Pattern Recognition Letters 176 (2023) 14–20

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

FrankenMask: Manipulating semantic masks with transformers for face parts editing

Tomaso Fontanini [a,*], Claudio Ferrari [a], Giuseppe Lisanti [b], Leonardo Galteri [d], Stefano Berretti [c], Massimo Bertozzi [a], Andrea Prati [a]

[a] Department of Engineering and Architecture, University of Parma, Via delle Scienze, 181/a, Parma, 43124, Italy
[b] Department of Computer Science and Engineering, University of Bologna, Mura Anteo Zamboni 7, Bologna, 40126, Italy
[c] Department of Information Engineering, University of Florence, Via di S. Marta, 3, Florence, 50139, Italy
[d] Pegaso Telematic University, Italy

ARTICLE INFO

Editor: Maria De Marsico

Dataset link: https://github.com/switchablenorms/CelebAMask-HQ, https://github.com/TFonta/FrankenMask_semantic

Keywords:
Generative adversarial networks
Semantic image synthesis
Face analysis
Transformers

ABSTRACT

In this paper, we propose FrankenMask, a novel framework that allows swapping and rearranging face parts in semantic masks for automatic editing of shape-related facial attributes. This is a novel yet challenging task as substituting face parts in a semantic mask requires to account for possible spatial misalignment and the adaptation of surrounding regions. We obtain such a feature by combining a Transformer encoder to learn the spatial relationships of facial parts, with an encoder–decoder architecture, which reconstructs a complete mask from the composition of local parts. Reconstruction and attribute classification results demonstrate the effective synthesis of facial images, while showing the generation of accurate and plausible facial attributes. Code is available at https://github.com/TFonta/FrankenMask_semantic.

1. Introduction

Semantic image synthesis refers to the task of generating photo-realistic images conditioned on a semantic segmentation mask. Given a pixel-wise segmentation mask and a reference style image, it is possible to generate an image with a spatial layout corresponding to that defined by the mask, and the style defined by the image. A major breakthrough in this field was achieved by Park et al. [1], who proposed SPADE, a conditional de-normalization layer that modulates the activations through transformations learned from the input semantic layouts. Since then, several variants have been subsequently proposed [2–5]. This surge of interest is motivated by the astonishing generation results achieved, and the versatility that this solution brings along. In fact, using a segmentation mask allows controlling the image generation in many ways and at different levels of detail. It becomes possible to locally control and transfer styles [5], or manipulate attributes [2]. In such a framework, different styles can be applied both globally and locally by simply picking them from different semantic classes [5]. Oppositely though, performing local shape manipulations is more difficult, since local regions of the semantic mask need to be changed. To do so, current models either allow manipulating mask parts using some graphical interface, or swapping parts across masks, yet only in case of a very precise alignment. For example, to perform smiling transfer Lee et al. [2] selected suitable pairs of masks with the aid of a pose estimator, and simply exchanged the mouth parts on the masks to transfer the smile. As we will show, doing so on random pairs of masks and/or parts would introduce severe artifacts due to spatial inconsistencies in the masks.

Given the above limitation, the capability of automatically manipulating and rearranging semantic parts without strict alignment constraints or manual intervention could represent an important improvement for semantic image synthesis approaches. Addressing this problem though requires facing several challenges. In fact, other than spatial misalignment, substituting a face part in a semantic mask demands for adapting the surrounding regions accordingly. This becomes even more challenging if we wish to change several parts simultaneously.

In this paper, we propose a solution to the above problem by introducing *FrankenMask*, an encoder–decoder network that allows us to locally manipulate the shape of local parts in semantic masks on an individual basis. This module is completely independent from the image generators, and can be ideally applied on top of most of the

* Corresponding author.
E-mail addresses: tomaso.fontanini@unipr.it (T. Fontanini), claudio.ferrari2@unipr.it (C. Ferrari), giuseppe.lisanti@unibo.it (G. Lisanti), leonardo.galteri@unipegaso.it (L. Galteri), stefano.berretti@unifi.it (S. Berretti), massimo.bertozzi@unipr.it (M. Bertozzi), andrea.prati@unipr.it (A. Prati).

https://doi.org/10.1016/j.patrec.2023.10.010
Received 21 December 2022; Received in revised form 14 July 2023; Accepted 16 October 2023
Available online 21 October 2023

Fig. 1. Automatic manipulation of face images: firstly, different parts of the face (hair, mouth, *etc.*) are selected from different subjects using multiple semantic masks (the parts are highlighted in the RGB images using their original semantic mask color). A direct combination of the parts (Hard Swap) would result in inconsistent masks. **FrankenMask** instead reorganizes the selected parts in a consistent way so that a realistic RGB image can be produced using any pre-trained model.

current semantic face image synthesis models, making them fully and automatically controllable in terms of both style and shape. Some examples of the possibilities enabled by the proposed method are shown in Fig. 1, where we generate a face image by combining the shape of face parts from several semantic masks, with the style of the input image (leftmost column). Our network is composed of an encoder, a transformer encoder and a decoder. The encoder processes each part of the semantic segmentation mask independently, encoding it into a latent vector. Once all the N mask parts are encoded, they are stacked to form a sequence. The most peculiar characteristic of our architecture is the transformer encoder, which takes the above sequence as input to learn the relationships across different parts. It then outputs a sequence of N latent vectors that are finally concatenated and fed to the decoder, which outputs the complete semantic mask. The output of FrankenMask can then be used as an input to any state-of-the-art semantic image synthesis generators like [2] or [5], unlocking several new editing possibilities.

