Shift-Net: Image Inpainting via Deep Feature Rearrangement

Zhaoyi Yan[1], Xiaoming Li[1], Mu Li[2], Wangmeng Zuo[1⋆], Shiguang Shan[3,4]
yanzhaoyi@outlook.com, csxmli@hit.edu.cn, csmuli@comp.polyu.edu.hk,
wmzuo@hit.edu.cn, sgshan@ict.ac.cn

[1]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
[2]Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China
[3]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing 100190, China
[4]CAS Center for Excellence in Brain Science and Intelligence Technology

Abstract. Deep convolutional networks (CNNs) have exhibited their potential in image inpainting for producing plausible results. However, in most existing methods, e.g., context encoder, the missing parts are predicted by propagating the surrounding convolutional features through a fully connected layer, which intends to produce semantically plausible but blurry result. In this paper, we introduce a special shift-connection layer to the U-Net architecture, namely Shift-Net, for filling in missing regions of any shape with sharp structures and fine-detailed textures. To this end, the encoder feature of the known region is shifted to serve as an estimation of the missing parts. A guidance loss is introduced on decoder feature to minimize the distance between the decoder feature after fully connected layer and the ground-truth encoder feature of the missing parts. With such constraint, the decoder feature in missing region can be used to guide the shift of encoder feature in known region. An end-to-end learning algorithm is further developed to train the Shift-Net. Experiments on the Paris StreetView and Places datasets demonstrate the efficiency and effectiveness of our Shift-Net in producing sharper, fine-detailed, and visually plausible results. The codes and pre-trained models are available at https://github.com/Zhaoyi-Yan/Shift-Net.

Keywords: Inpainting, feature rearrangement, deep learning

1 Introduction

Image inpainting is the process of filling in missing regions with plausible hypothesis, and can be used in many real world applications such as removing distracting objects, repairing corrupted or damaged parts, and completing occluded regions. For example, when taking a photo, rare is the case that you are

⋆ Corresponding author.

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

Fig. 1. Qualitative comparison of inpainting methods. Given (a) an image with a missing region, we present the inpainting results by (b) Content-Aware Fill [1], (c) context encoder [2], and (d) our Shift-Net.

satisfied with what you get directly. Distracting scene elements, such as irrelevant people or disturbing objects, generally are inevitable but unwanted by the users. In these cases, image inpainting can serve as a remedy to remove these elements and fill in with plausible content.

Despite decades of studies, image inpainting remains a very challenging problem in computer vision and graphics. In general, there are two requirements for the image inpainting result: (i) global semantic structure and (ii) fine detailed textures. Classical exemplar-based inpainting methods, e.g., PatchMatch [3], gradually synthesize the content of missing parts by searching similar patches from known region. Even such methods are promising in filling high-frequency texture details, they fail in capturing the global structure of the image (See Fig. 1(b)). In contrast, deep convolutional networks (CNNs) have also been suggested to predict the missing parts conditioned on their surroundings [2,4]. Benefited from large scale training data, they can produce semantically plausible inpainting result. However, the existing CNN-based methods usually complete the missing parts by propagating the surrounding convolutional features through a fully connected layer (i.e., bottleneck), making the inpainting results sometimes lack of fine texture details and blurry. The introduction of adversarial loss is helpful in improving the sharpness of the result, but cannot address this issue essentially (see Fig. 1(c)).

In this paper, we present a novel CNN, namely Shift-Net, to take into account the advantages of both exemplar-based and CNN-based methods for image inpainting. Our Shift-Net adopts the U-Net architecture by adding a special shift-connection layer. In exemplar-based inpainting [5], the patch-based replication and filling process are iteratively performed to grow the texture and structure from the known region to the missing parts. And the patch processing order plays a key role in yielding plausible inpainting result [6,7]. We note that CNN is effective in predicting the image structure and semantics of the missing parts. Guided by the salient structure produced by CNN, the filling process in our Shift-Net can be finished concurrently by introducing a shift-connection layer to connect the encoder feature of known region and the decoder feature of missing parts. Thus, our Shift-Net inherits the advantages of exemplar-based and CNN-based methods, and can produce inpainting result with both plausible semantics and fine detailed textures (See Fig. 1(d)).

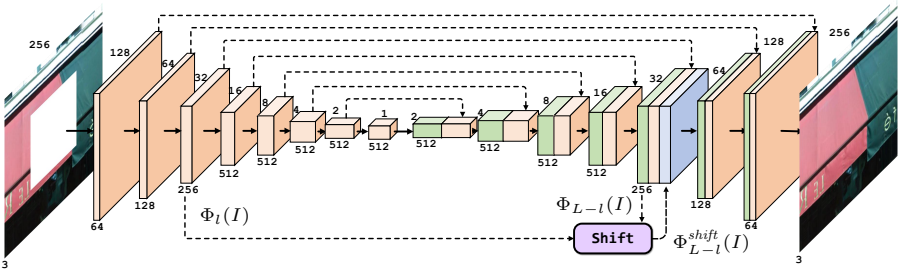


Fig. 2. The architecture of our model. We add the shift-connection layer at the resolution of 32×32. shift跳连接只在32上做

Guidance loss, reconstruction loss, and adversarial learning are incorporated to guide the shift operation and to learn the model parameters of Shift-Net. To ensure that the decoder feature can serve as a good guidance, a guidance loss is introduced to enforce the decoder feature be close to the ground-truth encoder feature. Moreover, ℓ_1 and adversarial losses are also considered to reconstruct the missing parts and restore more detailed textures. By minimizing the model objective, our Shift-Net can be end-to-end learned with a training set. Experiments are conducted on the Paris StreetView dataset [8], the Places dataset [9], and real world images. The results show that our Shift-Net can handle missing regions with any shape, and is effective in producing sharper, fine-detailed, and visually plausible results (See Fig. 1(d)).

Besides, Yang *et al.* [4] also suggest a multi-scale neural patch synthesis (MNPS) approach to incorporating CNN-based with exemplar-based methods. Their method includes two stages, where an encoder-decoder network is used to generate an initial estimation in the first stage. By considering both global content and texture losses, a joint optimization model on VGG-19 [10] is minimized to generate the fine-detailed result in the second stage. Even Yang *et al.* [4] yields encouraging result, it is very time-consuming and takes about 40, 000 millisecond (ms) to process an image with size of 256×256. In contrast, our Shift-Net can achieve comparable or better results (See Fig. 4 and Fig. 5 for several examples) and only takes about 80 ms. Taking both effectiveness and efficiency into account, our Shift-Net can provide a favorable solution to combine exemplar-based and CNN-based inpainting for improving performance.

To sum up, the main contribution of this work is three-fold:

1. By introducing the shift-connection layer to U-Net, a novel Shift-Net architecture is developed to efficiently combine CNN-based and exemplar-based inpainting.
2. The guidance, reconstruction, and adversarial losses are introduced to train our Shift-Net. Even with the deployment of shift operation, all the network parameters can be learned in an end-to-end manner.
3. Our Shift-Net achieves state-of-the-art results in comparison with [2–4] and performs favorably in generating fine-detailed textures and visually plausible results.

2 Related Work

In this section, we briefly review the work on each of the three sub-fields, i.e., exemplar-based inpainting, CNN-based inpainting, and style transfer, and specially focus on those relevant to this work.

2.1 Exemplar-based inpainting

In exemplar-based inpainting [3, 5–7, 11–22], the completion is conducted from the exterior to the interior of the missing part by searching and copying best matching patches from the known region. For fast patch search, Barnes *et al.* suggest a PatchMatch algorithm [3] to exploit the image coherency, and generalize it for finding k-nearest neighbors [11]. Generally, exemplar-based inpainting is superior in synthesizing textures, but is not well suited for preserving edges and structures. For better recovery of image structure, several patch priority measures have been proposed to fill in structural patches first [5–7]. Global image coherence has also been introduced to the Markov random field (MRF) framework for improving visual quality [16, 18, 21]. However, these methods only work well on images with simple structures, and may fail in handling images with complex objects and scenes. Besides, in most exemplar-based inpainting methods [16–18], the missing part is recovered as the shift representation of the known region in pixel/region level, which also motivates our shift operation on convolution feature representation.

