Self-Prior Guided Pixel Adversarial Networks for Blind Image Inpainting

Juan Wang*,Chunfeng Yuan*,Bing Li†,Ying Deng, Weiming Hu, Stephen Maybank

Abstract—Blind image inpainting involves two critical aspects, *i.e.*, "where to inpaint" and "how to inpaint". Knowing "where to inpaint" can eliminate the interference arising from corrupted pixel values; a good "how to inpaint" strategy yields high-quality inpainted results robust to various corruptions. In existing methods, these two aspects usually lack explicit and separate consideration. This paper fully explores these two aspects and proposes a self-prior guided inpainting network (SIN). The self-priors are obtained by detecting semantic-discontinuous regions and by predicting global semantic structures of the input image. On the one hand, the self-priors are incorporated into the SIN, which enables the SIN to perceive valid context information from uncorrupted regions and to synthesize semantic-aware textures for corrupted regions. On the other hand, the self-priors are reformulated to provide a pixel-wise adversarial feedback and a high-level semantic structure feedback, which can promote the semantic continuity of inpainted images. Experimental results demonstrate that our method achieves state-of-the-art performance in metric scores and in visual quality. It has an advantage over many existing methods that assume "where to inpaint" is known in advance. Extensive experiments on a series of related image restoration tasks validate the effectiveness of our method in obtaining high-quality inpainting.

Index Terms—Blind image inpainting, semantic-discontinuity detection, layout map prediction, pixel generative adversarial network

1 Introduction

Image inpainting is an active topic in computer vision, and numerous applications benefit from it. Image inpainting methods generally require a clear knowledge about the location of any corrupted regions. This knowledge is typically represented using a binary mask. Most image inpainting methods assume the mask to be known in advance. We call this case non-blind setting. However, accurate masks are not available in many scenes. *Blind image inpainting* aims to inpaint an image with unknown corrupted regions. This blind setting is more effective in real-world applications.

Due to the lack of masks, a greater challenge is posed for blind image inpainting. It is required to decide "where to inpaint", *i.e.*, the location of corruptions. In applications the "corruption" can be divided into two types. The first type does not exist in the original image. It is imposed by external factors in applications, such as repairing corrupted artworks, restoring old photos or removing watermarks. The corruption may be splatted inks, creases or watermarks, respectively. The second type includes corruptions that belong to the original image, such as distracting objects and occlusions. In this paper, we mainly focus on the first type of "corruption".

Existing works usually regard the blind inpainting problem as an image restoration problem. Leveraging the powerful representation ability of convolutional neural networks (CNNs), they directly map a corrupted image to a complete image [1], [2], [3], [4]. However, the uncertainty of the size, shape, color and transparency of the corruption set a huge barrier for these methods, and the inpainted results are often accompanied by color discrepancies and blurring especially when the corrupted regions are large. Other methods separate the image and the corruption by capturing their pattern differences. For example, three decoders are adopted in [5] to predict the mask, the clean image and the corruption simultaneously. A two-stage network is designed in both [6] and [7], where a mask is firstly estimated and then it is used to guide the inpainting.

In addition, both blind and non-blind inpainting face the "how to inpaint" problem, especially when the regions to be inpainted take up a large part of the image. Traditional methods usually sample and paste similar patches from databases or from the image into the corrupted regions [8], [9] or propagate background data into the corrupted regions by following a diffusion process typically implemented using differential operators [10], [11]. However, their generated inpaintings are often unnatural and implausible due to insufficient semantic information. In recent years CNNs have produced remarkable advances in inpainting, but there is still a lot of room for improvement. Some CNNs-based methods employ additional networks. For example, Nazeri *et al.* [12] and Xiong et al. [13] respectively introduce an edge map and a saliency map to boost inpainted results. Good performance is obtained. However, the inpainted results of [12] are sensitive to the parameter settings of the edge detector. In addition, some methods progressively repair large corrupted regions, working from boundary to centre or from coarse to fine [14], [15], [16], [17]. These methods generally divide the image inpainting process into several stages

- * *The first two authors contribute equally to this paper.*
- † *Bing Li is the corresponding author.*
- *J. Wang, C. Yuan, B. Li and W. Hu are with the CAS Center for Excellence in Brain Science and Intelligence Technology, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, University of Chinese Academy of Sciences, Beijing 100190, China. E-mail: jun_wang@ia.ac.cn, {cfyuan, bli, wmhu}@nlpr.ia.ac.cn.*
- *Y. Deng is with the School of Aeronautical Manufacturing Engineering, Nanchang Hangkong University, Nanchang 330063, China. E-mail: dengying@nchu.edu.cn.*
- *S. Maybank is with the Department of Computer Science and Information Systems, Birkbeck College, Malet Street, London WC1E 7HX, United Kingdom. E-mail: sjmaybank@dcs.bbk.ac.uk.*

and input the result of the current stage to the next stage for refinement. In these methods, different stages typically share the same model. However, due to the variations in the data distribution in different stages, it is difficult to make the training converge. More recently, there have emerged some works utilizing transformers [18] for image inpainting [19], [20], [21], which achieve competitive and even better performance compared with CNNs-based methods.

Based on the above analysis, blind image inpainting involves two critical aspects, *i.e.*, "where to inpaint" and "how to inpaint". This is a ill-posed problem, making it necessary to leverage prior information for assistance. Traditionally, the priors are designed manually and rely on human assessment. In this paper, "where to inpaint" and "how to inpaint" are obtained automatically from the input image. They are defined as *self-priors* [22]. Consequently, a novel blind image inpainting model with two self-priors is proposed. As shown in Fig. 1, our model contains the following three innovative parts:

• **Self-prior learning networks**. They learn the two important self-priors. One is a semantic-discontinuity detection network (SDN) for estimating a soft mask. This mask shows the inpainting network "where to inpaint" by capturing the pattern difference between corruptions and images. Another is a layout prediction network (LPN) for providing a layout map. This map guides the inpainting network "how to inpaint" by inferring the global semantic structure.

• **Self-prior guided inpainting network (SIN)**. The corrupted regions are inpainted using an encoder-decoder architecture integrated with the two learnt self-priors. Guided by the estimated mask, the encoder can adaptively extract useful context information. Guided by the estimated layout map, the decoder can synthesize more realistic textures.

• **Self-prior embedded loss functions**. They improve the semantic continuity and the perceptual consistency of the inpainting. To achieve this, the two self-prior learning networks are reused to provide feedback signals for optimizing the SIN.

Overall, the proposed three modules, *i.e.,* the SDN, LPN and SIN, form a united framework in which the learnt self-priors play an important role in guiding and optimizing the inpainting network. We conduct experiments on multiple challenging datasets. Both the quantitative and qualitative results demonstrate that our model generates more accurate and visually appealing results than many existing models, even when the existing models are provided with masks. The way in which the accuracy of the self-priors affects the inpainting performance is discussed. Furthermore, we extend our model to two related image restoration tasks, *i.e.,* image super-resolution and old photo restoration. It is proved that the proposed self-priors are effective in improving the performance of these tasks.

The rest of this paper is organized as follows: Sec. 2 summarizes recent work on image inpainting. Sec. 3 introduces the proposed self-prior learning networks in detail. Sec. 4 gives an elaborate description for integrating self-priors into the inpainting process. Sec. 5 presents the self-prior embedded loss functions and formulates the overall optimization objective. In Sec. 6, we provide implementation details, performance comparisons, ablation studies as well as extended applications. Sec. 7 is a conclusion.

2 Related Work

Recent years have witnessed significant advances in image inpainting with the help of CNNs. In the early stage, most methods aim to solve "how to inpaint" in a non-blind setting. With the real-world applications increasing, more and more works also take "where to inpaint" into consideration, and a series of blind image inpainting methods have been proposed. In the following, we briefly review related work of both non-blind and blind image inpainting.

2.1 Non-Blind Image Inpainting

Given the mask, traditional image inpainting methods which rely on hand-crafted features fail to handle "how to inpaint" for large holes, since they lack high-level structural understanding. By learning from a large corpus of data, deep learning based methods can understand the semantic structure of the input image and hence they are able to generate more content semantically consistent with the surrounding. In the following, we review the non-blind image inpainting methods related to our work. These methods include variants of convolutions, progressive inpainting and GANs-based inpainting.

Variants of Convolutions. In the vanilla convolution, the convolutional filters are shared among all the input pixels. This sharing can lead to visual artefacts, such as color discrepancy and blurring, due to interference by corrupted pixel values. To address this problem, some works propose various variants of convolutions, which are typically utilized with the mask. For example, Liu *et al.* [23] propose a PConv, in which the convolution is masked and normalized to be conditioned only on clean pixel values. Yu *et al.* [24] propose a gated convolution, which learns a soft mask based on a dynamic feature gating mechanism. Navasardyan *et al.* [25] design an onion convolution. They carry out an onion-peel patch match to preserve feature continuity between corrupted and uncorrupted regions. In our work, we also propose a novel variant of convolutions, which is integrated with our automatically estimated soft mask to adaptively decide how much attention is paid to each location.

Progressive Inpainting. Inpainting large corrupted regions is particularly challenging, since there is often very limited context information. In this case the model has to infer global structure and local texture at the same time. To address this challenge, a series of works resorts to progressive models, which allow the inpainting of corrupted regions to be completed step by step. For example, Zhang *et al.* [26] adopt a curriculum learning idea, which divides the image inpainting process into multiple sub-region inpainting phases. Oh *et al.* [25] propose an onion-peel network that completes the image from boundary to center. Zeng *et al.* [27] propose to complete the image by using only high-confidence pixels at each iteration and focusing on the remaining pixels in the next iteration. Both Yu *et al.* [14] and Ma *et al.* [16] present coarse-to-fine image inpainting frameworks. In addition, some methods first obtain a structure, such as an edge map [12], [13], [28], and then employ the structure for content filling. The most relevant work to our method is [29], which also generates a segmentation map to guide the subsequent inpainting. In contrast with [29], we design a new segmentation generation and embedding

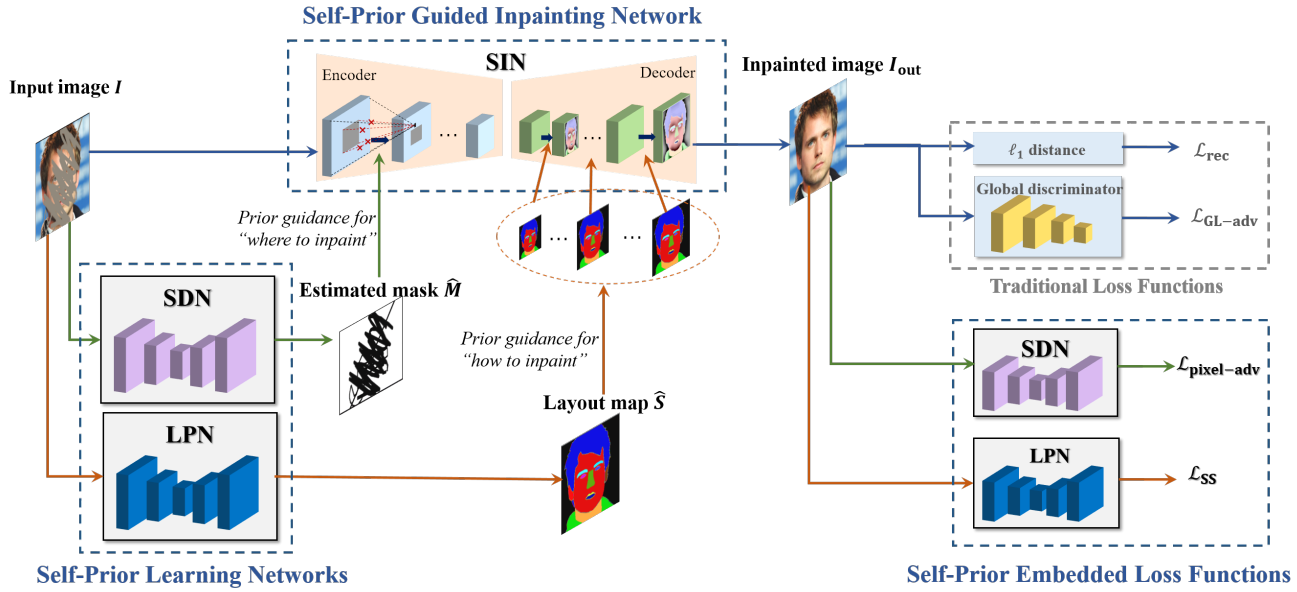


Fig. 1. Overview of our proposed blind image inpainting model, which consists of three critical parts: self-prior learning networks, self-prior guided inpainting network and self-prior embedded loss functions.

method, and present an improved loss function based on the segmenter.

