Generator Pyramid for High-Resolution Image Inpainting

Leilei Cao, Tong Yang
OPPO Research

Yixu Wang
Johns Hopkins University

Bo Yan, Yandong Guo
OPPO Research

Abstract

Inpainting high-resolution images with large holes challenges existing deep learning based image inpainting methods. We present a novel framework—PyramidFill for high-resolution image inpainting task, which explicitly disentangles content completion and texture synthesis. PyramidFill attempts to complete the content of unknown regions in a lower-resolution image, and synthesis the textures of unknown regions in a higher-resolution image, progressively. Thus, our model consists of a pyramid of fully convolutional GANs, wherein the content GAN is responsible for completing contents in the lowest-resolution masked image, and each texture GAN is responsible for synthesizing textures in a higher-resolution image. Since completing contents and synthesising textures demand different abilities from generators, we customize different architectures for the content GAN and texture GAN. Experiments on multiple datasets including CelebA-HQ, Places2 and a new natural scenery dataset (NSHQ) with different resolutions demonstrate that PyramidFill generates higher-quality inpainting results than the state-of-the-art methods. To better assess high-resolution image inpainting methods, we will release NSHQ, high-quality natural scenery images with high-resolution 1920×1080.

1. Introduction

Image inpainting, as a fundamental low-level vision task, has attracted much attention from academic and industry. A wide range of vision and graphics applications refer to image inpainting, e.g., object removal[2, 1], image restoration[4], manipulation[18], and super-resolution[26]. Image inpainting aims to synthesize alternative contents in missing regions of an image, which is visually realistic and semantically correct[29].

Existing inpainting methods generally fall into two categories: copying from low-level image features and generating new contents from high-level semantic features. The former ones attempt to fill holes by matching and copying patches from known regions[1] or external database images[6]. These approaches are effective for easy cases, e.g., uniform textured background inpainting. However, they cannot solve challenge cases where missing regions involve complex textures or nonrepetitive structures, due to lacking of high-level semantic understanding[29]. The latter approaches learn to capture the context representation of known regions and synthesize the missing contents by using deep convolutional neural networks (CNN) in an end-to-end training manner. Context Encoder[19] is the first CNN-based method to solve the image inpainting task, wherein a deep convolution encoder-decoder architecture was proposed and an adversarial loss[5] was added to generate visually realistic results. The encoder-decoder architecture was extended by many CNN-based inpainting models to get more reasonable and fine-detailed contents[7, 29, 30, 15, 9, 28].

Original Input PConv Ours

Figure 1. Free-form inpainting results on 512×512 images by PConv and our model. Zoom-in to see the details.

However, the end-to-end training encoder-decoder architecture still remains challenging to fill large holes in high resolution images. As shown in Figure 1, the results of PConv[15] are obtained from its online demo[1], which illustrates that PConv fails to generate reasonable contents for these two cases. Because the guidance for filling the holes has lost as some regions in holes are far away from the surrounding known regions, which causes the generator to produce semantically ambiguous contents or visually artifacts for the holes[14]. In addition, higher resolution im-

[1]https://www.nvidia.com/research/inpainting

ages make the discriminator easier to tell the fully inpainted images apart from training images, thus drastically amplifying the gradient problem[11]. An alternative way is to use two encoder-decoder architectures to separately fill contents and textures for the unknown regions[29]. Even so, only a few methods can process images with resolution smaller than 512 $\times$ 512, wherein the masked area is generally less than 25%[30].

We observe that filling the contents and textures demand quite different abilities from the generator. Content completion relies more on capturing the high-level semantics and the global structure of the image, yet low-level features and local texture statistics of the image is more critical for the texture synthesis. Furthermore, content completion can be regarded as an image generation task[5], and texture synthesis can be treated as an image to image translation task[8].

Motivated by the observation that high-resolution image inpainting could be disentangled into content completion and texture synthesis, we propose a simple yet powerful high-resolution image inpainting framework named PyramidFill, which attempts to fill large holes in images with high resolution reaching to 1024$\times$1024. Our key insight is that we can fill the contents for the easier low-resolution images and synthesize the textures for the higher-resolution details progressively, wherein the high-resolution image is formed into a pyramid of different scale images by downsampling. PyramidFill consists of a pyramid of fully convolutional Generative Adversarial Networks (GANs), wherein the content GAN is responsible for generating contents in the lowest-resolution masked image in image pyramid, and each texture GAN is responsible for synthesizing textures in a higher-resolution image.

Our major contributions can be summarized as follows:

- We provide a new perspective that high-resolution image inpainting could be disentangled into low-resolution content completion and higher-resolution texture synthesis.
- Following the new perspective, we design a novel framework consisting of a pyramid of GANs, wherein the content GAN is responsible for generating contents in the lowest-resolution masked image in image pyramid, and each texture GAN is responsible for synthesizing textures in a higher-resolution image.
- We introduce a new dataset of natural scenery with high resolution 1920 $\times$ 1080 for real image inpainting applicatons.

