2018 13th IEEE International Conference on Automatic Face & Gesture Recognition

Deep Transfer Network with 3D Morphable Models for Face Recognition

Zhanfu An, Weihong Deng, Tongtong Yuan, Jiani Hu
Beijing University of Posts and Telecommunications
Beijing, 100876, China
{anzhanfu, whdeng, yuantt, jnhu} @bupt.edu.cn

***Abstract*—Data augmentation using 3D face models to synthesize faces has been demonstrated to be effective for face recognition. However, the model directly trained by using the synthesized faces together with the original real faces is not optimal. In this paper, we propose a novel approach that uses a deep transfer network (DTN) with 3D morphable models (3DMMs) for face recognition to overcome the shortage of labeled face images and the dataset bias between synthesized images and corresponding real images. We first utilize the 3DMM to synthesize faces with various poses to augment the training dataset. Then, we train a deep neural network using the synthesized face images and the original real face images. The results obtained on LFW show that the accuracy of the model utilizing synthesized data only is lower than that of the model using the original data, although the synthesized dataset contains much considerably images with more unconstrained poses. This result shows that a dataset bias exists between the synthesized faces and the real faces. We treat the synthesized faces as the source domain, and we treat the actual faces as the target domain. We use the DTN to alleviate the discrepancy between the source domain and the target domain. The DTN attempts to project source domain samples and target domain samples to a new space where they are fused together such that one cannot distinguish the domain from which a specific image is from. We optimize our DTN based on the maximum mean discrepancy (MMD) of the shared feature extraction layers and the discrimination layers. We choose AlexNet and Inception-ResNet-V1 as our benchmark models. The proposed method is also evaluated on the LFW and SLLFW databases. The experimental results show that our method can effectively address the domain discrepancy. Moreover, the dataset bias between the synthesized data and the real data is remarkably reduced, which can thus improve the performance of the convolutional neural network (CNN) model.**

I. Introduction

Deep learning, particularly convolutional neural networks (CNNs), has achieved promising results in face recognition in recent years. Though CNNs are impressive, training a robust and reliable neural network requires large-scale labeled data. The reported CNNs, [1], [2], [3], [4] and so on, are trained on different face databases; unfortunately, most of these databases are not publicly available. One commonly used face dataset that is publicly available is the CASIA-WebFace collection [5] with only 495K images, which are not enough images to train many large CNNs such as FaceNet [2]. Therefore, in most real-world applications, harvesting and labeling large datasets have become an effective approaches to enhance the performance of CNNs. Not only the quantity but also the variation are of importance in data collection. To train a model with good generalization ability, the training data should simultaneously consider inter-class variations (differences between different people) and intra-class variations (differences within the same person), which is difficult and requires considerable effort.

Figure 1. Data bias across both domains can be alleviated by using transfer learning. First, we maximize domain confusion by making the marginal distributions of the two domains as similar as possible. Second, we match the distributions of the labels given features.

Masi I et al. [6] realize that collecting and labeling massive training sets is not easy for improving networks. They synthesize training data using a 3D generic face model to augment the training dataset. The idea that face images can be synthetically generated by using 3D rendering technology to aid face recognition systems was proposed long ago. This idea was originally proposed in [7] and then effectively used in [1] [8] [9]. In constrast to the above method, Masi I et al. [6] use other transformations to generate new images (*e.g.,* other poses, different shapes, and facial expressions) rather than generating frontal faces, which increases the size of the CASIA WebFace collection to several times its original size. Experimental results have demonstrated its effectiveness and have achieved state-of-the-art performances on the LFW and IJB-A datasets. Later, Masi I et al. [10] considered the computational cost of rendering and proposed a new method for the rapid synthesis of massive face sets for face recognition.

However, there are two limitations in the aforementioned methods. First, synthetic face images with a 3D face model are always "false" (virtual face images) and the images are not real photographs taken from a real camera. Do the real data and the synthesized data have the same distribution? Moreover, is there a dataset bias between these data? In other words, can networks trained by using synthesized images capture the features of the real data? Second, Masi I et al. [6] synthesize face images for

978-1-5386-2335-0/18/$31.00 ©2018 IEEE
DOI 10.1109/FG.2018.00067

both training and testing. As is known, 3D synthesis techniques that use a traditional 3D face model are related to the facial landmarks [12] and pose estimate. When facial landmarks are not accurately localized and the pose estimate is not proper, the synthesized faces will be meaningless images. Even if the pose was correctly estimated, warping images across poses involves interpolating intensities, which leads to smoothing artifacts and information loss. Although this may affect training, it is far more serious at test time, particularly when multiple images represent a subject, which may include lower resolution, extreme poses and so on.

