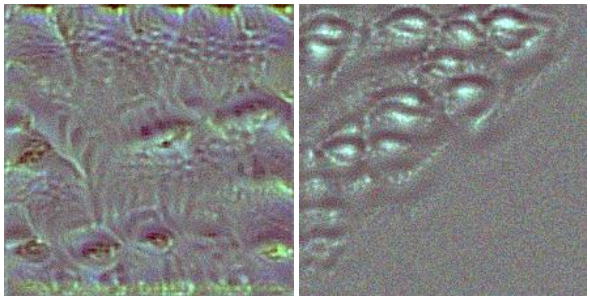


Fig. 4-6: Kernels from third layer of our model. Compared with kernels of VGG Face, the color of our filters tends to turn into gray scale. The texture with vertical and horizontal lines may be caused by the rectangular occlusions we added on.

Fifth Layer:

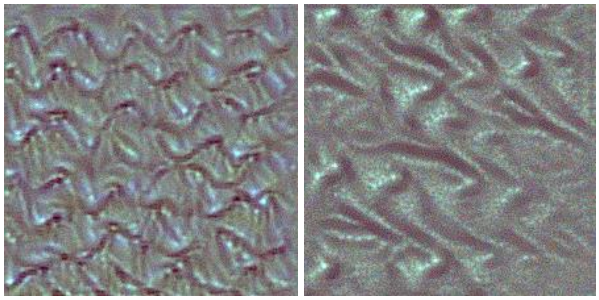
In the fifth layer, it is easy to see strong features that activate kernels in deeper layers. These kernels extract

the higher features such as eyes, nose, ears, and mouth, and they are illustrated in Figs. 4-10, 4-11, 4-12, and 4-13 respectively.



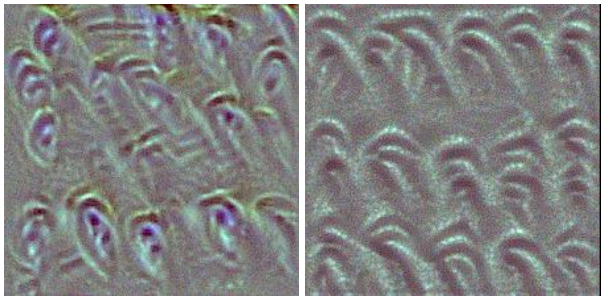
(a) (b)

Fig. 4-7: High level kernels similar to eyes. (a) The eye texture from VGG Face. (b) Eye texture from our model.



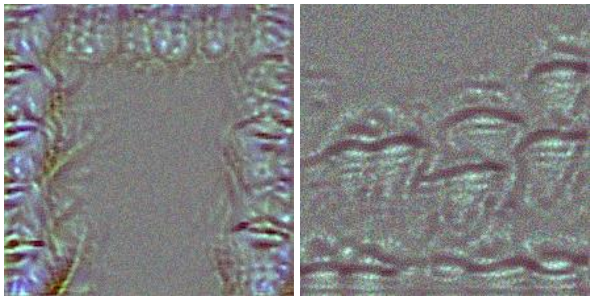
(a) (b)

Fig. 4-8: High level kernels similar to nose. (a) The nose texture from VGG Face. (b) The nose texture from our model.



(a) (b)

Fig. 4-9: High level kernels similar to ears. (a) The ear texture from VGG Face. (b) The ear texture from our model.



(a) (b)

Fig. 4-10: High level kernels similar to mouth. (a) The mouth texture from VGG Face. (b) The mouth texture from our model.

With this observation, we know how convolutional networks understand the world. At shallow layers, convolutional kernels extract fine features that hard to distinguish by human eyes. With the layer goes deeper, features are more understandable, so the high-level features are combined by low-level features. In addition, we can see the same high-level features vary from multiple directions, which is a strong proof that our model is rotation-invariant.

4.5. Implementation

Our system consists of a range sensor (Bellus3D Face Camera) and an Android smart phone (Samsung Galaxy S8). First, a user uses the range sensor for 3D face capture, then the mesh is transmitted to our server for registration or identification. Our model is tested on additional dataset of 625 3D faces over 25 identities, and the accuracy is 96.91%.

5. CONCLUSION AND FUTURE WORK

In this work, we develop a face recognition method using depth maps with small amount of data.

The recognition method works well in normal situations, even the images are occluded. In addition, range sensors are usually based on infrared patterns or time-of-flight, so 3D face recognition systems work in the dark. Due to the similarity of 2D face recognition and 3D face recognition, by using transfer learning technique, our model is very efficient at training.

However, there are also drawbacks. First, the cost for a range sensor is expensive. For instance, commercial range sensor such as Kinect V1 or V2 costs about $300. Second, for face classification and identification, we need a high-performance Graphics Processing Unit (GPU) card for parallel computing on matrices.

For these drawbacks, we hope to reduce the size of our network for efficiency, and collect more data in the real world for training in the future, instead of using augmented data.

6. ACKONLEDGEMENT

This research was supported by the Ministry of Science and Technology of Taiwan, R.O.C., under Grants MOST 104-2221-E-002-133-MY2 and MOST 106-2221-E-002-220, and by Test Research, Jorgin Technologies, III, Egistec, D8AI, and LVI.

REFERENCES

[1] J. F. Blinn. “Simulation of Wrinkled Surfaces,” *Proceedings of ACM SIGGRAPH Conference on Computer Graphics*, vol. 12, no. 3, pp. 286–292, 1978.

