# Semantic Image Inpainting with Progressive Generative Networks

Haoran Zhang
Hefei University of Technology
Hefei, China
lmc.haoran.edu@gmail.com

Zhenzhen Hu*
Hefei University of Technology
Hefei, China
huzhen.ice@gmail.com

Changzhi Luo
Hefei University of Technology
Hefei, China
changzhiluo@gmail.com

Wangmeng Zuo
Harbin Institute of Technology
Harbin, China
cswmzuo@gmail.com

Meng Wang
Hefei University of Technology
Hefei, China
eric.mengwang@gmail.com

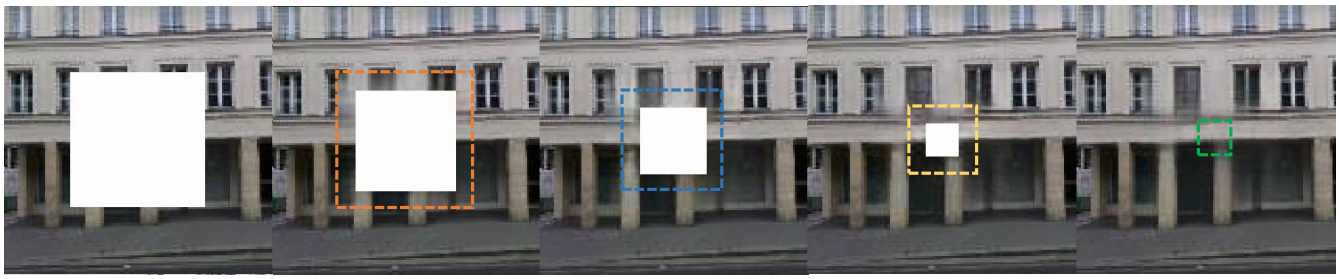


**Figure 1: The processing illustration of progressive image inpainting. Given an image with a missing region (a), the Progressive Generative Networks can fill the content from the outermost ring in the area of dashed line box at each phase and move inward gradually until the entire corrupted region is filled (b) - (e).**

## ABSTRACT

Recently, image inpainting task has revived with the help of deep learning techniques. Deep neural networks, especially the generative adversarial networks (GANs) make it possible to recover the missing details in images. Due to the lack of sufficient context information, most existing methods fail to get satisfactory inpainting results. This work investigates a more challenging problem, e.g., the newly-emerging semantic image inpainting - a task to fill in large holes in natural images. In this paper, we propose an end-to-end framework named progressive generative networks (PGN), which regards the semantic image inpainting task as a curriculum learning problem. Specifically, we divide the hole filling process into several different phases and each phase aims to finish a course of the entire curriculum. After that, an LSTM framework is used to string all the phases together. By introducing this learning strategy, our approach is able to progressively shrink the large corrupted regions in natural images and yields promising inpainting results. Moreover, the proposed approach is quite fast to evaluate as the entire hole filling is performed in a single forward pass. Extensive experiments on Paris Street View and ImageNet dataset clearly demonstrate the superiority of our approach. Code for our models is available at https://github.com/crashmoon/Progressive-Generative-Networks.

## CCS CONCEPTS

• **Computing methodologies** → *Computer vision problems*;

## KEYWORDS

Semantic image inpainting; progressive generative networks; curriculum learning; LSTM

**ACM Reference Format:**
Haoran Zhang, Zhenzhen Hu, Changzhi Luo, Wangmeng Zuo, and Meng Wang. 2018. Semantic Image Inpainting with Progressive Generative Networks. In *2018 ACM Multimedia Conference (MM '18), October 22–26, 2018, Seoul, Republic of Korea.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3240508.3240625

## 1 INTRODUCTION

Image inpainting refers to the process of reconstructing lost or deteriorated parts of images [5]. Although a human can guess the missing content and structure of one image based on the context and personal experience in most cases [19], it is still challenging

*Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*MM '18, October 22–26, 2018, Seoul, Republic of Korea*
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5665-7/18/10…$15.00
https://doi.org/10.1145/3240508.3240625

**Figure 2: Motivation of the PGN approach. Given an image with a large missing region, to inpaint the exterior subregion (marked with the red rectangle) is surely much easier than to inpaint the interior subregion (marked with the green rectangle).**

for a computer to recover the details that are consistent with the visual experience of the human eyes.

The inpainting task has been studied for nearly two decades, and various solutions have been proposed. One popular category of approaches is diffusion-based [5, 6, 17, 27]. It holds the high smoothness assumption of images and aims to propagate information from the known areas to the unknown ones. This kind of approaches cannot synthesize semantic content and thus are not suitable for recovering large corrupted areas. The other mainstream approach is example-based [2, 7, 9, 25, 31]. It holds the spatial coherence assumption, and completes images by copying pixels/patches from the known areas or rearranging their locations. As the searching and optimization processes are computationally expensive, this kind of approaches is usually not suitable for tackling high-resolution images. Obviously, inpainting results obtained by these two kinds of approaches are far from satisfactory.

