Unsupervised Holistic Image Generation from Key Local Patches Supplementary Material

Donghoon Lee[1], Sangdoo Yun[2], Sungjoon Choi[1], Hwiyeon Yoo[1],
Ming-Hsuan Yang[3,4], and Songhwai Oh[1]

[1] Electrical and Computer Engineering and ASRI, Seoul National University
[2] Clova AI Research, NAVER
[3] Electrical Engineering and Computer Science, University of California at Merced
[4] Google Cloud AI

In this supplementary material, we describe additional experimental results as summarized in Table 1 (see Section 4 of the manuscript for datasets).

Table 1. Numerous experiments in this work.

Experiment	Figure/Table
Image generation from key patches	Figure 1, Figure 2, Figure 3, Table 3
Image generation from random patches	Figure 4, Figure 5, Figure 6, Table 3
Part combination	Figure 7, Figure 8, Figure 9
Unsupervised feature learning	Table 4
An alternative objective function	Figure 10, Figure 11
An alternative network structure	Figure 12, Figure 13, Figure 14
Different number of input patches	Figure 15, Figure 16, Figure 17
Degraded input patches	Figure 18, Figure 19
User study	Figure 20, Figure 21, Table 5
Failure cases	Figure 22

1 Experimental Details

Table 2 shows detailed description of the proposed network for an image with a size of 256 × 256 × 3 pixels. The input parts are encoded into a 256-dimensional vector. The slope of 0.2 is used for the leaky ReLU activation. The filters in the network are initialized with a zero mean Gaussian distribution with a standard deviation of 0.02.

For the CelebA-HQ dataset, we randomly sample 128 images for evaluation and other images are used for training. For other databases, we randomly sample 10% of data for a test set.

Table 2. Details of each network for 256×256 pixels image generation. # Filter is the number of filters. BN is the batch normalization. Conv denotes a convolutional layer. F-Conv denotes a transposed convolutional layer that uses the fractional-stride.

Layer	# Filter	Filter Size	Stride	Padding	BN	Activation Function
Conv. 1	64	4×4×3	2	1	×	Leaky ReLU
Conv. 2	128	4×4×64	2	1	○	Leaky ReLU
Conv. 3	256	4×4×128	2	1	○	Leaky ReLU
Conv. 4	512	4×4×256	2	1	○	Leaky ReLU
Conv. 5	512	4×4×512	2	1	○	Leaky ReLU
Conv. 6	512	4×4×512	2	1	○	Leaky ReLU
Conv. 7	{256,1}	4×4×512	1	0	×	{Leaky ReLU, Sigmoid}

(a) Details of the {part encoding, discriminator} network

Layer	# Filter	Filter Size	Stride	Padding	BN	Activation Function
Conv. 1	4×4×512	1×1×256	1	0	○	ReLU
F-Conv. 2	512	4×4×{1024,1536}	2	1	○	ReLU
F-Conv. 3	512	4×4×{1024,1536}	2	1	○	ReLU
F-Conv. 4	256	4×4×{1024,1536}	2	1	○	ReLU
F-Conv. 5	128	4×4×{512,768}	2	1	○	ReLU
F-Conv. 6	64	4×4×{256,384}	2	1	○	ReLU
F-Conv. 7	{1,3}	4×4×{128,192}	2	1	×	{Sigmoid, tanh}

(b) Details of the {mask prediction, image generation} network

Table 3. Appearance losses for different inputs.

Input	Random patches	Key patches
Appearance loss	0.192	**0.0515**

2 Image Generation from Local Patches

From Figure 1 to Figure 6 show that the proposed network can generate high-resolution images either from key patches or random patches. We measure an average of the appearance loss as shown in Table 3. Although generated images from random patches are realistic, their appearance loss is larger than that of the key patch case. It is attributed to the fact that generating an image from random patches is more difficult.

3 Part Combination

In addition to the part combination results on the CelebA dataset in Figure 9 of the paper, we report results for the CelebA-HQ dataset in Figure 7 and Figure 8. The results show that the proposed algorithm generates realistic high-resolution images by combining parts of different person.

Table 4. Unsupervised feature learning results on the CIFAR10 dataset using a network trained on the CompCars dataset.

	DCGAN	Ours	
Observation	Whole image	Whole image	Part images
Accuracy	94.59%	**94.75%**	**87.15%**

Figure 9 shows generated images and masks when input patches are from different cars. It combines different styles of input patches into a new car image, e.g., in the second image at the second row of Figure 9, the generated image has a mixed color of two cars and its horizontal line at the bottom is similar to the first car. Overall, the proposed algorithm generates reasonable images despite large variations of input patches.

4 Unsupervised Feature Learning

We perform a classification task using features learned from our network. We train a network on the CompCars dataset and test on CIFAR10 for binary classification (car or not). We use the last layer of the discriminator as the feature descriptor and a linear SVM for classification. We use DCGAN as a baseline for comparison. Note that other methods, such as DCGAN, can learn features only when the whole image is presented. On the other hand, the proposed algorithm learn features from part images as shown in Table 4.

5 An Alternative Objective Function

In order to demonstrate the effectiveness of (4) in the paper, we show generation results in Figure 10 and Figure 11 using the following objective function:

$$\mathcal{L}_R(G_{\mathcal{I}}, D) = \mathbb{E}_{y \sim p_{data}(y)}[\log D(y)] + \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})}[\log(1 - D(G_{\mathcal{I}}(\mathbf{x})))]. \quad (1)$$

Both results are obtained after 25 epochs. The results show that generated images with (1) are less realistic compared to the results of (4) in the paper.

6 An Alternative Network Structure

We report results of three baseline networks as follows:

(i) Baseline 1: The proposed network without mask prediction branch,
(ii) Baseline 2: Conditional GAN based method,
(iii) Baseline 3: Auto-encoder based method.

For Baseline 1, the network is trained with an adversarial loss only. As shown in Figure 12, the generated images have low visual quality and lack diversity.

Figure 13(a) and Figure 13(b) show network structures for Baseline 2 and Baseline 3, respectively. Baseline 2 concatenates a random vector and a part encoding vector as an input for the generator. The whole network is trained using an adversarial loss. This method not only fails to preserve the appearance of input patches but also encounters a mode collapse problem as shown in Figure 14. On the other hand, the objective of Baseline 3 is minimizing a reconstruction loss. Although it contains input parts without significant modifications, generated images are blurry and unrealistic.

The results show that the proposed algorithm performs favorably against alternative approaches in terms of generating sharp images based on the appearance of inputs. In addition, ablation studies on each component of the network demonstrate motivations of each aspect of the model, e.g., inferring spatial arrangements is crucial for this task.

7 Different Number of Input Patches

In the manuscript, we show image generation results with three local patches using the proposed algorithm. We describe two different ways to take a different number of input patches. First, we simply train a new network with a different number of input patches. For example, Figure 15 and Figure 16 show generated images based on two local patches. The results show that the network can be trained with different number of input patches.

Second, we train a single network to cover various number of inputs. For the experiments, the original network structure is maintained except input nodes. Let N be a fixed number of input nodes of the network. To train the network, we first crop a set of N candidate input patches from an image. We then randomly sample N patches from the candidate set with replacement. As such, the network is trained with different number of unique patches. For evaluation, given $n < N$ patches, we randomly duplicate them to get a total number of N patches and then feed to the network. If $n = N$, then we can feed inputs directly to the network. The results in Figure 17 show that the proposed method can take different number of input patches using a single network.

8 Degraded Input Patches

As a one way of degrading inputs, we reduce the size of input patches. In the paper, the maximum area of a patch is 16% of the image size. Figure 18 describes new results when it is reduced to 9% and 4%. It shows that the proposed algorithm can generate realistic images when the input patches are small.

Figure 19 shows the results when input patches are degraded by noises. We apply the mean zero Gaussian noise at each pixel of the third input patch with the standard deviation of 0.1 (column 1-4) and 0.5 (column 5-8). The results show that the proposed algorithm is able to deal with certain amount of noise when generating realistic images.

