IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 7, 2019, accepted March 6, 2019, date of publication March 14, 2019, date of current version April 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2905147

Faster, Smaller, and Simpler Model for Multiple Facial Attributes Transformation

JONATHAN HANS SOESENO[1], DANIEL STANLEY TAN[1], WEN-YIN CHEN[2], AND KAI-LUNG HUA[1,3]

[1]Department of Computer Science and Information Technology, National Taiwan University of Science and Technology, Taipei 10607, Taiwan
[2]Department of Arts and Design, National Taipei University of Education, Taipei 10478, Taiwan
[3]Center for Cyber-Physical System Innovation, National Taiwan University of Science and Technology, Taipei 10607, Taiwan

Corresponding author: Kai-Lung Hua (hua@mail.ntust.edu.tw)

This work was supported in part by the Center for Cyber-Physical System Innovation and in part by the Center of Intelligent Robots from the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan and the Ministry of Science and Technology of Taiwan under Grant MOST107-2218-E-011-014, Grant MOST106-2221-E-011-154-MY2, and Grant MOST106-2218-E-011-009-MY2.

ABSTRACT There are many existing models that are capable of changing hair color or changing facial expressions. These models are typically implemented as deep neural networks that require a large number of computations in order to perform the transformations. This is why it is challenging to deploy on a mobile platform. The usual setup requires an internet connection, where the processing can be done on a server. However, this limits the application's accessibility and diminishes the user experience for consumers with low internet bandwidth. In this paper, we develop a model that can simultaneously transform multiple facial attributes with lower memory footprint and fewer number of computations, making it easier to be processed on a mobile phone. Moreover, our encoder–decoder design allows us to encode an image only once and transform multiple times, making it faster as compared to the previous methods where the whole image has to be processed repeatedly for every attribute transformation. We show in our experiments that our results are comparable to the state-of-the-art models but with 4× fewer parameters and 3× faster execution time.

INDEX TERMS Facial attribute transformations, generative adversarial networks, image translation.

I. INTRODUCTION

With the advancements in deep neural networks and generative adversarial networks (GAN) [2], models for transforming facial attributes have produced more and more realistic synthetic images. Many of these advancements rely on increasing the model capacity and adding more layers to improve the quality of the transformations. However, the additional number of parameters and number of layers come at the cost of larger memory requirements, larger number of computations, and longer processing times. This makes it harder to deploy these models to mobile phones since it has significantly fewer computational resources as compared to desktop computers.

A common work around to this problem is to setup a server where all the processing and transformations are performed. The resulting images are then sent back to the user's mobile phone. However, in this setup, the images have to be sent back and forth for every variation in the transformations, such as different hair colors. This is heavily dependent on the quality of the internet connection and limited by the bandwidth that is available to the user, which is a luxury for many developing countries.

In this paper, we propose a Simple Attribute Transformation model based on GANs (SAT-GAN) that can simultaneously perform multiple facial attribute transformations such as changing hair colors or changing the facial expression. Our model has significantly fewer number of parameters and more efficient computations, which leads to faster transformations with lower memory requirements. This allows our model to be easily deployed on mobile platforms. As shown in Figure 1, we designed our model to have an encoder-decoder type architecture which can encode an image into a lower dimensional latent representation and re-use that encoded vector for all the possible transformations. This is in contrast to one big feed-forward convolutional neural network of StarGAN [1], which is inefficient since they will need to perform all the computations again for every attribute transformation. The caching property of our encoder-decoder

The associate editor coordinating the review of this manuscript and approving it for publication was Amit Singh.

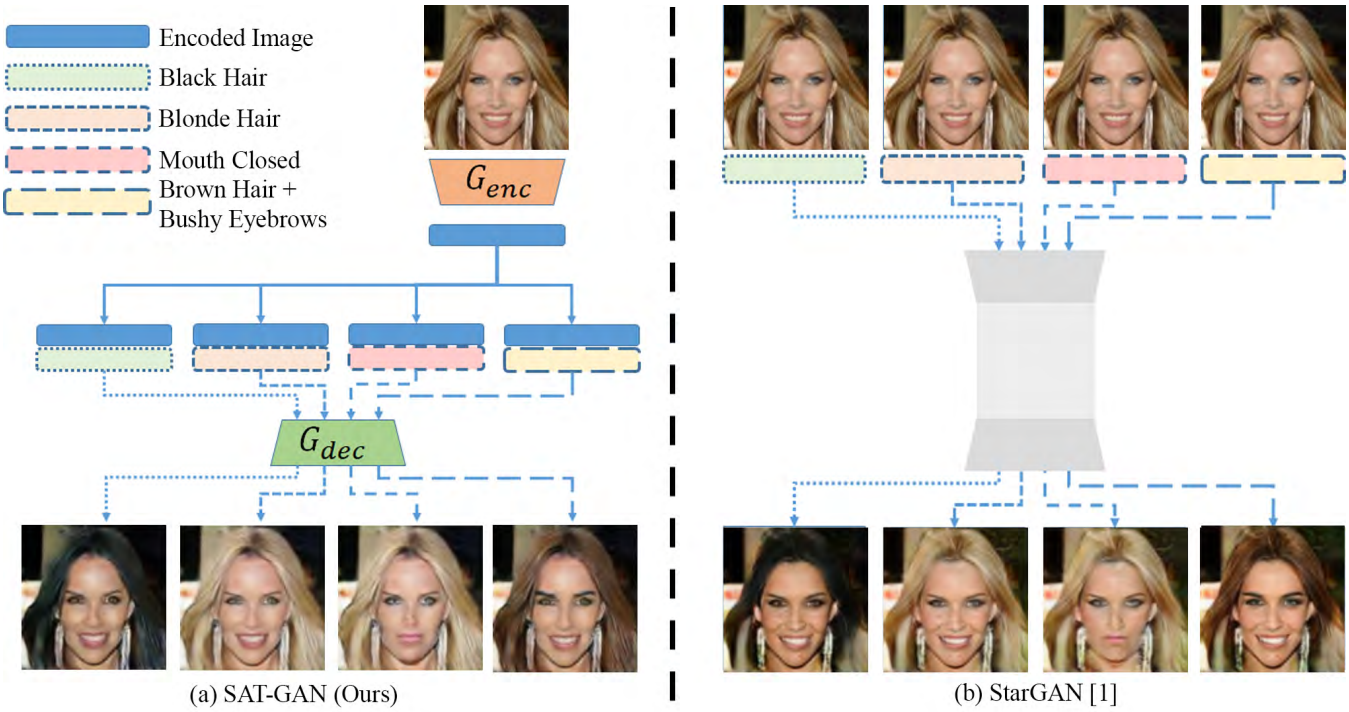

FIGURE 1. We present a simple attribute transformation model based on generative adversarial networks (SAT-GAN) that can simultaneously transform multiple facial attributes. (a) shows our proposed model for facial attribute transformation and (b) shows a state-of-the-art baseline StarGAN [1]. Our encoder-decoder architecture, as shown in (a), enables us to encode the image once and cache the lower dimensional encoded vector. This allows us to save half of the computations when transforming across multiple facial attributes, which makes it more computationally efficient. This is in contrast to (b) StarGAN [1] where they use one feed-forward network, which requires the whole image to be repeatedly processed again for every transformation.

architecture is a desirable property since users will most likely try different configurations for the images they are trying to transform. Our setup allows faster user interactions, thus improving the overall user experience. We show in our experiments (Section IV) that our model produces high quality images that are comparable with existing state-of-the-art models with 4× fewer parameters and 3× faster execution time.

