MaskFaceGAN: High-Resolution Face Editing With Masked GAN Latent Code Optimization

Martin Pernus, Vitomir Štruc, Senior Member, IEEE, and Simon Dobrišek, Senior Member, IEEE

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 FRGC, SiblingsDB-HQf, and XM2VTS 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 networks, 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 photorealistic 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 [5], [7], [8], [11], [12]. 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 on 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. 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. Mask-
FaceGAN 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., $1024 \times 1024$ pixels.
- We propose an efficient optimization procedure for esti-
mating 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 process-
ing 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 progres-
sive 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 interpo-
lation 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 uncon-
ditional 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 mod-
els 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

![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 ($1024 \times 1024$) examples show editing results for four target attributes. Best viewed zoomed-in.](image)
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. 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 literatures, e.g., [1], [2], [42], and [43]. The work in [6] and [44], for example, explored the use of user-supplied sketches to drive the editing procedure. The authors of [11] and [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] and [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.,

\[
\psi_a : I \mapsto I' \in \mathbb{R}^{3 \times m \times n},
\]

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 \( \psi_a \) through the process of GAN inversion [13], [14]. With these techniques the input image \( I \) is first embedded in the latent space of a pretrained 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 \( S_{\text{skin}} \) that should be preserved during editing and predict image areas \( S_{\text{attr}} \) that need to be edited given \( a \). See Fig. 2 for an example using the target attribute “Black hair".
imposing spatial constraints.

the targeted semantics, and (\(G\)) 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.

- **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 \((\mathbf{w})\) of StyleGAN2 that encodes high-frequency image details. The model, hence, outputs generated images \((I_G)\) based on the following mapping: \(I_G = G(\mathbf{w}, n)\).

- **Step 3: Blending.** Finally, a blending step is utilized to combine the (intermediate) image \(I_G\), generated from the optimized latent code, 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 \(W^+\) 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 \(\mathbf{w}_i\), one for each layer of the model, i.e., \(\mathbf{w} = [\mathbf{w}_i]_{i=1}^{n_c}\). To ensure photo-realism StyleGAN2 additionally uses \(N = 17\) stochastic (i.e., Gaussian noise) channels \(n = [\mathbf{n}_i]_{i=1}^{N}\) of different spatial resolutions (ranging from \(4 \times 4\) to \(1024 \times 1024\)) that encode high-frequency image details. The model, hence, generates output images \(I_G\) based on the following mapping: \(I_G = G(\mathbf{w}, n)\).

\(1\) The extended latent space \(W^+\) 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. 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 \((S_{\text{skin}})\) and regions that should be altered \((S_{\text{tar}})\). 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.

- **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 \(I_G = G(\mathbf{w}, n)\), each classification head \(C_k\) predicts the probability of a facial attribute being present in \(I_G\), i.e., \(c_k = C_k(I_G')\) 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.: \(S : \mathbb{R}^{3 \times n \times m} \rightarrow [0, 1]^{m \times n \times m}\), where \(L\) is the number of parsed categories (face components). For MaskFaceGAN, two principal channels are used. The first one is the skin region, \(S_{\text{skin}} \in [0, 1]^{m \times n \times m}\), 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, \(S_{\text{tar}}(I) \in [0, 1]^{m \times n \times m}\).

**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. We focus

---

**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.
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).

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

1) 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):

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

where \(S_{\text{skin}}(I)\) 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 \(S_{\text{skin}}\).

2) 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) [49], i.e.:

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

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 \(\epsilon\) can be used to set the intensity of the desired attribute, e.g., the intensity of lipstick presence when editing the “Wearing lipstick” attribute.

3) 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:

\[
\mathcal{L}_S = \|S_{\text{tar}}(I) - S_{\text{tar}}(I_G)\|_2^2,
\]

where \(S_{\text{tar}}\) 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.

4) 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 \(s_{\text{tar}} \in [0, 1]\) covered by a given target component be defined as:

\[
s_{\text{tar}}(I) = \sum_{k, y} S_{\text{tar}}(I) \frac{1}{|S_{\text{tar}}(I)|},
\]

where the operator \(|S_{\text{tar}}(I)|\) denotes the number of pixels in \(S_{\text{tar}}\). To be able to scale the size of the targeted facial

---

**TABLE I**

| 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 |

---

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 \(S_{\text{tar}}(I)\) 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.
region we introduce a scaling factor $\alpha$ 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 $s_{\text{tar}}(I)$ and the desired portion $s_{\text{tar}}(I_G)$ in the generated image $I_G$, i.e.:

$$L_p = D_{KL}(s_{\text{tar}}(I_G), \alpha s_{\text{tar}}(I)) .$$

(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.