Recently, image inpainting has revived after several years of decline. This is mainly due to the vigorous development of deep learning techniques, e.g., convolutional neural networks (CNN) and generative adversarial networks (GAN). A seminal inpainting work involving cutting-edge deep learning technique is Context Encoder, which is proposed by Pathak et al. [19]. This work aims at a more challenging inpainting task, i.e., semantic image inpainting, where a large proportion of area is missing in an image. They put the hole filling task into an encoder-decoder pipeline with a pixel-wise loss and an adversarial loss and achieved remarkable reconstruction results. This kind of approach leverages the powerful feature learning function of convolutional neural networks, and it is able to restore corrupted images even when the missing holes are quite large. Moreover, the generated areas are usually perceptually more realistic than those generated using diffusion-based approaches and example-based approaches on account of an adversarial training strategy. Soon after this work, Yang et al. [32], Yeh et al. [33], and Iizuka et al. [13] extended this strategy to other semantic inpainting scenarios and also gained good results.

Despite great progress made in recent years, semantic image inpainting task still remains challenging due to that the generated images must be as visually pleasing as possible and the processing time also must be as short as possible. Optimization-based approaches like [32] can yield naturalistic inpainting results. However, the computational speed is quite slow, and this issue is more severe when the resolution of the images are high. Context Encoder is very fast, but it fails to recover satisfactory structures sometimes.

To tackle these issues, in this paper, we formulate semantic image inpainting as a curriculum learning problem and propose a novel end-to-end framework named progressive generative networks (PGN), which is able to generate more realistic and visual pleasing results as shown in Fig 1. Our approach is motivated by the inpainting process of humans. Given an image with a large missing region, one is definitely not able to fill the region at one go. Inpainting from the exterior to the interior of the corrupted region is surely a better choice than inpainting unorderly (see Fig. 2). Specifically, we divide the inpainting task into several subtasks. The goal of each subtask is to finish a specific part of the inpainting task. All of the subtasks are predefined and highly organized just like the courses in a curriculum. Besides, an LSTM framework is introduced to string all the subtasks together. In this way, the essence learned from the previous subtasks are exploited to ease the learning of subsequent subtasks. Therefore, fine image structures can be recovered using our approach. As our approach does not introduce any optimization operation in the testing phase, the computational speed is very fast and the entire inpainting process can be accomplished in a single end-to-end forward pass. It is worth noting that, our approach is quite different from the aforementioned diffusion-based approaches mentioned before. Diffusion-based inpainting highly relies on known areas in the same image, while our approach is based on encoder-decoder generative networks and is able to synthesize meaningful textures and structures. We evaluate our approach on Paris Street View and ImageNet datasets and get promising results.

The main novelties and contributions of this work are threefold:

(1) We propose a progressive semantic image inpainting framework, an end-to-end neural network, associated with a curriculum learning strategy that can significantly reduce the unnatural transitions at borders of missing image regions.
(2) Different from the previous GAN-based image inpainting algorithms [19, 32], the discriminator of PGN is to discriminate the entire area of generated samples to maintain the structure and texture consistency.
(3) We introduce LSTM architecture into PGN to string all subtasks to control the information flow in the PGN. It is able to avoid the information disturbed and improve the quality of the inpainting image.

The rest of the paper is organized as follows. We review the relevant research work of this paper in Section 2. After that, we detail the proposed PGN approach in Section 3. Extensive experiments on representative inpainting datasets are introduced in Section 4. Section 5 concludes the paper.

## 2 RELATED WORK

In this section, we briefly review the relevant research work of this paper, including deep learning, curriculum learning and image inpainting work.

## 2.1 Deep Learning

Convolutional neural networks (CNN) [11, 15, 24, 26] have demonstrated to be very powerful for feature learning given sufficient data and supervision. However, it is not always the case when there are few or no labelled data. In an unsupervised scenario, autoencoders are more frequently used, especially in generative modelling, which is highly related to the proposed PGN approach. Recurrent neural networks (RNN) [20, 29, 30] are another important branch of deep learning techniques. Long Short-Term Memory (LSTM) [12] is probably the most widely-used RNN approach and it is particularly effective in sequence learning. In very recent years, another technique called generative adversarial networks (GAN) [10] has attracted tremendous interest in deep learning community due to the novel and powerful adversarial training strategy.

## 2.2 Curriculum Learning

Curriculum learning was first introduced by Bengio et al. [4]. It was inspired by the training process of humans, i.e., starting learning easier concepts first (e.g., learning to recognize the meaning of words and phrases) and gradually introducing more complex ones (e.g., learning to understand sentences and passages). The authors provided evidence in many different settings that such a curriculum strategy could benefit machine learning algorithms. Lots of later research works also demonstrated its effectiveness in various areas, including face recognition [23], machine translation [34], sequence prediction [3], etc.

## 2.3 Image Inpainting

Image inpainting research literature can be divided into two aspects: non-semantic inpainting and semantic inpainting. Non-semantic image inpainting approaches refer to those filling missing areas in images only from the aspect of appearance. This kind of approaches mainly consists of the aforementioned diffusion-based approaches and example-based approaches. Although non-semantic image inpainting approaches have been studied for decades [2, 5–7, 9, 17, 25, 27, 31], they cannot generate semantically meaningful content, and thus are not suitable to handle the semantic image inpainting task where large proportions of areas are corrupted in images.

