

Iterative Application of Autoencoders for Video Inpainting and Fingerprint Denoising



Le Manh Quan and Yong-Guk Kim

1 Introduction

Image inpainting plays an important role for the image restoration task. It can be used in many applications and has also useful preprocessing steps for other applications. Image inpainting helps to recover the missing pixel values of images and videos or removes unwanted information, even sophisticated pattern-like objects.

Video inpainting is in principle similar to image inpainting, but video inpainting is method oriented as a sequence of images. However, there are some important differences between image inpainting and video inpainting. In image inpainting the missing parts are usually covered by the relationships with surrounding pixels in the isolated image. So in order to separate the non-given missing region with other region in the image requires an algorithm to have an understanding the relation between those regions in the image and find the unrelated one, which is not simple for the single image. However, in the video inpainting, sequences of images with the same details are reproduced over multiple frames, so missing region can be easily identified and separated from other regions through the frames. Commonly, the regions that need to be recovered or filled in video are much larger and more complex, typically that have some associated stochasticity textures compared to a single image. It is important to clarify that it is not simple to apply the algorithm of image inpainting to video inpainting and vice versa. The key point of image inpainting considers only to spatial information and completely ignores the temporal component that presented in video sequences. Meanwhile, the missing region searching method in video inpainting will not accurate in image inpainting. In this

L. M. Quan · Y.-G. Kim (✉)

Department of Computer Engineering, Sejong University, Seoul, South Korea

e-mail: ykim@sejong.ac.kr

© Springer Nature Switzerland AG 2019

S. Escalera et al. (eds.), *Inpainting and Denoising Challenges*,

The Springer Series on Challenges in Machine Learning,

https://doi.org/10.1007/978-3-030-25614-2_5

work, we proposed the iterative approach using learning based method that could apply for both video inpainting and image inpainting with a high accuracy.

Besides its wide use, image inpainting is still facing many challenges. For example, the unknown region to fill and the region to recover is too large compared to images, or due to the complexity of a high dimension image, the texture patterns make the filled region unrealistic and blurred. Historical approaches, such as texture based [5, 10] method utilizing neighboring pixels, search based method by finding the region then approximating the region in the same given image [4], or searching from the database [8] to fill the missing parts were working well with many cases, but problems exist when images consist of complex pattern spaces like natural images.

2 Related Work

There are several ways to categorize the image inpainting. However based on the approach of the method, the extensive literature on image inpainting can be grouped into three categories: patch-based inpainting, sparsity inpainting and model based inpainting (learning based). Recently, the model based approach becomes popular and gives promising results. Given that deep neural networks have been applied to many fields in image processing and computer vision area, network based method shows the remarkable performance in many applications, such as object detection and classification [6, 13], image super-resolution [3, 14, 15], and etc. In image inpainting [1, 2, 11, 12, 16, 17, 19–22], deep neural network approach makes images more realistic and natural. Combining sparse coding and deep networks with an autoencoder denoising technique, [19] worked in image denoising and blind images inpainting task, remove complex patterns, like superimposed text from an image automatically. Yang [21] proposed a multi-scale neural path synthetic method which shown promising results in high-resolution images by using two trained networks for evaluating holistic content and for minimizing the local texture loss. In [17], Pathak published the context encoder method, closely related to the auto-encoder and used the convolution neural network with its surrounding relation to predict the missing region. Burlin [1] and Demir in [2] proposed some methods using Generative Adversarial Network (GAN) [7] framework into image inpainting and showed considerable improvements on challenging public datasets (CIFAR10, ImageNet and Paris Street View). However, these studies tried to find out which neural network model fits best to the dataset and resolve the image inpainting problem by presenting the single best model or two pipelines with a single neural network model on each pipeline. Most researchers worked on how to develop the network model structure by changing the number of layers, making a new function of layers, and preventing over fitting problems. They worked to invent the best model that can apply to several cases, since the complex operation computation and numerous parameters of each network (GoogleNet employ 5 million parameters, AlexNet with 60 million parameters and VGGNet approximate

180 million parameters) causes the black box, and it is difficult for authors to give the specific functioning of each layer's insights on each structure. Also, using the single network approach has faced limitations such as, finding the best-fit network for each dataset, re-training the whole process with a new network, and hard work for changing the number of layers and creating the new layers for network. Switching to a new sequential network has some advantages: utilizing the features of images by bringing different feature to the networks through each stage, network runs well with the combination of some well-known networks, a reusable the pre-trained model when stages are the same and training network stages are well-mannered so easy and flexible to change or modify the model (Fig. 1).

