Journal of Computers Vol. 34 No. 3, June 2023, pp. 237-250
doi: 10.53106/199115992023063403017

An End-to-End Multi-Scale Conditional Generative Adversarial Network for Image Deblurring

Fei Qi*, Chen-Qing Wang

College of Information Engineering, Longdong University, Qingyang 745000, Gansu, China
54515573@qq.com, 2434389000@qq.com

Received 1 July 2022; Revised 28 November 2022; Accepted 4 January 2023

Abstract. For image deblurring, multi-scale approaches have been widely used as deep learning methods recently. In this paper, a novel multi-scale conditional generative adversarial network (CGAN) is proposed to make full use of image features, which outperforms most state-of-the-art methods. We define a generator network and a discriminator network. First of all, we use the multi-scale residual modules proposed in this paper as main feature extraction blocks, and add skip connections to extract multi-scale image features at a finer granularity in the generator network. Secondly, we construct PatchGAN as the discriminator network to enhance the local feature extraction capability. In addition, we combine the adversarial loss based on Wasserstein GAN with gradient penalty (WGAN-GP) theory with the content loss defined by perceptual loss as the total loss function, which is conducive to improving the consistency between the generated images and the ground-truth sharp images in content. The experimental results show that the method in this paper outperforms the state-of-the-art methods in visualization and quantitative results.

Keywords: conditional generative adversarial network, image deblurring, multi-scale, end-to-end

1 Introduction

Image deblurring has become an important research topic for image processing and computer vision. During the camera shooting process, due to factors such as camera shake or rotation, target object movement or being out of focus, the images recorded may be blurred, and important details as well as edge structures will be lost or damaged, which will seriously affect the image quality. The goals of image deblurring are to restore the latent sharp images from corresponding blurred images, and recover the lost information.

Aiming at the problem of single image deblurring, explicit edge prediction [1-2] and statistical priors of natural images [3-9] are applied to construct deblurring models through traditional methods, and then the latent sharp images are recovered through the operation of deconvolution. However, the process of constructing a model requires time-consuming calculations. More importantly, because the constructed deblurring model is inaccurate, which still cannot be used to accurately approximate the actual or non-uniform blurs in complex scenes, the quality of the restored images will be directly reduced, resulting in undesired artifacts and ringing. With the development of deep learning, many researchers have tackled the image deblurring tasks through neural networks. There are two main different deep learning methods for image deblurring. The earlier method involves the estimation of non-uniform blur kernels [10-13] through neural networks, and then sharp images are restored relying on traditional deconvolution. The more recent method involves a direct restoration of latent sharp images from the input blurry images [14-17], without the process of blur kernel estimation. The end-to-end image deblurring methods have proved more effective, among which, through convolutional neural networks (CNNs), a significant success has been achieved in image deblurring with its inherent multi-scale feature extraction capability. Nah et al. [14] proposed a multi-scale CNN with a pyramid structure to restore end-to-end sharp images, which achieved excellent deblurring results. Tao et al. [15] proposed a scale-recurrent network (SRN) across multiple scales with a pyramid structure, and progressively restored the sharp images. However, it should be pointed out that the pyramid CNN structure is time-consuming and memory-demanding [18]. Generative adversarial networks (GANs) have also been applied to image deblurring due to their efficiency in preserving texture details of images. Kupyn et al. [18-19] designed the DeblurGAN and DeblurGAN-v2 model with multi-scale feature extraction, achieving state-of-the-art deblurring performance.

Aiming at the problems of poor performance and slow processing speed in the existing deep learning de-

* Corresponding Author

blurring methods, This paper presents an end-to-end multi-scale CGAN to efficiently deblur in complex scenes. First of all, a multi-scale residual block, the multi-scales Res2Net [20] block (MSR2B), is proposed as the main building model for a generator CNN architecture. Instead of extracting features at different scales using multiple hierarchical CNN layers, we construct the MSR2B model with hierarchical connections and multi-scale feature concatenation in a single residual block, and MSR2Bs are stacked to capture features at different scales while increasing the range of receiving fields at each network layer. Secondly, the skip connection is added to the generator CNN network, which not only avoids gradient problems from vanishing or exploding in the training process, but the local multi-scale features of each MSR2B output are also integrated to maximize the use of image features. For the discriminator architecture, we use PatchGAN [21] to obtain realistic texture details through capturing Markovian patches. We combine the adversarial loss and content loss as the total loss function, improving the sensory consistency between the generated images and the ground-truth sharp images.

The contributions of this paper are summarized as follows. In section 3.1, we constructed a multi-scale residual module named MSR2B in the Generator to enhance the multi-scale feature extraction capability and reduce the amount of parameters. In section 3.2, we designed a CGAN structure with the global and local skipping connections to improve the learning efficiency and the adaptive-expression capability. Meanwhile, the PatchGAN was introduced in the Discriminator to enhance the ability of local image feature extraction and representation. In section 3.3, we proposed a joint loss-function to maintain the stability of training process and better restore the details of the blurred images. In section 4.3, we verified the effectiveness of the MSR2B proposed in this paper. In section 4.4, the experimental results based on the GOPRO and Kohler dataset showed that our method achieves state-of-the-art performance among recent deep-learning image deblurring methods.