2.2 CNN-based inpainting

Recently, deep CNNs have achieved great success in image inpainting. Originally, CNN-based inpainting is confined to small and thin masks [23–25]. Phatak *et al.* [2] present an encoder-decoder (i.e., context encoder) network to predict the missing parts, where an adversarial loss is adopted in training to improve the visual quality of the inpainted image. Even context encoder is effective in capturing image semantics and global structure, it completes the input image with only one forward-pass and performs poorly in generating fine-detailed textures. Semantic image inpainting is introduced to fill in the missing part conditioned on the known region for images from a specific semantic class [26]. In order to obtain globally consistent result with locally realistic details, global and local discriminators have been proposed in image inpainting [27] and face completion [28]. For better recovery of fine details, MNPS is presented to combine exemplar-based and CNN-based inpainting [4].

2.3 Style transfer

Image inpainting can be treated as an extension of style transfer, where both the content and style (texture) of missing part are estimated and transferred from the known region. In the recent few years, style transfer [29–37] has been an active research topic. Gatys *et al.* [31] show that one can transfer style and texture of the style image to the content image by solving an optimization objective defined on an existing CNN. Instead of the Gram matrix, Li *et al.* [35] apply the MRF

regularizer to style transfer to suppress distortions and smears. In [29], local matching is performed on the convolution layer of the pre-trained network to combine content and style, and an inverse network is then deployed to generate the image from feature representation.

3 Method

Given an input image I, image inpainting aims to restore the ground-truth image I^{gt} by filling in the missing part. To this end, we adopt U-Net [38] as the baseline network. By incorporating with guidance loss and shift operation, we develop a novel Shift-Net for better recovery of semantic structure and fine-detailed textures. In the following, we first introduce the guidance loss and Shift-Net, and then describe the model objective and learning algorithm.

3.1 Guidance loss on decoder feature

The U-Net consists of an encoder and a symmetric decoder, where skip connection is introduced to concatenate the features from each layer of encoder and those of the corresponding layer of decoder. Such skip connection makes it convenient to utilize the information before and after bottleneck, which is valuable for image inpainting and other low level vision tasks in capturing localized visual details [39, 40]. The architecture of the U-Net adopted in this work is shown in Fig. 2. Please refer to the supplementary material for more details on network parameters. 缺失区域

Let Ω be the missing region and $\overline{\Omega}$ be the known region. Given a U-Net of L layers, $\Phi_l(I)$ is used to denote the encoder feature of the l-th layer, and $\Phi_{L-l}(I)$ the decoder feature of the $(L-l)$-th layer. For the end of recovering I^{gt}, we expect that $\Phi_l(I)$ and $\Phi_{L-l}(I)$ convey almost all the information in $\Phi_l(I^{gt})$. For any location $\mathbf{y} \in \Omega$, we have $(\Phi_l(I))_{\mathbf{y}} \approx 0$. Thus, $(\Phi_{L-l}(I))_{\mathbf{y}}$ should convey equivalent information of $(\Phi_l(I^{gt}))_{\mathbf{y}}$.

In this work, we suggest to explicitly model the relationship between $(\Phi_{L-l}(I))_{\mathbf{y}}$ and $(\Phi_l(I^{gt}))_{\mathbf{y}}$ by introducing the following guidance loss,

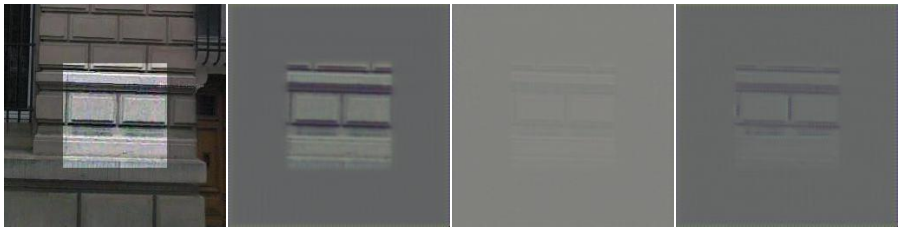
$$\mathcal{L}_g = \sum_{\mathbf{y}\in\Omega} \left\| (\Phi_{L-l}(I))_{\mathbf{y}} - (\Phi_l(I^{gt}))_{\mathbf{y}} \right\|_2^2. \tag{1}$$

We note that $(\Phi_l(I))_{\mathbf{x}} \approx (\Phi_l(I^{gt}))_{\mathbf{x}}$ for any $\mathbf{x} \in \overline{\Omega}$. Thus the guidance loss is only defined on $\mathbf{y} \in \Omega$ to make $(\Phi_{L-l}(I))_{\mathbf{y}} \approx (\Phi_l(I^{gt}))_{\mathbf{y}}$. By concatenating $\Phi_l(I)$ and $\Phi_{L-l}(I)$, all information in $\Phi_l(I^{gt})$ can be approximately obtained.

Experiment on deep feature visualization is further conducted to illustrate the relation between $(\Phi_{L-l}(I))_{\mathbf{y}}$ and $(\Phi_l(I^{gt}))_{\mathbf{y}}$. For visualizing $\{(\Phi_l(I^{gt}))_{\mathbf{y}} | \mathbf{y} \in \Omega\}$, we adopt the method [41] by solving an optimization problem

$$H^{gt} = \arg\min_{H} \sum_{\mathbf{y}\in\Omega} \left\| (\Phi_l(H))_{\mathbf{y}} - (\Phi_l(I^{gt}))_{\mathbf{y}} \right\|_2^2. \tag{2}$$

特征层之间的对比



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

Fig. 3. Visualization of features learned by our model. Given (a) an input image, (b) is the visualization of $\left(\Phi_l(I^{gt})\right)_{\mathbf{y}}$ (i.e., H^{gt}), (c) shows the result of $(\Phi_{L-l}(I))_{\mathbf{y}}$ (i.e., H^{de}) and (d) demonstrates the effect of $\left(\Phi^{shift}_{L-l}(I)\right)_{\mathbf{y}}$.

Analogously, $\{(\Phi_{L-l}(I))_{\mathbf{y}} \,|\mathbf{y} \in \Omega\}$ is visualized by

$$H^{de} = \arg\min_{H} \sum_{\mathbf{y}\in\Omega} \left\| (\Phi_l(H))_{\mathbf{y}} - (\Phi_{L-l}(I))_{\mathbf{y}} \right\|_2^2. \tag{3}$$

Figs. 3(b)(c) show the visualization results of H^{gt} and H^{de}. With the introduction of guidance loss, obviously H^{de} can serve as a reasonable estimation of H^{gt}, and U-Net works well in recovering image semantics and structures. However, in compared with H^{gt} and I^{gt}, the result H^{de} is blurry, which is consistent with the poor performance of CNN-based inpainting in recovering fine textures [4]. Finally, we note that the guidance loss is helpful in constructing an explicit relation between $(\Phi_{L-l}(I))_{\mathbf{y}}$ and $(\Phi_l(I^{gt}))_{\mathbf{y}}$. In the next section, we will explain how to utilize such property for better estimation to $(\Phi_l(I^{gt}))_{\mathbf{y}}$ and enhancing inpainting result.

3.2 Shift operation and Shift-Net

In exemplar-based inpainting, it is generally assumed that the missing part is the spatial rearrangement of the pixels/patches in the known region. For each pixel/patch localized at $\mathbf{y}$ in missing part, exemplar-based inpainting explicitly or implicitly find a shift vector $\mathbf{u}_{\mathbf{y}}$, and recover $(I)_{\mathbf{y}}$ with $(I)_{\mathbf{y}+\mathbf{u}_{\mathbf{y}}}$, where $\mathbf{y}+\mathbf{u}_{\mathbf{y}} \in \overline{\Omega}$ is in the known region. The pixel value $(I)_{\mathbf{y}}$ is unknown before inpainting. Thus, the shift vectors usually are obtained progressively from the exterior to the interior of the missing part, or by solving a MRF model by considering global image coherence. However, these methods may fail in recovering complex image semantics and structures.

