Towards Open-Set Identity Preserving Face Synthesis

Jianmin Bao[1], Dong Chen[2], Fang Wen[2], Houqiang Li[1], Gang Hua[2]
[1]University of Science and Technology of China [2]Microsoft Research
jmbao@mail.ustc.edu.cn {doch, fangwen, ganghua}@microsoft.com lihq@ustc.edu.cn

Abstract

We propose a framework based on Generative Adversarial Networks to disentangle the identity and attributes of faces, such that we can conveniently recombine different identities and attributes for identity preserving face synthesis in open domains. Previous identity preserving face synthesis processes are largely confined to synthesizing faces with known identities that are already in the training dataset. To synthesize a face with identity outside the training dataset, our framework requires one input image of that subject to produce an identity vector, and any other input face image to extract an attribute vector capturing, e.g., *pose, emotion, illumination, and even the background. We then recombine the identity vector and the attribute vector to synthesize a new face of the subject with the extracted attribute. Our proposed framework does not need to annotate the attributes of faces in any way. It is trained with an asymmetric loss function to better preserve the identity and stabilize the training process. It can also effectively leverage large amounts of unlabeled training face images to further improve the fidelity of the synthesized faces for subjects that are not presented in the labeled training face dataset. Our experiments demonstrate the efficacy of the proposed framework. We also present its usage in a much broader set of applications including face frontalization, face attribute morphing, and face adversarial example detection.*

1. Introduction

Realistic face image synthesis has many real-world applications, such as face super-resolution, frontalization, and morphing, among others. With the emergence of deep generative models, such as the Generative Adversarial Networks (GAN) [10] and the Variational Auto-encoder [16], we have made tremendous progress in building deep networks for synthesizing realistic faces [34, 19, 33, 19]. However, identity preserving face synthesis remains a challenge, especially when the identity of the face is not presented among the training face images.

Many previous works have attempted to synthesize face

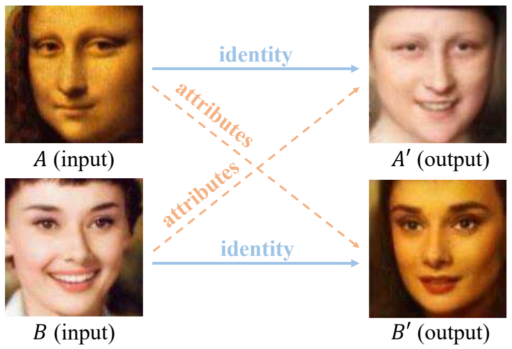

Figure 1. Our method can disentangle identity and attributes from a single face image. With the extracted identity and attributes from two input images A and B, we can generate two new face images A' and B' by recombining the identities and attributes.

images of a specific person. For example, TP-GAN [14] and FF-GAN [36] attempt to synthesize the frontal view of a face from a single face image. DR-GAN [33] can change the pose of an input face image. However, these methods can only manipulate limited types of attributes, such as poses. These methods also require full annotation of attributes for training the models. More recent work, such as CVAE-GAN [4], can produce a variety of attribute changes. Nevertheless, it is not able to synthesize a face with an identity outside the training dataset.

In view of the limitation of CVAE-GAN, we attempt to solve the problem of open-set identity preserving face synthesis. Our goal is to be able to synthesize face images with any specific identity, no matter if the identity is presented in the training dataset or not. To synthesize a face with an identity outside the training dataset, we require one input image of that subject to prodce an identity vector, and any other input face image to extract an attribute vector capturing, *e.g.*, pose, emotion, illumination, and even background. We then combine the identity vector and the attribute vector to synthesize a new face of the subject with the extracted attribute.

To this end, we propose a framework based on Generative Adversarial Networks to disentangle identity and attributes given a face image, and recombine different identities and attributes for identity preserving face synthesis.

As shown in Figure 2, our framework has five parts: 1) an identity encoder network I to encode the identities of subjects; 2) an attribute encoder network E to extract attributes of any given face image; 3) a generator network G to synthesize a face image from a combined input of identity and attributes; 4) a classification network C to preserve the identity of the generated face; and 5) a discriminate network D to distinguish real and generated examples. These five parts are trained end-to-end.

In this framework, we propose a new, simple yet elegant way to extract the attributes of an input face image. The proposed framework does not need to annotate the attributes of the faces at all. We use two loss functions: 1) a reconstruction loss of the attribute image, and 2) a KL divergence loss defined on the attribute vector. These functions enforce that network A extracts the attribute information.

We take full advantage of recent advancements in face recognition, and use the softmax loss on top of network I to encode the identity into an attribute independent vector representation. Therefore, in order to reconstruct the input, network A is forced to extract the attribute information. Meanwhile, we add a KL divergence loss to regularize the attribute vector, such that it dose not contain identity information.

Inspired by the CVAE-GAN [4], we adopt a new asymmetric loss function. More specifically, we adopt a cross-entropy loss when training the discriminative network D, and the classification network C, and use a pairwise feature matching loss when updating the generative network G. This does a better job of preserving the identity while stabilizing the training process.

Last but not least, the proposed framework enables us to effectively leverage large amounts of unlabeled training face images to further improve the fidelity of the synthesized faces for subjects that are not presented in the labeled training face dataset. These unlabeled data can increase intra-class and inter-class variation of the face distributions, and hence improve the diversity of the synthesized faces. As a result, the generated faces present larger changes in pose and expression.

Our experiments demonstrate the efficacy of the proposed framework. We also show that our model can be applied to other tasks, such as face frontalization, face attribute morphing and face adversarial examples detection.

2. Related Work

We briefly summarize the most related works, ranging from general literature of deep generative models to more specific models on face synthesis.

There have been many recent development on deep generative modeling, including deterministic generative models [9, 29], Generative Adversarial Networks (GAN) [10, 26], Variational Auto-encoders (VAE) [16, 30], and autoregression networks [18], to list a few. Among them, the most popular one is perhaps GANs [10, 26].

