Iterative Facial Image Inpainting using Cyclic Reverse Generator

Yahya Dogan  
Ankara University  
Computer Engineering Department  
yahyadogan@ankara.edu.tr

Hacer Yalim Keles  
Ankara University  
Computer Engineering Department  
hkeles@ankara.edu.tr

Abstract

Facial image inpainting is a challenging problem as it requires generating new pixels that include semantic information for masked key components in a face, e.g., eyes and nose. Recently, remarkable methods have been proposed in this field. Most of these approaches use encoder-decoder architectures and have different limitations such as allowing unique results for a given image and a particular mask. Alternatively, some approaches generate promising results using different masks with generator networks. However, these approaches are optimization-based and usually require quite a number of iterations. In this paper, we propose an efficient solution to the facial image painting problem using the Cyclic Reverse Generator (CRG) architecture, which provides an encoder-generator model. We use the encoder to embed a given image to the generator space and incrementally inpaint the masked regions until a plausible image is generated; a discriminator network is utilized to assess the generated images during the iterations. We empirically observed that only a few iterations are sufficient to generate realistic images with the proposed model. After the generation process, for the post processing, we utilize a Unet model that we trained specifically for this task to remedy the artifacts close to the mask boundaries. Our method allows applying sketch-based inpaintings, using variety of mask types, and producing multiple and diverse results. We qualitatively compared our method with the state-of-the-art models and observed that our method can compete with the other models in all mask types; it is particularly better in images where larger masks are utilized.

Keywords — Facial image inpainting, image completion, generative adversarial networks, deep learning, convolutional neural networks.

1 Introduction

The purpose in image inpainting is to fill missing or masked regions with perceptually plausible content. This is useful for repairing damaged images, removing distracting objects/parts or synthesizing alternative contents in masked regions. It is a long-standing topic but has recently been addressed by many researchers thanks to the advances in deep learning based techniques, especially the generative models. Image inpainting models can be divided into two main groups; non-semantic and semantic methods. The first group, which is also referred to as the traditional approaches, searches for patches from known regions of the same image or a database of images and replaces masked region with similar content that maximize the patch similarity [1–4]. These approaches produce texture-consistent results in completing the background inpainting problem (e.g., sky, sea, and grass), but often fail to complete complex scenes that require particular objects/parts, e.g., faces and missing parts of objects; it is hard to find patches that match exactly with a particular context. The second group, which is named as learning-based image inpainting methods, [5–13], can capture high-level semantics such as position, shape, color, and texture of objects, and fill masked regions with coherent structures using learned data distribution. Facial image inpainting approaches are in this group; they require semantic content

*Note that this paper is under consideration at Neural Computing and Applications Journal.
Fig. 1: Left to Right, on both columns: Input image, different mask types (regular, irregular or hand-drawn), masked image, course CRG result, fine result after post-processing. Right-bottom block shows the results of each cycle for the sample in (k) above.

generation. For instance, if the missing part of an image is in the nose region, it is difficult to synthesize a nose fitting naturally to a face using image patches from other facial parts, if not impossible.

More recently, deep learning-based approaches show remarkable success in the task of filling in visually realistic content for masked regions. However, different approaches have different limitations or constraints in region filling: (1) Many methods are indifferent to the mask content, i.e. the sketch or pixel colors in the mask. These methods either ignore the assigned pixel value to the masked region or use the same pixel value similar to the training phase [14] [7]. Therefore, the output remains the same even though the content of the masked region is different. (2) Some methods use rectangular masks and may not perform well for irregular shaped masks [8] [7], (3) Some models provide multiple and diverse results for masked regions, yet these models may require sketch/edge [10, 15, 16] or labels [17] as a condition during model training.

In the vast literature, most of the parametric models are based on encoder-decoder architecture; these networks learn how to fill the mask regions in a supervised manner end-to-end. The mapping is unique and constant for a particular image and the mask. There are recent approaches that show the utilization of a generator network for different promising results with varying masks [18, 19]. Our approach is similar to these works in that we also utilize a generator network for filling the content of the mask based on the surrounding context of the face image. Differently from them, however, we use an encoder to efficiently embed a given image to generator space instead of iterative latent vector optimization. This is very advantageous, since we incrementally generate plausible image after only a few iterations, i.e. 10 on the average, while optimization based embedding requires usually a few thousands of iterations.
In this context, we utilize the CRG architecture that we introduced in our preliminary work [20] to embed a given image to a generator latent space using the encoder network. CRG contains a generator and an encoder network, which is trained in an unsupervised manner to learn the inverse of the generator. The generator is designed as a slightly modified version of the Progressive GAN architecture and trained using 30K face images from CelebA dataset [21]. Our solution to image inpainting problem depends on the observation that the encoder embeds a given image to a latent point in the generator space using the context of the image, even some parts of the image is masked. Moreover, the content of the mask directly influences the embedded point, hence the generated image changes accordingly. Although the generated images in the mask region contains distorted, blurry version of some face features in the early steps, when we use the generated content iteratively, the face features gradually improve in each iteration. This incremental approach generates a variety of alternative fillings in the mask regions depending on the initial mask content; since each time the changed image content results in an embedding to a different latent point. As a result, iteratively embedding to a better latent code and generating better image in the generator space enables us at some point to reach at a plausible image content. Our heuristic works successfully by the help of a discriminator, which is used to assess the quality of the generated images in each iteration.