The term inpainting was started from art restoration. Art restoration is the process of recovering the damaged artwork, searching for a goal of making it look like the original work. This process requires a lot of works as well as a deep understanding of every detail of the given artwork. Normally, to recover a damaged image, it takes a lot of time and needs a lot of people involved to recover and verify. Our work was inspired by traditional painting restoration where the work should be done with a step by step process, rather than by one-shot inpainting. In this work, we will propose the new application for image inpainting, using multiple neural networks or an autoencoders in a single pipeline to solve the inpainting problem with the following contributions:

- We propose a new architecture for image inpainting. Instead of trying to solve the inpainting problem using a single deep neural network or an autoencoder at once, our method adopts an iterative approach by utilizing multiple autoencoders or multiple neural networks for image inpainting. It is believed that reusing an autoencoder in a sequential manner is a more natural way to restore the original image through several steps, rather than carrying out it at once.

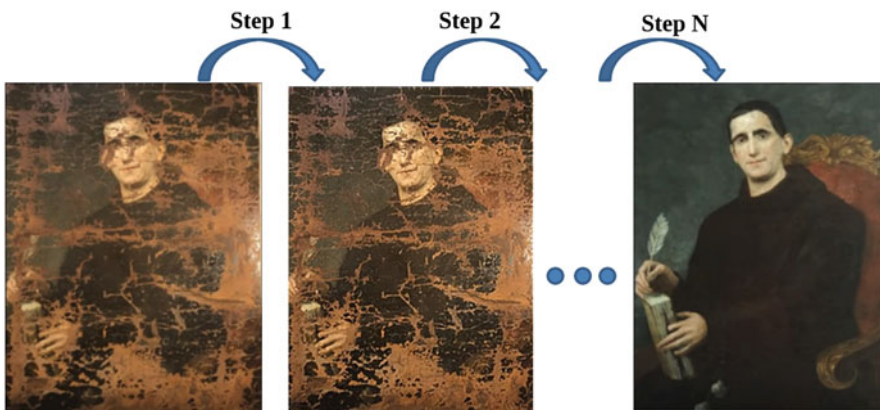
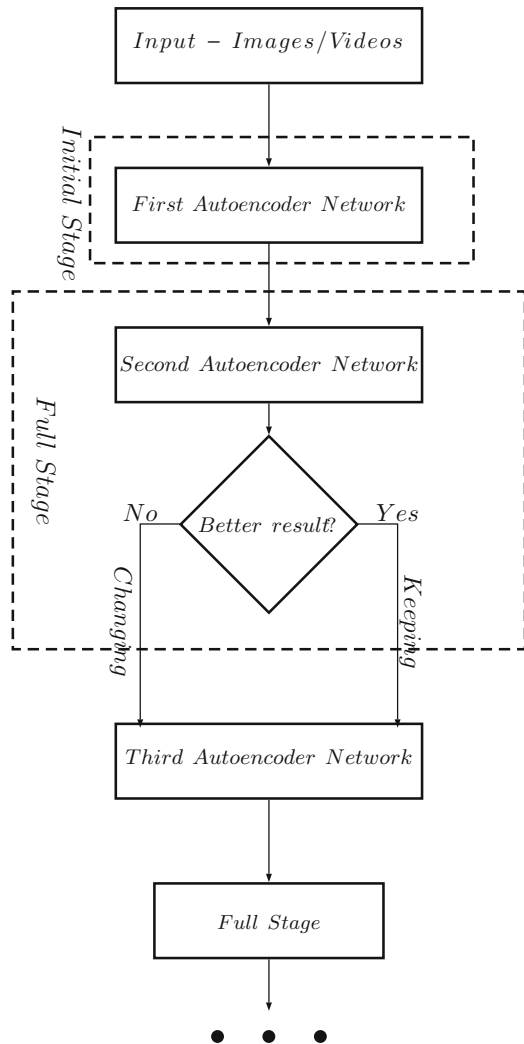


Fig. 1 Old painting restoration

- Applying our approach with a baseline neural network in two challenges: Video Decaptioning (track-2) and Fingerprint Denoising and Inpainting (track-3) of Image Inpainting workshop, we achieved promising results (second in track-2 and third in track-3) (Fig. 2).