Table 5. User study results for the first question type shown in Figure 20. Input pair denotes a set of ground truth labels of images in the question. Accuracy = # of correct answers / # of questions. R→F = # of real images labeled as fake / # of real images. F→R = # of fake images labeled as real / # of fake images.

Input pair	(Real, Real)	(Real, Fake)	(Fake, Fake)	Overall
Accuracy	43.8%	48.6%	37.2%	44.7%
R→F	34.6%	26.1%	-	30.2%
F→R	-	41.0%	44.8%	42.8%

9 User Study

We assess generated images by asking two types of questions to 130 people. The first question is to evaluate whether the generated image looks like a real image. As shown in Figure 20, it presents two images where each image is independently sampled from real or fake images at random. A question set is prepared with 6 pairs of real images, 6 pairs of fake images, and 13 pairs that are combined with a real and a fake images. Then, we ask users to pick a real image from the two images. The results are summarized in Table 5. Interestingly, less than a half of questions get correct answers on average. It shows that the proposed algorithm generates realistic images.

In addition to examine whether images are realistic, we also ask users to pick a reasonable image that looks like to be synthesized from input patches. As shown in Figure 21, we display three input patches; two of them are cropped from the same person and the rest is cropped from a different person. Then, a user is asked to pick the most likely image among five candidates. One of the candidates is a generated image based on the proposed algorithm. Other candidates are real images that are retrieved from the training set by four different baseline methods. Let I_0 and M_0 denote an original image and a mask map for the two patches from the same person. Then, the first baseline method search for the nearest neighbor from the trianing data $\mathcal{T}$ as follows:

$$\arg\min_{I \in \mathcal{T}} \|I \otimes M_0 - I_0 \otimes M_0\|_1. \tag{2}$$

The second baseline method uses ℓ_2 distance instead. Similarly, other baseline methods find nearest neighbors based on an original image and a mask map for the other patch. The results show that 85.3% of users preferred the generated image rather than real images on average. It demonstrates that the proposed algorithm generates not only realistic but also reasonable images based on the input.

10 Failure cases

Figure 22 shows failure cases of the proposed algorithm. It is difficult to generate images when detected key input patches include less informative regions (column 1 and 2) or rare cases (column 3). In addition, when input patches have conflicting information, e.g., the same nose-mouth patches that have different orientations, the proposed algorithm is not able to generate realistic images (column 4, 5, and 6). Furthermore, it becomes complicated when the inputs are low-quality patches (column 7 and 8).

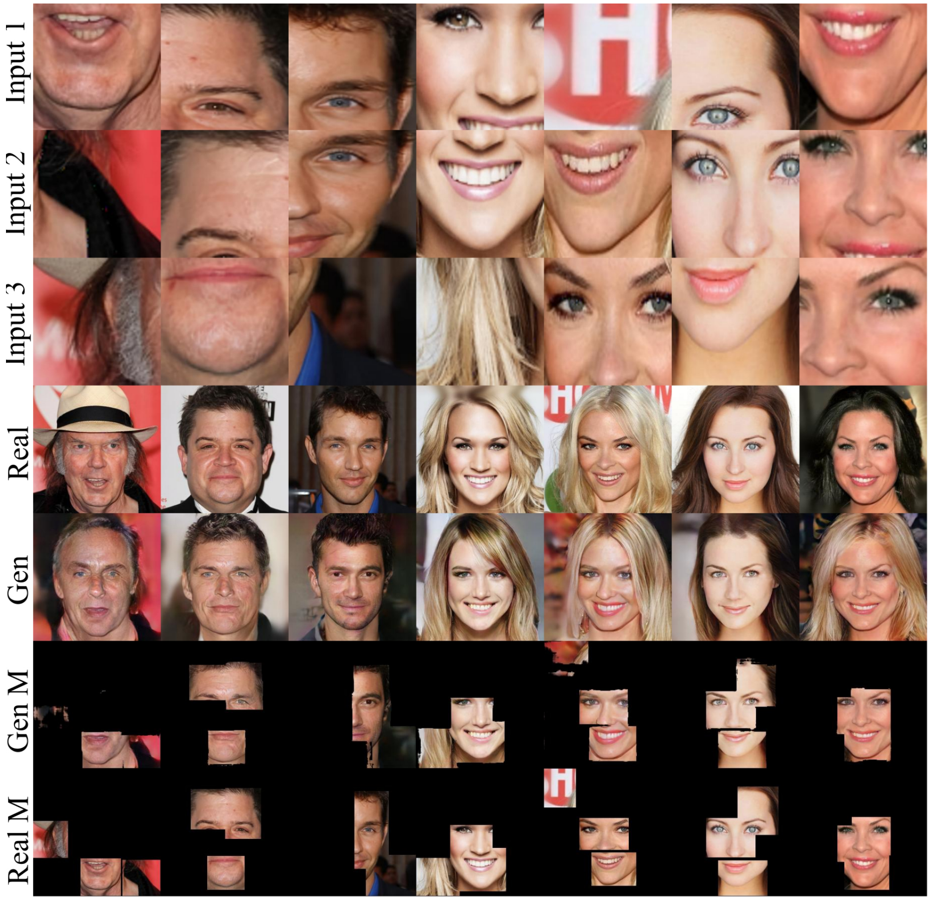


Fig. 1. Results of the proposed algorithm on the CelebA-HQ dataset. Input patches are cropped from an image (Real) based on the objectness score (Real M). Given inputs, the proposed algorithm generates the image (Gen) and mask (Gen M).

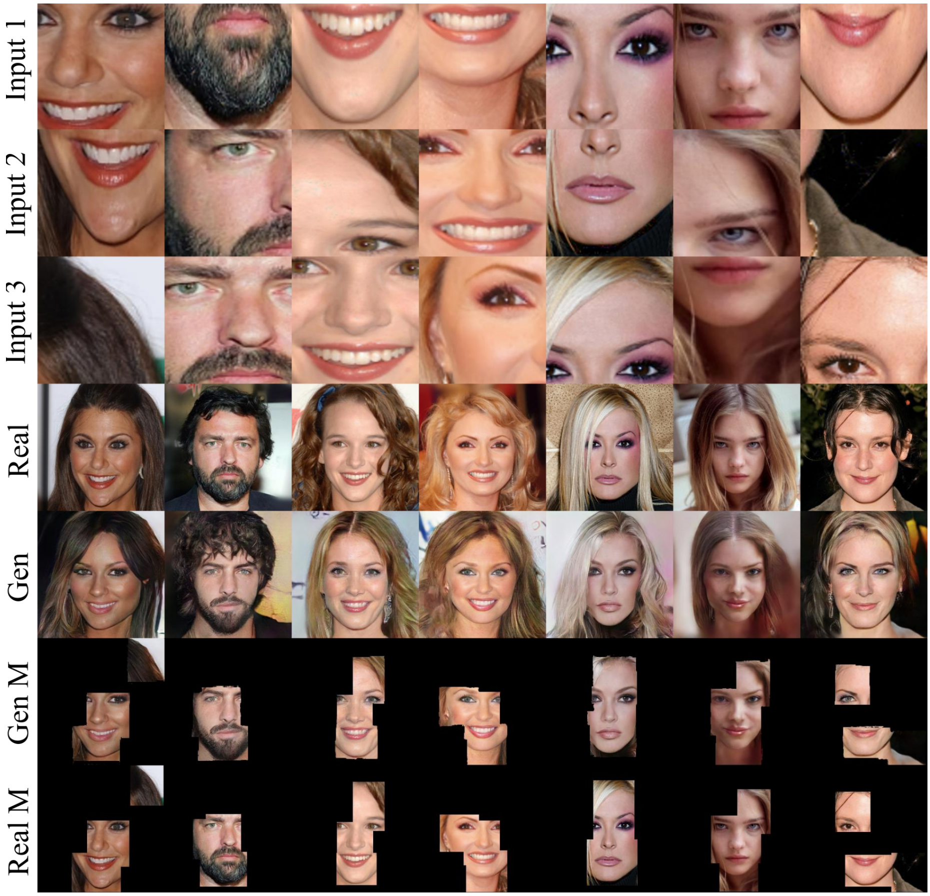


Fig. 2. Results of the proposed algorithm on the CelebA-HQ dataset.