2. Related Work

2.1. Deep Image Inpainting

A variety of CNN-based approaches have been proposed for image inpainting. Pathak et al.[19] first introduced an eoncoder-decoder architecture for the image inpainting task, as well as a pixel-wise reconstruction loss and an adversarial loss. Based on Pathak's work, Iizuka et al.[7] proposed a fully convolutional GAN model with an extra local discriminator to ensure local image coherency. Yang et al.[25] proposed a multi-scale neural patch synthesis approach based on joint optimization of image content and texture constraints. Based on two-stage encoder-decoder architectures, Observing ineffectiveness of CNN in modeling long-term correlations between distant contextual information and the hole regions, Yu et al.[29] presented a novel contextual attention layer to integrate in the second stage, which used the features of known patches as convolutional filters to process the generated patches. The aforementioned approaches were based on vanilla convolutions that treated all input pixels as same valid ones, which is not reasonable for masked holes. To address this limitation, Liu et al.[15] proposed a partial convolutional layer for irregular holes in image inpainting, comprising a masked and re-normalized convolution operation followed by a mask-update step. Partial Convolutions could be viewed as hard-mask convolutional layers, Yu et al.[30] proposed gated convolution to learns a dynamic feature gating mechanism for each channel and each spatial location across all layers, which could be viewed as a soft-mask convolutional layer. Yang et al.[27] proposed a multi-task learning framework to incorporate the image structure knowledge to assist image inpainting, which trained a shared generator to simultaneous complete the masked image and corresponding structures— edge and gradient. Liu et al.[16] proposed a mutual encoder-decoder CNN for joint recovering structures and textures, which used CNN features from the deep and shallow layers of the encoder to represent structures and textures of an input image, respectively.

2.2. High-Resolution Image Inpainting

Most recently, a few works have been presented for high-resolution image inpainting. Instead of directly filling holes in high-resolution images, Yi et al.[28] proposed a contextual residual aggregation mechanism that could produce high-frequency residuals for missing contents by weighted aggregating residuals from contextual patches, thus only requiring a low-resolution prediction from the network. Zeng et al.[32] presented a guided upsampling network to generate high-resolution image inpainting results, by extending the contextual attention module which borrowed high-resolution feature patches in the input image.

2.3. Pyramid in Image Generation

Pyramid has been explored widely in the image generation task. Denton et al.[3] introduced a cascade of CNNs within a Laplacian pyramid framework to generate images in a coarse-to-fine fashion. At each level of the pyramid, a

separate generation model was trained using the GAN approach. Shaham et al.[21] introduced SinGAN to learn from a single natural image, which contained a pyramid of fully convolutional GANs, each was responsible for learning the patch distribution at a different scale of the image. Inspired by classical image pyramid representations, Shocher et al.[22] proposed a semantic generation pyramid framework to generate diverse image samples, which utilized the continuum of semantic information encapsulated in deep features, ranging from low-level textural information contained in fine features to high-level semantic information contained in deeper features.

3. Method

In this section, we first introduce the pipeline of the proposed PyramidFill, and then present the generator and discriminator networks in each level, as well as the loss functions.

3.1. Pipeline of PyramidFill

Figure 2 illustrates the pipeline of our proposed algorithm PyramidFill, which consists of a pyramid of PatchGANs[13, 8]: $\{G_i, D_i\}$, $i = 0, 1, 2, 3, 4$. For training, given a high-resolution image x, we sample a binary image mask m at a random location (e.g., center area in Figure 2). Input image z is masked from the original image as $z = x \odot (1 - m) + m$. The original image, input image, mask are separately downsampled to be pyramids of images by a factor r^{4-i} (we choose $r = 2$): x_i, z_i, m_i $i = 0, 1, 2, 3, 4$. Each Generator G_i is responsible of filling the corresponding-scale masked image z_i, trained against the image x_i, where G_i learns to fool the discriminator D_i.

The filling of a high-resolution image sample starts with the lowest-scale image x_0, wherein $\{G_0, D_0\}$ are trained to complete the contents, and then progressively synthesis the finer textures at the higher-scale image by training $\{G_i, D_i\}$, $i = 1, 2, 3, 4$. The generator G_0 takes concatenation of z_0 and m_0 as input, and output the predicted image $x_0^{'}$ with the same size as input. We then replace the masked region of z_0 using the predicted image to get the inpainting result y_0,

$$y_0 = z_0 \odot (1 - m_0) + G_0([z_0, m_0]) \odot m_0 \qquad (1)$$

After training the generator G_0, the inpainting result y_0 and input image z_1 are given to the generator G_1 to output the predicted image $x_1^{'}$ with the same size as z_1, and it replaces the masked region of z_1 to get the inpainting result y_1. Training the other PatchGANs is similar to training $\{G_1, D_1\}$, given the corresponding input image z_i and the lower-scale inpainting result y_{i-1} to get the current-scale inpainting result y_i, i.e.,

$$y_i = z_i \odot (1 - m_i) + G_i(z_i, y_{i-1}) \odot m_i, i = 1, 2, 3, 4. \qquad (2)$$

3.2. Generators and Discriminators

Since completing contents and synthesising textures demand quite different abilities from generators, we customize different architectures for $\{G_0, D_0\}$ and $\{G_i, D_i\}$, $i = 1, 2, 3, 4$.