We introduce a special shift-connection layer in U-Net, which takes $\Phi_l(I)$ and $\Phi_{L-l}(I)$ to obtain an updated estimation on $\Phi_l(I^{gt})$.

For each $(\Phi_{L-l}(I))_{\mathbf{y}}$ with $\mathbf{y} \in \Omega$, its nearest neighbor searching in $(\Phi_l(I))_{\mathbf{x}}$ ($\mathbf{x} \in \overline{\Omega}$) can be independently obtained by, 对于缺失的y，根据解码器前景与编码器背景的相似度，寻找背景的x代替y

$$\mathbf{x}^*(\mathbf{y}) = \arg\max_{\mathbf{x}\in\overline{\Omega}} \frac{\left\langle (\Phi_{L-l}(I))_{\mathbf{y}}, (\Phi_l(I))_{\mathbf{x}} \right\rangle}{\|(\Phi_{L-l}(I))_{\mathbf{y}}\|_2 \|(\Phi_l(I))_{\mathbf{x}}\|_2}, \tag{4}$$

and the shift vector is defined as $\mathbf{u_y} = \mathbf{x}^*(\mathbf{y}) - \mathbf{y}$. Similar to [35], the nearest neighbor searching can be computed as a convolutional layer. Then, we update the estimation of $(\Phi_l(I^{gt}))_{\mathbf{y}}$ as the spatial rearrangement of the encoder feature $(\Phi_l(I))_{\mathbf{x}}$,

$$\left(\Phi_{L-l}^{shift}(I)\right)_{\mathbf{y}} = (\Phi_l(I))_{\mathbf{y}+\mathbf{u_y}}. \quad (5)$$

可以通过背景块卷积和反卷积得到

See Fig. 3(d) for visualization. Finally, as shown in Fig. 2, the convolution features $\Phi_{L-l}(I)$, $\Phi_l(I)$ and $\Phi_{L-l}^{shift}(I)$ are concatenated and taken as inputs to the $(L-l+1)$-th layer, resulting in our Shift-Net.

The shift operation is different with exemplar-based inpainting from several aspects. (i) While exemplar-based inpainting is operated on pixels/patches, shift operation is performed on deep encoder feature domain which is end-to-end learned from training data. (ii) In exemplar-based inpainting, the shift vectors are obtained either by solving an optimization problem or in particular order. As for shift operation, with the guidance of $\Phi_{L-l}(I)$, all the shift vectors can be computed in parallel. (iii) For exemplar-based inpainting, both patch processing orders and global image coherence are not sufficient for preserving complex structures and semantics. In contrast, in shift operation $\Phi_{L-l}(I)$ is learned from large scale data and is more powerful in capturing global semantics. (iv) In exemplar-based inpainting, after obtaining the shift vectors, the completion result can be directly obtained as the shift representation of the known region. As for shift operation, we take the shift representation $\Phi_{L-l}^{shift}(I)$ together with $\Phi_{L-l}(I)$ and $\Phi_l(I)$ as inputs to $(L-l+1)$-th layer of U-Net, and adopt a data-driven manner to learn an appropriate model for image inpainting. Moreover, even with the introduction of shift-connection layer, all the model parameters in our Shift-Net can be end-to-end learned from training data. Thus, our Shift-Net naturally inherits the advantages of exemplar-based and CNN-based inpainting.

3.3 Model objective and learning

Objective. Denote by $\Phi(I;\mathbf{W})$ the output of our Shift-Net, where $\mathbf{W}$ is the model parameters to be learned. Besides the guidance loss, the ℓ_1 loss and the adversarial loss are also included to train our Shift-Net. The ℓ_1 loss is defined as,

$$\mathcal{L}_{\ell_1} = \|\Phi(I;\mathbf{W}) - I^{gt}\|_1, \quad (6)$$

which is suggested to constrain that the inpainting result should approximate the ground-truth image.

Recently, adversarial learning has been adopted in many low level vision [42] and image generation tasks [39,43], and exhibits its superiority in restoring high-frequency details and photo-realistic textures. As for image inpainting, we use $p_{data}(I^{gt})$ to denote the distribution of ground-truth images, and $p_{miss}(I)$ to denote the distribution of input image. The adversarial loss is then defined as,

$$\mathcal{L}_{adv} = \min_{\mathbf{W}} \max_{D} \mathbb{E}_{I^{gt}\sim p_{data}(I^{gt})}[\log D(I^{gt})] \quad (7)$$

$$+ \mathbb{E}_{I\sim p_{miss}(I)}[\log(1 - D(\Phi(I;\mathbf{W})))], \quad (8)$$

where $D(\cdot)$ denotes the discriminator to predict the probability that an image is from the distribution $p_{data}(I^{gt})$.

Taking guidance, ℓ_1, and adversarial losses into account, the overall objective of our Shift-Net is defined as,

$$\mathcal{L} = \mathcal{L}_{\ell_1} + \lambda_g \mathcal{L}_g + \lambda_{adv} \mathcal{L}_{adv}, \tag{9}$$

where λ_g and λ_{adv} are the tradeoff parameters for the guidance and adversarial losses, respectively.

Learning. Given a training set $\{(I, I^{gt})\}$, the Shift-Net is trained by minimizing the objective in Eqn. (9) via back-propagation. We note that the Shift-Net and the discriminator are trained in an adversarial manner. The Shift-Net $\Phi(I; \mathbf{W})$ is updated by minimizing the adversarial loss $\mathcal{L}_{adv}$, while the discriminator D is updated by maximizing $\mathcal{L}_{adv}$.

Due to the introduction of shift-connection layer, we should modify the computation of the gradient w.r.t. the l-th layer of feature $F_l = \Phi_l(I)$. To avoid confusion, we use F_l^{skip} to denote the feature F_l after skip connection, and of course we have $F_l^{skip} = F_l$. According to Eqn. (5), the relation between $\Phi_{L-l}^{shift}(I)$ and $\Phi_l(I)$ can be written as,

$$\Phi_{L-l}^{shift}(I) = \mathbf{P}\Phi_l(I), \tag{10}$$

where $\mathbf{P}$ denotes the shift matrix of $\{0, 1\}$, and there is only one element of 1 in each row of $\mathbf{P}$. Thus, the gradient with respect to $\Phi_l(I)$ consists of three terms,

$$\frac{\partial \mathcal{L}}{\partial F_l} = \frac{\partial \mathcal{L}}{\partial F_l^{skip}} + \frac{\partial \mathcal{L}}{\partial F_{l+1}} \frac{\partial F_{l+1}}{\partial F_l} + \mathbf{P}^T \frac{\partial \mathcal{L}}{\partial \Phi_{L-l}^{shift}(I)}, \tag{11}$$

where the computation of the first two terms are the same with U-Net, and the gradient with respect to $\Phi_{L-l}^{shift}(I)$ can also be directly computed. Thus, our Shift-Net can also be end-to-end trained to learn the model parameters $\mathbf{W}$.

