Semantic Component Decomposition for Face Attribute Manipulation

Ying-Cong Chen[1] Xiaohui Shen[4] Zhe Lin[3] Xin Lu[3] I-Ming Pao[3] Jiaya Jia[1,2]
[1]The Chinese University of Hong Kong [2]Tencent Youtu Lab [3]Adobe Research [4]ByteDance AI Lab
{ycchen,leojia}@cse.cuhk.edu.hk shenxiaohui@gmail.com {zlin, xinl, pao}@adobe.com

Abstract

Deep neural network-based methods were proposed for face attribute manipulation. There still exist, however, two major issues, i.e., insufficient visual quality (or resolution) of the results and lack of user control. They limit the applicability of existing methods since users may have different editing preference on facial attributes. In this paper, we address these issues by proposing a semantic component model. The model decomposes a facial attribute into multiple semantic components, each corresponds to a specific face region. This not only allows for user control of edit strength on different parts based on their preference, but also makes it effective to remove unwanted edit effect. Further, each semantic component is composed of two fundamental elements, which determine the edit effect and region respectively. This property provides fine interactive control. As shown in experiments, our model not only produces high-quality results, but also allows effective user interaction.

1. Introduction

The popularity of sharing selfies and portrait photos online motivates the rapid development of face edit tools. Facial attribute manipulation is especially attractive with the functions of adding/removing face accessories, such as facial hair and eyeglasses, and/or changing intrinsic face properties, such as age and gender.

Facial attribute manipulation has attracted great interest [10, 2, 4, 3, 30, 26, 22], because of the great chance it brings to research and real-world application. Early work focuses on specific attributes of facial hair generation [1], expression change [31, 22, 30], beautification/de-beautification [14, 6], aging [10], etc. These approaches are designed for specific tasks, and require prior knowledge that is not applicable to new editing tasks.

Recently, with the development of deep neural networks, especially generative adversarial networks, several general face attribute manipulation frameworks were proposed [20, 19, 12, 16, 17, 32, 13, 8]. These approaches take facial attribute edit as an unpaired learning task, and thus are capable of handling different attributes by only changing the data. Our method can be categorized into this group, which aims to provide a general solution for different facial attributes.

Limitation of Existing Solutions Since most facial attributes of a person are immutable within a short period of time, collecting paired images with only desired change seems difficult and costly. Most frameworks resort to the Generative Adversarial Network (GAN) [8, 32, 29, 17, 27, 16, 11, 13, 20, 24, 35, 25, 7, 19, 35, 34], which is popular for unsupervised learning. However, despite recent progress [9, 25], training a suitable GAN is still difficult.

In addition, there is no effective indicator to monitor the training process, which makes it hard to choose the "best" model during training. These difficulties could cause non-optimal training and produce unsatisfying results. Note that when the result is not as expected, there is few options to guide the system to fix the problem.

Our Solution Inspired by CapsuleNet [18] that divides the final prediction to smaller parts, we seek for an approach that divides a high-level attribute edit into multiple semantic components, where each works on one semantic region of a human face. With this scheme, our model allows component-level user control, which is more flexible than existing solutions.

Moreover, we decompose each component into two factors, i.e., the *attention map* and the *semantic painter*. The attention map highlights altered region of the corresponding component, and the semantic painter corresponds to the kind of effect applied to that region. These two factors enable us to manipulate each component by changing either the edit region or the semantic painter. This further allows interactive manipulation.

It is notable that our system conceptually mimics the way image-editing software works: different semantic components can be viewed as "layers" in Photoshop, and all layers jointly compose the final edit result. During testing, our model generates an initial high-quality result for the target attribute. If users have other requirements, they can further adjust it in different levels of our system output.

Despite the simple and natural work-flow, decomposing

a high-level face attribute into different components is not trivial. For example, changing the age of a person may need to edit his/her eyes, nose, mouth, skin, etc, which are too complicated and tedious for people to label respectively. Our model instead learns such decomposition in an end-to-end manner using unpaired data.

Specifically, our model is composed of three parts, i.e., AttentionNet, PainterNet, and FusionNet. AttentionNet produces attention maps that figure out the edit region. PainterNet produces a vector that controls the edit effect on the corresponding region, while FusionNet combines each pair of edit region and semantic painter to produce the final result. Compared with GAN, this training strategy is more robust and works generally on different resolutions and styles. Our main contributions are the following.

- We propose a semantic-component-based framework for face attribute manipulation. It is the first attempt to learn semantic components from high-level attributes.
- Our model improves edit quality. It benefits from component decomposition that partitions an edit into relevant and irrelevant parts, where the latter can be removed for more dedicated focus on important regions.
- Our model also allows adjusting edit strength of different components and manipulating edit effect on each component. Thus it shows an effective way for interactive editing.

2. Related Work

Early Solutions Face edit has been studied for years [10, 2, 4, 3, 30, 26, 22]. Most early work is for specific tasks, such as face aging [10], relighting [26, 2], and expression editing [30]. Our method is contrarily a general framework for face attribute manipulation and is by nature different.

Generative Adversarial Network Several recent methods utilize GAN to build general face attribute manipulation frameworks. Face attribute disentangling [8, 32, 29, 17, 27, 16, 11, 13, 20, 24, 28] decomposes human faces in a deep space where part of the features serve as attribute tags. By altering these tags, the original face can be manipulated accordingly. However, disentangling is not easy to achieve, and reconstructing face images based on the altered tags may not be correct.

A simpler way to edit face attributes is to take it as a set-to-set image transformation task. Cycle-GAN [34] and its variants [35, 25, 7, 19, 35] are typical methods. This line avoids face disentangling and thus simplifies the problem. Nevertheless, the systems are hard to train and the results may contain visual artifacts. SaGAN [33] used spatial attention to avoid problems of irrelevant regions. It focuses on local attributes like facial hairs or eyeglasses, while our method can handle both local and global attributes like face aging. Further, our method discovers sub-properties for global attributes, allowing controlling each local region conveniently. HDGAN [25] and Progressive GAN [9] achieved much better results than previous GANs. But HDGAN does not edit images based on high-level attributes, while progressive GAN is designed for image generation, which handles tasks different from ours.