To solve the aforementioned problems, namely, the shortage of labeled face images and the different distributions between synthesized images and real images, we propose a novel approach that uses a deep transfer network (DTN) with 3D morphable models (3DMMs) for face recognition. Our approach performs transfer learning across both domains, as shown in Fig. 1. An overview of our proposed approach is illustrated in Fig. 2, which mainly consists of two main components: data augmentation and DTN. We first synthesize faces with various poses on the CASIA-WebFace collection to augment the training dataset using the 3DMM [11], which is similar to [6]. Then, we use the synthesized data and the original data to train the same neural network and we test the trained model on LFW [15]. The Experiments show that there is a dataset bias between the synthesized data and the real data (different distributions). We treat the synthesized faces as the source domain, and we treat the original real faces as the target domain. To alleviate the discrepancy between the source and target domains, we use a deep DTN that attempts to project the source domain samples and the target domain samples to a common subspace. We optimize our DTN based on the maximum mean discrepancy (MMD) [16] of the shared feature extraction layers and the discrimination layers to reduce the dataset bias, which is conducive to training strong classifiers by maximizing the data confusion.

For comparison, we choose two networks as our basic network to evaluate our proposed approach: AlexNet model [17] and Inception-ResNet-V1 model [18]. We evaluate our approach on the LFW [15] and SLLFW [19] databases. The experimental results show that the proposed method can effectively overcome the shortage of training data and the dataset bias between different domains.

II. DATA AUGMENTATION

A. Synthesizing Faces

Similar to [21], we employ 3DMM [11] as the parametric face model to encode 3D face geometry and extend the shape model to cover facial expressions by adding delta blendshapes. Specifically, the parametric face model describes 3D face geometry $\mathbf{S}$ with principal component analysis (PCA)

$$\mathbf{S} = \overline{\mathbf{S}} + \mathbf{A}_{id}\alpha_{id} + \mathbf{A}_{exp}\alpha_{exp}, \tag{1}$$

where $\overline{\mathbf{S}}$ denotes the shape of the average 3D face. $\mathbf{A}_{id}$ is the principal axes trained on the 3D face scans with a neutral expression, and α_{id} is the shape weight. $\mathbf{A}_{exp}$ is the principal axes trained on the offset between expression scans and neutral scans, and α_{exp} represents the expression weight. For diversity and mutual complement, we use the Basel face model (BFM) for α_{id} [13] and FaceWarehouse [14] for α_{exp}.

Given a 2D facial image, our goal is to predict the optimal shape parameter and projection vector that minimize the difference between the projected 3D shape and the ground truth. After obtaining the optimal pose parameters $\mathbf{R}$, identity parameters α_{id} and expression parameters α_{exp}, we can render new facial images in various poses. In addition, the appearance surrounding the face region also contains discriminative information for face recognition [20]. Inspired by [21], we aim to preserve the identity information as much as possible by a 3D transformation in our work, which is different from [6]. Fig. 3 shows synthetic examples of face images, including ideal and unsatisfactory images. The typical failures mainly come from poor facial landmark detection, as shown in Fig. 3(c), and strong facial expressions, as shown in Fig. 3(d), which is a common situation and is not beneficial for testing.

B. Datasets Bias

After obtaining the synthesized faces, the most straightforward way to use this information is to train the CNN with the synthesized data and original data together. However, image datasets are inherently biased [23]. In face synthesis process in particular, images will be introduced with more or less deformation (smoothing artifacts), as shown in Fig. 3(c)(d). Therefore, it is critical to address the dataset bias between the real data and the synthesized data for training the neural network. In our experiments, we use the synthesized data and the original data separately to train the AlexNet network and the Inception-ResNet-V1 network. Then, we evaluate models that we trained on LFW. Fig. 4 shows the performances on different networks which are trained by using different datasets.

Interestingly, as shown in Fig. 4, the recognition rate has decreased on both networks when using the synthesized face dataset rather than the real face dataset, although the synthesized dataset contains more diverse poses and faces. This result indicates that image synthesis alone cannot improve the face recognition rate. Moreover, this result proves that synthesized faces and real faces have different distributions. In other words, there is a dataset bias between them. In addition, the high recognition algorithms rely on high- capacity CNN, which requires millions of supervised images for initial training.

We extract the feature of the same person of synthesized faces and real faces on the Inception-ResNet-V1 network, which is trained by using synthesized faces and real faces. We use the PCA to reduce the dimension. Fig. 5 shows the distribution of the first two dimensions from PCA projections of the penultimate layer. As shown, the distributions of features between the synthesized faces and real faces are scattered.

We denote the synthesized faces as the source domain, and we denote the original 2D real faces as the target domain. Our goal is to minimize the distribution bias among different domains while preserving the properties of the data.

