MaskFaceGAN: High Resolution Face Editing with Masked GAN Latent Code Optimization

Martin Pernuš, *Student Member, IEEE,* Vitomir Štruc, *Senior Member, IEEE,* and Simon Dobrišek, *Member, IEEE*

Original Arched eyebrows Blond hair Big nose Wearing lipstick

Fig. 1: This paper introduces MaskFaceGAN, a novel approach to face attribute editing capable of generating high-resolution, artefact-free and photo-realistic editing results through a carefully designed optimization procedure applied over the StyleGAN2 latent space. The presented (1024×1024) examples show editing results for four target attributes. Best viewed zoomed-in.

***Abstract*—Face editing represents a popular research topic within the computer vision and image processing communities. While significant progress has been made recently in this area, existing solutions: (i) are still largely focused on low-resolution images, (ii) often generate editing results with visual artefacts, or (iii) lack fine-grained control over the editing procedure and alter multiple (entangled) attributes simultaneously, when trying to generate the desired facial semantics. In this paper, we aim to address these issues through a novel editing approach, called MaskFaceGAN that focuses on local attribute editing. The proposed approach is based on an optimization procedure that directly optimizes the latent code of a pre-trained (state-of-the-art) Generative Adversarial Network (i.e., StyleGAN2) with respect to several constraints that ensure: (i) preservation of relevant image content, (ii) generation of the targeted facial attributes, and (iii) spatially–selective treatment of local image regions. The constraints are enforced with the help of an (differentiable) attribute classifier and face parser that provide the necessary reference information for the optimization procedure. MaskFaceGAN is evaluated in extensive experiments on the CelebA-HQ, Helen and SiblingsDB-HQf datasets and in comparison with several state-of-the-art techniques from the literature. Our experimental results show that the proposed approach is able to edit face images with respect to several local facial attributes with unprecedented image quality and at high-resolutions (1024×1024), while exhibiting considerably less problems with attribute entanglement than competing solutions. The source code is publicly available from: https://github.com/MartinPernus/MaskFaceGAN.**

***Index Terms*—facial attribute editing, generative adversarial network, GAN inversion, latent code optimization.**

I. Introduction

FACE ATTRIBUTE EDITING refers to the task of manipulating facial images towards some predefined appearance. Techniques capable of automatic facial attributes editing (e.g., hair color, makeup, shape of facial components, age, identity) have important real-world applications not only in entertainment, graphics, arts, or the beauty industry, but also in problem domains related to visual privacy or security [1], [2], [3], [4]. As a result, considerable research effort has been directed towards face editing techniques over the years and resulted in powerful solutions capable of generating convincing photo-realistic editing results [5], [6], [7], [8], [9].

Recent progress in face attribute editing has largely been driven by the advances in convolutional neural networks (CNNs) and adversarial training objectives utilized in Generative Adversarial Networks (GAN) [10]. Existing solutions can broadly be categorized into two main groups. The *first* includes techniques that pose attribute editing as an *image-to-image* translation task and utilize dedicated learning objective to generate the desired target semantics [11], [12], [5], [7], [8]. Such techniques typically rely on some sort of encoder-decoder architecture and are, therefore, computationally efficient, but primarily designed for low-resolution editing (e.g., 128×128 or 256×256). Moreover, due to the nature of the learning objective used, they often induce visual artefacts in the edited images. The *second* (more recent) group of techniques is based the concept of *GAN inversion* [13] and exploits the generative capabilities of pre-trained GAN models for editing [14], [15], [16]. Here, a target image is first converted (embedded) into a latent code and then edited through manipulations (optimization) in the latent space. The main advantage of this group of techniques is the high-resolution and impressive image quality of the editing results. However, because the latent code commonly represents a *global image representation* with entangled attribute information, it is challenging to manipulate individual facial attributes without affecting others. Generating convincing local edits, therefore, represents a major challenge for this group of techniques.

In this paper, we propose a novel GAN-inversion based approach to (local) facial attribute editing, called MaskFaceGAN, that is capable of generating high-resolution visually convincing editing results (illustrated in Fig. 1) but does not suffer from the entanglement problems discussed above.

Supported by the Slovenian Research Agency ARRS through P2-0250(B), J2-2501(A) and the junior researcher program.

The approach is primarily focused on editing facial attributes which are well–defined by specific image regions. At the core of MaskFaceGAN is a carefully designed optimization procedure operating directly over the latent space of the recent StyleGAN2 model [17]. The procedure aims to determine a latent code that encodes the desired target semantics (i.e., presence/absence of a target attribute and original facial appearance) by considering multiple groups of optimization constraints during the process of GAN inversion. The *first* group is enforced through a facial attribute classifier and ensures that the edited image contains the correct attribute information. The *second* group of constraints is imposed through a face parser that defines image regions that belong to different facial components. Information on these components is then used as the basis for spatial constraints that encourage the optimization procedure to either preserve or alter image regions corresponding to specific facial regions. A blending procedure is also incorporated into MaskFaceGAN to help preserve important aspects of the original input image. MaskFaceGAN is evaluated on three high-resolution face datasets and in comparison to several state-of-the-art editing techniques from the literature. The results of rigorous (qualitative and quantitative) experiments show that the proposed approach generates highly competitive editing results, while exhibiting some unique characteristics not available with prior editing solutions. Overall, we make the following contributions in this paper:

- We present MaskFaceGAN, a novel approach to face image editing, capable of generating state-of-the-art, visually convincing, artefact-free, photo-realistic editing results at high resolutions, i.e., 1024×1024 pixels.
- We propose an efficient optimization procedure for estimating GAN latent codes of facial images that encode the selected target semantics. The procedure enforces various optimization constraints through the use of differentiable models applied over the edited images.
- We show how MaskFaceGAN can be used for attribute-intensity control, multi-attribute editing and component size manipulation while requiring only binary attribute labels for optimization.
- We demonstrate through comprehensive experiments that spatially constrained image editing significantly reduces entanglement problems compared to competing models.

II. Related work

In this section, we present prior work closely related to our paper. We discuss Generative Adversarial Networks (GANs), research on pre–trained GANs and face editing techniques.

A. Generative Adversarial Networks

Generative Adversarial Networks (GANs) are among the most popular generative models in the field of image processing and computer vision [10]. Existing GANs can be broadly split into two categories: (i) unconditional and (ii) conditional models. Unconditional GANs refer to models that rely only on random noise to generate image data. No additional signal is used to steer the generation process. Conditional GANs, on the other hand, exploit additional inputs to control the semantic content of the generated images and typically utilize random noise to ensure diversity. Different forms of the conditional signals have been used in the literature, including class labels [18], [19], graph representations, [20], [21], layouts of image objects [22] or text descriptions [23], [24], [25] among others.

The progress in GAN-based image generation can for the most part be attributed to advances in model design and training. DCGAN [26], for example, proposed a convolutional GAN architecture and defined several useful design principles, such as the use of batch normalization in all model layers and specific activation functions for the generator and the discriminator. In [27], Karras *et al.* introduced a progressive learning strategy for GANs that adds higher-resolution layers to the model once the lower-resolution layers have converged. The authors showed that using such a strategy results in GANs capable of generating convincing megapixel-sized images. The progressively learned model was further improved with the introduction of StyleGAN [28], a GAN model inspired by the style–transfer literature. Different from traditional Gaussian-shaped latent spaces, StyleGAN proposed the use of a non-linear mapping from the Gaussian latent space to an intermediate latent space (with better interpolation and disentanglement properties) that was then fed to convolutional layers via an adaptive instance normalization operation. Additionally, the model also introduced noise inputs for generating stochastic image details. The next iteration of the model, StyleGAN2 [17], modified the adaptive instance normalization operation to remove circular artefacts in the generated images, achieving state-of-the-art results on unconditional image generation. Considerable progress has also been made with GAN learning strategies, where different losses and regularizations were proposed to improve the generation quality [29], [30], [31], [32], [33]. Additional, up-to-date information on GANs can be found in one of the recent surveys on this topic [34], [35].

B. Studies on Pre-trained GANs

Training GAN models can be highly resource intensive. For example, the computational effort required for developing and training the recent StyleGAN2 model was estimated to be around 51 Volta GPU years [17]. This immense effort has motivated research into the capabilities of pre–trained GANs and resulted in powerful techniques that exploit existing models for various generative tasks. Bau *et al.* [36], for example, analyzed pre–trained GANs to achieve localized deletion and insertion of objects. Jahanian *et al.* [37] investigated linear and non-linear walks in the GAN latent space that resulted in basic image manipulations, such as changes in brightness or the zoom factor. Goetschalckx *et al.* [38] navigated the latent code manifold to improve the image's memorability. Yang *et al.* [39] analyzed the relationship between image semantics and layer activations and manipulated image characteristics, such as layout, scene attributes or the employed color scheme. PULSE [40] utilized StyleGAN to perform face superresolution. InsetGAN [41] utilized two separate pre–trained GANs to perform face–conditioned body synthesis and body–face image stitching, where the first GAN models the face region, whereas the other GAN models the entire body.

Similarly to the research discussed above, MaskFaceGAN also exploits a pre-trained GAN model to edit facial images.

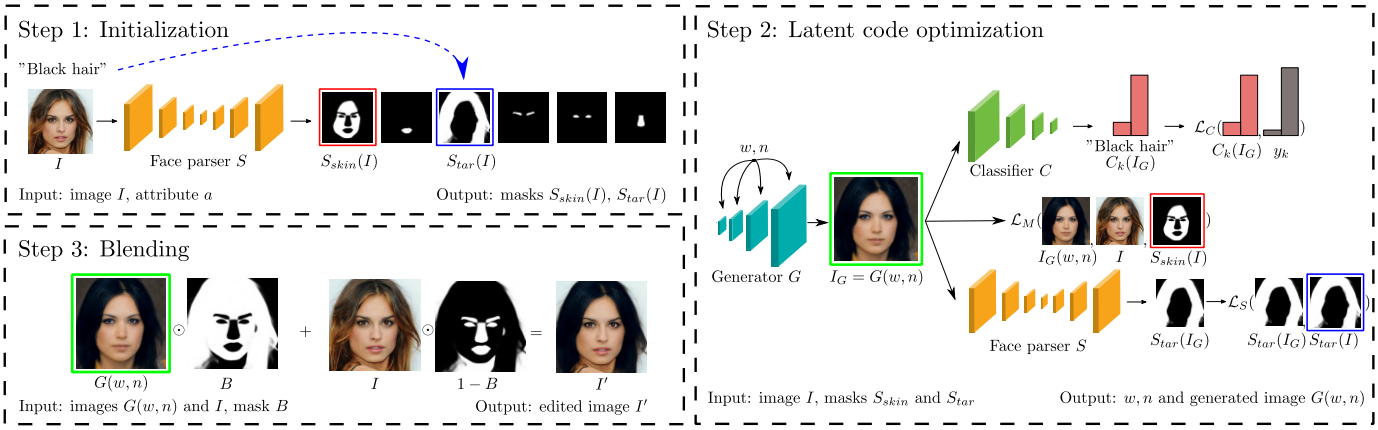


Fig. 2: Overview of MaskFaceGAN, illustrated with the "Black hair" target attribute. To initialize the editing procedure, MaskFaceGAN uses a face parser to define masks that correspond to image regions that should be preserved (S_{skin}) and regions that should be altered (S_{tar}). Next, latent code optimization is performed in accordance with semantic and spatial constraints to generate an intermediate image I_G with the targeted characteristics. Finally, blending is used to combine the generated image I_G with the original I and to produce the final editing result I'. The image is best viewed electronically.

However, different from existing work, the GAN model used in our approach serves only as a proxy for the editing procedure that synthesizes semantically meaningful content in spatially constrained face regions. The synthesized content is combined later with the original image for the final result.

C. Face Editing

Numerous approaches for face editing and manipulation have been presented in the literature, e.g., [1], [2], [42], [43]. The work in [44], [6], for example, explored the use of user-supplied sketches to drive the editing procedure. The authors of [11], [12] learned disentangled latent representations w.r.t. image formation to enable control of the image generation process. Lee *et al.* [45] proposed MaskGAN model, a conditional GAN, capable of modifying specific facial components and demonstrated the benefit of using spatially local face editing.

Particularly convincing editing results have been reported with encoder-decoder models. Choi *et al.* [5], for instance, introduced StarGAN, an image-to-image encoder-decoder network with cycle consistency [46], capable of manipulating the appearance of several face attributes. He *et al.* [7] proposed AttGAN, a notable encoder-decoder model that utilizes reconstruction constraints instead of cycle consistency. Liu *et al.* [8] improved on AttGAN with their STGAN design by modifying the input signal and improving the encoder-decoder architecture with selective transfer units. Encoder-decoder models were shown to generate visually convincing editing results, but are less suitable for editing high-resolution images, where visual artefacts are often observed.

More recent solutions approach the problem of face editing with the use of pre-trained GANs. Abdal *et al.* [15] showed that it is possible to embed a large variety of images in the extended latent space of StyleGAN and to perform various forms of image manipulation in the latent space, including face morphing, style transfer and expression transfer. In their follow up work [16], the authors improved the embedding algorithm by optimizing the StyleGAN noise component, and demonstrated additional capabilities, such as local editing or face inpainting. A convincing editing approach, called InterFaceGAN, was described by Shen *et al.* in [9], [14]. To edit an image, InterFaceGAN moves the latent code corresponding to the input image along a linear subspace. The direction of the displacement is determined through a support vector machine (SVM), trained on the StyleGAN latent space given labels of predefined facial attributes. While such approaches led to state-of-the-art editing results, they are based on linear operations applied over latent space representations and, therefore, often suffer from entanglement issues where changing one attribute also results in changes in other (entangled) image attributes.

With MaskFaceGAN we improve on previous methods by directly optimizing the latent code associated with an input image through multiple optimization constraints that not only control the manipulated semantic content but also the spatial area in which the editing occurs. This optimization procedure results in complex (non-linear) changes in the latent code of the input image and leads to editing results with significantly less entanglement problems than competing solutions, as we demonstrate in the experimental section.

III. Methodology

A. Background and Problem Formulation

The goal of face attribute editing is to manipulate (in a photo–realistic manner) a given input image $I \in \mathbb{R}^{3\times m\times n}$ in accordance with some specified target semantics a, i.e.,

$$\psi_a : I \mapsto I' \in \mathbb{R}^{3\times m\times n}, \tag{1}$$

where I' represents the edited image and the semantics are usually defined by predefined facial attributes (e.g., "Blond hair", "Big nose", etc.). As discussed in the previous section, the most recent methods implement the mapping ψ_a through the process of GAN inversion [14], [13]. With these techniques the input image I is first embedded in the latent space of a pre-trained GAN model G, resulting in a latent representation (or latent code) w. Next, the latent code is modified in accordance with a target objective determined by a, i.e., $\psi_a : w \mapsto w^*$,

and finally, the edited image I' is generated by evaluating w^* through G, that is, $I' = G(w^*)$.

MaskFaceGAN, presented in the following sections, follows this general GAN inversion framework, but different from competing solutions exhibits several unique characteristics, as illustrated in the experimental part of the paper.

B. Overview of MaskFaceGAN

A high–level overview of MaskFaceGAN is presented in Fig. 2. The key component of the proposed approach is a latent code optimization procedure that considers multiple constraints during optimization, including: (i) an *appearance–preservation constraint* that ensures that the edited image I' is as close to the original I in image areas that should not be altered by MaskFaceGAN, (ii) a *semantic constraint* that ensures meaningful target semantics within local image areas, and (iii) a *shape constraint* that determines the location and shape of the image regions to be manipulated during editing. The optimization procedure is used as part of a three–step editing operation in MaskFaceGAN, i.e.:

- **Step 1: Initialization.** Given an input image I and a binary (target) attribute label a, MaskFaceGAN uses a face parser S in the first step to identify facial regions S_{skin} that should be preserved during editing and predict image areas S_{tar} that need to be edited given a. See Fig. 2 for an example using the target attribute "Black hair".
- **Step 2: Latent–code optimization.** The second step of MaskFaceGAN involves the optimization of the GAN latent code. This step aims at estimating a latent code of the (intermediate) image I_G that exhibits the targeted semantics in image regions defined by S_{tar} and preserved appearance in S_{skin}. The semantics and spatial constraints are enforced through an attribute classifier C and a face parser S, respectively. A loss, defined over these models, is backpropagated to the latent space to facilitate the optimization.
- **Step 3: Blending.** Finally, a blending step is utilized to combine the (intermediate) image I_G, generated from the optimized latent code (with locally manipulated target semantics), with the original image I. This step adds the background and other facial components not considered during optimization to the final editing result I'.

C. Models

MaskFaceGAN relies on three distinct components to implement the optimization procedure, i.e., (i) a GAN–based generator (G) capable of producing high–resolution facial images, (ii) an attribute classifier (C) utilized for enforcing the targeted semantics, and (iii) a face parser (S) used for imposing spatial constraints.