5) 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 $S_{\text{tar}}(I_G)$. Next, we subtract this region from $S_{\text{skin}}(I)$ 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 $S_{\text{tar}}(I')$ 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 $L_M$ and target–shape $L_S$ constraint used by MaskFaceGAN are, hence, defined as:

$$L_M = || \min(S_{\text{skin}}(I) - S_{\text{tar}}(I_G), 0) \odot (I_G - I) ||_2^2,$$

(7)

$$L_S = || \max(S_{\text{tar}}(I) + S_{\text{tar}}(I_G), 1) - S_{\text{tar}}(I_G) ||_2^2,$$

(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.

6) Final Objective: The overall optimization objective ($L_w$) of MaskFaceGAN is defined as a linear combination of the objectives/constraints described above, i.e.:

$$L_w = \lambda_M L_M + \lambda_C L_C + \lambda_S L_S + \lambda_P L_P ,$$

(9)

where $\lambda_M$, $\lambda_C$, $\lambda_S$ and $\lambda_P$ are weighting factors that control the contribution of the individual objectives. Minimizing $L_w$ 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 the noise component $n$ is optimized to allow region growth. 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 $S_{\text{tar}}(I_G)$. Next, we subtract this region from $S_{\text{skin}}(I)$ 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 $S_{\text{tar}}(I')$ 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 $L_M$ and target–shape $L_S$ constraint used by MaskFaceGAN are, hence, defined as:

$$L_M = || \min(S_{\text{skin}}(I) - S_{\text{tar}}(I_G), 0) \odot (I_G - I) ||_2^2,$$

(7)

$$L_S = || \max(S_{\text{tar}}(I) + S_{\text{tar}}(I_G), 1) - S_{\text{tar}}(I_G) ||_2^2,$$

(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.

6) Final Objective: The overall optimization objective ($L_w$) of MaskFaceGAN is defined as a linear combination of the objectives/constraints described above, i.e.:

$$L_w = \lambda_M L_M + \lambda_C L_C + \lambda_S L_S + \lambda_P L_P ,$$

(9)

where $\lambda_M$, $\lambda_C$, $\lambda_S$ and $\lambda_P$ are weighting factors that control the contribution of the individual objectives. Minimizing $L_w$ 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 $I_G$.

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].

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

$$L_n = \lambda_M L_M + \lambda_R \sum_{i,j} L_{i,j},$$

(10)

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

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

(11)

The goal of this regularization term is to ensure that the noise components follow 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 is downsamplied to a resolution of 8 by averaging 2 by $2\times 2$ 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.
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, C, S | 70,000          | n/a         | Age, ethnicity, background |
| FRGC [50]          | 1704 × 2272      | Testing          | n/a              | 200         | Age, ethnicity, gender |
| XM2VTS [50]        | 720 × 576        | Testing          | n/a              | 200         | Age, gender          |
| SiblingsDB–HQf [51]| 4256 × 2832      | Testing          | n/a              | 200         | Age, gender          |

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

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_{hair}(I) for most attributes and B = S_{skin}(I) + S_{hair}(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. \quad (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
Four high-resolution images datasets are used in the experiment with MaskFaceGAN, i.e., Flickr-Faces-HQ (FFHQ) [17], FRGC [52], XM2VTS [53], and SiblingsDB–HQf [51]. The training and testing datasets were carefully selected based on their image licences and the presence of individual consent towards the use of personal face images towards research purposes, as well as technical criteria, such as dataset size, image resolution, image quality, and available annotation. 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. The faces contain considerable variation in terms of age, ethnicity and image background. The images come with Creative Commons licence and were made available from the authors of StyleGAN2. FFHQ is used as the primary training dataset, used to train the generator model (G), the attribute model (C), and the segmentation model (S). We generate binary attribute annotations by following MAAD-Face distillation protocol [54].

- **Face Recognition Grand Challenge (FRGC)** [52] consists of high resolution images and 3D scans. The imagery was shot under both controlled and unstructured illumination. The image sizes are either 1704 × 2272 or 1200 × 1600 pixels.

- **SiblingsDB-HQf** [51] 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 lighting.

- **XM2VTSDB (XM2VTS)** [53] consists of digital video recordings of 295 subjects, taken at one month intervals over a period of five months. The images were taken under controlled lighting conditions with uniform background.