It is worth noticing that, differently from modern vision Transformers [6] (ViTs), where visual encodings are directly learned on image patches, in our work the transformer encoder is used to learn shape and spatial relationships among face parts, represented as sequences of latent vectors obtained through a CNN. We exploit the fact that each face part is an independent channel in the input data. Hence, each latent encoding captures the parts shape, and the transformer can effectively learn their relationships, ultimately allowing us to obtain realistic outputs even in case of strong manipulation of the input.

Overall, the main contributions of this paper are as follows:

- We propose an alternative way of disentangling face parts in semantic image synthesis, introducing the problem of semantic mask manipulation;
- A new model, called *FrankenMask*, capable of automatically rearranging any number of parts from multiple segmentation masks without requiring alignment constraints;
- The possibility of enriching multiple state-of-the-art methods for mask-to-RGB face synthesis, allowing for fully automatic manipulation without any architectural change.

The proposed solution has been extensively evaluated both quantitatively and qualitatively. Qualitative results demonstrate the effective capability of synthesizing pleasant target images, where multiple regions of the face are changed based on source masks taken from different individuals. On the other hand, measures of reconstruction error and attribute classification evidence that processing the masks with our network does not compromise the image generation quality, while obtaining good results in terms of shape-related attribute transfer.

2. Related work

The task of generating and manipulating face images was deeply explored in the recent years with a large corpus of different approaches that were proposed [7–11]. Among all these, Semantic Image Synthesis (SIS) methods are all based on similar frameworks, which are composed of two main modules: a Style Encoder and a Generator network. The latter takes the semantic mask as input, while the style encoder is used to encode the style, *i.e.* texture, which is used to condition the generator so to output the final image. Style can be either extracted from an RGB image, or generated from noise. This difference leads to further divide methods in *diversity*, and *quality* driven. In their seminal work, Park et al. [1] first addressed this problem by noticing that in conventional synthesis architectures that stack convolutional, normalization, and non-linearity layers [12,13], the normalization layers tend to remove information contained in the input semantic masks. To solve this problem, they proposed the SPatially-Adaptive (DE)normalization (SPADE) method, where a conditional normalization layer modulates the generator activations through a spatially adaptive, learned transformation. The same model, also known as GauGAN [14] was deployed in a GAN-based framework. However, in SPADE only a single style code is used to control the entire style of an image, resulting in a lack of fine-grained style control. To solve this issue, Zhue et al. [5] proposed an architecture that extracts one style code per semantic class with a dedicated style encoder. Further, they designed a SEmantic region-Adaptive Normalization block (SEAN) that can use style input images to create spatially varying normalization parameters per semantic class. Simultaneously, Lee et al. [2] proposed MaskGAN to enable diverse and interactive face manipulation. The main idea is that of learning the face manipulation process as traversing on the mask manifold, rather than directly transforming images in the pixel space. This produces more diverse results with respect to facial components, shapes, and poses. Tan et al. [4] proposed the INstance-Adaptive DEnormalization (INADE) approach that is capable of producing diverse results at semantic or instance level. The intuition is to treat each semantic class as one distribution so that each instance of this class could be drawn from this distribution as a discrete sample. In [15], Tan et al. proposed a CLass-Adaptive (DE)normalization layer (CLADE) that, differently from the spatially adaptive solution of SPADE, uses the input semantic mask to modulate the normalized activation in a class-adaptive manner. Along this line, several other works were proposed [3,16–19].

The proposed FrankenMask positions itself upstream, providing an automatic tool to manipulate the semantic masks prior to input them to a SIS generator. All the aforementioned methods, instead, are not suited to handle heavy and automatic editing of the semantic masks. On the other side, the proposed FrankenMask allows us to change the shape of the masks locally by performing swaps of multiple parts and rearranging them to create a realistic result. This novel ability could lead us to, and inspire, significant improvements in semantic image synthesis methods.

3. FrankenMask network

Our goal is to train a network capable of manipulating, or swapping, local face regions of semantic segmentation masks, in order to

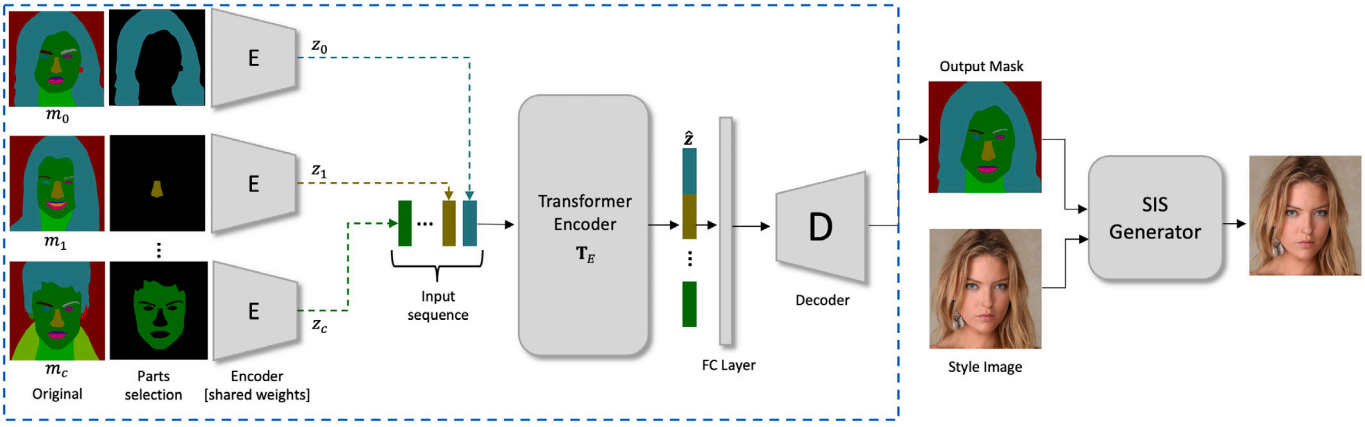

Fig. 2. FrankenMask (the dotted blue rectangle in the figure) takes a semantic segmentation mask m as an input, and uses the encoder $\mathbf{E}$ to encode one by one each channel m_i of the mask corresponding to a face part. We interpret the encoded channels z_i as a sequence, and feed them to a transformer encoder $\mathbf{T}_E$, producing a new sequence. A decoder $\mathbf{D}$ takes the concatenated and reshaped sequence and outputs a new segmentation mask. The generated mask can be used as input to any Semantic Image Synthesis (SIS) generator, so to output a manipulated RGB face image.