- **The Generator** (G) is based on StyleGAN2 [17], a state-of-the-art GAN model specialized for generating photo–realistic facial images. Following established editing methodology [15], [16] the extended latent space[1] $\mathcal{W}^+$ of StyleGAN2 is used for encoding image semantics. As a result, an image is represented through a concatenation of $n_c = 18$ different 512-dimensional latent vectors w_i, one for each layer of the model, i.e., $w = \{w_i\}_{i=1}^{n_c}$. To ensure photo-realism StyleGAN2 additionally uses $N = 17$ stochastic (i.e., Gaussian noise) channels $n = \{n_i\}_{i=1}^{N}$ of different spatial resolutions (ranging from 4×4 to 1024×1024) that encode high-frequency image details. The model, hence, generates output images I_G based on the following mapping: $I_G = G(w, n)$.
- **The Attribute Classifier** (C) is designed around the multi–task tree neural network from [47] and consists of several shared layers and K classification heads (leaf branches), one for each of the K attributes supported (for editing) by MaskFaceGAN. Given an image $I_G = G(w, n)$, each classification head C_a predicts the probability of a facial attribute being present in I_G, i.e., $c_k = C_k(I')$ for the k-th attribute.
- **The Face Parser** (S) is built around DeepLabV3 [48] and provides pixel-level probability predictions for various face components/regions, as illustrated in Fig. 3. These facial regions are then associated with specific attributes that can be edited within the given region in accordance with Table I. Formally, the model implements a mapping from an image I to a tensor of probabilities along the channel dimension, i.e.: $S : \mathbb{R}^{3\times n\times m} \rightarrow [0,1]^{L\times n\times m}$, where L is the number of parsed categories (face components). For MaskFaceGAN, two principal channels are used. The first one is the skin region, $S_{skin} \in [0,1]^{n\times m}$, which facilitates preservation of facial characteristics unrelated to the change in the targeted semantics. The second one is determined (dynamically) based on the targeted facial attribute, $S_{tar}(I) \in [0,1]^{n\times m}$.

[1]The extended latent space $\mathcal{W}^+$ allows for the embedding of arbitrary facial images in StyleGAN2 and represents an extension of the model's original latent space to all layers of the model.

Fig. 3: Illustration of the ground truth used to train the face parser S. The presented regions are associated with specific (local) attributes that can be edited, as summarized in Table I.

D. Supported Attributes and Local Embedding

MaskFaceGAN is designed around the assumption that changes in certain attributes are reflected only through changes in spatially local facial regions. As we demonstrate in the experimental section, this assumption is not only reasonable, but also helps to mitigate issues related to attribute entanglement often observed with competing techniques. Based on the attribute annotations and face components available with popular face datasets, such as CelebA [49] and CelebAMask-HQ [45], we focus on 14 facial attributes that can be associated with specific facial regions, as summarized in Table I. However, as shown in section V-F, given a slight method modification, other attributes can be edited as well (e.g., age and gender).

TABLE I: Attributes supported for editing by MaskFaceGAN and corresponding facial areas manipulated by the proposed approach to enforce desired semantics.

Face attribute (for editing)	**Face region (returned by S)**
Blond, Brown, Black, Grey, Straight, and Wavy hair	Hair
Wearing lipstick, Smiling, Mouth slightly open	Lower and Upper lip, Mouth
Bushy eyebrows, Arched eyebrows	Left and Right eyebrow
Pointy nose, Big nose	Nose
Narrow eyes	Left eye, Right eye

The local nature of the editing procedure allows MaskFaceGAN to embed only specific image regions into the StyleGAN latent space (similarly to [16]) and enforce targeted semantics only within local spatial areas. For example, we only embed the nose region along with the adjacent face region when targeting the "Pointy nose" attribute and optimize the latent code with the goal of ensuring the desired semantics exclusively within the nose area – irrespective of the visual changes to other facial parts[2]. This is achieved through a series of carefully designed optimization constraints presented in the next section.

E. Latent Code Optimization

Appearance Preservation. The goal of facial attribute editing is to alter specific (targeted) image semantics, while preserving all (or most) other visual characteristics of the input images. To ensure that image regions not associated with the targeted attributes are preserved, a (local) appearance–preservation constraint is used during optimization. The constraint is defined as a masked mean squared error (MMSE):

$$\mathcal{L}_M = ||S_{skin}(I) \odot (I_G - I)||_2^2, \tag{2}$$

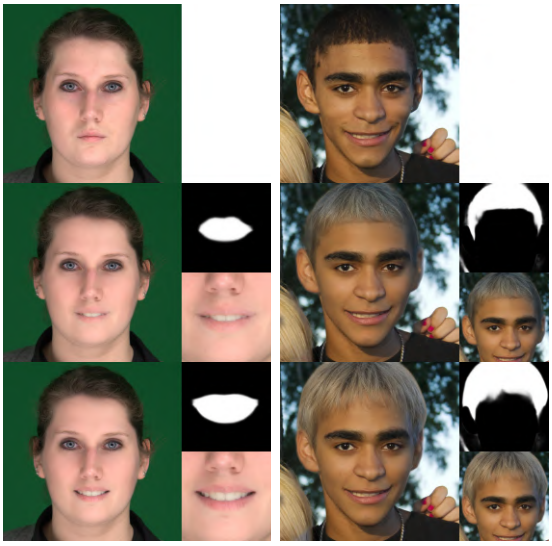
where $S_{skin}(I)$ is a probabilistic mask produced by the face parser S, $\odot$ is the Hadamard product, and $I_G = G(w, n)$. $\mathcal{L}_M$ encourages the generated image I_G and the input image I to be as similar as possible within the area defined by S_{skin}.

Semantic Content. The constraint in Eq. (2) forces certain image regions in I_G to be preserved w.r.t. I, while the rest is allowed to change. MaskFaceGAN, thus, synthesizes the remaining image pixels in accordance with the targeted semantics by considering a semantic–content constraint in the optimization procedure. The constraint ensures that the latent code w produces an image I_G with the desired facial attributes and is defined as the average Kullback–Leibler (KL) divergence D_{KL} between the smoothed ground truth probability distribution and classifier predictions for the targeted attribute(s) [50], i.e.:

$$\mathcal{L}_C = \frac{1}{K} \sum_{k=1}^{K} D_{KL}(C_k(I_G), y_k), \tag{3}$$

where K denotes the number of targeted facial attributes, C_k stands for the attribute classifier prediction corresponding to the k-th attribute and $y_k \in \{\epsilon, 1-\epsilon\}$ is the smoothed ground truth that denotes the absence or the presence of the desired attribute, respectively. The value of ϵ can be used to set the intensity of the desired attribute, e.g., the intensity of lipstick presence when editing the "Wearing lipstick" attribute.

[2]The modified visual information in other parts is corrected for through the blending procedure in the last step of the MaskFaceGAN editing procedure.



Smiling Straight hair

Fig. 4: Impact of flexible spatial constraints on the visual appearance of two sample images with two targeted attributes. The first row shows the original images, the middle row shows the editing results and the target region $S_{tar}(I)$ without flexible spatial constraints, and the final row shows the results with flexible constraints. Note how better semantics can be captured for the both the "Smiling" as well as the "Straight hair" target attributes by relaxing the spatial constraints.

Target Region Shape. Because image content with the targeted semantics is first synthesized by MaskFaceGAN and later blended with the original image, it is critical that the shape of the targeted facial regions is preserved. To this end, the proposed approach constrains the shape of the targeted region with the help of the face parser S using:

$$\mathcal{L}_S = ||S_{tar}(I) - S_{tar}(I_G)||_2^2, \tag{4}$$

where S_{tar} is again a probabilistic mask of the spatial region associated with the targeted attribute – see Table I.

Constraining the shape of the manipulated face components was found to be especially important for hair editing. If the synthesized hair in I_G does not cover at least the original hair region in I, the blending steps generates visible artefacts that affect the perceived quality of the edited images.

Component Size. While the main goal of MaskFaceGAN is to produce convincing manipulations of existing image content, additional components can be incorporated into the framework to enable further editing capabilities. Specifically, MaskFaceGAN can manipulate the size of the target facial region by considering a suitable optimization objective. Let the portion of the image $s_{tar} \in [0, 1]$ covered by a given target component be defined as:

$$s_{tar}(I) = \frac{\sum_{x,y} S_{tar}(I)}{|S_{tar}(I)|}, \tag{5}$$

where the operator $|S_{tar}(I)|$ denotes the number of pixels in S_{tar}. To be able to scale the size of the targeted facial region we introduce a scaling factor α and integrate it into an objective that considers the component size when optimizing for the latent code w. The objective is defined as the KL divergence between the initial component portion $s_{tar}(I)$ and the desired portion $s_{tar}(I_G)$ in the generated image I_G, i.e.:

$$\mathcal{L}_P = D_{KL}\left(s_{tar}(I_G), \alpha s_{tar}(I)\right). \tag{6}$$

We note that this term is optional and can be excluded from the optimization procedure by setting the corresponding weighting factor to 0 - see final objective in Eq. (9) for details.

Flexible Spatial Constraints. The appearance–preservation and target–shape optimization constraints, defined in Eqs. (2) and (4), impose significant limitations on the spatial regions associated with the targeted facial attributes. The appearance–preservation constraint does not allow to grow relevant facial components if they overlap with the skin region. Similarly, the target–shape objective forces the edited image to have exactly the same target component shape as the original image, which in some cases might not be desired. For example, when editing hair color, the target shape needs to be preserved, but when editing hair shape (e.g., "Straight hair" or "Wavy hair") modifications of the target image region must be allowed.

To deal with such issues, we relax the optimization constraints from Eqs. (2) and (4) and incorporate mechanisms into MaskFaceGAN that allow for more flexible spatial editing. Specifically, in each iteration of the optimization procedure, we first compute the target region on the generated image $S_{tar}(I_G)$. Next, we subtract this region from $S_{skin}(I)$ for the appearance–preservation constraint to preserve less pixels. For the target–shape objective, the region is added to the target region of the original image $S_{tar}(I')$ to allow region growth. Here, we also require that the combined region covers at least the original component shape to avoid visual artefacts. The final (relaxed) appearance–preservation $\mathcal{L}_M$ and target–shape $\mathcal{L}_S$ constraint used by MaskFaceGAN are, hence, defined as:

$$\mathcal{L}_M = ||\min(S_{skin}(I) - S_{tar}(I_G), 0) \odot (I_G - I)||_2^2, \text{ and} \tag{7}$$

$$\mathcal{L}_S = ||\max(S_{tar}(I) + S_{tar}(I_G), 1) - S_{tar}(I_G)||_2^2, \tag{8}$$

where $\min$ and $\max$ denote pixel–wise minimum and maximum operations. The impact of these constraints on the appearance of a few sample images is shown in Fig. 4.

Final Objective. The overall optimization objective ($\mathcal{L}_w$) of MaskFaceGAN is defined as a linear combination of the objectives/constraints described above, i.e.:

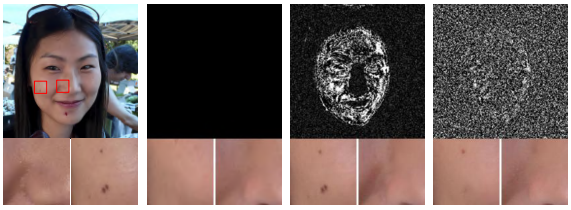
$$\mathcal{L}_w = \lambda_M \mathcal{L}_M + \lambda_C \mathcal{L}_C + \lambda_S \mathcal{L}_S + \lambda_P \mathcal{L}_P, \tag{9}$$

where λ_M, λ_C, λ_S and λ_P are weighting factors the control the contribution of the individual objectives. Minimizing $\mathcal{L}_w$ leads to an optimized latent code w with respect to the targeted semantics a that can be used to generate a (intermediate) synthetic attribute edited image I_G.

F. Noise Component Optimization

While the semantic content of the edited images is controlled by the latent code w, the high-frequency facial details that ensure photo realism are defined by the noise components n. After the latent code w is optimized, w is frozen and MaskFaceGAN proceeds to optimize n, similarly to [16]. Two key issues are considered during optimization, i.e.:

- *Adversarial solutions:* Due to the high-dimensionality of n, a naive optimization procedure based on Eq. (10) can lead to editing results akin to adversarial examples, i.e., generated images that satisfy all constraints but do not exhibit the desired semantics. To avoid such settings the objectives related to semantics and target–region shape are not considered when optimizing for n.
- *Overfitting:* The optimization procedure can lead to solutions that perfectly reproduce all stochastic details of the original face (e.g., freckles, wrinkles) except for facial areas altered by the editing procedure. This mismatch between preserved and altered image regions results in unnatural appearances and a "copy-paste" look. To avoid such overfitting and ensure a reasonable amount of details in the preserved as well as generated image regions, a noise regularization term is used, similarly to [17].



Original $n = 0$ $\lambda_R = 0$ $\lambda_R \neq 0$

Fig. 5: Embedding quality with respect to different noise optimization settings when editing a nose-region attribute. The first column shows the original image and a closeup of two face regions. Without noise optimization high-frequency details are not embedded, but smooth transitions are generated between the skin and the nose regions (second columns). Optimizing for the noise directly perfectly embeds fine skin details, but does not produce smooth transitions (third column), while the regularized optimization generates a reasonable trade–off between details and transitions (last column).

Based on the above considerations, MaskFaceGAN's noise-related optimization objective takes the following form:

$$\mathcal{L}_n = \lambda_M \mathcal{L}_M + \lambda_R \sum_{i,j} L_{i,j}, \tag{10}$$

where λ_M and λ_R are again weighting factors and the noise regularization term $L_{i,j}$ is defined as:

$$L_{i,j} = \left(\frac{1}{|n_{i,j}|}\sum_{x,y} n_{i,j}(x,y) \cdot n_{i,j}(x-1,y)\right)^2 + \left(\frac{1}{|n_{i,j}|}\sum_{x,y} n_{i,j}(x,y) \cdot n_{i,j}(x,y-1)\right)^2. \tag{11}$$

The goal of this regularization term is to ensure that the noise components follows a normal Gaussian probability distribution by preserving the mean and standard deviations of neighbouring values. At every step of the optimization, each noise component larger than 8×8 is downsampled in a pyramid-like

TABLE II: High–level summary of the dataset and experimental setup used with MaskFaceGAN. Note that datasets with different characteristics and diverse face images were selected for the experiments to demonstrate the merits of the proposed approach. The number of test images reported is used for the quantitative evaluations, e.g., the user study.

Dataset	Image Resolution	Purpose	#Training Images†	#Test Images	Variability Sources
FFHQ [17]	1024×1024	Training of G	$70,000$	n/a	Age, ethnicity, background
CelebA [49]	178×218	Training of C	$182,636$	n/a	Age, ethnicity, background,accessories
CelebA–HQ [45]	1024×1024	Training of S, testing	$24,183$	100	Age, ethnicity, background, accessories
Helen [51]	>500 in width	Testing	n/a	118	Age, ethnicity, background clutter
SiblingsDB–HQf [52]	4256×2832	Testing	n/a	163	Age, gender

† The number of training images reported includes both training and validation data.

fashion to a resolution of 8×8 by averaging 2×2 neighbouring values. In the above equation, $n_{i,j}$, thus, denotes the i-th noise component at the original resolution ($j = 0$) or a given level of the downsampling pyramid ($j > 0$). The number of elements of $n_{i,j}$ is denoted as $|n_{i,j}|$ and the corresponding regularization term as $L_{i,j}$. The impact of the term is illustrated in Fig. 5.

G. Blending

In the final step, MaskFaceGAN blends the image generated based on the optimized latent code w and noise components n, $I_G = G(w, n)$, with image regions in the input image I that were not considered during optimization. These regions correspond to the background and non-edited facial components. To facilitate this step, a blending mask is computed as $B = S_{skin}(I) + S_{tar}(I)$ for most attributes and $B = S_{skin}(I) + S_{tar}(I_G)$ when editing hair shape to account for potential modification of the hair region. The final image I' is generated as

$$I' = B \odot I_G + (1 - B) \odot I. \tag{12}$$

The blending step is visualized in the bottom left of Fig. 2. Note that the blending operation considers the skin region from the optimized image I_G. This is needed to allow MaskFaceGAN to also change the size and shape of the targeted attributes in a visually convincing manner, with smooth transitions and without visual artefacts.

IV. Experimental Setup

A. Datasets and Experimental Splits

Five high-resolution images datasets are used in the experiment with MaskFaceGAN, i.e., Flickr-Faces-HQ (FFHQ) [17], CelebA [49], CelebA–HQ [45], Helen [51] and SiblingsDB–HQf [52]. The datasets were selected based on different criteria, such as dataset size, image resolution, image quality, and available annotations. A brief summary of the datasets and experimental splits used is given below:

- **Flickr-Faces-HQ (FFHQ)** [17] contains $70,000$ high quality face images at a resolution of 1024×1024 pixels. Images from the dataset were crawled from Flickr and contain considerable variation in terms of age, ethnicity and image background. FFHQ is used to the train the generator model (G) of MaskFaceGAN.
- **CelebA** [49] is a large–scale face image dataset, consisting of more than $200,000$ celebrity images. Each image is annotated with identity information, 40 binary attributes and 5 landmark locations. CelebA is used to train the attribute classifier (C) of MaskFaceGAN in accordance with the official training and validation splits [49].
- **CelebA–HQ** [27] is a recent dataset, derived from CelebA. It consists of $30,000$ aligned, quality-improved images processed by JPEG artefact removal and super-resolution. For the experiments, the CelebAMask–HQ[3] version from [45] is utilized, which comes with pixel-level annotations of semantic face classes. CelebA–HQ is used to train the face parser (S) and to evaluate the performance of MaskFaceGAN. Training and validation sets are defined by mapping the predefined experimental splits of CelebA to CelebA–HQ. For the quantitative evaluations (user study), a disjoint set of images is selected based on perceived face quality and data diversity.
- **Helen** [51] consists of 2330 high-quality face images annotated with facial landmark locations. In comparison to CelebA-HQ, there are considerably larger variations in age, race and lightning conditions present in this dataset, posing a greater challenge to face editing technology. A subset of test images is selected for the quantitative evaluations of MaskFaceGAN based on similar criteria as discussed above for CelebA-HQ. The selected test images are manually annotated with binary attributes.
- **SiblingsDB-HQf** [52] contains frontal, expressionless images of 184 subjects – 92 sibling pairs with a resolution of 4256×2832. Images in this dataset were captured in front of a uniform background and under controlled lightning. We process the images with the CelebA–HQ pipeline using cropping and alignment [27], then manually annotate them with binary attributes.

