R^3 Adversarial Network for Cross Model Face Recognition

Ken Chen* Yichao Wu* Haoyu Qin Ding Liang Xuebo Liu Junjie Yan
Sensetime Group Limited
kenchen1024@gmail.com {wuyichao,qinhaoyu,liangding,liuxuebo,yanjunjie}@sensetime.com

Abstract

In this paper, we raise a new problem, namely cross model face recognition (CMFR), which has considerable economic and social significance. The core of this problem is to make features extracted from different models comparable. However, the diversity, mainly caused by different application scenarios, frequent version updating, and all sorts of service platforms, obstructs interaction among different models and poses a great challenge. To solve this problem, from the perspective of Bayesian modelling, we propose R^3 Adversarial Network (R^3AN) which consists of three paths: Reconstruction, Representation and Regression. We also introduce adversarial learning into the reconstruction path for better performance. Comprehensive experiments on public datasets demonstrate the feasibility of interaction among different models with the proposed framework. When updating the gallery, R^3AN conducts the feature transformation nearly 10 times faster than ResNet-101. Meanwhile, the transformed feature distribution is very close to that of target model, and its error rate is incredibly reduced by approximately 75% compared with a naive transformation model. Furthermore, we show that face feature can be deciphered into original face image roughly by the reconstruction path, which may give valuable hints for improving the original face recognition models.

1. Introduction

Face recognition models, due to their superior performance, have been widely applied in practical applications. Currently, face recognition approaches generally learn face representations through a cascade of blocks of several processing units for feature extraction [27, 25, 28, 26, 29, 24]. The trained systems successfully obtain generalization ability by embedding the input images into a feature space, where features are clustered in a sufficiently low intra-subject variation as well as high inter-subject variation.

*Contributed equally to this work.

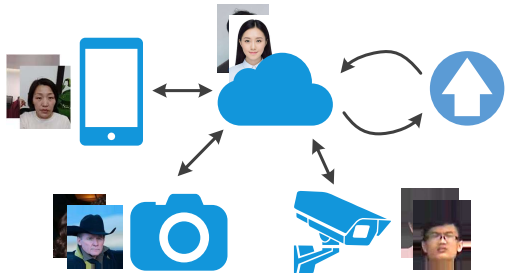


Figure 1: The application of cross model face recognition.

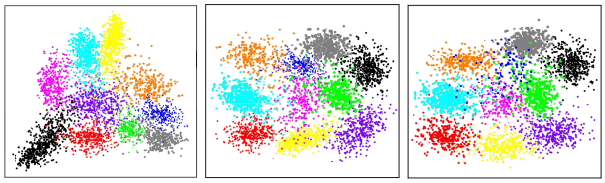
Meanwhile, pursuing interaction between information collected from various terminals is a new trend. Our daily life quality would be greatly heightened and safety in society would be ensured if practical connections among the images obtained from different scenarios, such as mobile devices, entrance guards and video surveillance, can be established, as shown in Fig. 1. This new kind of application has brought forth a new problem called cross model face recognition (CMFR), which is defined as recognizing feature extracted from one model with another model's gallery.

On the other hand, feature space is highly related to the corresponding model. As shown in Fig. 2, features learned by different systems often lie in diverse distributions, causing boundaries between different models that obstruct the interaction of their features. Therefore, as the feature distribution varies a lot, direct CMFR usually makes no sense.

Uploading all the captured face images to the server and using an unified model to extract features seems to be an alternative solution. However, following reasons reject this method:

- Single model lacks the ability to achieve satisfactory performance when various domains, applications and response time requirement are taken into account.
- It violates privacy policy in common sense. Uploading and storing user's face images are generally forbidden in industrial community.

Another solution is to model a mapping function from one model's feature space to another. However this map-



(a) (b) (c)

Figure 2: The feature distributions of two typical face recognition models.(a), (b) and (c) are feature distribution of source model, transformation model and target model, respectively.

ping function is difficult to build, especially when considering the diversity of face recognition models, including structure, parameter amount, application scenarios, and etc. Additionally, another challenge lies in the requirement that the mapping operation should be fast enough to conduct the transformation of millions of features within a short period. A slow model would be pointless as it shows no superiority comparing with extracting features via target model directly.

Based on the above discussion, we propose R^3 Adversarial Network (R^3AN) to solve this problem. R^3AN consists of three paths: reconstruction, representation and regression. Adversarial learning is introduced into the reconstruction path for better performance. To evaluate the proposed method, we conduct systematic experiments of R^3AN on a wide range of typical and efficient deep neural networks. R^3AN allows us to query feature extracted by source model in target system meaningfully, and vice versa. Comprehensive experiments on public datasets demonstrate the feasibility of interaction among different models with the proposed framework. When updating the gallery, R^3AN conducts the feature transformation nearly 10 times faster than ResNet-101 [9]. Meanwhile, the transformed feature distribution is very close to that of target model, and its error rate is incredibly reduced by approximately 75% compared with a naive transformation model.