Figure 3 shows the architecture of a PatchGAN for completing contents on the lowest-scale input image z_0. For the generator network, we maintain the spatial size of the feature maps since of low-scale input images, instead of encoder-decoder networks used in most of image inpainting methods. The input images are first passed through four gated convolutional layers[30], which are then split into two branches along the channel dimension. Four dilated gated convolutional layers are adopted in the upper branch to expand the size of receptive fields for exploring the global structures of input images, four gated convolutional layers are simultaneously adopted in the lower branch for exploiting fine contents. Feature maps from two branches are then concatenated to proceed the last four gated convolutional layers to predict the contents in the masked region. The discriminator network consists of eight convolutional layers, wherein no downsampling operations are adopted to ensure the spatial size of the output as the same with the input image. It can better capture fine details in the lowest-scale image.

The architecture of a PatchGAN for synthesising textures is presented in Figure 4, and $\{G_1, D_1\}$, $\{G_2, D_2\}$, $\{G_3, D_3\}$, $\{G_4, D_4\}$ share this same architecture yet progressively training. There are two stages in the generator, and we call the first stage as the super-resolution network, the second stage as the refinement network. The super-resolution network predicts a higher-resolution image from its lower-resolution counterpart which is taken from the inpainting result of the lower-scale generator. We use the predicted higher-resolution image to replace the masked region of the corresponding-scale input image, which then passes through the refinement network to output a finer result. Inspired by ESRGAN[24] and SRResNet[12], we design a simple yet powerful super-resolution network, wherein all vanilla convolutions in RRDB module[24] are replaced with gated convolutions, and we only adopt two RRDB modules. The upsampling operator uses the sub-pixel layer with the factor 2 to increase the resultion of the input image. In the refinement network, there are only six gated convolutional layers with a shortcut connection within the middle four layers. In the discriminator, we use four strided convolutions with stride 2 to reduce the spatial size of feature maps due to memory cost.