Deep Feature Interpolation In addition to the GAN based frameworks, deep feature interpolation (DFI) [23] was employed for face attribute manipulation. By shifting deep features of the query image with certain *attribute tensor*, the semantic facial attributes can be updated accordingly. The drawback is that estimation of shifting tensors may be noisy [5], which cause undesired changes. Also, the computational burden is high. In [5], these problems were alleviated by using a three-layer CNN to learn the attribute shifting tensor.

Our work is related to [5], and yet is fundamentally different in two aspects. First, in [5], a shallow CNN was used to avoid fitting noise. This also prevents the model from learning target attributes precisely. In contrast, our approach decomposes the attribute shifting tensor into several components, so that the noisy components can be suppressed individually. This advantage allows our model to use more powerful structures. Second, our model supports more user interaction while [5] only allows changing the overall edit strength. Thus our model can better meet different users' preference.

3. Our Method

Since face manipulation with traditional image processing tools only considers pixel-level information, we propose a model working in semantic level, which is more user-friendly.

Suppose there are two face domains that differ on certain semantic attribute $\mathcal{S}$. We denote the negative samples as $\mathcal{S}^-$ and the positive ones as $\mathcal{S}^+$. Our goal is to transfer image properties from $\mathcal{S}^-$ to $\mathcal{S}^+$. Note that the training images are collected from daily photos, which might vary in background, illumination, viewpoint, etc. We expect the model not to alter any region and property irrelevant to $\mathcal{S}$.

3.1. The Baseline Model

Generally speaking, face attribute edit can be represented as $I_{\mathcal{S}^+} = \mathcal{O}(I_{\mathcal{S}^-}, \mathcal{V}_{\mathcal{S}})$, where $I_{\mathcal{S}^-}, I_{\mathcal{S}^+} \in \mathbb{R}^{H\times W\times 3}$ are images with attribute $\mathcal{S}^-$ and $\mathcal{S}^+$, H and W are the height and width, $\mathcal{V}_{\mathcal{S}}$ denotes an "attribute tensor" that carries information about the attribute S, $\mathcal{O}(\cdot)$ denotes a function that transforms $I_{\mathcal{S}^-}$ to $I_{\mathcal{S}^+}$ based on $\mathcal{V}_{\mathcal{S}}$. By changing different $\mathcal{V}_{\mathcal{S}}$, corresponding attributes can be manipulated accordingly.

It is found that $\mathcal{O}(I_{\mathcal{S}^-}, \mathcal{V}_{\mathcal{S}})$ can be simplified to linear interpolation in proper deep space [23], i.e.,

$$\phi(I_{\mathcal{S}^+}) = \phi(I_{\mathcal{S}^-}) + \lambda \mathcal{V}_{\mathcal{S}}, \tag{1}$$

where $\phi(I_{\mathcal{S}^-}), \phi(I_{\mathcal{S}^+}) \in \mathbb{R}^{h \times w \times c}$ are deep feature maps of $I_{\mathcal{S}^-}$ and $I_{\mathcal{S}^+}$ respectively. h, w and c are the height, width and channel numbers of the feature map respectively. $\mathcal{V}_{\mathcal{S}} \in \mathbb{R}^{h \times w \times c}$ controls the shift direction in the deep space. $\lambda \in (0, \infty)$ controls the edit strength. According to [23], $\phi(\cdot)$ can be defined by a pretrained VGG [21] network. After computing Eq. (1), $I_{\mathcal{S}^+}$ is obtained by inverting $\phi(I_{\mathcal{S}^+})$ through back-propagation [23] or training an inversion network [5].

As different $I_{\mathcal{S}^-}$ could appear quite differently, $\mathcal{V}_{\mathcal{S}}$ should be adaptive to $I_{\mathcal{S}^-}$. DFI [23] used K neighbors of $I_{\mathcal{S}^-}$ to compute $\mathcal{V}_{\mathcal{S}}$. Specifically, it is calculated as

$$\mathcal{V}_{\mathcal{S}} = \frac{1}{K} \sum_{i \in N_K^{\mathcal{S}^+}} \phi(I_i) - \frac{1}{K} \sum_{i \in N_K^{\mathcal{S}^-}} \phi(I_i), \tag{2}$$

where $N_K^{\mathcal{S}^+}/N_K^{\mathcal{S}^-}$ refer to K positive/negative nearest neighbors of the query sample respectively. This averaging operation in Eq. (2) aims at suppressing irrelevant changes apart from the target attribute $\mathcal{S}$ [23, 5].

In spite of the insightful architecture and several decent results, this model has two drawbacks as follows.

1. Simply using averaging in Eq. (2) does not suppress all undesired changes (as discussed in Section 4.1). Although learning $\mathcal{V}_{\mathcal{S}}$ with a shallow CNN alleviates this problem [5], quality of attribute is sacrificed.

2. It does not allow altering edit strength of each part, which hinders friendly edit interaction by users, especially considering individual users may have their respective preference in face edit.

To address these issues, we propose a component-based model, which decomposes $\mathcal{V}_{\mathcal{S}}$ into different components $\mathcal{V}_{\mathcal{S}_i}, 1 \leq i \leq k$. Each component corresponds to one kind of change. Thus users can control the edit strength separately. With this design, the undesirable changes can be removed by simply setting the edit strength to 0 regarding these components. Our model also supports fine-grained component edit, where users can finely adjust edit style and region for each component $\mathcal{V}_{\mathcal{S}_i}$.

3.2. Key Components of Our Model

Generally, interactive digital image edit requires users to select a "painter", then applies it to certain regions of the image. This natural interaction philosophy prompts us to consider $\mathcal{V}_{\mathcal{S}}$ in Eq. (2) as the result of applying one or several different "painters" on distinct regions of face image $I_{\mathcal{S}^-}$. Specifically, we assume that

$$\mathcal{V}_{\mathcal{S}} = \mathcal{F}(\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_k, \mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_k), \tag{3}$$

where $\mathcal{P}_i$ denotes the i-th "Painter". $\mathcal{M}_i \in \mathbb{R}^{h \times w \times c}$ is the mask that defines the region where $\mathcal{P}_i$ acts on. $\mathcal{F}(\cdot)$ denotes the network that predicts $\mathcal{V}_{\mathcal{S}}$ based on $\mathcal{P}$ and $\mathcal{M}$.