[2] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and Andrew Zisserman, “VGGFace2: A Dataset for Recognising Faces Across Pose and Age,” *Proceedings of IEEE Conference on Automatic Face and Gesture Recognition*, Xi’an, China, pp. 1-11, 2018.

[3] F. B. Haar and R. C. Veltkamp, “Expression Modeling for Expression-Invariant Face Recognition,” *International Journal of Systems and Applications in Computer Graphics*, vol. 34, no. 3, pp. 231-241, 2010.

[4] I. A. Kakadiaris, G. Passalis, G. Toderici, M. N. Murtuza, L. Yunliang, N. Karampatziakis, and T. Theoharis, “Three-Dimensional Face Recognition in the Presence of Facial Expressions: An Annotated Deformable Model Approach,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, pp. 640-649, 2007.

[5] H. E. Khiyari and H. Wechsler, “Face Recognition across Time Lapse Using Convolutional Neural Networks,” *Journal of Information Security*, vol. 7, no. 3, pp. 141-151, 2016.

[6] D. Kim, M. Hernandez, J. Choi, and G. Medioni, “Deep 3D Face Identification,” *Proceedings of IEEE International Joint Conference on Biometrics*, Denver, CO, USA, pp. 133-142, 2017.

[7] K. Kim, T. Baltruaitis, A. Zadeh, L.-P. Morency, and G. Medioni. “Holistically Constrained Local Model: Going beyond Frontal Poses for Facial Landmark Detection,” *Proceedings of British Machine Vision Conference*, York, UK, pp. 95.1-95.12, 2016.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Proceedings of Conference on Neural Information Processing Systems*, Stateline, NV, USA, pp. 1106-1114, 2012.

[9] S. Liang, L. G. Shapiro, and I. Kemelmacher-Shlizerman. “Head Reconstruction from Internet Photos,” *Proceedings of European Conference on Computer Vision*, Amsterdam, also *LNCS*, vol. 9906, Springer, pp. 360-374, 2016.

[10] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[11] X. Lu and A. Jian, “Deformation Modeling for Robust 3D Face Matching,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 8, pp. 1346-1357, 2010.

[12] O. M. Parkhi, A. Vedaldi, and A. Zisserman. “Deep Face Recognition,” *Proceedings of British Machine Vision Conference*, Swansea, UK, pp. 1-12, 2015.

[13] S. J. Pan and Q. Yang, “A Survey on Transfer Learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 2, no. 10, pp. 1345-1359, 2010.

[14] H. Patil, A. Kothar, K. Bhurchandi, “3-d face recognition: features, databases, algorithms and challenges,” *Artificial Intelligence Review*, vol. 44, no. 3, pp. 393-441, 2015.

[15] P. Paysan, R. Knothe, B. Amberg, S. Romhani, and T. Vetter, “A 3D Face Model for Pose and Illumination Invariant Face Recognition,” *Proceedings of IEEE International Conference on Advanced Video and Signal Based Surveillance*, Genoa, Italy, pp. 296-301, 2009.

[16] K. Pearson, “On Lines and Planes of Closest Fit to Systems of Points in Space,” *Philosophical Magazine*, vol. 2, no. 11, pp. 559-572, 1901.

[17] M. Piotraschke and V. Blanz. “Automated 3D Face Reconstruction from Multiple Images Using Quality Measures,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, Nevada, pp. 3418-3427, 2016.

[18] Savran, B. Sankur, M. T. Bilge, “Regression-based Intensity Estimation of Facial Action Units,” *Image and Vision Computing*, vol. 30, no. 10, pp. 774-784, 2012.

[19] A. Savran, N. Alyüz, H. Dibeklioğlu, O. Çeliktutan, B. Gökberk, B. Sankur, and L.

Akarun, “Bosphorus Database for 3D Face Analysis,” *Biometrics and Identity Management*, pp. 47-56, 2008.

[20] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A Unified Embedding for Face Recognition and Clustering,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Boston, Massachusetts, pp. 815-823, 2015.

[21] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *Proceedings of International Conference on Learning Representations*, San Diego, CA, pp. 1-14, 2015.

[22] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller, “Multi-view Convolutional Neural Networks for 3D Shape Recognition,” *Proceedings of International Conference on Computer Vision*, Santiago, Chile, pp. 945-953, 2015.

[23] T. N. Tan, “CASIA-Face V5,” http://biometrics.idealtest.org/, 2010.

[24] A. Tran, T. Hassner, I. Masi, E. Paz, Y. Nirkin, and G. Medioni, “Extreme 3D Face Reconstruction: Seeing through Occlusions,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, Utah, pp. 1-14, 2018.

[25] P. Viola and M. Jones, “Rapid Object Detection Using a Boosted Cascade of Simple Features,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, HI, USA, pp. I-511-518, 2001.

[26] Wikipedia, “Convolutional Neural Network,” https://en.wikipedia.org/wiki/Convolutional_neural_network, 2018.

[27] Wikipedia, “Cross Entropy,” https://en.wikipedia.org/wiki/Cross_entropy, 2018.

[28] Wikipedia, “Depth Map,” https://en.wikipedia.org/wiki/Depth_map, 2018.

[29] Wikipedia, “Receiver Operating Characteristic,” https://en.wikipedia.org/wiki/Receiver_operating_characteristic, 2018.