II. RELATED LITERATURE

The main idea of traditional approaches for transforming facial attributes is to extract attributes from a separate reference image and then combine it with the input generating an image corresponding to the attributes of the reference image, some are done using colorization technique [3], [4] or style transfer [5]–[8]. For these approaches, a template or a reference image is required to perform the desired attribute transformation, this implies that different reference images are needed to perform different transformations thus making it difficult to control the transformations. These approaches also assume some constraints on the poses by requiring frontal faces [5], [6]. Yang *et al.* [8] worked around this limitation by using a 3D face model.

Recently, deep generative models have shown great potential in a wide range of applications [9]–[17]. One approach on generative modeling is using Variational Autoencoders (VAE) [12], [18] which impose a prior on the representation space z in order to regularize the model and generate images from it. Despite its successes, VAEs use pixel-wise reconstruction error which cause generated images to be blurry [11]. Another approach to generative modeling uses Generative Adversarial Networks (GANs). GANs have achieved impressive results in several tasks such as image generation [9], [10], image editing [19], and representation learning [20], [21]. GANs learn these generative models by setting up an adversarial game between the generator and the discriminator. First, the generator learns to transform noise vectors into fake samples which resemble the real samples drawn from the distribution of natural real images. The discriminator then tries to distinguish between the real and fake samples.

Isola *et al.* [22] showed the capacity of GANs to model transformations where most of the information is not present in the input image, such as generating realistic objects from the respective sketch, colorization of grayscale images and converting from day to night. However, their approach requires the ground truth image of the opposed transformation to be provided. Zhu *et al.* [23] proposed a solution to the problem of needing ground truth images through a cycle consistency constraint, but their framework only allows for binary transformations. Extending it to multiple attribute transformations requires training a different model for every pair-wise mapping.

Perarnau *et al.* [11] built on top of the work of Radford *et al.* [10] by conditioning the generator and discriminator with facial attribute labels, allowing them to perform multiple transformations in a single model. Antipov *et al.* [24] extended their work for face aging by adding an identity preservation constraint. They have shown that this helps the generator create a more realistic reconstruction. Choi *et al.* [1] introduced a similar framework that can handle multiple transformations and also incorporates cycle consistency to learn in an unpaired setting.

The current state-of-the-art image to image translation StarGAN [1] is able to handle multiple facial attribute transformations using a single model. However, StarGAN [1] uses one big feed forward convolutional neural network where they take an image together with a target attribute as an input, which we argue is computationally inefficient since we need to recompute everything for every transformation. We propose a model for facial attribute transformations that significantly reduces the computational requirements while still producing a comparable result to the state-of-the-art methods. Our framework uses an encoder-decoder setup for the generator where we can cache the encoded image and insert the information regarding the transformation attribute only to the decoder. This gives a speed up in the transformations since only half of the computations are needed for the succeeding transformations thus requiring fewer computational resources.

III. PROPOSED METHOD

A. BACKGROUND

In this subsection, we briefly discuss some concepts that our method is built on top of.

1) GENERATIVE ADVERSARIAL NETWORKS

The fundamental idea behind Generative Adversarial Networks (GANs) is the inclusion of the discriminator. It transforms the learning problem into a game, more specifically a two-player minimax game, where the optimal solution is a Nash equilibrium.

A typical framework of GANs includes a generator G and a discriminator D. The role of the discriminator is to learn how to tell apart real images from synthetic images. The generator G, on the other hand, synthesizes images from randomly sampled noise vector z and tries to make it as realistic looking as possible in order to trick the discriminator D into classifying the synthesized image $G(z)$ as real. This is represented as a minimax optimization in the form shown in Eq. 1, where $p(x)$ and $p(z)$ represent the distributions of the input images x and the distribution of the noise vector z respectively. In the formulation of these networks, the generator G has access to the gradients of the discriminator D and therefore has some form of instruction as to how to improve itself. This enables the generator to learn how to produce realistic looking images.

$$\min_G \max_D \mathcal{L}_{\text{GAN}} = \mathbb{E}_{x\sim p(x)}[\log D(x)] + \mathbb{E}_{z\sim p(z)}[\log\left(1 - D(G(z))\right)] \quad (1)$$

In the beginning of the training process, the fake images generated by G are extremely poor and are rejected by D with high confidences. Therefore it is better for G to optimize for $\log(D(G(z)))$ instead of $\log(1 - D(G(z)))$. Both objectives result in the same fixed point, but $\log(D(G(z)))$ provides stronger gradients in the early stages of learning.

2) WASSERSTEIN GENERATIVE ADVERSARIAL NETWORK

The original formulation of GANs uses the standard binary cross-entropy loss to classify real from fake images. If we solve for the optimal discriminator and replace it back to the original objective function, we can show that this objective optimizes for the Jensen-Shannon (JS) divergence. Several problems have been found from this formulation that contributes to the instability of GANs [21], [25]. One is that the binary cross-entropy innately uses the sigmoid function to model the probability of the image being real. While it has nice probabilistic interpretations, it suffers from vanishing gradients. More specifically, if the discriminator is too good at classifying real from fake, then the discriminator's loss will be close to zero and the gradients will also quickly drop close to zero. This is also related to the problem in the JS divergence formulation where the two distributions have to have an intersection for it to train. On the other hand, if the discriminator performs poorly, then the feedback that it gives to the generator will also be meaningless. Arjovsky *et al.* [25] also showed that JS-divergence are not sensible cost functions when learning distributions supported by low dimensional manifolds. This is most likely the case for images where the distribution of natural images lie in a much lower dimensional manifold. This is a reasonable assumption since there are many constraints that governs natural images, for example, simply setting the values of the pixels randomly will make a valid image but it won't be considered as a natural image since it will look very far from an image that we can observe when we capture a scene or an object from a camera.