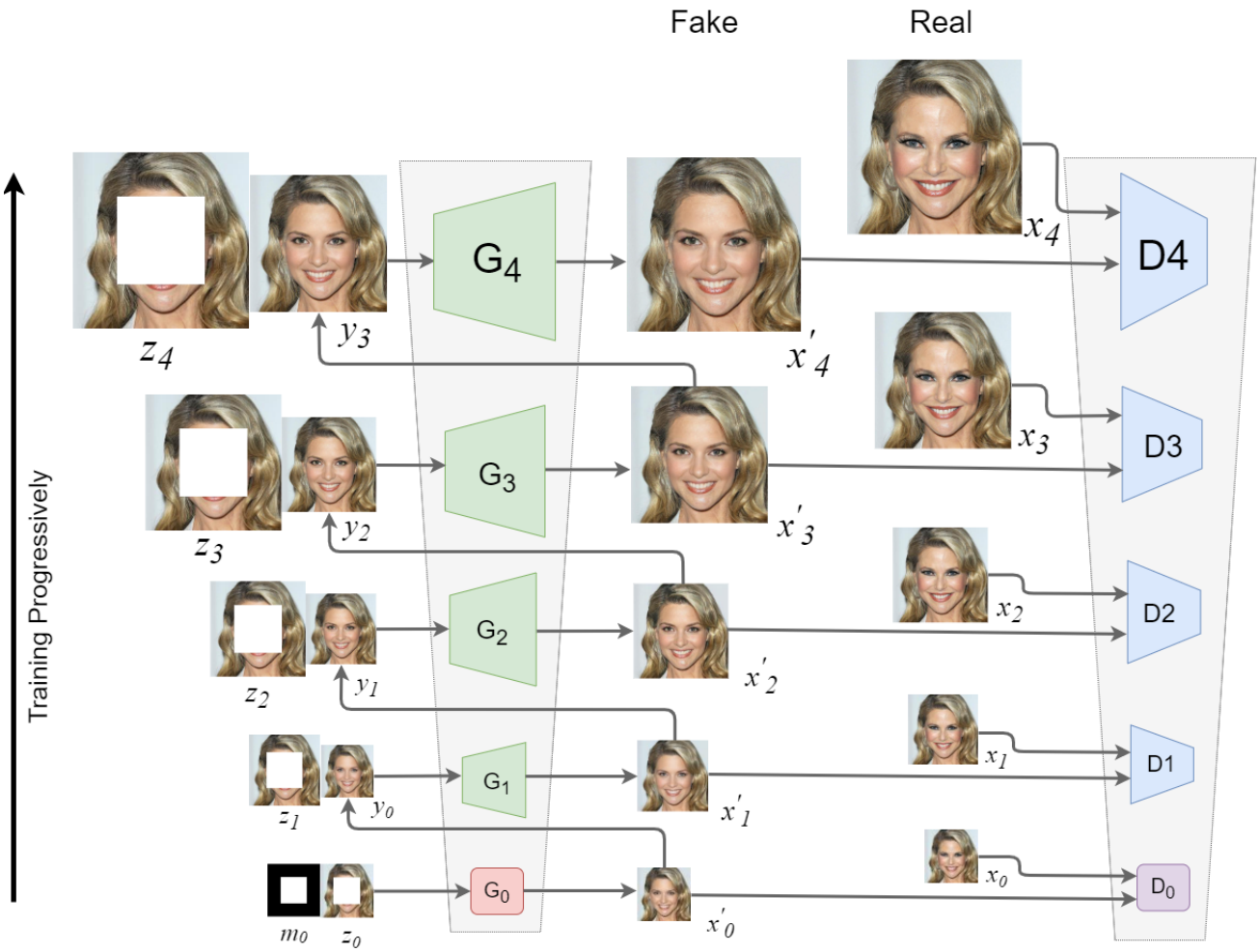

Figure 2. Pipeline of our high-resolution image inpainting algorithm—PyramidFill. Our model consist of a pyramid of PatchGANs, where training are progressive.

从小到大，渐进式补全

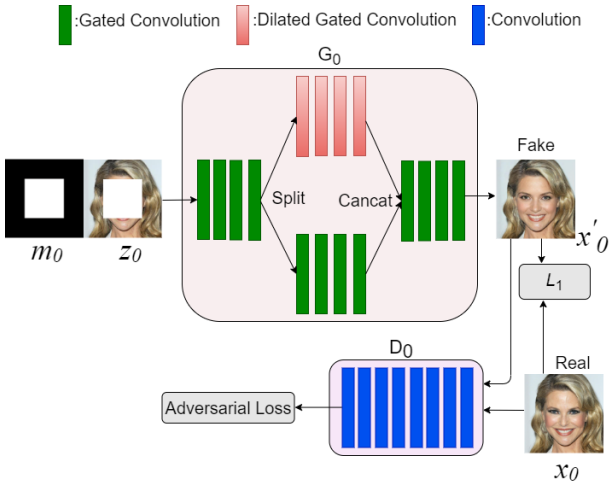

Figure 3. Architecture of a PatchGAN for completing contents.

结构两分支，dilate和Gated结合，无attention

3.3. Loss Function

To stabilize the training of PatchGANs, the spectral normalization[17] is applied in all discriminators. We use a hinge loss[30] as objective function for all PatchGANs $\{G_i, D_i\}$, $i = 0, 1, 2, 3, 4$:

$$\begin{aligned}\mathcal{L}_{D_i}^{adv} = &\mathbb{E}_{x_i \sim p_{data}(x_i)}[\text{ReLU}(\mathbb{1} - D_i(x_i))] \\ &+\mathbb{E}_{z_i \sim p_{z_i}(z_i)}[\text{ReLU}(\mathbb{1} + D_i(G_i(z_i)))]\end{aligned} \tag{3}$$

$$\mathcal{L}_{G_i}^{adv} = -\mathbb{E}_{z_i \sim p_{z_i}(z_i)}[D_i(G_i(z_i))] \tag{4}$$

where x_i and z_i represent the real image and the masked input image, respectively. D_i and G_i denote the spectral-normalized discriminator and the generator, respectively.

Inspired by U-Net GAN[20], we use a per-pixel consistency regularization technique when training D_0, encouraging the discriminator to focus more on semantic and structural changes between real and fake images. This technique

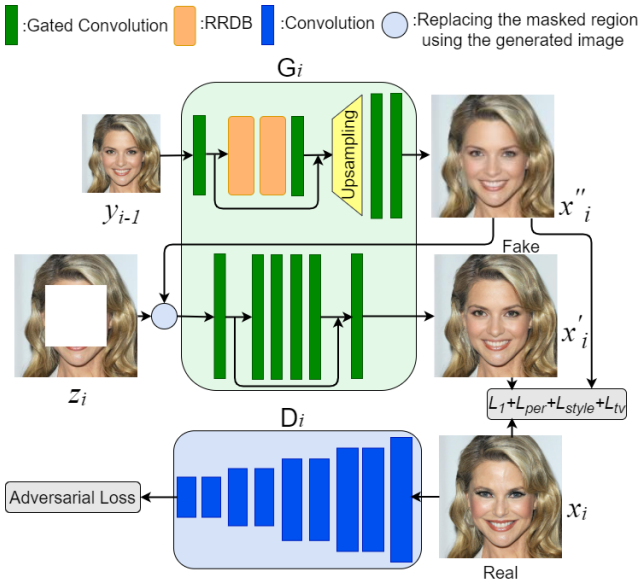

Figure 4. Architecture of a PatchGAN for synthesising textures.

迭代补全的流程

can further enhance the quality of the generated sample $x_0^{'}$. We train the discriminator to provide consistent per-pixel predictions, by introducing the consistency regularization loss term in the discriminator objective function:

$$\mathcal{L}_{D_0}^{cons} = \Big\| D_0(y_0) - \Big(D_0(x_0) \odot (1 - m_0) + D_0(x_0^{'}) \odot m_0 \Big) \Big\|^2 \tag{5}$$

where $\|\cdot\|$ denotes the L^2 norm. This consistency loss is taken between the per-pixel output of D_0 on the composing image y_0 and the composite of output of D_0 on real and fake images, penalizing the discriminator for inconsistent predictions.