The proposed method in this work has a two-stage process. In the first stage, the CRG models and a discriminator is used to generate a plausible image for a masked input image. In the second stage, inspired from [14, 22], a UNet model is trained to remedy the visual artifacts within the mask such as the artifacts around the boundary pixels, skin color variations etc. We preferred using a deep network in the second stage to avoid hand coded post-processing algorithms, e.g., Poisson image blending [23], fast marching methods [24].

The main contributions of this work can be summarized as follows:

1. We propose an iterative facial image inpainting method that allows regular/irregular/hand-drawn masks as input.

2. Our method allows different results depending on assigning pixel values to a masked region. To the best of our knowledge, we are the first to demonstrate this.

3. Contrary to other encoder-decoder models, our method enables image generation from latent space and allows modifying undesirable regions in the generated images.

4. Our model provides sketch-based inpainting, which allows multiple and diverse results for the same masked region, without requiring conditional training.

The rest of the paper is organized as follows. A brief summary of the related previous works are summarized in Section 2. The proposed method is discussed in detail in Section 3. The experimental results are provided in Section 4. The paper is concluded with future directions.

2 Related Works

Thanks to the advances in the deep learning models, recent works in the facial image inpainting domain provide plausible semantic content in images. Most of the recent works in the domain focus on designing a better encoder-decoder architecture to fill the missing parts in images. The encoder embeds an image with a missing region to a low-dimensional latent feature space, while the decoder takes the embedded latent code and reconstructs the original image. Autoencoder models are usually trained with pixel-wise reconstruction loss, which commonly causes blurry images. Therefore, recently proposed models use a joint objective function that combines reconstruction loss with the adversarial loss that is used in GAN training [25]. Context Encoder (CE) is one of the earliest works that introduces adversarial loss into the inpainting problem. CE repairs large holes using a deep convolutional neural network via learning semantic priors and meaningful hidden representations. The CE approach can produce new semantic content in the masked regions, but still it can produce inconsistent content close to the boundaries and is limited to work with fixed size masks. Iizuka et al. [8] extend the CE by
introducing both global and local discriminators in the adversarial loss computations, which improve visual quality significantly. Moreover, their method support different size masks since they use a fully convolutional network. Li et al. [6] used a pre-trained face parsing network to compute another loss, in addition to the global, local, and pixel-wise reconstruction losses, to enforce a more plausible and consistent result. This approach handles aligned faces well, but may require post-processing methods (such as Poisson blending [23]) to provide color consistency. Yu et al. [7] claim that CNN-based models are ineffective in terms of modeling long-term correlations between distant contextual information and a corrupted region. This may lead to various structural artifacts between a corrupted region and its surrounding areas. They proposed a novel contextual attention layer to capture relationship between feature patches at distant spatial locations. Recently, Wang et al. [9] stated that this mechanism is extremely addictive to surrounding information and may not remove all artifacts in a missing region of an image. To overcome this problem, they designed a deep network consisting of 3 parts, i.e. representation learning, feature reconstruction, and residual learning. The first part aims to capture image content and compresses it into a latent feature representation. The second part has a Laplacian pyramid to recover missing regions of a face image in a coarse-to-fine fashion, progressively. The third part is a convolutional residual learning architecture that is used to eliminate color discrepancies between the corrupted area and its surroundings. Their results are realistic and natural-looking regardless of the mask locations.

The aforementioned models are designed to work with rectangular masks. In real life scenarios, when inpainting in a slightly corrupted image with irregular patterns, there is a need for more flexible mask definitions. Recently, several methods have been proposed to support free-form masks to handle irregular image inpaintings. Liu et al. [14] introduced a partial convolution layer where the convolution is masked and renormalized using only the valid pixels; they then apply a mask-update. Their model produces good results for irregular masks, yet their method ignores the pixels in the masked region. Yu et al. [15] introduced gated convolution to generalize partial convolution by learning a dynamic feature gating mechanism and also added an extension to allow user-guided inpainting. Nazeri et al. [10] developed a new approach, which is named as Edge Connect, to provide hand-drawn and sketch-based inpainting using standard convolution operation. Their model consists of an edge generator and an image completion network, both followed by an adversarial network. The aim of the edge generator is to detect the edges of an unknown region of an image; image completion fills the missing region in the image using detected edges as a condition. The Edge Connect method recovers images with good semantic structural consistency, but an edge artifact may occur at the boundary of completion. To handle this problem, Hong et al. [13] proposed a a smooth transition named Deep Fusion Network (DFNet). They embed a few fusion blocks into different decoder layers in the Unet model to generate a flexible alpha composition map; it is similar to the hole mask but has smoother weights, especially on the boundary region.