Fig. 2 The overview of our proposed method



3 Method Description

Autoencoders and the Iterative Approach Autoencoder is one of the traditional neural networks and it can be applied to solve both the supervised learning problems as well as unsupervised learning problems. It consists of two parts: encoder part and decoder part (Figs. 3 and 4).

Encoder With input x , through the hidden layers E_{hidden} with the weight W and bias b , we received the output form as a mapping followed by a function f :

$$f_{\{W,b\}}(x) = E_{hidden}(Wx + b) \tag{1}$$

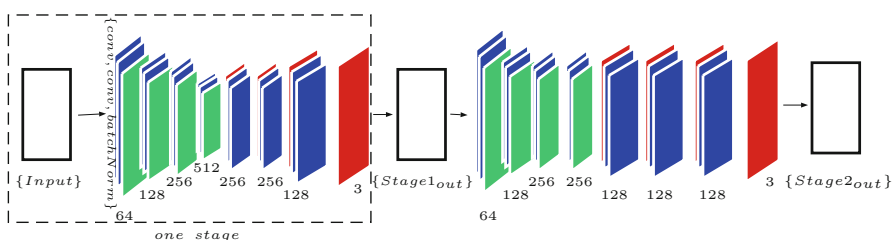


Fig. 3 Video decaptioning architecture including 2 stages

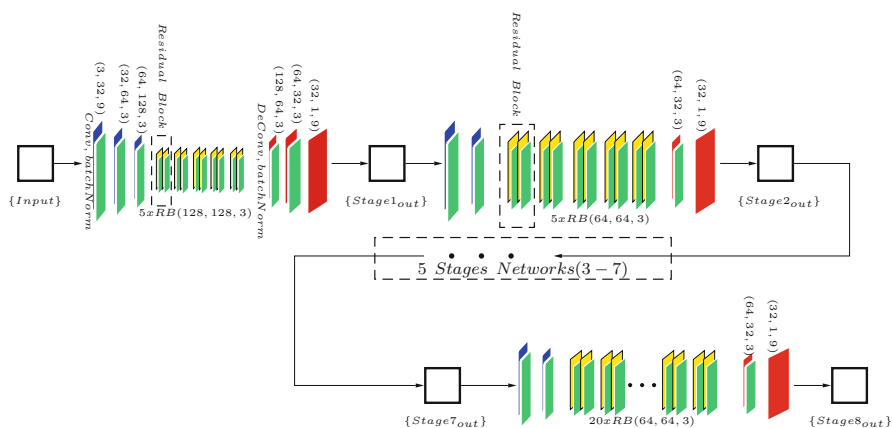


Fig. 4 Fingerprint verification—image inpainting architecture including 8 stages, each stage has different network (stages 3 and 8 have the same network architecture)-RB is notation of Residual Block

Decoder Then, the result of the encoder is mapped back through the hidden layers D_{hidden} with weight W and bias b to reconstruct the final result with the same dimension as the input x , follow by a function g :

$$g_{\{W',b'\}}(x) = D_{hidden}(W' \cdot f_{\{W,b\}}(x) + b') \quad (2)$$

Normally the final result $z = g(x)$ will never be the same as input x , so to evaluate the performance we use the Mean Squared Error (MSE) which less value of MSE means a better result of z and more similar with input x .

$$MSE = \frac{1}{n} \sum_1^n \|z - x\|^2 \quad (3)$$

However, using MSE for evaluating the image is typically used but two additional measures, i.e. Peak Signal-to-Noise Ratio and DSSIM (Structure similarity) index are adopted to improve on the traditional methods such as PSNR and MSE.

Our iterative approach consists of several stages, and each stage contains one full Autoencoder network and one examination step. The input could be images or videos for which we competed in two different tracks within the inpainting challenge. The input was put into the first autoencoder network without any examination. The role of examination is to test if the output shows any better performance compared to the previous stage. The output from the initial stage then becomes the input of the second stage. The examination step will ask if the result is better or not, if better we keep the same autoencoder architecture to the next stages, otherwise we change the number of it.

