Image Inpainting using Block-wise Procedural Training with Annealed Adversarial Counterpart

Anonymous ECCV submission

Paper ID 1283

Abstract. Recent advances in deep generative models have shown promising potential in image inpanting, which refers to the task of predicting missing pixel values of an incomplete image using the known context. However, existing methods can be slow or generate unsatisfying results with easily detectable flaws. In addition, there is often perceivable discontinuity near the holes and require further post-processing to blend the results. We present a new approach to address the difficulty of training a very deep generative model to synthesize high-quality photo-realistic inpainting. Our model uses conditional generative adversarial networks (conditional GANs) as the backbone, and we introduce a novel block-wise procedural training scheme to stabilize the training while we increase the network depth. We also propose a new strategy called adversarial loss annealing to reduce the artifacts. We further describe several losses specifically designed for inpainting and show their effectiveness. Extensive experiments and user-study show that our approach outperforms existing methods in several tasks such as inpainting, face completion and image harmonization. Finally, we show our framework can be easily used as a tool for interactive guided inpainting, demonstrating its practical value to solve common real-world challenges.

Keywords: deep generative model, image inpainting, image harmonization, image composition.

1 Introduction

Image inpainting is the task to fill in the missing part of an image with visually plausible contents. It is one of the most common operations of image editing [1] and low-level computer visions [2, 3]. The goal of image inpainting is to create semantically plausible contents with realistic texture details. The inpainting can either be consistent with the original image or be different but coherent with the known context. Other than restoring and fixing damaged images, inpainting can also be used to remove unwanted objects, or in the case of guided inpainting, be used to composite with another guide image. In the latter scenario, we often need inpainting to fill in the gaps and remove discontinuities between target region of interest on the guide image and the source context. Image harmonization is also required to adjust the appearance of the guide image such that it is compatible with the source, making the final composition appear natural.

(a) Input (b) GLI [10] (c) Ours

(a) Input (b) DH [9] (c) Ours

Fig. 1. An example of inpainting (top) and harmonzation (bottom) comparing with state-of-the-art methods. Zoom in for best viewing.

Traditional image inpainting methods mostly develop texture synthesis techniques to address the problem of hole-filling [4, 2, 5–8]. In [6], Barnes et al. proposes the Patch-Match algorithm which efficiently searches the most similar patch to reconstruct the missing regions. Wilczkowiak et al. [8] takes further steps and detects desirable search regions to find better match patches. However, these methods only exploit the low-level signal of the known contexts to hallucinate missing regions and fall short of understanding and predicting high-level semantics. Furthermore, it is often inadequate to capture the global structure of images by simply extending texture from surrounding regions. Another line of work for inpainting aims to fill in holes with contents from another guide image, by using composition and harmonization [3, 9]. The guide is often retrieved from a large database based on similarity with the source image and is then combined with the source. Although these methods are able to propagate high-frequency details from the guide image, they often lead to inconsistent regions and gaps which are easily detectable with human eyes.

More recently, deep neural networks have shown excellent performance in various image completion tasks, such as texture synthesis and image completion. For inpainting, adversarial training becomes the de facto strategy to generate sharp details and natural looking results [11–14, 10]. Pathak et al. [11] first proposes to train an encoder-decoder model for inpainting using both the reconstruction

loss and adversarial loss. In [12], Yeh et al. uses a pre-trained model to find the most similar encoding of a corrupted image and uses the found encoding to synthesize what is missing. In [14], Yang et al. proposes a multi-scale approach and optimizes the hole contents such that its neural feature extracted from a pre-trained CNN matches with the features of the surrounding context. The optimization scheme improves the inpainting quality and resolution at the cost of computational efficiency. In [10], Iizuka et al. proposes a deep generative model trained with global and local adversarial losses, and can achieve good inpainting performance for mid-size images and holes. However, it requires extensive training (two months as described in the paper), and the results often contain excessive artifacts and noise patterns. Another limitation of [14] and [10] is that they are unable to handle perceptual discontinuity, making it necessary to resort to post-processing (e.g. Poisson blending).

In practice, we found that directly training a very deep generative network to synthesize high-frequency details is challenging due to optimization difficulty and unstable adversarial training. As a result, as the network becomes deeper, the inpainting quality may worsen. To overcome this difficulty, we discuss a new training scheme that guides and stabilizes the training of a very deep generative model, which, combining with carefully designed training losses, significantly reduces the artifacts and improves the quality of results. The main strategy, referred to as **B**lock-wise **P**rocedural **T**raining (**BPT**), progressively increases the number of residual blocks and the depth of the network. With BPT, we first train a cGAN-based **Generator Head** until converging, followed by adding additional residual blocks one at a time to refine and improve the results. This enables us to train a network deeper than [10] and generates more realistic-looking details. We also observe that to reduce the noise level, it is essential to steadily reduce the weight of the adversarial loss given to the generator. We refer to this training scheme as **A**dversarial **L**oss **A**nnealing (**ALA**). Finally, we also propose several new losses specifically designed for inpainting: the **P**atch **P**erceptual **L**oss (**PPL**) and **M**ulti-**S**cale **P**atch **A**dversarial **L**oss (**MSPAL**). Experiments show that these losses work better than traditional losses used for inpainting, such as ℓ_2 and the more general adversarial loss.

To evaluate the proposed approach, we conduct extensive experiments on a variety of datasets. Shown by qualitative and quantitative results, also by the user-study, our approach generates high-quality inpainting results and outperforms other state-of-the-art methods. We also show that our framework, although being designed for inpainting, can be directly applied on general image translation tasks such as image harmonization and composition, and easily achieve results superior to other methods (Fig. 1). This enables us to train a multi-task model and use it for interactive guided inpainting, which is a very common and useful image editing scenario. We will describe it in detail in Sec. 4.4.

In summary, in this paper we present:

1. A novel and effective framework that generates high-quality results for several image editing tasks like inpainting and harmonization. We also provide

a thorough analysis and ablation study about the components in our framework.
2. Extensive qualitative and quantitative evaluations on a variety of datasets, such as scenes, objects, and faces. We also conduct a user-study to make fair and rigorous comparisons with other state-of-the-art methods.
3. Results of interactive guided inpainting as a novel and useful application of our approach.