4 Experiments

We evaluate our method on two datasets: Paris StreetView [8] and six scenes from Places365-Standard dataset [9]. The Paris StreetView contains 14,900 training images and 100 test images. There are 1.6 million training images from 365 scene categories in the Places365-Standard. The scene categories selected from Places365-Standard are *butte*, *canyon*, *field*, *synagogue*, *tundra* and *valley*. Each category has 5,000 training images, 900 test images and 100 validation images. Our model is learned using the training set and tested on the validation set. For both Paris StreetView and Places, we resize each training image to let its minimal length/width be 350, and randomly crop a subimage of size 256×256 as input to our model. Moreover, our method is also tested on real world images

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

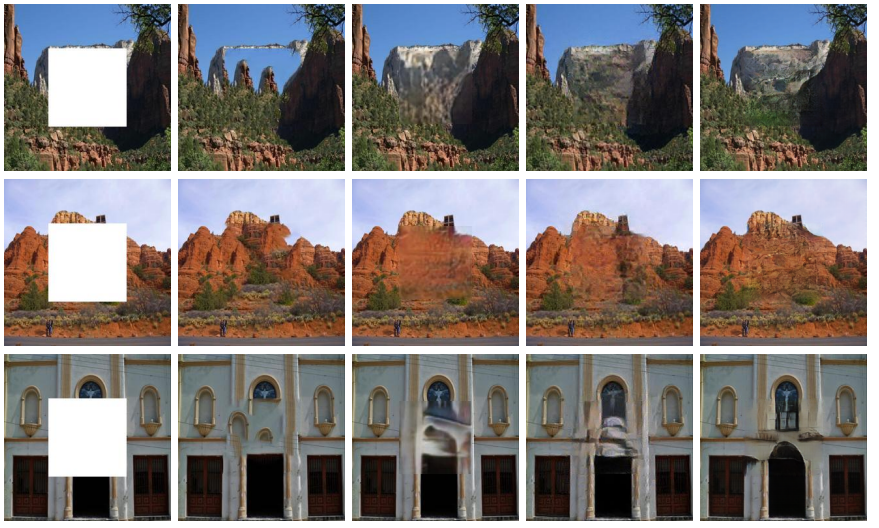
Fig. 4. Qualitative comparisons on the Paris StreetView dataset. From the left to the right are: (a) input, (b) Content-Aware Fill [1], (c) context encoder [2], (d) MNPS [4] and (e) Ours. All images are scaled to 256×256.

for removing objects and distractors. Our Shift-Net is optimized using the Adam algorithm [44] with a learning rate of 2×10^{-4} and $\beta_1 = 0.5$. The batch size is 1 and the training is stopped after 30 epochs. Data augmentation such as flipping is also adopted during training. The tradeoff parameters are set as $\lambda_g = 0.01$ and $\lambda_{adv} = 0.002$. It takes about one day to train our Shift-Net on an Nvidia Titan X Pascal GPU.

4.1 Comparisons with state-of-the-arts

We compare our results with Photoshop Content-Aware Fill [1] based on [3], context encoder [2], and MNPS [4]. As context encoder only accepts 128×128 images, we upsample the results to 256×256. For MNPS [4], we set the pyramid level be 2 to get the resolution of 256×256.

Evaluation on Paris StreetView and Places. Fig. 4 shows the comparisons of our method with the three state-of-the-art approaches on Paris StreetView. Content-Aware Fill [1] is effective in recovering low level textures, but performs slightly worse in handling occlusions with complex structures. Context encoder [2] is effective in semantic inpainting, but the results seem blurry and detail-missing due to the effect of bottleneck. MNPS [4] adopts a multi-stage scheme to combine CNN and examplar-based inpainting, and generally works better than Content-Aware Fill [1] and context encoder [2]. However, the multi-scales in MNPS [4] are not jointly trained, where some adverse effects produced in the first stage may not be eliminated by the subsequent stages. In comparison to the competing methods, our Shift-Net combines CNN and examplar-based inpainting in an end-to-end manner, and generally is able to generate visual-



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

Fig. 5. Qualitative comparisons on the Places. From the left to the right are: (a) input, (b) Content-Aware Fill [1], (c) context encoder [2], (d) MNPS [4] and (e) Ours. All images are scaled to 256×256.

Table 1. Comparison of PSNR, SSIM and mean ℓ_2 loss on Paris StreetView dataset.

Method	PSNR	SSIM	Mean ℓ_2 Loss
Content-Aware Fill [1]	23.71	0.74	0.0617
context encoder [2] (ℓ_2 + adversarial loss)	24.16	0.87	0.0313
MNPS [4]	25.98	0.89	0.0258
Ours	**26.51**	**0.90**	**0.0208**

pleasing results. Moreover, we also note that our Shift-Net is much more efficient than MNPS [4]. Our method consumes only about 80 ms for a 256×256 image, which is about $500\times$ faster than MNPS [4] (about 40 seconds). In addition, we also evaluate our method on the Places dataset (see Fig. 5). Again our Shift-Net performs favorably in generating fine-detailed, semantically plausible, and realistic images.

Quantitative evaluation. We also compare our model quantitatively with the competing methods on the Paris StreetView dataset. Table 1 lists the PSNR, SSIM and mean ℓ_2 loss of different methods. Our Shift-Net achieves the best numerical performance. We attribute it to the combination of CNN-based with examplar-based inpainting as well as the end-to-end training. In comparison, MNPS [4] adopts a two-stage scheme and cannot be jointly trained.

Random mask completion. Our model can also be trained for arbitrary region completion. Fig. 6 shows the results by Content-Aware Fill [1] and our Shift-Net. For textured and smooth regions, both Content-Aware Fill [1] and our Shift-Net perform favorably. While for structural region, our Shift-Net is more effective

Fig. 6. Random region completion. From top to bottom are: input, Content-Aware Fill [1], and Ours.

in filling the cropped regions with context coherent with global content and structures.

4.2 Inpainting of real world images

We also evaluate our Shift-Net trained on Paris StreetView for the inpainting of real world images by considering two types of missing regions: (i) central region, (ii) object removal. From the first row of Fig. 17, one can see that our Shift-Net trained with central mask can be generalized to handle real world images. From the second row of Fig. 17, we show the feasibility of using our Shift-Net trained with random mask to remove unwanted objects from the images.

5 Ablative Studies

The main differences between our Shift-Net and the other methods are the introduction of guidance loss and shift-connection layer. Thus, experiments are first conducted to analyze the effect of guidance loss and shift operation. Then we respectively zero out the corresponding weight of $(L-l+1)$-th layer to verify the effectiveness of the shift feature Φ_{L-l}^{shift} in generating fine-detailed results. Moreover, the benefit of shift-connection does not owe to the increase of feature map size. To illustrate this, we also compare Shift-Net with a baseline model by substituting the nearest neighbor searching with random shift-connection.

5.1 Effect of guidance loss

Two groups of experiments are conducted to evaluate the effect of guidance loss. In the first group of experiments, we add and remove the guidance loss $\mathcal{L}_g$ for U-

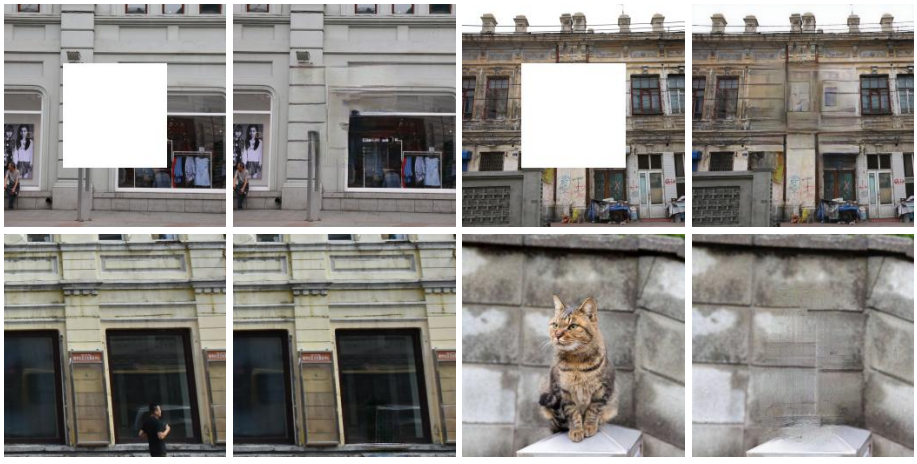


Fig. 7. Results on real images. From the top to bottom are: central region inpainting, and object removal.

(a) U-Net (w/o $\mathcal{L}_g$) (b) U-Net (w/ $\mathcal{L}_g$) (c) Ours (w/o $\mathcal{L}_g$) (d) Ours (w/ $\mathcal{L}_g$)

Fig. 8. The effect of guidance loss $\mathcal{L}_g$ in U-Net and our Shift-Net.

(a) $\lambda_g = 1$ (b) $\lambda_g = 0.1$ (c) $\lambda_g = 0.01$ (d) $\lambda_g = 0.001$

Fig. 9. The effect of the tradeoff parameter λ_g of guidance loss.

Net and our Shift-Net to train the inpainting models. Fig. 8 shows the inpainting results by these four methods. It can be observed that, for both U-Net and Shift-Net the guidance loss is helpful in suppressing artifacts and preserving salient structure.