Utilizing R^3AN to overcome such obstacles is meaningful in practical applications and reasons are listed as below. First, R^3AN enables us to break the boundaries of different models from various terminals, for example, querying in one system with the feature that is extracted from another system. Second, for the sake of privacy protection, original images may not be stored in gallery. Gathering the set of images again is prohibitive when updating models. However, R^3AN merely updates features, which avoids storing face images. Third, constructing feature gallery from raw images is time consuming when updating is required, and storing images also requires huge storage cost. In contrast, R^3AN only transforms features from old to new with little cost, and storage of features is much more efficient.

Our contribution lies in three aspects:

- We raise the CMFR problem for the first time, which possesses considerable economic and social significance.
- To address this issue, we propose R^3AN, which has the capability of transforming feature distribution of source model into its counterpart of target model. R^3AN is super fast and valid when solving this problem.
- We illustrate that face feature can be roughly decoded into original image. Adversarial learning greatly improves the performance of R^3AN and recovers higher quality face image, which may give valuable hints for improving the original face recognition models.

2. Related Works

2.1. Transfer Learning

The essential of CMFR problem is to transform the feature distribution of the source model to that of the target model, which can be considered as a typical transfer learning problem [19, 31, 12, 22, 3, 15, 18]. In many real-world applications, the training and future data are usually drawn from different distributions. Transfer learning has emerged as a learning framework to bridge the gap between them. According to [19], approaches to transfer learning can be summarized into four cases: instance-transfer approach [31, 12], feature-representation-transfer approach [22, 3], parameter-transfer approach [15], and relational-knowledge-transfer [18].

Although these works can transform the model from the source domain to the target domain and successfully boost the performance, most of them concentrate on learning a well-performed model on a different target data distribution from a trained one of source domain. Whereas in our problem, it is required to learn an effective feature transformation between two different feature spaces without affecting original distribution. Feature-representation-transfer approach [22, 3] is closest to our problem. However, it aims at finding common feature representation between two domains, different from our goal of finding a mapping from source model to target model.

2.2. Generative Adversarial Nets

Generative Adversarial Net (GAN) is firstly proposed in [6], which consists of two models: a generative model G and a discriminative model D. G manages to transform input data into a verisimilar sample, while D estimates the probability that a sample came from real world rather than generated by G. A minimax two-player game is the essential of GAN. One typical application of GAN is to transform

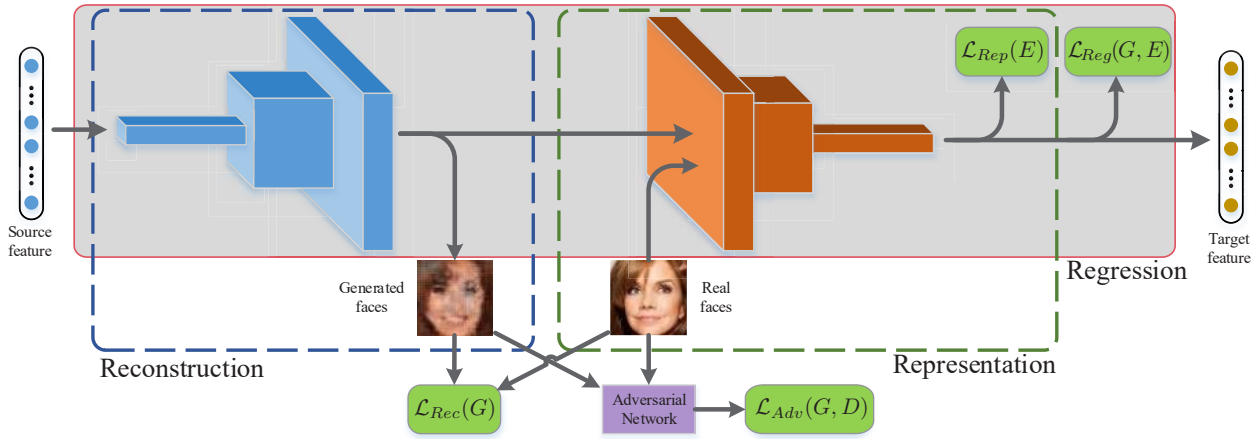


Figure 3: The architecture of R^{3}AN is shown. Reconstruction path, within the blue frame, is supervised by a L2 Loss ($\mathcal{L}_{Rec}$) and Adversarial Loss ($\mathcal{L}_{Adv}$), and transforms source features into a face image. Representation path, encompassed by green box in figure, extracts features from images, no matter from real world or generated, and a L2 Loss ($\mathcal{L}_{Rep}$). Regression path shares weights with the other two paths and a L2 Loss ($\mathcal{L}_{Reg}$) is adopted to allow the whole system to acquire the ability of transforming source features to target features.

a 1-D vector into a 2-D image. Besides, there are many other brilliant works [21, 7, 2, 17] derive from [6].