III. DEEP TRANSFER NETWORK FOR FACE RECOGNITION

A. Problem Description and Notations

For clarity, we formulate our problem and some notations. In the following, upper-case and lower-case characters represent matrices and vectors, respectively. Unless otherwise specified, the symbols s and t used in superscript denote the source domain and target domain, respectively. $\mathrm{x}^s \in \mathbb{R}^d$ and $\mathrm{x}^t \in \mathbb{R}^d$ represent samples in the source and target datasets, respectively. $D^s = \{(\mathrm{x}_1^s, y_1^s), .., (\mathrm{x}_{n^s}^s, y_{n^s}^s)\}$ and $D^t = \{(\mathrm{x}_1^t, y_1^t), .., (\mathrm{x}_{n^t}^t, y_{n^t}^t)\}$ represent the source dataset and the target dataset, respectively, where n^s and n^t are the numbers of the samples. $y_i^\bullet$ is their class label, and $y_i^\bullet \in \{1, 2, ..., c\}$. Denote $\mathrm{X}^s = [\mathrm{x}_1^s, ..., \mathrm{x}_{n^s}^s] \in \mathbb{R}^{d \times n^s}$ and $\mathrm{X}^t = [\mathrm{x}_1^t, ..., \mathrm{x}_{n^t}^t] \in \mathbb{R}^{d \times n^t}$ as the data matrices of D^s and D^t , respectively. $P(\mathrm{x}^s)$ and $P(\mathrm{x}^t)$ represent the marginal distributions of the source and target datasets, while $P(y^s|\mathrm{x}^s)$ and $P(y^t|\mathrm{x}^t)$ represent the conditional distributions.

Our task is to use 3D rendering technology with the limited faces to generate faces with various poses and natural expressions

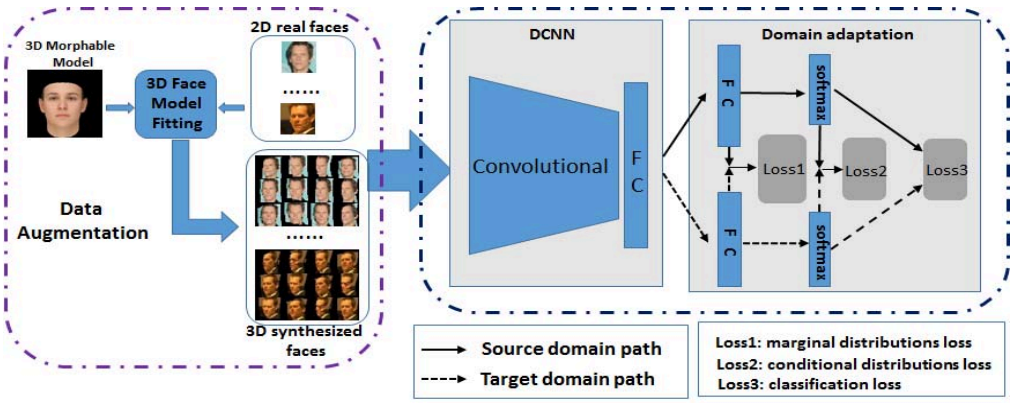
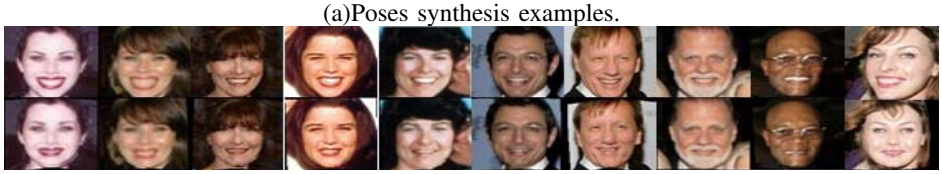


Figure 2. An overview of our approach, which learns a representation to reduce classification error and improve domain invariance simultaneously based on maximum mean discrepancy (MMD). Our approach consists of two main components: data augmentation and deep transfer network (DTN).

(a)Poses synthesis examples.



(b)Expression synthesis examples. First row: Original face images. Second row: Synthetic images with natural expression

(c)Examples render failures due to strong facial expressions

(d)Examples render failures due to poor facial landmark localization

Figure 3. Qualitative rendered faces

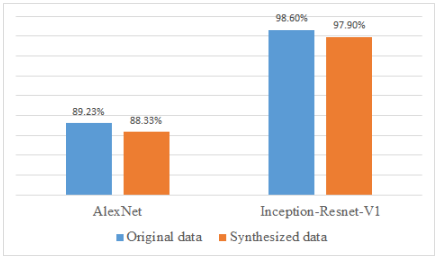
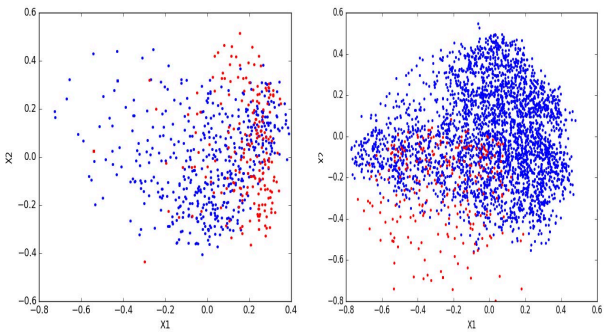