enable fully automatic face editing. A challenge here is that obtaining sufficient training data, *i.e.*, manipulated masks, to train such a module would require huge manual effort. In addition, differently from natural images, segmentation masks contain only shape/silhouette information, making it difficult to disentangle possible factors of variation. To overcome these issues, we devised a model that, in a totally unsupervised manner, is capable of learning relationships across face parts and manipulating the masks. The main design intuition consists of processing each mask channel, *i.e.*, face part, separately, producing a sequence of part embeddings. Before decoding the embeddings to reconstruct the mask, those are passed through a transformer encoder. The latter is designated to learn relationships among face parts, while the encoder–decoder learns to reconstruct, or re-compose, the input mask from its parts. In this way, when changing one or more parts, all the others are adjusted accordingly in order to output a rearranged mask, without the need of supervising the process.

The proposed network architecture is rather simple, and it is illustrated in Fig. 2. It is composed of three main modules: the encoder $\mathbf{E}$, the transformer encoder $\mathbf{T}_E$ and the decoder $\mathbf{D}$. The model takes as input a segmentation mask $m \in \mathbb{R}^{H\times W\times C}$ and produces as output a new segmentation mask $\hat{m} \in \mathbb{R}^{H\times W\times C}$. The mask m is an image composed of $C = 18$ channels, one for each face part. The background channel is not considered and is subsequently added to the reconstructed mask by simple difference with respect to the other channels.

More in detail, each mask channel $m_i \in \mathbb{R}^{H\times W}$ is fed to the encoder separately. The encoder is composed of a series of convolutional layers that iteratively down-sample the input in order to obtain a compressed representation z_i for each part of the mask. The resulting embeddings $\mathbf{z} = [z_1, \dots, z_C]$ are arranged and interpreted as a sequence and used as input to the transformer encoder, which produces a new representation $\hat{\mathbf{z}} = [\hat{z}_1, \dots, \hat{z}_C]$. In this way, each element $\hat{z}_i$ of the new sequence depends on all the other elements thanks to the attention mechanism of the transformer. Then, $\hat{\mathbf{z}}$ is concatenated into a single latent code and reshaped by passing through a fully connected layer. Finally, the decoder produces a complete output mask $\hat{m} \in \mathbb{R}^{H\times W\times C}$. The decoder is composed of a series of upsampling blocks and convolutions that revert the latent vector to the original input image shape producing the output. The main advantage of this architecture is the capability of the transformer encoder $\mathbf{T}_E$ of modeling the relationship between all of the different mask parts. In particular, the transformer will rearrange the latent space allowing impressive robustness when swapping multiple parts of the input mask m. Ultimately, this leads the decoder to produce varied yet realistic and plausible results.

3.1. Transformer encoder

As previously mentioned, the encoder $\mathbf{E}$ produces a set of embeddings, corresponding to the different segmentation mask parts. Then, in order to provide the model with meaningful order information between each mask part, a fixed positional encoding (as used in [20]) is summed to each embedding. Therefore, the input sequence is defined as:

$$\mathbf{z} = [\mathbf{E}(m_1) + PE(m_1), \dots, \mathbf{E}(m_C) + PE(m_C)], \tag{1}$$

where PE is the positional encoding, which has the same dimension as the embedding produced by the encoder.

The sequence $\mathbf{z}$ is fed to the transformer encoder $\mathbf{T}_E$, which is composed of six layers of stacked multi-head self-attention and fully connected layers. In particular, the self-attention models dependencies and relationships between the different elements in the sequence and, therefore, between each of the parts that compose the segmentation mask m. We will show that this property of the transformer is crucial for guiding the decoder to output consistent and realistic masks without noticeable artifacts. On the other hand, a possible negative outcome of the manipulation performed by the transformer is that of changing their shape. However, since the network is trained to correctly reconstruct the shape of the parts so as to build a consistent mask, it will also fix inaccurate parts arrangement. Minor shape changes though occur in the case of unrealistic compositions. For example, if the swapped mouth is too large to fit the face, the involved parts will be slightly modified so that they are placed in a plausible face-like configuration, ultimately looking more realistic.

3.2. Loss function

FrankenMask is trained only to self-reconstruct the input mask from its parts using a reconstruction loss, and no swap is performed during its optimization. The only training supervision is a pixel-wise cross-entropy loss $\mathcal{L}_{CE}$ between the input and the output of the model:

$$\mathcal{L}_{CE} = -\frac{\sum_{j=1}^{H}\sum_{k=1}^{W} m^{j,k} \log \hat{s}^{j,k}}{H \times W}, \tag{2}$$

where $H \times W$ is the number of pixels in the input mask and $\hat{s} = \text{Softmax}(\hat{m})$. Using only a reconstruction loss and no additional supervision proved to be enough to allow robustness during the swapping, showing the learning capability of the transformer encoder. Despite its simplicity, we will show that this framework allows us to consistently manipulate, swap or interpolate one or more face parts across different masks.