However, the input of these models are usually random vectors without unique encoding, i.e. there exists no strict one-to-one mapping constraint between the input and the output image. In our reconstruction path, we investigate the feasibility of generating the original face image from the corresponding extracted feature. Recently, Zachary et al. [16] show that generator's output can be inverted to original latent space. This research reveals the feasibility of transformation between feature and image, and gives inspiration to the design of reconstruction path in our work to some extent.

3. Rationale

3.1. Basic Model

The cross model face recognition (CMFR) problem is defined as recognizing feature extracted from one model with another model's gallery. The core to solve the CMFR problem is to make features extracted from different models comparable. Ideally, features extracted by two models are the same. In order to achieve this target, transforming the space of the source model to that of the target feature is a natural scheme. In this way, if we denote $\boldsymbol{X}$, $\boldsymbol{Y}$ as the source and target feature, the goal of our system is to find a mapping function to maximize the conditional probability $P(\boldsymbol{Y}|\boldsymbol{X})$. From this perspective, we could map $\boldsymbol{X}$ to $\boldsymbol{Y}$ in one-dimensional space directly by a naive model like a multi-layer perception (MLP). However, this basic model yields unsatisfactory results, which will be illustrated in Section 5.

3.2. Bayesian based Model

We further model our problem from the view of Bayesian, to make it more reasonable and effective. By introducing a class of latent variables $h \in H = \{h_1, ...h_K\}$, $P(\boldsymbol{Y}|\boldsymbol{X})$ can be expressed as follows:

$$P(\boldsymbol{Y}|\boldsymbol{X}) = \sum_{h \in H} P(h|\boldsymbol{X}) P(\boldsymbol{Y}|\boldsymbol{X}, h). \tag{1}$$

From the head-to-tail view of Bayesian modeling, $\boldsymbol{X}$ and $\boldsymbol{Y}$ is conditionally independent when h is given. Therefore, Eq. 1 can be simplified as:

$$P(\boldsymbol{Y}|\boldsymbol{X}) = \sum_{h \in H} P(h|\boldsymbol{X}) P(\boldsymbol{Y}|h). \tag{2}$$

In our problem, h should be a latent variable independent of models. So, the original image $\boldsymbol{I}$ is a good choice. Then, Eq. 2 can be rewritten as:

$$P(\boldsymbol{Y}|\boldsymbol{X}) = P(\boldsymbol{I}|\boldsymbol{X}) P(\boldsymbol{Y}|\boldsymbol{I}) \tag{3}$$

Based on the theorem above, we propose a novel framework, R^{3}AN, to transform face representations between different feature spaces.

4. R^3 Adversarial Network

In order to keep consistency with Eq. 3, R^{3}AN is designed to consists of three paths, i.e., *Reconstruction*, *Representation* and *Regression* respectively, as shown in Fig. 3. *Reconstruction* corresponds to $P(\boldsymbol{I}|\boldsymbol{X})$ in Eq. 3, which is used to recover the original image, while *Representation*

corresponds to $P(\boldsymbol{Y}|\boldsymbol{I})$, which is adopted to extract feature from latent face images. *Regression* views reconstruction and representation as a unified problem, and is used to jointly optimize the whole model. In the following, we will elaborate the framework.

4.1. Reconstruction Path

In order to maximize the conditional probability $P(\boldsymbol{I}|\boldsymbol{X})$ in Eq. 3, we design a reconstruction path to recover the original images. As this module aims to restore original face images from extracted features, it can be regarded as a generator (G).

For fast transformation, we adopt a lightweight architecture based on the fractional-strided convolutions, as shown in the upper part of Tab. 1. The topology is inspired from the one in [21]. First, the normalized high level representation is reshaped into a tensor of 4 dimensions, which are batch, channel, height and width, respectively. In other words, normalized input feature would be copied into the channel dimension of a tensor, whose height and width dimensions are left to be 1. Then a cascade of fractional-strided convolution layers upsample this high level representation to generate a restored face image.

The reconstruction path is optimized by the ground truth of corresponding face images. There are two modes of reconstruction mode in this paper, named as naive reconstruction and adversarial reconstruction.

Naive Reconstruction In order to recover the original face images as real as possible, we evaluate similarity of the generated face images and the real ones in a traditional manner by L2 loss. So, the reconstruction loss can be formulated as follows:

$$\mathcal{L}_{Rec}(G) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{I}}\left[||\boldsymbol{I} - G\left(\boldsymbol{X}\right)||_2\right], \tag{4}$$

However, as we all know, it is tough to recover an image with sufficient detail from small-scale feature because much trivial information is dropped during the representation learning process. Thus, demanding the generated image to be as close to the real ones as possible may result in a blurry averaged image, which lacks of many significant details of face. To alleviate this difficult problem, we could resort to adversarial reconstruction.