The purpose of changing the autoencoder architecture in difference stages is getting the different feature images which the single autoencoders could not work. In this approach, we showed that the features of the image could be all exploited by making a few changes in the old neural network. Instead of the output given after the single network, we will receive the final output after n th stage, which n is the number of total stages and not determined. Our proposed method has advantages and disadvantages. Because the methods use several stages independently to train and test so we face to time consumption problem at the first training. However if we work with the light weight networks in several stages, we still get the faster result compared to the method using massive depth layers architecture. Besides that, our method gives several big advantages, such as better accuracy, lower loss after several stages, a well-mannered stage by correcting the process and choosing the better affection networks after each stage, and reusing the pre-trained model if stages are same.

In this work, we use the residual block and VGG-like encoder with bottle neck layers that had been provided in baseline to put input dataset as X (videos for track-2 and images for track-3) into our initial stage.

Residual Block In finger inpainting, consider the single image x_0 passes through the autoencoder network to give the output which presented by $F = g_{\{W',b'\}}(x)$ in (2). The autoencoders comprise i th layers which implemented by non-linear transform F_i , F_i can be composed by the operations of Convolution(conv), Relu, Pooling and Batch normalization. However, when the depth of the network comes deeper then a degradation problem has been exposed, accuracy gets saturated and then degrades rapidly. To handle that problem in each Autoencoders we add l number of Resnet block (RB). We denote x_l is the output of RB th, each RB adds the skip-connection by passing the non-linear transformations which presented by:

$$x_l = RB_l(x_{l-1}) + x_{l-1} \quad (4)$$

When we change the stage, we focus on changing the number of residual block to get the better effect on networks converging with less degradation problem.

Bottleneck Layer In video inpainting, the input is video with 125 frames 128×128 pixels. The computation time is big and should be considered carefully. In [9, 18] noted that using the bottle neck layers could reduce the input of feature maps, thus improving the computational efficiency. In the VGG-like autoencoders architecture, the bottleneck layers could have a size much less than last size of the last layer in the encoder part and first layer in the decoder part. The bottleneck layer could be a single dimension layer or multiple dimension layer but the size should be small. In this work we use the bottle neck layer of $512 \times 1 \times 1$ in stage 1 and increase one more on the second stage.

The iterative approach considers the input X is the set of images or videos x . We put the input x to the initial stage with model architecture M . After the n stages and n is not determined, we receive the output Y . The output function Y is mapped with the same dimension of input x and presented by non-linear transform:

$$\begin{aligned} Y_1(x) &= M_1(x)X(x) \\ Y_2(x) &= M_2(x)Y_1(x) \\ &\dots \\ Y_n(x) &= M_n(x)Y_{n-1}(x) \end{aligned} \quad (5)$$

Table 1 Our result over 8 stages on fingerprint verification (Track-03). Best results are bolded

Stage	01	02	03	04	05	06	07	08
MSE	0.0252	0.0244	0.0243	0.0242	0.02415	0.02413	0.02411	0.0238
PSNR	16.404	16.550	16.561	16.575	16.590	16.593	16.597	16.647
SSIM	0.8006	0.8008	0.8027	0.8039	0.8039	0.8037	0.8034	0.8033

Table 2 Our result over 2 stages on video decaptioning (Track-02). Best results are bolded

Stage	01	02
MSE	0.0014	0.0012
PSNR	31.204	33.023
SSIM	0.0482	0.0424

The composite of network with n stages is the product of all individual operations in each stage.

$$Y(x) = M_1(x)M_2(x)M_3(x) \dots M_n(x)X(x) \quad (6)$$

To measure performance of our model, Mean Square Error (MSE) is adopted to optimize the loss, and each measurement is the composite function of n stage:

$$MSE = MSE(Y_{groundtruth} - M_n(\dots(M_0(X(x))\dots))) \quad (7)$$

The question is to be how we can determine n and M_i . We could find these through experience with many stages on a single pipeline. One small trick to find M is that we can reuse the same network in difference stages if we find the loss decreasing over the stages and when the loss stops decreasing, then we can change to the other network by slightly changing the number of layers and parameters. We can also save the training time of different stages by using a pre-trained model from previous stages. To make our network becomes more autonomous and run without the human intervention, we can design a switcher function of each stage. The role of the switcher is to compare the output from the current stage to the previous one. If the output is getting better, then the switcher will keep the network, otherwise the switcher will ask to change the network. The changing layer can be random in a range of $[a, b]$, which a, b is the nature number and less than 20 layers. Because in our experiment, we decided to change the number of layers to less than 20 layers to make the new network architecture light weight and not meet the time consumption problem (Tables 1 and 2).