Table 1
Reconstruction metrics using FrankenMask in conjunction with different generators, namely SEAN, MaskGAN, SC-GAN and INADE.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	FRD ↑
SEAN [5]	18.03	0.566	0.204	21.09	0.574
FrankenMask+SEAN	17.87	0.546	0.216	22.55	0.555
MaskGAN [2]	9.03	0.272	0.546	47.59	0.334
FrankenMask+MaskGAN	9.02	0.274	0.545	55.11	0.328
V-Inade [4]	13.33	0.437	0.342	22.21	0.567
FrankenMask+V-Inade	13.23	0.431	0.348	23.93	0.537
SCGAN [17]	15.74	0.508	0.286	20.23	0.639
FrankenMask+SCGAN	15.50	0.493	0.297	22.03	0.599

Table 2
Attribute classification; for each attribute, transfer results are averaged over 5 different randomly chosen reference masks m^a_{ref}. Considered attributes: Big Nose (B.N.), Oval Face (O.F.), Smiling (S.), Narrow Eyes (N.E.), Arched Eyebrows (A.E.) and Eyeglasses (E.).

Method	Attribute classification % (Sensitivity)					
	B.N.	O.F.	S.	N.E.	A.E.	E.
Num. Samples	555	601	916	216	684	129
GT	66.8	47.4	92.8	39.1	79.9	99.2
FrankenMask + MaskGAN	67.2	32.8	99.8	31.5	88.7	94.6
FrankenMask + SEAN	84.8	50.9	98.9	69.9	87.8	99.2
FrankenMask + V-Inade	62.7	36.8	99.7	16.2	87.3	96.9
FrankenMask + SCGAN	75.7	48.8	100	13.4	94.4	96.2

3.3. Composing new masks with FrankenMask

In order to swap or compose new masks with FrankenMask, at inference time, we select: *(i)* a target mask m_{trg}; *(ii)* one (or more) source mask m_{src}; *(iii)* a set of t parts $\mathbf{p} = [p_0, \dots, p_t]$ to be swapped. Given the FrankenMask model, FM, we define the swapping operation as $\hat{m}_{sw} = FM(m_{src}, m_{trg}, \mathbf{p})$. More in detail, we embed the mask parts through the encoder *i.e.* $z_i = \mathbf{E}(m_i)\ \forall i$, choosing the part(s) $\mathbf{p}$ to be swapped from the source mask(s) m_{src}, and picking all the others from m_{trg}. For example, when swapping the part i, the sequence to be used as input to the transformer encoder becomes: $\mathbf{z} = [z^{trg}_1, \dots, z^{src}_i, \dots, z^{trg}_C]$. Note that, as shown in Fig. 1, having a target mask is not strictly necessary, and it is also possible to compose a new mask by mixing parts from C different masks. The complete mask is obtained by feeding $\mathbf{z}$ to the decoder, *i.e.*, $\hat{m}_{sw} = \mathbf{D}(\mathbf{z})$. When generating the final image by means of a synthesis network, different styles can be picked from other faces as well.

4. Experimental results

In the following, we report both quantitative (Sections 4.1 and 4.2) and qualitative (Section 4.3) results along with an ablation study to highlight the importance of the transformer encoder (Section 4.4).

Training details. Training was performed on an NVIDIA Quadro RTX 6000 for 50 epochs. We employed Adam [21] with a learning rate of 0.00001. We did not perform any data augmentation except for random horizontal flipping.

Dataset. We conducted all our experiments on the CelebAMask-HQ [2], which is a face dataset built upon CelebA-HQ [22]. It contains 30,000 segmentation masks and face images, divided into 28k for training, and 2k for testing.

Network configuration. The parameters used in all the experiments are the following: the size of each embedding z_i was set to 512; the transformer encoder $\mathbf{T}_E$ is composed of 6 layers, and 8 attention heads; finally, both the encoder and decoder have 7 downsampling and upsampling layers, respectively.

4.1. Reconstruction results

First, we aim at showing that FrankenMask can reconstruct accurate masks such that the performance of synthesis generators is not degraded if using reconstructed masks in place of ground-truth ones. The idea is: the closer the generated masks to real ones, the smaller the gap in the reconstruction metrics of the synthesized face images. To this end, in Table 1 we report results for some widely used reconstruction metrics (*e.g.*, PSNR, SSIM, LPIPS [23] and FID [24]). For comparison, we considered four state-of-the-art semantic image synthesis methods, namely SEAN [5], MaskGAN [2], SCGAN [17] and INADE [4], when equipped with a variational autoencoder to encode the styles (V-Inade). Note that absolute values of the metrics change depending on the specific generator; we remark that we are specifically interested in verifying the stability of the results in case original or reconstructed masks are used. Using semantic masks obtained with FrankenMask has a small degrading effect when provided as input to any of the above generators. In addition, we also used a Face Recognition Distance (FRD) metric to verify the reconstructed masks do not change the perceived identity when generating face images from them, highlighting that shape properties are correctly preserved. To compute FRD, we calculate the cosine similarity between deep embeddings obtained from the original and generated face images using a InceptionV1 model pre-trained on VggFace2 [25]. The FRD remains stable, indicating that FrankenMask compromises the face generation to a little extent.