Adversarial Reconstruction In face representation learning task, the identity knowledge is embedded in the distribution of images. To recover these primary information, we enhance the reconstruction path by introducing adversarial learning. The method of adversarial learning, first proposed in [6], is powerful to model complex data distributions. It is applied to learn the distribution of images from uniform distribution in [21] and transform pictures to a specific distribution in [13]. Inspired by these works, we adopt this method to encourage the generator G to learn the particular characteristics of face images from feature representation. In this work, we construct a discriminator D, whose structure is shown in Tab. 1, to distinguish the generated images from real faces, while G tries to confuse D by producing images with higher quality. Learning in this case, the loss function for adversarial learning is logistic likelihood:

Module	Layer	Operator	Output Size
Generator	Input	–	256× 1× 1
	fConv1	ConvTranspose2d	512× 4× 4
	fConv2	ConvTranspose2d	128× 7× 7
	fConv3	ConvTranspose2d	32×14×14
	fConv4	ConvTranspose2d	8×28×28
	fConv5	ConvTranspose2d	3×56×56
Extractor	Conv1	bottleneck	32×28×28
	Conv2	bottleneck	64×14×14
	Conv3	bottleneck	96×14×14
	Conv4	bottleneck	160× 7× 7
	Conv5	bottleneck	320× 7× 7
	Conv6	Conv2d 1×1	1280× 7× 7
	Pooling	AvgPool2d	1280× 1× 1
	FC	Fully Connection	256× 1× 1
Discriminator	Conv1	Conv2d	4×28×28
	Conv2	Conv2d	8×14×14
	Conv3	Conv2d	16×7×7
	Conv4	Conv2d	32×4×4
	Conv5	Conv2d	64×2×2
	Conv6	Conv2d	128×1×1
	FC	Fully Connection	1

Table 1: Architerture of the generator, extractor and discriminator. The output sizes are described in $channels \times height \times width$. The *ConvTranspose2d* means fractional-strided convolution, and *bottleneck* represents convolutional bottleneck block in [23].

$$\begin{aligned}\mathcal{L}_D(G,D) =& \mathbb{E}_{\boldsymbol{I}}\left[\log\left(D\left(\boldsymbol{I}\right)\right)\right] \\ &+ \mathbb{E}_{\boldsymbol{X}}\left[\log\left(1 - D\left(G\left(\boldsymbol{X}\right)\right)\right)\right] \qquad (5)\\ \mathcal{L}_{Adv}(G,D) =& -\mathbb{E}_{\boldsymbol{X}}\left[\log\left(1 - D\left(G\left(\boldsymbol{X}\right)\right)\right)\right]. \qquad (6)\end{aligned}$$

By merging the Eq. 4 and 6, the overall loss is

$$\mathcal{L}_G(G,D) = \lambda_{Rec}\mathcal{L}_{Rec}(G) + \lambda_{Adv}\mathcal{L}_{Adv}(G,D). \tag{7}$$

4.2. Representation Path

Representation path is used to maximize $P(\boldsymbol{Y}|\boldsymbol{I})$ of Eq. 3. In the ideal case, the reconstructed images are infinitely close to the real images. Therefore, based on this assumption, representation path acts as an feature extractor,

denoted as E. It takes the original face images as input and learns representation of the target model.

Taking the time costs and computational resources into account, we adopt the convolution-based inverted residual structure [23]. Knowledge is gathered through stack of convolutional blocks and the final high-level representation of input face image is formed at the top of the extraction module, as shown in the lower part of Tab. 1.

As $\boldsymbol{Y}$ denotes the feature of the target model. The representation path can be also considered as a knowledge distiller, which can transfer the knowledge of the target model to the feature extractor module. For a teacher-student based knowledge distillation framework, many sophisticated loss functions have been proposed [10, 1]. Considering the generalization of our method, we adopt the simple L2 distance loss to supervise the training process of this module as follows:

$$\mathcal{L}_{Rep}(E) = \mathbb{E}_{\boldsymbol{I},\boldsymbol{Y}}\left[||\boldsymbol{Y} - E\left(\boldsymbol{I}\right)||_2\right]. \tag{8}$$

4.3. Regression Path

Practically, the original face image $\boldsymbol{I}$, as a latent variable of Eq. 3, can not be fully recovered by G. Therefore, it is necessary to synchronize the G and E in our feature-to-feature learning framework. The regression path, shown in Fig. 3, combines reconstruction and representation together into a unified framework, and is used to jointly optimize the above two path. Since the ultimate problem to be settled is to map one feature to another, the regression loss exists in a simple form of L2 distance and can be expressed as:

$$\mathcal{L}_{Reg}(G, E) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[||\boldsymbol{Y} - E\left(G\left(\boldsymbol{X}\right)\right)||_2\right]. \tag{9}$$

4.4. Optimization

Considering all of above, the final optimization goal of the whole system is:

$$\begin{aligned}(G^*,E^*) = \arg\min_{G,E}\max_{D}[&\lambda_{Rec}\mathcal{L}_{Rec}(G) + \lambda_{Adv}\mathcal{L}_{Adv}(G, D) \\ &+ \lambda_{Reg}\mathcal{L}_{Reg}(G, E)] + \lambda_{Rep}\mathcal{L}_{Rep}(E).\end{aligned} \tag{10}$$

For better performance, we optimize the framework in an iterative process, as shown in Algorithm 1. The training procedure can be divided into three stages. First, with joint loss of naive and adversarial reconstruction, G is optimized by playing the minimax game with D. Then, we train the E via a typical representation learning process, while taking real images $\boldsymbol{I}$ as input and target feature $\boldsymbol{Y}$ as ground truth. Finally, we take advantage of joint training to complete the global optimization for G and E. To be emphasized, after the system optimized, the G and E can be later reused for representation mapping.