GANs-based Inpainting. Recently, GANs-based image inpainting methods have yielded promising results. In these methods, the inpainting network generates plausible contents for the corrupted regions and a discriminator is introduced to provide the generator with a learning signal to synthesize realistic images. Pathak *et al.* [30] are the pioneers that introduce GANs into the image inpainting. For the generator, the size of the receptive field should be large enough to ensure that the model can access a complete scene. For this purpose, the dilated convolution (DC) [14], [31], non-local operation [16], [32], attention mechanisms [14], [33] and multi-scale networks [34], [35] are widely used to obtain long-distance information. The discriminator is usually a network which classifies the input image as real or fake. However, a single fake/non-fake signal is not sufficient to learn the distinction between natural images and synthesized images. For this reason, PatchGANs [36] are adopted in [31], [36], [24], [28], [34], [35] to decide whether each small patch is real or fake. In this paper, we extend PatchGANs to pixel-GANs in order to provide pixel-wise feedback to the generator to remove boundary artefacts, which often appear in the border of inpainted regions.

2.2 Blind Image Inpainting

In blind image inpainting the mask that indicates the location of corrupted regions is unknown. Therefore, the blind setting requires an extra consideration about "where to inpaint" before "how to inpaint". To address this issue, Xie *et al.* [37] adopt a deep neural network with a pre-trained sparse denoising auto-encoder and demonstrate its ability in blind image inpainting. Cai *et al.* [1] propose a lightweight network called BICNN, which takes a corrupted image of any size as input. They first estimate which pixels are contained in the corruption and subsequently reconstruct a latent image. Wang *et al.* [6] propose a two-stage framework consisting of mask prediction and robust inpainting. A discriminative model is first employed to predict semantically inconsistent areas. The predicted mask is used to guide the inpainting process. Kim *et at.* [38] train a video decaptioning model by a residual learning algorithm to automatically detect corrupted pixels and prevent global tone distortion. Watermark removal and old photo restoration are two important applications of blind image inpainting. For the watermark removal, both Hertz *et al.* [5] and Liu *et al.* [7] propose to separate the watermark from the image to avoid the color discrepancy. For the old photo restoration, Wan *et al.* [4] train two variational autoencoders to map old photos and clean photos into two latent spaces and learn their translation. Furthermore, they develop a recurrent transformer network [39] to exploit the hidden knowledge learned from the adjacent frames to infer the occluded information for old film restoration.

Most of these methods take the corrupted image as an input, but this may cause unpleasant visual artefacts. Our method is closest in spirit to [6], where a soft mask is estimated. The main difference is that we combine the soft mask with our proposed variant of convolutions for adaptive attention.

3 Self-Prior Learning Networks

Let $I \in \mathbb{R}^{H\times W\times 3}$ be an input image, which is a degraded version of the ground truth image $I_{\text{gt}} \in \mathbb{R}^{H\times W\times 3}$ corrupted by visual signals $N \in \mathbb{R}^{H\times W\times 3}$ (*e.g.*, constant values, random noise and watermarks) in some regions. Let $M \in \mathbb{R}^{H\times W}$ denote a binary mask, indicating the locations of N, where each pixel $M(i,j)$ is 0 for a corrupted pixel or 1 for a clean pixel. Mathematically, the degraded image I is formulated as:

$$I = \underbrace{I_{gt} \odot M}_{\text{uncorrupted regions}} + \underbrace{(1-\phi)I_{gt} \odot (\mathbf{1}-M) + \phi N \odot (\mathbf{1}-M)}_{\text{corrupted regions}}, \tag{1}$$

where $\odot$ denotes the Hadamard product, and $0 < \phi \leq 1$ is a scalar denoting the intensity of N, $\phi = 1$ if N is opaque, otherwise $\phi \in (0, 1)$ if N is semi-transparent. Let $\hat{I} \in \mathbb{R}^{H \times W \times 3}$ be the output image after inpainting. The aim of an image inpainting model is to make $\hat{I}$ as close as possible to the ground truth, *i.e.*, minimizing $\mathcal{L}(\hat{I}, I_{\text{gt}})$, where $\mathcal{L}$ denotes a metric evaluating the discrepancy between $\hat{I}$ and I_{gt}.

Blind image inpainting is more intractable than the non-blind setting, since the lack of the mask M makes it difficult to distinguish between trustworthy and unreliable regions. Inpainting is even more difficult if the corrupted signals N cover a large part of I. In our model, two types of prior information are automatically learnt from the input image to solve these two difficult problems. On the one hand, a "where to inpaint" prior is learnt to guide the inpainting network to bypass corrupted regions. On the other hand, a "how to inpaint" prior is exploited to allow the inpainting network to progressively inpaint the corrupted regions from coarse to fine. In the following, we introduce the two self-prior learning networks in detail.

3.1 Learning "Where to Inpaint"

The corruption pattern of the matrix N in Eq. (1) is typically random and irregular. It is substantially different from the ground truth image pattern [6], making it possible to locate the corrupted regions. Since the corruption pattern breaks the semantic structure of the image, we frame *learning "where to inpaint"* as a semantically-discontinuity detection problem. In the proposed model, we construct a semantically-discontinuity detection network (SDN) to estimate a soft mask, *i.e.*, $f_{\text{SDN}} : I \rightarrow M$, by distinguishing the two different patterns, *i.e.*, the image pattern and the corruption pattern, in a corrupted image.

Semantic-discontinuity detection is a dense prediction task, where each pixel is judged whether it has a value continuous with its surroundings from the perspective of global semantics. Intuitively, we adopt a U-Net [40] architecture for SDN, which is composed of an encoder and a decoder. The encoder takes a corrupted image as input and exploits its latent representations by perceiving the inherent semantic structure. The decoder finds the semantically-discontinuity regions using the latent representations. It then outputs a soft mask such that each pixel value is equal to the probability that the corresponding pixel in the image is clean. The encoder consists of four *conv-BN-ReLU* modules (*conv*: convolution, *BN:* Batch Normalization) except that the first does not have a BN layer. In each module, strided convolutions are adopted to down-sample an image into a compact latent representation. With the encoding feature map down-sampled with a factor of 2 by strided convolutions, its channel number gradually grows in size from 64 to 512. The decoder contains four *upsample-concat-conv-BN-LeakyReLU* modules (*concat*: concatenation). In each module, a skip connection is adopted to concatenate the decoding features and the corresponding-scale encoding features along the channel. With successive up-sampling with a factor of 2, the decoding feature map gradually grows in size. Meanwhile, the channel number decreases from 512 to 64. Finally, a sigmoid activation layer follows the last decoder module to map the data to the range of [0,1].

The loss function of SDN is defined as the binary cross entropy (BCE) between the ground truth mask $M \in \mathbb{R}^{H \times W}$ and the network output $\hat{M} \in \mathbb{R}^{H \times W}$. Let R and $\overline{R}$ denote the corrupted and uncorrupted regions, respectively. The loss function is formulated as follows:

$$\begin{aligned}
&\mathcal{L}_{\text{SDN}}(M, \hat{M}) \\
&= -\sum_{(i,j)} [M(i,j) \log(\hat{M}(i,j)) + (1 - M(i,j)) \log(1 - \hat{M}(i,j))], \\
&= -[\sum_{(i,j) \in \overline{R}} \log(\hat{M}(i,j)) + \sum_{(i,j) \in R} \log(1 - \hat{M}(i,j))],
\end{aligned} \tag{2}$$

where (i, j) denotes the coordinates of each pixel.

3.2 Learning "How to Inpaint"

Although CNNs-based image inpainting methods surpass traditional methods by a long way, they still struggle to recover vivid details, especially if there are large holes. The proposed method reduces this problem by disentangling the inference of global structure and the synthesis of local details into two separate steps. Image structure is well represented in the layout map, which contains a set of segments. The pixels in a given segment all belong to the same category. The layout map draws the outline of the objects with different categories, providing an explicit guidance for the subsequent inpainting. Consequently, a layout prediction network (LPN) is constructed to learn a mapping from the corrupted image to the layout map S, *i.e.*, $f_{\text{LPN}} : I \rightarrow S$, as a prior to guide the progressive inpainting. Compared with learning a mapping from I to I_{gt}, learning a mapping to S is much easier, since it only involves the inference of high-level semantics and it is free from the synthesis of low-level details.

DeepLabV3Plus [41] (V3Plus is omitted later for simplicity) is a well-known image segmentation algorithm. Benefiting from a detailed architecture design, it produces accurate predictions along object boundaries. Nevertheless, this model has a limited ability to produce a complete layout map for a corrupted image. To obtain more accurate maps, we increase the network depth and employ the improved model as our LPN. Since the corrupted regions of the input image can degrade the prediction accuracy, we simply binarize the estimated soft mask and multiply the binarized version with the image in order to remove corrupted signals. The resulting image is input to our LPN for complete layout prediction. Post processing is employed to remove noise and smooth edges using morphological operators.