Recall that $\mathcal{V}_{\mathcal{S}}$ should be adaptive to the input image $\phi(I_{\mathcal{S}^-})$ [23, 5], we further let

$$\mathcal{P}_i = \mathcal{F}_{\mathcal{P}_i}(\phi(I_{\mathcal{S}^-})), \ \mathcal{M}_i = \mathcal{F}_{\mathcal{M}_i}(\phi(I_{\mathcal{S}^-})), \tag{4}$$

where $\mathcal{F}_{\mathcal{P}_i}(\cdot)$ and $\mathcal{F}_{\mathcal{M}_i}(\cdot)$ are neural networks that predict $\mathcal{P}_i$ and $\mathcal{M}_i$ based on $\phi(I_{\mathcal{S}^-})$ respectively. In practice, $\mathcal{V}_{\mathcal{S}}$ can be estimated with Eq. (2). This process makes the parameters of $\mathcal{F}$, $\mathcal{F}_{\mathcal{P}_i}$ and $\mathcal{F}_{\mathcal{M}_i}$ be learned in an end-to-end manner – that is, we learn to predict $\mathcal{V}_{\mathcal{S}}$ based on $\phi(I_{\mathcal{S}^-})$.

Since different pairs of $\mathcal{P}_i$ and $\mathcal{M}_i$ are expected to be unrelated, Eq. (3) is further simplified to

$$\mathcal{V}_{\mathcal{S}} = \sum_{i=1}^{k} \mathcal{V}_{\mathcal{S}_i}, \tag{5}$$

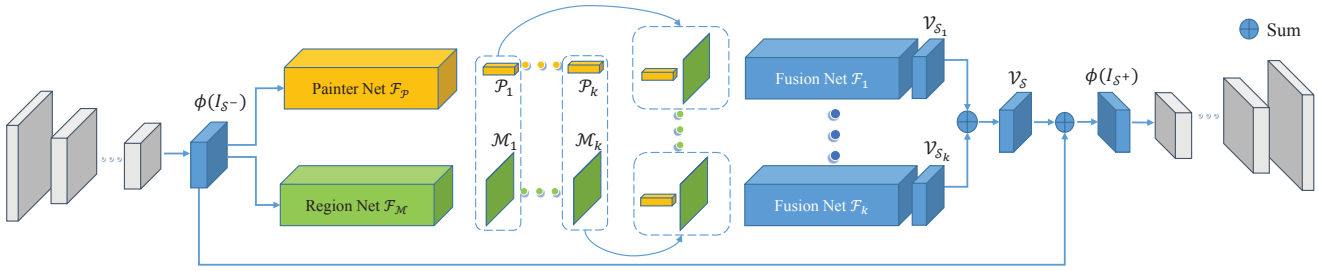
where $\mathcal{V}_{\mathcal{S}_i} = \mathcal{F}_i(\mathcal{F}_{\mathcal{P}_i}(\phi(I_{\mathcal{S}^-})), \mathcal{F}_{\mathcal{M}_i}(\phi(I_{\mathcal{S}^-})))$. The $\mathcal{F}_i(\cdot)$ is a *Fusion Network* that predicts $\mathcal{V}_{\mathcal{S}_i}$ based on the estimated $\mathcal{P}_i$ and $\mathcal{M}_i$. Here, a high-level attribute tensor $\mathcal{V}_{\mathcal{S}}$ is viewed as a linear combination of different components $\mathcal{V}_{\mathcal{S}_i}, 1 \leq i \leq k$. Each component $\mathcal{V}_{\mathcal{S}_i}$ handles only *one* specific kind of effect (determined by $\mathcal{P}_i$) on the corresponding region (determined by $\mathcal{M}_i$).

Spatial/Channel Information Concentration Note that in our model, $\mathcal{P}_i$ and $\mathcal{M}_i$ have precise physical meaning. As $\mathcal{P}_i$ plays the role of "painter", it should carry only information about the "effect" instead of the spatial region. On the contrary, $\mathcal{M}_i$ denotes the location where $\mathcal{P}_i$ acts on. Thus it should contain only spatial information instead of the effect data. Hence, we design the shape of $\mathcal{P}_i$ and $\mathcal{M}_i$ as Eq. (6) to remove redundant information.

$$\mathcal{P}_i \in \mathbb{R}^{1 \times 1 \times c}, \ \mathcal{M}_i \in \mathbb{R}^{h \times w \times 1}. \tag{6}$$

In the Fusion Network $\mathcal{F}_i(\cdot)$, $\mathcal{P}_i$ is spatially repeated to $\mathbb{R}^{h \times w \times c}$ and concatenated with $\mathcal{M}_i$. $\mathcal{V}_i$ is predicted with this concatenated feature. Since $\mathcal{P}_i$ and $\mathcal{M}_i$ together determine $\mathcal{V}_{\mathcal{S}_i} \in \mathbb{R}^{h \times w \times c}$, this setting forces them to consider spatial and channel data respectively. As a result, $\mathcal{P}_i$ and $\mathcal{M}_i$ encode different types of information as expected.

Also, note that one component is conditioned on one painter vector $\mathcal{P}_i$. This enforces each component to handle only one effect, and makes corresponding regions located correctly rather than spread arbitrarily to unrelated regions. This simplifies and clarifies the semantic meaning of $\mathcal{V}_{\mathcal{S}_i}$ for users and makes it easy to adjust the corresponding component later if needed.



(a) VGG Encoder (b) $\mathcal{F}_{\mathcal{P}}$ and $\mathcal{F}_{\mathcal{M}}$ (c) Painter $\mathcal{P}_i$ and Region $\mathcal{M}_i$ (d) Fusion Net $\mathcal{F}$ (e) VGG Decoder

Figure 1. Pipeline illustration. (a) is a pretrained VGG to encode the input image in the deep space $\phi(I_{\mathcal{S}^-})$ where weights are fixed. (b) is our proposed Painter Network $\mathcal{F}_{\mathcal{P}}$ and $\mathcal{F}_{\mathcal{M}}$. (c) illustrates the painter vectors $\mathcal{P}_i$ and region maps $\mathcal{M}_i$. $\mathcal{P}_i$ and $\mathcal{M}_i$ are paired before fed into the Fusion Network. (d) is the Fusion Network $\mathcal{F}$ that fuses $\mathcal{P}_i$ and $\mathcal{M}_i$ to produce semantic component $\mathcal{V}_{\mathcal{S}_i}$. All semantic components are finally summed to form $\mathcal{V}_{\mathcal{S}}$. After that, $\phi(I_{\mathcal{S}^+})$ is computed by Eq. (1). (e) is the VGG decoder trained to invert $\phi(I_{\mathcal{S}^+})$ to $I_{\mathcal{S}^+}$. The architecture and training of the decoder follow [5].