2 Related Work

Deep Image Generation and Manipulation Generative Adversarial Network (GAN) [15] uses a mini-max two-player game to alternatively train a generator and a discriminator, and has shown impressive ability to generate natural-looking images of high-quality. However, training instability of the original GAN makes it hard to scale to large images, and many more advanced techniques have been proposed [16–20]. Recently, Progressive GAN [21] is proposed which can be trained to generate images of unprecedented quality. Our procedural block-wise training and Progressive GAN share the basic idea of progressively increasing the depth of the network while training. However, both the problems being studied (image generation vs conditional synthesis) and the model architectures used are different.

Adversarial training has also been applied to many image editing tasks [22–26]. For image inpainting, many DNN-based approaches achieve good performance using different network topology and training procedure [11, 14, 12, 10]. For image harmonization which aims to adjust the appearances of image composition such that it looks more natural and plausible, recent approaches mostly use deep neural network to leverage its expressive power and semantic knowledge [27, 9].

Non-neural Inpainting and Harmonization Traditional image completion algorithms can be either diffusion-based [4, 28] or patch-based [7, 6]. Diffusion-based methods usually cannot synthesize plausible contents for large holes or textures, due to the fact that it only propagates low-level features. Patch-based methods, however, largely rely on the assumption that the desired patches exist in the database. For harmonization, traditional methods usually apply color and tone optimization, by matching global or multi-scale statistics [29, 30], extracting gradient domain information [31, 32], utilizing semantic clues [33] or leveraging external data [34].

3 Our Method

In this section, we describe our model and several training schemes. First, we illustrate the details of our basic components: the generator head and the training losses, in Sec. 3.1. Then, we describe the block-wise procedural training scheme and the adversarial loss annealing in Sec. 3.3. Finally, we summarize our implementation and training details.

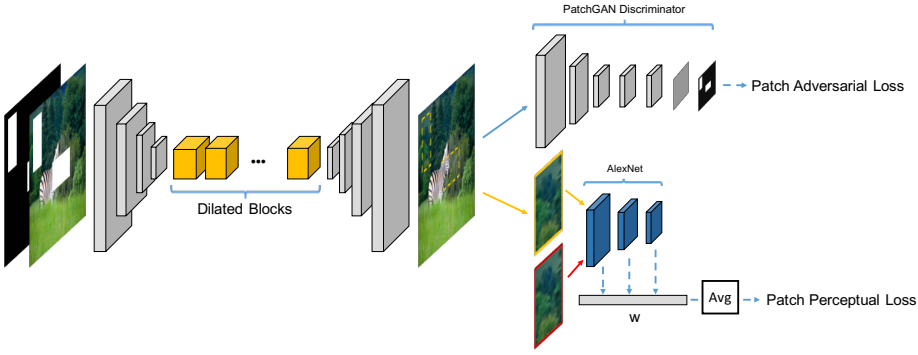


Fig. 2. Generator head and the training losses. We only illustrate one scale of Patch Adversarial Loss and Patch Perceptual Loss. Note that to compute the Patch Adversarial Loss, we need to use the mask to find out which patch overlaps with the hole.

Our generator head is a conditional GAN network [35], which takes an incomplete image as the input, and outputs a complete image. Conditional GANs for image inpainting usually consist of a generator G and a discriminator D. The generator G learns to predict the hole contents and restore the complete image, while the discriminator D learns to distinguish real images from the generated ones. The model is trained in a self-supervised manner via the following minimax game:

$$\min_G \max_D E_{(s,x)}[\log D(s,x)] + E_s[\log(1 - D(s, G(s)))], \tag{1}$$

where s and x are the incomplete image and the original image respectively, and $G(s)$ is the generator prediction given the input s. Note that if $G(s)$ predicts an entire image as output, we only keep the hole contents and concatenate with the known context of s.

3.1 The Generator Head

Previous research experimented with different architecture of G, most notably the U-Net style generator of [11] and the FCN style generator of [10]. [10] shows that FCN style inpainting network produces less blurred results than U-Net, as it adopts fully convolutional layers instead of an intermediate fully connected (FC) layer, which avoids significant resolution reduction or information loss.

Similar to [10], our generator head is based on FCN and leverage the many properties of convolutional neural networks, including translation invariance and parameter sharing. Nevertheless, a major limitation of FCN is the constraint of the receptive field size, since the convolution layers are locally connected, making pixels far away from the hole carry no influence on the predicted hole content. We rely on several strategies to alleviate such drawback. First, like [10], we use a down-sampling front end to reduce the feature size, followed by multiple residual blocks [36], and then an up-sampling back end to restore the full dimension. By

downsampling, we increase the receptive field of the ResNet blocks. Second, we stack multiple ResNet blocks to further enlarge the receptive field. Finally, we adopt the dilated convolutional layers [37] in all ResNet blocks. Dilated convolutions use spaced kernels, making it compute each output value with a wider view of input without increasing the number of parameters and computational burden. We set the dilation factor to 2 in all layers. Overall, we observe context is critical for realism, and the receptive size poses as an important factor for inpainting quality. This differentiates it from other image translation tasks.

Specific to our network, the down-sampling front end consists of three convolutional layers, each with stride 2. The intermediate residual blocks contain 9 blocks stacked together, and the up-sampling back-end consists of three transposed convolution of stride 2. Each convolutional layer is followed by batch normalization (BN) and ReLu as the activation layer, except for the last layer which outputs the image. For down-sampling and up-sampling, an alternative would be using interpolated convolution to reduce the checkerboard effect, as suggested by [38]. The interpolated convolution uses a dimension-preserving convolution layer of stride 1, followed by max pooling or bilinear up-sampling. However, we observed that using interpolated convolution creates overly smooth effects. A detailed ablation study is presented in Sec. 5.

3.2 The Training Losses

Different losses have been used to train an inpainting network. These losses can be cast into two categories. The first category, which can be referred to as *similarity loss*, is used to measure the similarity between the output and the original image. The second category, which we refer to as the *realism loss*, is used to measure how realistic-looking the output image is. We summarize the losses used in different approaches in Table 1.

Method	Similarity Loss	Realism Loss
Context Encoder [11]	ℓ_2	Global Adversarial Loss
Global Local Inpainting [10]	ℓ_2	Global and Local Adversarial Loss
Our Approach	Patch Perceptual Loss (PPL)	Improved Multi-Scale Adversarial Loss

Table 1. Comparison of training losses in different methods.