Many variants of GANs have been proposed to improve the stability of training process [31, 3, 5]. Meanwhile, there are also many works that have added condition information to the generative network and the discriminative network for conditional image synthesis. The condition information could be a discrete label [24, 7, 4, 19], a reference image [22, 8, 37], or even a text sentence [28, 38, 8].

Due to its abundant useful applications, the face is a major focus of image generation. Antipov *et al.* [1] propose a variant of GANs for face aging. Li *et al.* [21] propose a method for attribute-driven and identity-preserving face generation. However, the attribute is only limited to some simple ones. TP-GAN [14] adopt a two-pathway generative network to synthesize frontal faces. Both DR-GAN and TP-GAN obtained impressive results on face frontalization, but they need to explicitly label the frontal faces.

Prior work also explores disentangled representation learning. For example, the DC-IGN [17] uses a variational auto-encoder based method to learn the disentangled presentation. However, DC-IGN needs to fix one attribute in one batch training, which also needs explicit annotations of the attributes. Luan *et al.* [33] proposed DR-GAN to learn a generative and discriminative representation, which explicitly disentangles the pose leveraging the pose annotations.

In contrast, this paper proposes an Identity Preserving Generative Adversarial Network framework, which does not require any attribute annotations. This framework disentangles the identity and attributes representations, and then uses different recombinations of representations for identity preserving face synthesis. This disentaglement allows us to synthesize faces with identities outside what is presented in the training datasets. This addresses a serious limitation of a previous deep generative model-based identity preserving face synthesis method [4]. It simply can not generate faces of identities outside the training dataset.

3. Identity Preserving GANs

In this section, we introduce our face synthesis networks. To synthesize a face of any specific identity, our framework requires two input images, *i.e.*, one input image $\boldsymbol{x^s}$ of a certain subject identity, and another input image $\boldsymbol{x^a}$ to extract the attributes, *e.g.*, pose, emotion, illumination, and even background. Our network synthesizes a new face image $\boldsymbol{x'}$ of the subject with the extracted attributes.

As show in Figure 2, our framework is based on Generative Adversarial Networks. It contains five parts: 1) the identity encoder network I; 2) the attributes encoder network A; 3) the generative network G; 4) the classification network C; and 5) the discriminative network D. The function of the network I is to extract the identity vector $\boldsymbol{f_I}(\boldsymbol{x^s})$ from the subject image $\boldsymbol{x^s}$. The network A is adopted to

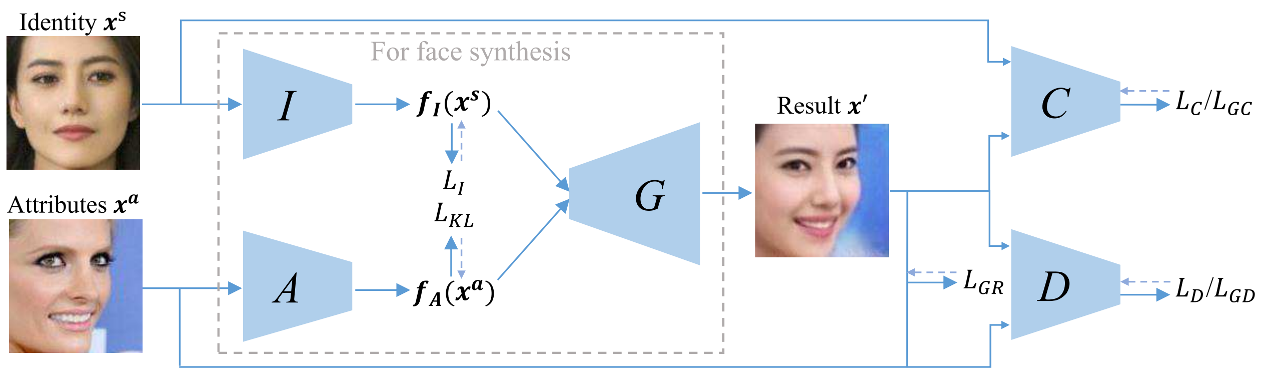

Figure 2. Framework overview: we train a face synthesis network to disentangle identity and attributes from face images and recombine them differently to produce result $\boldsymbol{x}'$, which uses the identity of $\boldsymbol{x}^s$ and attributes of $\boldsymbol{x}^a$. The input/output are drawn with solid lines. The loss functions are drawn with the dashed lines.

extract the attribute vector $\boldsymbol{f_A}(\boldsymbol{x^a})$ of the attributes image $\boldsymbol{x^a}$.

The network G generates a new face image $\boldsymbol{x'}$ using the combined identity vector and attribute vector $[\boldsymbol{f_I}(\boldsymbol{x^s})^T, \boldsymbol{f_A}(\boldsymbol{x^a})^T]^T$. The network C and the network D are only included in the training phase. The network C is used to preserve the identity by measuring the posterior probability $P(c|\boldsymbol{x^s})$, where c is the subject identity of $\boldsymbol{x^s}$. The discriminative network D distinguishes between real and generated examples.

As we do not want to annotate the attributes, extracting the attribute vector from a face image poses a challenge. In the following sections, we first introduce our method of disentangling the identity vector and the attribute vector from a face image in Section 3.1. Then, in Section 3.2, we introduce an asymmetric training method to generate identity-preserving and realistic face images, and to make the training process more stable.

In Section 3.3, in order to further improve the fidelity of the synthesized faces for subjects that are not presented in the labeled training face dataset, we use an unsupervised learning method with a large amount of unlabeled training images. Finally, in Section 3.4, we analyze the objective function of the proposed method and provide the implementation details of the training pipeline.