Towards Non-overlapping Regions In addition to restricting each semantic component to only encode one effect, another key issue is to ensure that each facial region is only affected by one dominant component. Otherwise, users could be confused if one edit alters many components in following interactive manipulation. This isolation requires to keep all region masks $\mathcal{M}_i, k = 1, 2, \cdots, k$ non-overlapping.

Specifically, let $\boldsymbol{\mathcal{M}}(m,n) = [\mathcal{M}_1(m,n), \mathcal{M}_2(m,n), \cdots, \mathcal{M}_i(m,n)]$, where (m,n) is the location. The non-overlapping constraint is formulated as $||\boldsymbol{\mathcal{M}}(m,n)||_0 = 1$, which indicates that for each location, $\boldsymbol{\mathcal{M}}(m,n)$ is encoded as a one-hot vector.

Intuitively, it is achieved by setting the maximum element of $\boldsymbol{\mathcal{M}}(m,n)$ to 1 and all others to 0. But unfortunately, this scheme is not differentiable. So we seek for a soft version to approximate it, so that back-propagation gradients exist to train the network. Our approximation is expressed as

$$\mathcal{M}_i(m,n) \leftarrow \frac{e^{\beta \mathcal{M}_i(m,n)}}{\sum_{j=1}^{k} e^{\beta \mathcal{M}_j(m,n)}}, \tag{7}$$

where β is a positive scalar that controls the degree of sharpness. A large β exaggerates the difference of elements in $\boldsymbol{\mathcal{M}}(m,n)$. In extreme cases, the maximum element becomes 1 while the others are 0, making $\boldsymbol{\mathcal{M}}(m,n)$ a nearly one-hot vector. It is noted that a very large β may cause the network difficult to train. In this paper, we set $\beta = 2$.

The softly encoded region masks are also leveraged as attention maps to highlight the dominant components and suppress all others. This can be formulated as

$$\mathcal{V}_{\mathcal{S}_i}(m,n) \leftarrow \mathcal{M}_i(m,n)\mathcal{V}_{\mathcal{S}_i}(m,n), \tag{8}$$

where $\mathcal{V}_{\mathcal{S}_i}(m,n) \in \mathbb{R}^{1\times 1\times c}$ refers to the (m,n) location of $\mathcal{V}_{\mathcal{S}_i}$. With these strategies, for each location, the dominant component is highlighted while the others are suppressed.

3.3. Network Architecture

Our network architecture is based on the key components discussed in Section 3.2. Specifically, we use a Painter Network $\mathcal{F}_{\mathcal{P}}(\phi(I_{\mathcal{S}^-}))$ to produce k vectors that control the effect of semantic components. We also propose a Attention Network $\mathcal{F}_{\mathcal{M}}(\phi(I_{\mathcal{S}^-}))$ to produce corresponding attention maps. Finally, we use k Fusion Networks to fuse the k pairs of painter vectors and attention maps. Each pair outputs one component $\mathcal{V}_{\mathcal{S}_i}$. These k components are then summed to $\mathcal{V}_{\mathcal{S}}$. The pipeline is visualized in Fig 1.

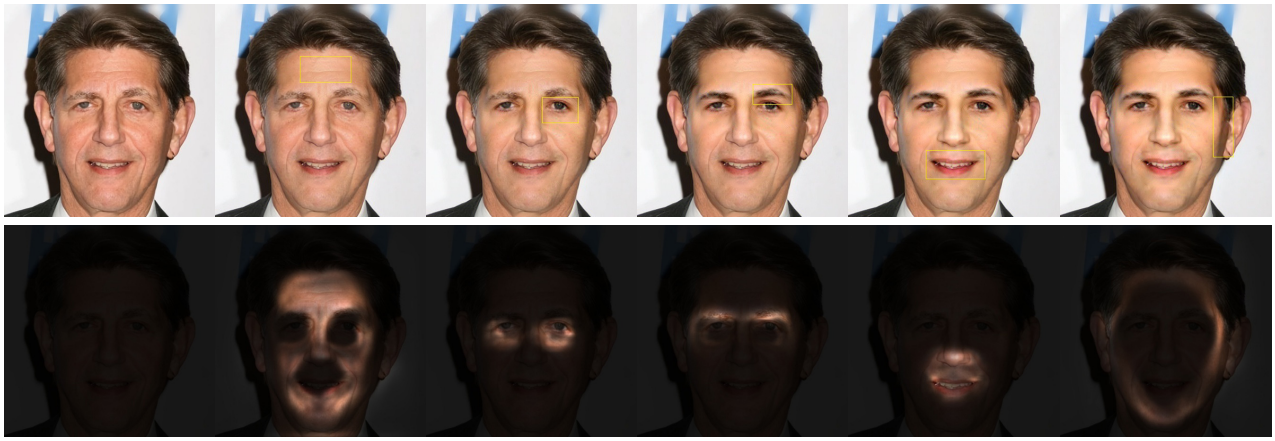
3.4. Interactive Editing

The proposed architecture is highly flexible and thus allows users to adjust results in different levels. The coarsest level is to change the global effect like [23, 5], which is to update λ in Eq. (1). In addition, our network linearly decomposes $\mathcal{V}_{\mathcal{S}}$ to different semantic components $\mathcal{V}_{\mathcal{S}_i}$ as indicated in Eq. (5). This allows us to control the edit strength of different components separately. Unwanted edit can be totally removed by setting the edit strength to zero.

Further, because each component $\mathcal{V}_{\mathcal{S}_i}$ is determined by $\mathcal{P}_i$ and $\mathcal{M}_i$, they can also be edited by changing $\mathcal{P}_i$ and $\mathcal{M}_i$. Note that $\mathcal{M}_i$ is a one-channel attention map, which contains large values on updated regions and small ones otherwise. It is thus intuitive for users to directly "draw" on $\mathcal{M}_i$. $\mathcal{P}_i$, contrarily, is a spatial-invariant vector related to the type of effect. With it, we can easily alter styles of $\mathcal{V}_{\mathcal{S}_i}$ by replacing $\mathcal{P}_i$.