In the second group of experiments, we evaluate the effect of the tradeoff parameter λ_g for guidance loss. For our Shift-Net, the guidance loss is introduced for both recovering the semantic structure of the missing region and guiding the

(a) ground-truth (b) $L-4$ (c) $L-3$ (d) $L-2$

Fig. 10. The effect of performing shift operation on different layers $L-l$.

shift of the encoder feature. To this end, proper tradeoff parameter λ_g should be chosen, where too large or too small λ_g values may be harmful to the inpainting results. Fig. 9 shows the results by setting different λ_g values. When λ_g is small (e.g., $= 0.001$), the decoder feature may not serve as a suitable guidance to guarantee the correct shift of the encoder feature. From Fig. 9(d), some artifacts can still be observed. When λ_g becomes too large (e.g., ≥ 0.1), the constraint will be too excessive, and artifacts may also be introduced in the result (see Fig. 9(a)(b)). Thus, we empirically set $\lambda_g = 0.01$ in all our experiments.

5.2 Effect of shift operation at different layers

The superiority of Shift-Net against context encoder [2] has demonstrated the effectiveness of shift operation. By comparing the results by U-Net (w/$\mathcal{L}_g$) and Shift-Net (w/$\mathcal{L}_g$) in Fig. 8(b)(d), one can see that shift operation does benefit the preserving of semantics and the recovery of detailed textures. Note that the shift operation can be deployed to different layer, e.g., $(L-l)$-th, of the decoder. When l is smaller, the feature map size goes larger, and more computation time is required to perform the shift operation. When l is larger, the feature map size becomes smaller, but more detailed information may lost in the corresponding encoder layer, which may be harmful to recover image details and semantics. Thus, proper l should be chosen for better tradeoff between computation time and inpainting performance. Fig. 10 shows the results of Shift-Net by adding the shift-connection layer to each of the $(L-4)$-th, $(L-3)$-th, and $(L-2)$-th layers, respectively. When the shift-connection layer is added to the $(L-2)$-th layer, Shift-Net generally works well in producing visually pleasing results, but it takes more time (i.e., ~ 400 ms per image) to process an image (See Fig. 10(d)). When the shift-connection layer is added to the $(L-4)$-th layer, Shift-Net becomes very efficient (i.e., ~ 40 ms per image) but tends to generate the result with less textures and coarse details (See Fig. 10(b)). By performing the shift operation in $(L-3)$-th layer, better tradeoff between efficiency (i.e., ~ 80 ms per image) and performance can be obtained by Shift-Net (See Fig. 10(c)).

5.3 Effect of the shifted feature

As we stacks the convolutional features $\Phi_{L-l}(I)$, $\Phi_l(I)$ and Φ_{L-l}^{shift} as inputs of $(L-l+1)$-th layer of U-Net, we can respectively zero out the weight of the corresponding slice in $(L-l+1)$-th layer. Fig. 11 demonstrates the results of

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

Fig. 11. Given (a) the input, (b), (c) and (d) are respectively the results when the 1st, 2nd, 3rd parts of weights in $(L-l+1)$-th layer are zeroed. (e) is the result of Ours.

Fig. 12. From top to bottom are: Shift-Net with random shift-connection and nearest neighbor searching.

Shift-Net by only zeroing out the weight of each slice. When we abandon the decoder feature $\Phi_{L-l}(I)$, the central part fails to restore any structures (See Fig. 11(b)), which indicates main structure and content are constructed by the subnet between $l \sim L-l$ layers. However, if we ignore the feature $\Phi_l(I)$, we get general structure (See (Fig. 11(c)) but quality inferior to the final result Fig. 11(e). This exhibits the fact that encoder feature $\Phi_l(I)$ has no significant effect on recovering features, which manifests the guidance loss is forceful to explicitly model the relationship between $(\Phi_{L-l}(I))_{\mathbf{y}}$ and $(\Phi_l(I^{gt}))_{\mathbf{y}}$ as illustrated in Sec. 3.1. Finally, when we discard the shift feature Φ^{shift}_{L-l}, the result is totally a mixture of structures (See Fig. 11(d)). Therefore, we can conclude that Φ^{shift}_{L-l} acts as a refinement and enhancement role in recovering clear and fine details in our Shift-Net.

5.4 Comparison with random shift-connection

Finally, we implement a baseline Shift-Net model by substituting the nearest neighbor searching with random shift-connection. Fig. 12 shows five examples of inpainting results by Shift-Net and baseline model. Compared to the nearest neighbor searching, the results by random shift-connection exhibit more artifacts, distortions, and structure disconnections. When training with random neighbor searching, random shifted feature continuously acts as dummy and confusing input. The network gradually learns to ignore Φ^{shift}_{L-l} in order to minimizing the

total loss function. Thus, the favorable performance of Shift-Net should owe to the correct shift-operation.

6 Conclusion

This paper has proposed a novel architecture, i.e., Shift-Net, for image completion that exhibits fast speed with promising fine details via deep feature rearrangement. The guidance loss is introduced to enhance the explicit relation between the encoded feature in known region and decoded feature in missing region. By exploiting such relation, the shift operation can be efficiently performed and is effective in improving inpainting performance. Experiments show that our Shift-Net performs favorably in comparison to the state-of-the-art methods, and is effective in generating sharp, fine-detailed and photo-realistic images. In future, more studies will be given to improve the speed of nearest searching in the shift operation, introduce multiple shift-connection layers, and extend the shift-connection to other low level vision tasks.

A Definition of masked region in feature maps

As shift-connection works based on the boundary of masked region and unmasked region in feature maps. Thus, we need to give a definition of masked region in feature maps. Denote by Ω^0 the missing part of the input image, and we should determine Ω^l for the l-th convolutional layer. In our implementation, we introduce a mask image M with $(M)_{\mathbf{y}} = 1$ ($\mathbf{y} \in \Omega$) and 0 otherwise. Then, we define a CNN $\Psi(M)$ that has the same architecture with the encoder but with the network width of 1. All the elements of the filters are $1/16$, and we remove all the nonlinearity. Taking M as input, we obtain the feature of the l-th layer as $\Psi_l(M)$. Then, Ω^l is defined as $\Omega^l = \{\mathbf{y}|(\Psi_l(M))_{\mathbf{y}} \geq T\}$, where T is the threshold with $0 \leq T \leq 1$. Fig. 13 shows the results of Shift-Net by setting $T = 4/16, 5/16, 6/16$, respectively. It can be seen that Shift-Net is robust to T, which may be attributed to that we take the shift and encoder, decoder features as the inputs to the $L - l + 1$ layer. We empirically set $T = 5/16$ in our experiments.

(a) Ground-truth (b) $T = 4/16$ (c) $T = 5/16$ (d) $T = 6/16$

Fig. 13. The effect of different thresholds in shift-connection.

B Details on Shift-Net

B.1 Architecture of generative model G.

For the generative model of our Shift-Net, we adopt the architecture of U-Net proposed in [39, 43]. Each convolution or deconvolution layer is followed by instance normalization [45]. The encoder part of G is stacked with Convolution-InstanceNorm-LeakyReLU layers, while the decoder part of G consists of seven Deconvolution-InstanceNorm-ReLU layers. Following the code of pix2pix, we zero out the biases of all convolution and deconvolution layers in the generative model in training. In this way, we can promise the correctness of **Line 208**. L denotes the total number of convolution/deconvolution layers in our model. We add guidance loss and shift operation in $(L-3)$-th layer, which results in the concatenated features of $\Phi_{L-3}(I)$, $\Phi_3(I)$ and $\Phi^{shift}_{L-3}(I)$ as inputs of the adjacent deconvolution. Details about the architecture of our generative model G is shown in Table 2. It is noted that we do not apply InstanceNorm on the bottleneck layer. The activation map of the bottleneck layer is 1×1, which means

we only get one activation per convolutional filter. As we train our network with batchsize 1, activations will be zeroed out once InstanceNorm is applied on the bottleneck layer. Please to pix2pix[1] for more explanation.