MaskFaceGAN is applied for attribute editing at a resolution of 1024×1024 pixels with all experimental datasets. The facial images are, therefore, rescaled where necessary before applying the proposed face editing approach. A high–level summary of the datasets and experimental splits is presented in Table II. The reported number of test image corresponds to the amount of face imagery used for the quantitative evaluations.

B. Implementation details

The models used by MaskFaceGAN are implemented with publicly available source code. Further details are given below.

- **The Generator** (G) of MaskFaceGAN is implemented using the official StyleGAN2 release [17] to foster reproducibility and ensure a fair comparison with competing techniques from the literature designed around this model.

[3]We use the *CelebA–HQ* to refer to this dataset hereafter for brevity.

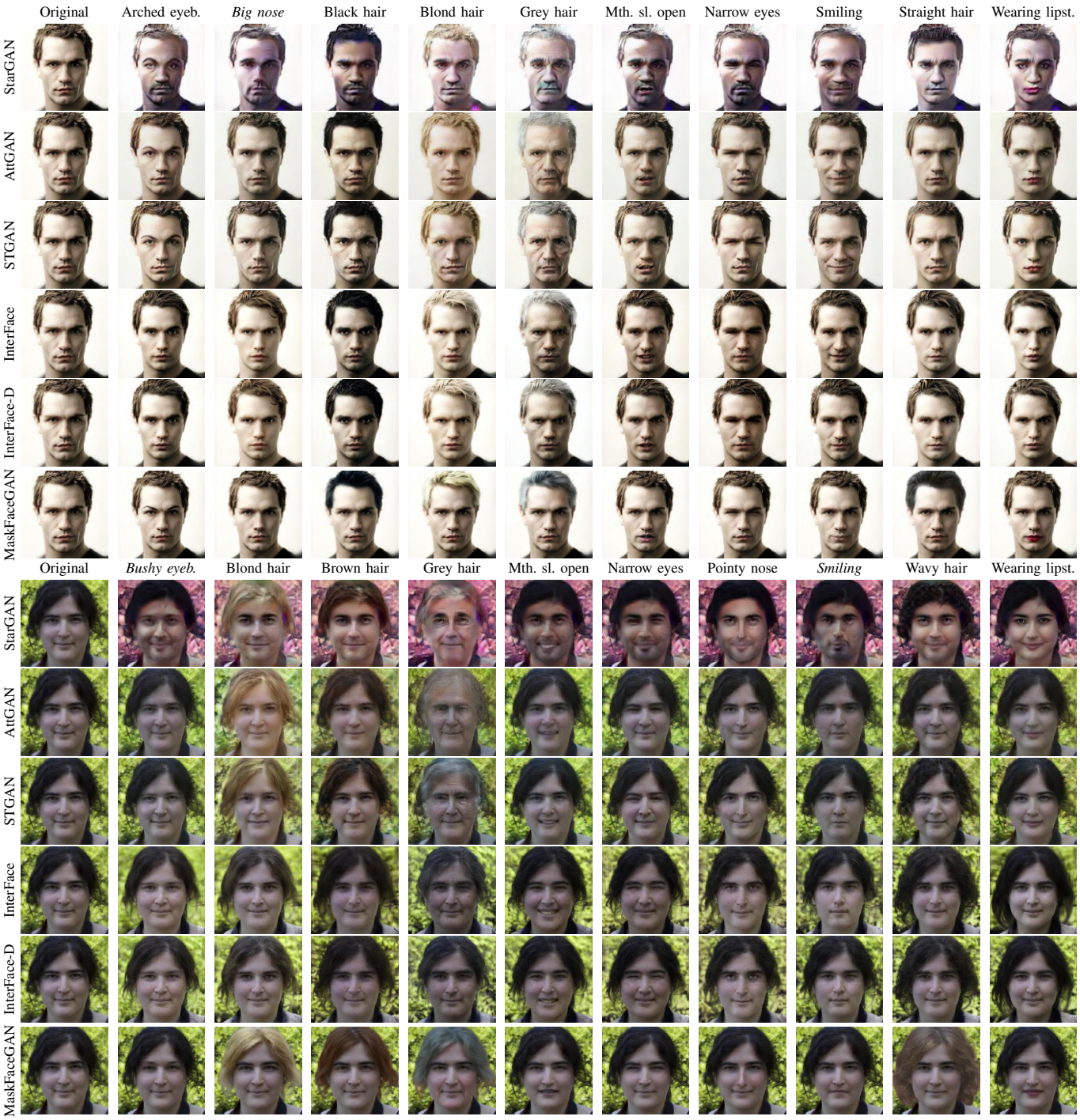


Fig. 6: Comparison of MaskFaceGAN and five state–of–the–art attribute editing models from the literature. Editing results are presented for 14 distinct facial attributes with spatial correspondences. For attributes already present in the image, editing inverts the result (e.g., removes the lipstick for "wearing lipstick" if it is already there) - displayed in italic. Results on the top correspond to a sample image from CelebA–HQ and results at the bottom to an image from Helen. Best viewed zoom in.

- **The Attribute Classifier** (C) is implemented based on the model from [47] and trained on CelebA [49] for 16 epochs with weighted binary cross entropy to account for the class imbalance in the training data. The learning rate is initially set to 0.05 and decayed to 0.005 on the $40,000$–th training step. The model is optimized with the Nesterov momentum algorithm [53] using a batch size of 32. Augmentations with random horizontal flipping and affine transformations are used to avoid over-fitting.
- **The Face Parser** (S) is trained on the CelebA-HQ dataset to generate segmentation masks of the following 9 classes: "hair", "skin", "eyebrows", "nose", "eyes", "mouth", "neck & clothes", "earrings", and "ears". Weighted cross entropy is again used as the learning objective. The model is learned for 5 epochs using the Adam optimizer [54] with a learning rate of $3 \cdot 10^{-4}$ and

TABLE III: FID scores produced by the evaluated models on three experimental datasets (lower is better).

Method	CelebA–HQ	Helen	SiblingsDB–HQf
StarGAN [5]	140.87	152.64	176.84
AttGAN [7]	67.55	79.90	81.34
STGAN [8]	49.53	57.44	48.88
InterFaceGAN [9], [14]	79.79	90.93	83.40
InterFaceGAN–D [9], [14]	76.87	90.90	81.57
MaskFaceGAN (ours)	**32.56**	**34.00**	**34.53**

a batch size of 24. Data augmentation includes horizontal flipping and affine transformations.

The latent code optimization procedure is conducted in several stages. First, w is initialized with the mean latent code $\bar{w}$, computed from $50,000$ sampled codes $w \in \mathcal{W}^+$. Next, it is optimized w.r.t. Eq. (2), so it approximately corresponds to the target face. This initial step was found to be important for the visual quality of the edited images. Finally, the remaining terms are added to enforce semantic and spatial constraints. Similarly to [16], the noise component n is set to 0 and kept constant while optimizing w. Once w converges, it is frozen and the noise component n is optimized independently of w.

To identify parameter values that yield visually pleasing editing results, hyper-parameter optimization is used, resulting in weighting factors of $\lambda_M = 2, \lambda_C = 0.005, \lambda_S = 0.5, \lambda_R = 1$. We set $\lambda_S = 0$ for operations where no hair editing is done. The default value for Eq. (3) is set to $\epsilon = 0.05$. The Adam algorithm [54] is again for the optimization process. The learning rate is set to 0.001 for the latent code w and to 0.1 for the noise component n. Additional implementation details can be found in the publicly released code of MaskFaceGAN.

C. Methods

MaskFaceGAN is evaluated in comparisons with multiple competing models, i.e., StarGAN [5], AttGAN [7], STGAN [8] and two versions of the InterFaceGAN approach from [9], [14]. For a fair comparison, StarGAN, AttGAN and STGAN are trained on the same attributes as MaskFaceGAN (see Table I), using the models' official code repository. We implement InterFaceGAN [9] on the StyleGAN2 latent space, following the linear SVM framework. In addition to the vanilla InterFaceGAN, the authors of [14] also introduced the concept of *conditional manipulation* that tries to disentangle attributes when editing facial images. We also consider such type of model in the experiments and denote it as InterFaceGAN-D hereafter. Both InterFaceGAN models edit images by moving latent codes along attribute–dependent latent space directions. The magnitude of this displacement/movement is set to 1 based on preliminary experiments.

Note that the implemented models manipulate images at different resolutions, i.e., StarGAN produces edited images of 128×128 pixels, AttGAN and STGAN generate 384×384 images, while InterFaceGAN, InterFaceGAN–D and MaskFaceGAN edit images at a resolution of 1024×1024 pixels. We note again that MaskFaceGAN is specialized towards *local face image editing*, and therefore excels at manipulating facial attributes that can be associated with specific image regions.

TABLE IV: User study results, where human raters were shown editing results of all tested models and asked to select the best one. Reported is the fraction of times [in %] a model was chosen as the overall best (higher is better).

Method	CelebA–HQ	Helen	SiblingsDB–HQf
StarGAN [5]	2.96%	2.76%	4.26%
AttGAN [7]	4.95%	6.53%	6.38%
STGAN [8]	13.93%	8.70%	12.60%
InterFaceGAN [9], [14]	7.49%	18.62%	18.80%
InterFaceGAN–D [9], [14]	9.99%	14.71%	10.82%
MaskFaceGAN (ours)	**60.68%**	**48.68%**	**47.14%**

TABLE V: User study results, where human raters were asked to rate the quality of the edited images on a 5-point Lickert scale (higher is better). Reported is the average score and corresponding standard deviation.

Method	CelebA–HQ	Helen	SiblingsDB–HQf
StarGAN [5]	1.46 ± 0.92	1.30 ± 0.63	1.52 ± 0.89
AttGAN [7]	2.85 ± 1.01	2.39 ± 1.06	2.48 ± 0.90
STGAN [8]	3.07 ± 1.13	2.44 ± 1.12	2.66 ± 0.98
InterFaceGAN [9], [14]	3.00 ± 1.03	3.03 ± 1.18	3.29 ± 1.02
InterFaceGAN–D [9], [14]	2.94 ± 1.12	2.78 ± 1.22	3.12 ± 1.10
MaskFaceGAN (ours)	$\mathbf{4.07 \pm 1.21}$	$\mathbf{3.80 \pm 1.23}$	$\mathbf{3.85 \pm 1.15}$

V. Results and Discussion

This section reports results that: (i) compare MaskFaceGAN to state-of-the-art attribute editing models, (ii) highlight some unique characteristics of the proposed approach, (iii) explore the contribution of various components through an ablation study, (iv) study the impact of the attribute classifier and face parser, (v) provide insight into the optimization procedure and blending, (vi) illustrate MaskFaceGAN's global editing capabilities, and (vii) analyze its limitations.

A. Comparison to the State-Of-The-Art

Visual Analysis. We first demonstrate the capabilities of MaskFaceGAN for the task of single attribute editing and include the 14 binary attributes from Table I in the analysis. Images from different datasets are used for the experiments to explore the generalization capabilities of the evaluated approaches across various data distributions. If a face already exhibits a given attribute (e.g., a face wearing lipstick), we generate *inverted* attributes, (i.e., a face without lipstick).

Fig. 6 compares editing results produced by MaskFaceGAN and the five competing models on a couple of sample images from CelebA-HQ and Helen. Note that 10 attributes are considered per example image to ensure a reasonable image size for the presentation. Among the encoder–decoder models, StarGAN generates the highest amount of visual artefacts and also introduces a background change for some of the attributes, as illustrated with the sample image from Helen in Fig. 6. AttGAN and STGAN produce more convincing results, but still induce a certain amount of visual artefacts. These can, for example, be seen with the "Mouth slightly open" attribute in Fig. 6. The artefacts generated by the encoder-decoder methods stem from difficulties in balancing multiple loss terms commonly used when training such methods.

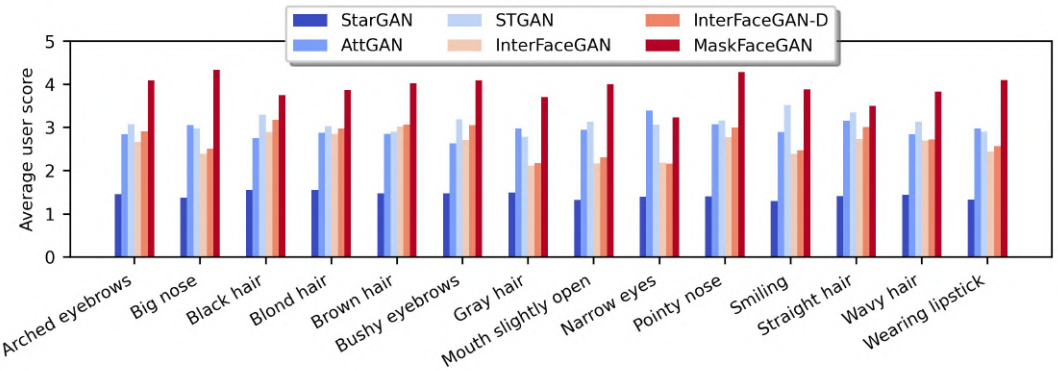


Fig. 7: Comparison of user study scores, averaged across all three dataset for individual attributes. As can be seen, MaskFaceGAN achieves highly competitive results with all targeted attributes. The figure is best viewed in color.

Fig. 8: Editing multiple attributes with MaskFaceGAN. Every image is the result of a separate optimization procedure and is generated independently from all others.

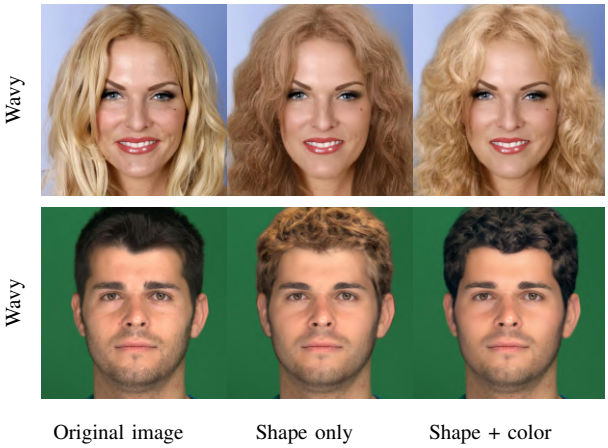


Fig. 9: Editing hair shape with MaskFaceGAN. Editing only the hair shape can lead to changes in hair color (second column). Adding hair color (from the input image) as an additional optimization constraint helps towards preserving the input hair color semantics.

InterFaceGAN and InterFaceGAN–D are most closely related to MaskFaceGAN and achieve higher–quality editing result than the encoder–decoder models due to the use of the StyleGAN2 generator. The vanilla version of InterFaceGAN yields convincing target semantics, but due to the information entanglement in the latent codes, often changes correlated attributes in the process. This is best seen with the "Grey hair" attribute in Fig. 6, where the edited faces appear much older than the originals. InterFaceGAN–D is able to remove some of this entanglement, but this requires a manual analysis of attribute correlations to exclude unwanted facial semantics from the editing procedure. We also observe an interesting behavior with the InterFaceGAN models, in that the same hyper–parameter setting (i.e., the magnitude of the latent code movement), results in attribute changes of different intensity for images of different characteristics – see, for example, the "Blond hair" results in Fig. 6.

Compared to the competing models, the proposed MaskFaceGAN approach: (i) exhibits better disentanglement characteristics due to the latent space optimization procedure, which relies on semantic and spatial constraints, (ii) ensures artefact–free high–resolution attribute editing with convincing image semantics, (iii) preserves important image details (e.g., facial areas not related to the target attribute or background), and (iv) does not require manual hyper–parameter tuning for each probe image separately.

Quantitative evaluation. To evaluate attribute editing performance in a quantitative manner, prior works [7], [8] reported a measure quantifying attribute generation accuracy. Because MaskFaceGAN tries to maximize this exact measure during latent code optimization, we use an alternative approach to ensure a fair comparison. Specifically, we first report Fréchet Inception Distances (FID) to quantify performance and then present results of a user study, similarly to [8].

- **FID Score Analysis.** The Fréchet Inception Distance (FID) [55] represents a common measure of image quality, predominantly used in the evaluation of GANs. We report FID scores for each dataset considered in our evaluation by first generating attribute specific FID scores and then averaging over all attributes. The facial images are rescaled to 299×299 pixels before extracting features. Table III shows that MaskFaceGAN achieves the lowest FID scores on all three test datasets, significantly outperforming all five competing editing models. The lower scores can mostly be attributed to the high quality of the edited images and lack of artefacts, which are the

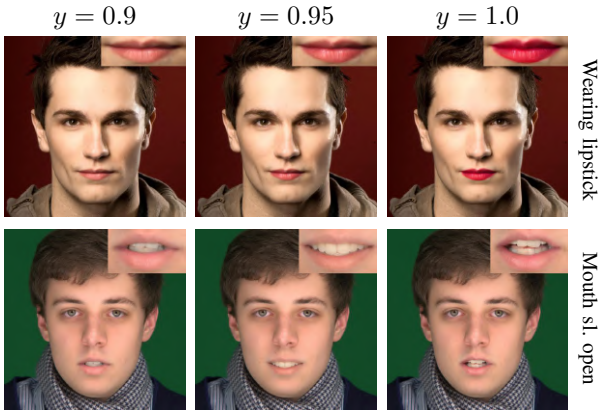


Fig. 10: Visual examples of MaskFaceGAN's capability to control attribute intensity during editing.