4. Experiments

We use the face attribute dataset CelebA [15] to train and test our model. The large and diverse face dataset contains 202,599 images of 10,177 identities. In addition, there are 40 types of attribute annotation. They can be used to divide the dataset into two domains to estimate $\mathcal{V}_{\mathcal{S}}$ in Eq. (2). Three attribute-altering tasks are tested in this paper, includ-



(a) (b) (c) (d) (e) (f)

Figure 2. Illustration of visual effect of different components of attribute "younger". (a) is the input. (b-f) shows gradual visual change by incorporating different semantic components. The upper line shows the visual changes, and the lower line shows the edited region $\mathcal{M}_i$ (**Best view in original resolutions**).

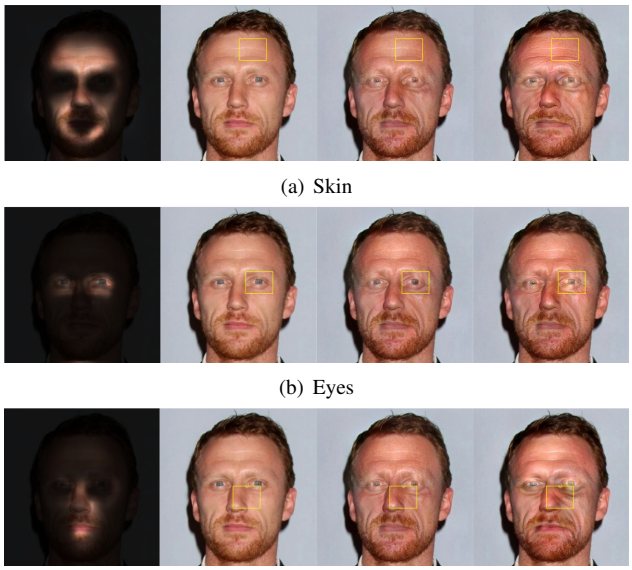
ing adding facial hair, turning older, and getting younger. These attributes include various semantic components and thus are suitable to evaluate the effectiveness of our network. We unify the image resolution to 448×448, and train our model with its training set. Since the resolution of original CelebA is limited, we use its high-quality version [9] during testing.

4.1. Evaluating Our Model

One key benefit of our approach stems from the ability to decompose an integrated attribute into several components so that users can manipulate each of them separately. In this section, we provide an extensive analysis of the component decomposition property. We set k in Eq. (5) to 9, which is large enough for most attributes. Although this may cause some network branches to learn insignificant components, our experiments show that it does not affect results because insignificant components can be disabled during testing.

Visual Effect of Different Components Intuitively, a high-level facial attribute boils down to appearance change of different face regions. For example, a young man/woman usually has brighter eyes, fewer wrinkles, smoother skin than a more senior person. So to make a face look younger, these properties should be changed accordingly.

Our model captures these properties by components. Fig. 2 shows different semantic components of attributes. It reveals that our model decomposes attribute "younger" into five components. The first two components are in accordance with our intuition that a younger person has brighter eyes and smoother skin. Interestingly, the 3rd-5th components indicate that "younger" means thicker eyebrow, brighter lip color, and tighter face contour. These components work subtly and jointly to make the result look more realistic.



(a) Skin

(b) Eyes

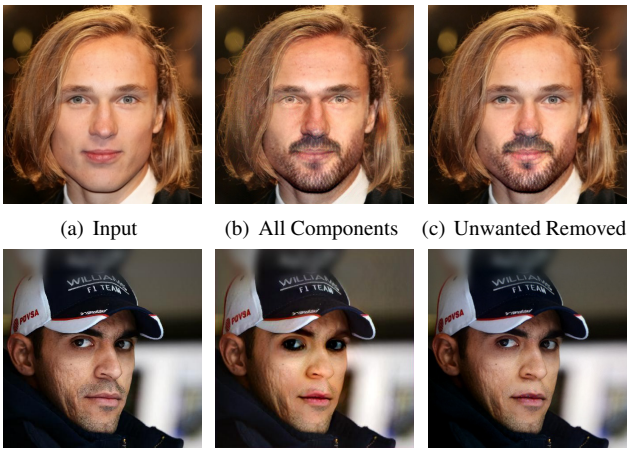
(c) Nose

Figure 3. Illustration of changing edit strength of the "older" attribute. For each row, the first image indicates the edit region $\mathcal{M}_i$ of the corresponding component. The second image is the input image. The 3rd and 4th images are results of setting different edit strength of the corresponding components respectively. When changing the strength of one component, we fix weights of others (**Best view in original resolutions**).

Changing Component Strength Note that Eq. (5) can be extended to

$$\mathcal{V}_{\mathcal{S}} = \sum_{i=1}^{k} \lambda_i \mathcal{V}_{\mathcal{S}_i}, \tag{9}$$

where $\lambda_i \geq 0$ is the weight of the i-th component. This



(a) Input (b) All Components (c) Unwanted Removed

(d) Input (e) All Components (f) Unwanted Removed

Figure 4. Examples of component-level manipulation. (a) Original image. (b) Result of the "facial hair" attribute using all components. (c) improves (b) using only the significant components. (d) Original image. (e) Result of attribute "femininity". (f) Result of "remove facial hair". Note that (f) is obtained by only keeping the component with edit on mouth and removing all others (**Best view in original resolutions**).

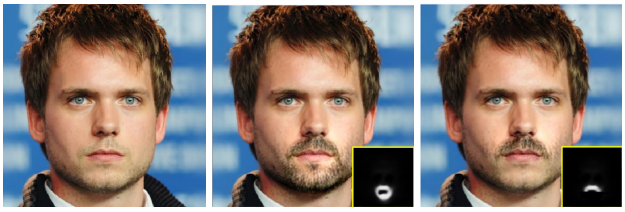
allows us to control the weight of each component during testing. Fig. 3 shows the result of changing the strength of semantic components of the "older" attribute. This is more advantageous compared with existing models [34, 19] that do not support changing edit strength or only do that globally [23, 5].

Removing Unwanted Components One special case of Eq. (9) is to set certain λ_i to 0, i.e., totally removing these components. This is important to improve output quality. Recall that $\mathcal{V}_S$ is computed with Eq. (2), which is only an approximation to the genuine attribute tensor.