The loss function of the LPN is defined as the multi-category focal loss [42] between the ground truth layout map $S \in \mathbb{R}^{H \times W \times C}$ and the network output $\hat{S} \in \mathbb{R}^{H \times W \times C}$, where $C > 1$ denotes the number of categories, including the background category. Each element $S(i, j)$ is a one-hot encoding, where the location of the correct label is 1 and otherwise 0. The loss function of the LPN is formulated as follows:

$$\mathcal{L}_{\text{LPN}}(S, \hat{S}) = -\sum_{c=1}^{C} (1 - \hat{S}(i,j,c))^{\gamma} \log(\hat{S}(i,j,c)), \tag{3}$$

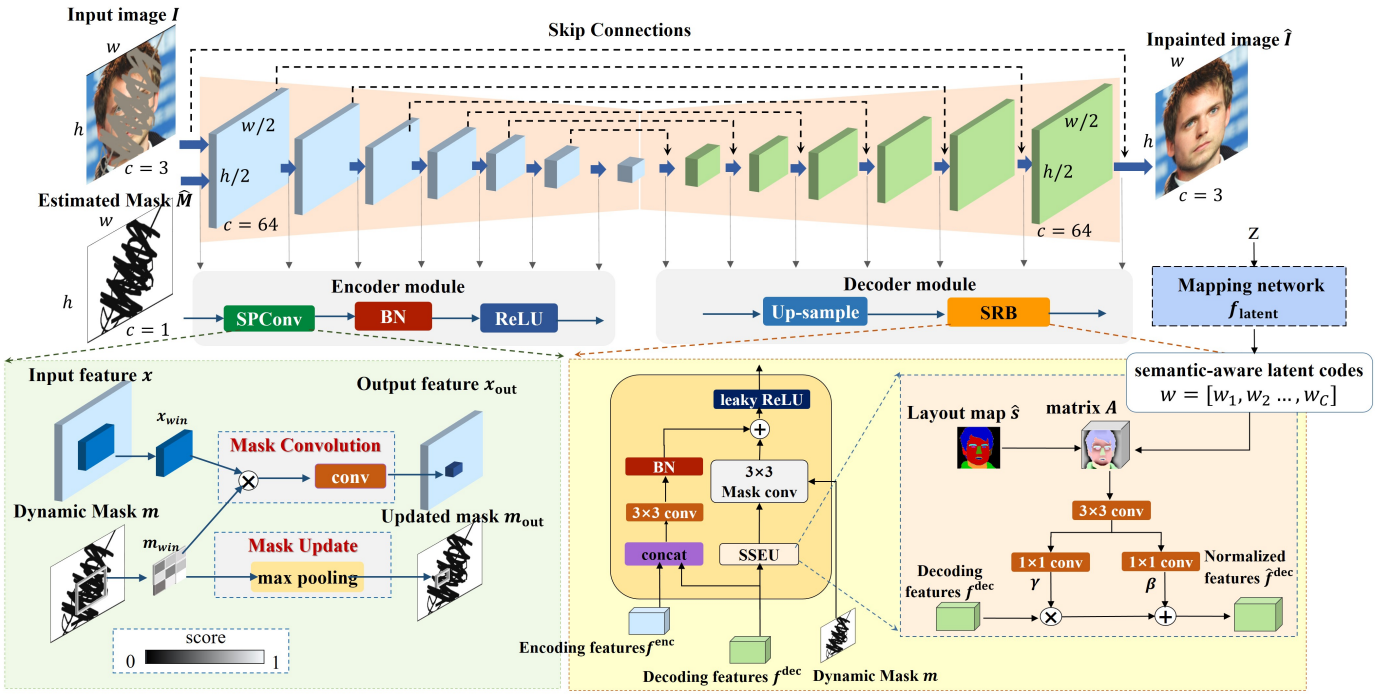


Fig. 2. Network architecture of our SIN, where *SPConv* (lower left-hand side) and *SSEU* (lower right-hand side) are critical components to utilize the two self-priors for inpainting.

where $\gamma \geq 0$ is a focusing parameter and $(1-\hat{S}(i,j,c))^{\gamma}$ is a modulating factor, which is used to down-weight easily classified pixels and focus training on hard pixels.

4 Self-Prior Guided Inpainting Network

With the soft mask and layout map, a SIN is constructed to utilize the two important self-priors $\hat{M}$ and $\hat{S}$ to generate a complete image I, *i.e.*, $f_{\text{SIN}} : I, \hat{M}, \hat{S} \rightarrow I_{\text{gt}}$. The network architecture of SIN is an encoder-decoder based U-Net formulation, as shown in Fig. 2. Moreover, an improved convolution operation called *soft-partial convolution* (SPConv) is proposed to embed M into the encoder for guiding SIN "where to inpaint"; a novel unit architecture called *semantic structure embedding unit* (SSEU) is proposed to embed S into the decoder for guiding SIN "how to inpaint". Specifically, the encoder adopts seven *SPConv-BN-ReLU* modules except that the first BN layer is omitted from the first module. Each SPConv adopts a stride of 2 to down-sample an image and gradually increases the channel number from 64 to 512. The decoder is also composed of seven modules, and each module begins with an up-sampling layer with a factor of 2, which is followed by a SSEU ResNet block (SRB). Symmetrical to the encoder, the decoding feature map gradually grows its size and decreases its channel number from 512 to 64, which are finally consistent with the input image I. In the following, we describe the critical components of the SIN, *i.e.*, SPConv and SSEU, in detail.

4.1 Soft-Partial Convolution

Recall that by detecting the semantic-discontinuity regions, our SDN estimates a soft mask $\hat{M}$ in which each pixel value is in the range of [0,1]. Each pixel in I has a score which is large if the pixel is likely to be in an uncorrupted region. The SPConv computes the weighted response of each pixel according to the score. The corrupted pixel values have low scores and contribute less to the response. The pixels with high scores dominate the response.

The flow chart of SPConv is shown in the bottom left of Fig. 2. It takes a feature map x and a dynamic mask m as input, and implements the following two steps:

(i) Mask convolution. In the vanilla convolution, the convolutional filters are shared among all the input pixels. This sharing can lead to visual artefacts, such as color discrepancy and blurring, due to interference by corrupted pixel values. In contrast, the proposed SPConv adaptively extracts information from each pixel according to the score of the pixel. Let x_{win} and m_{win} denote the current region covered by the convolution window W on x and m. Let b denote the bias. Then the value of the output feature map x_{out} at the (u,v)-th position is given by:

$$x_{\text{out}}(u,v) = W^T(x_{\text{win}} \odot m_{\text{win}}) \frac{\sum_{(i,j)\in R_{\text{win}}}(\mathbf{1}(i,j))}{\sum_{(i,j)\in R_{\text{win}}}(m_{\text{win}}(i,j))} + b, \tag{4}$$

where R_{win} denotes the current region, and $\mathbf{1}$ has the same shape as W.

(ii) Mask update. After mask convolution, some corrupted regions are filled with generated contents computed from surrounding pixels with different scores. Correspondingly, the mask should also adjust the score to cater for the updated feature map. Intuitively, the higher score a pixel has, the greater the contribution it makes to the updated value. Consequently, we update the mask by selecting the maximum score in the current window, *i.e.*,

$$m_{\text{out}}(u,v) = \max_{(i,j)\in R_{\text{win}}} m(i,j). \tag{5}$$

With successive mask updates, the pixel values with higher scores are gradually propagated into the corrupted regions.

Note that the dynamic mask m is initialized to the estimated soft mask $\hat{M}$ at the first SPConv layer of our SIN. After each update, the output mask is used as the input of next layer. After being processed by seven SPConv layers in the encoder, the corrupted regions will gradually shrink away.

4.2 Semantic Structure Embedding Unit

Each semantic item, such as eyes, nose and mouth for face images, or dog, tree and boat for natural-scene images, has an inherent structure, which makes modeling the images possible. However, modeling images with complicated semantics is still challenging, since the model is required to infer the semantics and synthesize semantically consistent details at the same time. Fortunately, we have obtained the predicted layout map $\hat{S}$, which contains high-level semantics. Under its guidance, the SIN only needs to focus on synthesizing the low-level details.

In order to embed $\hat{S}$ in the SIN for guiding the inpainting, we design the semantic structure embedding unit (SSEU). The core technique of the SSEU is the adaptive affine transform (AAT) [43], which has been widely used for transferring the style information from an image to another image. Inspired by this, we resort to the AAT to transfer the semantic information contained in $\hat{S}$ to the corrupted image I. The first step of the SSEU is to transform $\hat{S}$ into a feature representation. To this end, we train a non-linear mapping function f_{latent} to learn the feature representation of each semantic item. The mapping function network consists of 8 multi-layer perceptrons (MLP), which takes a normally distributed noise vector $z \sim N(0, I)$ as input and outputs the latent code w. The code w has C channels, *i.e.*, $w = [w_1, ..., w_C]$, where C denotes the number of semantic categories, including the background category. Each channel w_c, which corresponds to one semantic code, is a vector with the dimension of 512. We call w the semantic-aware latent codes. Then we broadcast w_c to its corresponding semantic regions in $\hat{S}$ and form a new matrix $A \in \mathbb{R}^{H \times W \times 512}$. The construction of A is formulated as:

$$A(i, j) = w_c, \qquad \text{if } \underset{k \in [1,C]}{\arg\max}\, \hat{S}(i, j, k) = c. \tag{6}$$

As a result, we obtain the feature representation A of $\hat{S}$. By training, using the inherent structure of each semantic item, f_{latent} can effectively learn the semantic-aware latent codes, and thus make A semantically meaningful.

The working mechanism of SSEU is illustrated in the bottom right of Fig. 2. The SSEU in the l-th decoder module takes two inputs: i) features $f_l^{\text{dec}} \in \mathbb{R}^{h \times w \times n}$, where h, w and n denote the height, width and channel number, f_l^{dec} is output from the upsample layer of the l-th decoder module, and ii) features $a_l \in \mathbb{R}^{h \times w \times 512}$, which is a down-sampled version of A. Then, an AAT is learnt to transfer the semantic information of a_l to f_l^{dec}. This is achieved by learning two affine parameters from a_l, including a scaling factor $\gamma \in \mathbb{R}^{h \times w \times n}$ and a bias factor $\beta \in \mathbb{R}^{h \times w \times n}$, which are used to normalize f_l^{dec}. Mathematically, the normalized values of f_l^{dec} at the k-th channel are formulated as follows:

$$\hat{f}_l^k = \gamma^k \frac{f_l^k - \mu^k}{\sigma^k} + \beta^k, \tag{7}$$

where the superscript "dec" is omitted for simplicity, $\mu^k \in \mathbb{R}^{h \times w}$ and $\sigma^k \in \mathbb{R}^{h \times w}$ denote the mean and standard deviation of f_l^{dec} in the k-th channel, and γ^k and β^k in (7) are the k-th channels of γ and β, respectively.

Since the contents in the uncorrupted regions can be warped after the normalization, we leverage the mask convolution defined in Sec. 4.1 to reduce the impact of the normalization on these regions. We re-use $\hat{m}_l \in \mathbb{R}^{h \times w}$, which is obtained by the mask update defined in Sec. 4.1 in the encoder, to mask the uncorrupted regions in the output of the SSEU, *i.e.*, $\hat{f}_l \in \mathbb{R}^{h \times w \times n}$. Specifically, taking $\hat{f}_l$ and (1-$\hat{m}_l$) as input, the mask convolution is performed according to Eq. (4).

Furthermore, we put the SSEU into a block in a similar form with the residual connection [44], which we call SRB. As shown in Fig. 2, SRB has two branches. The left branch adopts the *concat-conv-BN* architecture as the U-Net. The right branch contains a SSEU and a mask convolution. The proposed SRB can be expressed in the form $f_{\text{SRB}} = W^T \hat{f}_l + f_{\text{left}}$ as in the ResNet block, where W denotes the mask convolution, f_{left} and f_{SRB} denote the outputs of the left branch and the SRB, respectively. When W is set as zero, the SIN degenerates into the vanilla U-Net. The right branch introduces the high-level semantic information about the corrupted regions from the layout map, which facilitates the synthesis of fine-grained details.

5 Self-Prior Embedded Loss Functions

Traditionally, image generation tasks adopt *reconstruction loss*, *i.e.*, the ℓ_1 or ℓ_2 distance between the generated image and the ground truth in the pixel domain as the optimization objective. However, it is well-known that reconstruction loss leads to blurred results. In recent years, perceptual loss and adversarial loss have greatly improved the quality of generated images, with sharp edges and rich details. Nevertheless, when the two losses are employed in the image inpainting task, unpleasant visual artefacts, such as boundary artefacts and semantic inconsistencies still occur. To alleviate these problems, two self-prior embedded loss functions, *pixel adversarial loss* and *semantic structure loss*, are defined by reusing the self-prior learning networks. In the following, we introduce the two losses in detail and finally define the overall loss function for the SIN.