Arjovsky *et al.* [25] proposed to change the metric from JS-divergence to the earth-mover distance or also called Wasserstein-1 distance, which can be interpreted as how much work you need to spend in order to transport a pile of dirt / earth. The earth-mover distance is continuous and differentiable almost everywhere and it can still provide meaningful smooth distances between two non-overlapping distributions in low dimensional manifolds [25]. The new objective function is shown in Eq. 2. In this formulation, the discriminator loses its direct interpretation of distinguishing real images from fake ones but the idea of learning guiding the generator to produce samples that gets closer to real data distribution is still there.

$$\min_G \max_D \mathbb{E}_{x\sim p(x)}[D(x)] - \mathbb{E}_{z\sim p(z)}[D(G(z))] \quad (2)$$

To make the problem more tractable, they also constrained the search space of the discriminator to the set of 1-Lipschitz functions. This bounds the magnitude of the gradients to be less than one. Intuitively, this has an effect that the functions

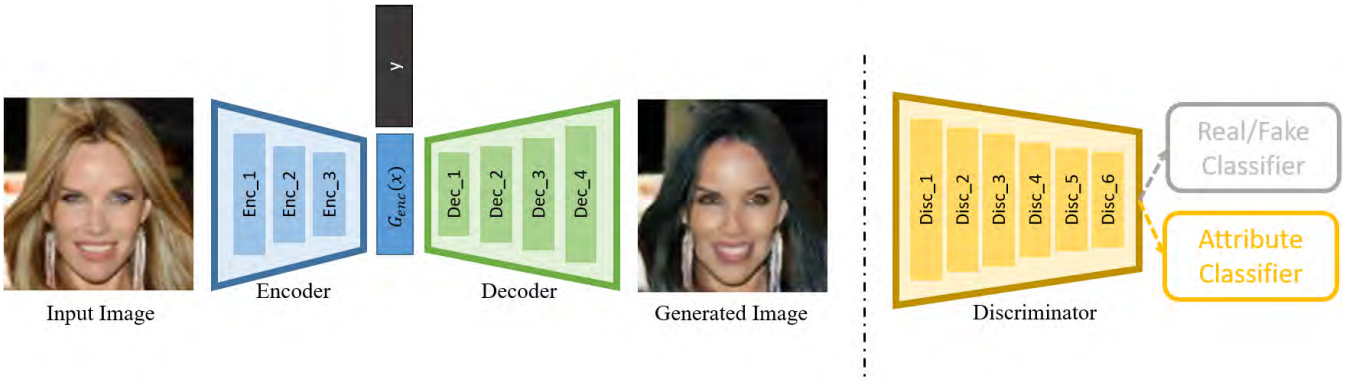

FIGURE 2. This figure shows an overview of our framework. Our model first encodes the input image producing a lower dimensional encoded vector. Next, we combine the encoded vector with the target attribute vector and feed it into the decoder which produces a synthesized image with the desired attributes. Our model is trained under the GAN framework where we have a discriminator that encourages the generator to produce realistic outputs. Details of each layer are shown in Table 2.

will not be too steep and will roughly be linear which, makes it more well behaved. In the original paper, they used weight clipping to enforce the Lipschitz constraint. However, this is not a good approach since it hampers the flexibility of the neural networks. Gulrajani *et al.* [26] proposed to use a soft constraint where they add a gradient penalty term in the objective function that encourages the magnitude of the gradients to be close to one, as shown in Eq. 3.

$$\mathbb{E}\left[(\|\nabla_x D(x)\|_2 - 1)^2\right] \tag{3}$$

B. PROBLEM FORMULATION

Suppose we have a natural face image x, the task is to be able to freely transform across k facial attributes such as hair color, skin tone, and facial expression. We can formulate this problem mathematically as learning a non-linear function $\mathcal{F} : (x^{(a)}|y^{(b)}) \rightarrow x^{(b)}$, that takes in an input image $x^{(a)}$ having some facial attribute a and transform it to an image $x^{(b)}$ having the facial attribute b corresponding to the target attributes specified by $y^{(b)}$. The attribute vector y is a vector of size k wherein the elements correspond to target attributes that the user wants to be present or absent. A positive number denotes the presence of the particular attribute and a zero denotes absence. We show in our experiments (section IV) that this formulation allows us to control the degree of the transformations. The function $\mathcal{F}$ is approximated using a deep convolutional neural network trained to optimize an adversarial loss, a self cycle consistency loss, and an attribute classification loss.

Our goal for this paper is not only to be able to perform the task of facial attribute transformation, but also to make it fast and easily deployable to mobile platforms. The core motivation is to improve the user experience of facial attribute transformation applications. A typical scenario would be a user selects a photo and try to change the hair color and maybe make the person smile more. However, prior to any visual feedback of the transformation, there is no way to know how the image would look like on a particular parameter setting. This means that the user will have to try different parameter settings such as how much blonde or how much smile until they see a result that they like. A few seconds delay on the visual feedback is enough to spoil the user experience, thus, for these types of applications, speed is crucial.

C. SAT-GAN

An overview of our framework is shown in Figure 2. Our model follows a generative adversarial network (GAN) framework where there are two networks, a generator G and a discriminator D, that plays a minimax game during training.

The generator takes in a facial image with a target attribute label as input and synthesizes a transformed version of the input having the desired attributes. The discriminator D performs two tasks where it tries to predict whether the synthesized image looks real or fake (denoted as D_{GAN}), and also predict the facial attributes present in it (denoted as D_{attr}). The discriminator D acts as guide or a critic in the sense that it guides the generator in learning to produce more realistic outputs that correctly follows the facial attributes specified by the user.

Since a typical use case of our application will require the model to process the same image multiple times, it would be desirable if there were parts of the computations that can be cached. With this in mind, we divided our generator G into two networks namely, encoder G_{enc} and decoder G_{dec}. The role of the encoder G_{enc} is to extract features from the image and project it in a lower dimensional representation while still preserving as much information that are relevant to the transformations. The encoder G_{enc} produces image representations that is independent of the attributes. Intuitively, we can think of the encoder as extracting structure such as the shape of the hair and face, and identifying features which remains constant regardless of the target attributes specified by the user.

The decoder G_{dec} then accepts this lower dimensional representation together with the target attribute labels y and synthesizes the desired attributes on the facial image.

TABLE 1. Comparison in terms of memory requirements and number of parameters. For CycleGAN [23] the number of parameters have to be multiplied by the number of possible permutations among k facial attributes taken 2 at a time $P(k, 2)$. This is because it can only transform from one specific attribute to another per model. The formula for permutation P with n objects taken r at a time is defined in Eq. 9.

Model	Memory Requirements (1 forward pass)	Parameters
StarGAN [1]	21,430,272 bytes	8,468,160
CycleGAN [23]	**10,354,688 bytes**	2,110,272 * $P(k, 2)$
SAT-GAN (Ours)	16,191,488 bytes	**2,026,368**

The attribute labels are tiled and concatenated channel-wise to the encoded representation. With this framework, we can cache the encoded representations and only use the decoder in synthesizing images with different attributes.