B.2 Architecture of discriminative network D.

D shares the similar design pattern with the encoder part of G, however, is only 5-convolution-layer network. We exclusively use convolution layers with filters of size 4×4 pixels with varying stride lengths to reduce the spatial dimension of the input down to a size of 30×30 where we append sigmoid activation at the final output. InstanceNorm is not applied to the first convolutional layer, and we use leaky ReLU with slope of 0.2 for activations except for the sigmoid in the last layer. See Table 3 for more details.

C More comparisons and object removals

C.1 Comparisons on Paris StreetView and Places datasets

More comparisons with context encoder [2], Content-Aware-Fill [1], pix2pix [39] and MNPS [4] on both Paris StreetView [8] and Places [9] are also conducted. Please refer to Fig. 14 and 15 for more results on Paris StreetView. For comparison on Places, please refer to Fig. 16. Our Shift-Net outperforms state-of-the-art approaches in both structural consistency and detail richness. Both global structure and fine details can be preserved in our model, however, other methods either perform poorly in generating clear, realistic details or lack global structure consistency.

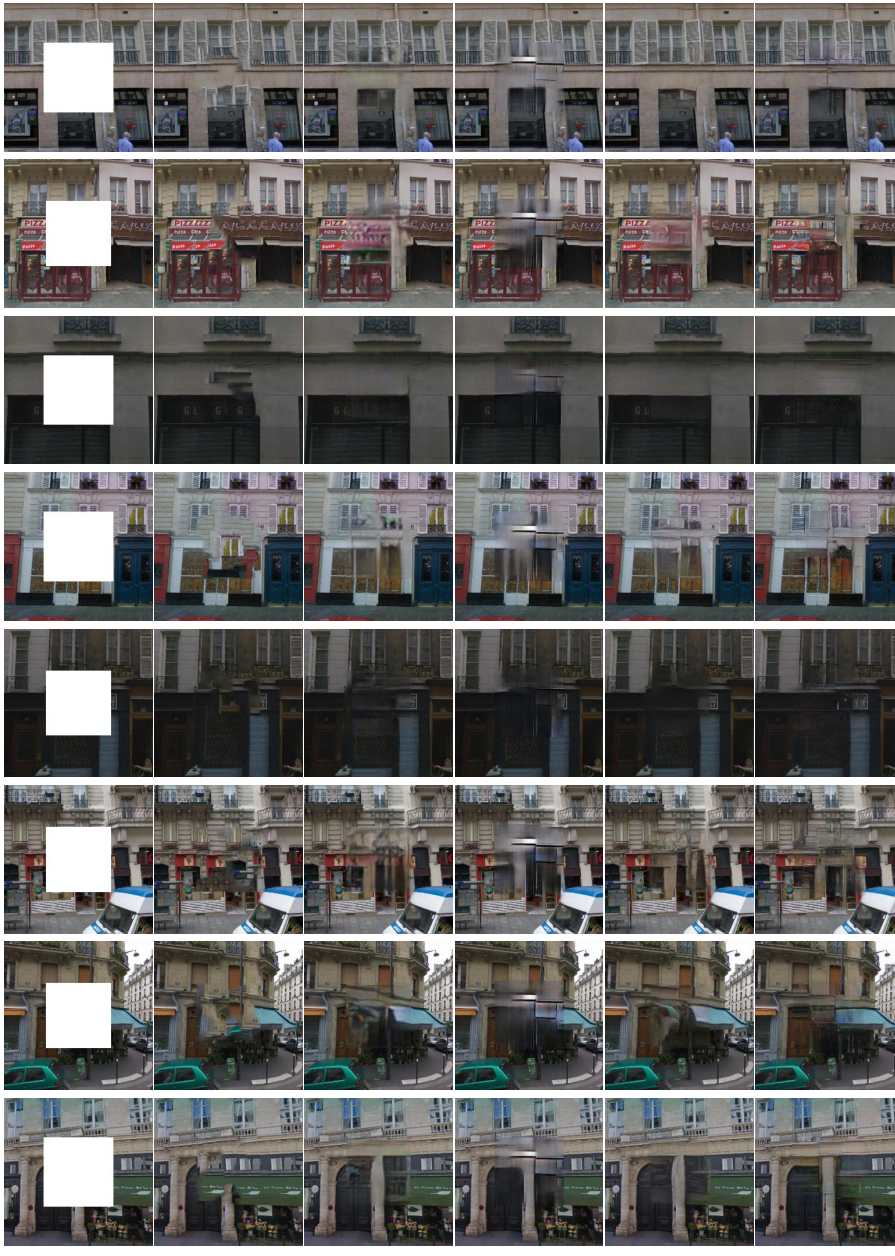
Table 3. The architecture of the discriminative network. "IN" represents InstanceNorm and "LReLU" donates leaky ReLU with the slope of 0.2.

The architecture of discriminative model D	
Input: Image ($256 \times 256 \times 3$)	
[layer 1]	Conv. (4, 4, 64), stride=2; $LReLU$;
[layer 2]	Conv. (4, 4, 128), stride=2; IN; $LReLU$;
[layer 3]	Conv. (4, 4, 256), stride=2; IN; $LReLU$;
[layer 4]	Conv. (4, 4, 512), stride=1; IN; $LReLU$;
[layer 5]	Conv. (4, 4, 1), stride=1; *Sigmoid*;
Output: Real or Fake ($30 \times 30 \times 1$)	

[1] https://github.com/phillipi/pix2pix/commit/b479b6b

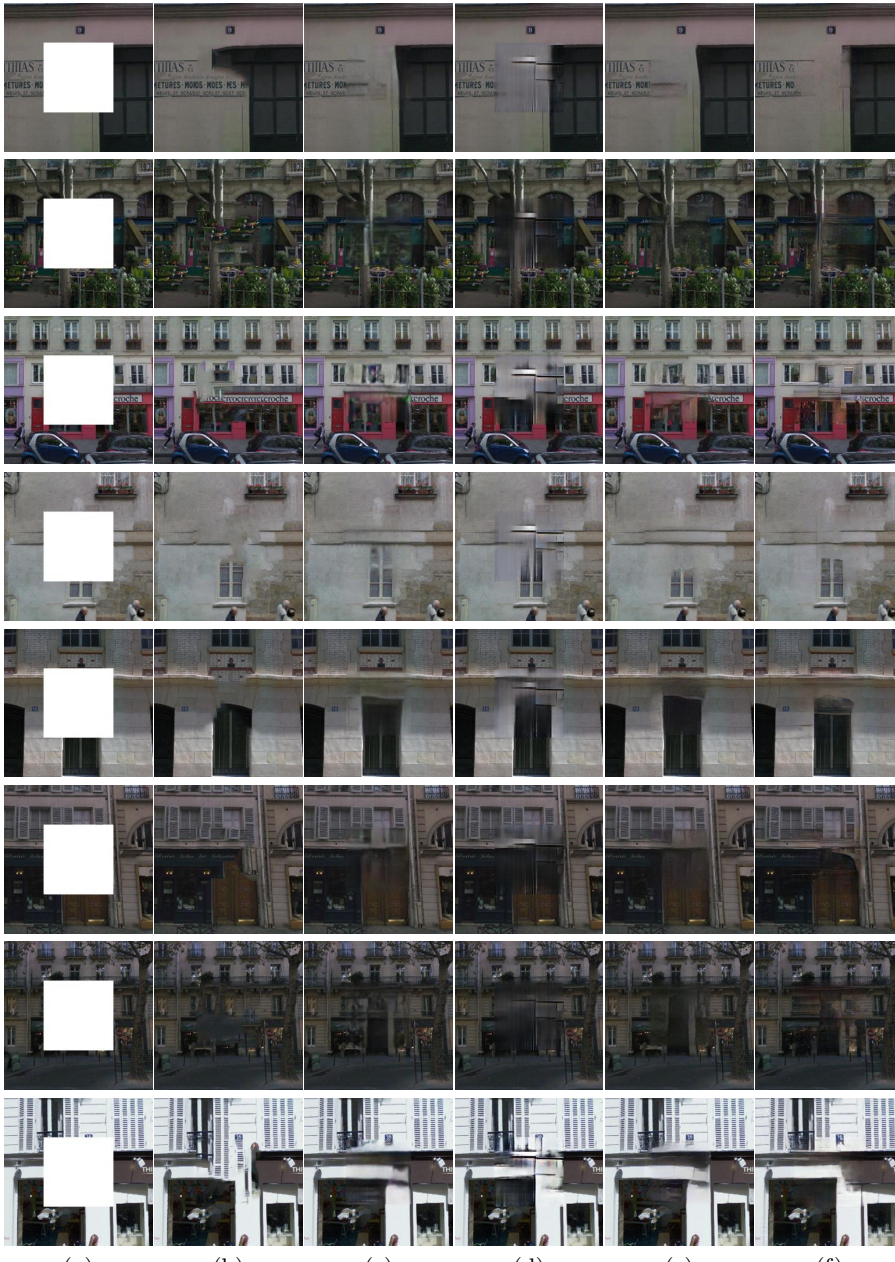
Table 2. The architecture of the G network. "IN" represents InstanceNorm and "LReLU" donates leaky ReLU with the slope of 0.2.