3.1. Disentanglement of Identity and Attributes

In this section, we introduce the technical details of disentangling the identity vector and the attribute vector using network I and network A, respectively. In our training data, we only have the annotation of the identity of each face, without any annotation of the attribute information. This is because face images with category annotations are relatively easy to obtain. Many large datasets are publically available, such as the FaceScrub [25], CASIA-WebFace [35] and MS-Celeb-1M [12] datasets. However, the annotation of attributes is often more difficult, and sometimes even impossible, such for illumination and the background.

Extracting the identity vector is relatively straightforward. Here, we take full advantage of recent improvement in face recognition. Given a set of face images with the identity annotation $\{\boldsymbol{x}_i^s, c_i\}$, we use the softmax loss for training network I to perform face classification task. Therefore, the same individuals have approximately the same feature which can be used as the identity vector. Formally, the loss function of the network I is

$$\mathcal{L}_\mathcal{I} = -\mathbb{E}_{\boldsymbol{x}\sim P_r}[\log P(c|\boldsymbol{x^s})], \tag{1}$$

where $P(c|\boldsymbol{x^s})$ represents the probability of $\boldsymbol{x^s}$ having identity c. Then, we use the response of the last pooling layer of I as the identity vector.

In order to train network A in a fully unsupervised manner to extract the attribute vector. We propose a new, simple yet elegant way to extract the attributes of each face. In the training process, we consider two loss functions: reconstruction loss and KL divergence loss.

Reconstruction loss. Here we have two situations: whether subject image $\boldsymbol{x^s}$ is the same as attribute image $\boldsymbol{x^a}$ or not. In both cases, we require the result image $\boldsymbol{x}'$ to reconstruct attribute image $\boldsymbol{x^a}$, but with different loss weight. Formally, the reconstruction loss is

$$\mathcal{L}_{GR} = \begin{cases} \frac{1}{2}||\boldsymbol{x^a} - \boldsymbol{x}'||_2^2 & \text{if } \boldsymbol{x^s} = \boldsymbol{x^a} \\ \frac{\lambda}{2}||\boldsymbol{x^a} - \boldsymbol{x}'||_2^2 & \text{otherwise} \end{cases}, \tag{2}$$

where λ is the reconstruction loss weight. Next, we analyze the reconstruction loss under these two situations.

When subject image $\boldsymbol{x^s}$ is the same as attribute image $\boldsymbol{x^a}$, output $\boldsymbol{x}'$ must to be the same as $\boldsymbol{x^s}$ or $\boldsymbol{x^a}$. Supposing there are various face images of an identity, the identity vector $\boldsymbol{f_I}(\boldsymbol{x})$ is almost the same for all samples. But the reconstruction using the $\boldsymbol{f_I}(\boldsymbol{x})$ and $\boldsymbol{f_A}(\boldsymbol{x})$ of different samples are all different. Therefore, the reconstruction loss will force the attribute encoder network A to learn different attributes representation $\boldsymbol{f_A}(\boldsymbol{x})$.

When subject image $\boldsymbol{x^s}$ and attribute image $\boldsymbol{x^a}$ are different, we do not know exactly what the reconstructed result should look like; but we can expect the reconstruction to be approximately similar to the attribute image $\boldsymbol{x^a}$, such as the background, overall illumination, and pose. Therefore, we adopt a raw pixel reconstruction loss with a relatively small weight to maintain the attributes. We set $\lambda = 0.1$ in our experiments. As expected, a large λ causes poor results. we further demonstrate this in the supplementary material due to space limits.

KL **divergence loss.** To help the attributes encoder network learn better representations, we also add a KL divergence loss to regularize the attribute vector with an appropriate prior $P(\boldsymbol{z}) \sim N(0,1)$. The KL divergence loss will limit the distribution range of the attribute vector, such that it dose not contain much identity information, if at all. For each input face image, the network A outputs the mean $\boldsymbol{\mu}$ and covariance of the latent vector. We use the KL divergence loss to reduce the gap between the prior $P(\boldsymbol{z})$ and the learned distributions, *i.e.*,

$$\mathcal{L}_{KL} = \frac{1}{2}(\boldsymbol{\mu}^T\boldsymbol{\mu} + \sum_{j-1}^{J}(\exp(\boldsymbol{\epsilon}) - \boldsymbol{\epsilon} - 1)), \tag{3}$$

where j denotes the j-th element of vector $\boldsymbol{\epsilon}$. Similar to the Variational Auto-encoder [16], we sample the attribute vector using $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{r} \odot exp(\boldsymbol{\epsilon})$ in the training phase, where $\boldsymbol{r} \sim N(\boldsymbol{0}, \boldsymbol{I})$ is a random vector and $\odot$ represents the element-wise multiplication.

3.2. Asymmetric Training for Networks G, C, and D

After extracting the identity vector $\boldsymbol{f_I}(\boldsymbol{x}^s)$ and attribute vector $\boldsymbol{f_A}(\boldsymbol{x}^a)$, we concatenate them in the latent space and feed the combined vector, $\boldsymbol{z} = [\boldsymbol{f_I}(\boldsymbol{x^s})^T, \boldsymbol{f_A}(\boldsymbol{x^a})^T]^T$ into the network G to synthesize a new face image. In this section, we introduce our asymmetric training method. It can generate identity-preserving and realistic face images, which also makes the training process more stable.