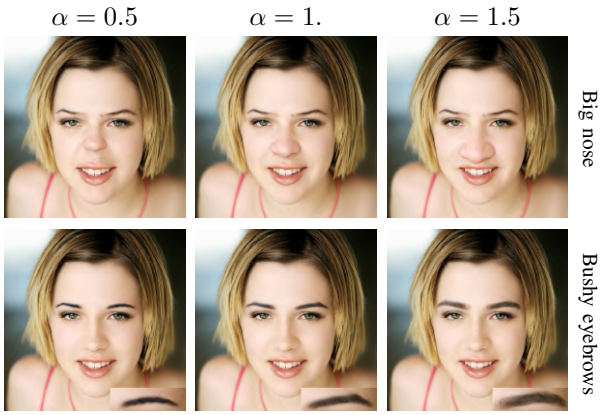


Fig. 11: Visual examples of MaskFaceGAN's capability to control the size of the edited facial components.

result of spatially constrained image modifications that only alter a small portion of the image for a given target attribute, while keeping other parts of the images intact.

- **User Study.** Following [8], we conduct a user study using a crowdsourcing platform to analyze the quality of the edited images. Here, the users (raters) were shown edited images of all considered models and asked to select the most convincing one based on the following instructions: *"Choose the image that changes the attribute more successfully, is of higher image quality and better preserves the identity and fine details of the source image."*. Additionally, they were also instructed to rate images on a 5–point Likert scale, where a higher number represents better image quality. A single user study covered all test images from a given dataset and was performed over all 14 attributes. Images were shown in random order for a fair comparison. The results, reported in Tables IV and V, show that MaskFaceGAN was most frequently selected as the best among the evaluated techniques and also received the highest average scores (on the 5–point Likert scale) on all three dataset. These observations are further supported by the results in Fig. 7, where user scores are reported for each edited attribute separately. The reported results speak of the excellent performance of MaskFaceGAN and competitiveness with respect to existing models.

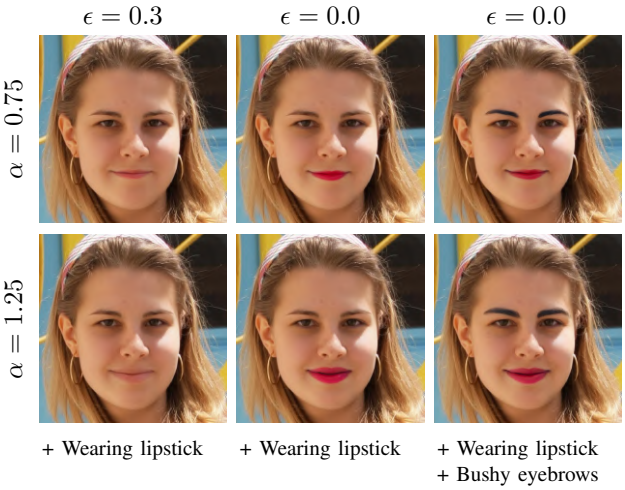


Fig. 12: An example of simultaneous component size manipulation and attribute intensity control. Results are shown for editing a single (i.e., "Wearing lipstick") or two attributes (i.e., "Wearing lipstick" and "Bushy eyebrows") at the same time.

B. Characteristics of MaskFaceGAN

MaskFaceGAN exhibits several desirable characteristics, such as the capability to (i) edit multiple attributes with a single optimization procedure, (ii) control the intensity of edited attribute, and (iii) modify the size of the edited region. We illustrate these characteristics through several visual examples.

Multiple Attribute Editing. By averaging the KL divergence of the semantic constraint over multiple attributes, MaskFaceGAN can edit multiple binary attributes through a single optimization procedure. Examples of such editing results are presented in Fig. 8 for different numbers of attributes, i.e., $K = \{1, 2, 3\}$. Two interesting observations can be made here: (i) even when multiple attributes are edited, the results are still visually convincing and artefact-free, and (ii) the joint optimization of several attributes retains considerable correspondence with the original image.

The multiple-attribute editing capabilities of MaskFaceGAN are especially useful when editing hair shape. Because the model is not explicitly aware of characteristics of the original facial region considered during the editing process, it can in a limited number of cases also alter some additional attributes in addition to the targeted attribute, e.g., change the hair color when hair shape edits are targeted. While this may not be desired, MaskFaceGAN can address such problems by defining multiple target attributes. For example, when editing hair shape, all other hair-related characteristics considered by the attribute classifier C can be set to the same target value as in the input image. In Fig. 9, we illustrate this characteristic on a couple of sample images. Observe how the inclusion of hair color in the optimization procedure helps to better preserve the initial image properties when editing hair shape.

Attribute Intensity Control. MaskFaceGAN's semantic constraint is defined by the KL divergence between the predictions of the attribute classifier (C) and the corresponding ground truth. Because the ground truth is smoothed and for a given attribute consists of $y \in \{\epsilon, 1 - \epsilon\}$, varying the smoothing parameter ϵ affects the strength (or intensity) of

Fig. 13: Qualitative results of the ablation study. The figure shows (left to right): (i) original images, (ii) results without noise optimization and without the shape constraint, (iii) effect of the noise optimization, and (iv) results with noise optimization and the shape term enabled.

the targeted attribute in the edited images. A few illustrative examples of the impact of ϵ are presented in Fig. 10. As can be seen, MaskFaceGAN allows for fine-grained control over the attribute intensity in the edited images, although the generated variations may not necessarily be smooth w.r.t. the visual appearance change. For example, the intensity change for the "Mouth slightly open" attribute in Fig. 10 does not simply open the mouth more, it actually turns into a half–smile. The results primarily depend on the trained classifier and what it considers an attribute presence with $1 - \epsilon$ probability.

Component Size Manipulation. MaskFaceGAN can be adapted to include additional constraints. An example constraint is the desired size of the facial component being manipulated. This is done by including the size manipulation objective from Eq. (6) in the overall optimization objective in Eq. (9). In Fig. 11 we display results when specifying the portions of the original component size for $\alpha = \{0.5, 1.0, 1.5\}$. The presented examples show MaskFaceGAN's capability to change the size of facial attributes in a photo realistic manner.

Combining Editing Constraints. Multiple attribute editing, intensity control and component size manipulation can also be used simultaneously to change several aspects of the input face image with a single application of MaskFaceGAN. Fig. 12 presents an example, where various aspects of the "Wearing lipstick" and "Bushy eyebrows" attributes are manipulated. Note that despite considerable changes to different facial attributes, the results still appear visually convincing.

C. Ablation Study

To evaluate the impact of different MaskFaceGAN components on the editing quality, we perform an ablation study on CelebA-HQ. Specifically, we focus on two major components: (i) the shape constraint from Eq. (4), and (ii) the noise optimization procedure. We note that the shape term only affects the hair region and does not impact other attributes.

TABLE VI: Quantitative results of the ablation study. FID scores computed on CelebA-HQ averaged over all attributes are reported (lower is better).

MaskFaceGAN variant	FID
Local latent code optimization, no shape term	59.90
+ noise optimization	34.86
+ shape term (complete MaskFaceGAN)	**32.56**

Qualitative Analysis. Fig. 13 demonstrates the effect of different MaskFaceGAN settings. When the noise component is not optimized ($n = 0$), the edited images contain low frequency image areas, which is most apparent in the hair region, as shown in the second column of Fig. 13. Similarly, the skin region is also missing details, e.g., beauty marks. The noise optimization ensures that such facial details are present in the image, as can be seen in the third column of Fig. 13.

The absence of the shape term ($\lambda_S = 0$) results in suboptimal blending when dealing with hair modifications. In such settings, the background synthesized by the generator (G) is blended with the original background, resulting in unconvincing results with visible artefacts. The optimization of this term assures that the generator model considers information about the shape of the hair region during the synthesis step and produces photo realistic editing outputs.

Quantitative Analysis. For a quantitative analysis of the ablation results, we report in Table VI mean FID scores computed over the test images of CelebA–HQ dataset and averaged over all attributes. Interestingly, the largest gain is obtained by the noise optimization procedure. Enabling the shape term to ensure blending consistency results in additional FID gains. We hypothesize that these gains are a consequence of visually more convincing images, as shown in Fig. 13

D. Component Analysis

The optimization procedure designed for MaskFaceGAN depends on the utilized attribute classifier (C) and face parser (S). In this section, we analyze the impact of these two components on the editing results.

Attribute Classifier. To explore the impact of the attribute classifier C, we implement 5 additional models in addition to the tree-like classifier from [47] that is used in the original MaskFaceGAN design: two attribute classifiers based on the VGG architecture [56], i.e., VGG16 and VGG19, and three ResNet-based [57] models, i.e., ResNet34, ResNet50 and ResNet101. These models come with different designs and complexity (in terms of parameters). We modify the models to predict multiple attributes by constructing a dedicated fully connected layer at the top with one output for each attribute. The models are trained on the CelebA dataset using the same optimization procedure and learning rate schedule as utilized with the original tree-like classifier [47]. Table VII shows the classification accuracy (averaged across all attributes) on the CelebA test split achieved by the different models. We observe that despite differences in design and complexity most models weigh in at a classification accuracy around 85%, whereas the tree-like classifier performs slightly better.

In Fig. 14 we investigate how the characteristics of the attribute classifiers impact results. As can be seen, all clas-

TABLE VII: Average prediction accuracy on CelebA for the attribute classifiers selected for the component analysis.

Model	VGG16	VGG19	ResNet34	ResNet50	ResNet101	Ours
Accuracy	85.4%	85.5%	85.8%	85.8%	85.6%	90.1%

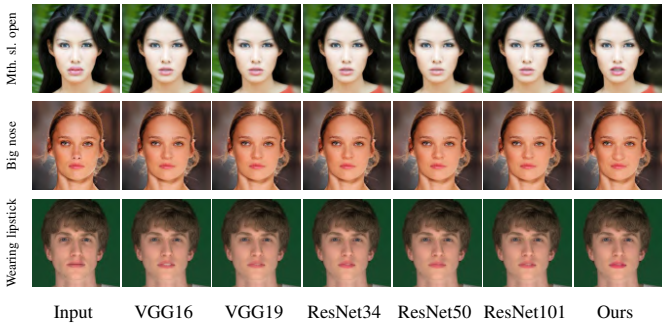


Fig. 14: Impact of different attribute classifiers on the generated edits. For all results, the smoothing factor is set to $\epsilon = 0.05$. The tree-based attribute classifier used in MaskFaceGAN [47] leads to more expressive semantics in the final image compared to other attribute classification models.

sifiers lead to semantically meaningful editing outputs and, even though their performance is close, they produce slight variations in the appearance of the targeted attributes - see, for example, the mouth region in the bottom row of Fig. 14. This observation can be attributed to the different model topologies and differences in the gradients produced during the optimization procedure. Furthermore, it can be noted that the results generated with the tree-based attribute classifier (marked Ours) contain the most expressive semantic content with the most pronounced targeted attributes. We ascribe this fact to the superior classification performance of our attribute classifier, which consequently provides better gradient information compared to the competing models.

Face Parser. MaskFaceGAN uses DeepLabv3 [48] to implement the face parser (S). In this section, we analyze editing results produced with 3 other parsers. We select the following competing models for the comparison: UNet [58] and FCN [59] with two different backbones, i.e., FCN–ResNet50 and FCN–ResNet101, and train them on CelebA-HQ using the same procedure/protocol and optimization method as with DeepLabv3. The performance in terms of the average Intersection over Union (IoU) [60] over the test split of the data is reported in Table VIII. Note that the average IoU scores (in %) are close and vary from around 75% to up to 77%.

A few qualitative editing results produced with the different parsers are shown in Fig. 15. We find the results to be less affected by changes in the parser than by changes in the attribute classifier. The observed differences mostly consist of slight variations in the shape and look of the targeted facial attribute. However, all edits are still semantically reasonable and visually convincing.

E. Optimization Procedure and Blending

In this section, we study the optimization procedure and blending step used in MaskFaceGAN to provide better insight into their behavior and impact on the final results.

TABLE VIII: Average Intersection over Union (IoU) [in %] over the CelebA-HQ test data for the selected face parsers.

Model	UNet	FCN–ResNet50	FCN–ResNet101	DeepLabv3 (ours)
IoU	75.34%	77.18%	76.70%	77.24%

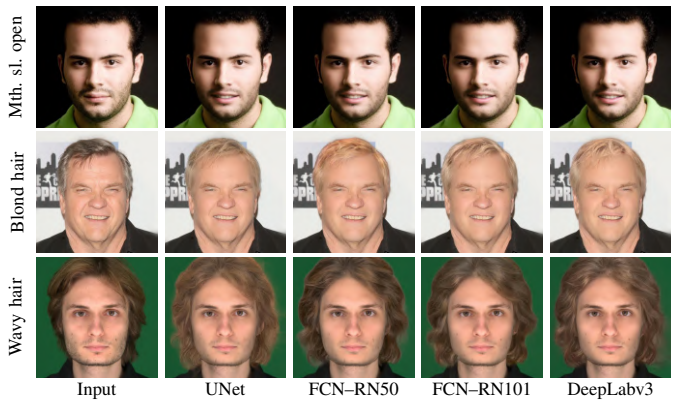


Fig. 15: Impact of different face parsers on the generated edits. The impact on most attributes is negligible, as illustrated, for example, by the "Mouth slightly open" edit in the first row. When the target-region shape loss term is used (hair edits in the second and third row), minute differences can be observed in the generated images.

Optimization Procedure. A two-step process is used in MaskFaceGAN to find the optimal latent representation of the desired output image $G(w, n)$ given the constraints enforced through the attribute classifier and face parser. This process first optimizes the latent code w, and, after convergence, freezes the code and optimizes the high-frequency details encoded through the noise component n. In Fig. 16, we illustrate the evolution of the generated (intermediate) image $G(w, n)$ across different iterations of the optimization procedure for two attributes, i.e., "Black hair" and "Big nose" together with the computed blending mask and final output. Note how the first steps gradually adjust the shape of the targeted facial region and introduce the desired semantics, while the latter steps ensure that high-frequency details are present that contribute towards the realism of the generated images.

Blending. Next, we explore the impact of different blending-mask definitions, which lead to different trade-offs in the generated images. With MaskFaceGAN, the skin-region $S_{skin}(I)$ and targeted-attribute area S_{tar} are used as the basis for the blending process. However, an important consideration here is whether the inclusion of S_{skin} is truly required. Given that the GAN inversion cannot perfectly embed the skin region with the MMSE loss from Eq. (2), a blending mask without the skin region, such as $\hat{B}_1 = S_{tar}(I)$ or $\hat{B}_2 = S_{tar}(I_G)$ could also be used for the blending operation.

In Fig. 17, we show a few illustrative examples, where such masks are utilized, as well as comparative results with the original process using B. In the top row, the blending procedure with $\hat{B}_1$ leads to poor semantics because the targeted region (the mouth) in I_G has grown and changed shape during the optimization process. In the second row, the blending based on $\hat{B}_2$ produces visual artifacts in the blended output $\hat{I}'$, because

Fig. 16: Evolution of the intermediate $G(w, n)$ results throughout the optimization procedure. The leftmost image shows the input I. The next examples show the intermediate results for $G(w, n)$, where the numbers at the bottom correspond to the current optimization iteration of w and n, that is, w_{itr} and n_{itr}, respectively. The initial image ($w_{itr} = 0, n_{itr} = 0$) is always the same. During the w latent code optimization procedure, the face shape and target-attribute appearance is adjusted in accordance with the considered constraints. After convergence, the noise component is optimized to introduce realistic fine image details. The final optimization result I_G is then blended based on B to generate the output image I'. The attributes edited in the presented examples are "Black hair" (top) and "Big nose" (bottom).

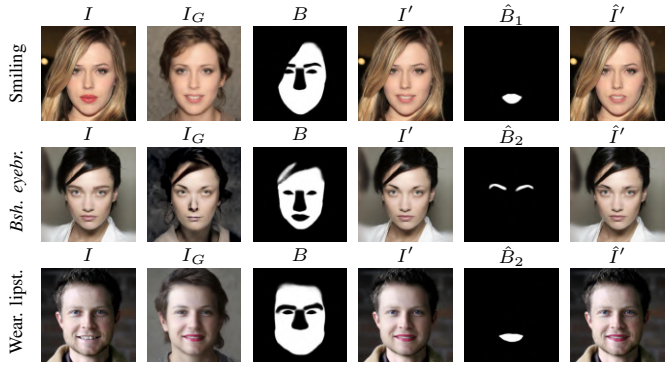


Fig. 17: Analysis of the blending procedure with different blending masks. From left to right: The input image I, the optimized intermediate results I_G, the original blending mask B and output image I', the alternative blending mask ($\hat{B}_1$ or $\hat{B}_2$), and the alternative output $\hat{I}'$. Note how different definitions lead to different trade-offs in the results.

Fig. 18: MaskFaceGAN uses a modified appearance-preservation constraint to enable editing of global image characteristics. The constraint (left) is based on the LPIPS perceptual similarity [61] and is applied over the entire face region, shown in parsed form on the the right.

of shrinkage of the targeted facial region (i.e., the eyebrows) in I_G. In the bottom row, the targeted facial area does not change in shape or size and the exclusion of the skin area in $\hat{B}_2$ leads to editing results with better high-frequency content compared to what is generated with MaskFaceGAN. Nonetheless, both results, I and I', are visually convincing.