The mentioned models for image inpainting can produce plausible results for regular, irregular, or hand-drawn masks, but these models produce only one result for a single masked input. Recently, some image inpainting methods [12, 17] generate multiple and diverse plausible solutions for a single mask. Chen et al. [17] proposed a completely end-to-end progressive GAN model [26] to repair face images in high-resolution, and they designed a conditional version to provide multiple results with controllable attributes, e.g., gender, smiling, etc., under arbitrary masks. Zheng et al. [12] presented a probabilistic framework with two parallel paths to generate diverse solutions for image completion, and introduce a short+long term attention layer that exploits distant relations among decoder and encoder features to improve realism. These approaches allow producing different results that are useful in facial image inpainting, e.g., if the masked region is a mouth, both smiling and non-smiling results can be produced.

There are very few works [18, 19] that allow both image generation and filling a masked region in an image. In Yeh et al. [18]’s work, semantic inpainting is considered as a constrained image generation problem. Utilizing a GAN model [25], they search for an encoding of the masked image that is closest to the image in the latent space. In [18], an optimization-based method is proposed to find the image closest to the original image using the rest of the image, i.e. masked part of the image is not included in the search. Abdal et al. [19] proposed an embedding algorithm into the W+
space of StyleGAN [27] instead of the initial Z latent space. Their method provides multiple image editing and manipulation applications, such as image inpainting, and makes the optimization-based approach interesting. However, these methods require many iterations (e.g., 1500) and a different set of hyper-parameters for each image to achieve good results.

In our work, we utilize the CRG framework that we developed in our preliminary work [20] to provide an efficient solution to the facial image inpainting problem via an encoder-generator architecture. Unlike the other works that have a generator network, we proposed an iterative algorithm to fill a masked region semantically with very few iterations using our encoder. Our method is flexible; it allows applying sketch-based inpaintings, using different mask types, and producing multiple and diverse results depending on an assigned pixel value to a masked region.

3 The Method

In this section, we introduce our two-stage coarse-to-fine approach for facial image inpainting. The first stage makes an initial coarse prediction using our preliminary encoder-generator architecture, i.e. the CRG, which allows learning the inverse function of the generator accurately via an encoder in an unsupervised setting [20]. In this part, we establish a cycle between the encoder and the generator and score the generated image at the end of each cycle using a discriminator network (Section 3.2). The CRG cycle is run until a plausible image content is generated (usually less than 10 times), and finally, the image with the highest discriminator score is selected as the coarse prediction. In general, the score is low in the first cycle, the score increases as the cycle progresses, and after a certain number of iterations, the score generated by the discriminator starts decreasing. However, sometimes the score may fluctuate up and down; therefore, after running 10 iterations, we select the image with the highest score as the coarse prediction.

The second stage of our approach can be considered as a postprocessing stage. Our aim is to improve and eliminate some minor artifacts of the resultant coarse image in and around the boundaries of the masked regions. The second stage takes the generated coarse image as input, and predicts a refined result using a Unet type deep network (Section 3.3). The aim of using a network in this stage is to avoid handcrafted post-processing operations. We trained a Unet model that we adapted from [14,22] from scratch. The overall design of the proposed model is depicted in Fig. 2.

3.1 The CRG Architecture

In our preliminary work [20], we proposed the CRG architecture to find an accurate latent vector representation of an image in the generator space. CRG is composed of an encoder and a generator network; the encoder maps the image to a lower dimension space, i.e. latent vector representation, and the generator maps a given latent code to image space. This allows reconstruction of both generated and real images. In [20], we used CRG for editing desired face attributes using two reference images, one for a neutral face image and the other is the image with the desired attribute, i.e. smiling etc. We compute the latent unit vector direction using the encoded latent vectors of these reference images and edit any image using the computed direction. Using this attribute direction, the CRG enables manipulation of only the desired attributes at any rate keeping the other attributes unchanged.

In this work, we use the CRG encoder-generator models for filling the masked region in an image. The coarse prediction works as follows: First, the masked image is given to the CRG encoder network to compute its latent vector representation. Since masked image contains some missing regions in the masked area, the CRG encoder maps it to a point in the latent space utilizing the context in the unmasked regions; the mask content also provides additional context. Its latent code is initially away from the one that belong to the original unmasked image in the generator latent space. Therefore, the generated image as a result of the first cycle only contains a very coarse content in the masked regions and looks quite different from the original one. Still, it’s a step further towards a realistic looking image than the initial mask. Hence, we copy the estimated mask content to the initial image to initiate
another cycle. Formally, we use (1) to prepare the input for the next cycle.

\[
I_0 = M \odot I_{org} \\
I_{i+1} = M \odot I_{org} + (1 - M) \odot I_i
\]  \hspace{1cm} (1)

Where \(I_{org}\) is the original image, \(M\) is the binary mask (0 for masked region), \(I_i\) is the output of the previous cycle and \(I_{i+1}\) is the input of next cycle. \(I_0\), which is the masked image, represents the input of the model in the first cycle. This iterative process is performed until a plausible image is obtained; more reasonable content is incrementally generated at each step in the masked area, after only a few iterations, i.e. usually less than 10.