D. NETWORK ARCHITECTURE

Our network architecture is detailed in Table 2. The encoder is composed of three convolutional layers which downsamples the image representations by a factor of 4. Similarly, the decoder is composed of four convolutional layers that upsamples the image representations back to the original dimensions. We used instance normalization and ReLU activation function for every layer except the output layer of the decoder where we used tanh. We explored adding more layers but we found that the improvements are very minimal which is not worth the added parameters and operations as we show in our experiments in section IV.

The discriminator consists of six convolutional layers and branches at the end to produce two outputs. One branch outputs the probability of being real (which we denote as D_{GAN}) and the other predicts the attributes present in the image (which we denote as D_{attr}). There were no normalization layers in the discriminator and we use LeakyReLU to avoid sparse gradients.

E. LOSS FUNCTIONS

Our network optimizes for three objectives, the reconstruction loss, the GAN loss and the attribute classification loss. The reconstruction loss encourages the network to preserve the content in the input image. This is usually implemented as a mean squared error or mean absolute error of the input image with the ground truth image. However, in our setting, we do not have access to ground truth transformed version of our input images. It is also very costly and impractical to collect ground truth transformed images for every attribute we want to consider. Inspired by Zhu *et al.* [23], we use a cycle-consistency loss to train our network in an unpaired setting, without ground truth images. The core idea of the cycle-consistency loss is to first transform the image $x^{(a)}$ having some facial attribute a and transform it to image $x^{(b)}$ having some facial attribute b, then back to $x'^{(a)}$ again. This forms a transformation cycle $x^{(a)} \rightarrow x^{(b)} \rightarrow x'^{(a)}$. Since these are all the operating on the same content image, we expect $x^{(a)}$ to be equal to $x'^{(a)}$. This is enforced with a

TABLE 2. Detailed network architecture of our proposed SAT-GAN model. In order to save space, we simplified the notations for the convolutional layers where 'Conv64_k7_s1_p3' denotes a convolutional layer with 64 filters, kernel size of 7, stride 1, and padding 3.

Layer Name	Modules
Enc_1	Conv64_k7_s1_p3-InstanceNorm-ReLU
Enc_2	Conv128_k4_s2_p1-InstanceNorm-ReLU
Enc_3	Conv256_k4_s2_p1-InstanceNorm-ReLU
Dec_1	Conv256_k3_s1_p1-InstanceNorm-ReLU
Dec_2	ConvTrans128_k4_s2_p1-InstanceNorm-ReLU
Dec_3	ConvTrans64_k4_s2_p1-InstanceNorm-ReLU
Dec_4	Conv3_k7_s1_p3-Tanh
Disc_1	Conv64_k4_s2_p1-LeakyReLU
Disc_2	Conv128_k4_s2_p1-LeakyReLU
Disc_3	Conv256_k4_s2_p1-LeakyReLU
Disc_4	Conv512_k4_s2_p1-LeakyReLU
Disc_5	Conv1024_k4_s2_p1-LeakyReLU
Disc_6	Conv2048_k4_s2_p1-LeakyReLU
Disc_GAN	Conv1_k3_s1_p1
Disc_attr	Conv[N_{attr}]_k2_s1_p0

mean absolute error or L_1 norm, as shown in Eq. 4. We choose the intermediate attribute b in the transformation cycle by randomly shuffling the attributes in the mini-batch. Without the cycle-consistency constraint, the generator will not have any incentive to synthesize images that still look the same as the input image.

$$\mathcal{L}_{\text{rec}} = \mathbb{E}\Big[\Big\|x - G\Big(G(x|y^{(b)})|y^{(a)}\Big)\Big\|_1\Big] \tag{4}$$

The reconstruction loss alone produces blurry images since the mean absolute error will tend to produce outputs around the median image. To address this, we include a GAN loss that encourages the outputs to look like the images in the training data. We employ Wassertein's GAN [25] with gradient penalty [26] (WGAN-GP) to train our model to synthesize realistic images. The WGAN loss is expressed mathematically as shown in Eq. 5, where x is the input image, y is the attribute label. The gradient penalty term is shown in Eq. 6 where $\hat{x}$ is a randomly interpolated vector from the of the real and synthesized image.

$$\mathcal{L}_{\text{GAN}} = \mathbb{E}[D_{\text{GAN}}(x)] - \mathbb{E}[D_{\text{GAN}}(G(x|y))] \tag{5}$$

$$\mathcal{L}_{\text{GP}} = \mathbb{E}\big[(\|\nabla_{\hat{x}} D_{\text{GAN}}(\hat{x})\|_2 - 1)^2\big] \tag{6}$$

Lastly, we need to add an objective that would ensure that the generator follows the target attribute specified by the user. We impose this by adding an additional task for the discriminator, which is to classify the attributes present in the input image. We use a simple binary cross-entropy loss, as shown in Eq. 7, since more than one attribute may be present in one image. Our final objective is defined in Eq. 8, where λ_{rec}, λ_{GAN}, λ_{GP}, and λ_{attr} are hyper-parameters that controls the relative importance of the terms.

$$\begin{aligned}\mathcal{L}_{\text{attr}} = \mathbb{E}\big[&y \log D_{\text{attr}}(G(x|y)) \\ &+ (1-y)\log(1 - D_{\text{attr}}(G(x|y)))\big]\end{aligned} \tag{7}$$

$$\begin{aligned}\min_G \max_D \mathcal{L} = &\lambda_{\text{rec}}\mathcal{L}_{\text{rec}} + \lambda_{\text{GAN}}\mathcal{L}_{\text{GAN}} \\ &+ \lambda_{\text{GP}}\mathcal{L}_{\text{GP}} + \lambda_{\text{attr}}\mathcal{L}_{\text{attr}}\end{aligned} \tag{8}$$

TABLE 3. Comparison in terms of execution time. The execution time is measured as the total number of seconds the model takes on a Xiaomi Mi 5s mobile phone. We show the time it takes for each model for different number of transformations (e.g. varying degrees of blonde hair) and different number of changed attributes (simultaneously changing hair color and facial expression counts as two changed attributes).