In practice, there inevitably exist unwanted edit in $\mathcal{V}_S$, which may harm the final result. Our model decomposes $\mathcal{V}_S$ into multiple components, each controls one kind of effect. Thus it is likely that noise is fitted by certain components of our model. Interestingly, discarding them leads to even better results.

Fig. 4(a)-(c) shows an example of "facial hair" attribute. It is a simple property that should cause change only on the mouth region. However, as shown in Fig. 4(b), noise in $\mathcal{V}_S$ contaminates other unrelated regions (such as eyes and nose). By removing these noisy components, the unwanted edits are suppressed, as shown in Fig. 4(c).

Learn One and Get More Our model also allows us to obtain different edits from only one attribute. Note that a complex attribute is usually composed of several sub-attributes. Fig. 2(b)-(f) shows a few sub-attributes for the "younger" change. In many cases, these sub-attributes are very useful for new effect generation. Taking the "femininity" attribute as an example, as shown in Fig. 4(e), we train this attribute with the gender label and converts a male-look towards a female-look. Interestingly, during this training process, another semantic edit – "remove facial hair" – is also learned as one component, as shown in Fig. 4(f).



(a) Input (b) Initial Result (c) Edited Result

Figure 5. Illustration of edit on the attention map $\mathcal{M}_i$. (a) is the input image. (b) shows the initial output automatically produced by our model. (c) is the editing result. The bottom right images of (b) and (c) are their corresponding $\mathcal{M}_i$. (**Best view in original resolutions**).

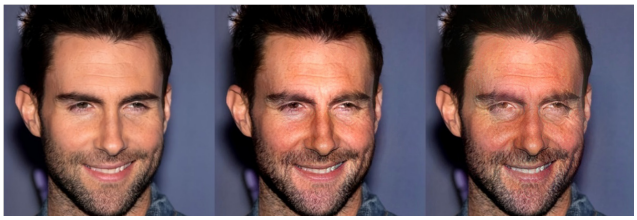
Fine-grained Adjustment In addition to changing the edit strength of different components, our model also allows fine-grained control, which is to directly manipulate each component. Note that each component $\mathcal{V}_{S_i}$ is decomposed into two factors: the painter $\mathcal{P}_i$ and the edited region $\mathcal{M}_i$. By manipulating $\mathcal{M}_i$, the edited region can be controlled. Contrarily, with modified $\mathcal{P}_i$, the edit style is updated. We showcase the effects in the following.

- Effect 1: Editing $\mathcal{M}_i$ $\mathcal{M}_i$ provides spatial information of component $\mathcal{V}_{S_i}$ according to Eqs. (5) and (6). As $\mathcal{M}_i$ is 2-D attention map, manipulating $\mathcal{M}_i$ is simple and straightforward. Fig. 5 shows an example of manipulating the shape of facial hair by only changing the shape of $\mathcal{M}_i$.

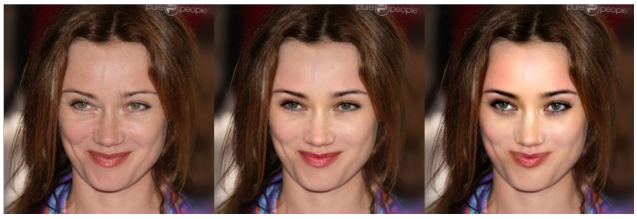
- Effect 2: Changing $\mathcal{P}_i$ $\mathcal{P}_i$ serves as the complementary part of $\mathcal{M}_i$ to instantiate $\mathcal{V}_{S_i}$. Thus it is deemed as controlling the type of effect on the corresponding region $\mathcal{M}_i$. Note that most high-level attributes have more than one instantiations, which means $\mathcal{V}_S$ in the training set contains different $\mathcal{P}_i$. So during testing, if the user is not satisfied with the initial instantiation of the attribute change, s/he can replace $\mathcal{P}_i$ with others to update results, as illustrated in Fig. (6).

Running Time We report the running time with image size 448×448, and the system runs on a Titan X graphics card. Our framework can be divided into 3 parts, i.e., the VGG encoder, the decoder, and our semantic component model. The VGG encoder takes 0.008 second, and the decoder takes 0.014 second. The time cost of our model varies from 0.026 (with 1 component) to 0.048 (with 5 components) second. The time complexity variation is due to various types of attributes that require different numbers of components to compute. For simple attributes, such as fa-

(a) Facial hair



(b) Older



(c) Younger

Figure 6. Illustration of changing instantiation by modifying $\mathcal{P}_i$. We pre-compute different $\mathcal{P}_i$ in the training set, and randomly pick one to replace the predicted $\mathcal{P}_i$ of the query image. (a) illustrates facial hair attribute. The color and texture of the facial hair are different. (b) illustrates the "older" attribute. The cheek, eyebrows and eyes are different. (c) illustrates the "younger" attribute. The skin color, eyebrows and mouth are different. (**Best view in original resolutions**).

cial hair, it requires only 1 component and thus can run really fast. For more global and complicated effect like getting younger or older, it needs more components. Nevertheless, even for the very complicated edit, the running time is still acceptable for interactive editing with above statistics.

4.2. Comparison with State-of-the-Arts

In addition to high flexibility, a fundamental advantage of our method is that in most cases, it produces high-quality results even on its initial proposals. To validate this, we compare our approach with two categories of approaches, i.e., DFI (Deep Feature Interpolation) based approaches [23, 5] and GAN based approaches [27, 19].

For fair comparison, the weights of all semantic components are set equal. After training, we discard branches that produce noise, and only use branches that produce significant semantic components.

Qualitive Evaluation Fig. 7(a) shows the comparison with DFI [23] and Facelet Bank [5]. DFI manipulates an image by directly using Eq. (2), which takes about 1 minute to process an image. Facelet Bank accelerates it by using a 3-layer CNN to predict Eq. (2). Since the methods are related, the quality regarding the target attribute is comparable. Nevertheless, benefiting from component decomposition, our work yields the advantage of removing unwanted edit. For example, for the "older" attribute in the 1st and 3rd rows, DFI and Facelet Bank adversely modify the background and hair respectively, while our approach correctly updates the face regions.

	Ours>DFI	Ours>Facelet
Facial hair	52.8%	52.9%
Older	50.2%	54.0%
Younger	65.3%	87.6%
	Ours>CycleGAN	Ours>ResGAN
Facial hair	61.7%	85.7%
Older	58.2%	87.0%
Younger	71.3%	91.5%

Table 1. A/B test of image quality. Each entry reports the percentage among the 1,000 comparisons that images generated by our approach are better than those of other solutions.