Except the adversarial loss, a pixel reconstruction loss is introduced for training the generators, because the task of generators is not only fool the discriminators but also to generate image being similar to the real image.

$$\mathcal{L}_{x_{gen}}^{re} = \frac{1}{N_{x_{gen}}} \left\| (x_{gen} - x_i) \right\|_1 \tag{6}$$

where $N_{x_{gen}}$ is the number elements in the image x_{gen}, which is the predicted image by the generators.

We introduce additional losses for training the texture generators, perceptual loss[10] and style loss[15, 9]. The perceptual loss penalizes results that are not perceptually similar by computing L^1 distance between feature maps of a pretrained network,

$$\mathcal{L}_{x_{gen}}^{per} = \sum_q \frac{1}{N_q} \left\| \phi_q(x_{gen}) - \phi_q(x_i) \right\|_1 \tag{7}$$

where N_q indicates the number of elements in the q-th layer, and ϕ_q is the feature map of the q-th layer extracted from the VGG-16 network pretrained on ImageNet[23], and we choose the feature maps from layers $pool1$, $pool2$ and $pool3$. Style loss compares the content of two images by using Gram matrix. Given feature maps of sizes $C_q \times H_q \times W_q$, style loss is computed as follows:

$$\mathcal{L}_{x_{gen}}^{style} = \left\| G_q^{\phi}(x_{gen}) - G_q^{\phi}(x_i) \right\|_1 \tag{8}$$

where G_q^{ϕ} is a $C_q \times C_q$ Gram matrix constructed from feature maps ϕ_q, which are the same with that used in the perceptual loss.

The overall loss functions for training $\{G_0, D_0\}$ are shown as below:

$$\begin{aligned} \mathcal{L}_{D_0} &= \mathcal{L}_{D_0}^{adv} + \mathcal{L}_{D_0}^{cons} \\ \mathcal{L}_{G_0} &= \mathcal{L}_{G_0}^{adv} + \mathcal{L}_{x_0^{'}}^{re} \end{aligned} \tag{9}$$

And the whole loss functions for training $\{G_i, D_i\}, i = 1, 2, 3, 4$ are as follows:

$$\mathcal{L}_{D_i} = \mathcal{L}_{D_i}^{adv} \tag{10}$$

$$\begin{aligned} \mathcal{L}_{G_i} = \lambda_a \mathcal{L}_{G_i}^{adv} + \lambda_r \mathcal{L}_{x_i^{''}}^{re} + \lambda_p \mathcal{L}_{x_i^{''}}^{per} + \lambda_{s_i} \mathcal{L}_{x_i^{''}}^{style} \\ + \lambda_r \mathcal{L}_{x_i^{'}}^{re} + \lambda_p \mathcal{L}_{x_i^{'}}^{per} + \lambda_{s_i} \mathcal{L}_{x_i^{'}}^{style} \end{aligned} \tag{11}$$

For our experiments, we choose $\lambda_a = 0.001$, $\lambda_r = 0.1$, $\lambda_p = 0.1$ for all generators, and $\lambda_{s_1} = 1$, $\lambda_{s_2} = 50$, $\lambda_{s_3} = 120$, $\lambda_{s_4} = 250$ for different generators, respectively.

4. Experiments

We evaluate PyramidFill on three datasets: CelebA-HQ[11], Places2[33], and our new collected NSHQ dataset. CelebA-HQ contains 30,000 high-quality images at 1024×1024 resolution focusing on human faces. For Places2, we randomly select 20 categories from the 365 categories to form a subset of 100,000 images. The NSHQ dataset includes 5000 high-quality natural scenery images at 1920×1080 or 1920×1280 resolution. For Places2 and NSHQ, images are randomly cropped to 512×512 and 1024×1024, respectively. Therefore, there is no $\{G_4, D_4\}$ for training Places2 subset. For all datasets, we randomly partition into 90% of images for training and 10% of images for testing.

We compare PyramidFill with five state-of-the-art approaches: Global&Local[7], DeepFillV1[29], PConv[15], PEN-Net[31], DeepFillV2[30]. When training CelebA-HQ, we use regular masks and random free-form masks[30], wherein the regular masks cover image center with half of image size. For Places2 and NSHQ, random free-form masks[30] are employed for training, while irregular masks from PConv[15] are used for testing.