We select 200 high-quality images from each of the testing dataset, i.e. FRGC, SiblingsDB-HQf, and XM2VTS, for the experiments. The images are then processed with a standard face-processing pipeline [27] that involves cropping the face region and resampling to 1024×1024 pixels. This resolution is used in all MaskFaceGAN editing experiments. 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.

- **The Attribute Classifier** (C) is implemented based on the model from [47] and trained on the attribute-annotated FFHQ dataset, which was split into a training, validation, and test split. The training is done for 26 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 [55] 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 based on DeepLabv3 [48]. It is trained based on the DatasetGAN [56] protocol for generating synthetic face-parsing ground truth. We use the ground truth provided by DatasetGAN for the facial parsing task that includes 34 facial categories. These categories are merged to produce the following 9 facial regions/classes: “hair”, “skin”, “eyebrows”, “nose”, “eyes”, “mouth”, “neck & clothes”, “earrings”, and “ears”. We generate 100,000 synthetic facial images and their corresponding ground truth segmentation masks.
Similarly to [16], the noise component terms are added to enforce semantic and spatial constraints. 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 [57] 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] and [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 \( 128 \times 128 \) pixels, AttGAN and STGAN generate \( 384 \times 384 \) images, while InterFaceGAN, InterFaceGAN–D and MaskFaceGAN edit images at a resolution of \( 1024 \times 1024 \) 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.

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

1) 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 FRGC and XM2VTS. 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. AttGAN and STGAN produce more convincing results, but still induce a certain amount of visual artefacts. These can, for example, be seen with the “Blond hair” attribute in the Fig. 6. The artefacts generated by the encoder-decoder methods stem from difficulties in balancing multiple loss terms commonly used when training such methods.

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.

2) Quantitative Evaluation: To evaluate attribute editing performance in a quantitative manner, prior works [7] and [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
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 FRGC and results at the bottom to an image from XM2VTS. Best viewed zoom in.

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) [58] 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 $299 \times 299$ 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

| Method             | FRGC | Siblings/DB-HQf | XM2VTS |
|--------------------|------|-----------------|--------|
| StarGAN [5]        | 189.82 | 209.38 | 160.80 |
| ArtGAN [7]         | 53.63  | 47.27   | 40.47  |
| STGAN [8]          | 43.12  | 44.24   | 29.81  |
| InterFaceGAN [9], [14] | 41.35 | 46.48  | 44.48  |
| InterFaceGAN-D [9], [14] | 39.49 | 45.39  | 40.34  |
| MaskFaceGAN (ours) | **19.96** | **17.38** | **20.81** |
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.

| Method              | FRGC | SiblingsDB–HQf | XM2VTS |
|---------------------|------|----------------|--------|
| StarGAN [5]         | 2.97 % | 4.16 % | 0.57 % |
| AttGAN [7]          | 6.27 % | 7.01 % | 8.24 % |
| STGAN [8]           | 14.69 % | 9.93 % | 7.57 % |
| InterFaceGAN [9], [14] | 9.24 % | 21.09 % | 22.32 % |
| InterFaceGAN–D [9], [14] | 11.22 % | 12.04 % | 13.31 % |
| MaskFaceGAN (ours)  | **55.61 %** | **45.77 %** | **47.99 %** |

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).

Table V

User Study Results, Where Human Raters Were Asked to Rate the Quality of theEdited Images on a 5–Point Likert Scale (Higher Is Better). Reported Is the Average Score and Corresponding Standard Deviation.

| Method              | FRGC       | SiblingsDB–HQf | XM2VTS    |
|---------------------|------------|----------------|-----------|
| StarGAN [5]         | 1.43 ± 0.89 | 1.52 ± 0.91 | 1.31 ± 0.65 |
| AttGAN [7]          | 2.82 ± 1.04 | 2.46 ± 0.92 | 2.42 ± 1.07 |
| STGAN [8]           | 3.01 ± 1.21 | 2.65 ± 1.02 | 2.47 ± 1.12 |
| InterFaceGAN [9], [14] | 3.00 ± 1.05 | 3.26 ± 1.04 | 3.07 ± 1.20 |
| InterFaceGAN–D [9], [14] | 2.97 ± 1.15 | 3.11 ± 1.09 | 2.78 ± 1.23 |
| MaskFaceGAN (ours)  | **4.20 ± 1.31** | **4.01 ± 1.22** | **3.94 ± 1.32** |

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

of the edited images and lack of artefacts, which are the 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.

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.

1) 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., $K \in \{1, 2, 3\}$. 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.
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.