4.2. Attribute classification

Standard metrics normally employed for assessing generative models are hardly applicable to semantic masks. A possible solution is to evaluate the accuracy of a facial attribute classifier when applied to face images generated by using masks in which face parts are swapped using FrankenMask. In this way, we can evaluate how accurately our network can maintain shape properties of swapped face parts, and generate semantic masks that lead to the generation of face images containing a given attribute. Clearly, we are not interested in attributes that are related to style, *e.g.*, pale skin, or heavy makeup. Some examples of suitable attributes are smiling, big nose, or eyeglasses. It would look natural in this scenario to apply this strategy to all the available test masks. Practically though, manipulating an attribute is not always possible, for instance in case of occlusions. In such cases, the classification would be necessarily incorrect. To ensure a meaningful assessment, for each attribute we only consider images that originally included it, in order to make sure the attribute can be manipulated. In this way, we can compare the classifier performance on the original images, against that obtained after transferring the attributes.

The chosen test attributes are the following: big nose, oval face, smiling, narrow eyes, arched eyebrows and eyeglasses. These were selected in order to include all the relevant face parts, *i.e.*, nose, face, mouth, eyes, eyebrows. For each one, following [2], we fine-tune a ResNet-18 model as binary classifier on the training set of CelebMask-HQ. Table 2 reports the classifiers sensitivity obtained on the ground-truth images of the test set. We underline that we report the sensitivity and not the accuracy since all the samples should include the attribute, thus being positive instances.

To perform the transfer, for each attribute a, we first identify which semantic parts $\mathbf{p}^a$ of the mask are responsible for its detection, *e.g.*, smiling is defined by mouth, upper lip and lower lip. A set of $k = 5$ random reference masks m^a_{ref} containing a are then selected from the test set of CelebMask-HQ. The reference masks are chosen such that the corresponding ground-truth image is classified correctly. We then use FrankenMask to swap face parts $\mathbf{p}^a_{ref}$ from the reference masks m^a_{ref} across other masks m_{trg}, and generate manipulated masks that contain a, that is $\hat{m}^a_{sw} = FM(m^a_{ref}, m_{trg}, \mathbf{p}^a_{ref})$. Finally, we use these

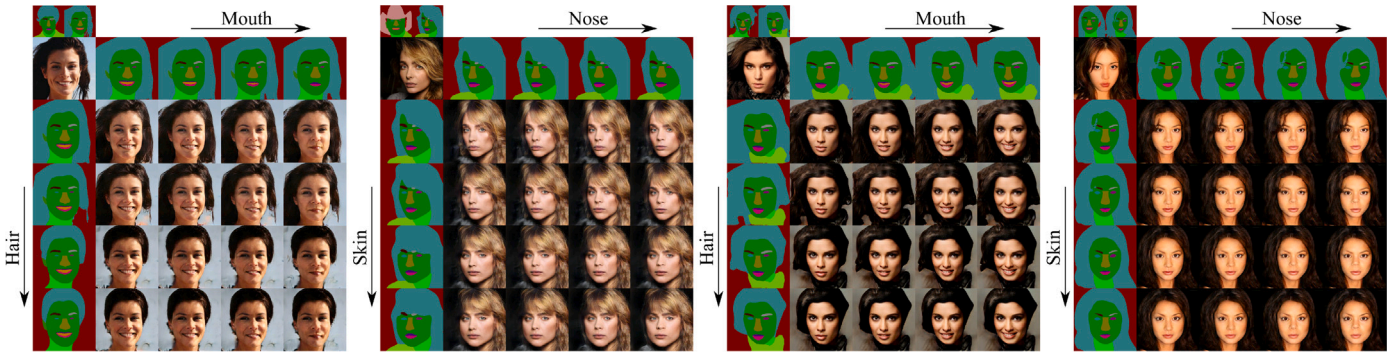

Fig. 3. Examples of interpolation of different swapped parts. The first two results (from left) were obtained by combining FrankenMask and SEAN, while the third and fourth results were obtained by combining FrankenMask and MaskGAN.

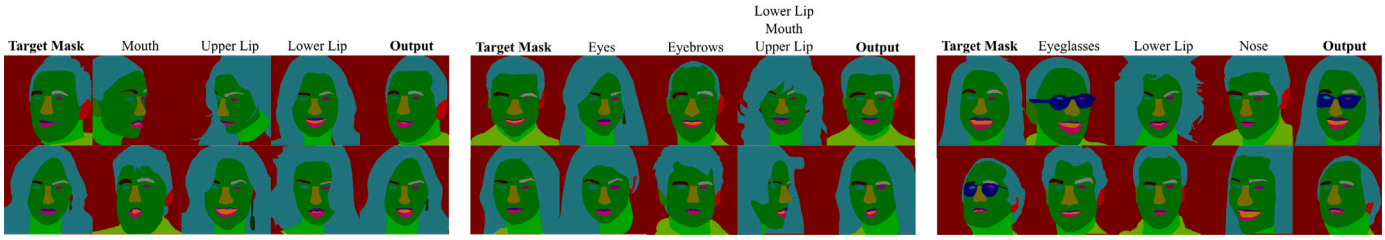

Fig. 4. Examples of manipulation using FrankenMask to swap multiple parts from different masks of various shapes and orientations.

new masks $\hat{m}^a_{sw}$ as input to a semantic image synthesis network and generate images to be used for attribute classification.

Table 2 reports the attribute recognition sensitivity. It turns clearly out that, by using FrankenMask, the manipulated masks are precise enough to generate faces for which the transferred attribute is easily recognizable. Among the tested ones, results for two specific attributes *i.e. Oval Face* and *Narrow Eyes*, show a decreased sensitivity except in case the SEAN generator is used. Such attributes though are challenging to detect, due to the minimal change that is applied to the mask after the swapping (*e.g.*, "skin" always involves most of the face region, while for "narrow eyes" the difference is itself tiny, and could not be noticed). One can indeed observe that even the sensitivity obtained using the ground-truth masks is itself low, if compared to attributes like *Big Nose*, *Smiling*, *Arched Eyebrows* and *Eyeglasses*, which instead exhibit more noticeable differences in the final mask. The fact that SEAN can instead render images that lead to higher detection is due to its improved generation precision. Finally, FrankenMask contribution and ability to preserve shape properties can be better appreciated in Fig. 5.