# of Transformations	# of Changed Attributes	SAT-GAN (ours)	StarGAN [1]	CycleGAN [23]
1	1	**0.2245 ± 0.0058 secs**	0.7528 ± 0.0136 secs	0.3396 ± 0.0019 secs
10	1	**2.2061 ± 0.0605 secs**	7.5634 ± 0.1237 secs	3.4284 ± 0.0334 secs
1	3	**0.2291 ± 0.0049 secs**	0.7608 ± 0.0209 secs	0.3381 ± 0.0023 secs
10	3	**2.1712 ± 0.0559 secs**	7.6388 ± 0.1342 secs	10.449 ± 0.1369 secs

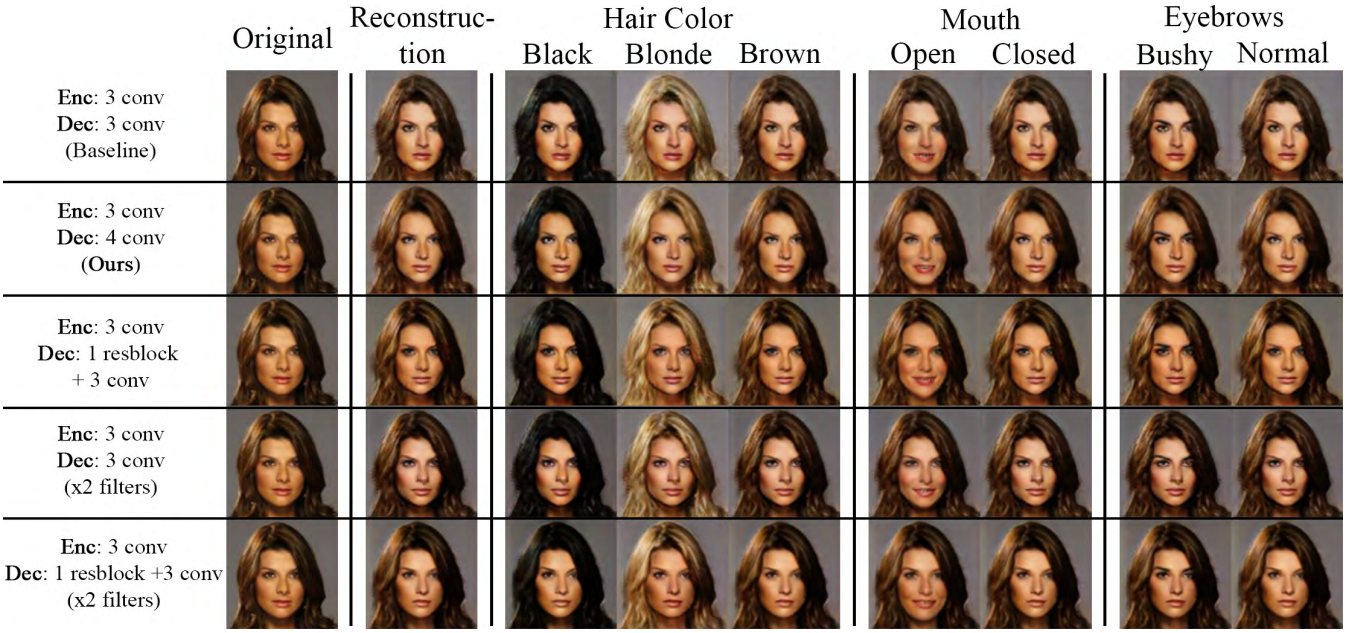

FIGURE 3. This figure shows sample transformations of different network architectures that we explored. We started with the simplest architecture as our baseline and slowly added more layers and filters to increase the capacity. We found that simply increasing the capacity does not lead to significantly better quality results. (This figure is best viewed in color.).

IV. EXPERIMENTAL RESULTS

A. DATASET

For our experiments, we used the CelebFaces Atrributes (CelebA) dataset. It contains 202,599 facial images of celebrities with 10,177 unique identities each annotated with 40 attribute labels. Similar to the epxeriments of Perarnau *et al.* [11], we used a subset of 17 facial attributes that had the most noticeable visual changes. We excluded the attribute "wearing hat" since it is an accessory not a facial attribute. We set aside some images as our test images in order to evaluate the performance of our model on unseen faces. We used the aligned images and center cropped a square patch of size 128×128. All the image intensities were scaled to be in the range of $[-1, 1]$.

B. IMPLEMENTATION DETAILS

We trained SAT-GAN using the Adam optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.999$. We initialized the weights randomly from a Gaussian distribution with 0 mean and 0.02 standard deviation. We used a learning rate of 1×10^{-4} for the discriminator D and 2×10^{-5} for the generator G. We use a mini-batch size of 16 for both the generator and discriminator. In our experiments, we found $\lambda_{\text{GAN}} = 1$, $\lambda_{\text{rec}} = 10$, $\lambda_{\text{attr}} = 1$, and $\lambda_{\text{GP}} = 10$ to work well. The execution times were computed on a Xiaomi Mi 5s.

C. VARYING THE NETWORK ARCHITECTURE

We experimented on various network architectures to find the most suitable one for our application. Figure III-E, shows the results of some of the architectures that we explored. Since processing time and number of parameters are important factors for our goal, we started with the simplest architecture as our baseline where the encoder and decoder each consist of 3 convolutional layers. It can be observed that it already gives satisfactory results. We tried adding more convolutional layers to see its effects. The quality of the transformation slightly improved as we increased the layers, however, the improvements are minimal. We can see in Figure III-E that adding a single convolutional layer removes some of the artifacts on the lower left hair for the blonde attribute. We also experimented on adding residual blocks but while it supposedly has a larger capacity than a normal convolutional layer, it introduces a large amount of parameters and its improvements on the quality of the images are minimal.