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.

2) 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 \( \epsilon \) affects the strength (or intensity) of the targeted attribute in the edited images. A few illustrative examples of the impact of \( \epsilon \) 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. The results depend on the trained classifier and what it considers an attribute presence with \( 1 - \epsilon \) probability.

3) 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.

4) 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 FRGC. 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.

1) 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.

2) Quantitative Analysis: For a quantitative analysis of the ablation results, we report in Table VI mean FID scores computed over the test images of FRGC 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.

### Table VI

| MaskFaceGAN variant                             | FID  |
|------------------------------------------------|------|
| Local latent code optimization, no shape term  | 58.85|
| + noise optimization                           | 22.78|
| + shape term (complete MaskFaceGAN)            | 19.96|

D. Component Analysis

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

1) 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 [59], i.e., VGG16 and VGG19, and three ResNet-based [60] 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 FFHQ 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 FFHQ 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 93%, 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 classifiers 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.

2) Face Parser: MaskFaceGAN uses DeepLabv3 [48] to implement the face parser ($F$). In this section, we analyze editing results produced with 3 other parsers. We select the following competing models for the comparison: UNet [61] and FCN [62] with two different backbones, i.e., FCN–ResNet50 and FCN–ResNet101, and train them on FFHQ using the same procedure/protocol and optimization method as with DeepLabv3. The performance in terms of the average Intersection over Union (IoU) [63] 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 80% to up to 83%.
TABLE VIII
AVERAGE INTERSECTION OVER UNION (IoU) [IN %] OVER THE FFHQ TEST DATA FOR THE SELECTED FACE PARSERS

| Model          | UNet | FCN-ResNet50 | FCN-ResNet101 | DeepLabv3 (ours) |
|----------------|------|--------------|---------------|-----------------|
| IoU            | 80.69% | 81.95%       | 80.95%        | 83.19%          |

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.

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.

1) 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.

2) 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 $B_1 = S_{tar}(I)$ or $B_2 = S_{tar}(I_G)$ 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 $B_1$ 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 $B_2$ produces visual artifacts in the blended output $\hat{I}$, because 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 $B_2$ 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:

$$\hat{\mathcal{L}}_M = \sum_{l=1}^{L} \left\| S_{union}'(I) \odot (\phi_l'(I_G) - \phi_l'(I)) \right\|^2_2$$  \hspace{1cm} (13)

where $\phi(\cdot)$ are LPIPS [64] activations from the $l$-th layer of a pretrained VGG network, $S_{union}'$ 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...
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 $n$, 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 [64] and is applied over the entire face region, shown in parsed form on the right.

visual artefacts. InterFaceGAN and MaskFaceGAN generate higher-resolution results, again with the desired semantic content. 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.

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.