Similar to GANs, the generative network G competes in a two-player minimax game with the discriminative network D. Network D tries to distinguish real training data from synthesized data, while network G tries to fool the network D. Concretely, network D tries to minimize the loss function

$$\mathcal{L}_D = -\mathbb{E}_{\boldsymbol{x}\sim P_r}[\log D(\boldsymbol{x^a})] - \mathbb{E}_{\boldsymbol{z}\sim P_{\boldsymbol{z}}}[\log(1 - D(G(\boldsymbol{z}))], \tag{4}$$

However, if the network G directly attempts to maximize $\mathcal{L}_D$ as the traditional GAN, the training process of network G will be unstable. This is because, in practice, the distributions of “real” and “fake” images may not overlap with each other, especially at the early stages of the training process. Hence, network D can separate them perfectly. That is, we always have $D(\boldsymbol{x}^a) \to 1$ and $D(\boldsymbol{x'}) \to 0$, as a result, $\mathcal{L}_D \to 0$. Therefore, when updating the network G, the gradient $\partial\mathcal{L}_D/\partial G \to 0$. This causes gradient vanishing. Recent works [2, 3] also theoretically analyze the gradient vanishing problem in training GANs.

To address this problem, inspired by CVAE-GAN [4], we propose a pairwise feature matching objective for the generator. To generate realistic face image quality, we match the feature of the network D of real and fake images. Let $\boldsymbol{f_D}(\boldsymbol{x})$ denote features on an intermediate layer of the discriminator, then the pairwise feature matching loss is the Euclidean distance between the feature representations, *i.e.*,

$$\mathcal{L}_{GD} = \frac{1}{2}||\boldsymbol{f_D}(\boldsymbol{x}') - \boldsymbol{f_D}(\boldsymbol{x^a})||_2^2. \tag{5}$$

In our experiment, for simplicity, we choose the input of the last Fully Connected (FC) layer of network D as the feature $\boldsymbol{f_D}$.

Meanwhile, classification network C tries to classify faces of different identities, meaning it tries to minimize the loss function

$$\mathcal{L}_{\mathcal{C}} = -\mathbb{E}_{\boldsymbol{x}\sim P_r}[\log P(c|\boldsymbol{x^s})]. \tag{6}$$

In order to generate identity-preserving face images, we also use pairwise feature matching to encourage $\boldsymbol{x'}$ and $\boldsymbol{x^s}$ to have similar feature representations in network C. Let $\boldsymbol{f_C}(\boldsymbol{x})$ denote features produced from an intermediate layer of the classification network C. The feature reconstruction loss is the Euclidean distance between feature representations, *i.e.*,

$$\mathcal{L}_{GC} = \frac{1}{2}||\boldsymbol{f_C}(\boldsymbol{x}') - \boldsymbol{f_C}(\boldsymbol{x^s})||_2^2. \tag{7}$$

Here, we choose the input of the last FC layer of network C as the feature for simplicity. We also try to combine features of multiple layers, it only marginally improves the ability to preserve the identity of network G. network C and network I can share the parameters and be initialized by a pretrained face classification network to speed up the convergence.

3.3. Unsupervised Training

Generating face images of identities which are not presented in the labeled training dataset remains a challenge. It requires the generative network to cover all intra-person and inter-person variations. Existing publically available training datasets with labeled identities are often limited by size, usually do not contain extreme poses or illuminations. In other words, they are not diverse enough.

To solve this problem, we randomly collect about 1 million face images from flicker and Google, and use a face detector to locate the face region. These images have much larger variation, and hence are much more diverse than any existing face recognition datasets. We add these data into

Net	I	A	G	D	C
Loss	$\mathcal{L}_I$	$\mathcal{L}_{KL}, \mathcal{L}_{GR}$	$\mathcal{L}_{GR}, \mathcal{L}_{GC}, \mathcal{L}_{GD}$	$\mathcal{L}_D$	$\mathcal{L}_C$

Table 1. Networks and their related loss function.

the training dataset, and perform an unsupervised training process to help train our generative model to better synthesize face images that do not appear in the training set.

These unlabeled images can be used either as the subject image $\boldsymbol{x^s}$ or the attribute image $\boldsymbol{x^a}$. When used as the attribute image $\boldsymbol{x^a}$, the whole training process remains unchanged. When used as the subject image $\boldsymbol{x^s}$, since it does not have a class label, we ignore the loss function $\mathcal{L}_I$ and $\mathcal{L}_C$. In other words, since networks I and C are fixed, we update the other parts of the end-to-end framework.

These unlabeled data can increase intra-class and inter-class variation of the face distributions, hence improving the diversity of the synthesized faces. As a result, the generated faces present larger changes in poses and expressions. We demonstrate this in the experiments.

3.4. Overall Objective Function

The final synthesis loss function is the sum of all the losses defined above in Equation 1 - 7. Although there are many loss functions, as shown in Table 3.4, each network relates to only one part of the loss function. Therefore, our framework is easy to train and does not need to balance the various loss functions.

In the training phase, we separate each iteration into two steps: one step for the reconstruction process when $\boldsymbol{x^s} = \boldsymbol{x^a}$ and one step for the transformation process when $\boldsymbol{x^s} \neq \boldsymbol{x^a}$. The details of the training algorithm are described in Algorithm 1.