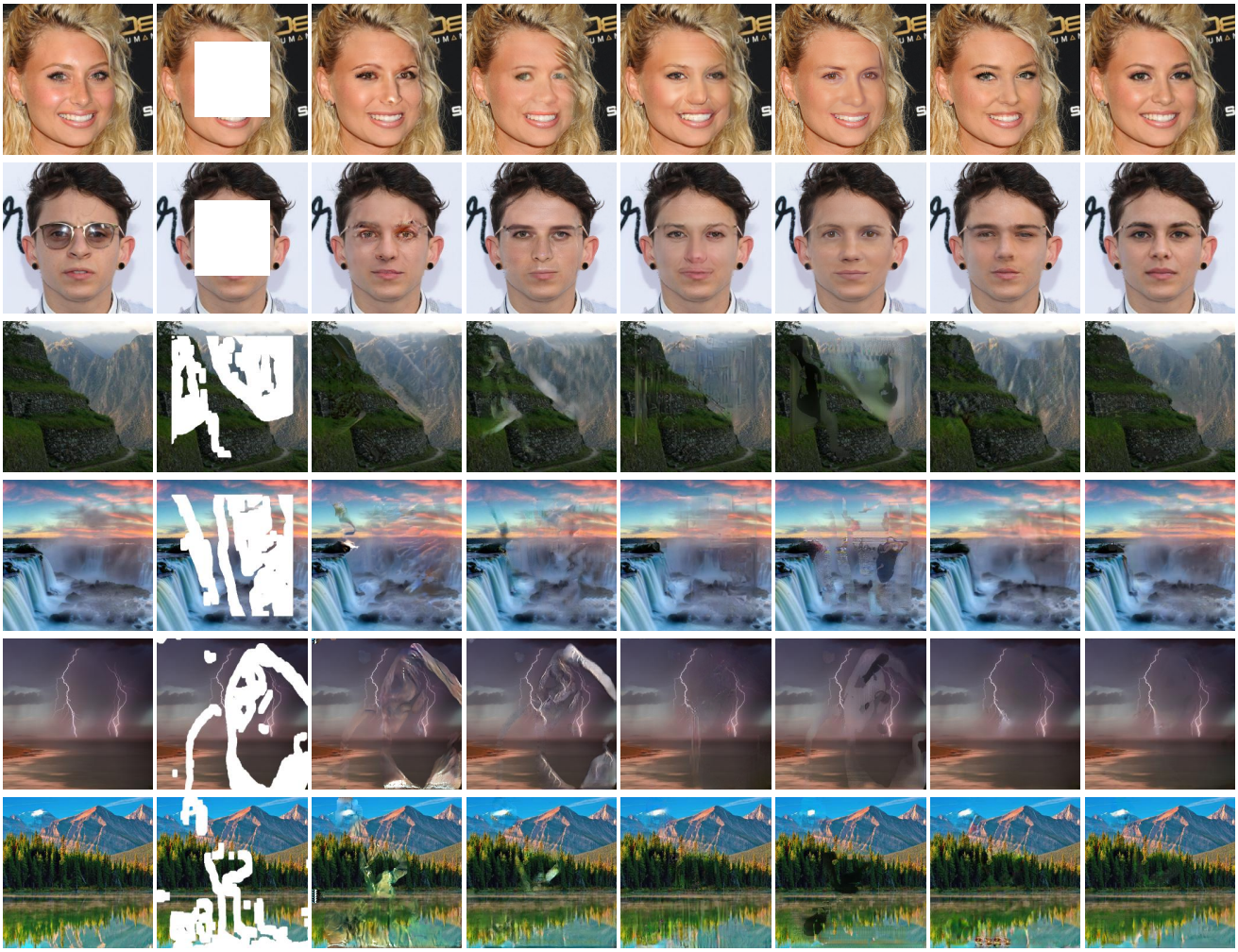

Figure 5. Example cases of qualitative comparisons on CelebA-HQ, Places2 and NSHQ testing sets with 256×256 images.

Table 1. Quantitative comparisons on CelebA-HQ testing set with input size 256×256 resolution, where the inputs are with center hole regions. The ↓ indicates lower is better while ↑ indicates higher is better.

Methods	L1 Loss↓	PSNR↑	SSIM↑
Global&Local	0.0386	24.09	0.8328
DeepFillV1	0.0418	23.75	0.8348
PConv	0.0262	25.18	0.8757
PEN-Net	0.0322	25.06	0.8536
DeepFillV2	0.0367	25.19	0.8527
Ours	**0.0227**	**26.01**	**0.8878**

Table 2. Quantitative comparisons on CelebA-HQ dataset with input size 512×512 resolution, where the inputs are with center hole regions.

Methods	L1 Loss↓	PSNR↑	SSIM↑
PConv	0.0302	23.07	0.8791
DeepFillV2	0.0344	22.23	0.8707
Ours	**0.0237**	**25.61**	**0.8860**

4.1. Quantitative Evaluation

We conduct quantitative comparisons on CelebA-HQ dataset and Places2 subset. For a fair evaluation, all input images are resized to 256×256 and 512×512, respectively. Because a few compared official pre-trained models are only trained on 256×256 images. Table 1 and Table 2 report the quantitative comparisons on CelebA-HQ testing set with 256×256 images and 512×512 images, respectively. The images are masked with center holes, and we use L1 loss, PSNR and SSIM as metrics. It shows that our PyramidFill outperforms existing state-of-the-art methods with evident superiorities. Table 3 and Table 4 present the quantitative comparisons on Places2 testing set with 256×256

Table 3. Quantitative comparisons on Places2 subset with input size 256×256 resolution, where the inputs are with irregular holes.