Recent years have witnessed a surge of interest in semantic image inpainting task (filling large holes in images). Pathak et al. [19] were among the pioneers of this challenging task. They brought the adversarial training strategy [10] into a novel encocder-decoder pipeline and got visually pleasing semantic inpainting results. Soon after that, Iizuka [13] extended this work and proposed a locally and globally consistent image inpainting approach. Yang et al. [32] proposed a multi-scale neural patch synthesis approach, which works well in high-resolution semantic inpainting. Yeh et al. [33] also adopted the similar strategy and proposed an optimization based inpainting approach which is able to predict meaningful content for corrupted images.

Although the aforementioned semantic image inpainting approaches are able to generate visually pleasing content, they fail to generate satisfactory structures sometimes, especially when the object layouts are complex. To this end, we proposed our PGN approach, which is detailed in Section 3.

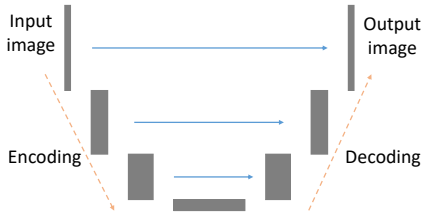


**Figure 3: The illustration of UNet architecture.**

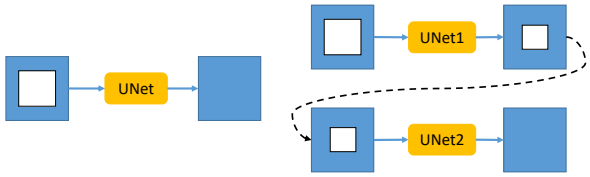


**Figure 4: Inpainting process of student A (the left figure) and student B (the right figure).**

# 3 PROPOSED APPROACH

This section elaborates on the proposed progressive generative networks. We start with a toy example and make clear the motivation, then we introduce the proposed approach in detail. For ease of description, we use the same setting as in Context Encoder [19] and assume the corrupted image is resized to $128 \times 128$; the missing region always locates in the center of the image, which occupies a quarter of the entire image(i.e., $64 \times 64$).

## 3.1 Motivation: A toy example

Assume that we have two learners, i.e., student A and student B. These two students have nearly the same learning ability to inpaint images. Given an inpainting task, student A always wants to finish it at one go, i.e., he does not care about the inpainting order, while student B always schedules the inpainting order and finish the task step by step.

More formally, suppose student A and student B grasp the same inpainting technique, i.e., using an encoder-decoder framework with a reconstruction loss and an adversarial loss. This framework is similar to Context Encoder [19] except that a different generator called UNet [21] (see Fig 3) is used as in [14]. The difference is that student A inpaints images in one action while student B progressively finishes this task. The inpainting process of them is shown in Fig 4.

We adopt the same training strategy for student A and student B, and conduct experiments on Paris Street View dataset [8]. The results are shown in Fig. 5, from which one can see that student B is able to recover much better structures and textures than student A. This gives a hint that inpainting in a reasonable order can yield a better result. Actually, this strategy was also utilized in some early works [7, 16].

## 3.2 Progressive Generative Networks

The preliminary experiment results shown in Section 3.1 have demonstrated the importance of the processing order of inpainting task. However, some questions remain unclear, e.g., which kind of processing order can benefit the inpainting result and how to define it; and how to control the information flow given lots of processing phases. This section aims to answer these questions by introducing


**Figure 5: Inpainting results of student A (column (a)) and student B (column (b)). The ground truth images are shown in column (c).**

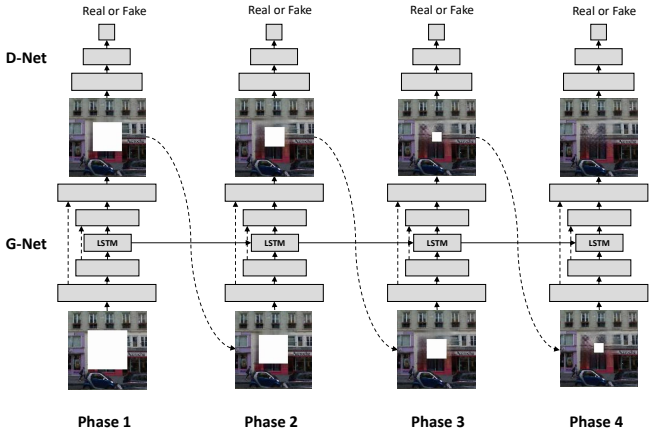


**Figure 6: The pipeline of the proposed progressive generative networks.**

a progressive generative networks (see Fig. 6 for the processing pipeline).

Our PGN approach is based on the inpainting process of humans, i.e., starting with easier proportions and gradually extending to more difficult ones. This spirit was first introduced by Bengio et al. [4], and it is termed curriculum learning. A key to this approach is to define a reasonable curriculum strategy. In semantic image inpainting task, a natural choice is to define the curriculum according to the inpainting order. This paper also adopts such a curriculum strategy. Specifically, we divide the inpainting process into different phases, and each phase tackles a specific inpainting subtask. Considering both the effectiveness and efficiency, we define four phases totally (other quantities are also experimented, see Section 4 for more details). In the first phase, PGN inpaints the outermost ring of the corrupted region, and move inward gradually until the entire corrupted region is filled. To complete such a process, some key components are required, including a generator to generate meaningful content, a discriminator to judge the quality of the generated content, an LSTM component to control the information flow and a proper loss function to guide the network training. We elaborate on these components one by one in the following.