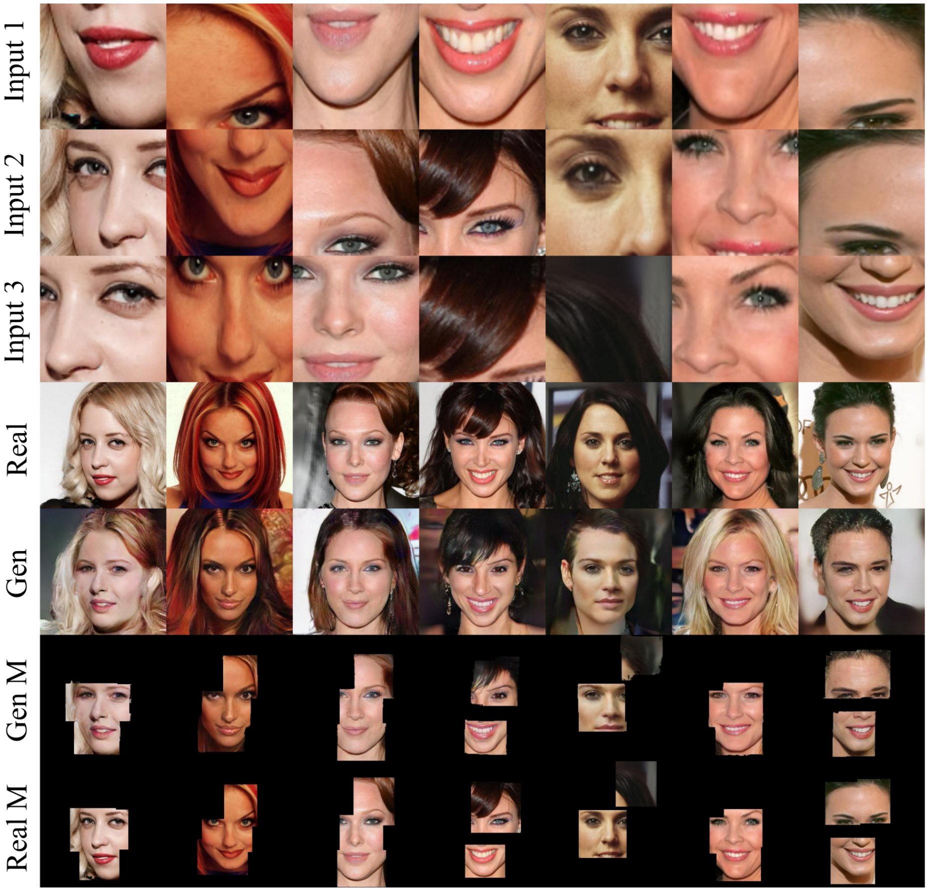


Fig. 3. Results of the proposed algorithm on the CelebA-HQ dataset.

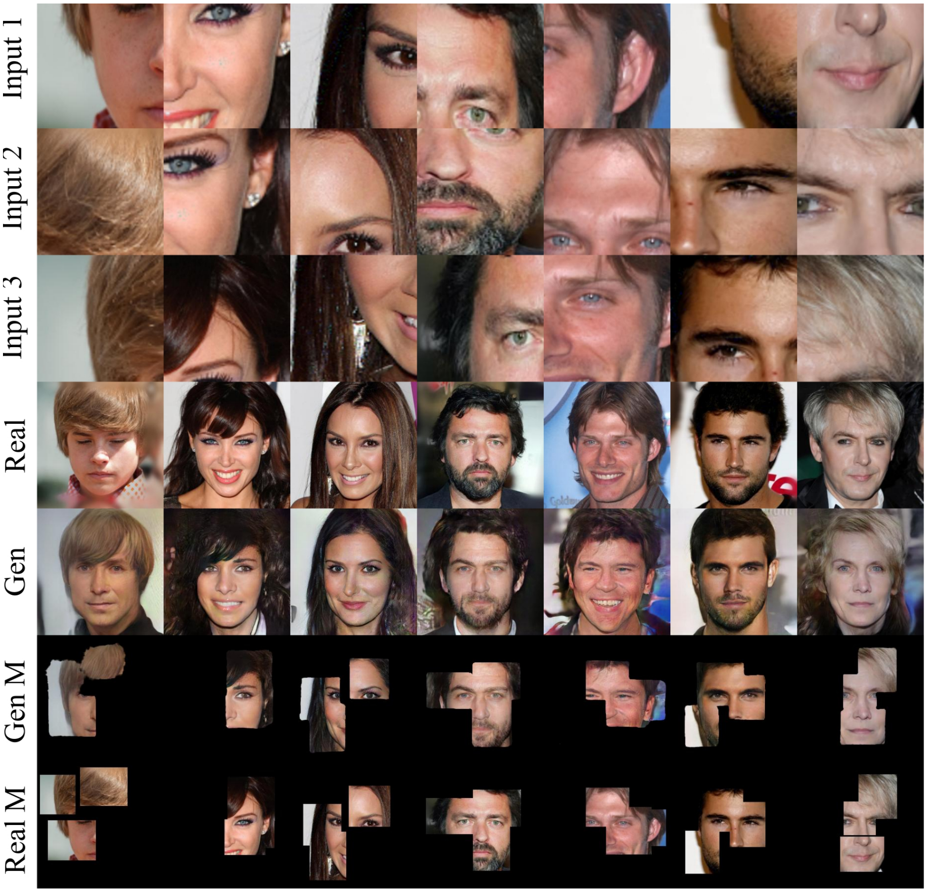


Fig. 4. Results on the CelebA-HQ dataset when inputs are random patches.

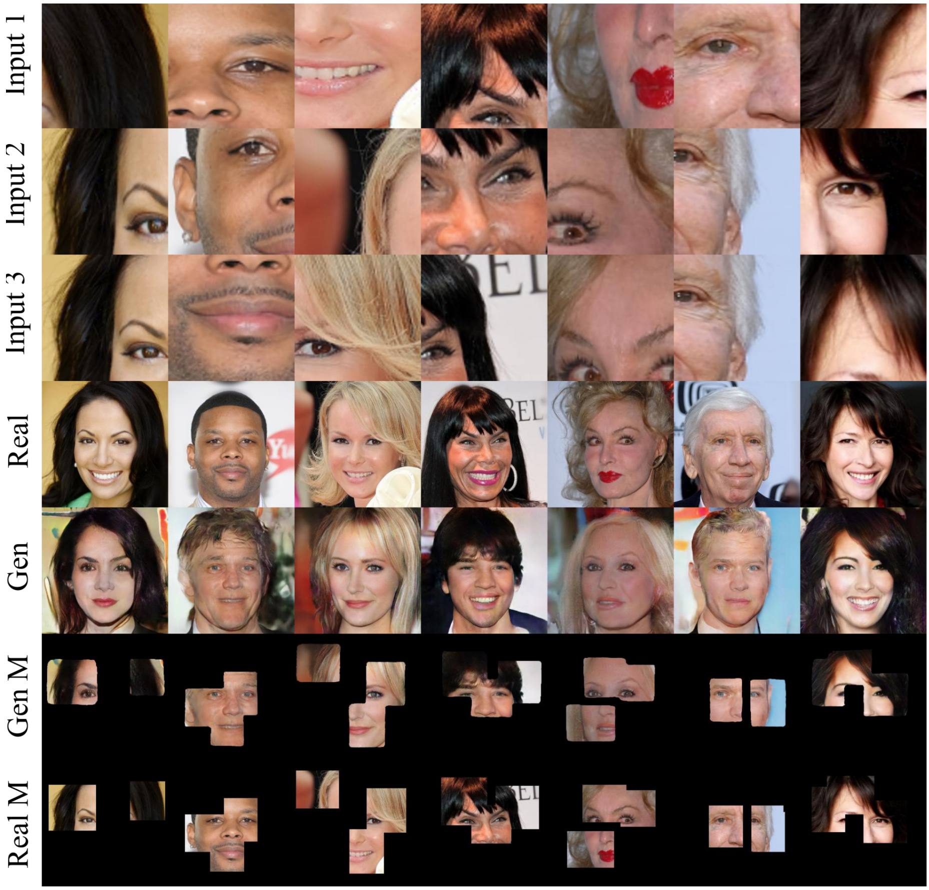


Fig. 5. Results on the CelebA-HQ dataset when inputs are random patches.