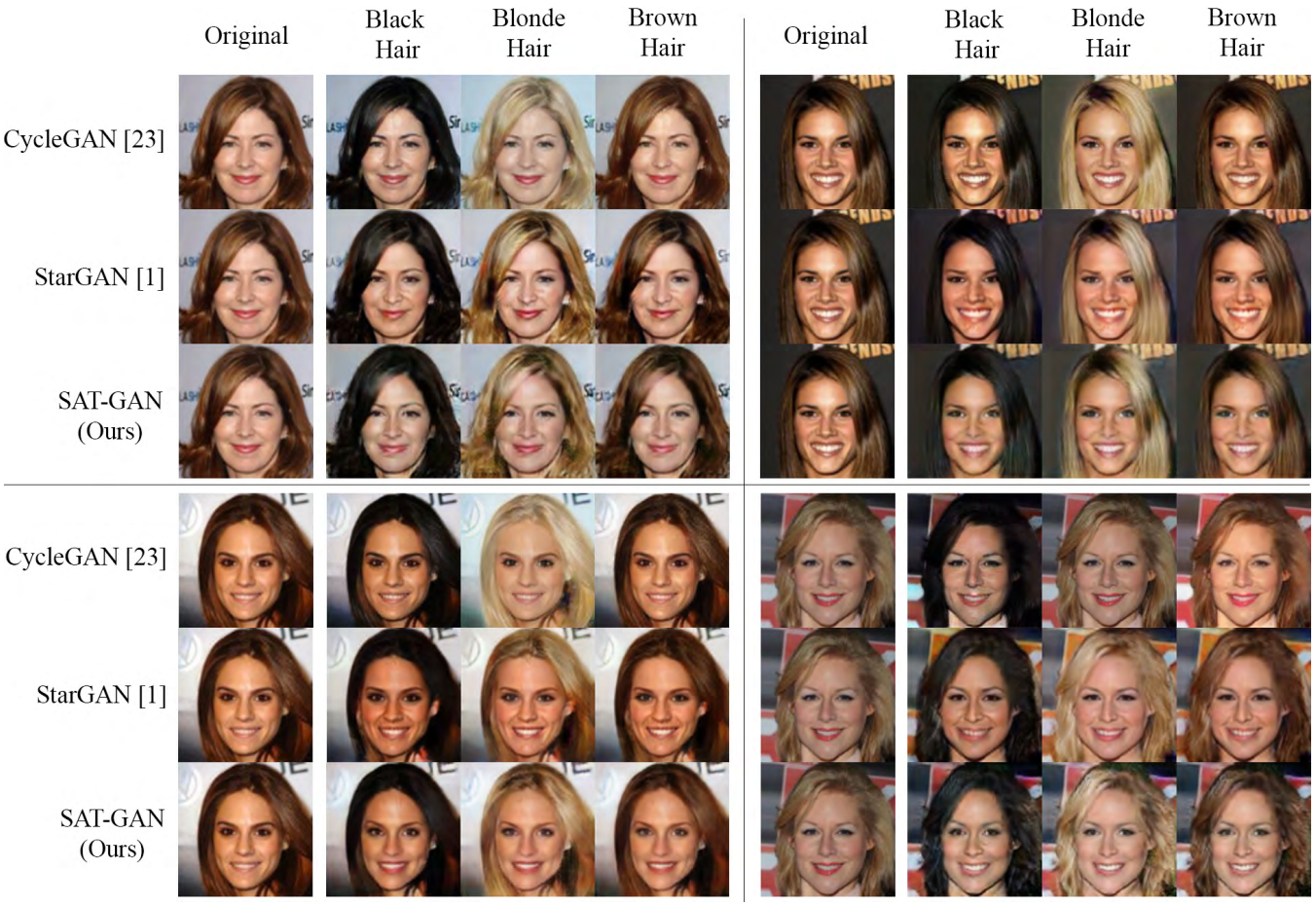

FIGURE 4. This figure shows a comparison of our model with state-of-the-art facial attribute transformation models namely, StarGAN [1] and CycleGAN [23]. We can observe that our method achieves comparable results with significantly fewer parameters. (This figure is best viewed in color.).

Since increasing depth did not seem to have significant effects, we tried to increase the number of filters instead to see if it gives us more improvements. Even after doubling the number of filters per layer (which also doubles the number of parameters), it does not seem to be significantly better. Thus, in our final model, we used 3 convolutional layers for the encoder and 4 convolutional layers in the decoder since it has the least number of parameters that still achieves good quality transformations.

D. COMPARISONS WITH PREVIOUS APPROACHES

We first compare against StarGAN [1] and CycleGAN [23], which are two state-of-the-art models for facial attribute transformation. We used their publicly available code for all the experiments. Figure III-E shows sample transformations of hair colors from the models. Memory requirements and number of parameters of each model are shown in Table 1. We also measured execution times on a Xiaomi MI 5s as shown in Table 3. We measured the time it takes for each model to change multiple facial attributes such as changing hair color together with changing facial expression (as denoted by ''# of changed attributes'' column). We also measured the total time each model takes for performing multiple transformations with different configurations such as different degrees of blonde (denoted by ''# of transformations'' column).

CycleGAN [23] arguably achieves the best quality of transformation. However, it can only perform one attribute transformation per model. If we wanted to transform freely across three hair colors, we would need to train six different models. This means that the number of parameters that we need to store will multiply by the number of permutations among k attributes taken 2 at a time $P(k, 2)$, defined in Eq. 9. Moreover, we would need a separate classifier to determine which model to choose for the transformation, i.e., if the input image has black hair and we want to transform it to blonde, we need to correctly choose the black $\rightarrow$ blonde model and not brown $\rightarrow$ blonde model.

$$P(n, r) = \frac{n!}{(n-r)!} \tag{9}$$

Another drawback of CycleGAN [23] is that its processing time scales with the number of attributes you want to simultaneously apply since it needs to chain multiple models in order to combine multiple attributes. For example,

FIGURE 5. This figure shows a comparison of our model with ModGAN [27] and StarGAN [1] on single and multiple facial attribute transformations. Note that we exclude CycleGAN [23] in this comparison since it is not designed to handle multiple facial attributes simultaneously. (This figure is best viewed in color.).

if we want to apply blonde hair and smiling, we would need to first transform it to blonde hair and then transform the resulting image to smiling. This makes it difficult and impractical to scale to more attributes. In contrast, our model can perform multiple attribute transformation simultaneously using only a single model. It also has constant time

FIGURE 6. This figure shows a comparison of our model with StarGAN [1] on changing facial expressions on the RaFD Dataset [28]. (This figure is best viewed in color.).

FIGURE 7. **This figure shows the capability of our model to perform simultaneous multiple attribute transformation as well as control the degree of these transformations. (This figure is best viewed in color.).**

complexity with respect to the number of attributes being considered.

Visually, our model produces images that are comparable with that of StarGAN [1]. But as shown in Tables 1 and 3, our model is approximately 3× faster and has 4× fewer parameters. Moreover, our encoder-decoder architecture is computationally more efficient than StarGAN [1] when we perform more transformations on the same image since StarGAN [1] has to process the entire image all over again for every transformation, while our model can cache the encoded image and only needs to perform the decoding, as illustrated in Figure 1.

We also compare against a more recent work on multiple facial attribute transformation called ModGAN [27], as shown in Figure 5. Since ModGAN [27] has no publicly available code, we compare against the image examples they provided in their paper. We can observe that our model (SAT-GAN) is able to better preserve the features of the original image while still being able to produce the desired attributes. This is most noticeable in the first example of Figure 5. In changing the expression, ModGAN [27] also changed the face and therefore changing the identity of the person, while our method is able to preserve the identity of the person. We hypothesize that this is due to the errors compounded during the chaining of multiple modules in ModGAN [27]. StarGAN [1], on the other hand, is not able to preserve the skin tones well, as evidenced in the ''Black Hair'' and ''Blonde Hair'' attributes, where the skin tone becomes darker and lighter respectively. In contrast, our model has a more consistent skin tone across all attribute transformations. We hypothesize that the consistency of our model is due to the encoder-decoder design where the encoder learns an attribute agnostic representation that preserves the features of the input image regardless of the desired target attributes.

To further validate our model, we experimented on the RaFD Facial Expressions dataset [28]. Figure 6 shows the comparison of our model with StarGAN [1]. We can observe that both our model (SAT-GAN) and StarGAN [1] can handle different poses such as facing left, facing right, and facing front. In terms of transformation quality, we can observe that there are minimal differences between the two methods.

These experiments show that our model is able to achieve the same image quality and more consistent transformations as compared to CycleGAN [23], ModGAN [27], and StarGAN [1], even though our model has much lesser parameters and computational costs.