Algorithm 1 Two training process strategy. $\theta_I, \theta_A, \theta_G, \theta_D$, and θ_C are the initial parameters of networks I, A, G, D, and C. $iter \leftarrow 1$.

```
while $\theta_G$ not converaged do
    Sample $\boldsymbol{x^s}, c$ a batch from the dataset;
    if $iter\%2 = 1$ then
        // Training at reconstruction process
        $\lambda \leftarrow 1$
        $\boldsymbol{x^a} \leftarrow \boldsymbol{x^s}$
    else
        // Training at transformation process
        Sample $\boldsymbol{x^a}, c$ a batch from the dataset;
        $\lambda \leftarrow 0.1$
    end if
    $\mathcal{L}_I \leftarrow -\log(P(c|\boldsymbol{x^s}))$
    $\mathcal{L}_C \leftarrow -\log(P(c|\boldsymbol{x^s}))$
    $\boldsymbol{f_I}(\boldsymbol{x^s}) \leftarrow I(\boldsymbol{x^s})$; $\boldsymbol{f_A}(\boldsymbol{x^a}) \leftarrow A(\boldsymbol{x^a})$
    $\mathcal{L}_{KL} \leftarrow KL(\boldsymbol{f_A}(\boldsymbol{x^a})||P(\boldsymbol{z}))$
    $\boldsymbol{x'} \leftarrow G([\boldsymbol{f_I}(\boldsymbol{x^s})^T, \boldsymbol{f_A}(\boldsymbol{x^a})^T]^T)$
    $\mathcal{L}_D \leftarrow -(\log(D(\boldsymbol{x^a})) + \log(1 - D(\boldsymbol{x'})))$
    $\mathcal{L}_{GR} \leftarrow \frac{1}{2}||\boldsymbol{x^a} - \boldsymbol{x'}||_2^2$
    $\mathcal{L}_{GD} \leftarrow \frac{1}{2}||\boldsymbol{f_D}(\boldsymbol{x^a}) - \boldsymbol{f_D}(\boldsymbol{x'})||_2^2$
    $\mathcal{L}_{GC} \leftarrow \frac{1}{2}||\boldsymbol{f_C}(\boldsymbol{x^s}) - \boldsymbol{f_C}(\boldsymbol{x'})||_2^2$
    $\theta_I \xleftarrow{+} -\nabla_{\theta_I}(\mathcal{L}_I)$
    $\theta_C \xleftarrow{+} -\nabla_{\theta_C}(\mathcal{L}_C)$
    $\theta_D \xleftarrow{+} -\nabla_{\theta_D}(\mathcal{L}_D)$
    $\theta_G \xleftarrow{+} -\nabla_{\theta_G}(\lambda\mathcal{L}_{GR} + \mathcal{L}_{GD} + \mathcal{L}_{GC})$
    $\theta_A \xleftarrow{+} -\nabla_{\theta_A}(\lambda\mathcal{L}_{KL} + \lambda\mathcal{L}_G)$
    $iter \leftarrow iter + 1$
end while
```

4. Experiments

We use a subset of the MS-Celeb-1M [12] dataset, which contains about $5M$ images of $80K$ celebrities for training. For each face image, we first detect the facial region with the JDA face detector [6], and then align and resize it to 128×128 pixels.

For networks I, C, and A, we use the same VGG network [32] structure. Meanwhile, I and C share parameters in the training stage to speed up convergence. For network G, it is an inverse VGG structure. Its pooling layers are replaced by upsampling layers, and the convolution layers are replaced with deconvolution layers. For network D, we use the same discriminative network structure as the DCGAN [26]. The batch normalization [15] layer is also applied after each convolution and deconvolution layer. The model is implemented using the deep learning toolbox Torch.

4.1. Analysis of the Proposed Framework

In this section, we perform an ablation study of our framework, sweeping loss combinations and different training strategies to understand how each component works in our framework. Both quantitative and qualitative results are reported.

We compare five variations of our framework: 1) removing the loss $\mathcal{L}_{GD}$; 2) removing the loss $\mathcal{L}_{GC}$; 3) training without the transformation training process (denoted as w/o T); 4) training without the unsupervised learning (denoted as w/o U); 5) our best model with all components. The network structure and training strategy remain the same for all settings.

To quantitatively evaluate our framework, we conduct two face identification experiments to compare the performance of each setting. For identities that are appeared in datasets, We randomly pick $10K$ identities form the MS-Celeb-1M [12] dataset, each with six photos, one for gallery and five for queries. None of these photos are in our training data. For each person, we generate 5 images using the queries and 5 randomly selected attribute images. Then we use the generated image to find the most similar faces in the gallery, and measure the top-1 accuracy.

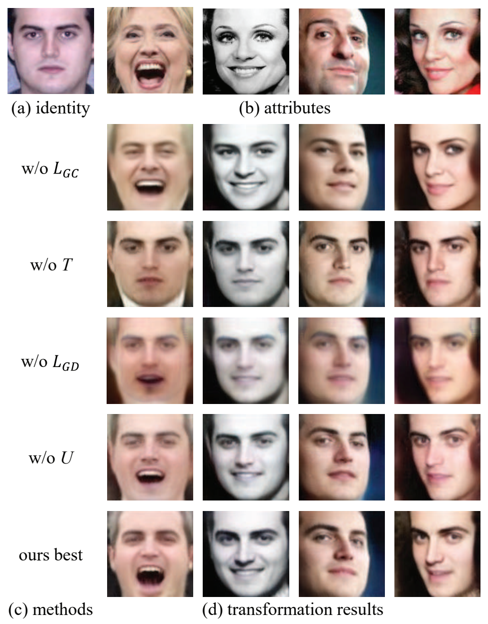

Figure 3. Qualitative comparison between different generative models using different loss combinations and training strategies.

Method	original data	w/o L_{GC}	w/o T	w/o L_{GD}	w/o U	ours best
Top-1 acc	87.39	5.71	79.19	80.52	80.24	**81.11**

Table 2. Model comparison: Top-1 identification rates (%) on the MS-Celeb-1M [12] dataset.

For identities that are not appeared in the datasets, We use the Multi-PIE dataset. We choose 6 kinds of attributes from Multi-PIE dataset. Then for each person in the dataset, faces with one kind of attributes are set as the galleries, queries are the original face images and the generated images with the rest kinds of attributes.

In Table 2 and Table 3, we report the face identification top-1 accuracy of each setting. All components can improve the identity preserving capability of the framework. Among them, the $\mathcal{L}_{GC}$ contributes the most. Meanwhile, we also measure the top-1 accuracy using the real query image, Our generated images achieves comparable results.

Figure 3 illustrates the qualitative results of four variants. We can observe that removing the transformation training process cause the generated results lose attribute details, e-specially the emotion. Removing the loss $\mathcal{L}_{GD}$ will cause the generated image to be blurry. Removing the loss $\mathcal{L}_{GC}$, the generated samples cannot keep the identity information. The results generated by our full model achieves better results. After using unlabeled data for the unsupervised learning, our model can generate images with larger variations. For example, as the bottom left image shown in Figure 3, the mouth can be opened wider.