Patch Perceptual Loss As shown in Table 1, using ℓ_2 loss for reconstruction and measure the disparity between the output and the original image has been the default choice of previous inpainting methods. However, it is known that ℓ_2 loss does not correspond well to human perception of visual similarity (Zhang et al. [39]). This is because ℓ_2 losses wrongly assumes each output pixel is conditionally independent of all others. A well-known issue, for example, is that blurring an image leads to small changes in terms of Euclidean distance but causes significant perceptual difference. Recent research suggests that a better metric for perceptual similarity is the internal activations of deep convolutional networks,

usually trained on a high-level image classification task. Such loss is called "perceptual loss", and is used in various tasks such as neural style transfer [40], image super-resolution [41], and conditional image synthesis [42, 43].

Based on this observation, we propose a new "patch perceptual loss" as the substitute of the ℓ_2 losses. Traditional perceptual loss typically uses VGG-Net, and computes the ℓ_2 distance of the activations on a few feature layers. Recently, [44] specifically trained a patch perceptual network to measure the perceptual differences between two image patches based on AlexNet, making it an ideal candidate for our task. The patch perceptual network computes the activations across all feature layers and sums up the ℓ_2 distances scaled by learned weights at each layer. Furthermore, to take into account both the local view and the global view of perceptual similarity, we compute PPL at two scales. Local PPL considers the local hole patch, while the global PPL slightly zooms out to cover a larger contextual area. More formally, our PPL is defined as:

$$\sum_{p=1,2} PPL_k(G(s)_p, x_p) = \sum_{k=1,2} \sum_{l} \frac{1}{H_l W_l} \sum_{h,w} \parallel w_l^T \odot (\hat{F}(x_p)^l_{hw} - \hat{F}(G(s)_p)^k_{hw}) \parallel_2^2 \quad (2)$$

Here p refers to the hole patch. $\hat{F}$ is the AlexNet and l is the feature layer. Abalation study in Sec. 5 shows that PPL gives better inpainting quality than both ℓ_2 and VGG-based perceptual losses.

Multi-Scale Patch Adversarial Loss Adversarial losses are given by trained discriminators to discern whether an image is real or fake. The global adversarial loss of [11] takes the entire image as input and outputs a single real/fake prediction, which does not consider the realism of local patches, especially the holes. The additional local adversarial loss of [10] adds another discriminator specifically for the hole, but it requires the hole to be fixed shape and size during training to fit the discriminator. To consider both the global view and the local view, and to be able to use holes of arbitrary number and shapes, we propose to use PatchGAN discriminators [25] at three scales of image resolutions. The discriminator at each scale is identical, only the input is a differently scaled version of the *entire image*. The PatchGAN discriminator at each scale is a convolutional discriminator which outputs a vector of predictions, where each value corresponds to an image patch. In this way, the discriminators are trained to classify global and local image patches across the image, and also enable us to use random holes and shapes during training. However, directly using PatchGAN is problematic in our case, as for the output restored image, only the patches overlapping with the hole area should be considered fake patches. Therefore when computing the discriminator loss of the restored image, instead of forcing the entire output to be fake, only the patches overlapping with the holes are labeled as fake. More formally, our Patch-wise Adversarial Loss is defined as:

$$\min_G \max_{D_1,D_2,D_3} \sum_{k=1,2,3} L_{GAN}(G, D_k) \quad (3)$$

$$= \sum_{k=1,2,3} E_{(s_k,x_k)}[\log(s_k, x_k)] + E_s[\log(Q - D_k(s_k, G(s_k)))]. \quad (4)$$

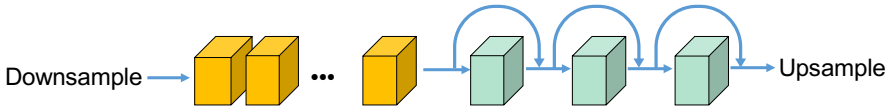


Fig. 3. Illustration of block-wise procedural training. The yellow residual blocks refer to the generator head which are trained first. The green residual blocks are progressively added one at a time. We also draw the skip connections between the already trained residual blocks and the up-sampling back end.

Here k refers to the image scale, and Q is a patch-wise real/fake vector, based on whether the patch overlaps with the holes. Using multiple GAN discriminators at the same or different image scale has been proposed in unconditional GANs [45] and conditional GANs [46]. Here we extend the design to take into account inpainting hole locations, which is critical in obtaining semantically and locally coherent image completion results.

To summarize, our full objective combines both losses is therefore defined as:

$$\min_{G}((\max_{D_1,D_2,D_3} \lambda_{Adv} \sum_{k=1,2,3} L_{GAN}(G,D_k)) + \lambda_{PPL} \sum_{k=1,2} PPL_k(G(s)_p, x_p)). \quad (5)$$

We use λ_{Adv} and λ_{PPL} to controls the importance of the two terms. In our experiment, we set $\lambda_{Adv} = 1$ and $\lambda_{PPL} = 10$ to begin with. As training progresses, we gradually decrease the weight of λ_{Adv} as explained in Sec. 3.3.

Fig. 2 illustrates the architecture of our generator head and the training losses.

3.3 Blockwise Procedural Training with Adversarial Loss Annealing

Our experiments show that using the described generator head and training losses already give inpainting results better than state-of-the-art. However, the results can still lack fine details, especially when synthesizing textures. Result may contain noise patterns when the image is complicated. A straight-forward effort to improve the results would be to stack more intermediate residual blocks to further expand the receptive view and increase the expressiveness of the model. However, we found that directly stacking more residual blocks makes it more difficult to stabilize the training. As the search space becomes much larger, it is also more challenging to find local optimum. In the end, the inpainting quality deteriorates as the model depth increases.

Recently, Progressive GAN [21] was proposed as a new training methodology for generative adversarial networks. The key idea is to grow both the generator and discriminator progressively, by adding new layers that model increasingly fine details as training progresses. This strategy makes the training faster and more stable and enables it to synthesize mega-pixel images with unprecedented visual quality.

We adapt this idea for image inpanting and propose to use procedural block-wise training to gradually increase the depth of the inpainting network. More specifically, we begin by training the generator head until it converges. Then, we

add a new residual block after the already trained residual blocks, right before the back-end upsampling layers. In order to smoothly introduce the new residual block without suddenly breaking the trained model, we add another skip path from the trained residual blocks to the upsampling layers. Initially, the weight of the skip path is set to 1, while the weight of the path containing the new block is set to 0. This essentially makes the initial network identical to the already trained network. We then slowly decrease the weight of the skip path and increase the weight of the new residual block as training progresses. In this way, the newly introduced residual block is trained to be a fine-tuning component, which adds another layer of fine details to the original results. This step are repeated multiple times, where each time we expand the model by adding a new residual block. In our experiment, we found that the results improve significantly after fine-tuning with the first additional residual block. The output becomes stabilized after three residual blocks, when little discernible changes can be detected if even more residual blocks are introduced. The procedural training process is illustrated in Fig. 7.