Figure 4. The performances of networks trained by using source domain and target domain data.

to enhance the performance of our model. Our goal is that the 3D synthesized data properties are preserved and the data distributions in different domains are close to each other (marginal distribution). Furthermore, to uncover the knowledge hidden in the relations between the data labels from the source and target domains, we



(a) (b)

Figure 5. The distribution of the first two dimensions from principal component analysis (PCA) projections of the penultimate layer. Red points and blue points represent features of the original faces and the synthesized faces after PCA, respectively, and (a) and (b) represent two different people.

match the distribution of the labels given features (conditional distribution), as shown in Fig. 1.

B. Maximum Mean Discrepancy

To match the distributions, a metric of difference between two distributions needs to be defined. Suppose that we are given two sets of samples $X := \{x_i\} \in \mathbb{R}^{d\times N}$ and $Y := \{y_j\} \in \mathbb{R}^{d\times M}$, independently and identically distributed $(i.i.d)$ from p and q, respectively, and asked whether the generating distributions $p = q$. Many criteria exist that can be used to estimate the distance. MMD is a commonly used metric of discrepancy of two distributions due to its efficiency in computation and optimization [25]. Most MMD-based approaches can be viewed as minimizing a certain distance between the weighted sum of all raw moments. In this paper, we adopt the empirical MMD as the conditional distributions metric (Eq. 2) and the MMD in reproducing kernel Hilbert space (RKHS) as the marginal distribution metric (Eq. 3)

$$\mathrm{MMD_b}[\mathscr{F}, X, Y] := \sup_{f\in\mathscr{F}} \left(\frac{1}{N}\sum_{i=1}^{N} f(x_i) - \frac{1}{M}\sum_{j=1}^{M} f(x_j)\right), \tag{2}$$

$$\mathrm{MMD}^2[\mathscr{F}, p, q] = \mathrm{E}_{x,x'}[k(x,x')] - 2\mathrm{E}_{x,y}[k(x,y)] + \mathrm{E}_{y,y'}[k(y,y')], \tag{3}$$

where x' is an independent copy of x with the same distribution and y' is an independent copy of y.

C. Deep Transfer Network

The entire network structure is illustrated in Fig. 2. Not only do we want to minimize the distance between real images and synthetic images (or maximize the data confusion), but we also seek to find a representation that is conducive to training strong classifiers. We choose the deep neural network to model and match both the marginal and conditional distributions, which makes our model more suitable for achieving domain transfer. We achieve this using two different types of layers in the CNN model: the shared feature extraction layer, which learns a subspace to match the marginal distributions of the source and target samples, and the discrimination layer, which matches the conditional distributions by the average labels given features. We first calculate the classification loss for all face images. Given n labeled training data $\{x_i, y_i\}$, where $y_i \in \{1, 2, ..., c\}$ represents the label with one active output node per class, the objective function of DTN in the form of the empirical log-likelihood loss function is given as

$$L(\mathrm{W}) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{c} y_k^{(i)} log([g(x^{(i)})]_k), \tag{4}$$

where $n = [n^s n^t]$ is all data including the source and target datasets. W is the projection matrix of the neural networks.

We use MMD^2_{mar} and MMD^2_{con} to represent the metrics of the marginal and conditional distributions, respectively. The definitions of MMD^2_{mar} and MMD^2_{con} are described in detail below. Therefore, our overall objective function of DTN is

$$J(\mathrm{W}) = L(\mathrm{W}) + \lambda \mathrm{MMD}^2_{mar} + \mu \mathrm{MMD}^2_{con}, \tag{5}$$

where λ and μ are regulation hyperparameters that determine how strongly the marginal distribution and the conditional distribution influence the optimization. The larger the values are, the less the difference of the data bias is. Such an optimization would enable the model to learn a strong classifier that transfers knowledge across datasets. The optimization of MMD^2_{mar} and MMD^2_{con} is similar to [29] and [24], respectively. Once the model is trained, we do not use the synthesized images at test time, which is different from [6], because using the 3D model does not guarantee that all the images in the test dataset are accurately synthesized.