4.2. Function of KL Divergence Loss

In this section, we will verify whether the KL divergence loss is able to help to remove the identity information in the attribute vector. We first train two models with and without the KL divergence loss, respectively. Then, we conduct the experiment using the FaceScrub [25] dataset. We randomly split these faces into two parts, one for training and the other for validation. For each setting, we use the network A to extract the attribute vector for all the faces in the Facescrub dataset. We then use a multilayer perceptron (MLP) to train a classification model to distinguish the face features of different identities in the training set. We also test the top-1 accuracy on the validation set.

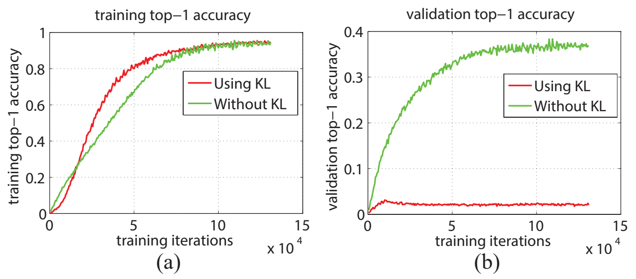

Figure 4. Analysis of KL divergence loss. The attribute vector learned with KL gets a lower top-1 accuracy.

Method	original data	w/o L_{GC}	w/o T	w/o L_{GD}	w/o U	ours best
Top-1 acc	97.47	11.76	95.47	95.53	96.41	**96.80**

Table 3. Model comparison: Top-1 identification rates (%) on Multi-PIE dataset.

The results are presented in Figure 4. We can see that in the validation set, the attribute vector learned using KL loss has the lower top-1 accuracy, which means that it contains less identity information. It validates that the KL divergence loss is able to help network A to remove identity information in the attribute vector.

4.3. Face Attributes Transformation

This section presents the results of face attribute transformation. The goal of face attribute transformation is to generate an image $\boldsymbol{x}'$ that combines the identity of a input face $\boldsymbol{x}^s$ and the attributes of another input face $\boldsymbol{x}^a$. To demonstrate that our framework has the ability to generate faces with identities that do not exist in the training dataset, we conduct two experiments: generating face images of i-dentities inside or outside the training dataset, respectively.

Figure 5 presents the face synthesis results of the identities that appeare in the training dataset. Our method performs well in face synthesis, preserving both identity and attributes.

Another important feature of our method is that it can synthesize unseen faces from the training set. Figure 6 shows the zero-shot identity face synthesis results. Although our model never see these identities, but we can also generate high quality face images which keep the identity and attributes of the given faces.

In Figure 7, we show that our framework can also be

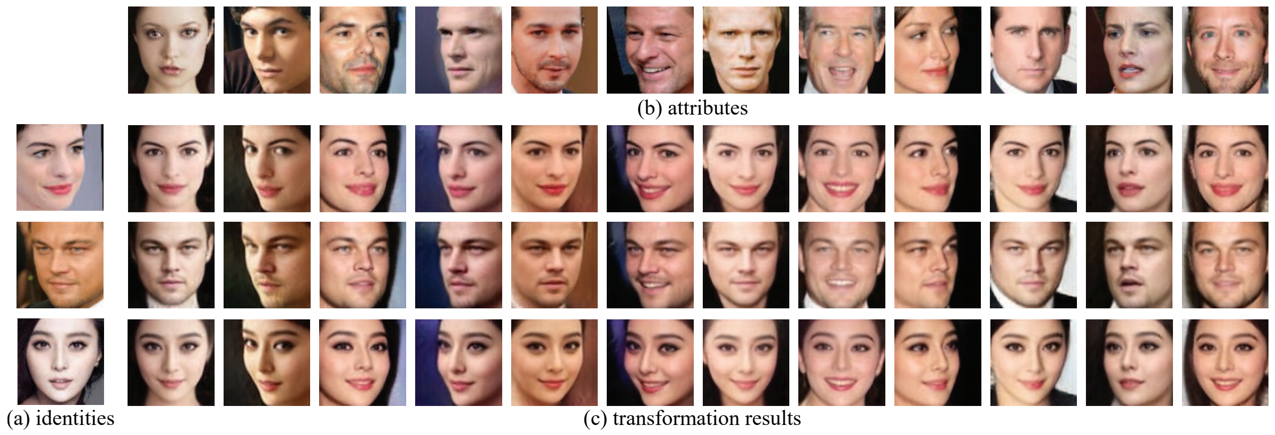

Figure 5. Face synthesis results using the identities that appear in training dataset and randomly chosen images as attributes.

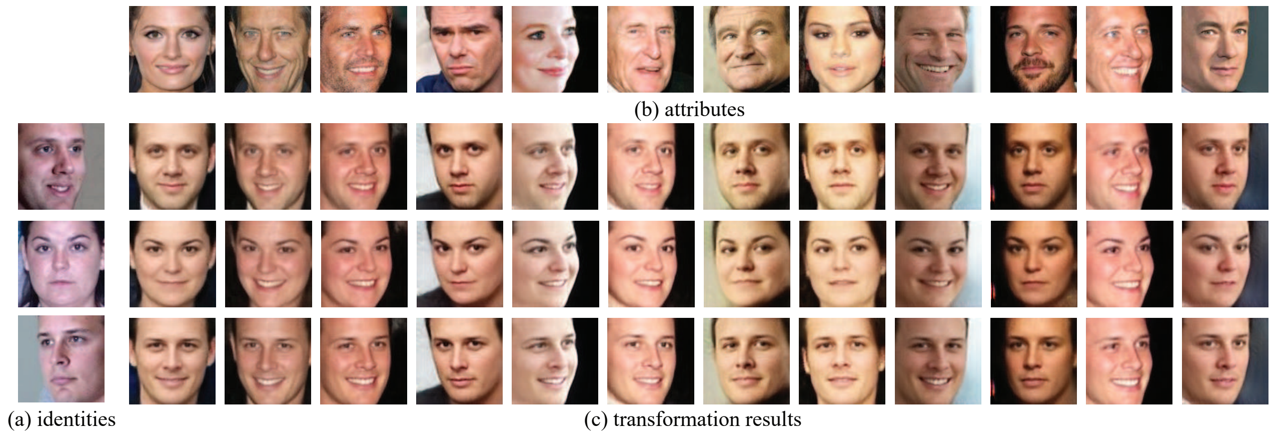

Figure 6. Face synthesis results using the zero-shot identities and randomly chosen images as attributes.