Algorithm 1 R^{3}AN Optimization

Input: Dataset $(\boldsymbol{x}, \boldsymbol{y}) \in (\boldsymbol{X}, \boldsymbol{Y}), \boldsymbol{i} \in \boldsymbol{I}$, randomly initialized the generator G , extractor E and discriminator D parameterized by $\theta_g, \theta_e, \theta_d$

Output: the optimized G, E and D parameterized by $\hat{\theta}_g, \hat{\theta}_e, \hat{\theta}_d$

Random initialize G, E and D
repeat
  for number of training epochs **do**
    for number of mini-batches **do**
      // for discriminator D
      $\theta_d \leftarrow \theta_d - \mu \frac{\partial \mathcal{L}_D(\boldsymbol{X},\boldsymbol{I};\theta_g,\theta_d)}{\partial \theta_d}$
      // for generator G
      $\theta_g \leftarrow \theta_g - \mu \frac{\partial \mathcal{L}_G(\boldsymbol{X},\boldsymbol{I};\theta_g,\theta_d)}{\partial \theta_g}$
      // for extractor E
      $\theta_e \leftarrow \theta_e - \mu \frac{\partial \mathcal{L}_{Rep}(\boldsymbol{I},\boldsymbol{Y};\theta_e)}{\partial \theta_e}$
      // for generator G and extractor E
      $(\theta_g, \theta_e) \leftarrow (\theta_g, \theta_e) - \mu \frac{\partial \mathcal{L}_{Reg}(\boldsymbol{X},\boldsymbol{Y};\theta_g,\theta_e)}{\partial (\theta_g,\theta_e)}$
    end for
  end for
until convergence, got $\hat{\theta}_g = \theta_g$, $\hat{\theta}_e = \theta_e$, $\hat{\theta}_d = \theta_d$
return $\hat{\boldsymbol{\theta}}_{\boldsymbol{g}}, \hat{\boldsymbol{\theta}}_{\boldsymbol{e}}, \hat{\boldsymbol{\theta}}_{\boldsymbol{d}}$

5. Experiments

To confirm the advantages of our framework, we design experiments on CMFR, where probe features from one model are taken to query in the gallery features from another model by a one-to-many mode. We first train several typical networks to learn face representation as prior models. Then we set pairs of source-target models from the prior models and train the transformation model to break the boundary within those pairs. Finally, we evaluate our R^{3}AN in various conditions of sub-modules, prior models and datasets to systematically study its performance. All experiments are implemented on the platform of PyTorch [20] with batch size 512 for all cases. Besides, we test the speed of R^{3}AN on Nvidia Tesla V100 and find it is 10 times faster than ResNet-101.

5.1. Experiments on Prior Models

We establish baselines of face recognition on several advanced and effective architectures [9, 32, 11, 23], as prior models, based on the criterion of ArcFace [5] with $m = 0.5$. All input RGB face images are cropped in the size of 110×110 and resized to 224×224, except for PolyNet as 235 × 235, while each pixel is normalized to $[-1.6, 1.6]$. Outputs before the last classifier, which are taken as face representation for the following CMFR experiments, keep the dimension of 256. The learning rate is started from 0.1 and divided by 10 at the 100k, 140k, 160k iterations,

while momentum is 0.9 and weight decay is $5e^{-4}$. The training is terminated at the iteration of 200k. Based on the above settings, we implement experiments on training datasets of MS-Celeb-1M [8] and VGG2-Face [4] and evaluate on MegaFace [14]. Finally, we evaluate models' identification capability by querying between their own probe and gallery feature sets. The results of evaluation are presented in Tab. 2.

Networks	Abbreviation	Top1 Acc
MobileNetV2(T=6) [23]	Mb-6	92.84
MobileNetV2(T=10) [23]	Mb-10	93.94
MobileNetV2(T=16) [23]	Mb-16	94.29
ResNet-50 [9]	Res50	97.48
ResNet-101 [9]	Res101	98.12
DenseNet121 [11]	Dns121	97.45
DenseNet161 [11]	Dns161	97.70
PolyNetE [32]	Poly	98.46

Table 2: Identification results of different models on MegaFace dataset. 'Top1 Acc' refers to the top-1 face identification accuracy rate with 1M distractors.