1) Matching Marginal Distributions: Suppose that we choose the CNN model with l layers and that the first $l-1$ layers are all considered to be feature extraction layers. We use $\phi_{l-1}(\mathrm{x})$ to express the feature in the $l-1$th layer. Denote $P(\phi_{l-1}(\mathrm{x}))$ as the distribution of the feature space. The goal of our algorithm is to force $P(\phi_{l-1}(\mathrm{x}^s))$ and $P(\phi_{l-1}(\mathrm{x}^t))$ to be close. We use the MMD in RKHS to minimize the feature mapping function. The distance between $P(\phi_{l-1}(\mathrm{x}^s))$ and $P(\phi_{l-1}(\mathrm{x}^t))$ is modeled by the marginal MMD as follows

$$\mathrm{MMD}^2_{mar}(\mathrm{X}^s, \mathrm{X}^t) = \frac{\mathrm{Tr}(\mathrm{K}_{\mathrm{x}ss})}{(n^s)^2} + \frac{\mathrm{Tr}(\mathrm{K}_{\mathrm{x}tt})}{(n^t)^2} - 2\frac{\mathrm{Tr}(\mathrm{K}_{\mathrm{x}st})}{n^s n^t}, \tag{6}$$

where $[\mathrm{K}_{\mathrm{x}\bullet\bullet}]_{ij} = k(\phi_{l-1}(\mathrm{x}_i^\bullet), \phi_{l-1}(\mathrm{x}_j^\bullet))$. In our experiment, we choose the Gaussian RBF kernel, which is considered to be a universal approximator, with the kernel function as $k(x,y) = exp(-\frac{\|x-y\|^2}{2\delta^2})$, where δ is the bandwidth.

2) Matching Conditional Distributions: We consider the logistic regression as the model of the discrimination layer and use the softmax function to calculate the posterior probability of each category. Assume that there are a total of C categories. For an arbitrary c, the hyperplane of category c is denoted w_c. The posterior probability of $y = c$ given feature x can be written as $P(y = c|\phi_{l-1}(\mathrm{x})) = \frac{e^{w_c^T \phi_{l-1}(\mathrm{x})}}{\sum_k e^{w_k^T \phi_{l-1}(\mathrm{x})}}$. The distance between the conditional distribution of the source and target datasets is measured by the conditional MMD, defined as

$$\begin{aligned}\mathrm{MMD}^2_{con}(\mathrm{X}^s, \mathrm{X}^t) &= \sum_{c=1}^{C} \| \frac{1}{n^s}\sum_{i=1}^{n^s} P(y^s = c|\phi_{l-1}(\mathrm{x}_i^s)) \\ &\quad - \frac{1}{n^t}\sum_{j=1}^{n^t} P(y^t = c|\phi_{l-1}(\mathrm{x}_j^t))\|_2^2 \\ &= \sum_{c=1}^{C} \mathrm{Tr}(\mathrm{q}_c^T \mathrm{M} \mathrm{q}_c),\end{aligned} \tag{7}$$

where $\mathrm{q}_c \in \mathbb{R}^{n^s+n^t}$ is the posterior distribution output vector of the cth category for all data samples, and M is the MMD matrix. Let M_{ij} be one element of M. M_{ij} can be calculated as

$$M_{ij} = \begin{cases} \frac{1}{(n^s)^2} & i \leqslant n^s, j \leqslant n^t \\ \frac{1}{(n^t)^2} & i > n^s, j > n^t \\ -\frac{1}{n^s n^t} & otherwise. \end{cases} \tag{8}$$

D. Details of Deep Transfer Network

To illustrate the effectiveness of our method and ensure fast convergence of the model, it is crucial to appropriately select the model and training data.

1) Data Selection: When computing the marginal MMD and the conditional MMD, in theory, we need to take all the source and target samples into consideration. However, this approach is inefficient and not applicable because of its expensive complexity. Inspired by the idea of [24], we divide the samples into mini-batches. Rather than computing MMD over the entire datasets, we compute MMD over every single batch.

Assume that there are C subjects in the source and target domains. The batch size is set to $2S$ with half from the source domain and half from the target domain. Each batch contains N subjects, and each subject has M images. We upsample the target domain when certain subjects have fewer images than M.

2) DCNN Model Selection: We choose AlexNet [17] and Inception-ResNet-V1 [18] as our deep transfer network models. They have achieved very good performance in image recognition. The latter has a deeper structure than the former. Two classic networks were chosen to prove that the performance of our method is not accidental, while the network through transfer can achieve good recognition results on the LFW and SLLFW databases.

After the structure of the networks is determined, we can optimize the parameters of the networks using stochastic gradient descent (SGD) with backpropagation.

IV. EXPERIMENTS

A. Network Parameter Settings

All implementations are produced using the open source TensorFlow [26] framework. The initial learning rate is set to 0.01, and it is decreased by a factor of 10 when the loss flattens. We set the hyperparameters to $\lambda = \mu = 10$ throughout the training. We set the batch size as $2S = 300$, which means that every mini-batch consists of 150 source samples and 150 target samples, where $M = 15$ and $N = 10$. Why the regularization parameters and batch size need to be set to such values will be described in detail later. The bandwidths δ of the Gaussian RBF kernel are set to $\{1;2;5;10;20;40\}$. All input images are z-scored to have zero mean and unit variance. We use the output of the penultimate layer of the network as the features to perform face verification. For the two different networks, the image size and feature dimensions are shown in Table I.