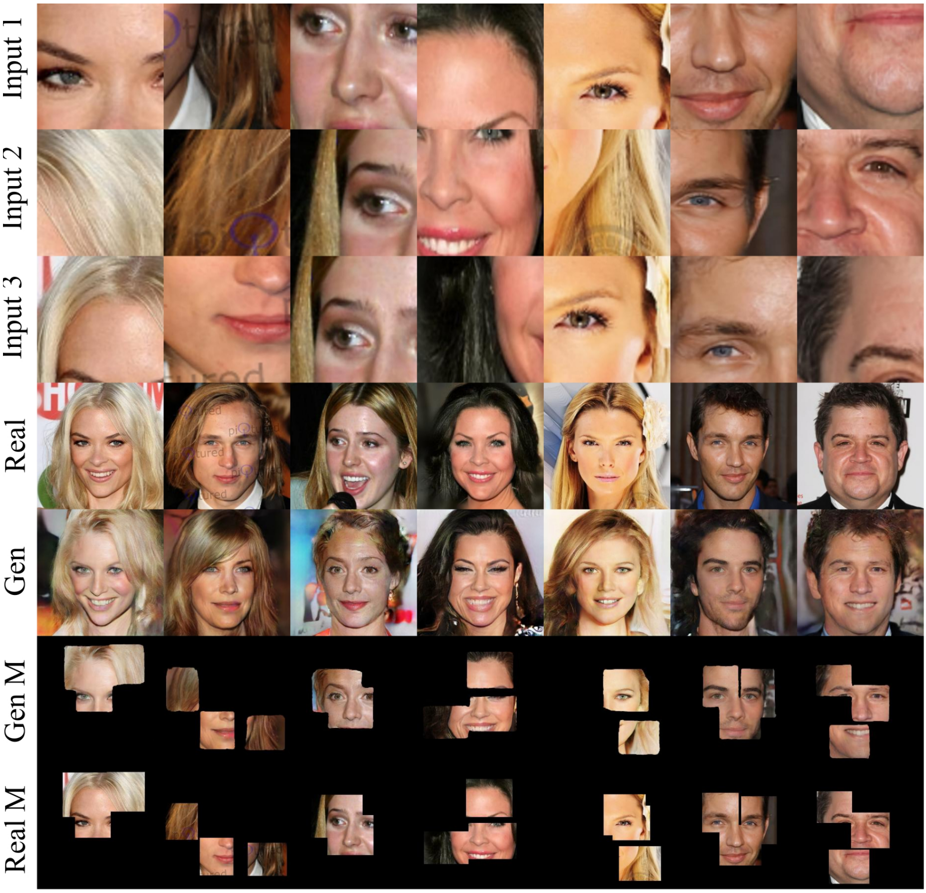


Fig. 6. Results on the CelebA-HQ dataset when inputs are random patches.

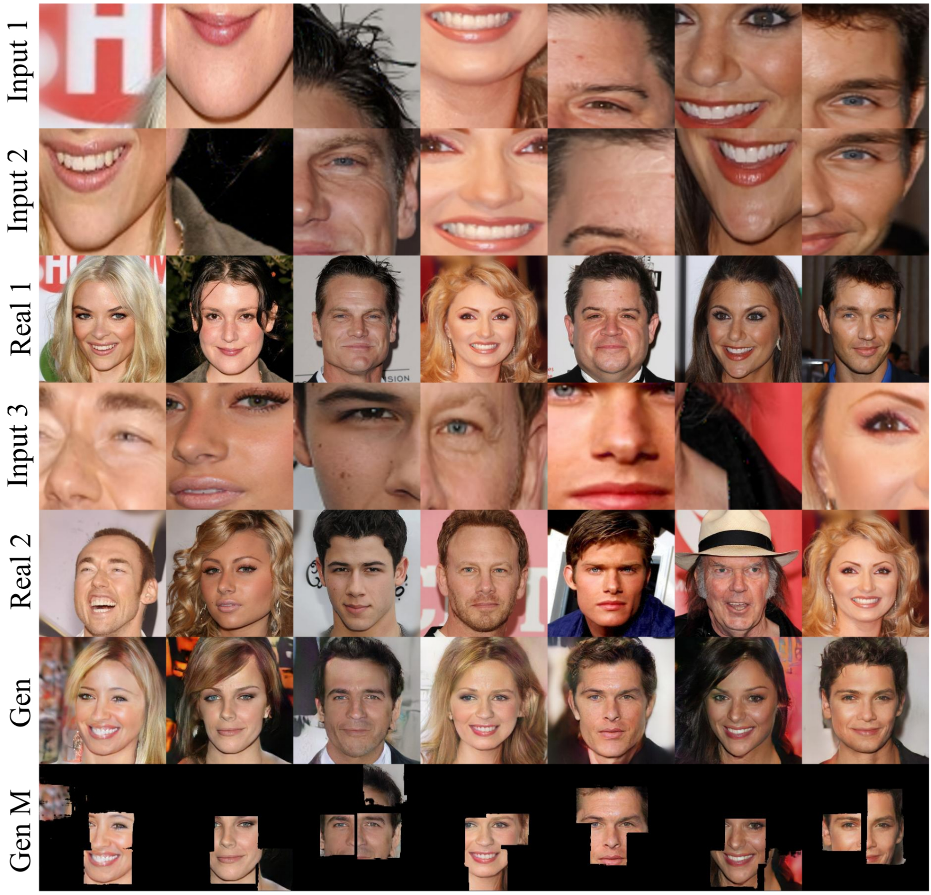


Fig. 7. Results of the proposed algorithm when input parts are combined with other people on the CelebA-HQ dataset.

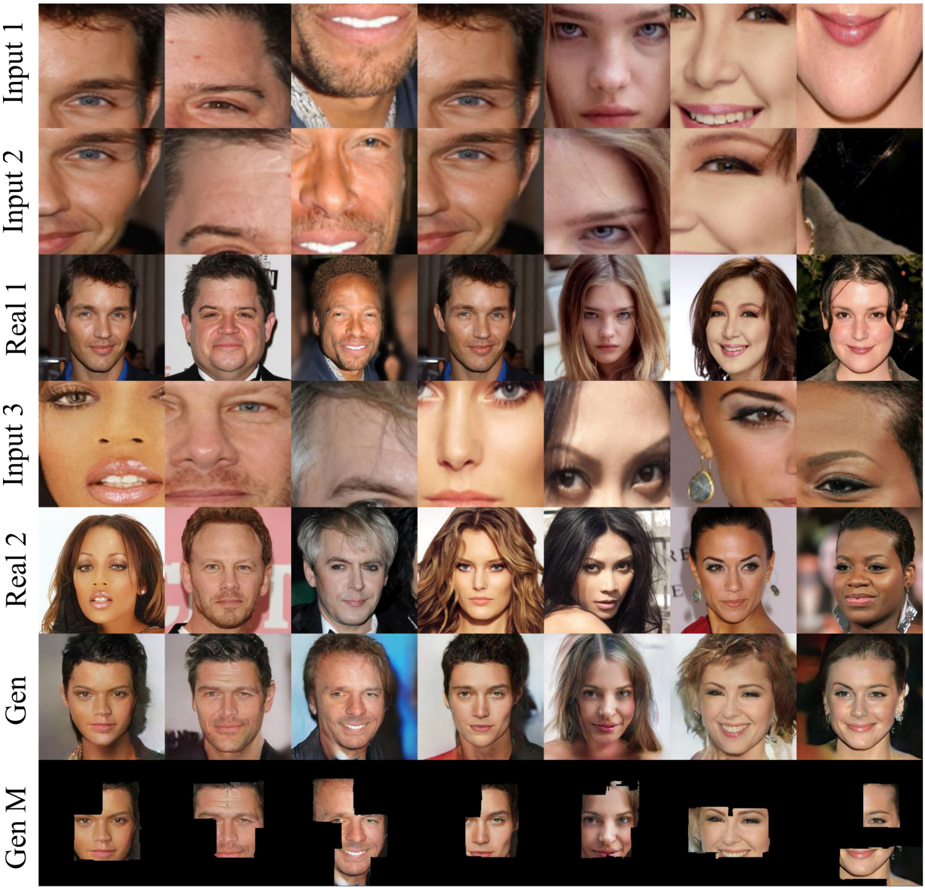


Fig. 8. Results of the proposed algorithm when input parts are combined with other people on the CelebA-HQ dataset.