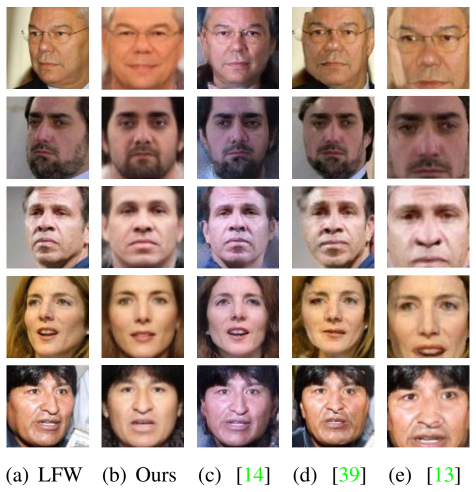

Figure 7. Face frontalization results on LFW datasets. Results of other methods are from [14].

used for face frontalization. The results of other methods are from paper [14]. Unlike other methods our framework is not trained using pose annotations. With a frontal face as the input image to extract attributes, our framework can generate identity preserving frontal faces. Compared with [14], the advantage of our method is that we are able to keep the lighting and skin color.

4.4. Face Attributes Morphing

In this part, we validate that the attribute in the generated images will continuously change with the latent vector. We call this phenomenon attribute morphing. We test our model on the Multi-PIE [11] dataset. We first select a pair of images $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, and then extract the attribute vector $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ using the attribute network A. Then, we obtain a series of attribute vectors $\boldsymbol{z}$ by linear interpolation, *i.e.*, $\boldsymbol{z} = \alpha \boldsymbol{z}_1 + (1-\alpha)\boldsymbol{z}_2, \alpha \in [0,1]$. Figure 8 presents the results of face attribute morphing. We can gradually change the pose, emotion, or lighting. For more results, please refer to the supplementary material.

4.5. Face Adversarial Example Detection

Deep network based face verification systems have been widely used in surveillance and access control. However, the existence of adversarial examples puts the security or safety of these systems at risk. In this experiment, we show that our framework can be used for face adversarial example detection without any modification.

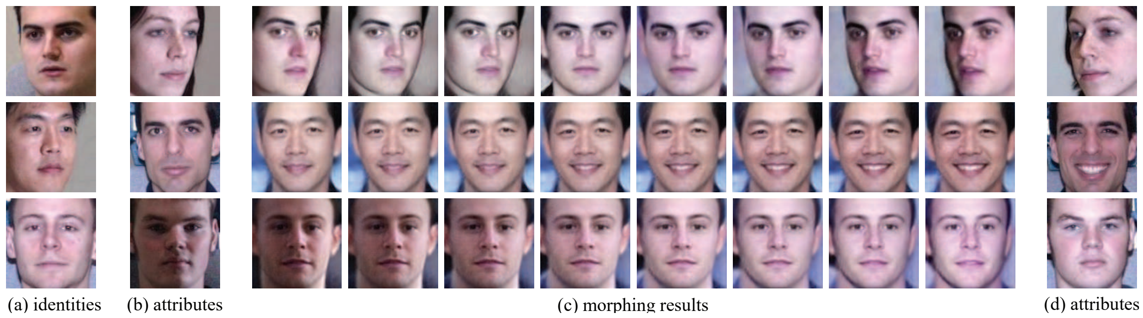

Figure 8. Face morphing results using unseen identities between two attributes. Our framework can gradually change pose, emotion, and lighting.

For face verification, given two faces, we first extract the features for the two faces using a pretrained face classification DNN model. Then, we calculate the distance of the two features, and compare it to a threshold. If the feature distance is smaller than the threshold, they are predicted to have the same identity, and vice versa.

Supposing two faces $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ have different identities, we can find imperceptible perturbations $\boldsymbol{r}$, such that $\boldsymbol{x}_1 + \boldsymbol{r}$ will be regarded as the same person as $\boldsymbol{x}_2$ using the above face verification system. Here, $\boldsymbol{x}_1 + \boldsymbol{r}$ is the adversarial sample. To find an adversarial sample, we optimize the problem.

$$\begin{aligned} &\min \|\boldsymbol{r}\|_2^2 \\ &s.t. \|\boldsymbol{f}_C(\boldsymbol{x}_1 + \boldsymbol{r}) - \boldsymbol{f}_C(\boldsymbol{x}_2)\|_2^2 < \theta, \end{aligned} \tag{8}$$

Where $\boldsymbol{f}_C$ is the extracted feature from the pretrained network, and θ is the predefined threshold.

As shown in Figure 9, (a) and (c) are the two inputs $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, and (b) is the adversarial example $\boldsymbol{x}_1 + \boldsymbol{r}$. Since the adversarial examples have similar identity features with others faces, if we reconstruct the image from the feature using the proposed framework, it will generate an image of the other person. (e). The adversarial example and its reconstruction clearly have different identities. Based on this observation, we can use our generative model to reconstruct the faces, and compare the identity of the original faces and the reconstruction results to identify adversarial examples.