4 Experiments

We applied our approach to two challenges of ChaLearn Challenges on Image and Video Inpainting ECCV satellite 2018: Video decaptioning (track-2) and Inpainting and denoising for fingerprint verification (track-3). In both tracks, we used the baseline network for the first stage of our pipeline.

Track no.	MSE	PSNR	DSSIM
Track 2-ours	0.0012	33.0228	0.0424
Baseline	0.0022	30.1856	0.0613
Track 3-ours	0.0238	16.6465	0.8033
Baseline	0.0241	16.4109	0.7965

Video decaptioning: The dataset consists of 90,000 video clips and each clip has 125 frames having 128×128 pixels. 70% of the dataset were used for training, 15% for evaluation and 15% for testing, respectively. Within the dataset, video content, style and resolution and various font transformation and subtitles overlays are different. The subtitle regions are often large compared to the entire frame images. To remove the subtitle, initially we need to define where is the region of the subtitles but sometimes it can be confused by running text content inside the a given video. The baseline was acquired using a deep convolution autoencoder with a VGG-like encoder. Each decoder consists of four blocks functions: convolution, upsampling, batch normalization and ReLU activations. For the second stage, to change the stage slightly, we increased one bottle neck layer and decreased one block of the decoder. However, because the time for training with the present huge dataset was increased exponentially, we could not test our approach with more stages. And yet, by utilizing 2 stages only, we achieved a decent result, i.e. 0.0011 in MSE compared to the baseline, 0.0022.

Inpainting and denoising for fingerprint verification: This dataset consists of three sub-sets: synthetic training set, synthetic test set and real test set. Synthetic training set contains 84,000 images with 275×400 pixels free-noise or scratches, but 84,000 images were generated by applying random artifacts. Synthetic test set contains 16,800 images and the process to generate them is similar to that for the training set case. Finally, the real test set contains 1680 images including 140 fingerprints with 12 impressions using the high quality scan condition. The challenge for this dataset is that although the quantity is not very big, the finger print pattern is very complicated. Since MSE, PSNR and DSSIM are used to evaluate the

recovered finger pattern, but we do not know if this pattern unique id is changed or not. In reality, the id could be more important than the quality of the images. For the baseline, a standard deep neural network (DNN), consisting of four convolution layers, five residual blocks [9] (each block comprises two convolution layers), two deconvolution layers, and another convolution layer, was used. We trained the first stage with a baseline within 200,000 iterations and slightly changed the number of residual blocks for the next stages. In this experiment, 8 stages network was used. Based on the performance after the first 7 stages, it is found that when the residual blocks number was 18 layers, then the network performs most efficiently among 7 stages. Then we reused that network at stage eighth. With 8 stages network, our proposed method achieved the better result, 0.0238 MSE compare to the baseline, 0.0241.

5 Conclusion

Inpainting and denoising are a new research topic which is interesting and very challenging. Given that autoencoder is a useful and powerful network block, we propose a new iterative approach in dealing with these problems. We successfully apply this approach for two challenge tracks: video inpainting and fingerprint denoising. Since art restoration process is similar to that of inpainting in principle, it makes sense to adopt an iterative approach to solve the problem. The restoration of old painting is a very hard and time-consuming job simply because the restorer has to master sophisticated painting skills as well as shall have historic knowledge for the painting. It is also known that when one plans to restore a damaged painting, he usually carries out restoration through several laborious steps rather than at once, i.e. the iterative approach. Despite the fact that a deep neural network or a sparse autoencoder stacked in such network is a powerful tool by which someone wishes to accomplish inpainting without necessary processes, our proposed approach is somehow akin to the authentic way of restoring precious old paintings.

Acknowledgements This research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2018-2016-0-00312) supervised by the IITP (Institute for information and communications Technology Promotion).