Table I
INPUT AND OUTPUT OF NETWORK.

	AlexNet	**Inception-ResNet-V1**
Image size	227x227x3	160x160x3
Feature dimensions	4096	1792

B. Results on Labeled Faces in the Wild

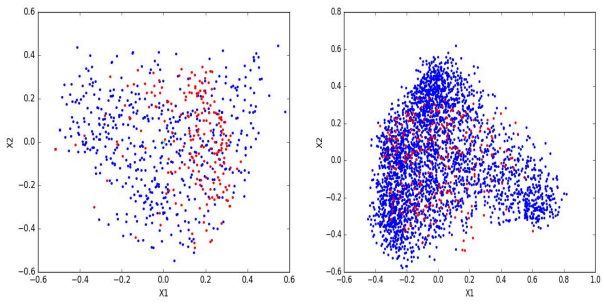
LFW was the standard benchmark for unconstrained face verification. Recent methods [2] [3] are almost reaching near-perfect performances with deep learning using millions of images. To test our approach, we follow the standard protocol for unrestricted, labeled outside data and report the mean classification accuracy.

1) Performances on Different Networks: We implemented two well-known DCNNs for comparison, namely, AlexNet and Inception-ResNet-V1. After the model is trained, we align the LFW data according to the size of the training model image and then feed it into the appropriate trained model to extract the features. The recognition rates on LFW are shown in Table II. Non-transfer represents that the model is directly trained by using the synthesized faces together with the original real faces, and DTN represents the deep transfer network used in this paper. We observe that the accuracy increases 1.22% and 0.22% using DTN compared to using no-transfer, respectively, as shown in Table II. Adding synthesized images to the original training set increases the performance by 0.97% and 0.38%. This result indicates that data augmentation using a 3D face model to synthesize virtual faces is an effective approach for face recognition. Interestingly, as shown in the second row, the performance has decreased on both networks using the synthesized face dataset rather than the real face dataset, although the synthesized dataset contains more diverse poses and a greater number of faces. This result shows that the synthesized images and real images have different distributions, and using the proposed DTN mitigates the differences between both distributions as much as possible. Moreover, the Table II shows that the accuracy rate is not enhanced too much in Inception-ResNet-V1 because the network itself has achieved a good recognition rate.

Table II
PERFORMANCES ON DIFFERENT NETWORKS

	AlexNet(%)	**Inception-ResNet-V1(%)**
OD	89.23	98.60
SD	88.33	97.90
OS+SD (non-transfer)	91.20	98.98
OS+SD (DTN)	92.42	99.20

We extract the same person's features of the original image and synthesized image from the trained DTN and then use PCA to project the output features. Fig. 6 shows the distribution of the first two dimensions of the same person's features after PCA. As shown, after using the DTN, the same person's features of synthesized images and the original images are more concentrated, and the features of the original images are fused with the features of the synthesized images.



(a) (b)

Figure 6. The distribution of the first two dimensions from PCA projections of the same person's features that are extracted from the trained DTN.

2) Comparison with the State-of-the-art: We have also compared our approach with the existing methods that have achieved a high recognition rate. Our comparison results are shown in Table III. As shown in this table, the Inception-ResNet-V1 network can reach a 98.60% accuracy rate, which is trained on the original CASIA WebFace alone. The recognition rate is higher than [1] [27] [6]. This result shows that the Inception-ResNet-V1 network is already a very effective network. We use it as the basic network to optimize our DTN by using MMD between synthesized face images and the original real face images. The DTN can achieve a 99.20% recognition rate on LFW, which improves 0.6% compared to the original Inception-ResNet-V1. The experimental results show that synthesizing training images leads to an increase in recognition accuracy, as in [6] and our results. This may be attributed to the potential of specific augmentation to infuse training data with significant intra-subject appearance variations. More importantly, performance can be further improved using DTN to mitigate the difference between both distributions. In addition, using the synthesis method also has another benefit in that it is a more accessible means of increasing training set sizes than downloading and labeling millions of additional faces. Note that the reasons why we did not used the VGG network as a basic network are as follows. First, compared to [6], our proposed method requires learning a network from scratch, which becomes quite time consuming if using VGG. Second, since basic Inception-ResNet-V1 has outperformed all improvements on VGG, performing experiments on an old baseline does not make much sense. Thus, a higher baseline is chosen in our paper to demonstrate our methods ability to improve a state-of-the-art result.

Table III
COMPARISON WITH THE EXISTING METHODS

Method	Real	Synth	Acc.(%)
DeepFace[1]	4M	-	97.35
Fusion[27]	500M	-	98.37
FaceNet + Alignment[2]	200M	-	99.63
VGG Face[3]	2.6M	-	97.27
VGG Face(triple loss)[3]	2.6M	-	98.95
no Aug.(VGG)[6]	495K	-	95.31
Aug.data(VGG)[6]	495K	2.4M	98.06
Inception-ResNet-V1	495K	-	98.60
Ours,non-transfer	495K	4.8M	98.98
Ours,Inception-ResNet-V1(DTN)	495K	4.8M	99.20

C. Results on the SLLFW Database

Deng et al. [19] realize that almost all the negative face pairs are quite easy to distinguish in the original LFW. They deliberately select 3000 similar-looking face pairs within the original image folders by human crowdsourcing to replace the random negative pairs, generating the new SLLFW database.