The architecture of generative model G	
Input: Image ($256 \times 256 \times 3$)	
[Layer 1]	Conv. (4, 4, 64), stride=2;
[Layer 2]	$LReLU$; Conv. (4, 4, 128), stride=2; IN;
[Layer 3]	$LReLU$; Conv. (4, 4, 256), stride=2; IN;
[Layer 4]	$LReLU$; Conv. (4, 4, 512), stride=2; IN;
[Layer 5]	$LReLU$; Conv. (4, 4, 512), stride=2; IN;
[Layer 6]	$LReLU$; Conv. (4, 4, 512), stride=2; IN;
[Layer 7]	$LReLU$; Conv. (4, 4, 512), stride=2; IN;
[Layer 8]	$LReLU$; Conv. (4, 4, 512), stride=2;
[Layer 9]	$ReLU$; DeConv. (4, 4, 512), stride=2; IN;
	Concatenate(Layer 9, Layer 7);
[Layer 10]	DeConv. (4, 4, 512), stride=2; IN;
	Concatenate(Layer 10, Layer 6); $ReLU$;
[Layer 11]	DeConv. (4, 4, 512), stride=2; IN;
	Concatenate(Layer 11, Layer 5); $ReLU$;
[Layer 12]	DeConv. (4, 4, 512), stride=2; IN;
	Concatenate(Layer 12, Layer 4); $ReLU$;
[Layer 13]	DeConv. (4, 4, 256), stride=2; IN;
	Concatenate(Layer 13, Layer 3); $ReLU$;
[Layer 14]	**Guidance loss layer**;
[Layer 15]	**Shift-connection layer**;
[Layer 16]	DeConv. (4, 4, 128), stride=2; IN;
	Concatenate(Layer 16, Layer 2); $ReLU$;
[Layer 17]	DeConv. (4, 4, 64), stride=2; IN;
	Concatenate(Layer 17, Layer 1); $ReLU$;
[Layer 18]	$ReLU$; DeConv. (4, 4, 3), stride=2; $Tanh$;
Output: Final result ($256 \times 256 \times 3$)	



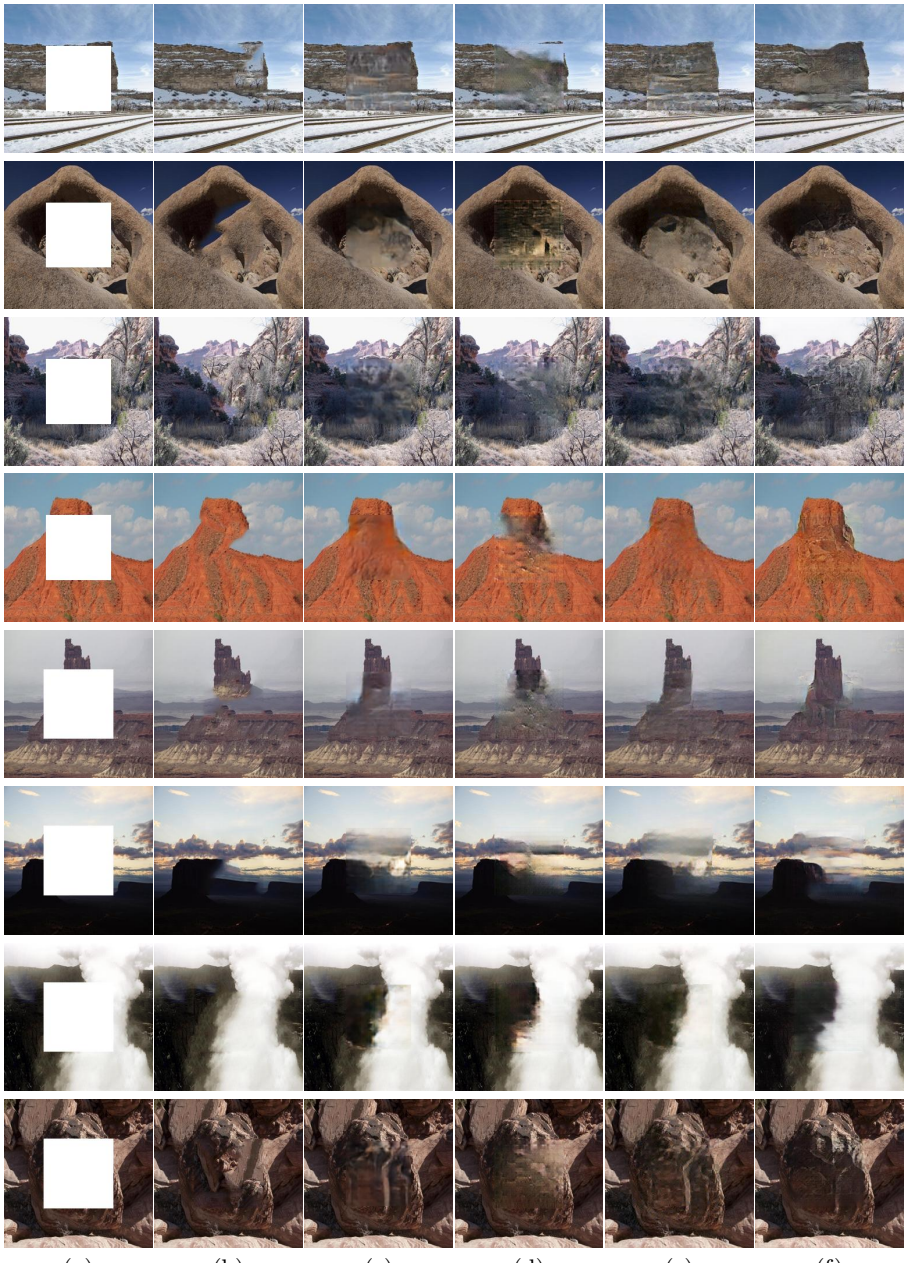
(a) (b) (c) (d) (e) (f)

Fig. 14. Qualitative comparisons on Paris StreetView. From the left to the right are: (a) input, (b) Content-Aware Fill [1], (c) context encoder [2], (d) pix2pix [39], (e) MNPS [4] and (f) Ours. All images are scaled to 256×256.



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

Fig. 15. Qualitative comparisons on Paris StreetView. From the left to the right are: (a) input, (b) Content-Aware Fill [1], (c) context encoder [2], (d) pix2pix [39], (e) MNPS [4] and (f) Ours. All images are scaled to 256×256.



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

Fig. 16. Qualitative comparisons on Places. From the left to the right are: (a) input, (b) Content-Aware Fill [1], (c) context encoder [2], (d) pix2pix [39], (e) MNPS [4] and (f) Ours. All images are scaled to 256×256.

C.2 More object removal on real world images by our Shift-Net

We apply our model trained on Paris StreetView [8] or Places [9] to process object removal on real world images, as shown in Fig. 17 for results. These real world images are complex for large area of distractors and complicated background. Even so, our model can handle them well, which indicates the effectiveness, applicability and generality of our model.