While different formulations of blending masks could be defined for specific tasks in MaskFaceGAN, this would introduce an additional layer of complexity to the model. MaskFaceGAN, therefore, opts for an approach that supports attribute growth and shrinking without being overly complex.

F. Global Editing

While the primary purpose of MaskFaceGAN is editing of local attributes that correspond to specific facial regions, the approach can also be extended towards editing of global facial characteristics. The proposed modification allows MaskFaceGAN to edit other attributes, e.g., skin tone, gender, and age. To facilitate global editing, we relax the original appearance-preservation constraint from Eq. (2), so it allows for global image changes. Specifically, instead of using the MMSE-based constraint over the skin area to preserve the initial image appearance, we introduce a perceptual loss over the whole face region, as illustrated in Fig. 18. The perceptual loss allows us to preserve high-level semantics of the original image, while providing room for global appearance modifications. Formally, the modified appearance-preservation constraint is defined as:

$$\widehat{\mathcal{L}}_M = \sum_{l=1}^{L} \left|\left| S^l_{union}(I) \odot \left(\phi^l(I_G) - \phi^l(I)\right) \right|\right|_2^2 \qquad (13)$$

where $\phi^l(\cdot)$ are LPIPS [61] activations from the $l-th$ layer of a pretrained VGG network, S^l_{union} is a mask constructed through the union of all facial regions returned by the face parser (see right part of Fig. 18), downscaled to match the spatial resolution of l-th activations. Similarly to the LPIPS reference implementation, we use $L = 5$ layers to compute the Learned Perceptual Image Patch Similarity (LPIPS). The generated image is then blended with the original based on S_{union}, and not S_{skin}. All other part of the optimization framework are kept unchanged.

To demonstrate the global editing capabilities of MaskFaceGAN, we select two challenging attributes that affect the whole facial area, i.e., "Young" and "Male", and show some sample results in Fig. 19. We again compare the results to the competitors, StarGAN, AttGAN, STGAN and InterFaceGAN. We observe that StarGAN, AttGAN, and STGAN are able to enforce the desired semantics to some extent, but especially with images with cluttered background also often introduce visual artefacts. InterFaceGAN and MaskFaceGAN generate higher-resolution results, again with the desired semantic con-

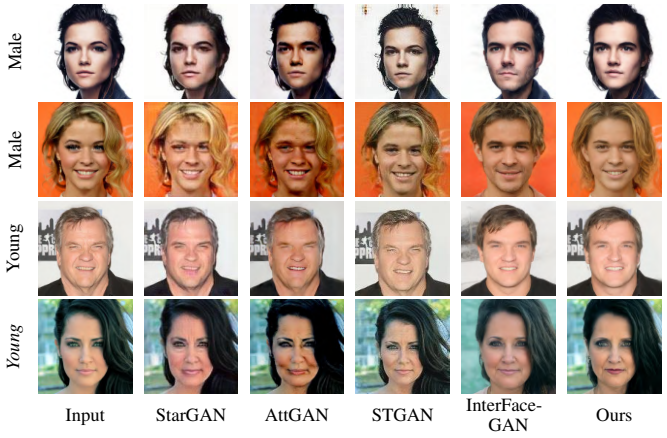


Fig. 19: Illustration of the global editing capabilities of MaskFaceGAN for the "Young" and "Male" attributes. Compared to the competitors, MaskFaceGAN generates higher-resolution edits without visual artefacts and better preserves correspondences with the input image (appearance characteristics, background), while still producing the desired global semantics.

Fig. 20: Examples of MaskFaceGAN limitations. The original input images on the top are edited according to the listed target attributes. MaskFaceGAN is affected by the performance of the attribute classifier (C) and the face parser (S). Difficulties with these components are reflected in the editing results.

tent. However, while the targeted global attributes are clearly expressed with InterFaceGAN, much of the correspondence with the input image is lost. MaskFaceGAN, on the other hand, produces the desired semantic content but also better preserves correspondences with the initial input image both in terms of facial appearance as well as background.

G. Limitations

The results presented so far show that MaskFaceGAN generates competitive (high–quality) editing results when compared to state–of–the–art models from the literature. Nevertheless, the approach still exhibits a number of limitations.

MaskFaceGAN is based on gradient optimization that takes between 2 and 5 minutes per image on a GeForce GTX 1080. In comparison with encoder–decoder methods that are capable of editing images in milliseconds, the proposed approach is slower by orders of magnitude. However, when compared to related methods, e.g., InterFaceGAN [14], the local embedding procedure requires considerably fewer steps to converge.

MaskFaceGAN relies on an attribute classifier (C) to steer the editing process. While this is an effective way of controlling the semantic content in the edited image, it may produce inconsistent results for certain attributes. Our user study showed (see Fig. 7) that especially for the "Narrow eyes" attribute MaskFaceGAN does not outperform STGAN and AttGAN in terms of user scores. An analysis showed that MaskFaceGAN exhibits a tendency to close the eyes instead of trying to narrow them, (see the first column of Fig. 20). While the edited image still looks convincing, such inconsistencies represent one of the limitations of MaskFaceGAN.

The second source of errors is the face parser (S). For images, where S produces incorrect parsing results, the editing procedure operates with inappropriate spatial constraints and results in image changes in (partially) incorrect regions. A couple of examples of such editing results are presented in the second and third column of Fig. 20, where the "Smiling" and "Straight hair" attributes were considered.

VI. Conclusion

In this paper, we introduced MaskFaceGAN, a novel approach to high–resolution face image editing. At the core of the approach is a GAN latent code optimization procedure that generates targeted image regions in accordance with spatial and semantic constraints, enforced by pre–trained face parsing and classification networks. Through rigorous experiments on three face datasets, MaskFaceGAN was shown to convincingly alter a wide variety of facial attributes and ensure competitive performance when compared to the state-of-the-art. Additionally, the approach was demonstrated to enable unique editing characteristics, including attribute intensity control and component size manipulation.

References

[1] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Information Fusion*, vol. 64, pp. 131–148, 2020.

[2] S. Jiang, Z. Tao, and Y. Fu, "Geometrically editable face image translation with adversarial networks," *IEEE Transactions on Image Processing*, vol. 30, pp. 2771–2783, 2021.

[3] V. Mirjalili, S. Raschka, and A. Ross, "Privacynet: semi-adversarial networks for multi-attribute face privacy," *IEEE Transactions on Image Processing*, vol. 29, pp. 9400–9412, 2020.

[4] H. Deng, C. Han, H. Cai, G. Han, and S. He, "Spatially-invariant style-codes controlled makeup transfer," in *Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 6549–6557.

[5] 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 *Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8789–8797.

[6] Y. Jo and J. Park, "Sc-fegan: Face editing generative adversarial network with user's sketch and color," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 1745–1753.

[7] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen, "Attgan: Facial attribute editing by only changing what you want," *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5464–5478, 2019.

[8] M. Liu, Y. Ding, M. Xia, X. Liu, E. Ding, W. Zuo, and S. Wen, "Stgan: A unified selective transfer network for arbitrary image attribute editing," in *Computer Vision and Pattern Recognition (CVPR)*, 2019.

[9] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of gans for semantic face editing," in *Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9243–9252.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems (NIPS)*, 2014.

[11] Z. Shu, E. Yumer, S. Hadap, K. Sunkavalli, E. Shechtman, and D. Samaras, "Neural face editing with intrinsic image disentangling," in *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[12] S. Sengupta, A. Kanazawa, C. D. Castillo, and D. W. Jacobs, "Sfsnet: Learning shape, reflectance and illuminance of faces in the wild," in *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[13] W. Xia, Y. Zhang, Y. Yang, J.-H. Xue, B. Zhou, and M.-H. Yang, "GAN Inversion: A Survey," *arXiv preprint arXiv:2101.05278*, 2021.

[14] Y. Shen, C. Yang, X. Tang, and B. Zhou, "Interfacegan: Interpreting the disentangled face representation learned by gans," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[15] R. Abdal, Y. Qin, and P. Wonka, "Image2stylegan: How to embed images into the stylegan latent space?" in *International Conference on Computer Vision (ICCV)*, 2019, pp. 4431–4440.

[16] ——, "Image2stylegan++: How to edit the embedded images?" in *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[17] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8110–8119.

[18] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *International Conference on Machine Learning (ICML)*, 2019, pp. 7354–7363.

[19] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," in *International Conference on Learning Representations (ICLR)*, 2018.

[20] J. Johnson, A. Gupta, and L. Fei-Fei, "Image generation from scene graphs," in *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[21] O. Ashual and L. Wolf, "Specifying object attributes and relations in interactive scene generation," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 4561–4569.

[22] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2337–2346.

[23] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in *Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1316–1324.

[24] H. Tan, X. Liu, M. Liu, B. Yin, and X. Li, "Kt-gan: Knowledge-transfer generative adversarial network for text-to-image synthesis," *IEEE Transactions on Image Processing*, vol. 30, pp. 1275–1290, 2021.

[25] Y. Yang, L. Wang, D. Xie, C. Deng, and D. Tao, "Multi-sentence auxiliary adversarial networks for fine-grained text-to-image synthesis," *IEEE Transactions on Image Processing*, vol. 30, pp. 2798–2809, 2021.

[26] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2016.

[27] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *International Conference on Learning Representations (ICLR)*, 2018.

[28] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4401–4410.

[29] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *International Conference on Computer Vision (ICCV)*, 2017, pp. 2794–2802.

[30] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning (ICML)*, 2017, pp. 214–223.

[31] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 5767–5777.

[32] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[33] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for gans do actually converge?" in *International Conference on Machine learning (ICML)*, 2018, pp. 3481–3490.

[34] Z. Wang, Q. She, and T. E. Ward, "Generative adversarial networks in computer vision: A survey and taxonomy," *ACM Computing Surveys*, vol. 54, no. 2, pp. 1–38, 2021.

[35] D. Saxena and J. Cao, "Generative adversarial networks (gans) challenges, solutions, and future directions," *ACM Computing Surveys*, vol. 54, no. 3, pp. 1–42, 2021.

[36] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "Gan dissection: Visualizing and understanding generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2019.

[37] A. Jahanian, L. Chai, and P. Isola, "On the" steerability" of generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2020.

[38] L. Goetschalckx, A. Andonian, A. Oliva, and P. Isola, "Ganalyze: Toward visual definitions of cognitive image properties," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 5744–5753.

[39] C. Yang, Y. Shen, and B. Zhou, "Semantic hierarchy emerges in deep generative representations for scene synthesis," *International Journal of Computer Vision*, pp. 1–16, 2021.

[40] S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin, "Pulse: Self-supervised photo upsampling via latent space exploration of generative models," in *omputer Vision and Pattern Recognition (CVPR)*, 2020.

[41] A. Frühstück, K. K. Singh, E. Shechtman, N. J. Mitra, P. Wonka, and J. Lu, "Insetgan for full-body image generation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[42] J. Lin, R. Zhang, F. Ganz, S. Han, and J.-Y. Zhu, "Anycost gans for interactive image synthesis and editing," in *Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 14 986–14 996.

[43] M. Afifi, M. A. Brubaker, and M. S. Brown, "Histogan: Controlling colors of gan-generated and real images via color histograms," in *Computer Vision and Pattern Recognition (CVPR)*, 2021.

[44] T. Portenier, Q. Hu, A. Szabo, S. Arjomand, P. Favaro, and M. Zwicker, "Faceshop: Deep sketch-based image editing," *ACM transactions on graphics*, vol. 37, no. 4, pp. 1–13, 2018.

[45] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 5549–5558.

[46] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *International Conference on Computer Vision (ICCV)*, 2017, pp. 2223–2232.

[47] S. Vandenhende, S. Georgoulis, B. De Brabandere, and L. Van Gool, "Branched multi-task networks: deciding what layers to share," in *British Machine Vision Conference (BMVC)*, 2020.

[48] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," in *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[49] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *International Conference on Computer Vision (ICCV)*, 2015, pp. 3730–3738.

[50] T. Van Erven and P. Harremos, "Rényi divergence and Kullback-Leibler divergence," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3797–3820, 2014.

[51] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang, "Interactive facial feature localization," in *European Conference on Computer Vision (ECCV)*, 2012, pp. 679–692.

[52] A. Vieira, Tiago F.and Bottino, A. Laurentini, and M. De Simone, "Detecting siblings in image pairs," *The Visual Computer*, vol. 30, no. 12, pp. 1333–1345, Dec 2014.

[53] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International Conference on Machine Learning (ICML)*, 2013, pp. 1139–1147.

[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2014.

[55] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in neural information processing systems (NIPS)*, 2017, pp. 6626–6637.

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

[57] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[58] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, pp. 234–241.

[59] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[60] P. Rot, M. Vitek, K. Grm, Ž. Emeršič, P. Peer, and V. Štruc, "Deep sclera segmentation and recognition," in *Handbook of vascular biometrics*, 2020, pp. 395–432.

[61] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 586–595.

MaskFaceGAN: High Resolution Face Editing with Masked GAN Latent Code Optimization – Supplementary Material

Martin Pernuš, *Student Member, IEEE,* Vitomir Štruc, *Senior Member, IEEE,* and Simon Dobrišek, *Member, IEEE*

***Abstract*—In the main part of the paper, we introduced MaskFaceGAN for face editing and presented several experiments to demonstrate its capabilities. This *Supplementary material* provides further details on the proposed approach and reports additional results to highlight its merits. Specifically, the supplementary material: (i) elaborates on the importance and effect of the local GAN inversion process used by MaskFaceGAN, (ii) further motivates the use of spatial constraints for face attribute editing, (iii) presents additional editing results for single attribute editing, attribute intensity control, component size manipulation, multiple attribute editing and compares editing quality at higher resolution, (iv) reports per–attribute results for the ablation experiments, FID analysis and user study, (v) discusses details with respect to the implementation of the user study, (vi) elaborates on the implementation details of the competing models included in the experimental evaluations, and (vii) provides information on the reproducibility of the experiments.**

I. Embedding quality and local GAN inversion

Unlike most competing approaches to face attribute editing that rely on latent code optimization, e.g., [1], [2], MaskFaceGAN does not embed the entire face image into the GAN latent space. Instead, it only embeds facial region(s) needed for editing, which results in higher quality embeddings. In the main part of paper, we showed a visual example to illustrate this characteristic. Here, we report results of a simple experiment aimed at evaluating image quality when embedding smaller image regions and blending the result with the rest of the original input image.

For this experiment, the target face region is embedded with a masked MSE loss (MMSE), as defined by Eq. (2) in the main part of the paper. We optimize the latent code w for 2000 iterations, reconstruct the output image, blend it with the original and then report average PSNR, SSIM, MSE and perceptual loss (PL) (computed over conv1-conv5 features of the VGG ImageNet model as in [3]) scores computed over 10 randomly selected face images from CelebA–HQ. The results reported Table I reflect the similarity between the blended and the original input image and (among others) serve as indicators of the visual quality of the outputs of MaskFaceGAN. We note at this point that image manipulation techniques based on GAN inversion need to ensure that the edited images are as close to the originals as possible and that important characteristics, such as the identity of the subject shown, the background and other semantically critical image aspects are captured by the embedding process.

Table I shows the performance scores achieved by MaskFaceGAN, when embedding different (local) facial regions, and a comparison with the state–of–the–art Image2StyleGAN++ embedding algorithm from [4]. For the latter, the whole face is embedded in the latent space using a combination of pixel–wise MSE and perceptual losses [5], [6]. We observe that choosing to embed a smaller portion of the image results in a considerable gain in embedding quality. It is important to note that the results should not be regarded as an improvement over existing embedding algorithms – one could achieve a perfect score by simply choosing not to embed any part of the image. Nevertheless, the results motivate the idea of focusing the editing procedure on local image areas. This local embedding step allows us to perform a lower number of iterations during the optimization procedure and still generate visually convincing and photo–realistic images. We therefore report the editing results on larger datasets than, for example, [4].

TABLE I: Comparison of StyleGAN latent code embedding quality, where the quality is measured by the similarity of the image generated from the optimized latent code and the original input image. The arrows next to the performance scores indicate whether a higher (↑) or lower (↓) score corresponds to better performance.

Method	MSE $\cdot 10^4$ ↓	PSNR ↑	SSIM ↑	PL ↓
Image2StyleGAN++ [4]	89.9	20.87	0.81	0.23
MMSE – face	24.0	26.73	0.90	0.20
MMSE – no hair	8.0	31.01	0.97	0.06
MMSE – skin only	3.9	34.11	0.99	0.05

II. Spatial constraints and facial attribute correspondence

MaskFaceGAN relies on spatial constraints when optimizing the StyleGAN2 latent–codes for face attribute editing. The targeted attributes, therefore, need to have spatial correspondences in the image and, similarly, visually convincing changes in the attributes need to be reflected in local image regions only. To demonstrate that utilizing spatial constraints for image editing is reasonable, we visualize image areas contributing most to the decision of MaskFaceGAN's attribute classifier C for a number of target attributes in Fig. 1. Here, GradCAM [7] is utilized to generate spatial heatmaps for the visualization.