We observe that the block-wise procedural training has several benefits. First, it guides the training process of a very deep generator. Starting with the generator head and gradually fine-tuning with more residual blocks makes it easier to discover the mapping between the incomplete image and the complete image, even though the search space is huge given the diversity of natural images and the random holes. Another benefit is reduced training time, as we found decoupling the training of the generator head and the fine-tuning of additional residual blocks requires significantly less training time comparing with training the network all at once.

Adversarial Loss Annealing During training, the generator adversarial loss updates the generator weight if the discriminator successfully detects the generated image as fake:

$$\sum_{k=1,2,3} E_s[\log(\bar{Q} - D_k(s_k, G(s_k)))]. \tag{6}$$

Note that here $\bar{Q}$ reverses Q of 3.2 as this is the loss to the generator. We observe that the generator adversarial loss becomes dominant over PPL as training progresses. This is because the discriminator becomes increasingly good at detecting fake images during training. This is less of a problem for image synthesis tasks. However, for the inpainting task which requires the output to be faithful to the original image, the outcome is that the generator deliberately adds noise patterns to confuse the discriminators, bringing more artifact to output or even synthesizing wrong textures. Based on this observation, we propose to use adversarial loss annealing, which decreases the weight of the generator adversarial loss when adding new residual block. More formally, let the initial weight of the generator adversarial loss be λ^0_{adv}, and the weight of the generator adversarial loss be λ^i_{adv} after adding the i_{th} residual block. We found that simply decay the weight linearly by setting $\lambda^i_{G_{adv}} = 0.1^i \lambda^0_{G_{adv}}$ gives satisfying results. Detailed analysis are described in Sec. 5.

Finally, we summarize the discussed training schemes in Alg. 1.

Algorithm 1 Training the Inpainting Network

1: Set batch size to 8 and basic learning rate to $lr_0 \leftarrow 0.0002$.
2: Set $\lambda_{PPL} \leftarrow 10$ and $\lambda^0_{adv} \leftarrow 1$.
3: Train the Generator Head G_0 using MSPAL and PPL (3.2) for 150,000 iterations.
4: **for** i=1 to 3 **do**
5: Add the skip path and the residual block r_i.
6: Set $lr_i \leftarrow 0.1^i lr_0$ and $\lambda^i_{G_{adv}} \leftarrow 0.1^i \lambda^0_{G_{adv}}$.
7: Train the generator G_3 with the added r_i for 1,500 iterations.
8: **return** G_3

4 Results

In this section, we first describe our dataset and experiment setting (Sec. 4.1). We then provide quantitative and qualitative comparisons with several methods, and also report a subjective human perceptual test with user-study (Sec. 4.2). In Sec. 4.3, we conduct several ablation study about the design choice of the models, losses, and training scheme. Finally, we show how our method can be applied to real use cases of object removal and guided inpanting. In particular, the inpainting model can be adapted to train an image harmonization network, which generates state-of-the-art harmonization results. We then demonstrate that by jointly training a model for inpainting and harmonization we can easily achieve guided inpainting (Sec. 4.4).

4.1 Experiment Setup

We evaluate our inpainting method on several representative datasets. COCO [47] is a large dataset containing both object and scene images; Place2 [48] includes images of a diversity of scenes and was originally meant for scene classification; finally, we train and test on CelebA [49], which consists of 202,599 face images with various viewpoints and expressions. For a fair comparison, we follow the standard split with 162,770 images for training, 19,867 for validation and 19,962 for testing.

In order to compare with existing methods, we train on images of size 256x256. We also train another network at a larger scale of 512x512 to demonstrate its ability to handle higher resolutions and compare with neural patch synthesis [14]. As the pre-processing step, we first resize the image and then conduct random cropping. We then apply data augmentation with random flipping. For each image, we create a mask containing one or two rectangle holes. The size of the hole ranges from 1/4 to 1/2 of the image's dimension, and are positioned at random locations. Note that during inference, our network is able to handle masks with an arbitrary number of holes of any shape. Finally, we shift and rescale the pixel value from [0,255] to [-1,1] and fill in the masked regions with zeros. We then concatenate the corrupted image and the mask as input.

For all our training, we set the learning rate with polynomial decay starting from 0.0002, and adopt Adam for optimization. We set the batch size to 8, and

regardless of the actual dataset size, we train 150,000 iterations for the generator head and another 1,500 iterations for each additional residual block. For each dataset, the training takes around 2 days to finish on a single Titan X GPU.