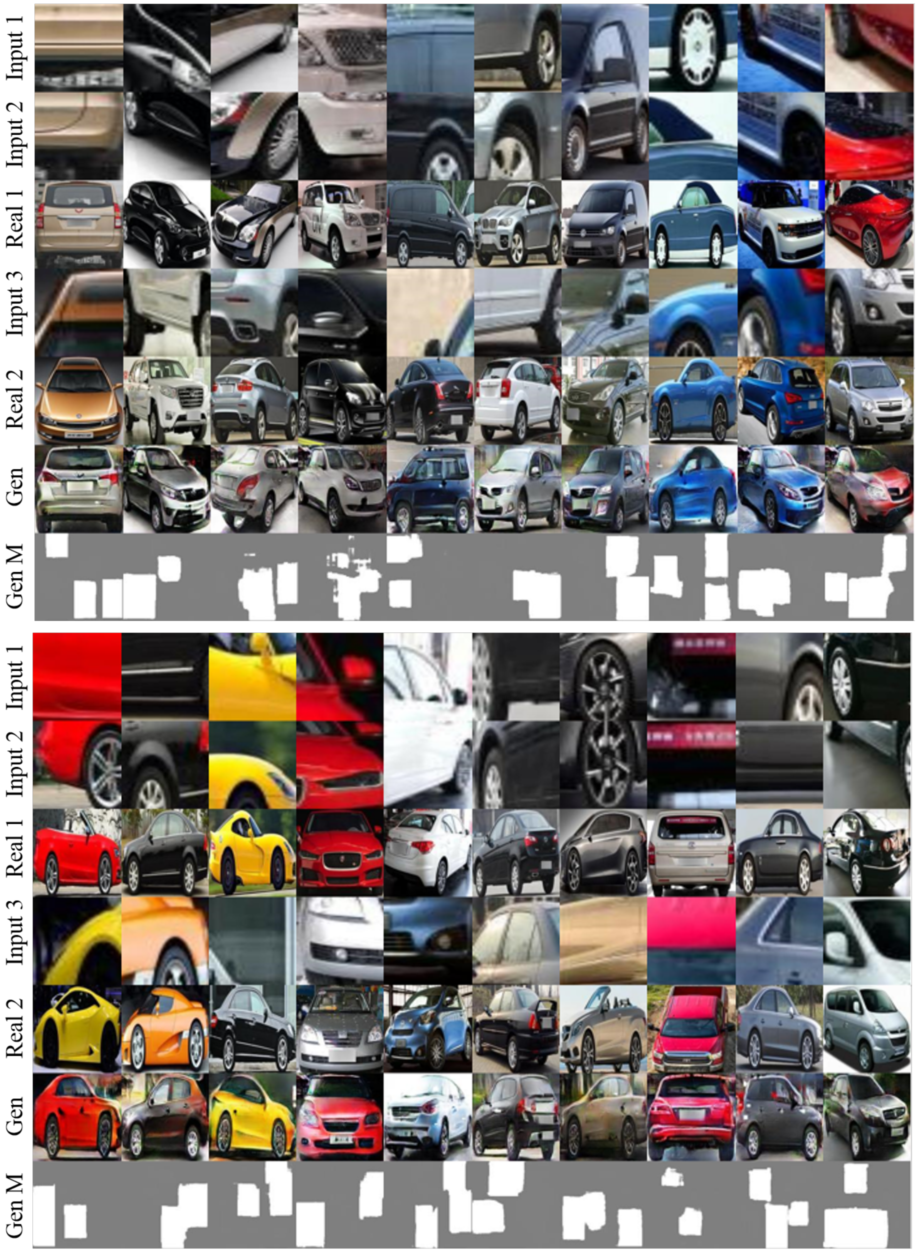


Fig. 9. Results of the proposed algorithm on the CompCars dataset when input patches are from different cars. Input 1 and Input 2 are patches from Real 1. Input 3 is a local region of Real 2. Given inputs, the proposed algorithm generates the image (Gen) and mask (Gen M). The size of the generated image is of 128 × 128 pixels.

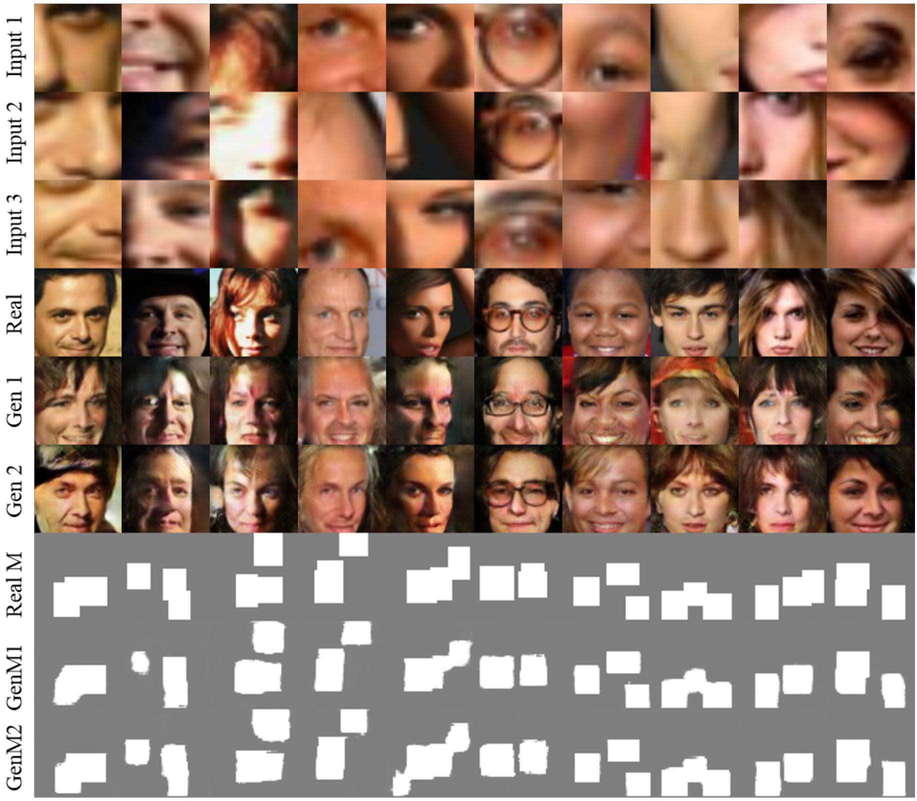


Fig. 10. Image generation results on the CelebA dataset. Gen 1 and GenM1 are generated by (1). Gen 2 and GenM2 are obtained using (4) in the paper.