Another important point is deciding how many iterations we need to perform and which cycle result will be used as the coarse prediction. To automatize this process, we established a discriminator network that assesses the quality of the generated images and produce scores between 0 and 1, where score 1 indicates high image quality. We provide the generated image at the end of each cycle to the discriminator and go on this process as long as image scores continue ascending. As a precaution for small fluctuations in the scores, based on our empirical observations, we set the number of iterations to be at least 10 and choose the generated image with the highest score, if it stopped increasing in this interval. Our proposed method has these advantages: (1) Arbitrarily structured masked regions can be used during the inference of our generator; CRG model is a generic framework that does not need any supervision depending on the masks, (2) Multiple and diverse results can be generated since masked image could be embedded to a different point in the generator latent space depending on the
pixel values in the masked regions, and (3) Sketch-based inpainting is possible since the masked image with sketch would potentially be embedded to a different related point in the generator space, e.g. a mask put around the eye region with roughly sketched eyeglasses in the mask image, generates images with eyeglasses.

In Fig. 2, we first generate an input image randomly using the CRG generator. For the relevant example, we wanted to generate a different hairstyle and background keeping the face unchanged. In this context, we created a hand-drawn mask to keep the face part unchanged and the region around the face to be filled. Using the mask, we obtained the first input, i.e. masked image, of the CRG encoder, by applying the created mask to the image (Eqn. 1). The pixel values corresponding to the masked region in the original image at the end of each cycle are replaced with the corresponding pixels in the generated image of that cycle. In this example, we iterate this process 10 times (n in Fig. 2 is 10) and we use the discriminator to visually decide on the best image, i.e. the image with the highest discriminator score. Depending on the capacity of the generator, the generated images usually converge to images that are similar to the original image (for the unmasked region) as the cycle progresses. However, small variations in the pose or color in the converged image may cause very small, yet noticeable, boundary artifacts when the masked region pixels are copied into the original image. We use Unet model to correct those artifacts as a post processing.

In our method, depending on the initial pixel values assigned to the masked regions, image can be embedded to a different point in the latent space, which is useful since this allows different alternative image generations. In Fig 3, we masked the 128x128 facial image to cover the eyes and assigned the average pixel values in the unmasked image region. Above each image, we provide the discriminator score of the image obtained at the end of that cycle. Scores vary between 0 and 1, and increases as the algorithm iterates using the generated image in the previous cycle. We iterate the CRG model 10 times and get the image with the highest score of 0.909 in the 8th cycle as a coarse prediction. Note that the coarse result is without eyeglasses.

In the next example, we assigned random pixel values with zero mean to the masked region of the same face. When the initial mask content is changed this way, the encoder embeds the masked image to a different latent point in the generator space, which results in different content at the end. In Fig. 4, the masked region is filled with eyeglasses, differently from the final image shown in Fig. 3. To the best of our knowledge, this feature in image inpainting is novel; almost all the methods either ignore the assigned pixel value in the masked area or use the same/fixed value assignment regime to the mask content that is used during the training phase.

In the proposed method, there is no restriction in the shape of the masks. As a show case, we depicted a sample inpainting in Fig 5 using an arbitrarily shaped, irregular mask for 5 iterations. The resultant image is displayed in the upper right part of the Figure. Note that the generated image converges to the original image for this sample, and the masked content incrementally becomes more reasonable.
Fig. 4: Coarse CRG result obtained when random pixel values are assigned to the masked region.

Fig. 5: Coarse CRG result obtained using a random irregular mask.

3.2 The Discriminator Network

In this section, we provide the details for the proposed discriminator network. In our method, it is important to determine which image looks visually realistic. For this purpose, we established a multi-layer convolutional neural network ending with a sigmoid unit. The structure of the discriminator is depicted in Fig. 6.

In general, a difference in brightness, blurriness, and contrast occurs between the masked region and the rest of the image in inpainting works. Our aim is to establish a model that will generate a score taking these differences into account. For this purpose, firstly, we created a dataset by generating regular and irregular masks and adding the visual artifacts we mentioned earlier to the masked regions to a certain amount. We applied artifacts by generating random values in the related ranges for blurriness [1.0 – 2.5], brightness [0.4 – 0.8], and contrast [0.4 – 0.8], where a high value in blurriness and a low value in others increases the artifact ratio. Then, we labeled real images as 1 and images with the visual artifacts as 0. In the training phase, we used adversarial loss based on GAN, which is defined as:
\[
\max_D \mathbb{E}_{x \sim p_{\text{data}}(x)} \log (D(x)) \ + \ \mathbb{E}_{v \sim p_{\text{VisArt}}(v)} \log (1 - D(v))
\]  

(2)