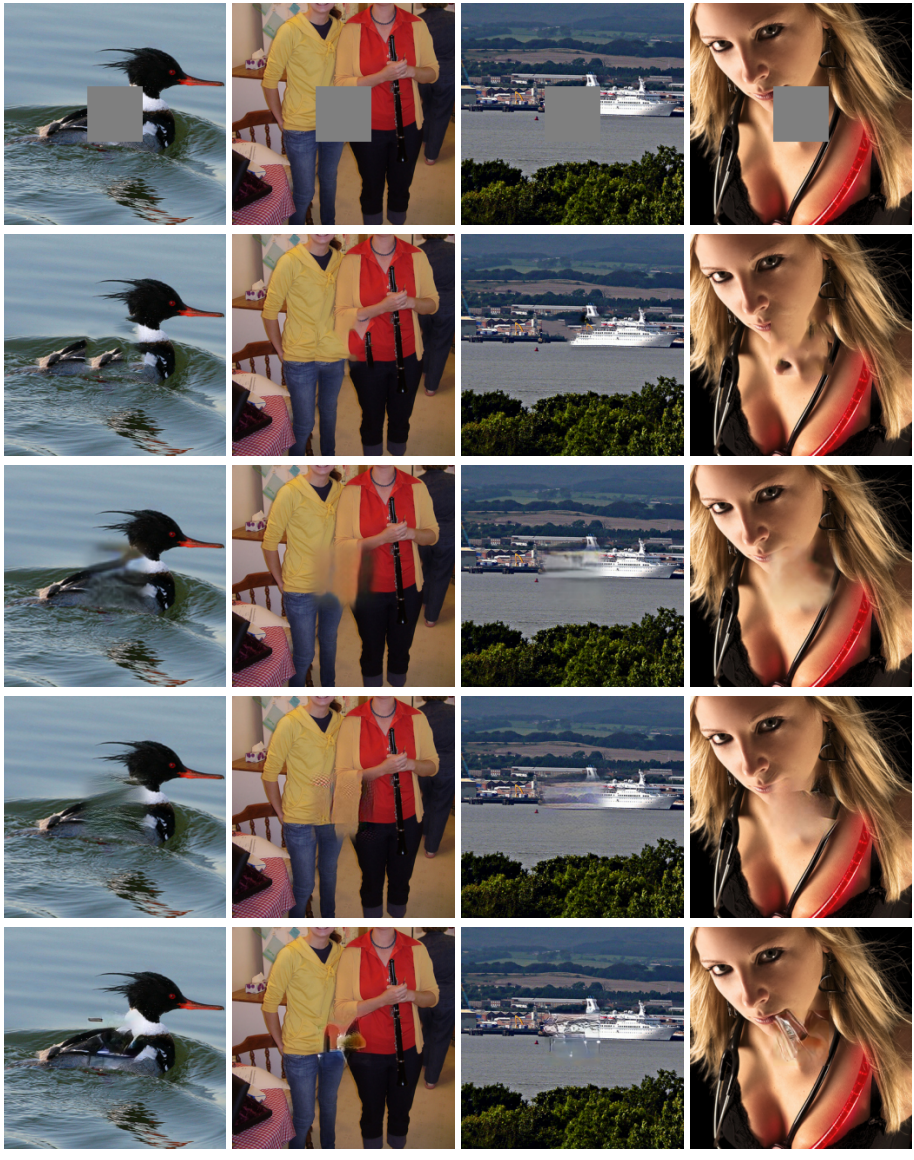


Table 2: Fixed hole completion comparison. From top row to bottom row: input image, CAF, CE, NPS, GLI and our final results.

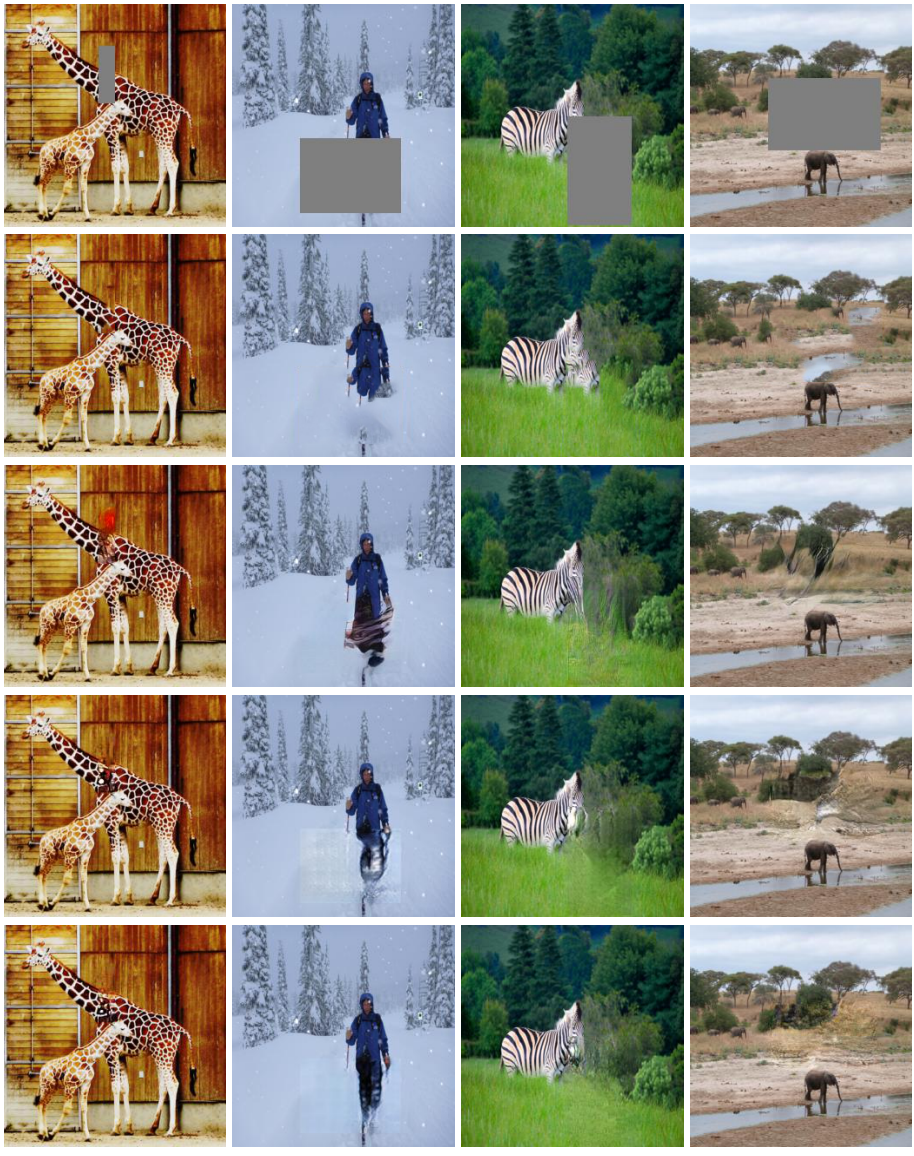


Fig. 4. Random hole completion comparison. From top row to bottom row: input image, CAF results, GLI results, our generator head results, our final results.