5.1 Pixel Adversarial Loss

In the current state-of-the-art GAN models, the discriminator is generally a network which classifies the input image as real or fake. However, there is too little information in one bit to classify the image. In image inpainting, boundary artefacts frequently occur at the edge of inpainted regions and can easily fool the discriminator, but are evident to a human viewer. An improved dense-prediction discriminator is defined to provide a pixel-wise feedback to the generator for removing the boundary artefact.

The SDN is a pixel-wise binary classification network, which estimates a soft mask by distinguishing the corrupted pixel values from the uncorrupted pixel values. In addition, we expect that the SDN can also discriminate between the synthesized pixel values and the uncorrupted pixel

values in an inpainted image. To this end, we construct improved GANs called pixel-GANs, where the SIN plays a role as a generator G, and the SDN is employed as the dense-prediction discriminator D. During their adversarial training, the SIN maps a corrupted image I to a clean image, while the SDN classifies the input image as real or fake in a pixel-wise manner. By learning to discriminate the synthesized and uncorrupted pixel values based on multi-level semantics, the SDN can effectively recognize the boundary artefact, which often appears in the boundary between the synthesized and uncorrupted regions.

Let $D(I(i,j))$ denote the decision of the discriminator at pixel (i,j). Let R and $\overline{R}$ denote the corrupted/synthesized regions and uncorrupted regions, respectively. The loss for D is computed as the mean over all the pixels as follows:

$$\begin{aligned}\mathcal{L}_D(I,\hat{I},I_{\text{gt}}) &= -\frac{1}{\Phi}\sum_{(i,j)}\log[D(I_{\text{gt}}(i,j))] \\ &- \frac{1}{\Phi}[\sum_{(i,j)\in\overline{R}}\log[D(\hat{I}(i,j))] + \sum_{(i,j)\in R}\log[1-[D(\hat{I}(i,j))]]] \\ &- \frac{1}{\Phi}[\sum_{(i,j)\in\overline{R}}\log[D(I(i,j))] + \sum_{(i,j)\in R}\log[1-[D(I(i,j))]]],\end{aligned} \tag{8}$$

where Φ is the number of pixels in the image I. The first term in Eq. (8) denotes that when the input is ground truth image I_{gt}, D learns to classify each pixel as real (labelled by 1). The second term denotes that when the inpainted image $\hat{I}$ is input, D learns to classify the synthesized and uncorrupted regions, in which the pixels are labelled by 0 and 1, respectively. To prevent the SDN from losing the ability to estimate the soft mask, the loss function of the SDN defined in Eq. (2) is used as the third term of Eq. (8), where $D(I) = \hat{M}$. Correspondingly, the objective of adversarial training for G is defined as:

$$\mathcal{L}_{\text{pix-adv}}(\hat{I}) = -\frac{1}{\Phi}\sum_{(i,j)}\log[D(\hat{I}(i,j))], \tag{9}$$

which we call *pixel adversarial loss*.

5.2 Semantic Structure Loss

To compute the perceptual loss, the VGG network [45] pre-trained on the ImageNet dataset [46] is typically adopted to extract feature maps from two images. However, there may be a semantic gap if the target dataset for training the SIN is very different from the ImageNet dataset. For example, the target set could be face images (CelebA dataset [47]) or cityscape images (CityScapes Dataset [48]). To narrow the semantic gap, we use the pre-trained LPN, which is optimized for layout map prediction on the target dataset, as a substitute for the VGG network. The pre-trained LPN incorporates rich semantic structure information in its parameters, thus we call the LPN-based perceptual loss the *semantic structure loss*.

Let $f^t_{\text{LPN}}(x)$ denote the t-th layer output of the LPN for the input x. The formulation of the semantic structure loss is mathematically expressed as follows:

$$\mathcal{L}_{\text{SS}}(\hat{I},I_{\text{gt}}) = \sum_{t=1}^{T}\frac{1}{\Theta_t}||f^t_{\text{LPN}}(\hat{I}) - f^t_{\text{LPN}}(I_{\text{gt}})||_1, \tag{10}$$

where T is the total number of layers used to calculate $\mathcal{L}_{\text{SS}}$, Θ_t is the number of elements in the t-th layer output. Concretely, we adopt the ResNet-101 [44] as the backbone of the LPN and extract the features of the 2-nd, 3-rd and 4-th pooling layers to compute $\mathcal{L}_{\text{SS}}$.

In contrast with the VGG network, our LPN provides a more specialized semantic structure similarity measurement, which is not only more effective in obtaining inpainted images semantically consistent with the ground truth, but is also more effective for training.

5.3 Overall Loss Function for SIN

The overall loss function for the SIN is the sum of the four losses: i) reconstruction loss $\mathcal{L}_{\text{rec}}$ using the ℓ_1 distance for the pixel consistency, ii) semantic structure loss $\mathcal{L}_{\text{SS}}$ for the semantic consistency, iii) global adversarial loss $\mathcal{L}_{\text{GL-adv}}$ for the distribution consistency, which is achieved by constructing a global discriminator[1], and iv) pixel adversarial loss $\mathcal{L}_{\text{pix-adv}}$ for removing the boundary artifact. This can be formulated as follows:

$$\begin{aligned}\mathcal{L}_{\text{overall}}(I,\hat{I},I_{\text{gt}}) = \frac{1}{N}\sum_{n=1}^{N}[&\mathcal{L}_{\text{rec}}(\hat{I},I_{\text{gt}}) + \varepsilon\mathcal{L}_{\text{SS}}(\hat{I},I_{\text{gt}}) \\ &+ \rho\mathcal{L}_{\text{GL-adv}}(\hat{I}) + \mu\mathcal{L}_{\text{pix-adv}}(\hat{I})],\end{aligned} \tag{11}$$

where the hyper-parameters ε, ρ and μ specify the trade-off among the four losses, and each loss is normalized by the batch size N.

6 Experiments

We introduce the experimental datasets and the parameter settings in Section 6.1 and Section 6.2, respectively. Then we compare our method with the state-of-the-art image inpainting methods in Section 6.3. The ablation studies and a discussion of our method are in Section 6.4 and Section 6.5, respectively. Finally, we extend our method to several related tasks in Section 6.6.

6.1 Datasets

We evaluated our method for image inpainting on multiple public datasets: CelebA-HQ [47], CityScapes [48], Paris Street View (PSV) [30], ImageNet [46], Places2 [49], CLWD [7] and Pascal VOC [50] datasets. All the images are resized to 256×256. In the CelebA-HQ and CityScapes datasets, each image has ground truth semantic segmentation maps with 19 categories (*i.e.*, C=19). These semantic segmentation maps can be used for the LPN training. The PSV dataset is similar to the CityScapes dataset. Therefore the two datasets share the same LPN, which is trained on CityScapes. For other datasets, we use SBD [51], which is a well-known semantic segmentation dataset with 21 categories (*i.e.*, C=21), for training the LPN.

In addition, we employ three types of masks and three types of visual signals to generate corrupted images I. The three masks include: i) quick draw irregular mask dataset[2], ii) irregular mask dataset[3] and iii) randomly generated

1. More details are provided in the appendix
2. https://github.com/karfly/qd-imd.git
3. https://nv-adlr.github.io/publication/partialconv-inpainting

TABLE 1
Quantitative comparisons of different image inpainting methods (♯ means the non-blind methods with ground truth masks, and otherwise the blind methods; † means a higher score is better and $*$ means a lower score is better)

Metrics	Mask	CelebA-HQ						Paris Street View				
		CA♯ [14]	PC♯ [23]	GC♯ [24]	EC♯ [12]	PDGAN ♯ [52]	Ours	PC♯ [23]	GC♯ [24]	EC♯ [12]	PRVS♯ [28]	Ours
$\ell_1^*(\%)$	10-20%	2.48	0.83	0.88	0.85	0.90	**0.79**	1.23	1.26	1.11	**1.05**	1.12
	20-30%	3.98	1.46	1.54	1.54	1.45	**1.43**	2.12	2.07	1.95	**1.82**	1.97
	30-40%	5.64	2.36	2.33	2.37	2.68	**2.09**	3.09	3.00	2.87	2.66	**2.61**
	40-50%	7.35	4.01	3.29	3.37	3.02	**2.99**	4.21	4.06	3.93	3.63	**3.57**
PSNR†	10-20%	25.32	33.05	32.69	32.53	32.40	**33.13**	30.76	31.42	31.05	**32.00**	31.14
	20-30%	22.09	29.10	29.45	29.19	29.64	**29.78**	27.62	28.12	28.05	**28.79**	28.65
	30-40%	19.94	27.24	27.01	26.72	27.23	**27.77**	25.51	25.80	25.98	26.62	**26.79**
	40-50%	18.41	23.46	24.98	24.67	24.39	**25.51**	23.81	23.93	24.29	24.87	**24.99**
SSIM†	10-20%	0.888	0.978	0.976	0.978	0.847	**0.986**	0.953	0.959	0.956	**0.964**	0.962
	20-30%	0.819	0.956	0.954	0.957	0.808	**0.971**	0.910	0.920	0.917	**0.928**	**0.928**
	30-40%	0.750	0.926	0.927	0.928	0.748	**0.942**	0.858	0.873	0.869	0.885	**0.887**
	40-50%	0.678	0.839	0.892	0.891	0.687	**0.909**	0.780	0.815	0.811	0.832	**0.834**
FID*	10-20%	13.74	7.63	25.49	5.03	18.32	**4.72**	27.21	53.68	27.66	21.04	**20.11**
	20-30%	17.78	8.74	29.68	7.37	21.02	**5.69**	41.97	82.41	41.72	25.32	**23.17**
	30-40%	24.07	19.21	70.37	10.66	24.63	**10.27**	55.79	137.49	60.28	40.51	**32.56**
	40-50%	37.35	27.11	75.04	14.53	27.12	**13.13**	66.72	182.02	86.52	46.70	**41.19**
IS†	10-20%	3.43	3.44	3.32	3.42	3.09	**3.97**	3.02	2.88	2.97	3.03	**3.11**
	20-30%	3.34	3.42	3.17	3.34	3.02	**3.84**	3.01	2.82	2.94	3.00	**3.07**
	30-40%	3.10	3.25	3.06	3.13	2.90	**3.61**	2.95	2.56	2.92	2.99	**3.01**
	40-50%	3.01	3.06	2.88	2.94	2.70	**3.35**	2.87	2.48	2.91	2.81	**2.97**
LPIPS*	10-20%	0.043	0.037	0.080	**0.028**	0.070	**0.028**	0.045	0.112	0.047	0.056	**0.044**
	20-30%	0.078	0.056	0.116	0.051	0.094	**0.043**	0.067	0.172	0.071	0.123	**0.062**
	30-40%	0.127	0.101	0.192	0.074	0.129	**0.070**	0.107	0.267	0.118	0.268	**0.102**
	40-50%	0.195	0.143	0.261	0.106	0.173	**0.098**	0.159	0.384	0.173	0.305	**0.134**