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 InterFaceGAN 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” attribute was 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. Ferrer, A. Morales, and J. Ortega-Garcia, “Deepfakes and beyond: A survey of face manipulation and fake detection,” Inf. Fusion, vol. 64, pp. 131–148, Dec. 2020.
[2] S. Jiang, Z. Tao, and Y. Fu, “Geometrically editable face image translation with adversarial networks,” IEEE Trans. Image Process., vol. 30, pp. 2771–2783, 2021.
[3] V. Mirjalili, S. Raschka, and A. Ross, “PrivacyNet: Semi- adversarial networks for multi-attribute face privacy,” IEEE Trans. Image Process., vol. 29, pp. 9400–9412, 2020.
[4] H. Deng, C. Han, H. Cai, G. Han, and S. He, “Spatially-variant style-codes controlled makeup transfer,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 6545–6553.
[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 Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 8790–8797.
[6] Y. Jo and J. Park, “SC-FEGAN: Face editing generative adversarial network with user’s sketch and color,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 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 Trans. Image Process., vol. 28, no. 11, pp. 5464–5478, Nov. 2019.
[8] M. Liu et al., “STGAN: A unified selective transfer network for arbitrary image attribute editing,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 3668–3677.
[9] Y. Shen, J. Gu, X. Tang, and B. Zhou, “Interpreting the latent space of GANs for semantic face editing,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 9240–9249.
[10] I. Goodfellow et al., “Generative adversarial nets,” in Proc. Adv. Neural Inf. Process. Syst. (NIPS), 2014, pp. 1–9.
[11] Z. Shu, E. Yumer, S. Hadap, K. Sunkavalli, E. Shechtman, and D. Samaras, “Neural face editing with intrinsic image disentangling,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5444–5453.
[12] S. Sengupta, A. Kanazawa, C. D. Castillo, and D. W. Jacobs, “SISNet: Learning shape, reflectance and illuminance of faces ‘in the wild’,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 6296–6305.
[13] W. Xia, Y. Zhang, Y. Yang, J.-H. Xue, B. Zhou, and M.-H. Yang, “GAN inversion: A survey,” 2021, arXiv:2101.05278.
[14] Y. Shen, C. Yang, X. Tang, and B. Zhou, “InterFaceGAN: Interpreting the disentangled face representation learned by GANs,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 4, pp. 2004–2018, Apr. 2022.
[15] R. Abdal, Y. Qin, and P. Wonka, “Image2StyleGAN: How to embed images into the StyleGAN latent space?,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 4431–4440.
[16] R. Abdal, Y. Qin, and P. Wonka, “Image2StyleGAN++: How to edit the embedded images?,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 8293–8302.
[17] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of StyleGAN,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 8107–8116.
[18] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, “Self-attention generative adversarial networks,” in Proc. Int. Conf. Mach. Learn. (ICML), 2019, pp. 7354–7363.
[19] A. Brock, J. Donahue, and K. Simonyan, “Large scale GAN training for high fidelity natural image synthesis,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2018, pp. 1–35.
[20] J. Johnson, A. Gupta, and L. Fei-Fei, “Image generation from scene graphs,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 1219–1228.
[21] O. Ashual and L. Wolf, “Specifying object attributes and relations in interactive scene generation,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 4560–4568.
[22] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, “Semantic image synthesis with spatially-adaptive normalization,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 2332–2341.
[23] T. Xu et al., “AttGAN: Fine-grained text to image generation with attentional generative adversarial networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 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 Trans. Image Process., 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 Trans. Image Process., vol. 30, pp. 2798–2809, 2021.
[26] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2016, pp. 1–16.
[27] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of GANs for improved quality, stability, and variation,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2018, pp. 1–26.
[28] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 4396–4405.
[29] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, “Least squares generative adversarial networks,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2813–2821.
[30] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in Proc. Int. Conf. Mach. Learn. (ICML), 2017, pp. 214–223.
[31] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of Wasserstein GANs,” in Proc. Adv. Neural Inf. Process. Syst. (NIPS), 2017, pp. 5767–5777.
[32] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, “Spectral normalization for generative adversarial networks,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2018, pp. 1–26.
[33] L. Mescheder, A. Geiger, and S. Nowozin, “Which training methods for GANs do actually converge?,” in Proc. Int. Conf. Mach. Learn. (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 Comput. Surv., vol. 54, no. 2, pp. 1–38, Mar. 2022.
[35] D. Saxena and J. Cao, “Generative adversarial networks (GANs): Challenges, solutions, and future directions,” ACM Comput. Surv., vol. 54, no. 3, pp. 1–42, Apr. 2022.
[36] D. Bau et al., “GAN dissection: Visualizing and understanding generative adversarial networks,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2019, pp. 1–18.
[37] A. Jahanian, L. Chai, and P. Isola, “On the ‘steerability’ of generative adversarial networks,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2020, pp. 1–31.
[38] L. Goetschelucks, A. Andonian, A. Oliva, and P. Isola, “GANalyze: Toward visual definitions of cognitive image properties,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 5743–5752.

[39] C. Yang, Y. Shen, and B. Zhou, “Semantic hierarchy emerges in deep generative representations for scene synthesis,” Int. J. Comput. Vis., vol. 129, pp. 1451–1466, Feb. 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 Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 2343–2442.

[41] A. Frühstück, K. K. Singh, E. Shechtman, N. J. Mitra, P. Wonka, and J. Lu, “InsetGAN for full-body image generation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 7713–7722.

[42] J. Lin, R. Zhang, F. Ganz, S. Han, and J.-Y. Zhu, “Anycost GANs for interactive image synthesis and editing,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 14981–14991.

[43] M. Alifi, M. A. Brubaker, and M. S. Brown, “HistGAN: Controlling colors of GAN-generated and real images via color histograms,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 7937–7946.

[44] T. Portenier, Q. Hu, A. Szabó, S. A. Bigdeli, P. Favaro, and M. Zwicker, “Face2shop: Deep sketch-based face image editing,” ACM Trans. Graph., vol. 37, no. 4, pp. 1–13, Aug. 2018.

[45] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, “MaskGAN: Towards diverse and interactive facial image manipulation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 5545–5557.

[46] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2242–2251.