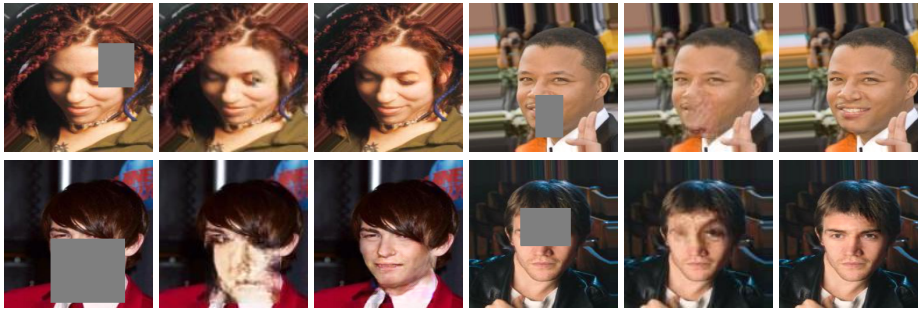
4.2 Comparison with Existing Methods

At 256x256, we compare our results with Content-Aware Fill (CAF) [6], Context Encoder (CE) [11], Neural Patch Synthesis (NPS) [14] and Global Local Inpainting (GLI) [10]. For CE and GLI, we use off-the-shelf pre-trained models from the web. Note CE's model is trained with center holes and other models are

able to handle random holes. We evaluate on both settings, using random holes (Fig. 4) or center holes (Fig. 4.1). For center hole completion, we compare with CAF, CE, NPS and GLI on ImageNet [50] test images. In this case, our results are directly generated by models trained on COCO. For random completion, we compare with CAF and GLI using images from COCO and Place2. Note GLI's results are after Poisson Blending as post-processing, which other methods do not use. For our approach, we show the results generated by the Generator Head alone as well as the final results using procedural training as refinement. The comparison results shown are randomly sampled from the entire test set.

Based on the visual results we can see that, for CAF as a non-learning and patch-based approach, its main issue is the inability to generate novel objects not available in the known context. This is especially an issue for highly specific and complex structure such as face inpainting. Furthermore, while CAF is able to generate realistic-looking details, they do not always capture the global structure and the inpainting is often inconsistent when the contexts are complex. CE's result is blurrier, and the border between the hole and the context is easily detectable. Comparing with NPS, our approach is much faster, as it is purely feed-forward while NPS is optimization-based. In fact, NPS takes about 20 seconds to inpaint an image of 256x256 and 60 seconds for an 512x512 image. Our results are also visually better than NPS on 256x256. Comparing with GLI, our results do not need fine-tuning, and are less noisy, more coherent, and have better quality.

For CelebA, we compare with Generative Face Completion [13] in Fig. Since [13]'s model inpaints images of 128x128, we have to upsample the results to 256x256 to compare. The images shown are chosen at random, not cherry-picked. We can see that although [13] is a specifically designed model for face inpainting, our model generates much better results.



(a) Input (b) GFC [13] (c) Ours (d) Input (e) GFC [13] (f) Ours

Fig. 5. Face completion results comparing with [13].

Quantitative Evaluation Table 3 shows quantitative comparison between CAF, CE, GLI and our approach. The values are computed based a random subset of 200 images selected from the test set. Both the ℓ_1 and ℓ_2 errors are

computed with pixel values normalized between $[0, 1]$ and are summed up using all pixels of the image. We can see that our method performs better than other methods in terms of SSIM and ℓ_1 error. For ℓ_2, other methods have smaller errors, possibly due to the fact that our model is trained with perceptual loss rather than ℓ_2 loss. In addition, from the numerical values, we can see that procedural fine-tuning further reduces the error of the generator head. This is also validated by qualitative improvements shown in Fig. 4.1 and Fig. 4.

Method	Mean ℓ_1 Error	Mean ℓ_2 Error	SSIM
CAF [6]	968.8	**209.5**	0.9415
	1660	**363.5**	0.9010
CE [11]	2693	545.8	0.7719
	N/A	N/A	N/A
GLI [10]	868.6	269.7	0.9452
	1640	378.7	0.9020
Ours (Generator Head)	913.8	245.6	0.9458
	1629	439.4	0.9073
Ours (Final)	**838.3**	253.3	**0.9486**
	1609	427.0	**0.9090**

Table 3. Numerical comparison between CAF, CE and GLI, our generator head results and our final results. Up/down are results of center/random region completion. Note that for SSIM, larger values mean greater similarity in terms of content structure and are indicators of better performance.

User Study To more rigorously evaluate the performance, we conduct a user study based on the random hole results. We ask 20 users to compare the results, by giving each user 30 image sets to rank the visual quality. Each set contains NPS, GLI, and our result respectively. The survey results give convincing evidence that our method works better than other approaches. Among 300 comparisons, 72.3% of the time our results are ranked highest. In particular, our results are overwhelmingly better than NPS, and in 95.8% of the comparisons our results are better. 73.5% of our results are better than GLI, and 15.5% have similar qualities. This shows that our results have significant advantages over GLI, and are also better or comparable with CAF most of the time.

4.3 Ablation Study

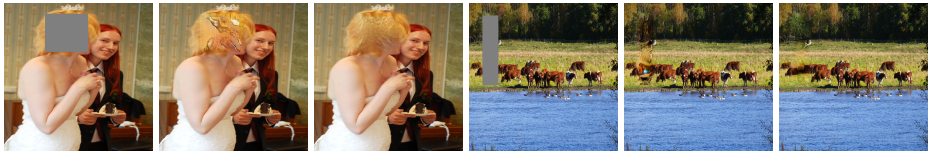
Comparison of Convolutional Layer Choosing the proper convolutional layer improves the inpainting quality and also reduces the noise. We consider three types of convolutional layers: vanilla, dilated [37] and interpolated [38], and train three networks to specifically test the effects of different convolutional layers. Fig. shows an example of qualitative comparisons. We can see that using dilation significantly improve the inpainting quality comparing with vanilla convolutional layer and interpolated convolutional layer, as the latter generates results that are over-smoothed.

Effect of Procedural Training Qualitative comparisons in Sec. 4.2 shows that procedurally adding residual blocks to fine-tune improves the results. However,

(a) (b) (c) (d) (e)

Fig. 6. Effects of different types of convolutional layers. (a) Input. (b) Vanilla (c) Interpolated (d) Dilated+Interpolated (e) Dilated (ours).

one may wonder whether we can directly train a deeper network. To find out, we trained the same number of iterations for a network with 12 residual blocks from scratch. Fig. 7 shows two examples of the results of inpainting using the trained model comparing with the results of the block-wise trained model. For all the test images, we found that increasing number of layers actually has a negative effect on inpainting, due to a variety of factors such as gradient vanishing, optimization difficulty, etc. This experiment demonstrates the necessity and importance of the procedural training scheme.



(a) Input (b) DT (c) PT (d) Input (e) DT (f) PT

Fig. 7. Result comparison between directly training a network of 12 blocks ((b),(e)) and procedural training ((c), (f)).