We use the LFW [20] to conduct the experiments. For each of the 3000 pairs of different identities, we generate two adversarial examples by performing adversarial attacks with each other. In total we obtain 6000 adversarial examples for each predefined threshold. Here, we choose four different thresholds to conduct the experiments: $[0.4, 0.6, 0.8, 1]$. At the same time, we have 6000 source images and their reconstruction. Although we can train another neural network to distinguish adversarial and source examples [23], the problem is that this neural network can be attacked again. Instead, we use the LBP [27] feature. We extract LBP features from the input and its reconstruction image and then concatenate them. Finally, we train a linear SVM to conduct binary classification. The results are

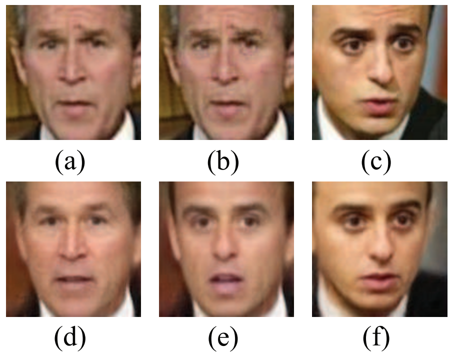

Figure 9. Adversarial example detection in face verification systems. (a) is the source image, (b) is the adversarial example which aims to attack face image (c). (d), (e) and (f) are the reconstruction results from our framework. We can clearly observe that although the adversarial example shares a similar appearance with the source image, their reconstruction results have different appearances.

threshold	1.0	0.8	0.6	0.4
acc	76.73%	82.58%	87.18%	92.41%

Table 4. Results of adversarial examples detection at different thresholds.

shown in Table 4, we can achieve 92.41% accuracy if we require the feature distance to be less than 0.4.

5. Conclusion

In this paper, we propose an Open-Set Identity Preserving Generative Adversarial Network framework for disentangling the identity and attributes of faces, synthesizing faces from the recombined identity and attributes. The framework shows superior performance in generating realistic and identity preserving face images, even for identities outside the training dataset. Our experiments demonstrate that our framework can also be applied to other tasks, such as face image frontalization, face attribute morphing, and adversarial example detection in face verification systems. In future work, we hope to explore whether our framework can be applied to other datasets, like birds, flowers, and rendered chairs.

References

[1] G. Antipov, M. Baccouche, and J.-L. Dugelay. Face aging with conditional generative adversarial networks. *arXiv preprint arXiv:1702.01983*, 2017. 2

[2] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. In *NIPS 2016 Workshop on Adversarial Training. In review for ICLR*, volume 2016, 2017. 4

[3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017. 2, 4

[4] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua. Cvae-gan: Fine-grained image generation through asymmetric training. *arXiv preprint arXiv:1703.10155*, 2017. 1, 2, 4

[5] D. Berthelot, T. Schumm, and L. Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017. 2

[6] D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun. Joint cascade face detection and alignment. In *European Conference on Computer Vision*, pages 109–122. Springer, 2014. 5

[7] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016. 2

[8] H. Dong, S. Yu, C. Wu, and Y. Guo. Semantic image synthesis via adversarial learning. *arXiv preprint arXiv:1707.06873*, 2017. 2

[9] A. Dosovitskiy, J. Springenberg, M. Tatarchenko, and T. Brox. Learning to generate chairs, tables and cars with convolutional networks. 2016. 2

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 1, 2

[11] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010. 7

[12] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*, pages 87–102. Springer, 2016. 3, 5, 6

[13] T. Hassner, S. Harel, E. Paz, and R. Enbar. Effective face frontalization in unconstrained images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4295–4304, 2015. 7

[14] R. Huang, S. Zhang, T. Li, and R. He. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. *arXiv preprint arXiv:1704.04086*, 2017. 1, 2, 7

[15] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 5

[16] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1, 2, 4

[17] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547, 2015. 2

[18] H. Larochelle and I. Murray. The neural autoregressive distribution estimator. In *AISTATS*, volume 1, page 2, 2011. 2

[19] A. B. L. Larsen, S. K. Sønderby, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015. 1, 2

[20] E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li, and G. Hua. Labeled faces in the wild: A survey. In *Advances in Face Detection and Facial Image Analysis*, pages 189–248. Springer, 2016. 8

[21] M. Li, W. Zuo, and D. Zhang. Convolutional network for attribute-driven and identity-preserving human face generation. *arXiv preprint arXiv:1608.06434*, 2016. 2

[22] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. *arXiv preprint arXiv:1703.00848*, 2017. 2

[23] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017. 8

[24] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2

[25] H.-W. Ng and S. Winkler. A data-driven approach to cleaning large face datasets. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 343–347. IEEE, 2014. 3, 6

[26] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2, 5

[27] M. A. Rahim, M. S. Azam, N. Hossain, and M. R. Islam. Face recognition using local binary patterns (lbp). *Global Journal of Computer Science and Technology*, 2013. 8

[28] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016. 2

[29] S. E. Reed, Y. Zhang, Y. Zhang, and H. Lee. Deep visual analogy-making. In *Advances in neural information processing systems*, pages 1252–1260, 2015. 2

[30] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014. 2

[31] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016. 2

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[33] L. Tran, X. Yin, and X. Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, volume 4, page 7, 2017. 1, 2

[34] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2image: Conditional image generation from visual attributes. *arXiv preprint arXiv:1512.00570*, 2015. 1

[35] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014. 3

[36] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker. Towards large-pose face frontalization in the wild. *arXiv preprint arXiv:1704.06244*, 2017. 1

[37] D. Yoo, N. Kim, S. Park, A. S. Paek, and I. S. Kweon. Pixel-level domain transfer. In *European Conference on Computer Vision*, pages 517–532. Springer, 2016. 2

[38] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016. 2

[39] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 787–796, 2015. 7