5.2. Experiments on Cross Model Face Recognition

To explore the capability for boundary breaking of the proposed R^{3}AN framework, we implement CMFR experiments between prior models. First, we select a pair of prior models as source and target. Meanwhile, their face representations are taken as input and ground truth, respectively. Besides, the corresponding original face images are taken as supervised signals to guide the reconstruction path. However, different from input images of prior models, these images are resized to 56×56 to make it easier for reconstruction. Then, we train the R^{3}AN with learning rate starting from 0.08 and divided by 10 at the 60k, 100k ,140k iterations. The training, with momentum of 0.9 and weight decay of $1e^{-3}$, is terminated at the iteration of 200k. Finally, we evaluate the performance by taking R^{3}AN to transform the probe set of source model to a new distribution and query in the gallery set of target model, if not specifically stated.

Effects on different paths in R^{3}AN Since there are 3 paths in architecture of a complete R^{3}AN, we need to verify the effectiveness of paths. Taking MobileNetV2 as source model and ResNet-101 as target model, we design different topologies with various combination of paths to conduct the feature transformation. The results on CMFR experiment are listed in Tab. 3. First, we find that the performance of 'Arch3' is much better than both 'Arch1' and 'Arch2', which are composed of reconstruction and representation paths. This phenomenon shows the necessity of synchronizing the other two paths, as the original face image cannot be completely recoverd. Second, it can be seen that by introducing naive reconstruction path to 'Arch4' is slightly elevated. By further introducing the representation path, the accuracy is heightened again, reaching 94.93% at this time ('Arch5'). Moreover, given the great success of adversarial learning, we integrate it into the framework, and this combination increases the top1 accuracy to 94.80% ('Arch6' in Tab. 3). Finally, when all the three paths are employed, R^{3}AN's capability reaches its summit in our experiments. The highest top1 accuracy of R^{3}AN is 95.97%, outperforming all the architetures mentioned above.

Architecture	Rec Adv	Rec L2	Rep L2	Reg L2	Top1 Accuracy
FC	×	×	×	×	83.92
Arch1	×	✓	✓	×	85.65
Arch2	✓	×	✓	×	83.41
Arch3	×	×	×	✓	94.05
Arch4	×	✓	×	✓	94.21
Arch5	×	✓	✓	✓	94.93
Arch6	✓	×	×	✓	94.80
Arch7	×	×	✓	✓	NAN
R^{3}AN	✓	✓	✓	✓	**95.97**

Table 3: Identification results of CMFR between MobileNetV2 (T=6) and ResNet-101 based on different architectures. Each row in this table is an architecture, and each column means a specific training process. The '✓' and '×' represent for whether the architecture contains the process or not. 'Rec:Adv' and 'Rec:L2' means optimizing the generator by adversarial loss or L2 loss; 'Rep:L2' is extractor's optimization; 'Reg:L2' represents the regression path. 'Top1 Accuracy' refers to the top-1 face identification accuracy rate with 1M distractors.

It should be mentioned that we find that the network is hard to converge ('Arch7'), if we only adopt representation and regression path in the system. We conjecture that the distributions learned by the two parts are quite different if there is no constrains on the mid-level supervision of original images. Therefore, it is of great importance to integrate the reconstruction path into the system.

Comparison with naive model As mentioned in Section 3.1, naive transformation model such as MLP can be used conduct feature transformation as well. Results of an MLP (FC) with equivalent number of parameters to R^{3}AN is shown in Tab. 3. The comparison between them evidently verify the superiority of R^{3}AN, which can reduce the error rate by 75%.

Performance on different prior models Source and target prior model pairs can be mainly divided into four categories by model size, which are small-to-small, small-to-

Src	Tgt	Training Set	Src	Tgt	FC|Tgt	R^3AN|Tgt
Mb-6	Mb-16	MS1M&VGG2	92.84	94.29	83.54	94.18
		IMDB	92.84	94.29	80.02	93.78
Mb-6	Res101	MS1M&VGG2	92.84	98.12	83.92	95.97
		IMDB	92.84	98.12	81.57	94.84
Poly	Mb-6	MS1M&VGG2	98.46	92.84	88.51	98.34
		IMDB	98.46	92.84	84.66	98.13
Res50	Poly	MS1M&VGG2	97.48	98.46	87.24	98.29
		IMDB	97.48	98.46	83.71	98.16

Table 4: Results of CMFR with R^3AN trained on extra dataset. We use proposed models to map distribution of 'Src' (source model) to the distribution 'Tgt' (target model). The evaluation is established by taking the learned representation from the left model of '|' as probe and right model's output as gallery. 'FC' and 'R^3AN' refers to the architecture in the first and last row of Tab. 3. Results are the top-1 face identification accuracy rates with 1M distractors.

large, large-to-small and large-to-large, as shown in Tab. 5. In order to evaluate the effectiveness and generalization of R^3AN, we designed a series of experiments on different cases. First, with the help of three paths, R^3AN performs better than base architecture in all cases, and average improvements from base model are 10.50%, 12.45%, 9.99%, and 9.58% respectively. Second, when querying generated features in target gallery, the results are not inferior to the results of querying features generated by target model. The superiority is obvious on all pairs of source and target models regardless their scales, confirming the generalization ability of R^3AN.