*3.2.1 Generative networks.* A good generator is quite significant to the semantic image inpainting task. Most existing works adopt an autoencoder or its variants [13, 19, 32, 33]. However, very recently, Isola et al. [14] found that the UNet [21] is more powerful in generative modelling, and they achieved remarkable results in image-to-image translation task by using UNet. Therefore, this paper adopts UNet as the generator. As in [14, 21], the UNet used in this work is fully convolutional, and the information in the encoding part is directly passed to the corresponding place in the decoding part as they are symmetric (see Fig. 3), which reduces information loss in traditional autoencoder architecture.

*3.2.2 Discriminative networks.* As adversarial training is exploited in this paper, the discriminator is also vital. The discriminator is used to judge whether an image is real or generated by the generator. Simply utilizing a generator can yield good convergence property. However, it cannot ensure the generated image look naturalistic. The discriminator can be viewed as another guide to the training process of the generative networks. Similar to previous work [14], we adopt fully convolutional discriminative networks (PatchGAN) in this paper, which are capable of judging whether a small patch is real or not.

*3.2.3 Long short-term memory.* As we divide the inpainting task into different phases, it is crucial to pass the information from previous phases to the subsequent ones. In this paper, we introduce the long short-term memory (LSTM) [12] to string all the phases together due to its ability of capture long-distance statistical regularities. Associated with LSTM units between every two adjacent phases, the essence learned from the previous phases flows to the subsequent subtasks and eases the learning process.

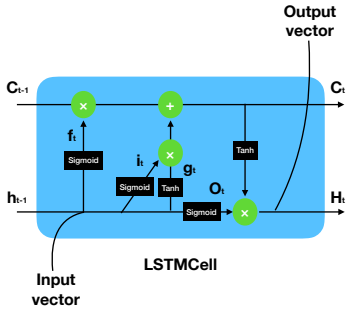


**Figure 7: Structure of the LSTM networks.**

**Specific Structure** We use a one-layer LSTM networks in PGN. The LSTM networks can be divided into 4-time steps. For each time step, the input data is a 1024-dimension vector which is generated by the encoding part of the UNet. The structure of LSTM networks is shown in Fig 7. Note that $c_t$ is the cell state of the LSTM cell at time $t$, $h_t$ is the hidden state of the LSTM cell at time $t$. Besides, $i_t$, $f_t$, $g_t$, $o_t$ are the input, forget, cell, and output gates, respectively. The output data is a 2048-dimension vector, which is generated by the hidden state of the LSTM. We concatenate the 2048-dimension

output vector and the 1024-dimension input vector to get a 3072-dimension vector of $feature_t$, and send it to the decoding part of the UNet. For each time step $t$, the LSTM networks compute the following functions:

$$i_t = sigmoid(W_{ii} input_t + b_{ii} + W_{hi} h_{t-1} + b_{hi}), \quad (1)$$
$$f_t = sigmoid(W_{if} input_t + b_{if} + W_{hf} h_{t-1} + b_{hf}), \quad (2)$$
$$ot = sigmoid(W_{io} input_t + b_{io} + W_{ho} h_{t-1} + b_{ho}), \quad (3)$$
$$g_t = tanh(W_{ig} input_t + b_{ig} + W_{hc} h_{t-1} + b_{hg}), \quad (4)$$
$$c_t = f_t * c_{t-1} + i_t * g_t, \quad (5)$$
$$h_t = o_t * tanh(c_t), \quad (6)$$
$$output_t = h_t, \quad (7)$$
$$feature_t = Concatenate(ouput_t, input_t). \quad (8)$$

As for the initialization of the LSTM networks, we set each element in $h_0$ and $c_0$ to zero.

*3.2.4 Loss function.* The PGN approach consists of several different phases, and each phase aims to solve a specific subtask of inpainting. Therefore, it is necessary to impose a loss function in each phase. As we use similar architecture in all phases, the loss functions used are similar, which include a reconstruction loss, an adversarial loss and a total variation loss.

**Reconstruction loss** We use the L1 loss as the reconstruction loss in this paper as it has been demonstrated to be effective in previous generative models [14, 33]. However, as the inpainting task is not completed at one go, the definition of L1 loss in this paper is slightly different from that in previous work. Assume the input of each phase is $\mathbf{x_i}$ (obviously $\mathbf{x_1}$ is the original image to inpaint), then we have

$$\mathbf{x}_{i+1} = G(\mathbf{x}_i), \quad (9)$$

where $G$ is the generator. Suppose the mask of phase $i$ is $\mathbf{M}_i$ (the pixel value in the corrupted region is 0 and otherwise is 1), then the reconstruction loss in this phase is defined as

$$L_{rec}^{(i)} = \|G(\mathbf{x}_i) - \widehat{\mathbf{x}} \odot \mathbf{M}_{i+1}\|_1, \quad (10)$$

where $\widehat{\mathbf{x}}$ is the ground truth image. As the reconstruction loss only penalize the pixel-wise error, and cannot ensure the data distribution to be similar to natural images. Therefore, it easily leads to blurry inpainting result. This can be alleviated by imposing an adversarial loss [19, 32, 33].