4.3. Qualitative results

Qualitative results are presented in Figs. 3 and 4. In Fig. 3, we show multiple interpolation results of different mask parts. The interpolation can be obtained thanks to the FrankenMask architecture. In particular, given two different masks m' and m'', and the index i of the part(s) we wish to swap, the interpolation is performed as:

$$z_i^{int} = \alpha z'_i + (1-\alpha) z''_i, \tag{3}$$

where α is a scalar going from 0 to 1, and z'_i and z''_i are the embeddings of the parts obtained from m' and m'', respectively. Once the interpolation is performed, z_i^{int} is inserted into the sequence of embeddings $\mathbf{z}$ at position i, then the whole sequence is fed to the transformer encoder. Looking at Fig. 3, we can observe that FrankenMask allows us to independently control different parts, resulting in an impressive freedom in generating the face images. Such behavior was not possible using previous methods like SEAN or MaskGAN. Indeed, we can effectively interpolate single or multiple parts simultaneously.

Table 3
Classification results for different configurations of FrankenMask combined with SEAN [5]. Attributes: Big Nose (B.N.), Oval Face (O.F.), Smiling (S.), Narrow Eyes (N.E.), Arched Eyebrows (A.E.), Eyeglasses (E.).

Method	Attribute classification % (Sensitivity)					
	B.N.	O.F.	S.	N.E.	A.E.	E.
Hard Swap	84.1	27.9	97.2	32.8	86.5	99.0
Swap with AE	67.4	**56.1**	96.6	29.2	89.8	97.7
FrankenMask (w/o $\mathbf{T}_E$)	83.6	48.0	97.8	59.2	**91.0**	98.4
FrankenMask (Full)	**84.8**	50.9	**98.9**	**69.9**	87.8	**99.2**

Finally, Fig. 4 shows some masks produced by the proposed FrankenMask. Each of the results is obtained by first selecting different parts from multiple semantic masks, then swapping them with the corresponding parts in the target mask using FrankenMask. Indeed, the selection is made with no particular care of imposing the same orientation. On the contrary, often these masks are not aligned at all, which makes the FrankenMask task more difficult. In addition, a difficulty is that the swapped parts, being taken from different faces, are not correlated, making their composition even more problematic. Even in this challenging setting, the proposed architecture is able to generate realistic results by adjusting the shape and orientation of the swapped parts to fit in the target masks. In particular, eyeglasses and the whole mouth (composed of *upper lip, mouth* and *lower lip* parts) are swapped convincingly even when the target mask presents a completely opposite orientation. This outperforms previous methods like [2], where automatic swapping was only possible in a very constrained setting with the source and target masks completely aligned. In addition, the model is able to handle a large number of swaps at the same time, without affecting the quality of the generation.

4.4. Ablation study

In Fig. 5 and Table 3, we report the results of an ablation study aimed at assessing the importance of the transformer in our architecture. In particular, the following alternatives are evaluated: the first simply consists of swapping one mask channel, without further

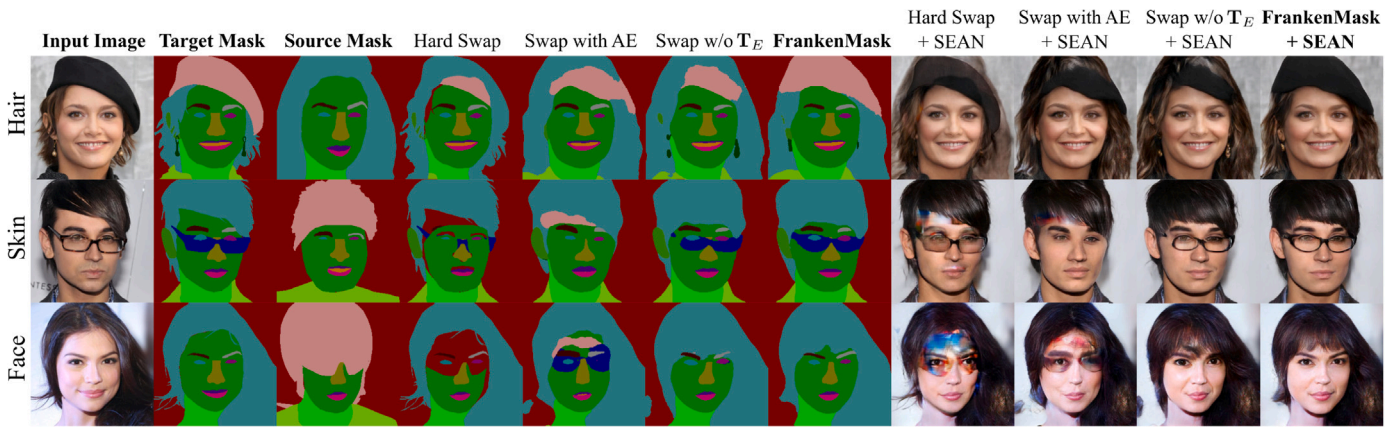

Fig. 5. Mask and RGB results in different scenarios when swapping different parts: first, the hard swap was performed without passing the resulting mask to FrankenMask; second, the full hard-swapped mask was passed through a simple Autoencoder; then, FrankenMask was used removing the transformer encoder; finally, the full FrankenMask was employed.