TABLE 4. This table shows the aggregated votes of participants for every image in the user-study comparing our method (SAT-GAN) against StarGAN [1]. The "instances of majority votes" column shows the number of images where a model has garnered the majority votes.

Image Number	Img-1	Img-2	Img-3	Img-4	Img-5	Img-6	Img-7	Img-8	Img-9	Img-10
SAT-GAN (ours)	9	11	**20**	**15**	3	**21**	**11**	10	**19**	8
StarGAN [1]	**17**	**15**	8	11	**20**	6	10	**14**	7	**21**
No Preference (both look similar)	4	4	2	4	7	3	9	6	4	1
Image Number	Img-11	Img-12	Img-13	Img-14	Img-15	Img-16	Img-17	Img-18	Img-19	Img-20
SAT-GAN (ours)	4	**15**	7	10	**23**	**17**	10	4	**16**	**23**
StarGAN [1]	**22**	12	**15**	**14**	5	5	**15**	**19**	8	3
No Preference (both look similar)	4	3	8	6	2	8	5	7	6	4

	Total votes	Instances of majority votes
SAT-GAN (ours)	256	10
StarGAN [1]	247	10
No Preference (both look similar)	97	-

TABLE 5. This table shows the aggregated preferences of each participant in the user-study comparing our method (SAT-GAN) against StarGAN [1]. The "instances of majority votes" column shows the number of times a model has garnered the majority of the votes.

Participant	P-1	P-2	P-3	P-4	P-5	P-6	P-7	P-8	P-9	P-10
SAT-GAN (ours)	**13**	**11**	**10**	6	9	**15**	**12**	**14**	2	**10**
StarGAN [1]	4	3	6	**13**	**10**	5	8	6	**16**	7
No Preference (both look similar)	3	6	4	1	1	0	0	0	2	3
Participant	P-11	P-12	P-13	P-14	P-15	P-16	P-17	P-18	P-19	P-20
SAT-GAN (ours)	**7**	**13**	**14**	6	**10**	5	6	7	**14**	**11**
StarGAN [1]	6	7	6	**8**	4	**12**	**10**	**11**	6	6
No Preference (both look similar)	7	0	0	6	6	3	4	2	0	3
Participant	P-21	P-22	P-23	P-24	P-25	P-26	P-27	P-28	P-29	P-30
SAT-GAN (ours)	3	6	**8**	5	3	5	**13**	6	**11**	1
StarGAN [1]	**10**	**11**	6	**7**	**11**	**11**	5	**9**	8	**15**
No Preference (both look similar)	7	3	6	8	6	4	2	5	1	4

	Total votes	Instances of majority votes
SAT-GAN (ours)	256	16
StarGAN [1]	247	14
No Preference (both look similar)	97	-

TABLE 6. This table shows the aggregated votes of participants for every image in the user-study comparing our method (SAT-GAN) against CycleGAN [23]. The "instances of majority votes" column shows the number of images where a model has garnered the majority votes.

Image Number	Img-1	Img-2	Img-3	Img-4	Img-5	Img-6	Img-7	Img-8	Img-9	Img-10
SAT-GAN (ours)	**23**	14	**20**	13	9	**17**	**17**	11	**15**	**19**
CycleGAN [23]	7	**15**	10	**15**	**21**	12	10	**15**	13	9
No Preference (both look similar)	0	1	0	2	0	1	3	4	2	2
Image Number	Img-11	Img-12	Img-13	Img-14	Img-15	Img-16	Img-17	Img-18	Img-19	Img-20
SAT-GAN (ours)	10	**18**	13	13	**16**	9	12	12	**15**	11
CycleGAN [23]	**19**	11	**16**	**16**	10	**20**	**17**	**16**	12	**18**
No Preference (both look similar)	1	1	1	1	4	1	1	2	3	1

	Total votes	Instances of majority votes
SAT-GAN (ours)	287	9
CycleGAN [23]	282	11
No Preference (both look similar)	31	-

E. CONTROLLING THE DEGREE OF TRANSFORMATIONS

Another good property of our model is that we can work on multiple facial attribute simultaneously while being able to control the degree of each facial attributes, unlike CycleGAN [23] where the transformations are fixed for every model. This is an emergent property of training the model with attribute vectors. Our network learns to associate different portions of the space in the lower dimensional latent space to different attributes. Since this latent space is continuous, we can move the encoded image towards the direction of a particular attribute in order to enhance or degrade the effects of the attribute. This can be done by using a floating point number for the attribute labels instead of just ones and zeros to indicate presence and absence of an attribute. Intuitively, a number greater than one would emphasize the attribute more and a number less than one would weaken the attribute effect. An example of this is shown in Figure 7, where we could gradually change the hair color, make the mouth slowly open and also make the eyebrows bushier. Note that if we move too far in this latent space, there is no guarantee that

TABLE 7. This table shows the aggregated preferences of each participant in the user-study comparing our method (SAT-GAN) against CycleGAN [23]. The "instances of majority votes" column shows the number of times a model has garnered the majority of the votes.

Participant	P-1	P-2	P-3	P-4	P-5	P-6	P-7	P-8	P-9	P-10
SAT-GAN (ours)	**13**	5	**13**	4	**12**	**11**	**18**	**14**	**15**	**13**
CycleGAN [23]	4	**15**	7	**16**	6	6	2	6	5	7
No Preference (both look similar)	3	0	0	0	2	3	0	0	0	0
Participant	P-11	P-12	P-13	P-14	P-15	P-16	P-17	P-18	P-19	P-20
SAT-GAN (ours)	9	2	**11**	4	**16**	2	8	**11**	6	9
CycleGAN [23]	**11**	**15**	9	4	4	**18**	**12**	8	**13**	**11**
No Preference (both look similar)	0	3	0	**12**	0	0	0	1	1	0
Participant	P-21	P-22	P-23	P-24	P-25	P-26	P-27	P-28	P-29	P-30
SAT-GAN (ours)	1	0	**15**	1	**14**	**12**	**15**	9	**13**	**11**
CycleGAN [23]	**19**	**20**	5	**13**	6	8	5	**11**	7	9
No Preference (both look similar)	0	0	0	6	0	0	0	0	0	0

	Total votes	Instances of majority votes
SAT-GAN (ours)	287	17
CycleGAN [23]	282	12
No Preference (both look similar)	31	1

the image will look realistic since it will over exaggerate the attributes.