Effect of Adversarial Loss Annealing We investigate the effect of adversarial loss annealing in terms of reducing the noise level. As discussed in Sec. 3.3, the adversarial loss becomes dominant at the end phase of training. This leads to noisy results with perceivable artifacts. We show two randomly selected examples of using and not using adversarial loss annealing in Fig. 8 From the qualitative comparison, we observe that using adversarial loss annealing not only reduces the noise level but does not sacrifice the overall sharpness and local details.

Comparing Patch Perceptual Losses with ℓ_2 We compare the quality of inpainting that is trained with patch perceptual loss and ℓ_2 loss. ℓ_2 is used to train

(a) Input (b) w/o AAL (c) AAL (d) Input (e) w/o AAL (f) AAL

Fig. 8. Result comparison between training without AAL and with AAL.

CE and GLI, but it does not correspond well to human perception of similarity, as discussed in Sec. 3.3. We investigate how our Patch Perceptual Loss compares with ℓ_2 loss by training the same network and only using different reconstruction loss. From the test cases, we see that our results are overwhelmingly better than ℓ_2 results in terms of sharpness and coherence with context. We leave detailed discussion of training losses to Supplementary Materials.

(a) Input (b) ℓ_2 (c) Ours (d) Input (e) ℓ_2 (f) Ours

Fig. 9. Effects of different types of reconstruction losses. Zoom in for best quality.

4.4 Interactive Guided Inpainting

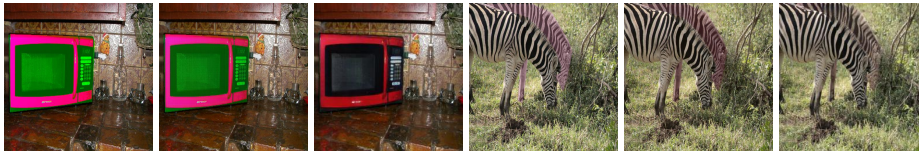
Guided inpainting refers to the task of using another image as guidance to fill in the missing part of the original image. This is extremely useful in practice as people often want to add new objects or replace sceneries. It can also fill in large holes, which is a challenging task without using another image to guide. Interactive guided inpainting allows the user to freely choose the guide image the and region of interest.

Here we consider the scenario of object-based guided inpainting. Specifically, we assume the user would like to add to the input image with objects from another guide image. Given an input image I and a guide image I_g, we allow the user to select a region of I_s by dragging a bounding box B_g containing the object that he desires to add. Note that unlike previous settings of image composition, we do not require the user to accurately segment the object, but instead only providing a bounding box is sufficient. This greatly reduces the workload and simplifies the task. Next, we perform segmentation use a network to extract the object foreground. The segmentation network is adapted from Mask R-CNN [51] and trained on COCO, but is agnostic to object categories and only considers the foreground/background classification. Finally, we resize and paste B_g to I.

To make the composition look natural and realistic, we need to inpaint the background of B and also perform harmonization on object foreground, such that its appearance is compatible with I.

To accomplish this task, we need to address two separate problems: inpainting and harmonization. Our model can be easily extended to train for both tasks at the same time, only requiring changing the input and output. Our data acquisition is similar to [9]. Specifically, at each iteration given the input image I, we randomly select another image I_g from the dataset. We then select and segment an object I_o from I, based on the segmentation mask of COCO. We then transfer the color from I_g to I_o, and paste I_o onto I_g at a random location. Finally, we crop a bounding box I_b from I_g that contains I_o, and paste I_b back to I. The result, together with the foreground/background segmentation mask of I_b, is given to the network as input. The model is modified to output two images, one for inpainting result and the other for harmonization result. We use the original I as the ground truth for both tasks.

The network jointly trained for inpainting and harmonization performs well in both tasks. Fig. 10 shows that it adjusts the color better than [9], which is the state-of-the-art deep harmonization method. Fig. 11 shows the interactive guided inpainting results.

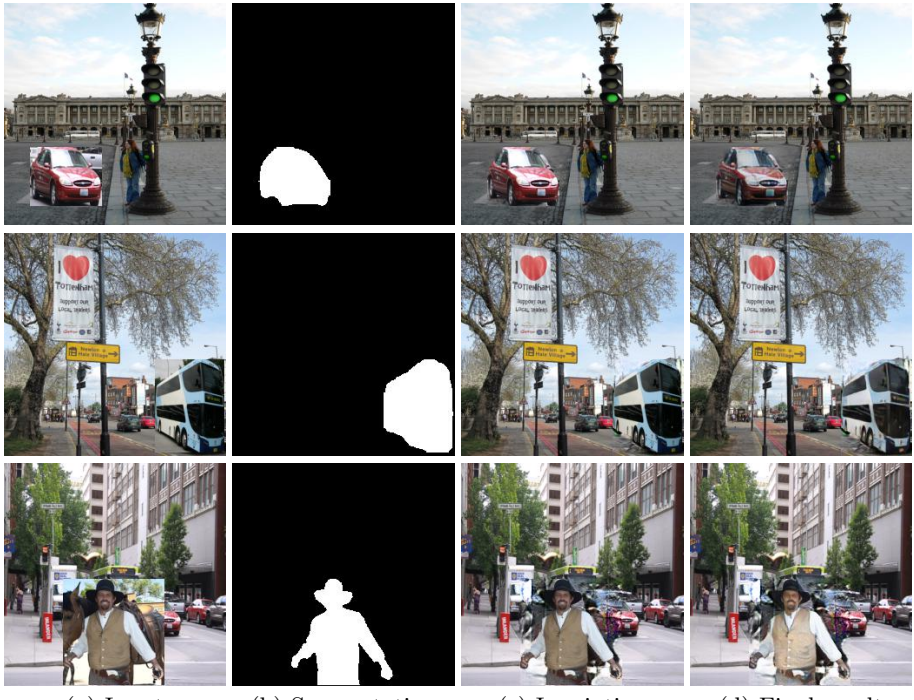


(a) Input (b) DH [9] (c) Ours (d) Input (e) DH [9] (f) Ours

Fig. 10. Examples of image harmonization results. For (a) and (d), the microwave and the zebra on the back have unusual color. Our method is able to adjust their appearance such that they look coherent and realistic. Our harmonization results are also more natural and plausible than state-of-the-art method [9].

5 Conclusions

We proposed a novel model and training scheme to address the inpainting problem and achieve excellent performance on several tasks, including general inpainting, image harmonization and guided inpainting. Although we only explore the ability of our approach in image editing related tasks, we believe it is a general methodology that could be easily applied to other generative problems, such as image translation, image synthesis, etc. As future work, we plan to study the effectiveness of the proposed network and losses, and especially the procedural training scheme on a larger scope.