Where \(x\) and \(v\) represents a real image and an image with visual artifacts, respectively. We try to maximize the discriminator \(D\) to distinguish real data, i.e. \(p_{\text{data}}\), from the data with visual artifacts, i.e. \(p_{\text{VisArt}}\). As the given task to the discriminator is simple, it converges quickly and produces only 0 and 1 scores for the input images, which is not useful for our purposes. We want to get gradual scores which reflect the visual quality of the images. Therefore, we changed our training policy to overcome this problem: First, we labeled the real images as 0.9 (instead of 1) and the distorted images as 0.1 (instead of 0). Second, we labeled some images that we applied very little artifacts by generating random values in the related ranges for bluriness \([0.3 - 0.8]\), brightness \([0.85 - 0.95]\), and contrast \([0.85 - 0.95]\) as real. Our dataset contains 60k images in total; 22k of these images are undistorted images and 8k of them are mildly distorted, both of which are labeled as real; and 30k images, which are highly distorted, are labeled as fake. We provide further details about the model architecture and training in the Supplementary Materials (Section A).

3.3 Elimination of Visual Artifacts

We need to preserve the structural continuity between a masked region and its surrounding context, for a realistic looking image inpainting. For this purpose, we train a Unet model, to eliminate visual artifacts in the converged CRG image around the mask boundaries. We adapted the Unet model from [14, 22] and trained it from scratch using different loss functions. During the training phase, a mask and CRG result is given to our Unet model as input and the original image is provided as the expected output. We aim to minimize the joint loss function in (4), which consists of per-pixel reconstruction loss and style loss [28] calculated for both \(I_{\text{Unet}}\) and \(I_{\text{comp}}\). Given the converged CRG result, \(I_{\text{CRG}}\), binary mask, \(M\), and the Unet prediction \(I_{\text{Unet}}\); \(I_{\text{comp}}\) is defined in (3).

\[
I_{\text{comp}} = M \odot I_{\text{CRG}} + (1 - M) \odot I_{\text{Unet}}
\]  

(3)

\[
L_{\text{joint}} = L_{\text{recons}} + 150(L_{\text{style}_{\text{Unet}}} + L_{\text{style}_{\text{comp}}})
\]  

(4)

Per-pixel reconstruction loss is defined in (5), where \(I_{\text{org}}\) and \(N_{\text{org}}\) denote original image and the number of pixels in \(I_{\text{org}}\), respectively.

\[
L_{\text{recons}} = \frac{1}{N_{\text{org}}} \| I_{\text{Unet}} - I_{\text{org}} \|_1
\]  

(5)

Style loss is provided for \(I_{\text{Unet}}\) in (6) and for \(I_{\text{comp}}\) in (7). We use layers pool1, pool2 and pool3 as in [14] in the pretrained VGG-16 network [29].

\[
L_{\text{style}_{\text{Unet}}} = \sum_{j=0}^{J-1} \frac{1}{C_j C_j} \| K_j \left( G_{\phi_j^{I_{\text{Unet}}}} - G_{\phi_j^{I_{\text{org}}}} \right) \|_1
\]  

(6)

\[
L_{\text{style}_{\text{comp}}} = \sum_{j=0}^{J-1} \frac{1}{C_j C_j} \| K_j \left( G_{\phi_j^{I_{\text{comp}}}} - G_{\phi_j^{I_{\text{org}}}} \right) \|_1
\]  

(7)

In (6) and (7), \(\phi_j^{I_*}\) corresponds to activation map of the \(j\)th layer (i.e. pool1, pool2 and pool3) for input \(I_*\). The shape of \(\phi_j^{I_*}\) is \(C_j \times H_j \times W_j\) where \(C\), \(H\) and \(W\) are the channel size, height and width.
of related layer, respectively. $G_{\phi^I_j}$ is a $C_j \times C_j$ Gram matrix, i.e. covariances of the activation map of $\phi^I_j$, and $K_j$ is normalization coefficient, i.e. $1/C_jH_jW_j$, for $j$th activation layer.

In Fig. 7, we show a sample removal of boundary artifacts with post-processing using our Unet model. In this sample, we first created a hand-drawn mask to keep the face part unchanged and applied the created mask to the image. We then roughly sketch black hair to the masked image to change the hair style to a darker one and use it as an input to our CRG model. After obtaining the inpainting result from the CRG model, we sampled three regions from both the raw CRG result and Unet result for comparison. The improvement after post-processing in the boundary regions can be seen clearly in the image pairs in the bottom row: we observe better structural consistency and smoothed color transition between the masked area and its surroundings after post-processing.

Fig. 7: Elimination of visual boundary artifacts after post-processing. The parts in the green, blue and red parts are shown with 300% zooming. (1)-left, (2)-left and (3)-top images are raw CRG outputs. (1)-right, (2)-right, (3)-bottom images are obtained after post-processing using Unet model.