For this experiment, we extract heatmaps from a subset of images from CelebA–HQ. Due to the tree–like architecture of

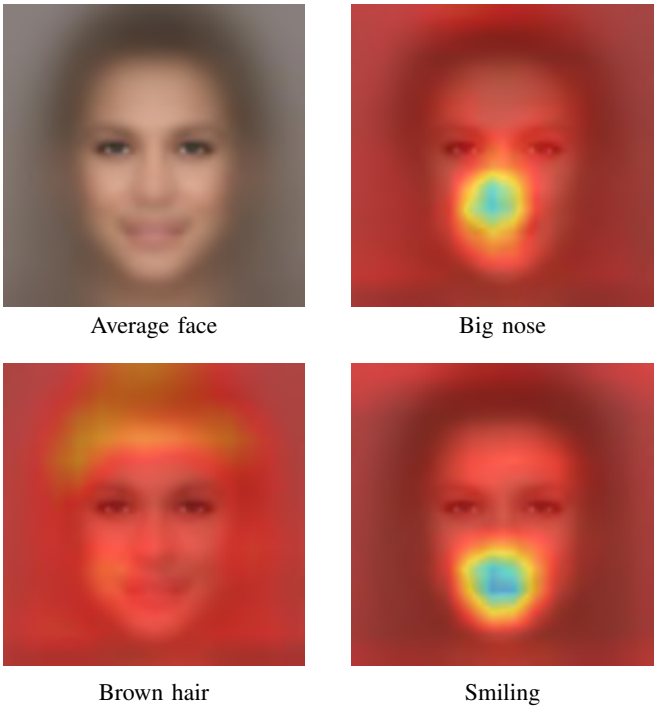


Fig. 1: Visualization of image regions with the greatest impact on the attribute classifier C for a given attribute. The heatmaps were generated with GradCAM. The left image in the top row shows the average CelebA–HQ face. The remaining images contain the average face overlayed with average GradCAM heatmaps for a few sample attributes. Note that the attributes are reflected in local image regions.

the classifier used in our implementation of MaskFaceGAN, only gradients and activations from the leaf convolutional layer of a given target facial attribute are considered. All other leaf convolutional layers do not receive any gradient and, therefore, have zero contribution to the heatmaps. In Fig. 1 the average CelebA–HQ face as well average GradCam heatmaps (computed over a subset of CelebA–HQ images) for a few example attributes are presented. Note that the classifier focuses predominantly on local image regions, suggesting that changing the image within a local image region is a viable approach for editing such attributes.

III. Additional Editing Results

In this section, we present additional visual results for all targeted facial attributes and across images from all three experimental datasets.

A. Single Attribute Editing

Figs. 2, 3 and 4 show editing examples for images from CelebA–HQ, Helen and SiblingsDB–HQf, respectively. The StarGAN model shows a somewhat weaker performance than other editing techniques considered and exhibits limited generalization capabilities across different datasets, as shown in Figs. 3 and 4. AttGAN and STGAN produce higher–quality editing results, but often struggle with the entanglement of different attributes, which is especially apparent with the "Grey hair" attribute. The disentangled version of InterFaceGAN, InterFaceGAN–D, is more successful when editing attributes that exhibit a high degree of entanglement. For example, it relatively convincingly removes age–dependent image characteristics, such as wrinkles, when trying to generate "Grey hair", as shown in Fig. 2. However, disentangling "Pale skin" from hair color does not have any apparent effect compared to the vanilla InterFaceGAN version, as shown in Fig. 2 under "Blond hair" and in Fig. 3 under "Black hair" despite the fact that these attributes are highly correlated. MaskFaceGAN, on the other hand, is less affected by entanglement problems due to the local nature of the editing procedure and produces convincing results for the majority of edited attributes.

B. High Resolution Results

Fig. 5 presents high–resolution editing results for the "Blond hair" attribute. In this example, the StarGAN model produces the most blurry result due to the low image resolution of the model. AttGAN and STGAN achieve better image quality. However, the edited images still exhibit visual artefacts. For the presented results, InterFaceGAN–D is disentangled with respect to the "Pale skin" attribute. Nonetheless, both InterFaceGAN as well as its disentangled version, InterFaceGAN–D, edit the image with very similar results that slightly change the skin tone as well as the color and shape of clothes, the background, the ears and the eye region. The proposed model does not exhibit such problems and preserves both appearance as well as identity of the original face well. Additionally, even at higher resolutions, no apparent artefacts are present in the edited image.

C. Attribute Intensity Control

Fig. 6 shows editing results generated by varying the ϵ parameter of MaskFaceGAN to achieve different intensities of the attribute presence/absence in the edited image. Lowering the value of the ϵ parameter towards 0 results in stronger semantic content (i.e., stronger presence) of the targeted attribute. As illustrated in Fig. 6, MaskFaceGAN enables a considerable level of control over the intensity of different facial attributes, such as hair color, eyebrow shape, smiling intensity and nose shape. Note how all edited images appear photo-realistic despite varying intensities of the targeted attributes.

D. Component Size Manipulation

Fig. 7 presents results that demonstrate MaskFaceGAN's ability to manipulate the size of the spatial region to be edited for a given target attribute. Specifically, the figure shows results for five targeted attributes and different scaling factors α, which defines the target attribute size with respect to the initial size of the region associated with a given attribute. Varying the scaling factor α allows MaskFaceGAN to grow or shrink certain face parts. Due to the shape term in the loss equation that assures consistent blending, we only allow growing of hair (not shrinking). Other face components, such

Fig. 2: Attributes editing examples for a sample image from the CelebA–HQ dataset. The results show manipulation of a single attribute and a comparison to the results generated with competing models. Inverted attributes are marked italic. The figure is best viewed electronically and zoomed in for details.

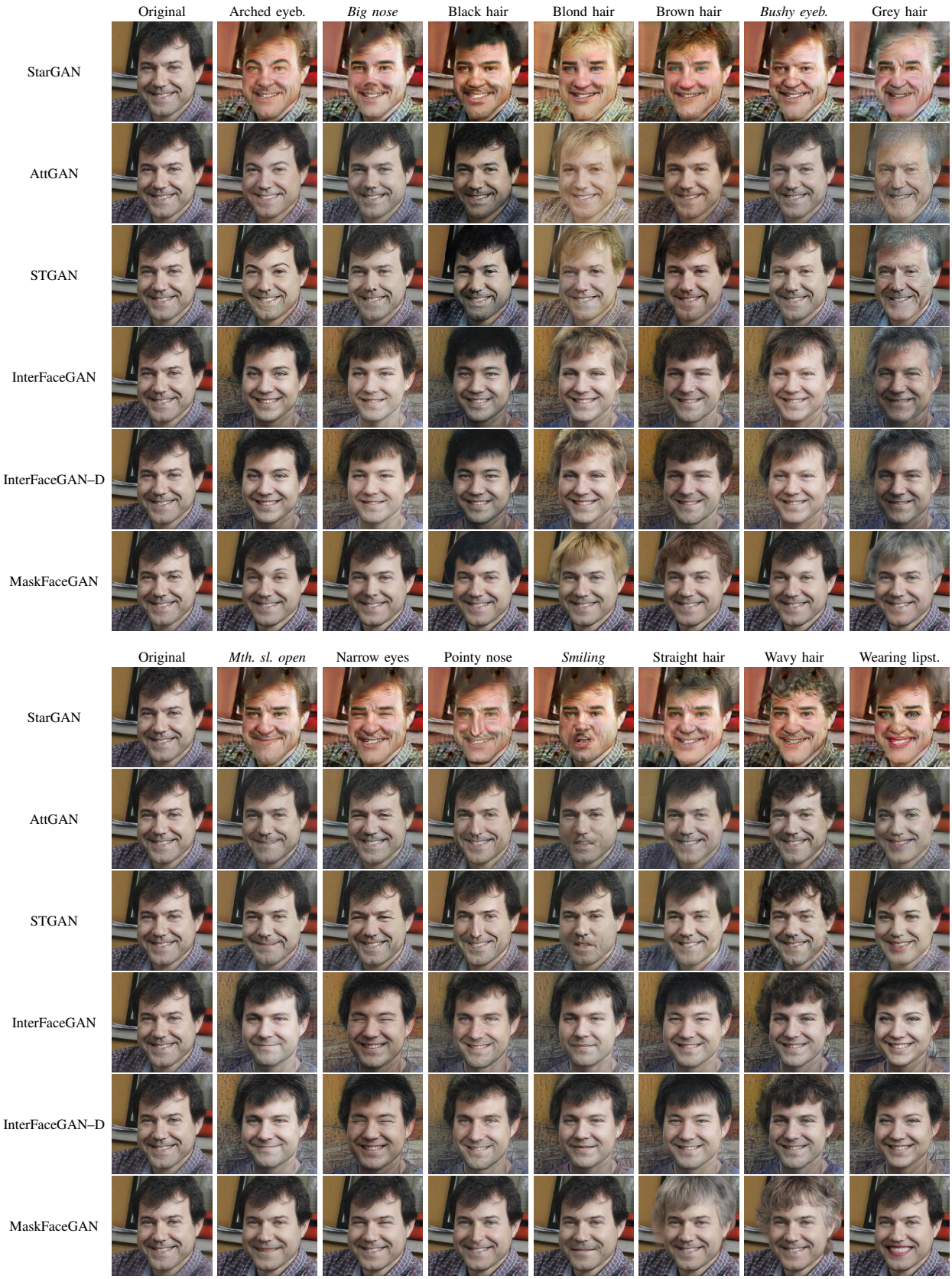


Fig. 3: Attributes editing examples for a sample image from the Helen dataset. The results show manipulation of a single attribute and a comparison to the results generated with competing models. Inverted attributes are marked italic. The figure is best viewed electronically and zoomed in for details.

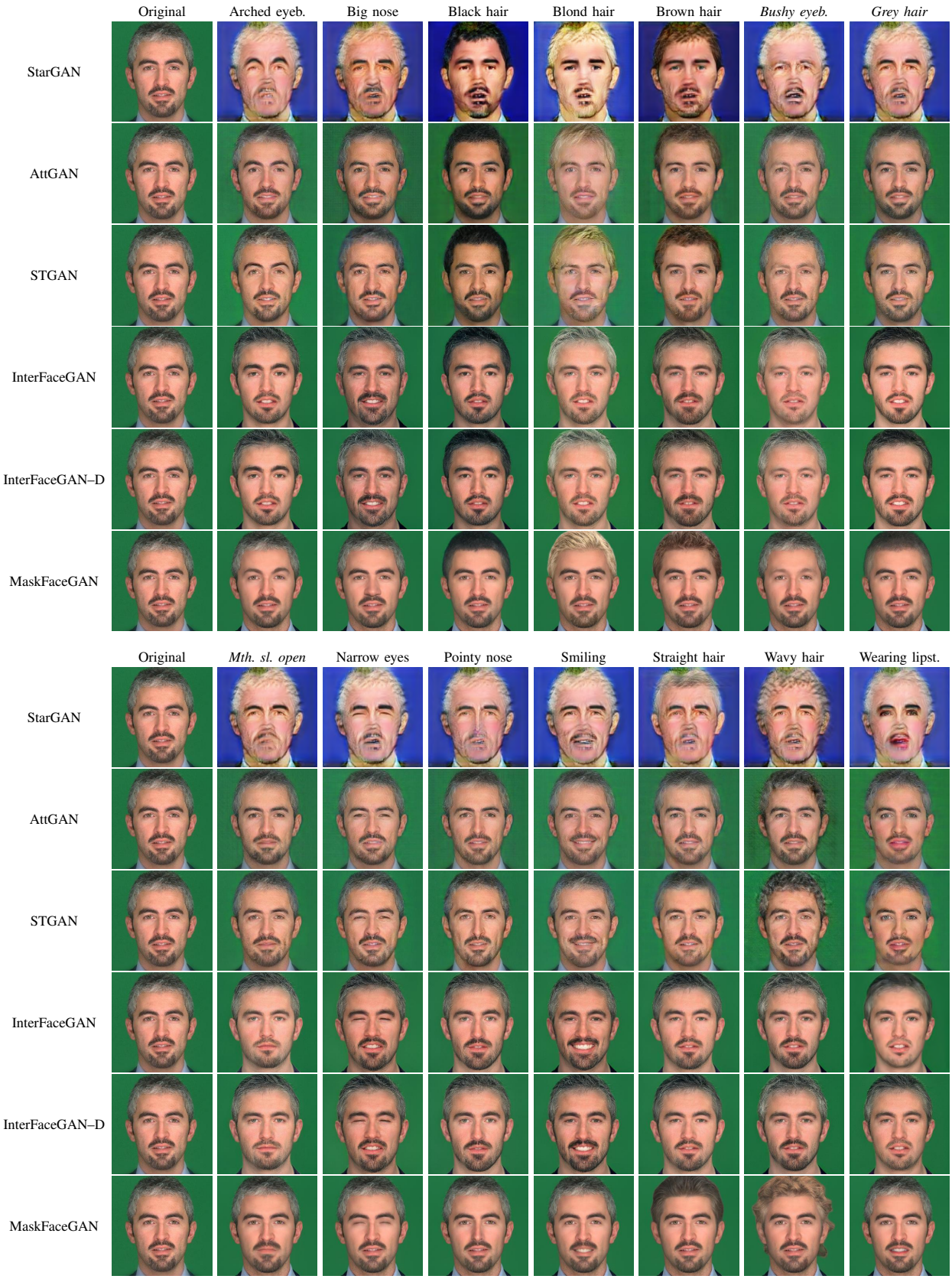


Fig. 4: Attributes editing examples for a sample image from the SiblingsDB–HQf dataset. The results show manipulation of a single attribute and a comparison to the results generated with competing models. Inverted attributes are marked italic. The figure is best viewed electronically and zoomed in for details.

Fig. 5: High–resolution comparison of MaskFaceGAN and several state-of-the-art editing techniques from the literature for the target attribute "Blond hair". Observe the quality of the generated images (the right two columns) and the details retained from the original input image presented on the left. While some of the competing models ensure solid results, MaskFaceGAN produces the most convincing image manipulations with the least amount of visible artefacts.

as the nose, eyebrows or mouth, on the other hand, can be shrunk as well. As can be seen from the presented examples, the ability of MaskFaceGAN to control the size of the spatial region during editing enables diverse image manipulations around the same targeted attribute and represents a unique feature of the proposed editing procedure.

E. Multiple Attribute Editing

Fig. 8 presents visual results when editing multiple attributes with a single optimization procedure. We show examples for jointly editing two attributes but our experiments suggest that considering more than two attributes leads to realistic

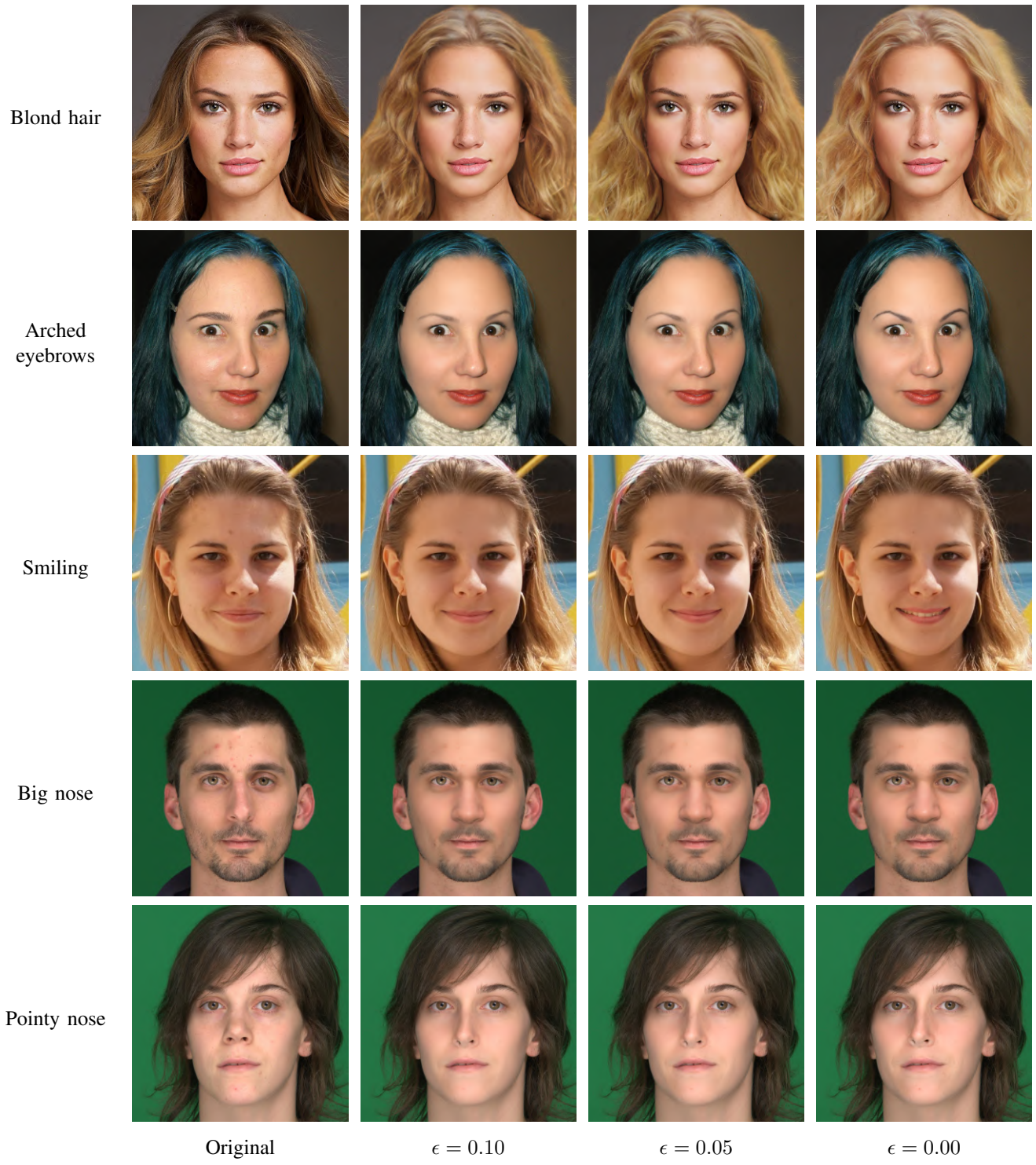


Fig. 6: Examples of intensity control for various values of the smoothing parameter ϵ. MaskFaceGAN allows for fine-grained control over attribute appearance despite being trained with binary attribute labels only. Intensity control can be applied to a wide variety of facial attributes that affect color, shape or even more complex changes of facial appearance.

results as well. As illustrated in Fig. 8, the editing procedure can also *target the same facial components/region* even if multiple attributes are edited at the same time – see the (*Wearing lipst.*, Mouth sl. open), (Bushy eyeb., Arched eyeb.), (*Smiling*, *Wearing lipst.*) and (Smiling, Mouth sl. open) results. Despite having to accomodate different target semantics in the same spatial region, MaskFaceGAN generates realistic facial appearances and produces virtually no visual artefacts.