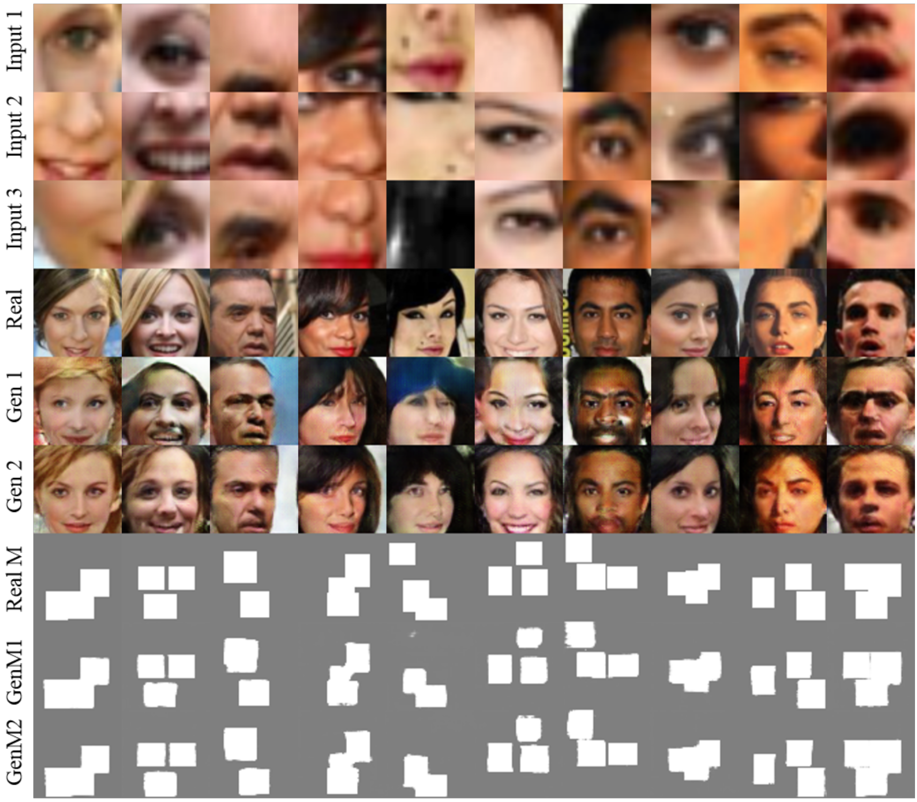


Fig. 11. Image generation results on the CelebA dataset. Gen 1 and GenM1 are generated by (1). Gen 2 and GenM2 are obtained using (4) in the paper.

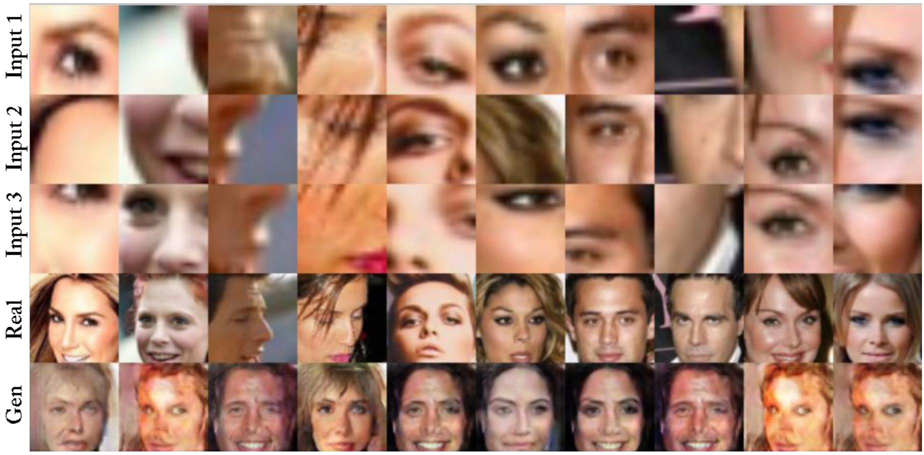
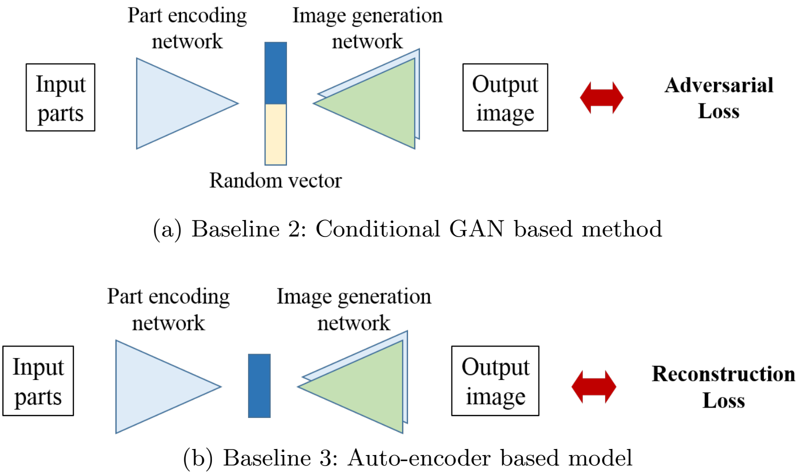


Fig. 12. Results of a baseline method (without mask prediction part from the proposed network), on the CelebA dataset.



(a) Baseline 2: Conditional GAN based method

(b) Baseline 3: Auto-encoder based model

Fig. 13. Baseline network structures. Baseline 1 is the proposed network without the mask prediction branch.

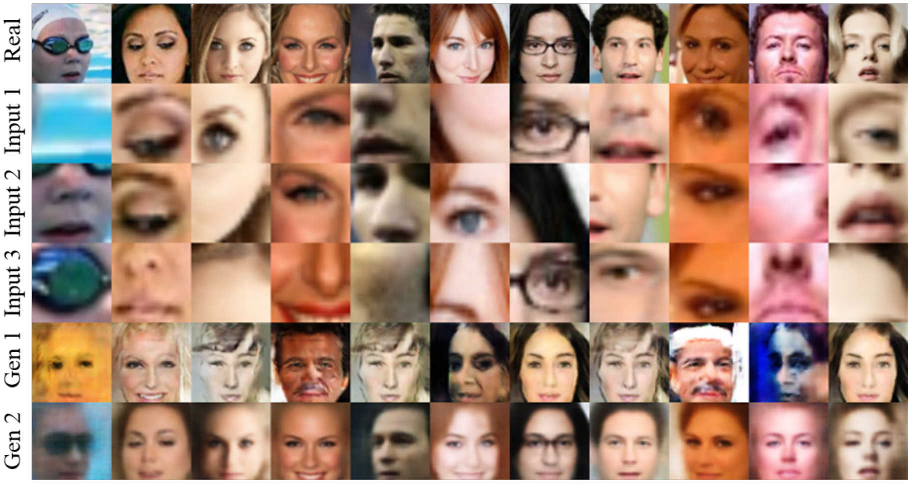


Fig. 14. Image generation results on the CelebA dataset. Gen 1 and Gen 2 are generated using networks in Figure 13(a) and Figure 13(b), respectively.

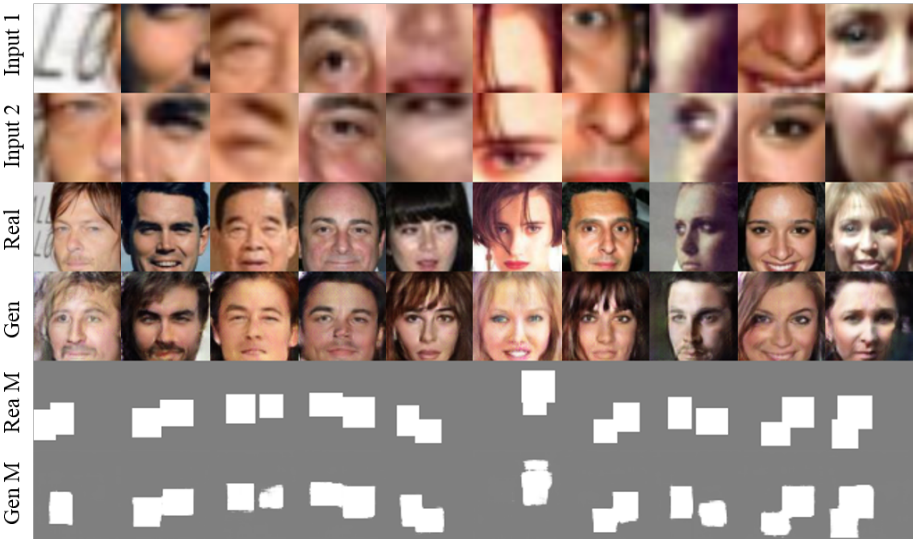
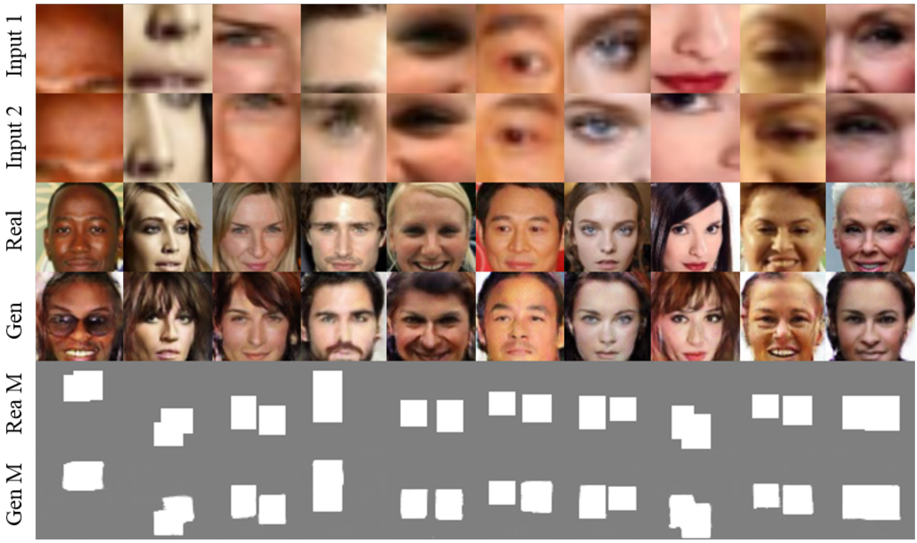