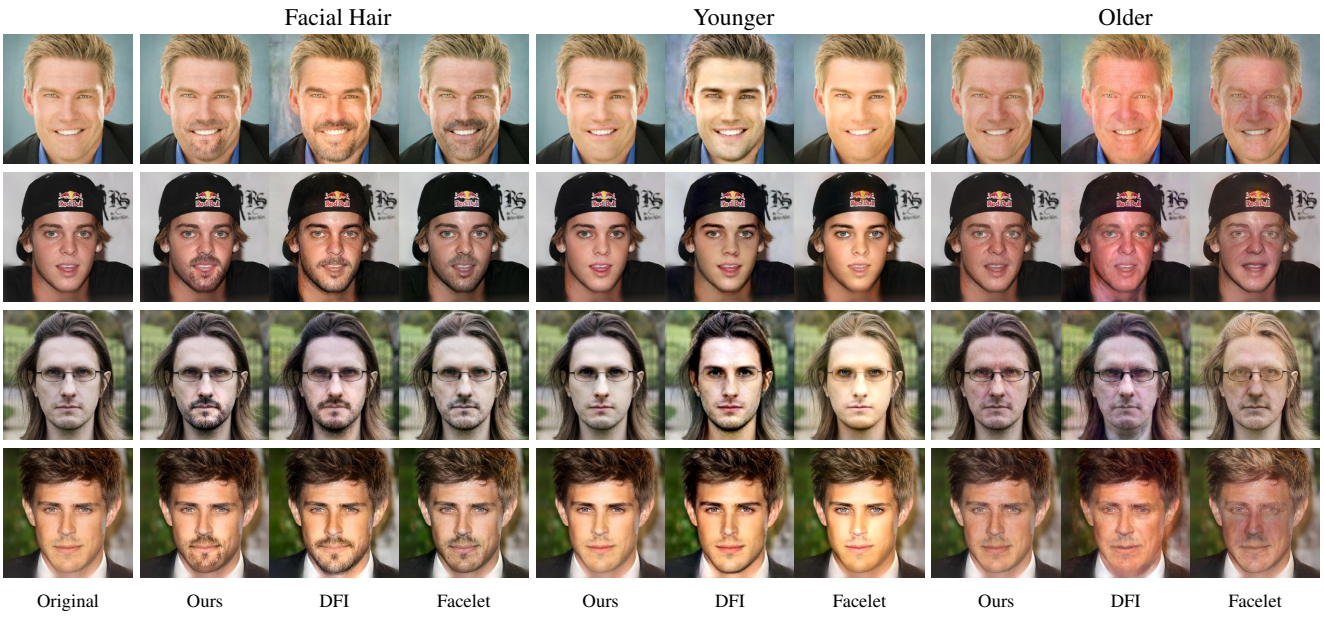
Fig. 7(b) shows the comparison among our work, CycleGAN [34] and ResGAN [19]. CycleGAN is a general framework for set-to-set image translation. ResGAN is specifically designed for face edit that learns the residual part of attribute change. Compared with these two methods, our work consistently generates more reliable and higher quality effect. For example, ResGAN fails to generate facial hair for the 4th person, and CycleGAN does not produce significant "younger" attribute for the 1st person.

Further, our method can handle images in higher resolutions compared with CycleGAN (256×256) and ResGAN (128×128). It is known that high-res images make it easier to tell the generated images apart from the training ones, making the discriminators hard to provide significant gradient information to update the generators. Our approach does not have such problems since it is trained with a simple L_2-loss. Higher resolutions do not increase the difficulty of producing similar-quality effect during face edit.

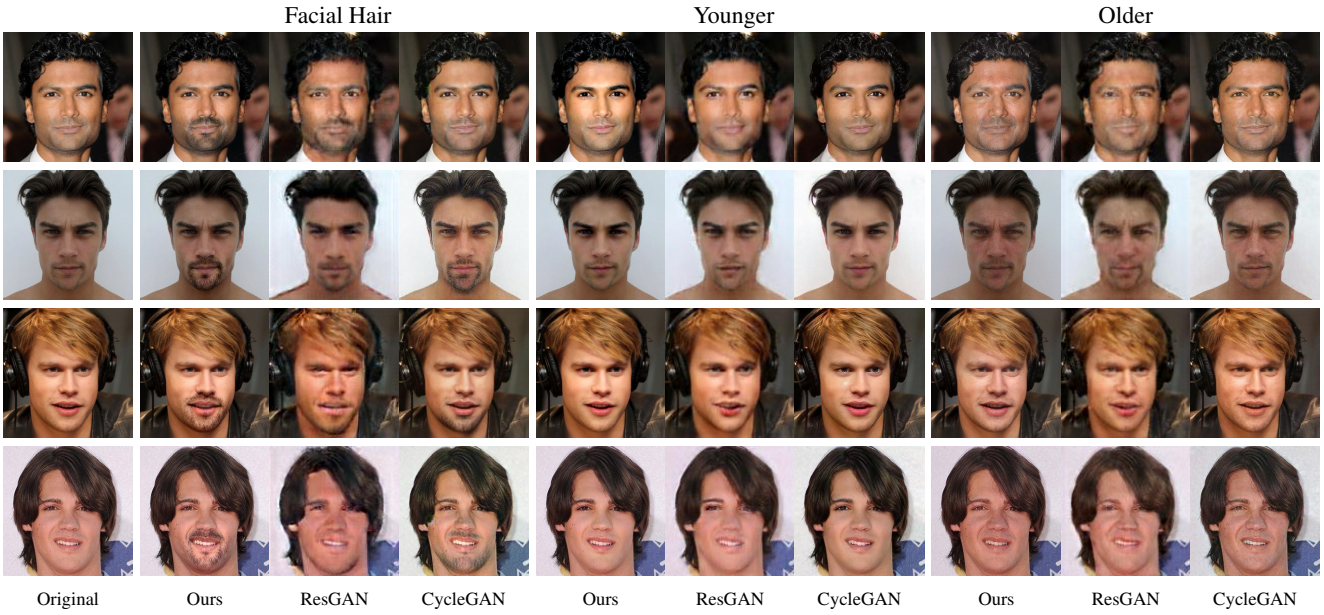
Quantitative Evaluation We also conduct a randomized pair-wise comparison of images edited by our approach and the baseline methods on the Amazon Mechanical Turk platform. Each time users are given an original image, two edited ones (ours vs. another method) and the edit target (facial hair, getting older or younger), and are asked to pick one with the higher quality and less incorrect edit. Statistics in Table 1 show that our approach outperforms all other alternatives.