IV. Results by Attributes

In this section we present additional quantitative results grouped by individual attributes. Specifically, we report per

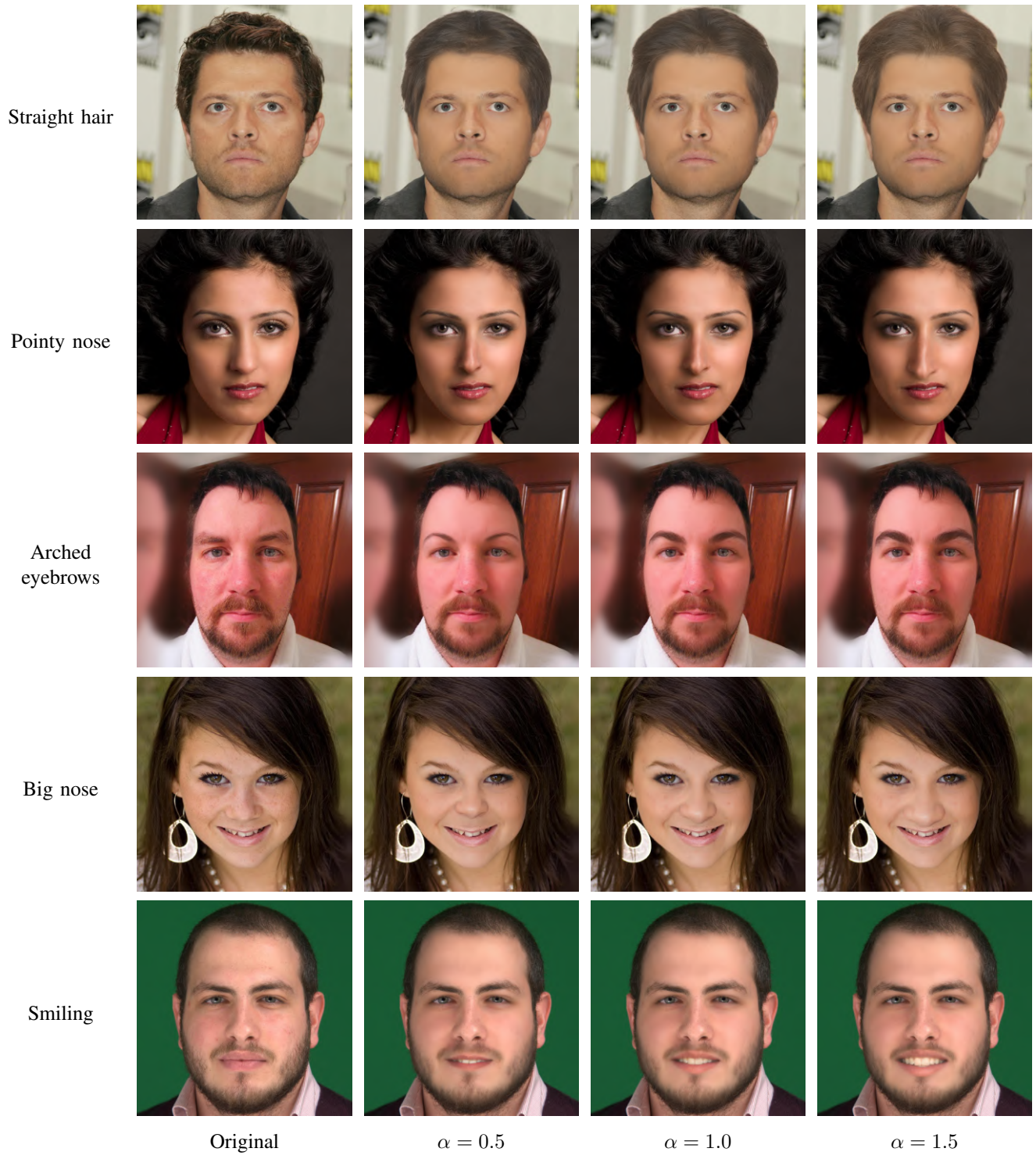


Fig. 7: Examples of component size manipulation for various scaling factors α. MaskFaceGAN allows growing the spatial region within which a targeted attribute is edited. This results in flexible image editing that can generate images with different variations of a targeted attribute in terms of size.

attribute results for the ablation study in terms of FID scores, analyze FID scores over the three experimental datasets and elaborate on per attribute user–study ratings.

A. *Ablation Study*

To ensure better insight into the contribution of individual components on the performance MaskFaceGAN, we again remove the noise optimization procedure and shape terms from our model and investigate the impact of these components on the FID scores generated per attribute on CelebA-HQ.

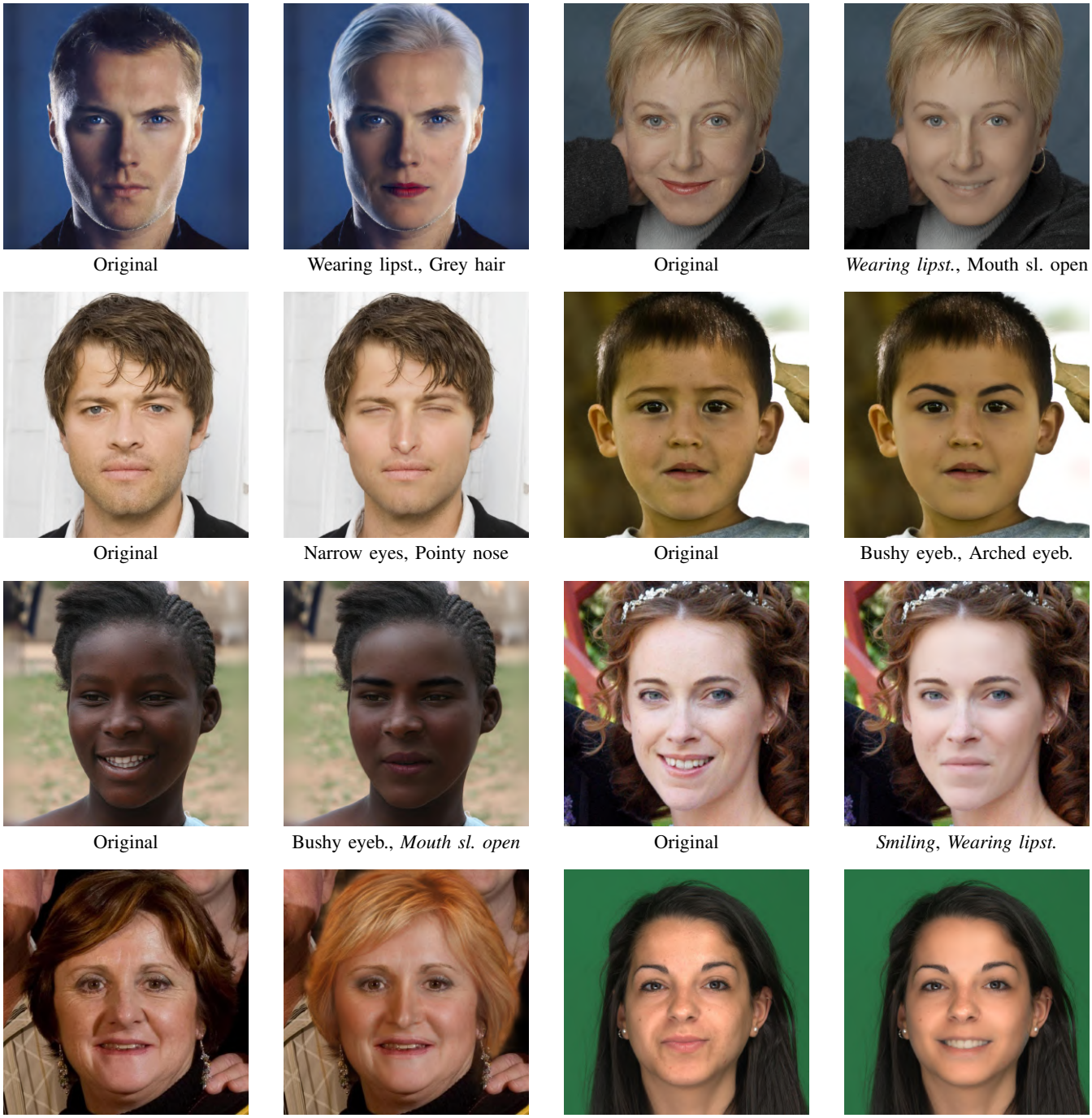


Fig. 8: Visual examples of image editing using MaskFaceGAN where multiple attributes are edited with a single optimization procedure. The presented examples show results when editing two attributes at the same time. Note that even for cases when the attributes correspond to the same spatial region, the generated results appear realistic and exhibit minimal distortions. Inverted (removed) attributes are displayed in italic.

The FID scores calculated for the ablation study are presented in Table II. We observe that the noise optimization procedure improves the FID scores of every attribute. This can be attributed to the fact that noise optimization further optimizes the quality of the embedded images and generates high frequency image details, such as hair. The inclusion of the shape term λ_S only affects attributes associated with the hair region. Most of these attributes achieve a significantly lower FID score when the shape term is included in the loss function. This can be attributed to the more natural looking results of the blending procedure, since the hair components from the generated and the original image are spatially aligned.

TABLE II: Ablation study results on CelebA–HQ. Reported are FID scores computed for each targeted attribute separately (↓ – lower is better). Note that both the noise optimization procedure as well as the shape term contribute toward higher quality results. The shape term affects only editing procedures associated with the hair region.

Target Attribute	FID scores ↓		
	$n = 0, \lambda_S = 0$	$\lambda_S = 0$	MaskFaceGAN
Arched eyebrows	49.69	24.17	24.17
Big nose	44.59	19.84	19.84
Black hair	78.77	55.27	43.39
Brown hair	72.14	50.53	37.12
Bushy eyebrows	52.67	22.80	22.80
Grey hair	81.35	47.98	37.47
Mouth sl. open	46.24	23.27	23.27
Narrow eyes	61.89	28.69	28.69
Pointy nose	47.47	23.38	23.38
Smiling	46.91	23.87	23.87
Straight hair	75.23	54.58	52.58
Wavy hair	73.00	52.00	59.83
Wearing lipstick	48.79	26.93	26.93

B. FID results

The FID scores are used as a measure of quality for the edited images. FID scores, grouped by attributes, are shown in Tables III, IV and V for CelebAHQ, Helen and Siblings dataset, respectively.

Note that MaskFaceGAN outperforms all competing models with a significant margin at all attributes except when editing the "Straight hair" and "Wavy hair" attributes. For these attributes, STGAN achieves better results than MaskFaceGAN. The single–attribute editing results shown in Figs. 2, 3 and 4 show that STGAN only slightly changes the hair shape, which preserves the quality of the initial image – but at the cost of a convincing semantic presence of the targeted attribute. MaskFaceGAN, on the other hand, generates stronger semantic content, but also hallucinates some of the image details.

C. User Study Results

Similarly as in the main part of the paper, we report user study scores for two experiments in this section: (i) in the first experiment, users were asked to rate editing results on a 5–point Likert scale, and (ii) in the second experiment, users were asked to select the best result among the edited images generated by the evaluated models.

The results for model ratings are shown in Tables VI, VII and VIII for the CelebA–HQ, Helen and Siblings datasets, respectively and the results for the best model selection ratings are shown in Tables IX, X and XI with the same dataset order. The proposed MaskFaceGAN model is consistently rated higher than other models on most of the considered attributes with the notable exception of the "Narrow eyes" attribute, where AttGAN achieves better performance across all datasets. The difference most likely stems from MaskFaceGAN's tendency to close the eyes instead of narrowing them. A similar effect can also be observed in Figs. 2, 3 and 4 for the InterFaceGAN and InterFaceGAN–D results.

V. User Study Details

The user study was performed on the Amazon Mechanical Turk platform. Access was only granted to "Master workers", that is, workers that have had good performance on other tasks at the time of the study. All images shown were of the same resolution (1024×1024) to discourage annotation based on resolution. Lower resolution images were resampled using bilinear interpolation. The question posed to worker was *"Choose the image that changes the attribute more successfully, is of higher image quality and better preserves the identity and fine details of the source image"*. The workers then had to score the editing results of each of the considered models. The images were reshuffled for every worker task to avoid cheating. The same setup was also used for the Likert scale study.

VI. Implementation Details

A. Encoder-decoder methods (StarGAN, AttGAN, STGAN)

For the encoder-decoder methods we strictly follow the implementation details specified by the source code of each model. The only hyperparameter specified is the input and output resolution of each model. For StarGAN model, the highest suggested resolution advocated by the authors for this model (256×256) resulted in considerable visual degradations of the editing results. We, therefore, trained the model with a smaller resolution (128×128) that ensured better performance. For AttGAN and STGAN model, we chose the highest advocated resolution, that is, 384×384 pixels.

B. InterFaceGAN

The InterFaceGAN method is implemented in the latent space of the StyleGAN2 model, similarly to MaskFaceGAN. We follow the procedure as outlined in [1], i.e.: first, we generate $500,000$ pairs of latent codes w and corresponding images. An attribute classifier scores the facial attribute of each image and the top $10,000$ positive and negative samples are selected for the next step. A linear SVM is then applied over the latent codes of the selected images to obtain the normal vector of the hyperplane needed for editing. The SVM regularization term is set to 1.

For InterFaceGAN, the magnitude of the movement in the latent space needs to be specified. A small magnitude barely changes the presence/absence of a facial attribute, but preserves the facial identity. A large magnitude tends to make larger changes that correspond to the desired facial attribute edit, however it usually comes at the expense of a change in identity. We experimented with several different magnitudes and chose 1 as the optimal value that provided the best trade-off between attribute editing performance and preservation of original image characteristics on our sample images.

C. InterFaceGAN–D

When implementing the disentangled version of the InterFaceGAN model, denoted as InterFaceGAN–D, one can choose multiple attributes that can be disentangled from the targeted attribute to be edited. In our experiments, we found that disentanglement works best when disentangling only a single attribute, i.e. with a simple vector projection method, as described in [1]. When choosing the attribute to disentangle for a given target attribute, we choose the attribute that displays the most entanglement issues in the generated results. For the attributes, where entanglement was not that problematic or the target attribute is hard to specify, the most correlated attribute (positive or negative correlation) from the CelebA–HQ training dataset was chosen. The disentangled attributes are presented in Table XII.

VII. Reproducibility

To foster reproducibility, all experiments presented in the main part of the paper as well as in the supplementary material are implemented with publicly available source code. For the competing models considered in the comparative experiments the official code released by the authors was used as detailed below.

- The StarGAN model was implemented with the following code: https://github.com/yunjey/stargan
- The AttGAN model was implemented with the following code: https://github.com/LynnHo/AttGAN-Tensorflow
- The STGAN model was implemented with the following code: https://github.com/csmliu/STGAN
- The InterFaceGAN(-D) model was implemented with the following code: https://github.com/genforce/interfacegan
- The StyleGAN2 model used as the generator for MaskFaceGAN was implemented with the following code: https://github.com/NVlabs/stylegan2

The source code for the MaskFaceGAN is also made publicly available via the following URL: https://github.com/MartinPernus/MaskFaceGAN.

References

[1] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of gans for semantic face editing," in *Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9243–9252.

[2] R. Abdal, Y. Qin, and P. Wonka, "Image2stylegan: How to embed images into the stylegan latent space?" in *International Conference on Computer Vision (ICCV)*, 2019, pp. 4431–4440.

[3] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 586–595.

[4] R. Abdal, Y. Qin, and P. Wonka, "Image2stylegan++: How to edit the embedded images?" in *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[5] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*, 2016, pp. 694–711.

[6] A. Dosovitskiy and T. Brox, "Generating images with perceptual similarity metrics based on deep networks," *Advances in neural information processing systems*, vol. 29, pp. 658–666, 2016.