Fig. 15. Image generation results with two input patches. Input 1 and 2 are local patches from the image Real.



(a)

Fig. 16. Image generation results with two input patches. Input 1 and 2 are local patches from the image Real.

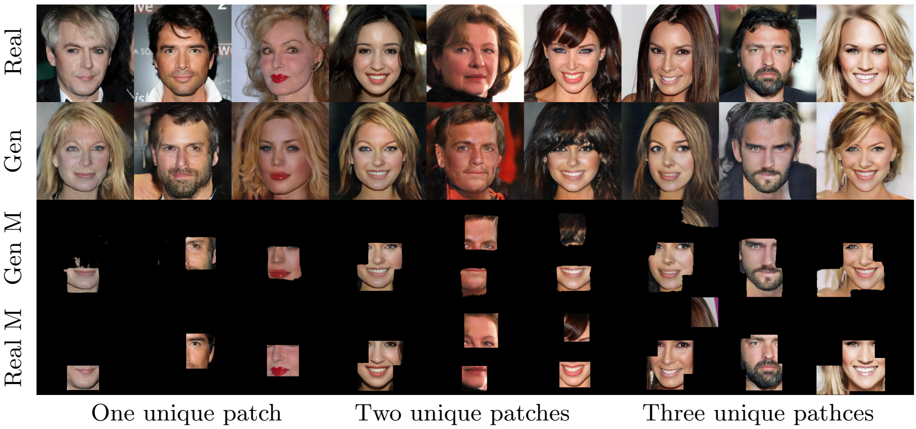


Fig. 17. Generated images with a different number of input patches. Results are obtained using a single network.

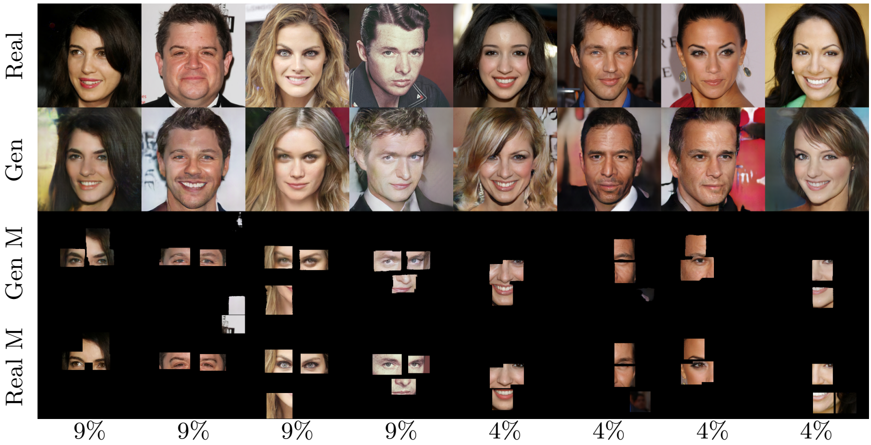


Fig. 18. Generated images based on smaller patches. A percentage below each column indicates the maximum area ratio (patch size/image size × 100) for each input patch.

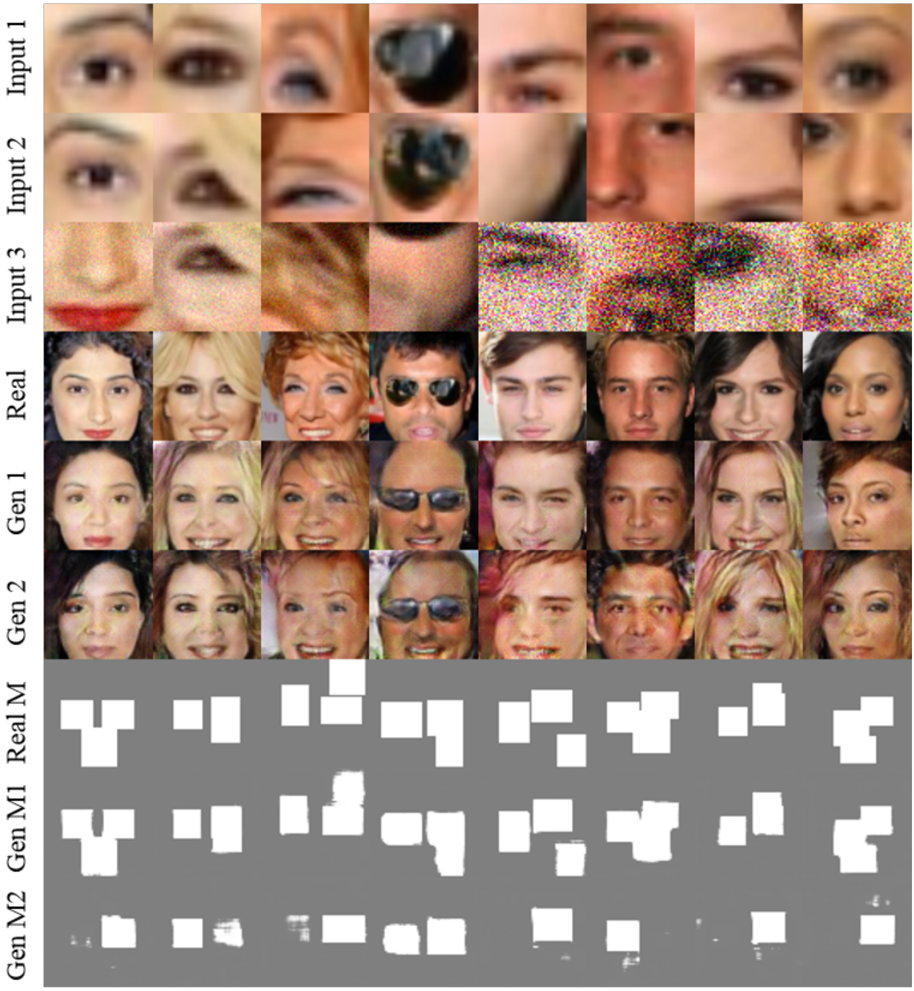
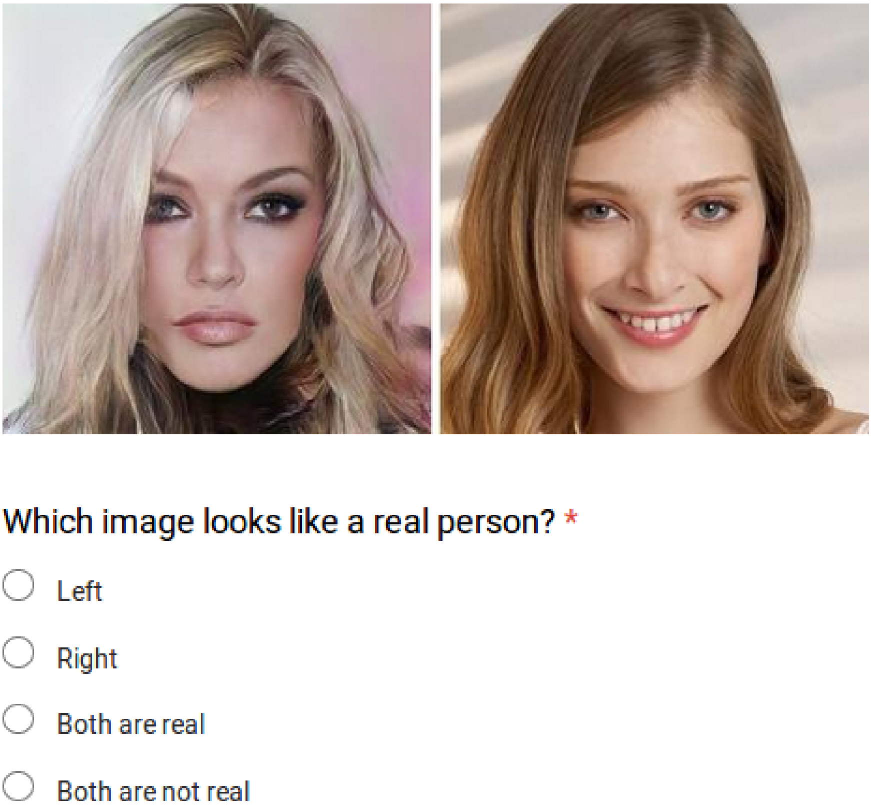