(a) Input (b) Segmentation (c) Inpainting (d) Final result

Fig. 11. Examples of interactive guided inpainting result. The segmentation mask is given by our foreground/background segmentation network trained on COCO. The final result combines the outputs of harmonization and inpainting.

References

1. Gatys, L., Ecker, A.S., Bethge, M.: Texture synthesis using convolutional neural networks. In: Advances in Neural Information Processing Systems. (2015) 262–270
2. Komodakis, N.: Image completion using global optimization. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on. Volume 1., IEEE (2006) 442–452
3. Hays, J., Efros, A.A.: Scene completion using millions of photographs. In: ACM Transactions on Graphics (TOG). Volume 26., ACM (2007) 4
4. Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C.: Image inpainting. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co. (2000) 417–424
5. Wexler, Y., Shechtman, E., Irani, M.: Space-time video completion. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on. Volume 1., IEEE (2004) I–I
6. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: Patchmatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics-TOG **28**(3) (2009) 24
7. Bertalmio, M., Vese, L., Sapiro, G., Osher, S.: Simultaneous structure and texture image inpainting. IEEE transactions on image processing **12**(8) (2003) 882–889
8. Wilczkowiak, M., Brostow, G.J., Tordoff, B., Cipolla, R.: Hole filling through photomontage. In: BMVC 2005-Proceedings of the British Machine Vision Conference 2005. (2005)
9. Tsai, Y.H., Shen, X., Lin, Z., Sunkavalli, K., Lu, X., Yang, M.H.: Deep image harmonization. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017)
10. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion. ACM Transactions on Graphics (TOG) **36**(4) (2017) 107
11. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: Feature learning by inpainting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2536–2544
12. Yeh, R., Chen, C., Lim, T.Y., Hasegawa-Johnson, M., Do, M.N.: Semantic image inpainting with perceptual and contextual losses. arXiv preprint arXiv:1607.07539 (2016)
13. Li, Y., Liu, S., Yang, J., Yang, M.H.: Generative face completion. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Volume 1. (2017) 6
14. Yang, C., Lu, X., Lin, Z., Shechtman, E., Wang, O., Li, H.: High-resolution image inpainting using multi-scale neural patch synthesis. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Volume 1. (2017) 3
15. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. (2014) 2672–2680
16. Denton, E.L., Chintala, S., Fergus, R., et al.: Deep generative image models using a laplacian pyramid of adversarial networks. In: Advances in neural information processing systems. (2015) 1486–1494
17. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)

18. Zhao, J., Mathieu, M., LeCun, Y.: Energy-based generative adversarial network. arXiv preprint arXiv:1609.03126 (2016)
19. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. arXiv preprint arXiv:1701.07875 (2017)
20. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.: Improved training of wasserstein gans. arXiv preprint arXiv:1704.00028 (2017)
21. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196 (2017)
22. Kim, J., Kwon Lee, J., Mu Lee, K.: Accurate image super-resolution using very deep convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1646–1654
23. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a deep convolutional network for image super-resolution. In: European Conference on Computer Vision, Springer (2014) 184–199
24. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. arXiv preprint arXiv:1609.04802 (2016)
25. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. arXiv preprint arXiv:1611.07004 (2016)
26. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv:1703.10593 (2017)
27. Zhu, J.Y., Krähenbühl, P., Shechtman, E., Efros, A.A.: Learning a discriminative model for the perception of realism in composite images. In: Computer Vision (ICCV), 2015 IEEE International Conference on. (2015)
28. Elad, M., Starck, J.L., Querre, P., Donoho, D.L.: Simultaneous cartoon and texture image inpainting using morphological component analysis (mca). Applied and Computational Harmonic Analysis **19**(3) (2005) 340–358
29. Reinhard, E., Adhikhmin, M., Gooch, B., Shirley, P.: Color transfer between images. IEEE Computer graphics and applications **21**(5) (2001) 34–41
30. Sunkavalli, K., Johnson, M.K., Matusik, W., Pfister, H.: Multi-scale image harmonization. In: ACM Transactions on Graphics (TOG). Volume 29., ACM (2010) 125
31. Pérez, P., Gangnet, M., Blake, A.: Poisson image editing. ACM Transactions on graphics (TOG) **22**(3) (2003) 313–318
32. Tao, M.W., Johnson, M.K., Paris, S.: Error-tolerant image compositing. In: European Conference on Computer Vision, Springer (2010) 31–44
33. Tsai, Y.H., Shen, X., Lin, Z., Sunkavalli, K., Yang, M.H.: Sky is not the limit: semantic-aware sky replacement. ACM Trans. Graph. **35**(4) (2016) 149–1
34. Johnson, M.K., Dale, K., Avidan, S., Pfister, H., Freeman, W.T., Matusik, W.: Cg2real: Improving the realism of computer generated images using a large collection of photographs. IEEE Transactions on Visualization and Computer Graphics **17**(9) (2011) 1273–1285
35. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
36. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778
37. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122 (2015)

38. Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill (2016)
39. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. arXiv preprint arXiv:1801.03924 (2018)
40. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on, IEEE (2016) 2414–2423
41. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: European Conference on Computer Vision, Springer (2016) 694–711
42. Dosovitskiy, A., Brox, T.: Generating images with perceptual similarity metrics based on deep networks. In: Advances in Neural Information Processing Systems. (2016) 658–666
43. Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. In: The IEEE International Conference on Computer Vision (ICCV). Volume 1. (2017)
44. Zhang, H., Xu, T., Li, H., Zhang, S., Huang, X., Wang, X., Metaxas, D.: Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. arXiv preprint arXiv:1612.03242 (2016)
45. Durugkar, I., Gemp, I., Mahadevan, S.: Generative multi-adversarial networks. arXiv preprint arXiv:1611.01673 (2016)
46. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. arXiv preprint arXiv:1711.11585 (2017)
47. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision, Springer (2014) 740–755
48. Zhou, B., Khosla, A., Lapedriza, A., Torralba, A., Oliva, A.: Places: An image database for deep scene understanding. arXiv preprint arXiv:1610.02055 (2016)
49. Ziwei Liu, Ping Luo, X.W., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of International Conference on Computer Vision (ICCV). (2015)
50. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3) (2015) 211–252
51. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Computer Vision (ICCV), 2017 IEEE International Conference on, IEEE (2017) 2980–2988