We use 3D a model to synthesize new face images that focus on changes within the class, which is useful for dealing with datasets such as SLLFW. We train two different networks with synthesized face images and original real face images, which are non-transfer network and DTN based on Inception-ResNet-V1. The accuracy rate using DTN improves by 1.50% compared to the non-transfer network. Fig. 7 provides ROC curves for two different methods used in our paper. The green curve represents our DTN method, and the blue curve represents the non-transfer method. Table IV shows some results compared to the existing method, whose results are provided by [19]. From the experimental results, we can note that our method achieves a state-of-the-art accuracy rate.

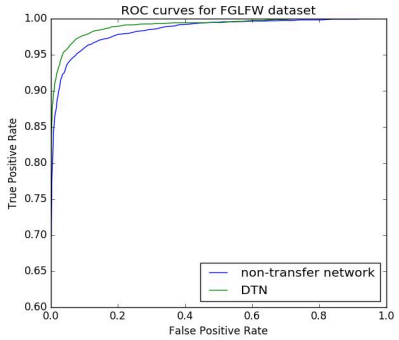


Figure 7. The ROC curves of two different methods.

Table IV
COMPARISON WITH THE EXISTING METHODS ON FGLFW

Method	Real	Synth	Acc.(%)
DeepFace[1]	0.5M	-	78.78
DeepID2[28]	0.5M	-	78.25
VGG-Face[3]	2.6M	-	85.78
DCMN[19]	0.5M	-	91.00
Ours,Inception-ResNet-V1	0.5M	4.8M	94.30
Ours,Inception-ResNet-V1 (DTN)	0.5M	4.8M	95.80

D. Parameter Sensitivity Analysis

We conduct a parameter sensitivity experiment on the LFW dataset. Distribution matching parameters λ, and μ and size of the mini-batch S are evaluated.

1) Distribution Regularization Parameter Analysis : In our paper, there are two regularization parameters λ and μ. λ controls the level of marginal matching, and μ controls the level of conditional distribution matching. Setting λ and μ too low will cause the MMD regularization to have no effect on the learned representation, but setting λ and μ too high will regularize too heavy and learn a degenerate representation in which all points are too close together. Fig. 8(a)(b) show the recognition results when λ and μ take different values from $\{0.1, 1, 10, 20, 50, 100\}$. As shown in Fig. 8(a)(b), λ and μ show the same trend as the theoretical analysis. For simplicity, we set $\lambda = \mu = 10$ in all of our all experiments. 'lambda' and 'mu' denote λ and μ in Fig. 8(a)(b), respectively.

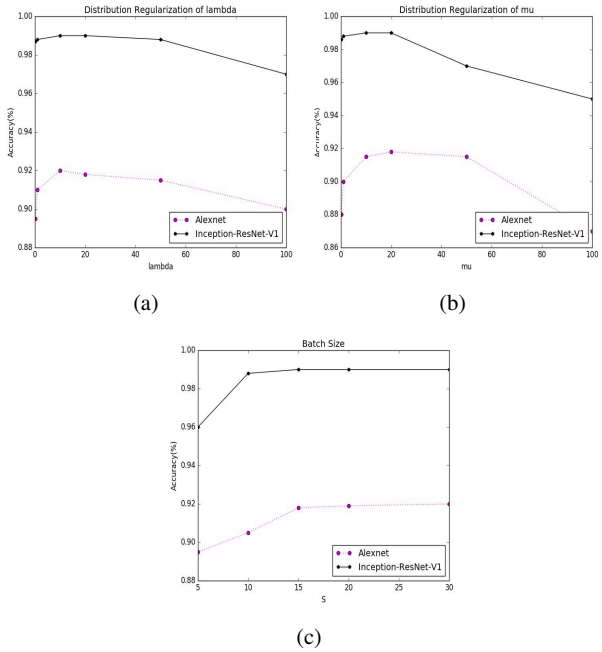


Figure 8. Distribution regularization parameter analysis. (a)The effect of the size of λ on the recognition result. (b)The effect of the size of μ on the recognition result.(c)The effect of batch size on the recognition rate

2) Batch Size: The mini-batch is the basic unit for evaluating the distribution of the database and optimizing the objective function. The size S should be large sufficient to contain enough samples in the batch such that it can reflect the distribution of the entire dataset. For simplicity, the number of people that we choose every time is fixed; we only change the number of pictures per subject. We set $N = 10$ in all of our all experiments. Fig. 8(c) shows the recognition accuracies when M takes different values from $\{5, 10, 15, 20, 30\}$. We observe that the larger batch always leads to better performance. However, the size of the batch should not be too large because of the limitation of GPU memory. Therefore, we set $S = 150 (N = 10, M = 15)$, which can not only achieve a good recognition rate, but also meet the needs of GPU memory.