Fig. 19. Examples of generated results when the input image contains noises. We add a Gaussian noise at each pixel of Input 3. Gen 1 and Gen M1 are generated without noises. Gen 2 and Gen M2 are generated with noises.



Which image looks like a real person? *

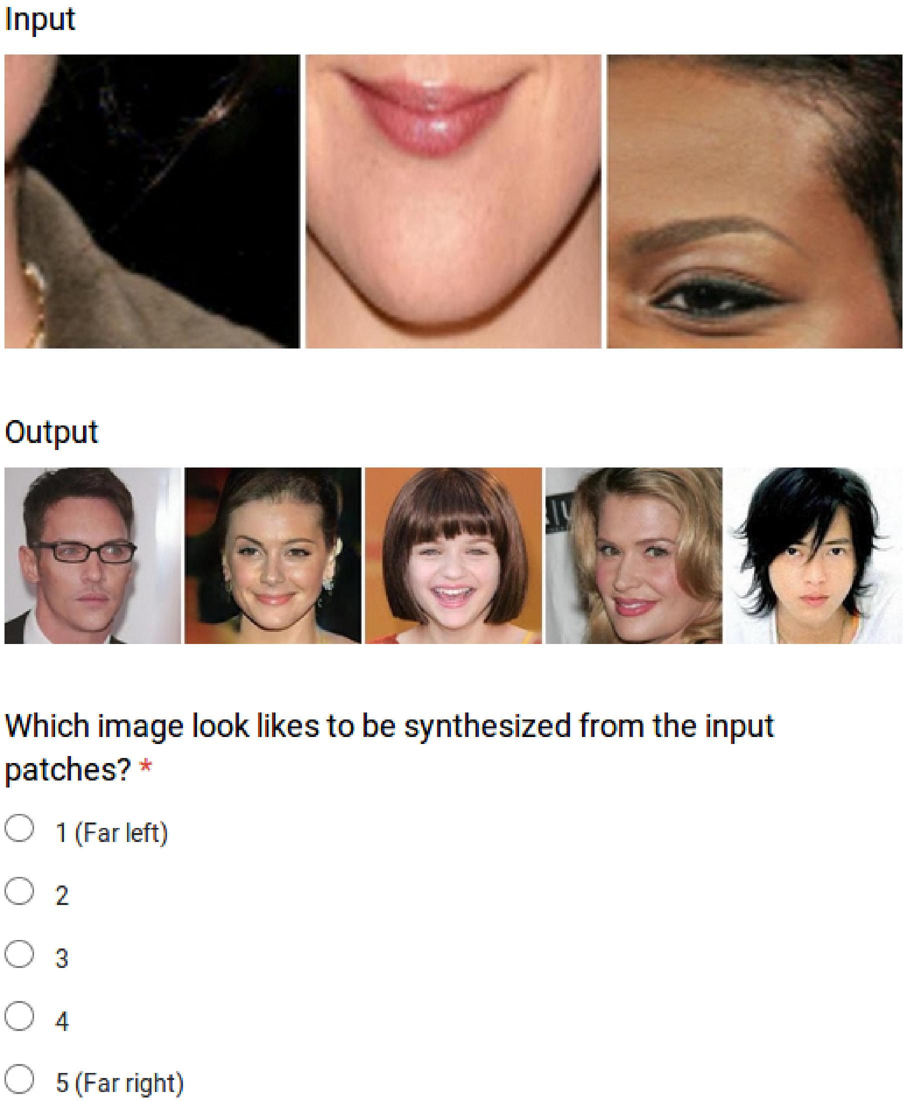
○ Left

○ Right

○ Both are real

○ Both are not real

Fig. 20. An example of the first question type for the user study.



Input

Output

Which image look likes to be synthesized from the input patches? *

○ 1 (Far left)

○ 2

○ 3

○ 4

○ 5 (Far right)

Fig. 21. An example of the second question type for the user study.

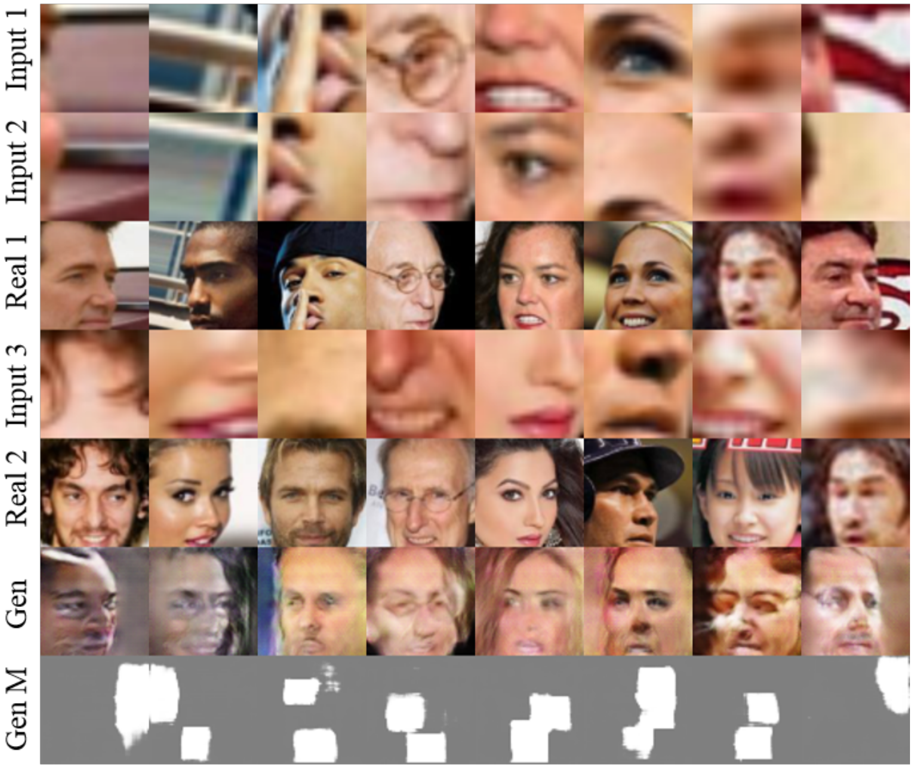


Fig. 22. Examples of failure cases of the proposed algorithm.