**Adversarial loss** The adversarial loss is based on generative adversarial networks (GAN) proposed by Goodfellow et al. [10]. Generally, a generator and a discriminator are involved in GAN, where the generator aims to fool the discriminator, while the discriminator aims to be not fooled by the generator. As such, the generator and the discriminator contest with each other in a zero-sum game. The proposed PGN approach also adopts such an adversarial training strategy, and the adversarial loss is defined as

$$L_{adv}^{(i)} = \max_D E[log(D(\widehat{\mathbf{x}} \odot \mathbf{M}_{i+1})) + log(1 - D(G(\mathbf{x}_i)))], \quad (11)$$

where the generator $G$ and the discriminator $D$ are jointly optimized. The adversarial loss is quite critical in the inpainting task, and using it yields more naturalistic generated content.

**Total variation loss** Total variation (TV) loss [22] was first designed for denoising and deblurring. However, as it is able to help propagate image intensity smoothly, it is also widely used in image inpainting task [1, 28, 32]. We also add a TV loss in the PGN approach as follows

$$L_{tv} = TV\_loss(\mathbf{x}) \quad (12)$$
$$= \sum_{m,n} ((\mathbf{x}_{m,n+1} - \mathbf{x}_{m,n})^2 + (\mathbf{x}_{m+1,n} - \mathbf{x}_{m,n})^2) \quad (13)$$

where $m$ and $n$ are the coordinate position of a pixel in an image.

**Total loss** The total loss of the proposed approach is then defined as

$$L = \sum_i (\alpha L_{rec}^{(i)} + \beta L_{adv}^{(i)} + \lambda L_{tv}), \quad (14)$$

where $\alpha$, $\beta$, $\lambda$ are weights on the three types of loss respectively. Practically, we set $\alpha = 1$, $\beta = 0.001$ and $\lambda = 1$.

*3.2.5 Inpainting by PGN.* With the four key components at hand, we are able to train the progressive generative networks for semantic image inpainting task. Given an undamaged image, we first corrupt it with four predefined masks (from the largest to the smallest). Then, in the first phase, we feed the first corrupted image (the image with the largest hole) into the generative network. In this phase, PGN aims to generate an image as similar as the second corrupted image (with the aim of fooling the discriminative network), i.e., inpaint the outermost ring of the corrupted region. The discriminative network is also an important part in the inpainting process. Specifically, in each training iteration, we regard the generated image as the fake one, and regard the ground truth image as the real one. The discriminative network is used to identify whether an image is real or machine generated. By optimizing the generative network and the discriminative network alternatively, PGN is able to generative images as naturalistic as possible. The generated image is then passed to the input of the second phase. Besides that, the "bottleneck layer" (a compact feature representation) is also passed to the second phase via an LSTM unit. It is worth noting that the LSTM unit established between two adjacent phases is quite vital, and with which PGN is able to pass the essence learn in the previous phase to the latter one. After finishing the inpainting subtask in the first phase, PGN starts the next few phases. Actually, the network architecture of the next few phases is exactly the same as that in the first phase. The only difference lies in the input and the output as each phase aims to inpaint a specific corrupted region. By formulating the semantic image inpainting task using such a curriculum strategy, the PGN approach is able to progressively shrink large corrupted regions and generate high-quality images.

## 4 EXPERIMENTS

In this section, we conduct experiments on two benchmark dataset and evaluate the proposed PGN visually and quantitatively. Practically, we first provide the details of dataset and experimental settings. Then we compare the image inpainting results with baselines to demonstrate the effectiveness of our method. We also provide an effects evaluation of different factors in PGN to make a comprehensive analysis.

## 4.1 Datasets

We implement the PGN method on two dataset: Paris StreetView and ImageNet dataset.

**Paris StreetView** dataset consists of 14,900 images for training and 100 images for testing. The visual class of dataset focuses on the Street Siew, such as houses, trees, street, skies, and other objects. The original size of each image is 936×537. To increase the number of iteration without overfitting, we randomly crop some 128×128 patches for one image to augment the dataset. Intuitively, the image inpainting on the Paris StreetView dataset is less difficult than ImageNet dataset due to the fewer object categories.

**ImageNet** dataset contains about 15 million images associated with more than 20,000 categories. Considering the experimental configuration of Context Encode [19], we extract a subset of the ImageNet dataset in this experiment. Specifically, we randomly select 100k images from 1000 object categories according to the manner provided by Context Encoder [19]. However, we do not use the additional label information of the image.