F. USER STUDY

We conducted a user study to further evaluate our model's performance. We transformed 20 images with unique identities using SAT-GAN, CycleGAN [23] and StarGAN [1]. We conducted two separate surveys where one compares our method against StarGAN [1] and the other compares our method against CycleGAN [23]. There were 30 participants on each survey. Each of them were asked to choose which image looks more realistic. A third option, ''No Preference'', was included to account for the case where the two images looks equally good and the participant had no preference between the two. Note that the participants did not know which image came from which model. Tables 4 and 5 show the results of our survey in comparison with StarGAN [1], while Tables 6 and 7 show the results of our survey in comparison with CycleGAN [23]. It can be observed that their preferences are roughly split in half with several of the responses ''No Preference'' between the two. This verifies that our model's outputs are on-par with StarGAN [1] and CycleGAN [23] in terms of quality of the generated images, but our proposed model requires significantly lesser computational resources.

V. CONCLUSION

In this paper, we proposed SAT-GAN, a simple attribute transformation model that can simultaneously transform multiple facial attributes. Our simple model produces results that are comparable to the state-of-the-art models but with up to 4× fewer parameters and 3× faster execution time. The encoder-decoder architecture design effectively saves half of the computations for every succeeding transformation, making it more practical for mobile applications.

ACKNOWLEDGMENT

(Jonathan Hans Soeseno and Daniel Stanley Tan contributed equally to this work.)

REFERENCES

[1] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 8789–8797.

[2] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[3] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Comput. Graph. Appl.*, vol. 21, no. 5, pp. 34–41, Sep./Oct. 2001.

[4] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 689–694, 2004.

[5] D. Guo and T. Sim, "Digital face makeup by example," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 73–79.

[6] L. Liu, J. Xing, S. Liu, H. Xu, X. Zhou, and S. Yan, "Wow! You are so beautiful today!" *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 11, no. 1s, 2014, Art. no. 20.

[7] W.-S. Tong, C.-K. Tang, M. S. Brown, and Y.-Q. Xu, "Example-based cosmetic transfer," in *Proc. IEEE 15th Pacific Conf. Comput. Graph. Appl. (PG)*, Oct./Nov. 2007, pp. 211–218.

[8] F. Yang, J. Wang, E. Shechtman, L. Bourdev, and D. Metaxas, "Expression flow for 3D-aware face component transfer," *ACM Trans. Graph.*, vol. 30, no. 4, 2011, Art. no. 60.

[9] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.

[10] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.

[11] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez, "Invertible Conditional GANs for image editing," in *Proc. NIPS Workshop Adv. Training*, 2016, pp. 1–9.

[12] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "DRAW: A recurrent neural network for image generation," in *Proc. 32nd Int. Conf. Mach. Learn. Res.*, vol. 37. Lille, France: PMLR, Jul. 2015, pp. 1462–1471.

[13] L. Tran, X. Yin, and X. Liu, "Disentangled representation learning GAN for pose-invariant face recognition," in *Proc. CVPR*, 2017, vol. 4, no. 5, pp. 1415–1424 .

[14] N. Shan, D. S. Tan, M. S. Denekew, Y.-Y. Chen, W.-H. Cheng, and K.-L. Hua, "Photobomb defusal expert: Automatically remove distracting people from photos," *IEEE Trans. Emerg. Topics Comput. Intell.*, to be published.

[15] T.-C. Lin *et al.*, "Pedestrian detection from lidar data via cooperative deep and hand-crafted features," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 1922–1926.

[16] D. S. Tan, W.-Y. Chen, and K.-L. Hua, "DeepDemosaicking: Adaptive image demosaicking via multiple deep fully convolutional networks," *IEEE Trans. Image Process.*, vol. 27, no. 5, pp. 2408–2419, May 2018.

[17] Y.-C. Chen, D. S. Tan, W. Cheng, and K.-L. Hua, "3D object completion via class-conditional generative adversarial network," in *Proc. Int. Conf. Multimedia Modeling* Cham, Switzerland: Springer, 2019, pp. 54–66.
[18] D. P. Kingma and M. Welling. (2013). "Auto-encoding variational Bayes." [Online]. Available: https://arxiv.org/abs/1312.6114
[19] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 597–613.
[20] M. F. Mathieu, J. J. Zhao, J. Zhao, A. Ramesh, P. Sprechmann, and Y. LeCun, "Disentangling factors of variation in deep representation using adversarial training," in *Proc. Adv. Neural Inf. Process. Syst.*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. New York, NY, USA: Curran Associates, Inc., 2016, pp. 5040–5048.
[21] T. Salimans *et al.*, "Improved techniques for training GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. New York, NY, USA: Curran Associates, Inc., 2016, pp. 2234–2242.
[22] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1125–1134.
[23] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2223–2232.
[24] G. Antipov, M. Baccouche, and J.-L. Dugelay. (2017). "Face aging with conditional generative adversarial networks." [Online]. Available: https://arxiv.org/abs/1702.01983
[25] M. Arjovsky, S. Chintala, and L. Bottou. (2017). "Wasserstein GAN." [Online]. Available: https://arxiv.org/abs/1701.07875
[26] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5767–5777.
[27] B. Zhao, B. Chang, Z. Jie, and L. Sigal, "Modular generative adversarial networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 150–165.
[28] O. Langner, R. Dotsch, G. Bijlstra, D. H. J. Wigboldus, S. T. Hawk, and A. van Knippenberg, "Presentation and validation of the radboud faces database," *Cognit. Emotion*, vol. 24, no. 8, pp. 1377–1388, 2010.

JONATHAN HANS SOESENO received the B.Eng. degree in informatics engineering from Petra Christian University. He is currently pursuing the M.Sc. degree with the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology. His research interests include digital image processing and deep learning applied to computer vision.

DANIEL STANLEY TAN received the M.S. degree in computer science from De La Salle University. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology. His research interests include digital image processing and deep learning applied to computer vision.

WEN-YIN CHEN received the M.A. degree in art and design from University for the Creative Arts, U.K. She is currently pursuing the Ph.D. degree with the Department of Arts and Design, National Taipei University of Education. Her research interests include 3D design, modeling, and digital image processing.

KAI-LUNG HUA received the B.S. degree in electrical engineering from National Tsing Hua University, Hsinchu, Taiwan, in 2000, the M.S. degree in communication engineering from National Chiao Tung University, Hsinchu, in 2002, and the Ph.D. degree from the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, in 2010. Since 2010, he has been with the National Taiwan University of Science and Technology, where he is currently a Professor with the Department of Computer Science and Information Engineering. His current research interests include digital image and video processing, computer vision, and machine learning. He is a member of Eta Kappa Nu and Phi Tau Phi. He was a recipient of the MediaTek Doctoral Fellowship. He received several research awards, including the 2018 Young Scholar Award from Taiwan Tech, the Top Performance Award from the 2017 ACM Multimedia Grand Challenges, the Top 10% Paper Award from the 2015 IEEE International Workshop on Multimedia Signal Processing, the Second Award from the 2014 ACM Multimedia Grand Challenge, the Best Paper Award from the 2013 IEEE International Symposium on Consumer Electronics, and the Best Poster Paper Award from the 2012 International Conference on 3D Systems and Applications.