Metrics	Mask	Places2				CityScapes				ImageNet			
		PC♯ [23]	EC♯ [12]	PRVS♯ [28]	Ours	PIC♯ [33]	SN♯ [53]	PHD [54]	Ours	PC♯ [23]	PIC♯ [33]	PHD [54]	Ours
$\ell_1^*(\%)$	10-20%	**1.18**	1.50	1.25	1.19	**0.83**	0.88	0.97	0.85	**1.77**	1.25	2.35	1.84
	20-30%	**2.07**	2.59	2.25	2.11	**1.46**	1.54	1.70	1.54	2.30	**2.00**	2.67	2.21
	30-40%	**3.19**	3.77	3.37	3.28	2.36	2.33	2.36	**2.09**	3.31	3.51	3.42	**3.27**
	40-50%	4.37	5.14	4.66	**4.28**	4.01	3.29	3.36	**2.99**	3.86	4.18	4.11	**4.07**
PSNR†	10-20%	27.71	27.95	28.87	28.48	**33.05**	32.69	31.23	32.53	35.06	**37.32**	33.19	35.21
	20-30%	24.54	**25.92**	25.66	25.84	29.10	**29.45**	29.13	29.08	33.95	**35.14**	32.83	34.68
	30-40%	22.01	**24.92**	23.46	24.68	27.24	27.01	26.98	**27.27**	32.63	32.90	32.12	**33.09**
	40-50%	20.34	21.16	19.29	**21.79**	23.46	24.98	24.34	**25.11**	31.95	31.94	31.28	**32.07**
SSIM†	10-20%	0.867	0.920	0.936	0.912	**0.978**	0.976	0.972	0.976	**0.911**	0.903	0.886	0.905
	20-30%	0.775	0.861	0.904	0.901	0.956	0.954	0.952	**0.961**	0.866	0.846	0.854	**0.869**
	30-40%	0.681	0.799	0.823	**0.830**	0.926	0.927	0.921	**0.932**	0.785	0.732	0.787	**0.790**
	40-50%	0.583	0.731	0.737	**0.742**	0.839	0.892	0.895	**0.909**	0.737	0.669	0.729	**0.745**
FID*	10-20%	**2.00**	2.55	3.66	2.04	47.05	34.99	44.52	**33.82**	8.00	17.60	8.07	**7.64**
	20-30%	4.02	5.07	5.89	**3.89**	48.54	36.69	46.11	**35.99**	10.68	23.14	9.67	**9.22**
	30-40%	6.65	6.32	7.24	**5.69**	58.49	51.12	55.32	**47.21**	22.66	54.29	17.27	**15.31**
	40-50%	11.45	7.01	10.17	**6.38**	64.31	66.58	63.90	**59.22**	27.64	64.08	24.36	**22.06**
IS†	10-20%	18.32	18.29	18.33	**18.37**	2.59	2.39	2.48	**2.62**	82.52	69.41	82.27	**84.07**
	20-30%	17.37	17.75	18.03	**18.19**	2.48	2.38	2.42	**2.55**	80.78	65.67	81.79	**81.88**
	30-40%	17.36	17.72	17.97	**18.02**	2.43	2.37	2.32	**2.50**	64.74	40.70	69.81	**70.03**
	40-50%	17.32	17.63	17.80	**17.90**	2.41	2.32	2.26	**2.43**	61.40	35.44	66.45	**67.89**
LPIPS*	10-20%	0.031	0.035	0.027	**0.025**	0.132	**0.084**	0.132	0.085	0.049	0.066	0.052	**0.047**
	20-30%	0.055	0.044	0.051	**0.031**	0.168	**0.106**	0.160	**0.106**	0.071	0.101	0.066	**0.062**
	30-40%	0.071	0.056	0.074	**0.043**	0.239	0.198	0.203	**0.163**	0.123	0.207	0.109	**0.099**
	40-50%	0.088	0.067	0.092	**0.057**	0.293	0.257	0.245	**0.221**	0.147	0.245	0.145	**0.136**

rectangular masks with various scales and aspect ratios. The three visual signals are: i) constant value, ii) Gaussian noise and iii) watermark logos [7].

6.2 Parameter Settings

To enable reliable prior-guidance for the SIN, we adopt the following training strategies: we first train the SDN and the LPN separately, and finally integrate the two pre-trained networks for the SIN optimization. Specifically, we train the SDN for 20 epochs with the learning rate of 10^{-3}. The Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ is adopted and the batch size is set to 16. For the LPN training, we use a mini-batch of 16 images for 100 epochs and an initial learning rate of 10^{-3}. The learning rate is multiplied by 0.1 every 40 epochs. The SGD optimizer with momentum of 0.9 and weight decay of 5×10^{-4} is adopted. As for the focusing parameter defined in Eq.(3), we set it to $\gamma = 2$. After pre-training the SDN and LPN, we use the two networks to generate estimated masks and layout maps, which are used as the input of the SIN. Then we adopt the SDN as the discriminator, which is fine-tuned for the SIN training. We adopt the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and with a learning rate of 10^{-4} for G (SIN) and D (SDN). The model is trained for 100 epochs with the batch size of 8.

Fig. 3. Qualitative comparison of different image inpainting models. The images from top to bottom are respectively from CelebA-HQ, PSV, ImageNet, Places2 and CityScapes datasets. From left to right, (a) input images, (b) ground truth images, the inpainted results of (c) CA$^\sharp$ [14], (d) PC$^\sharp$ [23], (e) EC$^\sharp$ [12], (f) PIC$^\sharp$ [33] and (g) SN$^\sharp$ [53], (h) PHD [54], (i) our generated layout maps and (j) our inpainted results ($\sharp$ means the non-blind methods with ground truth masks, and otherwise the blind method).

TABLE 2
Comparison of model complexity of different image inpainting models ($\sharp$ means the non-blind methods with ground truth masks, and otherwise the blind methods; $*$ means a lower value is better)

Model	Params* (MB)	Flops* (GB)	Inference time* (ms)
CA$^\sharp$ [14]	**3.60**	22.47	13.99
PC$^\sharp$ [23]	51.55	18.95	27.23
GC$^\sharp$ [24]	25.46	137.97	39.43
EC$^\sharp$ [12]	21.54	122.55	21.43
PIC$^\sharp$ [33]	6.03	**4.04**	29.28
SN$^\sharp$ [53]	54.41	18.16	9.25
WDNet [7]	20.07	69.99	15.22
PHD [54]	182.43	68.89	**7.56**
Ours	54.30	46.82	31.63

The hyper-parameters of the three loss functions defined in Eq.(11) are set to $\varepsilon = 0.05$ and $\mu = \rho = 1$.

6.3 Comparison to State-of-the-Art

Quantitative Evaluation. Table 1 lists the quantitative comparisons with state-of-the-art methods on five datasets with different mask ratios, which are number of pixels in the corrupted regions (*i.e.* with mask value 0) divided by the total number of pixels. For more comprehensive evaluations, we adopt six metrics to measure the inpainted results, including three image fidelity evaluation metrics, *i.e.*, ℓ_1 error, peak signal to noise ratio (PSNR), structure similarity index (SSIM), and three image perceptual quality evaluation metrics, *i.e.*, Frechet Inception Distance (FID) [55], Inception Score (IS) [56] and Learned Perceptual Image Patch Similarity (LPIPS) [57]. The compared methods include advanced non-blind image inpainting models, *i.e.*, CA [14], PC [23], GC [24], EC [12], PIC [33], SN [53], PRVS [28] and PDGAN [52], which all require known masks as input. In addition, we compare our method with PHD [54], which is a classical image translation model. Here we use it for blind image inpainting.

FID measures the similarity between two datasets of images. It is proven to correlate well with human judgement of visual quality and is often used to evaluate the quality of samples of GANs. FID is obtained by computing the Frechet distance between two Gaussians fitted to feature representations of the Inception network [58]. From Table 1, we can see that the FID scores of each model have large differences on different datasets. The differences are especially high for the PSV and CityScapes datasets. The reason may be that the data distribution of the two datasets is very different from the data distribution of the pre-trained dataset of the Inception network. IS also adopts the pre-trained Inception network, but it directly measures the distribution of the generated images instead of using the ground-truth images as references. It can be seen from Table 1 that the IS scores are particularly high on ImageNet. The reason may be that this dataset contains more semantic categories and diverse scenes. For the LPIPS, we employ the AlexNet [59] pretrained on BAPPS [57], a large-scale perceptual similarity dataset, which contains a large set of distortions and real algorithm outputs. The LPIPS is known to be highly effective in measuring the perceptual distance of two images.

Table 1 shows that our model achieves competitive performance with the non-blind image inpainting methods, which include ground truth binary masks as the input. Our model also obtains remarkably better scores than the blind

TABLE 3
Ablation studies on critical architecture components and loss functions of our model († means higher is better, while $*$ means lower is better)

Variant Model	Components						CelebA-HQ			CLWD		
	soft mask	layout	SPConv	$L_{\text{pix-adv}}$	$L_{\text{GL-adv}}$	L_{SS}	PSNR†	SSIM†	LPIPS*	PSNR†	SSIM†	LPIPS*
w/o mask		✓		✓	✓	✓	26.96	0.921	0.102	22.23	0.787	0.086
w/o layout	✓		✓	✓	✓	✓	27.81	0.933	0.091	24.01	0.845	0.057
w/o SPConv		✓		✓	✓	✓	28.92	0.939	0.069	24.75	0.852	0.052
w/o $L_{\text{pix-adv}}$	✓	✓	✓		✓	✓	28.21	0.923	0.089	25.08	0.869	0.047
w/o $L_{\text{GL-adv}}$	✓	✓	✓	✓		✓	28.75	0.930	0.081	25.26	0.871	0.044
w/o L_{SS}	✓	✓	✓	✓	✓		28.02	0.921	0.098	23.92	0.841	0.063
Full	✓	✓	✓	✓	✓	✓	**28.97**	**0.942**	**0.067**	**25.42**	**0.893**	**0.029**

Fig. 4. Qualitative comparison between the variant models and our full model. From left to right, (a) input images, (b) estimated masks, (c) estimated layout maps, (d) ground truth images, inpainted results of the variant models (e) w/o mask, (f) w/o layout map, (g) w/o SPConv, (h) w/o $L_{\text{pix-adv}}$, (i) w/o $L_{\text{GL-adv}}$ and (j) w/o L_{SS}, and inpainted results of (k) our full model. (Zoom in for more details)

method PHD. The performance of our model in terms of the fidelity metrics is not as prominent as the performance in terms of perceptual metrics. The fidelity metrics evaluate the image quality in a pixel-wise manner, while the perceptual metrics measure the visual quality of images, and they are more consistent with the human evaluation. The result indicates that the inpainted images obtained by our model have a better perceptual consistency with the ground truth images than the inpainted images obtained by the competing models. Moreover, as the mask ratio increases, our model degrades more slowly and even surpasses the competing models when the mask ratio is 30-50%. This indicates that our model is more robust under changes of the mask.

Qualitative Evaluation. In Fig. 3, we provide a visual comparison on five datasets. It can be seen that there exist prominent artefacts in the results of the models chosen for the comparison. For example, CA [14] and EC [12] fail to synthesize a plausible human face; PConv [23] and SN [53] suffer from blurring in the edges; PIC [33] fails to restore the partially masked person and generates unpleasant artefacts in the city street image. Originally, PHD [54] was not proposed for inpainting, but it surpasses many non-blind image inpainting methods in terms of visual quality. However, PHD suffers from blurry artefacts when the corrupted regions are large. In contrast, our model obtains better inpainted results with coherent structures and fine-grained details. Inpainting images with diverse semantics in datasets such as ImageNet and Places2 is considered challenging. Nevertheless, our model performs well on these datasets.

Model Complexity Evaluation. To evaluate the complexity of the inpainting models, parameter numbers (params), floating point operations (Flops) and inference time for an image with a resolution of 256×256 are computed. According to the results shown in Table 2, the inference time of our model is 31.63 ms. Our model requires more time than most of the competing models but not too much more, even though our model contains multiple stages to compute the soft mask and the layout map, while most of the other models require known masks. Meanwhile, our model is at an intermediate level with regard to parameters and Flops. It is comparable with and even has lower complexity than many non-blind models. More importantly, our model surpasses most of the competing models in terms of the quantitative and qualitative performance, as shown in Table 1 and Fig. 3. Therefore, our model achieves a good trade-off between the inpainting performance and the efficiency.