Mask	Methods	L1 Loss↓	PSNR↑	SSIM↑
0-10%	Global&Local	0.0109	29.43	0.9443
	DeepFillV1	0.0101	31.51	0.9571
	PConv	0.0091	32.27	0.9487
	PEN-Net	0.0079	32.66	0.9622
	DeepFillV2	0.0069	35.77	0.9727
	Ours	**0.0016**	**38.30**	**0.9864**
10-20%	Global&Local	0.0231	24.90	0.8724
	DeepFillV1	0.0225	25.44	0.8928
	PConv	0.0172	28.13	0.8927
	PEN-Net	0.0190	26.84	0.9017
	DeepFillV2	0.0136	30.22	0.9311
	Ours	**0.0091**	**31.86**	**0.9162**
20-30%	Global&Local	0.0372	22.42	0.7963
	DeepFillV1	0.0384	22.25	0.8179
	PConv	0.0265	25.68	0.8334
	PEN-Net	0.0331	23.71	0.8287
	DeepFillV2	0.0227	26.85	0.8768
	Ours	**0.0129**	**29.12**	**0.8569**
30-40%	Global&Local	0.0514	20.73	0.7265
	DeepFillV1	0.0544	20.27	0.7471
	PConv	0.0360	23.96	0.7765
	PEN-Net	0.0479	21.66	0.7588
	DeepFillV2	0.0325	24.62	0.8213
	Ours	**0.0132**	**30.06**	**0.8499**
40-50%	Global&Local	0.0665	19.40	0.6559
	DeepFillV1	0.0706	18.90	0.6757
	PConv	0.0475	22.40	0.7121
	PEN-Net	0.0656	19.94	0.6839
	DeepFillV2	0.0441	22.79	0.7596
	Ours	**0.0246**	**26.19**	**0.7680**

images and 512×512 images masked with irregular masks, respectively. The results are categorized according to the ratios of the hole regions versus the image size. It shows that our PyramidFill performs much better than the state-of-the-art methods, especially for filling large holes in images.

4.2. Qualitative Evaluation

The qualitative comparisons on three datasets are shown in Figure5. It shows that Global&Local and DeepFillV1 often generate heavy artifacts, even the holes are not very large. PConv, PEN-Net and DeepFillV2 can generate correct contents for completing faces, yet lacking of detailed textures. By contrast, our model PyramidFill can generate more realistic results on the corrupted faces with finer textures. When dealing with the corrupted images from Places2 and NSHQ, PConv and PEN-Net also produce obvious artifacts. DeepFillV2 and our PyramidFill both can generate reasonable contents and detailed textures, but results from our PyramidFill have better similarity to the original images than results from DeepFillV2.

Table 4. Quantitative comparisons on Places2 subset with input size 512×512 resolution, where the inputs are with irregular holes.

Mask	Methods	L1 Loss↓	PSNR↑	SSIM↑
0-10%	PConv	0.0068	34.05	0.9647
	DeepFillV2	0.0070	34.82	0.9710
	Ours	**0.0012**	**37.93**	**0.9869**
10-20%	PConv	0.0141	29.11	0.9168
	DeepFillV2	0.0146	29.14	0.9272
	Ours	**0.0075**	**31.19**	**0.9263**
20-30%	PConv	0.0238	26.09	0.8584
	DeepFillV2	0.0248	25.80	0.8716
	Ours	**0.0132**	**28.09**	**0.8460**
30-40%	PConv	0.0343	23.96	0.8006
	DeepFillV2	0.0343	23.59	0.8156
	Ours	**0.0131**	**28.92**	**0.8433**
40-50%	PConv	0.0472	22.09	0.7360
	DeepFillV2	0.0487	21.79	0.7548
	Ours	**0.0241**	**25.47**	**0.7549**

4.3. Real Applications

We study real use cases of image inpainting on high-resolution images using our PyramidFill, e.g., freckles removal, face editing, watermark removal, and general object removal in natural scenery. In the first row of Figure 6, our model successfully removes the freckles on the face, as well as changing eyes. In the second row, the watermarks on the original image are removed successfully. In the third row, we remove the giraffes on the grassland and recover the background. In the last row, a boat is removed from the river. All inpainting results are realistic and with fine-grained details. Furthermore, our PyramidFill can also be used for more real applications, e.g., removing wrinkles on faces, changing hairstyles, restoring old photos.

Table 5. Ablation study on CelebA-HQ dataset with input size 64×64 resolution, where the inputs are with center hole regions. Consistency regularization loss can improve the performance of our model.