[47] S. Vandenhende, S. Georgoulis, B. De Brabandere, and L. Van Gool, “Branched multi-task networks: Deciding what layers to share,” in Proc. Brit. Mach. Vis. Conf. (BMVC), 2020, pp. 1–9.

[48] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” in Proc. Comput. Vis. Pattern Recognit. (CVPR), 2017, pp. 1–14.

[49] T. Van Erven and P. Harremos, “Rényi divergence and Kullback–Leibler divergence,” IEEE Trans. Inf. Theory, vol. 60, no. 7, pp. 3797–3820, Jul. 2014.

[50] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang, “Interactive facial feature localization,” in Proc. Eur. Conf. Comput. Vis. (ECCV), 2012, pp. 679–692.

[51] T. F. Vieira, A. Bottino, A. Laurentini, and M. De Simone, “Detecting siblings in image pairs,” Vis. Comput., vol. 30, no. 12, pp. 1333–1345, Dec. 2014.

[52] P. J. Phillips et al., “Overview of the face recognition grand challenge,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2005, pp. 947–954.

[53] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre, “XM2VTSDB: The extended M2VTS database,” in Proc. 2nd Int. Conf. Audio Video-Based Biometric Pers. Authentication, vol. 964, 1999, pp. 905–906.

[54] P. Terzić, D. Fähmann, J. N. Kolfs, N. Damer, F. Kirchbuecher, and A. Kuijper, “MAAD-face: A massively annotated attribute database for face images,” IEEE Trans. Inf. Forensics Security, vol. 16, pp. 3942–3957, 2021.

[55] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, “On the importance of initialization and momentum in deep learning,” in Proc. Int. Conf. Mach. Learn. (ICML), 2013, pp. 1139–1147.

[56] Y. Zhang et al., “DatasetGAN: Efficient labeled data factory with large-scale image recognition,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (ICCV), Jun. 2015, pp. 3431–3440.

[57] P. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” in Proc. Med. Image Comput. Assist. Intervent., 2015, pp. 234–241.

[58] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 3431–3440.

[59] R. Zhang et al., “DatasetGAN: Efficient labeled data factory with large-scale image recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 586–595.

[60] Martin Perunš received the bachelor’s and master’s degrees from the Faculty of Electrical Engineering, University of Ljubljana, in 2016 and 2018, respectively, where he is currently pursuing the Ph.D. degree with the Laboratory for Machine Intelligence. He is a Junior Researcher with the Laboratory for Machine Intelligence, Faculty of Electrical Engineering, University of Ljubljana. His current research interests include deep learning, artificial intelligence, and image synthesis.

[61] Vitomir Štruc received the Ph.D. degree from the Faculty of Electrical Engineering, Ljubljana, in 2010. He is currently a Full Professor with the University of Ljubljana, Slovenia. His current research interests include problems related to biometrics, computer vision, image processing, pattern recognition, and machine learning. He has (co)authored more than 150 research papers for leading international peer-reviewed journals and conferences in these and related areas. He is a member of IAPR, EURASIP, and Slovenia’s National Contact Point for the EAB, and the President of the Slovenian Pattern Recognition Society (Slovenian branch of IAPR). He served in different capacities on the organizing committees of several top-tier vision conferences, including IEEE Face and Gesture, ICBF, WACV, and IJCB. He served as the Area Chair for WACV in 2018, 2019, and 2020; Eusipco in 2019; and FG in 2020. He is a Senior Area Editor of IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, a Subject Editor of Signal Processing (Elsevier), and an Associate Editor of Pattern Recognition and IET Biometrics.

[62] Simon Dobrišek (Senior Member, IEEE) received the Ph.D. degree from the Faculty of Electrical Engineering, University of Ljubljana, in 2001. He is currently an Associate Professor of electrical engineering and the Head of the Laboratory for Machine Intelligence, Faculty of Electrical Engineering, University of Ljubljana. In the last ten years, he has been involved as a key team member in eight national and international ICT projects and as a project manager and a principal researcher in two EU FP7 projects in the field of biometrics, speech technologies, pattern recognition, machine learning, and smart surveillance technology ethics. He has published popular journal articles and peer-reviewed scientific and conference papers on biometrics, ambient intelligence technology, spoken language technologies, and privacy protection. His current research interests include artificial intelligence, pattern recognition, biometrics, smart surveillance systems, and spoken language technologies. He is a member of IAPR, ISCA and the former President of the Slovenian Language Technologies Society. He was the Local Organizing Committee Chairperson of the 2018 ITTH Conference and the 2019 TSD Conference.