Table 4
Ablation study on different sizes of the mask embeddings. Results are reported as differences with respect to our final configuration for clarity (embedding size 512). Smaller embedding sizes lead to slightly worse results (red colored), while using a 1024 embedding size leads to only very slightly increased (green colored) or equal (= symbol) performance, at the cost of increased computational burden.

Size	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	FRD ↑	L1 ↓
512 (Ours)	17.87	0.546	0.216	22.55	0.555	0.0034
128	−0.07	−0.003	+0.001	+0.42	−0.005	−0.0005
256	−0.03	−0.002	+0.002	+0.20	−0.002	−0.0003
1024	+0.02	=	−0.001	−0.13	+0.001	=

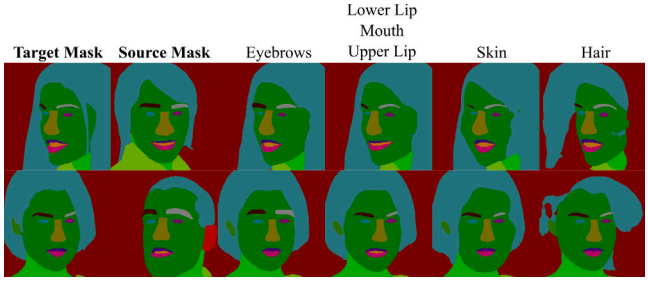

Fig. 6. Results when swapping parts with large pose variations between source and target mask.

processing (Hard Swap). This is intended to highlight how changing two mask channels leads to more or less severe spatial misalignment, which in turn results in holes appearing in the output mask. Then, we train a simple autoencoder to re-arrange the swapped channels (Swap with AE). Finally, we also train our architecture without the Transformer Encoder $\mathbf{T}_E$. Given the results of Sections 4.1 and 4.2, we used SEAN [5] as image generator for this experiment. Looking at Fig. 5, it can be observed that, when swapping the hair part, FrankenMask is able to adapt the new shape also in the presence of overlapping parts (like the hat in the first row), while the other configurations drastically fail. Also, in the third row, the whole face is swapped from the source to the target mask and, since the source mask face is obstructed by an helmet, FrankenMask needs to reconstruct a large portion of the face. Indeed, the transformer encoder allows for generating the missing portion of the face much more convincingly. One could argue that a simple hole-filling or similar approaches would suffice to fix the misalignment resulting from hard-swapping two mask channels. However, this is not the case since uniquely determining the correct class with which filling the holes is not possible, and might change depending on which part is swapped.

Table 3 instead reports classification results obtained as described in Section 4.2. Whereas the performance of different baselines are quite similar and might perform slightly better in cases where the source-target alignment is easier to maintain, the sensitivity of FrankenMask is way more stable across attributes and particularly pronounced when considering face parts that are more subject to misalignment, *e.g.*, narrow eyes, which require stronger re-arrangement. In these cases, simply swapping the parts is not suitable, and the ability of FrankenMask to correctly rearrange the semantic parts turns out fundamental.

Finally, in Table 4 an ablation of different part embedding sizes is performed. From the table, it turns out clearly that there is not much difference in the performance if changing the embedding size; generally, performance monotonically increases as we enlarge the embedding size, as expected. Visually, the generated images do not reveal significant changes, and so we eventually chose 512 as final size as it represents a good trade-off. Note that an additional reconstruction metric (L1) is added, which was not reported in Table 1 that is calculated directly between the reconstructed and ground-truth *semantic masks* rather than the synthesized RGB images. This specific metric further highlights that there is not much advantage is going beyond a size of 512, while a little accuracy is lost if using smaller embedding sizes.

5. Limitations and future perspectives

The proposed work is the first attempt of automatically manipulating semantic segmentation masks. On the one hand, we proposed an alternative way to disentangle face parts in semantic image synthesis, which can open the way to several interesting applications and improvements; on the other hand, there are still limitations and issues that need to be solved.

A major limitation is that of swapping larger parts such as hair or skin in case of large pose variations, *i.e.*, difference in head orientation. Whereas the transformer encoder can rearrange smaller parts to a large extent even in case of large pose differences (see for example Fig. 4 and in particular Fig. 6, where swaps are performed across multiple face masks with clear pose differences), larger ones are still difficult to handle.Arguably, the larger the portion of mask that is covered by a specific face part, the bigger its influence in defining the orientation and disposition of the others. Thus, in this scenario, it becomes ambiguous to determine the correct face orientation. Another limitation of the proposed framework design is that the semantic manipulation is performed at the class level, which partially prevents extending the use of our network to other datasets, where object classes might have one or more instances like for cars or pedestrians.

Finally, with the current framework, the manipulation of parts can be done only if a target mask is available. An extension of this work that we aim at exploring will include the possibility of manipulating the latent space using other external information such as attribute labels.

6. Conclusions

In this paper, we have proposed FrankenMask, an architecture that allows performing fine-grained editing on semantic masks by swapping any number of parts with no need for strict constraints over alignment and shape. The model is able to reorganize all the different parts consistently on the target semantic mask in order to produce a realistic output. This is achieved thanks to an encoder–decoder architecture with the notable addition of a transformer encoder. The latter helps learn the relationship between the semantic image parts and greatly boosts the generation quality. Indeed, the proposed network combined with semantic face image synthesis generators is able to unlock new possibilities in automatic face editing. We believe this novel alternative to disentangle shape and appearance modeling can open the way to several interesting applications.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data used for this research is publicly available at https://github.com/switchablenorms/CelebAMask-HQ for research purposes. Code can be found at https://github.com/TFonta/FrankenMask_semantic.