Appendix

Our code is available online (Tables 3 and 4).¹

¹The code of 2 tracks:









































































Track 2: https://github.com/quant6791/Track02_VideoDecaptioning.

Track 3: https://github.com/quant6791/Track03_FingerPrintVerification.

Table 3 Our result over 2 stages on video Decaptioning (Track-02)

Input	Stage01	Stage02
		
		
		
		
MSE	0.0014	0.0012

Table 4 Our result over 8 stages on fingerprint verification images (Track-03)

Input	Stage01	Stage02	Stage03	Stage04	Stage05	Stage06	Stage07	Stage08
								
								
								
								
								
								
								
								
MSE	0.0252	0.0244	0.0243	0.0242	0.02415	0.02413	0.02411	0.0238

References

1. Burlin, Charles, Yoann Le Calonnec, and Louis Duperier. Deep Image Inpainting. (2017).
2. Demir, Ugur, and Gozde Unal.: Patch-Based Image Inpainting with Generative Adversarial Networks. arXiv preprint arXiv:1803.07422 (2018).
3. Dong, Chao, Chen Change Loy, Kaiming He, and Xiaoou Tang : Image super-resolution using deep convolutional networks. IEEE transactions on pattern analysis and machine intelligence 38, no. 2, pp. 295–307, (2016).
4. Drori, Iddo, Daniel Cohen-Or, and Hezy Yeshurun.: Fragment-based image completion. In ACM Transactions on graphics (TOG), vol. 22, no. 3, pp. 303–312. ACM, (2003).
5. Efros, Alexei A., and William T. Freeman.: Image quilting for texture synthesis and transfer. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 341–346. ACM, (2001).
6. Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik.: Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587. (2014).
7. Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio.: Generative adversarial nets. In Advances in neural information processing systems, pp. 2672–2680. (2014).
8. Hays, James, and Alexei A. Efros.: Scene completion using millions of photographs. In ACM Transactions on Graphics (TOG), vol. 26, no. 3, p. 4. ACM, (2007).
9. He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.: Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778. (2016).
10. Igehy, Homan, and Lucas Pereira.: Image replacement through texture synthesis. In Image Processing, 1997. Proceedings., International Conference on, vol. 3, pp. 186–189. IEEE, (1997).
11. Iizuka, Satoshi, Edgar Simo-Serra, and Hiroshi Ishikawa.: Globally and locally consistent image completion. ACM Transactions on Graphics (TOG) 36, no. 4,107, (2014).
12. Jain, Viren, and Sebastian Seung.: Natural image denoising with convolutional networks. In Advances in Neural Information Processing Systems, pp. 769–776. (2009)
13. Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton.: Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pp. 1097–1105. (2012).
14. Kim, Jiwon, Jung Kwon Lee, and Kyoung Mu Lee.: Accurate image super-resolution using very deep convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1646–1654. (2016).
15. Ledig, Christian, Lucas Theis, Ferenc Huszr, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P. Aitken et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In CVPR, vol. 2, no. 3, p. 4. (2017).
16. Mao, Xiao-Jiao, Chunhua Shen, and Yu-Bin Yang.: Image restoration using convolutional auto-encoders with symmetric skip connections. arXiv preprint arXiv:1606.08921 (2016).
17. Pathak, Deepak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros.: Context encoders: Feature learning by inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2536–2544. (2016).
18. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z: Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818–2826 (2016).
19. Xie, Junyuan, Linli Xu, and Enhong Chen.: Image denoising and inpainting with deep neural networks. In Advances in neural information processing systems, pp. 341–349. (2012).
20. Xu, Li, Jimmy SJ Ren, Ce Liu, and Jiaya Jia.: Deep convolutional neural network for image deconvolution. In Advances in Neural Information Processing Systems, pp. 1790–1798. (2014).

21. Yang, Chao, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li.: High-resolution image inpainting using multi-scale neural patch synthesis. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, no. 2, p. 3. (2017).
22. Yeh, Raymond A., Chen Chen, Teck-Yian Lim, Alexander G. Schwing, Mark Hasegawa-Johnson, and Minh N. Do.: Semantic Image Inpainting with Deep Generative Models. In CVPR, vol. 2, no. 3, p. 4. (2017).