## 4.2 Implementation details

*4.2.1 Training.* For the training progress, we resize and crop each training image into 128×128 to fit the PGN architecture. Each image is painted by a 64×64 "light gray" (ImageNet mean) mask in the center. The output of PGN is divided into 2, 4 and 8 phases. For each phase, the generator outputs a 128×128 inpainted image. Each output of the generator corresponds to a discriminative network, called as the discriminator, to determine whether the output is real or fake. After that, we use the backpropagation algorithm to propagate the gradient from discriminator to generator. Meanwhile, the reconstruction loss and TV loss are back propagate to the generator directly. The training procedure is conducted on one GTX-1080 GPU. For the ParisStreetView dataset, the PGN model was trained for 200 epochs with two days. For the ImageNet dataset, it took 150 epochs with nearly two weeks. The training progress on two dataset is implemented in Python with PyTorch [18]. The detailed training procedure is formally presented in Algorithm 1.

*4.2.2 Testing.* For the testing procedure, only the generator of PGN is needed. The testing set is painted in the same way of training progress. The painted area of each input test image is invisible. Since different progress phases can generate different outputs, we also test the progressive performance of PGN.

## 4.3 Results and comparisons

We compare the proposed PGN with two well-established approaches: Context Encoders image inpainting [19] (abbreviated to CE) and High-resolution image inpainting [32] (abbreviated to HR). These two baselines are proposed in the last two years and achieved qualified results on the benchmark dataset.

The qualitative result on Paris StreetView and ImageNet dataset is shown in Fig 9 and Fig 8 respectively. From the results of ImageNet dataset, we can see that the visual performance of PGN is significantly better than two baselines. The transition at the boundary of the inpainted region is more natural and visually smooth in our results. This is mainly because the discriminative network of PGN considers the entire region of the testing image. The adversarial loss of PGN discriminator forces the image transition more smoothing at the boundary to make the output picture more visually natural. From the results of the Paris StreetView dataset, it is easily observed that the quality of the inpainted image of our method is greatly improved compared to Context Encoder, and even as much better as the results of High-resolution Image Inpainting. It is worth mentioning that the calculation speed of our method is as fast as Context Encoder. Emphatically, the entire inpainting process of a $128 \times 128$ image takes less than 0.1 seconds, while High-Resolution image inpainting approach needs several minutes or more for one image.

Following previous work, the quantitative results of PGN is reported in Table 1. From the numerical results on the Paris StreetView dataset we can see that our method has achieved the highest performance comparing with baselines. We also conducted quantitative evaluation via user study. Totally, 18 participants ranged from 22 to 33 years old are invited to participate in our experiments. All subjects are asked to vote for the inpainting results as poor, fair, satisfactory, good or excellent assigned with the score range from 1-5. We test 20 image groups generated from PGN and baselines. The statistic results of each approach is in Table 2.

**Algorithm 1.**

Minibatch stochastic gradient descent training of PGN.
We use $D_i$ to represent i-th discriminative network,
$M_i$ is the mask of phase i,
$\alpha$ is the weight of the reconstruction loss,
$\beta$ is the weight of the adversarial loss,
$\lambda$ is the weight of the total variation loss,
here we use $\alpha$ = 1, $\beta$ = 0.001 and $\lambda$ = 1.

**for** number of training iteration **do**:

1. Sample minibatch of k real 128*128 image $\{real^1, real^2 ... real^k\}$ from real image Dataset.
2. Destroy k real images sample in the center 64*64 region to get k destroyed images $\{destroy^1, destroy^2 ... destroy^k\}$
3. Send k destroyed images $\{destroy^1, destroy^2 ... destroy^k\}$ to generator and yield 4*k generated fake images $\{fake_1^1, fake_1^2 ... fake_1^k\}$, $\{fake_2^1, fake_2^2 ... fake_2^k\}$, $\{fake_3^1, fake_3^2 ... fake_3^k\}$, $\{fake_4^1, fake_4^2 ... fake_4^k\}$.
4. **for** number of phases **do**:
   Update the i-th discriminator of the PGN by ascending its stochastic gradient [19][10][18]:
   $$\nabla_{\theta_{D_i}} \frac{1}{k} \sum_{j=1}^{k} [\log D_i(real^j \odot M_{i+1}) + \log(1 - D_i(fake_i^j))]$$
   **end for**
5. Update the generator of the PGN by descending its stochastic gradient [19][10][22][18]:
   $$\nabla_{\theta_G} \frac{1}{k} \sum_i \{\alpha L_{rec}^{(i)} + \beta L_{adv}^{(i)} + \lambda L_{tv}\}$$

**end for**

**Note:** The gradient-based updates can use any standard gradient-based learning rule. We used Adam (lr = 0.002, beta = [0.5,0.999] ) in our experiments.