6.4 Ablation Study

In order to validate the contributions that key components make to our proposed model, we train a series of variant

TABLE 4
Quantitative comparison of watermark detection accuracy in terms of F1 score, AUC and MAE and removal performance in terms of PSNR and SSIM on CLWD dataset († means higher is better, while $*$ means lower is better)

Model	WDNet [7]			Ours		
ϕ	0.3	0.5	0.7	0.3	0.5	0.7
F1†	0.841	0.856	0.835	**0.909**	**0.912**	**0.873**
AUC†	0.970	0.971	0.933	**0.974**	**0.987**	**0.949**
MAE*	0.040	0.035	0.047	**0.023**	**0.015**	**0.034**
PSNR†	33.11	35.09	29.47	**34.88**	**35.74**	**31.66**
SSIM†	0.959	0.969	0.951	**0.976**	**0.983**	**0.959**

Fig. 5. Qualitative comparison of watermark detection and removal performance. From left to right, (a) watermarked images (with intensity of 0.3, 0.5 and 0.7 from top to bottom), (b) estimated masks of WDNet, (c) watermark removal results of WDNet, (d) estimated masks of our model and (e) watermark removal results of our model, respectively.

models: i) *w/o (without) mask*, where the soft mask is removed from the input of the SIN, and correspondingly, the proposed SPConv cannot be applied. We replace SPConv by vanilla convolution; ii) *w/o layout*, where the layout map is removed from the input of SIN, and correspondingly, the SSEU branch is removed from the SRB; iii) *w/o SPConv*, where the SPConv is replaced by the partial convolution (PConv) [23] in the SIN, and the soft mask is binarized using a threshold of 0.5. We also investigate how the loss functions affect the performance. To this end, we train another three variant models: *w/o* $L_{\text{pix-adv}}$, *w/o* $L_{\text{GL-adv}}$ and *w/o* L_{SS}, which remove the pixel adversarial loss, global adversarial loss and semantic structure loss, respectively from the overall loss function of the SIN. We train the above variant models on CelebA-HQ and CLWD datasets. The CLWD dataset contains 60,000 watermarked images, which are made by adding logos with different transparencies in the range (0.3, 0.7) to clean images. Each generated watermark image has a corresponding binary mask indicating the location of the logo. All the parameter settings are kept the same as those for our full model, as explained in Sec. 6.2.

Quantitative and qualitative comparisons between the variant models and our full model are demonstrated in Table 3 and Fig. 4, respectively. According to the results, all the variant models are inferior to our full model. The performance of the *w/o mask* decreases the most since there is no mask for indicating the corrupted regions. As Fig. 4 (e) shows, the inpainting is visually unrealistic. The *w/o layout* also performs badly. Fig. 4 (f) shows that the inpainted image regions are blurred. For the *w/o SPConv*, the scores achieved on the CelebA-HQ dataset compare well with the scores obtained by the full model. This is because the corruptions are simple, and our SDN accurately detects them with high scores. In this case, there is almost no difference between the SPConv and PConv. However, when the PConv is employed on the CLWD dataset, its performance shows a clear drop because i) semi-transparent watermarks increase the difficulty of the SDN, ii) the estimation error produced by binarization is propagated to the inpainting process, and iii) the PConv ignores the latent image information under the semi-transparent watermarks. As we can see in Fig. 4 (g), the PConv generates consistent results with our full model on CelebA-HQ dataset, but fails to remove all of the watermarks from the bottom two images. For another three variant models, *i.e.*, *w/o* $L_{\text{pix-adv}}$, *w/o* $L_{\text{GL-adv}}$ and *w/o* L_{SS}, removing $L_{\text{pix-adv}}$ or L_{SS} has a larger impact on the metric scores than removing $L_{\text{GL-adv}}$. This indicates that the proposed two loss functions contribute more to our model.

6.5 Discussion

Analysis on Semi-Transparent Corruption Detection. Our SDN can accurately detect the corrupted regions filled with the constant values or random noise. However, the accuracy is reduced when these regions are corrupted by semi-transparent signals, it largely increases the detection difficulty since image patterns are mixed in it. To investigate how the transparency of the corruption signal impacts the detection accuracy of the SDN, we apply the logos of the CLWD dataset to generate watermarked images by setting the logo transparency in the range of [0.1, 0.9]. Recall that in Eq.(1), ϕ denotes the intensity of the corruption signal. Here, we use $1 - \phi$ to denote the transparency of watermarks. We evaluate the performance of the SDN using three metrics, namely F1 score, area under curve (AUC) and mean absolute error (MAE). All three metrics are widely used for saliency detection.

Detection accuracy with varied corruption intensity ϕ in terms of F1 score, AUC and MAE, and the corresponding inpainting performance in terms of PSNR and SSIM are listed in Table 4. We use a state-of-the-art watermark removal method, *i.e.*, WDNet [7], for comparison. The WDNet

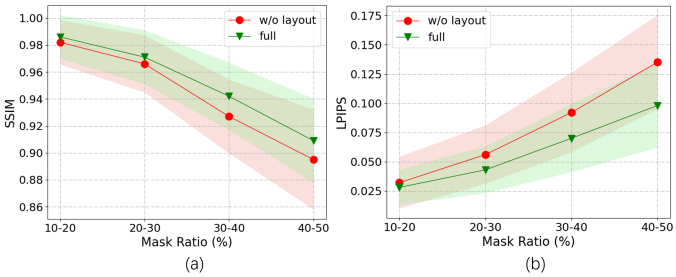


Fig. 6. Quantitative comparison between the variant *w/o layout* and our full model with different mask ratios. The shadow denotes the standard deviation.

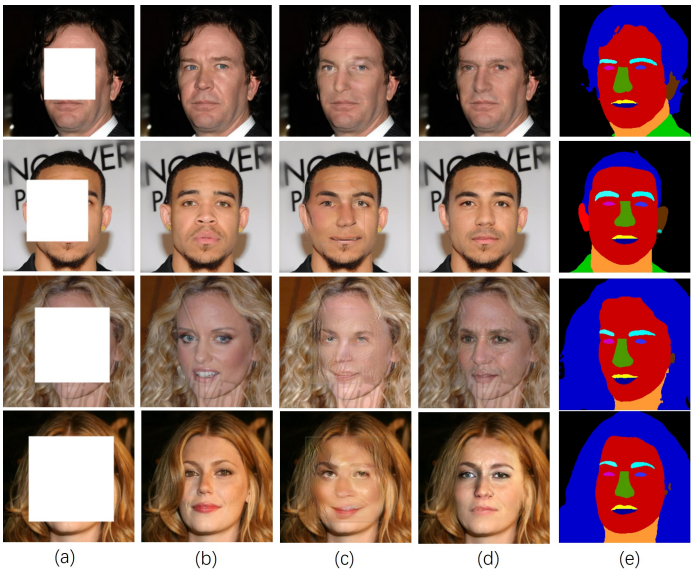


Fig. 7. Qualitative comparison between the inpainted results of the variant model *w/o layout* and our full model. From top to bottom, the images have mask ratios of 10-20%, 20-30%, 30-40% and 40-50%. From left to right, (a) input images, (b) ground truth images, (c) inpainted images of *w/o layout*, (d) inpainted images of our full model and (e) generated layout maps by our LPN.

uses a two-stage process. The first stage generates a coarse removal result, a potential watermark and its location. The second stage centers on the watermarked area to refine the result. According to Table 4, our SDN achieves a better watermark detection accuracy than the WDNet across all the three values of ϕ. In addition, we observe that the decrease of detection accuracy has a negative impact on the watermark removal. We show more examples in Fig. 5. From the comparison, we can see that our model not only generates more accurate masks, but also is more robust to the varying watermark intensities. Moreover, the mask estimation error has a significant influence on the results of the WDNet. Benefit from the proposed SPConv, our model is more resistant to the mask estimation error.

Analysis on the Impact of Layout Map. One of the important roles the layout map plays is to relieve the difficulty in inpainting the images with large corrupted regions by guiding the synthesis of semantically consistent textures. To investigate how much we could exploit the layout map to improve the inpainting, we compare the performance of the variant model *w/o layout* and our full model with different mask ratios. The metrics in terms of SSIM and LPIPS in

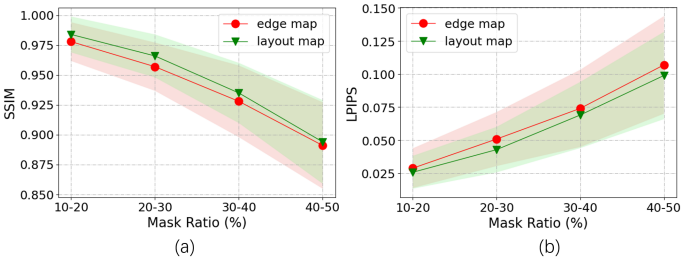


Fig. 8. Quantitative comparison between using the edge map and the layout map as extra inputs to the EC [12] with different mask ratios. The shadow denotes the standard deviation.

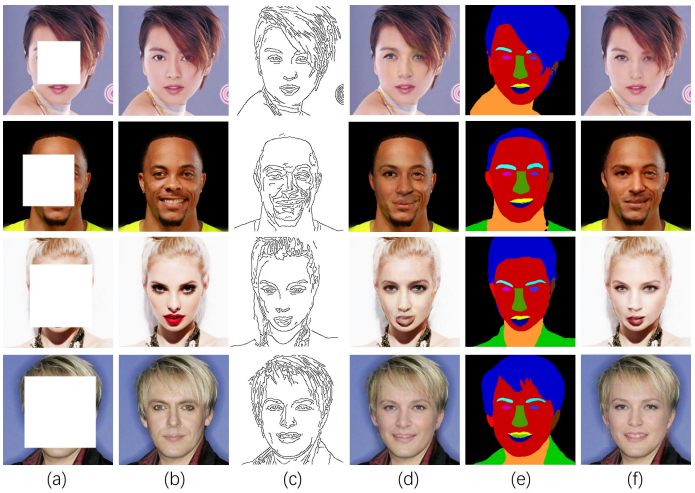


Fig. 9. Qualitative comparison between the inpainted results using the edge map and the layout map as extra inputs to the EC [12]. From top to bottom, the images have mask ratios of 10-20%, 20-30%, 30-40% and 40-50%. From left to right, (a) input images (b) ground truth images (c) edge maps (d) inpainted results using the edge map as the input (e) layout maps and (f) inpainted results using the layout map as the input.

Fig. 6 show that the performance of the variant *w/o layout* is always inferior to the full model across all the mask ratios, *i.e.*, with a lower SSIM mean value, a higher LPIPS mean value and higher standard deviations. From their qualitative comparison, shown in Fig. 7, we observe that when 10-20% regions are corrupted, both models show visually plausible results. When 20-30% regions are corrupted, the inpainted image of the variant *w/o layout* shows some unrealistic details, such as the noise in the second row in column (c). If the mask ratios are increased, the quality of the inpainting degrades more seriously, while our full model still produces reasonable images. From the generated layout maps shown in column (e), we can see that the predicted layout map is more robust to the increase of the mask ratio than directly predicting the inpainted image. Given this self-prior, our model can complete the image inpainting more easily.