Methods	L1 Loss↓	PSNR↑	SSIM↑
Ours without $\mathcal{L}_{D_0}^{cons}$	0.0209	27.21	0.9129
Ours	**0.0201**	**27.54**	**0.9172**

4.4. Ablation Study

Consistency regularization loss To demonstrate the effects of consistency regularization loss used in the discriminator D_0, we retrain $\{G_0, D_0\}$ yet without consistency regularization loss on CelebA-HQ dataset masked with center holes. The results are provided in Table 5, which shows that

Figure 6. Inpainting results on 1024×1024 images using our PyramidFill. Zoom-in to see the details. The original images at the first two rows are extracted from CelebA-HQ dataset, and the original images at the last two rows are extracted from NSHQ dataset.

Figure 7. Ablation study of Refinement network in Texture Generators. From left to right, we show the real image, the masked input, the result with a one-stage network in Texture Generators and our result with a two-stage networks.

the consistency regularization loss can definitely improve the performance of PyramidFill on completing contents.

Refinement network In Texture Generators, a refinement network is designed as a second-stage for generating a finer result. We provide ablation experiments on CelebA-HQ dataset with 128×128 images masked with center holes. For fair comparisons, two-stage networks are merged into a one-stage network, wherein the feature maps of the upsampling operator in the super-resolution network are directly inputted to the refinement network, and there is no coarse result to output. As shown in Figure 7, the two-stage network can generate a more photo-realistic inpainting result.

5. Conclusions

We present PyramidFill, a novel framework for the high-resolution image inpainting task. PyramidFill consists of a pyramid of PatchGANs, wherein the content GAN is responsible for generating contents in the lowest-resolution corrupted images, and each texture GAN is responsible for synthesizing textures at higher-resolution images progressively. We customized the generators and discriminators for the content GAN and texture GAN, respectively. Our model was trained on several datasets to evaluate its ability to fill correct contents and realistic textures for high-resolution image inpainting. Quantitative and qualitative results demonstrated the superiority of PyramidFill, comparing with several state-of-the-art methods.

References

[1] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3), 2009. 1

[2] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *CVPR*, 2003. 1

[3] Emily Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. 2015. 2

[4] Wenchao Du, Chen Hu, and Hongyu Yang. Learning invariant representation for unsupervised image restoration. In *CVPR*, 2020. 1

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. 2014. 1, 2

[6] James Hays and Alexei A Efros. Scene completion using millions of photographs. *Communications of the ACM*, 51:87–94, 2008. 1

[7] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Trans. Graph.*, 36(4), 2017. 1, 2, 5

[8] Phillip Isola, Jun-Yan Zhu, and Tinghui Zhou andAlexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 2, 3

[9] Youngjoo Jo and Jongyoul Park. Sc-fegan: Face editing generative adversarial network with user's sketch and color. In *ICCV*, 2019. 1, 5

[10] Johnson Justin, Alahi Alexandre, and Fei-Fei Li. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016. 5

[11] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. 2018. 2, 5

[12] Christian Ledig, Lucas Theis, Ferenc Husz´ar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 3

[13] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *ECCV*, 2016. 3

[14] Jingyuan Li, Ning Wang, Lefei Zhang, Bo Du, and Dacheng Tao. Recurrent feature reasoning for image inpainting. In *CVPR*, 2020. 1

[15] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregualr holes using partial convolutions. In *ECCV*, 2018. 1, 2, 5

[16] Hongyu Liu, Bin Jiang, Yibing Song, Wei Huang, and Chao Yang. Rethinking image inpainting via a mutual encoder-decoder with feature equalizations. In *ECCV*, 2020. 2

[17] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. 2018. 4

[18] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy, and Ping Luo. Exploiting deep generative prior for versatile image restoration and manipulation. In *ECCV*, 2020. 1

[19] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 1, 2

[20] Edgar Schönfeld, Bernt Schiele, and Anna Khoreva. A u-net based discriminator for generative adversarial networks. In *CVPR*, 2020. 4

[21] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *ICCV*, 2019. 3

[22] Assaf Shocher, Yossi Gandelsman, Inbar Mosseri, Michal Yarom, Michal Irani, William T. Freeman, and Tali Dekel. Semantic pyramid for image generation. In *CVPR*, 2020. 3

[23] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 5

[24] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *ECCVW*, 2018. 3

[25] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *CVPR*, 2017. 2

[26] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *CVPR*, 2020. 1

[27] Jie Yang, Zhiquan Qi, and Yong Shi. Learning to incorporate structure knowledge for image inpainting. In *AAAI*, 2020. 2

[28] Zili Yi, Qiang Tang, Shekoofeh Azizi, Daesik Jang, and Zhan Xu. Contextual residual aggregation for ultra high-resolution image inpainting. In *CVPR*, 2020. 1, 2

[29] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative image inpainting with contextual attention. In *CVPR*, 2018. 1, 2, 5

[30] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Free-form image inpainting with gated convolution. In *CVPR*, 2019. 1, 2, 3, 4, 5

[31] Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo. Learning pyramid-context encoder network for high-quality image inpainting. In *CVPR*, 2019. 5

[32] Yu Zeng, Zhe Lin, Jimei Yang, Jianming Zhang, Eli Shechtman, , and Huchuan Lu. High-resolution image inpainting with iterative confidence feedback and guided upsampling. In *ECCV*, 2020. 2

[33] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 5