The second and the last parts in Tab. 5 are quite close to practical application, so it is worthwhile to elaborate them. The first part matches the scenario of cloud-query. Transforming the feature on local devices with R^3AN and querying on the cloud is able to decrease 64% error rate at most, compared with directly querying on local devices. On the other hand, the last part imitates large models updating. In this case, R^3AN performs well that the results of transformation are lower than the results of normal model updating with only 0.03% on average. Besides, we find that huge structure difference between source model and target model, e.g. ResNet-50 and PolyNetE, never negatively affect the performance of R^3AN.

Effects on the domain of training samples We discuss whether R^3AN is senstitive to the domain of training samples. We train R^3AN with IMDb-Face [30] dataset rather than MS-Celeb-1M and VGG2-Face and evaluate in the same way as the above experiment. From the results in Tab. 4, we can see that though trained on datasets of different domains, R^3AN can also perform well in CMFR. This conclusion demonstrate that it is convenient for offsite training with our method.

Feasibility of gallery transformation Except for the transformation of probe sets, gallery updating also has broad application and practical value. Therefore, we evaluate R^3AN on CMFR experiments with gallery distribution transformed to that of the target model. From Tab. 6, we can see that R^3AN can even achieve higher accuracy than target models by about 1.3% when transform PolyNetE to MobileNetV2. And for the other three experiments, R^3AN can transform the source model to the target with almost no accuracy decline. Considering that the gallery usually contains millioins of people, it would take much less time to update the gallery with R^3AN (10 times faster than ResNet-101), compared with the original target model.

Src	Tgt	Src	Tgt	FC|Tgt	R^3AN|Tgt
Mb-6	Mb-10	92.84	93.94	83.66	94.36
Mb-10	Mb-6	93.94	92.84	84.17	94.69
Mb-6	Mb-16	92.84	94.29	83.54	94.18
Mb-10	Mb-16	93.94	94.29	84.19	94.33
Mb-6	Res50	92.84	97.48	83.78	94.48
Mb-6	Res101	92.84	98.12	83.92	95.97
Mb-6	Poly	92.84	98.46	82.92	97.44
Mb-6	Dns161	92.84	97.70	82.95	95.66
Res50	Mb-6	97.48	92.84	87.49	97.60
Res101	Mb-6	98.12	92.84	88.17	98.19
Poly	Mb-6	98.46	92.84	88.51	98.34
Dns161	Mb-6	97.70	92.84	87.63	97.64
Res50	Res101	97.48	98.12	89.29	97.69
Res101	Res50	98.12	97.48	89.38	97.86
Dns121	Dns161	97.45	97.70	87.41	97.81
Res50	Poly	97.48	98.46	87.24	98.29

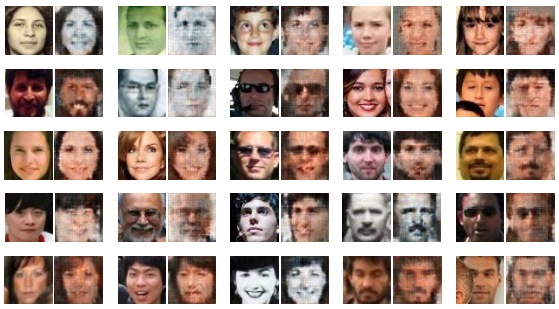
Table 5: Results of CMFR among different prior models. We use proposed models to map distribution of 'Src' (source model) to the distribution 'Tgt' (target model). The evaluation is established by taking the learned representation from the left model of '|' as probe and right model's output as gallery. 'FC' and 'R^3AN' refers to the architecture in the first and last row of Tab. 3. Results are the top-1 face identification accuracy rates with 1M distractors.

Src	Tgt	Src	Tgt	Tgt|FC	Tgt|R^{3}AN
Mb-6	Mb-16	92.84	94.29	84.17	94.23
Mb-6	Res101	92.84	98.12	88.38	97.37
Poly	Mb-6	98.46	92.84	80.79	94.16
Res50	Poly	97.48	98.46	88.67	98.40

Table 6: Results of CMFR with gallery transformation. We use proposed models to map distribution of 'Source Model' to the distribution 'Target Model'. The evaluation is established by taking the learned representation from the left model of '|' as probe and right model's output as gallery. 'FC' and 'R^{3}AN' refers to the architecture in the first and last row of Tab. 3. Results are the top-1 face identification accuracy rates with 1M distractors.

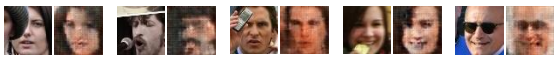
5.3. Visualization

In addition to numerical results, we also display the pair of real images and reconstructed face images from generator in R^{3}AN, as shown in Fig. 4. From Fig. 4a, it can be seen that the generator can almost recover the original face from specific feature. Though the generated images may be blurry, they contain plenty of knowledge for face identification. For the same identity, though face photos differ in pose, brightness, expression and hair style, we can generate almost same images, as shown in Fig. 4b. To our surprise, except for retaining essential face representations the generator can even remove interference information, such as the images in Fig. 4c. Besides, though images in Fig. 4d are in abnormal illumination, generated images are adjusted to a natural hue.