V. CONCLUSIONS

In this paper, we mainly elaborate on two issues. First, we synthesize face images with various poses and natural expressions using the 3DMM method. We have experimentally demonstrated that there is a dataset bias between synthesized faces and real faces. Second, we address the dataset bias between synthesized

data and real data with the help of DNT. DTN combines the best paradigms in object recognition (neural network) and domain adaptation (matching the marginal and conditional distributions of different domains). Our experiments show that the dataset bias between the synthesized data and the real data is remarkably reduced, which can thus improve the performance of the CNN model.

ACKNOWLEDGMENT

This work was partially supported by the National Natural Science Foundation of China under Grant Nos. 61573068, 61471048, and 61375031, and by the Beijing Nova Program under Grant No. Z161100004916088.

REFERENCES

[1] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, pp. 1701–1708, 2014. 1, 5, 6

[2] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pp. 815–823, 2015. 1, 5, 6

[3] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *BMVC*, pp. 41.1–41.12, 2015. 1, 5, 6

[4] E. Zhou, Z. Cao, and Q. Yin, Naive-deep face recognition: Touching the limit of lfw benchmark or not?. *Computer Science*, 2015. 1

[5] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *Computer Science*, 2014. 1

[6] I. Masi, A. T. Tran, T. Hassner, J. T. Leksut, and G. Medioni. *Do We Really Need to Collect Millions of Faces for Effective Face Recognition?* In *ECCV*, pp. 579-596, 2016. 1, 2, 4, 5, 6

[7] T. Hassner. Viewing real-world faces in 3d. In *ICCV*, pp. 3607–3614, 2013. 1

[8] T. Hassner, S. Harel, E. Paz, and R. Enbar. Effective face frontalization in unconstrained images. In *CVPR*, pp. 4295–4304, 2015. 1

[9] Z. Wu, W. Deng, and Z. An. Illumination-recovered pose normalization for unconstrained face recognition. In *ACCV*, pp. 217–233, 2016. 1

[10] Masi, I., Hassner, T., Trn, A.T., Medioni, G.: Rapid synthesis of massive face sets for improved face recognition. In *FG*, pp. 604–611, 2017. 1

[11] V. Blanz and T. Vetter. Face recognition based on fitting a 3d morphable model. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 25(9):1063–1074, 2003. 2

[12] L. MWang, X. Yu, and D.N.,Metaxas, A Coupled Encoder-Decoder Network for Joint Face Detection and Landmark Localization. In *FG*, pp. 251-257, 2017. 2

[13] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter, A 3d face model for pose and illumination invariant face recognition. In *AVSS*, pp. 296–301, 2009. 2

[14] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou, Facewarehouse: a 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, pp. 1, 2013. 2

[15] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. *Month*, 2008. 2

[16] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schlkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(1):723–773, 2012. 2

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pp. 1097–1105, 2012. 2, 5

[18] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *ICLR -Workshop*, 2016. 2, 5

[19] W. Deng, J. Hu, N. Zhang, B. Chen, and J. Guo. Fine-grained face verification: Fglfw database, baselines, and human-dcmn partnership. *Pattern Recognition*, 66:63–73, 2017. 2, 6

[20] A. Lapedriza, D. Masip, and J. Vitria. Are external face features useful for automatic face classification? In *CVPR - Workshops*, pp. 151, 2005. 2

[21] X. Zhu, Z. Lei, J. Yan, D. Yi, and S. Z. Li, High-fidelity pose and expression normalization for face recognition in the wild. In *CVPR*, pp. 787–796, 2015. 2

[22] T. Baltrusaitis, P. Robinson, and L. P. Morency. Openface: An open source facial behavior analysis toolkit. In *WACV*, pp. 1–10, 2016.

[23] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR*, pp. 1521–1528, 2011. 2

[24] X. Zhang, F. X. Yu, S. F. Chang, and S. Wang. Deep transfer network: Unsupervised domain adaptation. *Computer Science*, 2015. 4

[25] Quadrianto, N., Petterson, J., Smola, A.J.: Distribution matching for transduction. In *NIPS*, pp. 1500–1508, 2015. 4

[26] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, and M. Devin. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint,* arXiv:1603.04467, 2016. 5

[27] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Web-scale training for face identification. In *CVPR*, pp. 2746–2754, 2015. 5, 6

[28] Y. Sun, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *NIPS*, pp. 1988–1996, 2014. 6

[29] Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. Computer Science, pp. 97–105, 2015. 4