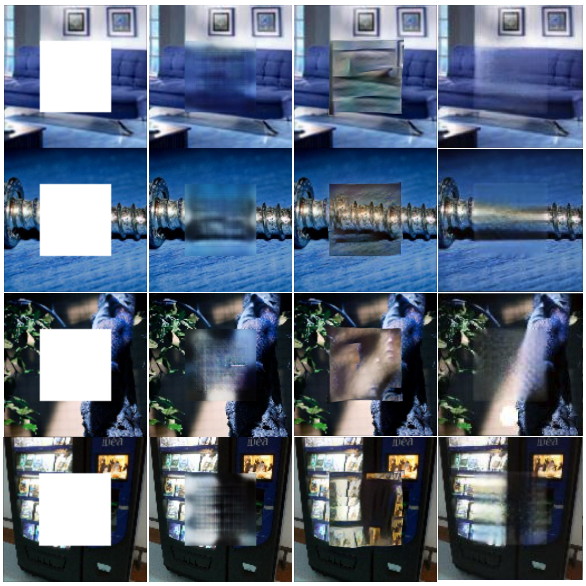


**Figure 8: Results of the ImageNet dataset.**


**Figure 9: Results of the ParisStreetView dataset.**

## 4.4 Analysis and discussions

**The effect of adversarial loss** We evaluate the different weights of the adversarial loss in our experiment. The results are shown in Fig 10. We found that the proper weight of adversarial loss is 0.001. Excessive weight will lead to distortion of the image. This is because that the discriminative network can only discriminate the low-level abstraction, such as local texture, due to its fewer layers. It is too shallow to discriminate the high-level semantic features. Although it is intuitively to learn a deeper discriminative network,

| Method | Mean L1 Loss | Mean L2 Loss | PSNR |
|---|---|---|---|
| CE [19] | 10.33% | 2.35% | 17.59 dB |
| HR [32] | 10.01% | 2.21% | 18.00 dB |
| PGN(our method) | **8.11**% | **1.51**% | **19.72** dB |

**Table 1: Numerical comparison on the Paris StreetView dataset. Higher PSNR value is better. Note % is to facilitate reading.**

| Method | Mean value | Stdev |
|---|---|---|
| CE [19] | 2.020 | 0.884 |
| HR [32] | 2.515 | 1.366 |
| PGN(our method) | 3.275 | 0.946 |

**Table 2: Results of our user study evaluation.**


**Figure 10: Results of different weights of adversarial loss.**

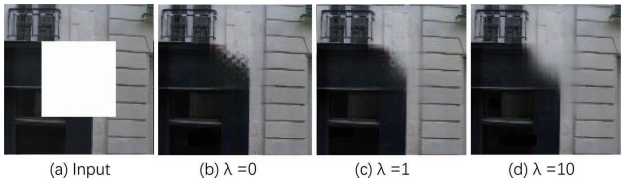


**Figure 11: Results of different weights of total variation loss.**

it will significantly increases the difficulty of training. Thus we utilize L1 reconstruction loss to constrain the semantic information of the whole image as the same as Context Encoder.

**The effect of total variation loss** For the total variation (TV) loss, we compare the results of different weights to evaluate its effects. The results are shown in Fig 11. The TV loss is set to deblurring, but when the weigh is oversized, the construction information of the image is destroyed. Finally, we make a trade-off between 'eliminate ripple' and 'picture information is corrupted' and set a proper weigh as 1.

**The effect of curriculum steps** As a hyper-parameter, we set the curriculum phrases as 4-step in the above experiments. To confirm the effect of curriculum steps on the inpainted results, we conduct the experiments on different number of curriculum phase. Especially, we evaluate the PGN with 2,4 and 8 phases. The result is shown in Fig 12. We can see that the results from 4-step PGN are better than others. For the 8-step PGN, there are two deficiencies. One is the information fading by passing 8 UNets. The other is the training instability due to the small batch size caused by the limitation of GPU memory.

**The effect of LSTM components** To demonstrate the effects of LSTM components in PGN framework, we compare the results with/without LSTM units and show them in Fig 13. Because the input information will be disturbed by passing multiple UNets, the inpainted results without the transmit information from LSTM are


**Figure 12: Results of different phases.**


**Figure 13: Results of with/without LSTM components.**

blurred. It has proved that introducing LSTM networks to connect multiple UNets can obviously compensate the interference information in the transmission process.

**The effect of other painted shape** Our PGN networks is not only applicable for the rectangular painted region but also can be generalized to other types of pained shapes, such as elliptic, triangular or other irregular shapes. Here we take elliptic painted region for example. As shown in Fig 14, from the left column to the right are the outputs from PGN at four stages. We can see that the PGN can inpaint the image from the edge to the center progressively and achieve the excellent results.

## 5 CONCLUSION

This paper presents a novel progressive image inpainting framework. We formulate the semantic image inpainting as a curriculum learning problem. By defining a proper curriculum strategy, our approach is able to shrink the corrupted regions progressively and obtain promising inpainting results. Moreover, the proposed approach is very fast to evaluate as the entire hole filling is performed