Acknowledgments

This work was partially supported by the Programme “FIL-Quota Incentivante” of University of Parma, co-sponsored by Fondazione Cariparma, Italy, and PRIN 2020 “LEGO.AI: LEarning the Geometry of knOwledge in AI systems”, grant no. 2020TA3K9N funded by the Italian MIUR. All authors approved the final version of manuscript to be published’ to Acknowledgment section.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.patrec.2023.10.010.

References

[1] T. Park, M.-Y. Liu, T.-C. Wang, J.-Y. Zhu, Semantic image synthesis with spatially-adaptive normalization, in: IEEE/CVF Conf. on Computer Vision and Pattern Recognition, 2019, pp. 2337–2346.
[2] C.-H. Lee, Z. Liu, L. Wu, P. Luo, Maskgan: Towards diverse and interactive facial image manipulation, in: IEEE/CVF Conf. on Computer Vision and Pattern Recognition, 2020, pp. 5549–5558.
[3] X. Liu, G. Yin, J. Shao, X. Wang, H. Li, Learning to predict layout-to-image conditional convolutions for semantic image synthesis, in: Int’L. Conf. on Neural Information Processing Systems, 2019.
[4] Z. Tan, M. Chai, D. Chen, J. Liao, Q. Chu, B. Liu, G. Hua, N. Yu, Diverse semantic image synthesis via probability distribution modeling, in: IEEE/CVF Conf. on Computer Vision and Pattern Recognition, 2021, pp. 7962–7971.
[5] P. Zhu, R. Abdal, Y. Qin, P. Wonka, Sean: Image synthesis with semantic region-adaptive normalization, in: IEEE/CVF Conf. on Computer Vision and Pattern Recognition, 2020, pp. 5104–5113.
[6] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, 2020, arXiv preprint arXiv:2010.11929.
[7] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, T. Aila, Analyzing and improving the image quality of stylegan, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 8110–8119.
[8] B. Zeno, I. Kalinovskiy, Y. Matveev, B. Alkhatib, CtrlFaceNet: Framework for geometric-driven face image synthesis, Pattern Recognit. Lett. 138 (2020) 527–533.
[9] B. Huang, W. Chen, X. Wu, C.-L. Lin, P.N. Suganthan, High-quality face image generated with conditional boundary equilibrium generative adversarial networks, Pattern Recognit. Lett. 111 (2018) 72–79.
[10] T. Fontanini, L. Donati, M. Bertozzi, A. Prati, Unsupervised discovery and manipulation of continuous disentangled factors of variation, ACM Trans. Multimed. Comput., Commun. Appl. 19 (6) (2023) 1–25.
[11] C. Ferrari, S. Berretti, P. Pala, A. Del Bimbo, Rendering realistic subject-dependent expression images by learning 3DMM deformation coefficients, in: Proceedings of the European Conference on Computer Vision (ECCV) Workshops, 2018.
[12] P. Isola, J. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: IEEE/CVF Conf. on Computer Vision and Pattern Recognition, 2017, pp. 5967–5976.
[13] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, B. Catanzaro, High-resolution image synthesis and semantic manipulation with conditional GANs, in: IEEE/CVF Conf. on Computer Vision and Pattern Recognition, 2018, pp. 8798–8807.
[14] T. Park, M.-Y. Liu, T.-C. Wang, J.-Y. Zhu, GauGAN: Semantic image synthesis with spatially adaptive normalization, in: ACM SIGGRAPH 2019 Real-Time Live!, 2019.
[15] Z. Tan, D. Chen, Q. Chu, M. Chai, J. Liao, M. He, L. Yuan, G. Hua, N. Yu, Efficient semantic image synthesis via class-adaptive normalization, IEEE Trans. Pattern Anal. Mach. Intell. (2021).
[16] Y. Li, Y. Li, J. Lu, E. Shechtman, Y.J. Lee, K.K. Singh, Collaging class-specific GANs for semantic image synthesis, in: IEEE/CVF International Conference on Computer Vision, 2021, pp. 14418–14427.
[17] Y. Wang, L. Qi, Y.-C. Chen, X. Zhang, J. Jia, Image synthesis via semantic composition, in: IEEE/CVF International Conference on Computer Vision, 2021, pp. 13749–13758.
[18] H. Ling, K. Kreis, D. Li, S.W. Kim, A. Torralba, S. Fidler, EditGAN: High-precision semantic image editing, in: Advances in Neural Information Processing Systems, 2021.
[19] Y. Shi, X. Yang, Y. Wan, X. Shen, SemanticStyleGAN: Learning compositional generative priors for controllable image synthesis and editing, 2021, arXiv preprint arXiv:2112.02236.
[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[21] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.
[22] T. Karras, T. Aila, S. Laine, J. Lehtinen, Progressive growing of gans for improved quality, stability, and variation, 2017, arXiv preprint arXiv:1710.10196.
[23] R. Zhang, P. Isola, A.A. Efros, E. Shechtman, O. Wang, The unreasonable effectiveness of deep features as a perceptual metric, in: IEEE/CVF Conf. on Computer Vision and Pattern Recognition, 2018, pp. 586–595.
[24] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, S. Hochreiter, Gans trained by a two time-scale update rule converge to a local nash equilibrium, Adv. Neural Inf. Process. Syst. 30 (2017).
[25] Q. Cao, L. Shen, W. Xie, O.M. Parkhi, A. Zisserman, Vggface2: A dataset for recognising faces across pose and age, in: IEEE Int’L. Conf. on Automatic Face & Gesture Recognition, 2018, pp. 67–74.