[7] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

TABLE III: FID scores per attribute computed for the CelebA-HQ dataset – lower is better. The table shows a comparison with competing state–of–the–art models. The best score for each attribute is highlighted in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	133.49	52.22	40.34	73.01	77.77	**24.17**
Big nose	137.24	57.78	46.94	82.30	76.20	**19.84**
Black hair	143.31	77.26	57.05	80.51	77.57	**43.39**
Brown hair	140.61	58.88	46.88	72.97	75.21	**37.12**
Bushy eyebrows	125.90	50.67	38.22	78.34	74.31	**22.80**
Grey hair	172.74	139.82	98.05	98.45	78.50	**37.47**
Mouth slightly open	137.74	61.47	38.75	80.88	76.24	**23.27**
Narrow eyes	137.76	61.63	39.42	82.20	77.63	**28.69**
Pointy nose	131.32	50.51	39.15	73.41	73.87	**23.38**
Smiling	141.69	61.63	42.83	78.88	75.41	**23.87**
Straight hair	137.27	60.57	**43.62**	73.65	76.94	52.58
Wavy hair	143.40	67.06	**48.65**	73.62	75.24	59.83
Wearing lipstick	148.91	78.70	63.99	89.00	84.45	**26.93**

TABLE IV: FID scores per attribute computed for the Helen dataset – lower is better. The table shows a comparison with competing state–of–the–art models. The best score for each attribute is highlighted in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	148.26	68.22	47.38	87.90	91.24	**21.65**
Big nose	145.10	70.57	56.79	94.05	90.71	**19.19**
Black hair	151.61	85.47	60.52	92.19	92.26	**42.74**
Brown hair	149.86	70.38	52.14	87.42	89.12	**37.88**
Bushy eyebrows	145.54	66.99	44.20	89.00	91.61	**21.29**
Grey hair	180.83	140.68	102.65	102.65	91.60	**43.04**
Mouth slightly open	150.45	72.40	46.06	88.16	89.00	**21.87**
Narrow eyes	154.31	81.84	50.00	90.56	93.07	**23.67**
Pointy nose	147.12	66.93	49.04	84.23	89.09	**20.98**
Smiling	150.66	74.05	48.94	90.17	88.20	**22.47**
Straight hair	148.26	78.32	**55.92**	88.24	92.06	66.85
Wavy hair	161.97	81.34	**65.33**	89.95	90.39	76.04
Wearing lipstick	150.33	81.43	67.74	97.29	93.76	**24.33**

TABLE V: FID scores per attribute computed for the SiblingsDB-HQf dataset – lower is better. The table shows a comparison with competing state–of–the–art models. The best score for each attribute is highlighted in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	181.24	66.69	38.29	75.42	81.55	**29.95**
Big nose	166.84	68.64	43.69	85.97	80.07	**16.93**
Black hair	176.82	87.83	49.60	85.91	83.24	**37.45**
Brown hair	175.56	64.17	37.08	78.72	81.90	**31.92**
Bushy eyebrows	171.77	62.65	32.34	81.63	82.24	**28.43**
Grey hair	199.54	141.63	106.99	90.54	83.61	**36.37**
Mouth slightly open	173.15	76.05	36.66	82.04	86.61	**23.80**
Narrow eyes	182.95	94.33	39.37	84.46	87.40	**35.49**
Pointy nose	169.58	67.07	36.67	77.21	81.77	**24.55**
Smiling	175.40	82.80	43.63	75.63	81.38	**22.81**
Straight hair	172.94	76.65	**49.29**	74.36	81.75	59.24
Wavy hair	184.82	84.39	**62.69**	77.47	84.81	70.06
Wearing lipstick	168.28	84.55	59.19	91.06	87.84	**31.84**

TABLE VI: Results of the user study on test images from the CelebA–HQ dataset. Users were asked to rate the models on a 5–point Likert scale, where 5 stands for a perfect result. The model with the best user rating for each attribute is presented in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	1.44 ± 0.94	2.75 ± 0.96	3.00 ± 1.02	3.05 ± 1.03	3.38 ± 1.19	$\mathbf{4.03 \pm 1.24}$
Big nose	1.37 ± 0.84	3.09 ± 1.05	2.87 ± 0.96	2.72 ± 0.99	2.60 ± 0.99	$\mathbf{4.35 \pm 1.14}$
Black hair	1.68 ± 1.02	2.74 ± 1.23	3.02 ± 1.12	3.16 ± 0.81	3.50 ± 1.06	$\mathbf{4.00 \pm 1.23}$
Blond hair	1.74 ± 1.15	2.81 ± 1.15	2.97 ± 1.06	2.98 ± 1.02	3.24 ± 1.06	$\mathbf{4.12 \pm 1.12}$
Brown hair	1.39 ± 0.82	2.73 ± 1.10	3.00 ± 0.98	3.13 ± 0.95	3.37 ± 0.97	$\mathbf{4.41 \pm 0.86}$
Bushy eyebrows	1.63 ± 1.21	2.61 ± 1.12	3.18 ± 0.97	3.17 ± 0.95	3.53 ± 1.03	$\mathbf{4.42 \pm 1.17}$
Grey hair	1.41 ± 0.84	3.15 ± 1.15	2.58 ± 1.11	2.10 ± 1.15	2.07 ± 1.18	$\mathbf{3.91 \pm 1.22}$
Mouth slightly open	1.27 ± 0.57	2.97 ± 1.21	2.88 ± 0.93	2.44 ± 1.10	2.68 ± 1.30	$\mathbf{4.11 \pm 1.28}$
Narrow eyes	1.38 ± 0.86	$\mathbf{3.60 \pm 1.27}$	3.00 ± 1.08	2.52 ± 0.98	2.62 ± 1.11	3.52 ± 1.46
Pointy nose	1.63 ± 1.09	3.00 ± 1.09	3.00 ± 0.96	3.14 ± 1.05	3.41 ± 1.15	$\mathbf{4.34 \pm 1.01}$
Smiling	1.36 ± 0.89	2.84 ± 1.20	3.17 ± 1.08	2.66 ± 1.12	2.80 ± 1.31	$\mathbf{4.11 \pm 1.18}$
Straight hair	1.31 ± 0.81	3.05 ± 1.04	3.36 ± 1.02	3.12 ± 0.94	3.64 ± 1.03	$\mathbf{3.80 \pm 1.28}$
Wavy hair	1.52 ± 0.97	2.71 ± 0.93	3.13 ± 1.01	3.05 ± 1.18	3.31 ± 1.27	$\mathbf{3.74 \pm 1.32}$
Wearing lipstick	1.27 ± 0.72	3.17 ± 1.13	2.89 ± 1.16	2.67 ± 0.85	2.80 ± 1.08	$\mathbf{4.04 \pm 1.35}$

TABLE VII: Results of the user study on test images from the Helen dataset. Users were asked to rate the models on a 5–point Likert scale, where 5 stands for a perfect result. The model with the best user rating for each attribute is presented in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	1.46 ± 0.88	2.79 ± 1.18	2.95 ± 1.06	2.53 ± 0.94	2.72 ± 1.09	$\mathbf{4.10 \pm 1.15}$
Big nose	1.25 ± 0.58	2.92 ± 1.15	3.07 ± 1.07	2.16 ± 0.88	2.42 ± 1.00	$\mathbf{4.39 \pm 0.89}$
Black hair	1.44 ± 0.76	2.54 ± 1.19	3.24 ± 1.19	2.79 ± 0.91	2.89 ± 1.12	$\mathbf{3.64 \pm 1.28}$
Blond hair	1.30 ± 0.55	2.73 ± 1.18	2.92 ± 1.14	2.88 ± 1.11	2.71 ± 1.05	$\mathbf{3.66 \pm 1.14}$
Brown hair	1.36 ± 0.65	2.70 ± 1.22	2.59 ± 1.08	3.00 ± 1.07	2.79 ± 1.11	$\mathbf{4.05 \pm 1.18}$
Bushy eyebrows	1.37 ± 0.72	2.35 ± 1.23	3.13 ± 1.21	2.40 ± 1.02	2.67 ± 1.14	$\mathbf{3.93 \pm 1.21}$
Grey hair	1.34 ± 0.65	2.74 ± 1.21	2.77 ± 1.20	1.94 ± 1.02	2.07 ± 1.06	$\mathbf{3.65 \pm 1.34}$
Mouth slightly open	1.24 ± 0.58	2.78 ± 1.32	3.14 ± 1.22	1.98 ± 0.88	2.12 ± 1.07	$\mathbf{3.73 \pm 1.51}$
Narrow eyes	1.32 ± 0.61	$\mathbf{3.18 \pm 1.43}$	2.98 ± 1.34	1.84 ± 0.92	1.65 ± 0.77	2.99 ± 1.38
Pointy nose	1.16 ± 0.52	3.02 ± 1.11	3.16 ± 1.04	2.56 ± 0.98	2.70 ± 0.95	$\mathbf{4.30 \pm 0.96}$
Smiling	1.15 ± 0.41	2.82 ± 1.49	$\mathbf{3.55 \pm 1.33}$	2.15 ± 1.03	2.14 ± 1.11	3.43 ± 1.33
Straight hair	1.42 ± 0.71	3.14 ± 1.01	$\mathbf{3.36 \pm 1.22}$	2.39 ± 1.01	2.62 ± 1.19	3.23 ± 1.32
Wavy hair	1.27 ± 0.63	2.64 ± 1.06	2.92 ± 1.13	2.52 ± 1.17	2.28 ± 1.16	$\mathbf{4.05 \pm 1.14}$
Wearing lipstick	1.19 ± 0.48	2.62 ± 1.21	2.65 ± 1.21	2.34 ± 0.98	2.37 ± 1.08	$\mathbf{4.08 \pm 1.20}$

TABLE VIII: Results of the user study on test images from the SiblingsDB–HQf dataset. Users were asked to rate the models on a 5–point Likert scale, where 5 stands for a perfect result. The model with the best user rating for each attribute is presented in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	1.48 ± 0.78	3.00 ± 1.15	3.27 ± 1.01	2.43 ± 0.90	2.64 ± 0.96	$\mathbf{4.14 \pm 1.12}$
Big nose	1.51 ± 0.87	3.15 ± 1.13	3.00 ± 0.86	2.29 ± 0.81	2.50 ± 0.88	$\mathbf{4.27 \pm 1.03}$
Black hair	1.55 ± 0.88	2.99 ± 1.10	$\mathbf{3.64 \pm 1.08}$	2.75 ± 0.87	3.15 ± 1.03	3.60 ± 1.12
Blond hair	1.62 ± 0.88	3.10 ± 1.15	3.20 ± 0.97	2.71 ± 0.86	2.97 ± 0.87	$\mathbf{3.82 \pm 1.15}$
Brown hair	1.67 ± 1.02	3.13 ± 1.12	3.14 ± 0.94	2.94 ± 0.94	3.04 ± 1.04	$\mathbf{3.62 \pm 1.10}$
Bushy eyebrows	1.42 ± 0.73	2.93 ± 1.08	3.27 ± 1.02	2.57 ± 0.98	2.97 ± 1.06	$\mathbf{3.92 \pm 1.23}$
Grey hair	1.72 ± 1.10	3.05 ± 1.11	2.99 ± 0.98	2.30 ± 0.96	2.40 ± 1.03	$\mathbf{3.55 \pm 1.30}$
Mouth slightly open	1.47 ± 0.88	3.11 ± 1.15	3.38 ± 0.95	2.08 ± 0.86	2.14 ± 0.93	$\mathbf{4.15 \pm 1.18}$
Narrow eyes	1.50 ± 0.89	$\mathbf{3.39 \pm 1.23}$	3.23 ± 1.19	2.20 ± 0.97	2.20 ± 0.95	3.20 ± 1.29
Pointy nose	1.43 ± 0.79	3.20 ± 1.11	3.32 ± 0.98	2.61 ± 0.84	2.91 ± 1.04	$\mathbf{4.20 \pm 1.03}$
Smiling	1.38 ± 0.85	3.04 ± 1.04	3.84 ± 0.97	2.36 ± 0.81	2.46 ± 0.84	$\mathbf{4.11 \pm 0.99}$
Straight hair	1.51 ± 0.85	3.27 ± 0.98	3.34 ± 1.07	2.67 ± 0.90	2.77 ± 0.96	$\mathbf{3.47 \pm 1.22}$
Wavy hair	1.53 ± 0.93	3.19 ± 1.10	3.35 ± 1.13	2.50 ± 0.93	2.56 ± 1.08	$\mathbf{3.70 \pm 1.21}$
Wearing lipstick	1.53 ± 1.00	3.15 ± 0.96	3.17 ± 1.02	2.33 ± 0.89	2.54 ± 1.07	$\mathbf{4.16 \pm 1.12}$

TABLE IX: Results of the user study on test images from the CelebA–HQ dataset in terms of percentage of selected images. Users were asked to select the best editing results when presented with image examples generated by all evaluated models. The model with the best user rating for each attribute is presented in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	1.43%	4.29%	7.14%	7.14%	24.29%	**55.71%**
Big nose	8.82%	10.29%	1.47%	2.94%	1.47%	**75.00%**
Black hair	6.15%	3.08%	7.69%	3.08%	26.15%	**53.85%**
Blond hair	5.71%	4.29%	8.57%	7.14%	18.57%	**55.71%**
Brown hair	2.94%	1.47%	2.94%	4.41%	16.18%	**72.06%**
Bushy eyebrows	3.12%	3.12%	4.69%	1.56%	9.38%	**78.12%**
Grey hair	3.08%	12.31%	10.77%	6.15%	10.77%	**56.92%**
Mouth slightly open	0.00%	11.43%	4.29%	2.86%	12.86%	**68.57%**
Narrow eyes	0.00%	**42.65%**	10.29%	1.47%	7.35%	38.24%
Pointy nose	1.45%	11.59%	2.90%	1.45%	13.04%	**69.57%**
Smiling	5.80%	8.70%	13.04%	2.90%	5.80%	**63.77%**
Straight hair	1.47%	8.82%	11.76%	14.71%	20.59%	**42.65%**
Wavy hair	1.52%	6.06%	6.06%	9.09%	19.70%	**57.58%**
Wearing lipstick	0.00%	11.76%	13.24%	4.41%	8.82%	**61.76%**

TABLE X: Results of the user study on test images from the Helen dataset in terms of percentage of selected images. Users were asked to select the best editing results when presented with image examples generated by all evaluated models. The model with the best user rating for each attribute is presented in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	3.74%	14.95%	9.35%	5.61%	9.35%	**57.01%**
Big nose	7.62%	13.33%	11.43%	5.71%	5.71%	**56.19%**
Black hair	3.74%	11.21%	19.63%	9.35%	15.89%	**40.19%**
Blond hair	0.93%	10.28%	16.82%	16.82%	14.02%	**41.12%**
Brown hair	7.55%	9.43%	7.55%	8.49%	6.60%	**60.38%**
Bushy eyebrows	0.00%	11.43%	20.00%	6.67%	9.52%	**52.38%**
Grey hair	9.43%	6.60%	19.81%	5.66%	6.60%	**51.89%**
Mouth slightly open	0.96%	17.31%	22.12%	2.88%	6.73%	**50.00%**
Narrow eyes	0.00%	**47.17%**	22.64%	4.72%	1.89%	23.58%
Pointy nose	1.89%	7.55%	9.43%	7.55%	8.49%	**65.09%**
Smiling	0.00%	25.47%	**45.28%**	0.94%	6.60%	21.70%
Straight hair	0.93%	16.82%	**32.71%**	7.48%	12.15%	29.91%
Wavy hair	0.00%	3.85%	11.54%	6.73%	9.62%	**68.27%**
Wearing lipstick	1.90%	10.48%	12.38%	2.86%	8.57%	**63.81%**

TABLE XI: Results of the user study on test images from the SiblingsDB–HQf dataset in terms of percentage of selected images. Users were asked to select the best editing results when presented with image examples generated by all evaluated models. The model with the best user rating for each attribute is presented in bold.

Attribute	StarGAN	AttGAN	STGAN	InterFaceGAN	InterFaceGAN–D	MaskFaceGAN
Arched eyebrows	2.11%	10.56%	14.79%	4.23%	8.45%	**59.86%**
Big nose	7.04%	4.23%	6.34%	6.34%	5.63%	**70.42%**
Black hair	2.80%	10.49%	**32.17%**	4.90%	31.47%	18.18%
Blond hair	4.26%	5.67%	12.77%	11.35%	15.60%	**50.35%**
Brown hair	11.43%	14.29%	5.71%	8.57%	17.14%	**42.86%**
Bushy eyebrows	2.13%	10.64%	15.60%	7.09%	16.31%	**48.23%**
Grey hair	5.59%	11.89%	8.39%	10.49%	13.99%	**49.65%**
Mouth slightly open	1.41%	7.04%	15.49%	4.93%	4.93%	**66.20%**
Narrow eyes	4.32%	30.22%	**33.81%**	5.76%	1.44%	24.46%
Pointy nose	3.57%	10.71%	6.43%	5.71%	23.57%	**50.00%**
Smiling	2.84%	3.55%	**63.12%**	1.42%	0.71%	28.37%
Straight hair	2.86%	20.00%	25.00%	7.14%	9.29%	**35.71%**
Wavy hair	3.55%	5.67%	13.48%	3.55%	13.48%	**60.28%**
Wearing lipstick	5.76%	6.47%	10.07%	7.91%	14.39%	**55.40%**

TABLE XII: Implementation details for the InterFaceGAN–D model used in the experiments. The table shows which attribute was disentangled for a given target attribute. The target attributes presented in italic font denote the attributes, for which the disentangled attribute was selected based on the maximum absolute correlation, computed over the CelebA training dataset.

Target attribute	**Disentangled attribute**
Arched eyebrows	Male
Big nose	Chubby
Black hair	Pale skin
Blond hair	Pale skin
Brown hair	Pale skin
Bushy eyebrows	Mustache
Grey hair	Young
Mouth slightly open	Smiling
Narrow eyes	Bald
Pointy nose	Rosy cheeks
Smiling	Mouth slightly open
Straight hair	Bald
Wavy hair	Heavy makeup
Wearing lipstick	Male