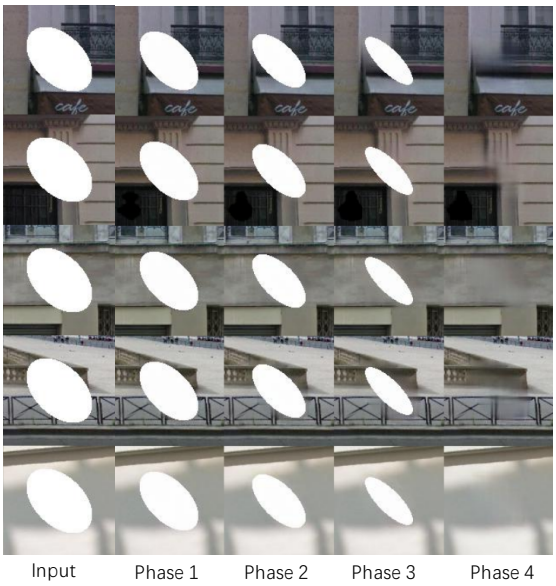


**Figure 14: Results of the elliptic painted region.**

in a single forward pass. To our knowledge, this is the first inpainting work involving curriculum learning idea, and we have demonstrated its effectiveness. However, we are far from applying this idea perfectly into inpainting task. Lots of issues need to be address, e.g., how to use this kind of approach in the case that the corrupted region is not fixed (it can be any shape in any location); and how to control the model complexity if the curriculum has lots of "courses". Such issues remain challenging and more superior curriculum strategy are expected. This is beyond the scope of this paper, and we leave it as the future work.

## ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 61632007, Grant 61732008 and Grant 61725203.

## REFERENCES

[1] Coloma Ballester, Marcelo Bertalmio, Vicent Caselles, Guillermo Sapiro, and Joan Verdera. 2001. Filling-in by joint interpolation of vector fields and gray levels. *IEEE transactions on image processing* 10, 8 (2001), 1200–1211.
[2] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. 2009. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on graphics* 28, 3 (2009), 24–1.
[3] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proc. NIPS*. 1171–1179.
[4] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proc. ICML*. 41–48.
[5] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. 2000. Image inpainting. In *Proc. SIGGRAPH*. 417–424.
[6] Marcelo Bertalmio, Luminita Vese, Guillermo Sapiro, and Stanley Osher. 2003. Simultaneous structure and texture image inpainting. *IEEE Transactions on image processing* 12, 8 (2003), 882–889.
[7] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. 2004. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on image processing* 13, 9 (2004), 1200–1212.
[8] Carl Doersch, Saurabh Singh, Abhinav Gupta, Josef Sivic, and Alexei Efros. 2012. What makes paris look like paris? *ACM Transactions on Graphics* 31, 4 (2012).
[9] Iddo Drori, Daniel Cohen-Or, and Hezy Yeshurun. 2003. Fragment-based image completion. In *ACM Transactions on graphics*, Vol. 22. 303–312.
[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial

nets. In *Proc. NIPS*. 2672–2680.
[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *arXiv preprint arXiv:1512.03385* (2015).
[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[13] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. 2017. Globally and locally consistent image completion. *ACM Transactions on Graphics* 36, 4 (2017), 107.
[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2016. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004* (2016).
[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Proc. NIPS*.
[16] Olivier Le Meur, Josselin Gautier, and Christine Guillemot. 2011. Examplar-based inpainting based on local geometry. In *Proc. ICIP*. 3401–3404.
[17] Anat Levin, Assaf Zomet, and Yair Weiss. 2003. Learning how to inpaint from global image statistics. In *Proc. ICCV*. 305–312.
[18] Adam Paszke, Sam Gross, and Adam Lerer. 2017. Automatic differentiation in PyTorch.
[19] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. 2016. Context encoders: Feature learning by inpainting. In *Proc. CVPR*. 2536–2544.
[20] Fernando J Pineda. 1987. Generalization of back-propagation to recurrent neural networks. *Physical review letters* 59, 19 (1987), 2229.
[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Proc. MICCAI*. 234–241.
[22] Leonid I Rudin, Stanley Osher, and Emad Fatemi. 1992. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena* 60, 1-4 (1992), 259–268.
[23] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proc. CVPR*. 815–823.
[24] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).
[25] Jian Sun, Lu Yuan, Jiaya Jia, and Heung-Yeung Shum. 2005. Image completion with structure propagation. *ACM Transactions on Graphics (ToG)* 24, 3 (2005), 861–868.
[26] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going Deeper With Convolutions. In *Proc. CVPR*.
[27] Joachim Weickert. 1999. Coherence-enhancing diffusion filtering. *International journal of computer vision* 31, 2 (1999), 111–127.
[28] You-Wei Wen, Raymond H Chan, and Andy M Yip. 2012. A primal–dual method for total-variation-based wavelet domain inpainting. *IEEE Transactions on Image Processing* 21, 1 (2012), 106–114.
[29] Paul J Werbos. 1988. Generalization of backpropagation with application to a recurrent gas market model. *Neural networks* 1, 4 (1988), 339–356.
[30] Ronald J Williams and David Zipser. 1995. Gradient-based learning algorithms for recurrent networks and their computational complexity. *Backpropagation: Theory, architectures, and applications* 1 (1995), 433–486.
[31] Zongben Xu and Jian Sun. 2010. Image inpainting by patch propagation using patch sparsity. *IEEE transactions on image processing* 19, 5 (2010), 1153–1165.
[32] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. 2016. High-Resolution Image Inpainting using Multi-Scale Neural Patch Synthesis. *arXiv preprint arXiv:1611.09969* (2016).
[33] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. 2017. Semantic image inpainting with deep generative models. In *Proc. CVPR*. 5485–5493.
[34] Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proc. EMNLP*. 1393–1398.