Furthermore, we also substitute the edge map input of the EC [12] for the layout map predicted by our LPN. From Fig. 8 we observe that substituting the edge map for the layout map improves the metrics. From Fig. 9 we see that as the mask ratio is increased, both models generate acceptable inpainted images, as they benefit from the extra input information. However, it is clear that the results using the layout map are more plausible, while the results using the edge map are less sharp, for example as shown by the

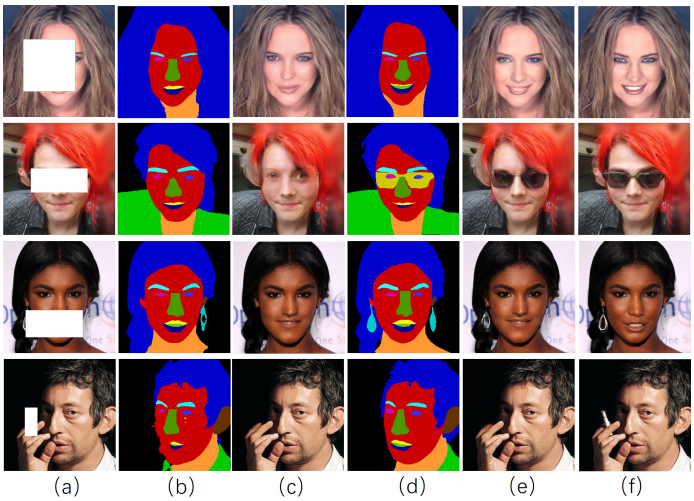


Fig. 10. Qualitative comparison of inpainted results using the generated and ground truth layout map. From left to right, (a) input images, (b) generated layout maps by our model and (c) inpainted images conditioned on them, (d) ground truth layout maps and (e) inpainted images conditioned on them and (f) ground truth images.

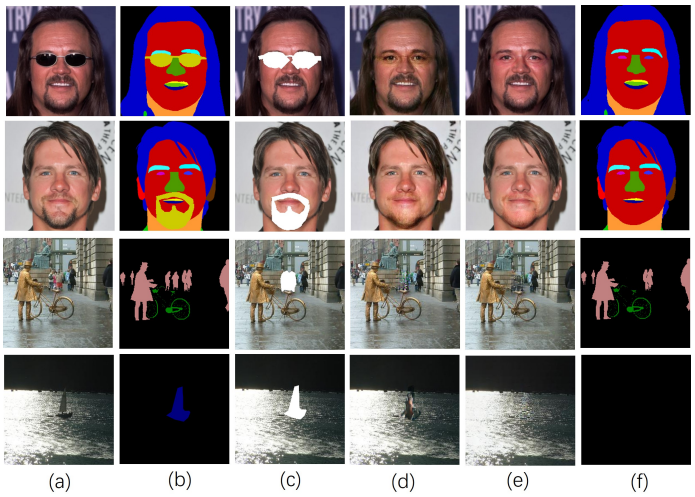


Fig. 11. Examples of object removal results on CelebA-HQ (top two rows) and CLWD (bottom two rows). From left to right, (a) original images, (b) layout maps for the original image, (c) masked images, (d) inpainted results of PHD [54], (e) inpainted results of our model and (f) layout maps for the masked image.

eyes and mouth. The reason may be that the edge map only describes the outlines while the layout map provides more semantic context information, which makes it possible to inpaint large corrupted regions.

Here, we further investigate the impact of the layout map quality on the inpainting performance. Specifically, we substitute the ground truth layout map for the layout map generated by our LPN. The quantitative results show that the PSNR and SSIM of inpainted images increase by 0.74 and 0.031, respectively. From the qualitative results shown in Fig. 10, we can see that the ground truth layout map improves inpainting by restoring correct expressions (*e.g.*, smile) and recovering lost attributes (*e.g.*, sunglasses and earrings). We note that when face attributes or objects are not included in the predefined categories, our method is unable to recover them. See for example the cigarette in the bottom right side of Fig. 10. In the future work, our model will be trained on more diverse-category datasets to make it more practical.

6.6 Extension Verification

In this section, we extended our method to three related tasks, namely object removal, image super-resolution and old photo restoration, to prove the advantages of our method in generating semantically consistent and visually pleasing images.

6.6.1 Extension to Object Removal

The layout map can be used for object removal. Image inpainting models are employed to fill the holes left when undesired objects are removed. Since the undesired objects are not corrupted by any visual signal (*e.g.*, watermarks, constant values or random noise), it is infeasible for our SDN to detect them. The layout map contains explicit boundaries for each object. It is thus straightforward to select the objects to be removed. Specifically, our LPN generates a layout map for an original image. Then users indicate the undesired object on the layout map. Next, the undesired object is masked. The LPN is then employed again to generate a complete layout map for the masked image. Finally, the masked region is inpainted under the guidance of the complete layout map. In this way, the users can efficiently label undesirable objects and avoid costly pixel-wise annotation.

TABLE 5
Quantitative evaluation on image super-resolution performance

Metrics	CelebA-HQ			Places2		
	SRGAN	PHD	Ours	SRGAN	PHD	Ours
$\ell_1(\%)^*$	2.257	3.735	**2.086**	4.331	4.852	**3.852**
PSNR†	29.12	25.97	**32.02**	27.26	24.55	**28.71**
SSIM†	0.949	0.857	**0.952**	0.863	0.835	**0.891**

Qualitative evaluation of object removal on CelebA-HQ and CLWD datasets is illustrated in Fig. 11, where PHD [54] is employed for comparison. For face images, we randomly remove a face attribute from the 18 attributes used in the CelebA-HQ dataset. For natural images from the CLWD dataset, we randomly remove an object in any of the 20 categories defined by the SBD dataset. Fig. 11 shows that our model produces cleaner face images, while traces of the sunglasses and beard are obvious in the results of PHD. In addition, our model also produces perceptually more plausible and pleasing images than those produced by PHD.

6.6.2 Extension to Image Super-resolution

We apply our method to image super-resolution. Specifically, given a low-resolution image $I_{\text{low}} \in \mathbb{R}^{H\times W}$ and an up-sampling factor s, we construct the input image $I'_{\text{low}} \in \mathbb{R}^{(sH)\times(sW)}$ as follows:

$$I'_{\text{low}}(su, sv) = \begin{cases} I_{\text{low}}(u, v), & 1 \le u \le H, 1 \le v \le W \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

At the same time, we construct a mask $M \in \mathbb{R}^{(sH)\times(sW)}$ by labelling $M(su, sv) = 1$ $(1 \le u \le H, 1 \le v \le W)$,

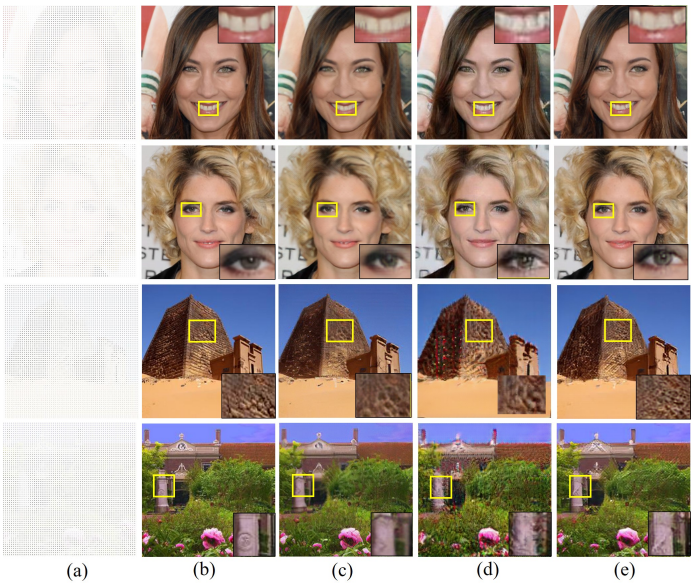


Fig. 12. Examples of image super-resolution results on CelebA-HQ (top two rows) and Places2 (bottom two rows). From left to right, (a) input images, (b) ground truth images, super-resolution results of (c) SRGAN [60], (d) PHD [54] and (e) our model.

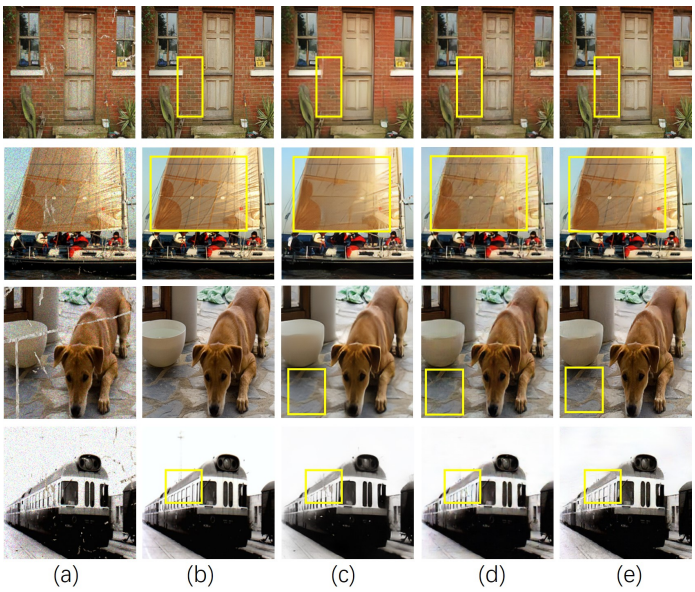


Fig. 13. Results obtained by restoring synthesized old photos on Pascal VOC. From left to right, (a) input images, (b) ground truth images, restored results of (c) the method in [3], (d) PHD [54] and (e)our model.

and otherwise 0. Here, we set $s = 4$ and $H = W = 64$. The image I'_{low} is input to LPN to obtain a layout map $\hat{S}$. Then, we input $\hat{S}$, M and I'_{low} to the SIN. The output image is the required super-resolution image. Note that since the ground truth masks are the same for all the input images, we cannot train the SDN effectively. For this reason, the pixel adversarial loss $L_{\text{pix-adv}}$ is removed from the overall loss functions. Other configurations are kept the same as in the blind inpainting task, as explained in Section 6.2. Our model is retrained on CelebA-HQ and Places2 datasets. In addition, we retrain PHD [54] and a well-known image super-resolution model called SRGAN [60] for comparison.

Quantitative evaluations in Table 5 show that our model achieves better metric scores, especially for the SSIM. Meanwhile, the qualitative evaluations in Fig. 12 demonstrate that the competing models suffer from blurring and noise artefacts, while our model generates sharp edges and fine details, as illustrated in the cropped regions.

Fig. 14. Results obtained by restoring real-world old photos. From left to right, (a) input images, restored results of (b) the method in [3], (c) PHD [54] and (d) our model.

6.6.3 *Extension to Old Photo Restoration*

We further extend our method to old photo restoration where some defects, such as scratches and blemishes, need to be removed and completed with reasonable contents using the inpainting techniques. Following the experimental setting in [3], we synthesize old photos using images from the Pascal VOC dataset, and collect scratches and paper textures to render realistic defects. In addition, random blurring and random noise are introduced to simulate the unstructured defects. We use the SDN to automatically detect the possible scratches, and then we adopt the LPN trained on the SBD dataset to generate the layout map. Finally the SIN is employed to restore the artefacts and output a clean photo. For comparison, we train PHD [54] on the same dataset. We also compare with the state-of-the-art old photo restoration model [3]. Their results are obtained by running their model[4] pre-trained on the Pascal VOC dataset and their collection of old photos.

Qualitative evaluations on synthesized old photos using the Pascal VOC dataset and real-world old photos collected from the Internet are shown in Fig. 13 and Fig. 14, respectively. From Fig. 13, we find that the results of the two competing models often suffer from blurring artefacts,

4. The pre-trained model is downloaded from https://github.com/microsoft/Bringing-Old-Photos-Back-to-Life

and sometimes the scratches are not removed. In contrast, our model generates more complicated structures and richer details. From Fig. 14, we observe that both the model in [3] and our model are better than PHD at dealing with the realistic blemishes and blurring in real old photos. Note that the model in [3] collects the real old photos for training while our model only adopts the synthesized old photos. This validates the generalization of our model from synthesized data distribution to real data distribution.