4 Experiments

4.1 Capabilities of the Proposed Method

In this section, we empirically examine and show the capabilities of our approach. Firstly, our method is advantageous for producing different alternative fillings in the mask region depending on the mask content. For instance, in Fig. 1, after masking the beard region in the facial image and filling it with white pixels in (a), and black pixels in (b), we observe that, depending on the initial mask content, our method generates different and semantically meaningful content in the masked region. The model generates a beardless image when the mask content is white and an image with beard when it is black. Secondly, our method allows the use of different mask types, i.e. hand-drawn (e.g., (a), (b), etc.), irregular, e.g., (e), and regular masks (e.g., (f), (g)) etc. Thirdly, our method enables sketch-based inpainting since the masked image with sketch embeds to a related latent vector position. For example, in (g) we created a mask around the eye region with a thin-sketch similar roughly to eyeglasses, and as a result, a facial image with eyeglasses is produced; similarly, in (h), we created a mask with a solid-sketch similar roughly to sunglasses, and consequently a facial image with sunglasses is produced. Finally, our method allows multiple and diverse results. To illustrate this, we provide the result of each cycle for (k) in the right-bottom block. Note that several cycle results are useful; the discriminator could be disabled and each cycle result could be given to the Unet model as an input to get multiple results.
Despite the advances in GANs, training a generator model is still challenging; two main problems arise in general [30]. The first one is the stability problem; competition between the two networks, i.e. generator and discriminator, creates instability during training since one dominates the other. The second problem is the mode collapse problem; the generator collapses which produces limited varieties of samples. Recently, many tips and tricks [31], loss functions [32,33], or network architectures [34,35] have been proposed for dealing with these problems, but still it is an active area of research. For instance, the generator models may sometimes produce distorted images in some regions, i.e. images with acceptable face content but fail in the surroundings. Our method could also be useful to remedy the problems in such generated images by masking the parts that are distorted or we want to change or improve parts of those images. In Fig. 8, we depict an example image that is generated randomly using the CRG generator. When the generated image is examined, it is seen that the face part is plausible but the surrounding context could be improved. We created a hand-drawn mask for that purpose, to keep the face part unchanged and enable reproduction of the surrounding regions. Note that as the number of cycles progresses, the masked part improves and becomes more coherent with the face.

4.2 Model Comparisons

In the vast literature, facial inpainting models have been trained using different mask types at different resolutions. In our method, we utilize the CRG model, which does not require any supervision depending on the mask types. Therefore, our method is flexible considering the mask types, sizes and placements in the image. We compare our method performance qualitatively with the state-of-the-art models in the facial inpainting domain, i.e. PIC ¹ [12], GLCIC ² [8], GIICA ³ [7], GMCNN⁴ [11], EdgeConnect⁵ [10], DFNet⁶ [13], Pconv [14] for different mask types, i.e. hand-drawn (1−5), regular (6−7), and irregular (8−9). Some of these works have trained their own models using regular masks, e.g., GIICA and GLCIC, while some of them, e.g., EdgeConnect, DFNet and Pconv, use irregular masks. There is also another group that support both regular and irregular masks, e.g., PIC and GMCNN. During the experiments we obtained the results using official pre-trained models, if available; otherwise, we sent the masks and inputs to the model authors and get the results from them or used available pre-trained models that are configured as explained in the original articles. We resize the original image and mask for models that require different resolutions. For the PIC model, there are two published pretrained models depending on the mask placement; one serves when the mask is in the center of the image, the other serves for random placement. Similarly, the GMCNN model has two different pre-trained models for regular and irregular masks. We used the corresponding pre-trained model depending on the mask type. Also, the PIC model can generate multiple results for one mask, so we chose the best result based on their discriminator score. The comparison of the models are depicted in Fig. 9. As can be seen from the results, our method can produce qualitatively good results in

¹https://github.com/lyndonzheng/Pluralistic-Inpainting
²https://github.com/otenim/GLCIC-PyTorch
³https://github.com/JiahuiYu/generative_inpainting/tree/v1.0.0
⁴https://github.com/shepnerd/inpainting_gmcnn
⁵https://github.com/kmazeri/edge-connect
⁶https://github.com/hughplay/DFNet
different mask types and produce better results particularly in the hand-drawn mask types (i.e, 1-5). Note that almost all other models, have failed, especially in the 3rd and 4th images, due to the larger mask sizes. These models do not use similar masks during their training phase. Our method does not use any mask in the training phase; thanks to the generator, our method allows inpainting in the case of larger masks.

For quantitative comparisons, facial inpainting works report results using L1 error, PSNR, SSIM [36], and inception score [31] in general. However, many works [7, 18, 37] mention that there are no good evaluation metrics due to many possible solutions different from the original image content. These metrics generate scores based on the similarity of the produced results to the original image. However, this is not something necessarily desired in this domain, e.g., sample shown in Fig. 8.

Fig. 9: Comparison of the model results.
4.3 Inpainting with Real Images

As well as the generated images, our method is also useful and effective for inpainting real images. As can be seen in Fig. 10, realistic inpaintings are obtained using a variety of different mask types, in different placements with different sizes using real images as inputs.

Fig. 10: Sample inpainting with real images using a variety of different masks.
4.4 Failure Cases

The sufficiency of the generator latent space is crucial for the success of our method; for a sufficient generator model, for almost every point from the prior z distribution, sampling results, i.e. generated images, contain plausible semantic content. In this context, it is important that masked image could be embedded in a region with sufficient variance in the generator latent space. In some locations in the latent space, however, the manifold of trained examples do not have sufficient density. These regions are referred to as dead zones [38]. Sampling from such regions will not be useful to fill the mask content properly for the next cycle due to lack of variance around that point in the latent space.