4.3. Limitations

Similar to DFI [23] and Facelet [5], our model leverages a pretrained VGG network as the encoder. Note that this encoder is not updated during training. This may restrict



(a)



(b)

Figure 7. (a) Comparison with (a) DFI [23] and Facelet-Bank [5] and with (b) CycleGAN [34] and ResGAN [19]. **Please zoom in to see more details.**

the scope of application. If the target attribute is not well represented in the latent space, our model may not learn any significant component. In our experiments, we found that our model is not suitable for tasks that require large geometry change, such as face frontalization.

5. Concluding Remarks

The presented method is the first of its kind to decompose semantic components from high-level attributes. This on the one hand allows more user control of the face attribute manipulation, avoids false edit, and improves results. On the other hand, it improves our fundamental understanding of what kind of changes lead to certain attributes and makes the edit more interpretable. Other benefits of our solution include the fast computation speed that even allows interactive editing. Extensive experiments manifest that our approach achieves high-quality results and show the new way to facilitate user-friendly realistic photo editing.

References

[1] T. Beeler, B. Bickel, G. Noris, P. Beardsley, S. Marschner, R. W. Sumner, and M. Gross. Coupled 3d reconstruction of sparse facial hair and skin. *ToG*, 2012. 1

[2] V. Blanz, C. Basso, T. Poggio, and T. Vetter. Reanimating faces in images and video. In *Comput. Graph. Forum*, 2003. 1, 2

[3] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Ann. Conf. Comput. Graph. Interactive Techn.*, 1999. 1, 2

[4] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *TVCG*, 2014. 1, 2

[5] Y.-C. Chen, H. Lin, M. Shu, R. Li, X. Tao, Y. Ye, X. Shen, and J. Jia. Facelet-bank for fast portrait manipulation. In *CVPR*, 2018. 2, 3, 4, 6, 7, 8

[6] Y.-C. Chen, X. Shen, and J. Jia. Makeup-go: Blind reversion of portrait edit. In *ICCV*, 2017. 1

[7] 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 *CVPR*, 2018. 1, 2

[8] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen. Arbitrary facial attribute editing: Only change what you want. *arXiv*, 2017. 1, 2

[9] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018. 1, 2, 5

[10] I. Kemelmacher-Shlizerman, S. Suwajanakorn, and S. M. Seitz. Illumination-aware age progression. In *CVPR*, 2014. 1, 2

[11] T. Kim, B. Kim, M. Cha, and J. Kim. Unsupervised visual attribute transfer with reconfigurable generative adversarial networks. *arXiv*, 2017. 1, 2

[12] J. Kossaifi, L. Tran, Y. Panagakis, and M. Pantic. Gagan: Geometry-aware generative adverserial networks. In *CVPR*, 2018. 1

[13] G. Lample, N. Zeghidour, N. Usunier, A. Bordes, L. Denoyer, et al. Fader networks: Manipulating images by sliding attributes. In *NIPS*, 2017. 1, 2

[14] T. Leyvand, D. Cohen-Or, G. Dror, and D. Lischinski. Digital face beautification. In *Siggraph*, 2006. 1

[15] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 4

[16] D. Ma, B. Liu, Z. Kang, J. Zhu, and Z. Xu. Two birds with one stone: Iteratively learn facial attributes with gans. *arXiv*, 2017. 1, 2

[17] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez. Invertible conditional gans for image editing. *arXiv*, 2016. 1, 2

[18] S. Sabour, N. Frosst, and G. E. Hinton. Dynamic routing between capsules. In *NIPS*, 2017. 1

[19] W. Shen and R. Liu. Learning residual images for face attribute manipulation. In *CVPR*, 2017. 1, 2, 6, 7, 8

[20] Z. Shu, E. Yumer, S. Hadap, K. Sunkavalli, E. Shechtman, and D. Samaras. Neural face editing with intrinsic image disentangling. In *CVPR*, 2017. 1, 2

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2014. 3

[22] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt. Real-time expression transfer for facial reenactment. *TOG*, 2015. 1, 2

[23] P. Upchurch, J. Gardner, K. Bala, R. Pless, N. Snavely, and K. Weinberger. Deep feature interpolation for image content changes. In *CVPR*, 2017. 2, 3, 4, 6, 7, 8

[24] C. Wang, C. Wang, C. Xu, and D. Tao. Tag disentangled generative adversarial network for object image re-rendering. In *IJCAI*, 2017. 1, 2

[25] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 1, 2

[26] Y. Wang, L. Zhang, Z. Liu, G. Hua, Z. Wen, Z. Zhang, and D. Samaras. Face relighting from a single image under arbitrary unknown lighting conditions. *TPAMI*, 2009. 1, 2

[27] T. Xiao, J. Hong, and J. Ma. Dna-gan: Learning disentangled representations from multi-attribute images. *arXiv*, 2017. 1, 2, 7

[28] T. Xiao, J. Hong, and J. Ma. Elegant: Exchanging latent encodings with gan for transferring multiple face attributes. In *ECCV*, 2018. 2

[29] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2image: Conditional image generation from visual attributes. In *ECCV*, 2016. 1, 2

[30] F. Yang, J. Wang, E. Shechtman, L. Bourdev, and D. Metaxas. Expression flow for 3d-aware face component transfer. *TOG*, 2011. 1, 2

[31] R. Yeh, Z. Liu, D. B. Goldman, and A. Agarwala. Semantic facial expression editing using autoencoded flow. *arXiv*, 2016. 1

[32] W. Yin, Y. Fu, L. Sigal, and X. Xue. Semi-latent gan: Learning to generate and modify facial images from attributes. *arXiv*, 2017. 1, 2

[33] G. Zhang, M. Kan, S. Shan, and X. Chen. Generative adversarial network with spatial attention for face attribute editing. In *ECCV*, 2018. 2

[34] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 1, 2, 6, 7, 8

[35] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. In *NIPS*, 2017. 1, 2