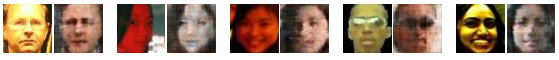


(a) Normal

(b) Same person



(c) Occlusion



(d) Lighting

Figure 4: The visualization of real faces and generated images from generator. Real faces are on the left, and generated images are on the right.

Furthermore, to visualize the feature distribution of source, target and transformation models, we randomly select 10 classes of subjects and plot their feature in 2D plain. In Fig. 5, source, transformation and target model are plotted from left to right in each rows. It can be seen, the R^{3}AN maps feature distribution from source model to the target model.

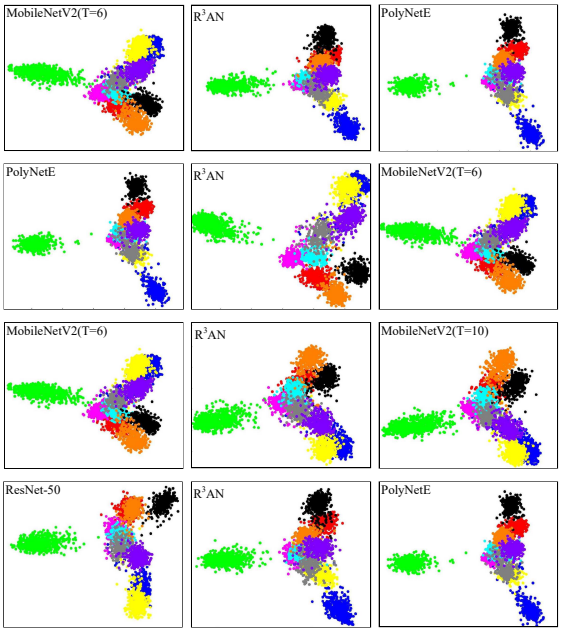


Figure 5: The visualization of feature distribution. From left to right in each row, images of distribution are from source model, R^{3}AN and target model, respectively.

6. Conclusion

In this paper, we raise a new challenging problem called cross model face recognition (CMFR), which is defined as making features extracted from different models comparable. To solve this problem, from the perspective of Bayesian modelling, we propose R^3 Adversarial Network, which can transform the feature distribution of source model to that of target model. Experimental results on public datasets demonstrate the feasibility of interaction between different models.

References

[1] Vasileios Belagiannis, Azade Farshad, and Fabio Galasso. Adversarial network compression. *arXiv:1803.10750*, 2018.

[2] David Berthelot, Thomas Schumm, and Luke Metz. Began: boundary equilibrium generative adversarial networks. *arXiv:1703.10717*, 2017.

[3] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, pages 440–447, 2007.

[4] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *FG*, pages 67–74, 2018.

[5] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *arXiv:1801.07698*, 2018.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014.

[7] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NIPS*, pages 5767–5777, 2017.

[8] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, pages 87–102, 2016.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015.

[11] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

[12] Jiayuan Huang, Arthur Gretton, Karsten M Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *NIPS*, pages 601–608, 2007.

[13] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[14] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *CVPR*, pages 4873–4882, 2016.

[15] Neil D Lawrence and John C Platt. Learning to learn with the informative vector machine. In *ICML*, page 65, 2004.

[16] Zachary C Lipton and Subarna Tripathi. Precise recovery of latent vectors from generative adversarial networks. 2017.

[17] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, pages 2813–2821, 2017.

[18] Lilyana Mihalkova, Tuyen Huynh, and Raymond J Mooney. Mapping and revising markov logic networks for transfer learning. In *AAAI*, pages 608–614, 2007.

[19] Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Trans. Knowledge and Data Engineering*, (10):1345–1359, 2010.

[20] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[21] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *arXiv:1511.06434*, 2015.

[22] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. Self-taught learning: transfer learning from unlabeled data. In *ICML*, pages 759–766, 2007.

[23] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018.

[24] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015.

[25] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *NIPS*, pages 1988–1996, 2014.

[26] Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. Deepid3: Face recognition with very deep neural networks. In *arXiv:1502.00873*, 2015.

[27] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *CVPR*, pages 1891–1898, 2014.

[28] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deeply learned face representations are sparse, selective, and robust. In *CVPR*, pages 2892–2900, 2015.

[29] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, pages 1701–1708, 2014.

[30] Fei Wang, Liren Chen, Cheng Li, Shiyao Huang, Yanjie Chen, Chen Qian, and Chen Change Loy. The devil of face recognition is in the noise. In *ECCV*, 2018.

[31] Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *ICML*, page 114, 2004.

[32] Xingcheng Zhang, Zhizhong Li, Chen Change Loy, and Dahua Lin. Polynet: A pursuit of structural diversity in very deep networks. In *CVPR*, pages 3900–3908, 2017.