The level of defect applied to an image reinforces the probability of sampling from these dead zones, due to insufficient semantic content. An example is depicted in Fig. 11; in (a), the defect created to the semantic content of the input image with the applied mask is not as severe as the one depicted in (b). The mask in (b), hides important semantic content in the image, and make embedding to the latent space more challenging for the CRG encoder. As can be seen from the second rows of both derivations, embedding the masked image in the first one, (a), could be done in a region with sufficient variance, while for the second one, (b), embedding is done in a dead zone. As can be seen in the generated images of (b), the density around the embedded point is not high to recover a plausible face image in that area.

![Fig. 11: A sample for a failure case. For (a) and (b), top-row: shows the result of each the CRG cycle, bottom row: produced image in the generator space for the relevant cycle.](image)

5 Conclusion and Future Works

We propose an iterative algorithm to fill a masked region semantically with very few iterations via encoder-generator architecture. The encoder embeds a masked image to a latent point in generator space; since the content of the masked region directly affects the embedded point, a variety of alternative fillings are generated depending on different drawn-sketches or different pixel values assigned to the
masked region. Also, since our solution uses generic frameworks, i.e. CRG, which does not need any supervision depending on the masks, it allows the use of different mask types. The empirical results show that our heuristic works successfully and outperforms the state-of-the-art models qualitatively, especially in images where the face or background is completely masked. The success of our method depends on the sufficiency of the generator space and how well its inversion is performed; so, as a future work, we are planning to make more comprehensive analysis on the manifold of trained examples in the latent space utilizing the encoder of the CRG model.

Acknowledgements

This research is funded by Ankara University (Scientific Research Projects Grant, grant id: 18L0443010). The numerical calculations reported in this paper were partially performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

References

[1] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani, “Summarizing visual data using bidirectional similarity,” in 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8, IEEE, 2008.

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, “Patchmatch: A randomized correspondence algorithm for structural image editing,” ACM Trans. Graph., vol. 28, no. 3, p. 24, 2009.

[3] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen, “Image melding: Combining inconsistent images using patch-based synthesis,” ACM Transactions on graphics (TOG), vol. 31, no. 4, pp. 1–10, 2012.

[4] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf, “Image completion using planar structure guidance,” ACM Transactions on graphics (TOG), vol. 33, no. 4, pp. 1–10, 2014.

[5] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, “Context encoders: Feature learning by inpainting,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2536–2544, 2016.

[6] Y. Li, S. Liu, J. Yang, and M.-H. Yang, “Generative face completion,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3911–3919, 2017.

[7] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, “Generative image inpainting with contextual attention,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5505–5514, 2018.

[8] S. Iizuka, E. Simo-Serra, and H. Ishikawa, “Globally and locally consistent image completion,” ACM Transactions on Graphics (ToG), vol. 36, no. 4, pp. 1–14, 2017.

[9] Q. Wang, H. Fan, G. Sun, Y. Cong, and Y. Tang, “Laplacian pyramid adversarial network for face completion,” Pattern Recognition, vol. 88, pp. 493–505, 2019.

[10] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi, and M. Ebrahimi, “Edgeconnect: Generative image inpainting with adversarial edge learning,” arXiv preprint arXiv:1901.00212, 2019.

[11] Y. Wang, X. Tao, X. Qi, X. Shen, and J. Jia, “Image inpainting via generative multi-column convolutional neural networks,” in Advances in neural information processing systems, pp. 331–340, 2018.

[12] C. Zheng, T.-J. Cham, and J. Cai, “Pluralistic image completion,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1438–1449, 2019.

[13] X. Hong, P. Xiong, R. Ji, and H. Fan, “Deep fusion network for image completion,” in Proceedings of the 27th ACM International Conference on Multimedia, pp. 2033–2042, 2019.

[14] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, “Image inpainting for irregular holes using partial convolutions,” in Proceedings of the European Conference on Computer Vision (ECCV), pp. 85–100, 2018.

[15] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, “Free-form image inpainting with gated convolution,” in Proceedings of the IEEE International Conference on Computer Vision, pp. 4471–4480, 2019.

[16] H. Shao, Y. Wang, Y. Fu, and Z. Yin, “Generative image inpainting via edge structure and color aware fusion,” Signal Processing: Image Communication, vol. 87, p. 115929, 2020.

[17] Z. Chen, S. Nie, T. Wu, and C. G. Healey, “High resolution face completion with multiple controllable attributes via fully end-to-end progressive generative adversarial networks,” arXiv preprint arXiv:1801.07632, 2018.

[18] R. A. Yeh, C. Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, “Semantic image inpainting with deep generative models,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5485–5493, 2017.