7 Conclusion

In this paper we propose a novel blind image inpainting model, which integrates two learnt priors to guide and optimize the inpainting process. To learn a "where to inpaint" prior, we construct a SDN to estimate for each pixel the probability that it has a valid value. To learn a "how to inpaint" prior, we introduce a LPN to predict the global semantic structure in a pixel-wise classification. Furthermore, we design *SPConv* and *SSEU* to embed the learnt priors into our inpainting network SIN. More importantly, the SDN and LPN are re-employed to optimize our SIN by pixel adversarial training and semantic structure similarity matching. The three networks form a united framework. The experimental results show that our method surpasses many state-of-the-art non-blind image inpainting methods in terms of both objective metrics and visual quality. In addition, ablation studies validate the effectiveness of key components of our model. Furthermore, extensions to other related tasks show the advantages of our method in promoting the semantic consistency and visual quality of generated images. In the future work, we intend to include more categories in our layout map, so that we can embed more diverse semantics into our SIN to boost the performance for images with more complicated semantic structures.

References

[1] N. Cai, Z. Su, Z. Lin, H. Wang, Z. Yang and B. Ling, "Blind inpainting using the fully convolutional neural network," *Vis. Comput.*, vol. 33, no. 2, pp. 249–261, 2017.

[2] Y. Liu, J. Pan and Z. Su, "Deep blind image inpainting," in *Proc. Conf. Intell. Sci. Big Data Engi.* Springer, 2019, pp. 128–141.

[3] D. Chen P. Zhang D. Chen J. Liao Z. Wan, B. Zhang and F. Wen, "Bringing old photos back to life," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2744–2754.

[4] Z. Wan, B. Zhang, D. Chen, P. Zhang, D. Chen, J. Liao and F. Wen, "Old Photo Restoration via Deep Latent Space Translation," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022.

[5] A. Hertz, S. Fogel, R. Hanocka, R. Giryes and D. Cohen-Or, "Blind visual motif removal from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6858–6867.

[6] Y. Wang, Y. Chen, X. Tao and J. Jia, "VCNet: A Robust Approach to Blind Image Inpainting," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 752–768.

[7] Y. Liu, Z. Zhu and X. Bai, "WDNet: Watermark-Decomposition Network for Visible Watermark Removal," in *Proc. Conf. Appl. Comput. Vis.*, 2021, pp. 3685–3693.

[8] S. Darabi, E. Shechtman, C. Barnes, D. Goldman and P. Sen, "Image melding: Combining inconsistent images using patch-based synthesis," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 1–10, 2012.

[9] Y. Wexler, E. Shechtman and M. Irani, "Space-time completion of video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 3, pp. 463–476, 2007.

[10] S. Esedoglu and J. Shen, "Digital inpainting based on the Mumford–Shah–Euler image model," *Eur. J. Appl. Math.*, vol. 13, no. 4, pp. 353–370, 2002.

[11] D. Liu, X. Sun, F. Wu, S. Li and Y. Zhang, "Image compression with edge-based inpainting," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 10, pp. 1273–1287, 2007.

[12] K. Nazeri, E. Ng, T. Joseph, F. Qureshi and M. Ebrahimi, "Edgeconnect: Generative image inpainting with adversarial edge learning," *arXiv preprint arXiv:1901.00212*, 2019.

[13] W. Xiong, J. Yu, Z. Lin, J. Yang, X. Lu, C. Barnes and J. Luo, "Foreground-aware image inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5840–5848.

[14] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu and T. Huang, "Generative image inpainting with contextual attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5505–5514.

[15] J. Li, F. He, L. Zhang, B. Du and D. Tao, "Progressive reconstruction of visual structure for image inpainting," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 5962–5971.

[16] Y. Ma, X. Liu, S. Bai, L. Wang, D. He and A. Liu, "Coarse-to-Fine Image Inpainting via Region-wise Convolutions and Non-Local Correlation.," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 3123–3129.

[17] J. Wang, Y. Duan, X. Tao, M. Xu and J. Lu, "Semantic perceptual image compression with a laplacian pyramid of convolutional networks," *IEEE Trans. Image Process.*, vol. 30, pp. 4225–4237, 2021.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L.Kaiser, and I. Polosukhin, "Attention is All You Need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[19] H. Liu, Y. Wang, M. Wang, and Y. Rui, "Delving Globally into Texture and Structure for Image Inpainting," in *Proc. ACM Int. Conf. on Multimedia*, 2022, pp. 1270–1278.

[20] W. Li, Z. Lin, K. Zhou, L. Qi, Y. Wang, and J. Jia, "MAT: Mask-Aware Transformer for Large Hole Image Inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10748–10758.

[21] Q. Dong, C. Cao, and Y. Fu, "Incremental Transformer Structure Enhanced Image Inpainting with Masking Positional Encoding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11348–11358.

[22] R. Hanocka, G. Metzer, R. Giryes and D. Cohen-Or, "Point2Mesh: A Self-Prior for Deformable Meshes," *ACM Trans. Graph.*, vol. 39, no. 4, 2020.

[23] G. Liu, F. Reda, K. Shih, T. Wang, A. Tao and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 85–100.

[24] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu and T. Huang, "Free-form image inpainting with gated convolution," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 4471–4480.

[25] S. Oh, S. Lee, J. Lee and S. Kim, "Onion-peel networks for deep video completion," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 4403–4412.

[26] H. Zhang, Z. Hu, C. Luo, W. Zuo and M. Wang, "Semantic image inpainting with progressive generative networks," in *Proc. ACM Int. Conf. on Multimedia*, 2018, pp. 1939–1947.

[27] Y. Zeng, Z. Lin, J. Yang, J. Zhang, E. Shechtman and H. Lu, "High-resolution image inpainting with iterative confidence feedback and guided upsampling," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 1–17.

[28] J. Li, F. He, F. Zhang, B. Du and D. Tao, "Progressive Reconstruction of Visual Structure for Image Inpainting," in *Proc. IEEE Int. Conf. Comput. Vis.*, October 2019.

[29] Y. Song, C. Yang, Y. Shen, P. Wang, Q. Huang and C. Kuo, "Spg-net: Segmentation prediction and guidance network for image inpainting," *arXiv preprint arXiv:1805.03356*, 2018.

[30] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell and A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2536–2544.

[31] S. Iizuka, E. Simo-Serra and H. Ishikawa, "Globally and locally consistent image completion," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–14, 2017.

[32] D. Liu, B. Wen, Y. Fan, C. Loy and T. Huang, "Non-local recurrent network for image restoration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1673–1682.

[33] C. Zheng, T. Cham and J. Cai, "Pluralistic image completion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1438–1447.

[34] Y. Wang, X. Tao, X. Qi, X. Shen and J. Jia, "Image Inpainting via Generative Multi-Column Convolutional Neural Networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 329–338.

[35] Y. Zeng, J. Fu, H. Chao and B. Guo, "Learning Pyramid-Context Encoder Network for High-Quality Image Inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1486–1494.

[36] P. Isola, J. Zhu, T. Zhou and A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1125–1134.

[37] J. Xie, L. Xu and E. Chen, "Image denoising and inpainting with deep neural networks," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, pp. 341–349, 2012.

[38] D. Kim, S. Woo, J. Lee, and I. Kweon, "Recurrent temporal aggregation framework for deep video inpainting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 5, pp. 1038–1052, 2019.

[39] Z. Wan, B. Zhang, D. Chen, and J. Liao, "Bringing Old Films Back to Life," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 17673–17682.

[40] O. Ronneberger, P. Fischer and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assisted Intervention*. Springer, 2015, pp. 234–241.

[41] L. Chen, Y. Zhu, G. Papandreou, F. Schroff and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 801–818.

[42] T. Lin, P. Goyal,R. Girshick, K. He, Kaiming and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.

[43] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1501–1510.

[44] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[45] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *Proc. Int. Conf. Learn. Represent.*, May 2015.

[46] J. Deng, W. Dong, R. Socher, L. Li, and K. Li and F. Li, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[47] Z. Liu, P. Luo, X. Wang and X. Tang, "Deep Learning Face Attributes in the Wild," in *Proc. IEEE Int. Conf. Comput. Vis.*, December 2015.

[48] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.

[49] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva and A. Torralba, "Places: A 10 million Image Database for Scene Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017.

[50] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes Challenge: A Retrospective," *Int. J. Comput. Vision*, vol. 111, no. 1, pp. 98–136, 2015.

[51] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji and J. Malik, "Semantic contours from inverse detectors," in *Proc. IEEE Int. Conf. Comput. Vis.* IEEE, 2011, pp. 991–998.

[52] H. Liu, Z. Wan, W. Huang, Y. Song, X. Han and J. Liao, "Pd-gan: Probabilistic diverse gan for image inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9371–9381.

[53] Z. Yan, X. Li, M. Li, W. Zuo and S. Shan, "Shift-net: Image inpainting via deep feature rearrangement," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 1–17.

[54] T. Wang, M. Liu, J. Zhu, A. Tao, J. Kautz and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8798–8807.

[55] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[56] T. Salimans, I. Goodfellow, W. Zaremba,V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016.

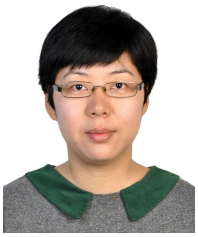
[57] R. Zhang, P. Isola, A. Efros, A. Alexei, E. Shechtman and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 586–595.

[58] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.

[59] A. Krizhevsky, I. Sutskever and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, no. 2, 2012.

[60] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4681–4690.

Juan Wang received the Ph.D. degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2020. She currently works as a research assistant with National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences(CASIA), Beijing. Her research interests includes deep learning, image restoration, and image generation.



Chunfeng Yuan received the Ph.D. degree from National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences(CASIA), Beijing, China, in 2010. She is currently an associate professor with CASIA. Her research interests and publications range from pattern recognition to computer vision, including sparse representation, deep learning, video understanding, and multimedia analysis.

Bing Li received the Ph.D. degree from the Department of Computer Science and Engineering, Beijing Jiaotong University, Beijing, China, in 2009. From 2009 to 2011, he worked as a postdoctoral research fellow with National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences(CASIA), Beijing. He is currently a professor with CASIA. His current research interests include computer vision, color constancy, visual saliency detection, multi-instance learning, and data mining.

Ying Deng is pursing for a doctorate in the College of Aerospace Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China. He is currently working in School of Aeronautical Manufacturing Engineering, Nanchang Hangkong University, Nanchang, China. His research interests include deep learning, image restoration, and video understanding.

Weiming Hu received the Ph.D. degree from the Department of Computer Science and Engineering, Zhejiang University, Zhejiang, China, in 1998. From 1998 to 2000, he was a postdoctoral research fellow with the Institute of Computer Science and Technology, Peking University, Beijing. He is currently a professor with the Institute of Automation, Chinese Academy of Sciences, Beijing. His research interests are visual motion analysis, recognition of web objectionable information, and network intrusion detection.

Stephen Maybank received a BA in Mathematics from King's College Cambridge in 1976 and a PhD in computer science from Birkbeck college, University of London in 1988. Now he is a professor emeritus in the Department of Computer Science and Information Systems, Birkbeck College. His research interests include the geometry of multiple images, camera calibration, visual surveillance etc.