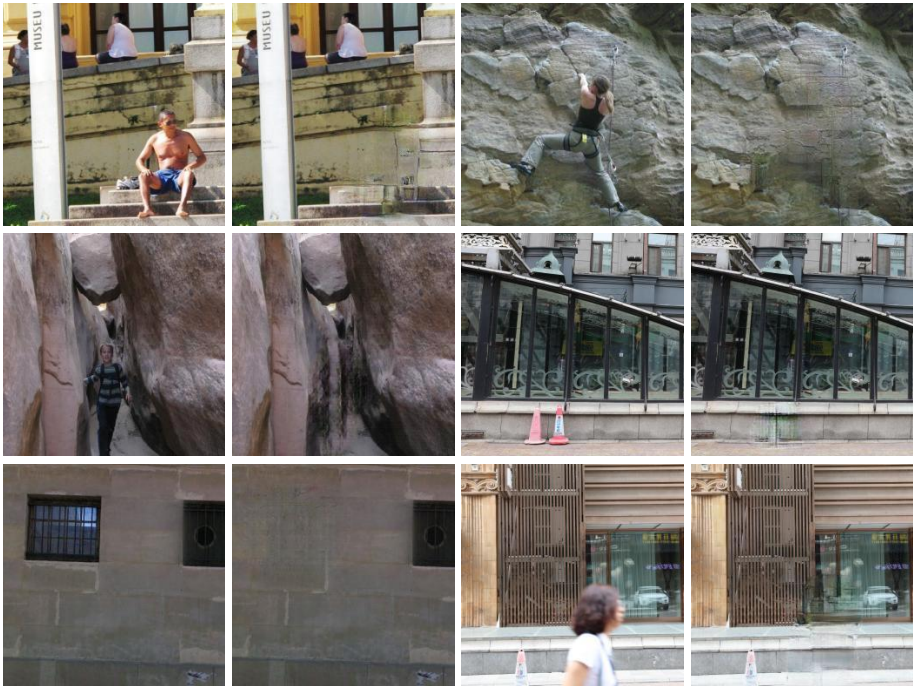


Fig. 17. Object removal on real images.

References

1. Goldman, D., Shechtman, E., Barnes, C., Belaunde, I., Chien, J.: Content-aware fill. https://research.adobe.com/project/content-aware-fill
2. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: Feature learning by inpainting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2536–2544
3. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: Patchmatch: a randomized correspondence algorithm for structural image editing. In: ACM Transactions on Graphics (TOG). Volume 28., ACM (2009) 24
4. Yang, C., Lu, X., Lin, Z., Shechtman, E., Wang, O., Li, H.: High-resolution image inpainting using multi-scale neural patch synthesis. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (July 2017)
5. Criminisi, A., Perez, P., Toyama, K.: Object removal by exemplar-based inpainting. In: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on. Volume 2., IEEE (2003) II–II
6. Le Meur, O., Gautier, J., Guillemot, C.: Examplar-based inpainting based on local geometry. In: Image Processing (ICIP), 2011 18th IEEE International Conference on, IEEE (2011) 3401–3404
7. Xu, Z., Sun, J.: Image inpainting by patch propagation using patch sparsity. IEEE transactions on image processing **19**(5) (2010) 1153–1165
8. Doersch, C., Singh, S., Gupta, A., Sivic, J., Efros, A.: What makes paris look like paris? ACM Transactions on Graphics **31**(4) (2012)
9. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (2017)
10. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
11. Barnes, C., Shechtman, E., Goldman, D.B., Finkelstein, A.: The generalized patchmatch correspondence algorithm. In: European Conference on Computer Vision, Springer (2010) 29–43
12. Drori, I., Cohen-Or, D., Yeshurun, H.: Fragment-based image completion. In: ACM Transactions on graphics (TOG). Volume 22., ACM (2003) 303–312
13. Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. In: Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on. Volume 2., IEEE (1999) 1033–1038
14. Jia, J., Tang, C.K.: Image repairing: Robust image synthesis by adaptive nd tensor voting. In: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on. Volume 1., IEEE (2003) I–I
15. Jia, J., Tang, C.K.: Inference of segmented color and texture description by tensor voting. IEEE Transactions on Pattern Analysis and Machine Intelligence **26**(6) (2004) 771–786
16. Komodakis, N.: Image completion using global optimization. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on. Volume 1., IEEE (2006) 442–452
17. Komodakis, N., Tziritas, G.: Image completion using efficient belief propagation via priority scheduling and dynamic pruning. IEEE Transactions on Image Processing **16**(11) (2007) 2649–2661
18. Pritch, Y., Kav-Venaki, E., Peleg, S.: Shift-map image editing. In: Computer Vision, 2009 IEEE 12th International Conference on, IEEE (2009) 151–158

19. Simakov, D., Caspi, Y., Shechtman, E., Irani, M.: Summarizing visual data using bidirectional similarity. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE (2008) 1–8
20. Sun, J., Yuan, L., Jia, J., Shum, H.Y.: Image completion with structure propagation. ACM Transactions on Graphics (ToG) **24**(3) (2005) 861–868
21. Wexler, Y., Shechtman, E., Irani, M.: Space-time video completion. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on. Volume 1., IEEE (2004) I–I
22. Wexler, Y., Shechtman, E., Irani, M.: Space-time completion of video. IEEE Transactions on pattern analysis and machine intelligence **29**(3) (2007)
23. Köhler, R., Schuler, C., Schölkopf, B., Harmeling, S.: Mask-specific inpainting with deep neural networks. In: German Conference on Pattern Recognition, Springer (2014) 523–534
24. Ren, J.S., Xu, L., Yan, Q., Sun, W.: Shepard convolutional neural networks. In: Advances in Neural Information Processing Systems. (2015) 901–909
25. Xie, J., Xu, L., Chen, E.: Image denoising and inpainting with deep neural networks. In: Advances in Neural Information Processing Systems. (2012) 341–349
26. Yeh, R.A., Chen, C., Lim, T.Y., Schwing, A.G., Hasegawa-Johnson, M., Do, M.N.: Semantic image inpainting with deep generative models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2017) 5485–5493
27. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and Locally Consistent Image Completion. ACM Transactions on Graphics (Proc. of SIGGRAPH 2017) **36**(4) (2017) 107:1–107:14
28. Li, Y., Liu, S., Yang, J., Yang, M.H.: Generative face completion. arXiv preprint arXiv:1704.05838 (2017)
29. Chen, T.Q., Schmidt, M.: Fast patch-based style transfer of arbitrary style. arXiv preprint arXiv:1612.04337 (2016)
30. Dumoulin, V., Shlens, J., Kudlur, M.: A learned representation for artistic style. arXiv preprint arXiv:1610.07629 (2016)
31. Gatys, L.A., Ecker, A.S., Bethge, M.: A neural algorithm of artistic style. arXiv preprint arXiv:1508.06576 (2015)
32. Gatys, L.A., Ecker, A.S., Bethge, M., Hertzmann, A., Shechtman, E.: Controlling perceptual factors in neural style transfer. arXiv preprint arXiv:1611.07865 (2016)
33. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. arXiv preprint arXiv:1703.06868 (2017)
34. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: European Conference on Computer Vision, Springer (2016) 694–711
35. Li, C., Wand, M.: Combining markov random fields and convolutional neural networks for image synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2479–2486
36. Luan, F., Paris, S., Shechtman, E., Bala, K.: Deep photo style transfer. arXiv preprint arXiv:1703.07511 (2017)
37. Ulyanov, D., Lebedev, V., Vedaldi, A., Lempitsky, V.S.: Texture networks: Feed-forward synthesis of textures and stylized images. In: ICML. (2016) 1349–1357
38. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention (MICCAI). (2015)
39. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. arXiv preprint arXiv:1611.07004 (2016)

40. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv:1703.10593 (2017)
41. Mahendran, A., Vedaldi, A.: Understanding deep image representations by inverting them. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2015) 5188–5196
42. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. arXiv preprint arXiv:1609.04802 (2016)
43. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
44. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic optimization. international conference on learning representations (2015)
45. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)