[19] R. Abdal, Y. Qin, and P. Wonka, “Image2stylegan++: How to edit the embedded images?,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8296–8305, 2020.
[20] Y. Dogan and H. Y. Keles, “Semi-supervised image attribute editing using generative adversarial networks,” Neurocomputing, vol. 401, pp. 338–352, 2020.

[21] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in Proceedings of the IEEE international conference on computer vision, pp. 3730–3738, 2015.

[22] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1125–1134, 2017.

[23] P. Pérez, M. Gangnet, and A. Blake, “Poisson image editing,” in ACM SIGGRAPH 2003 Papers, pp. 313–318, 2003.

[24] A. Telea, “An image inpainting technique based on the fast marching method,” Journal of graphics tools, vol. 9, no. 1, pp. 23–34, 2004.

[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in neural information processing systems, pp. 2672–2680, 2014.

[26] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” arXiv preprint arXiv:1710.10196, 2017.

[27] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4401–4410, 2019.

[28] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2414–2423, 2016.

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

[30] Y. Dogan and H. Y. Keles, “Stability and diversity in generative adversarial networks,” in 2019 27th Signal Processing and Communications Applications Conference (SIU), pp. 1–4, IEEE, 2019.

[31] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training gans,” in Advances in neural information processing systems, pp. 2234–2242, 2016.

[32] S. Martin Arjovsky and L. Bottou, “Wasserstein generative adversarial networks,” in Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 2017.

[33] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of wasserstein gans,” in Advances in neural information processing systems, pp. 5767–5777, 2017.

[34] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” arXiv preprint arXiv:1511.06434, 2015.

[35] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of stylegan,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8110–8119, 2020.

[36] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE transactions on image processing, vol. 13, no. 4, pp. 600–612, 2004.

[37] J. Wei, G. Lu, H. Liu, and J. Yan, “Facial image inpainting with deep generative model and patch search using region weight,” IEEE Access, vol. 7, pp. 67456–67468, 2019.

[38] T. White, “Sampling generative networks,” arXiv preprint arXiv:1609.04468, 2016.

[39] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[40] T. Tieleman and G. Hinton, “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude,” COURSERA: Neural networks for machine learning, vol. 4, no. 2, pp. 26–31, 2012.
A The Architectures and Training Details

**Unet architecture and training details:** We adapt our Unet architecture from those [14, 22], replacing all partial convolutional layers with standard convolutional layers. We only use masks to calculate style losses, so we do not follow the mask update steps. Table 1 provides other architectural details. Each convolution layer is followed by a batch normalization layer. Similar to [14], we perform down-sampling and up-sampling with a factor of 2, we use Relu activation function in all convolution layers in the encoder, and LeakyRelu, with slope 0.2, in the decoder.

| Encoder | Layer name | Filter size | Number of filters |
|---------|------------|-------------|-------------------|
| Conv1   | 7x7        | 64          |
| Conv2   | 5x5        | 128         |
| Conv3   | 5x5        | 256         |
| Conv4   | 3x3        | 512         |
| Conv5   | 3x3        | 512         |
| Conv6   | 3x3        | 512         |

| Decoder | Upsample1 | - | 512 |
|         | Concat (with Conv6) | - | 512+512 |
|         | Conv7 | 3x3 | 512 |
|         | Upsample2 | - | 512 |
|         | Concat (with Conv5) | - | 512+512 |
|         | Conv8 | 3x3 | 512 |
|         | Upsample3 | - | 512 |
|         | Concat (with Conv4) | - | 512+512 |
|         | Conv9 | 3x3 | 512 |
|         | Upsample4 | - | 512 |
|         | Concat (with Conv3) | - | 512+256 |
|         | Conv10 | 3x3 | 256 |
|         | Upsample5 | - | 256 |
|         | Concat (with Conv2) | - | 256+128 |
|         | Conv11 | 3x3 | 128 |
|         | Upsample6 | - | 128 |
|         | Concat (with Conv1) | - | 128+64 |
|         | Conv12 | 3x3 | 64 |
|         | Upsample7 | - | 64 |
|         | Concat (with Input) | - | 64+3 |
|         | Conv13 | 3x3 | 3 |

We train the Unet model from scratch for 100 epochs, using Adam optimizer [39] with a batch size of 16 and a small learning rate, i.e. 0.0002. To avoid overfitting or positional bias, we apply data augmentation where we shift the images left, right, up, or down and apply horizontal and vertical flips.

**The Discriminator network and training details:** We designed our discriminator as a multi-layer convolutional neural network depicted in Fig. 6. The encoder consists of 3 blocks and a fully connected layer with a total of approximately 437k trainable parameters. We use 64 filters in the first block and double the number of filters in consecutive blocks. We use a 3x3 filter size in the convolution layers. To avoid overfitting, we apply 50% spatial dropout. We train the discriminator from scratch for 300 epochs, using RMSProp optimizer [40] with a batch size of 128, setting learning rate to 0.0001, rho to 0.9 and epsilon to 1e-08. We reduce the learning rate by a factor of 2 when minimum validation loss stops improving for 15 epochs and saves the best model using model checkpoint monitoring validation loss.