2 Related Work

2.1 Image Deblurring

The blurred image B observed can be modeled based on the latent sharp image S as:

$$B = K * S + N , \tag{1}$$

where * denotes the convolution operator, *K* denotes blur kernel and *N* denotes additive noise. Depending on whether the blur kernel *K* is known or unknown, image deblurring can be either non-blind or blind. Through non-blind image deblurring methods, it is assumed that the blur kernel *K* is known, and the classical Wiener filter or the Richardson-Lucy algorithm is used to perform the deconvolution operation. In fact, the blur kernel *K* cannot be obtained in advance in most real-world situations. According to (1), the blind deblurring problem has shortcomings because there are many different solution sets (*K*, *S*) leading to the same result *B*. In early works, natural image statistical priors and estimated salient edges were mainly used for blur kernels [1-8, 22-24]. Fergus et al. [22] used the variational Bayes method to obtain the maximum edge probability of an image. Cho and Lee et al. [23] used edge gradient threshold to estimate edges of images. Natural image priors are applied to predict blurs, such as sparse gradient priors [6], l_0 -norm priors [5], low-rank priors [7], and dark channel priors [8].

In recent years, deep learning methods have been applied to the field of image deblurring. Sun et al. [10] used a CNN to predict the probabilistic distribution of motion blurs, who combined the patch-level image priors and with a non-uniform deblurring model to eliminate complex non-uniform motion blurs. Gong et al. [11] used a fully-convolutional network (FCN) to estimate the motion trajectories of blurred images, who restored the sharp images through deconvolution. Chakrabarti et al. [12] applied the complex Fourier coefficients predicted using a CNN to restore sharp images in a Fourier space. However, blur kernels first need to be predicted through CNNs for these methods, and then traditional non-blind deblurring algorithms are used to deconvolve blurred images as well as processes that are complex and time-consuming. In more recent works, end-to-end kernel-free trainable networks began to be emphasized for image deblurring. Nah et al. [14] proposed a multi-scale CNN to restore sharp images directly from the input deblurred images, avoiding the process of blur kernel estimation. Tao et al. [15] and Noroozi et al. [16] designed multi-scale CNNs with a pyramid structure through a coarse-to-fine scheme to restore sharp images. The number of parameters and calculations were decreased in all of these works. Zhang et al. [17] combined three CNNs with a recurrent neural network (RNN) to enlarge the receptive field for single image deblurring. Noting the excellent performance of GANs in the related fields including image inpainting and image-to-image translation, Ramakrishnan et al. [25] combined a dense convolutional network [26] with the pix2pix architecture [27] of a GAN to restore sharp images. Kupyn et al. proposed DeblurGAN [19] with the multi-component loss function and DeblurGAN-v2 [18] with a feature pyramid network (FPN) for blind image deblurring.

2.2 Conditional Generative Adversarial Network

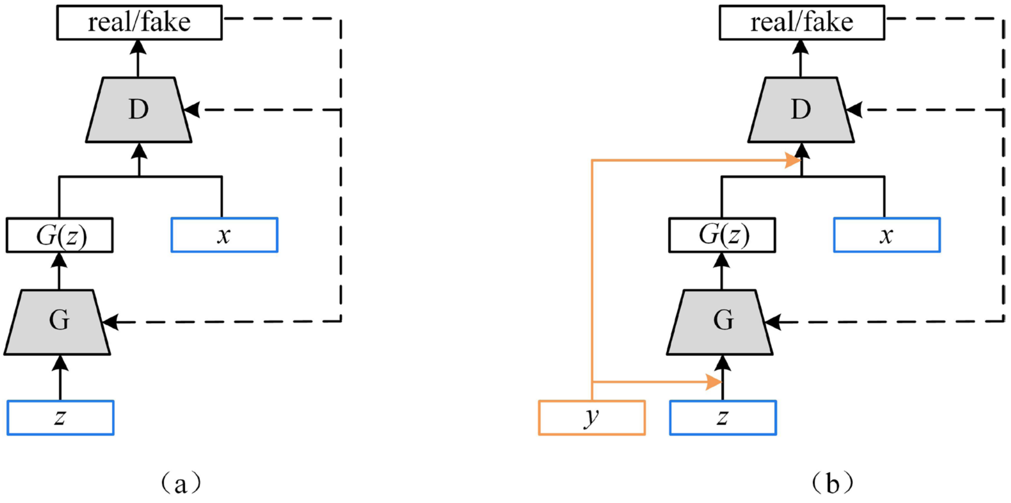
A GAN [28] consists of a generator model G and a discriminator model D, between which there is a two-player competing game: the random noise z is input to G, while the real data x is input to D during training process, the distribution of the real data x is continuously learned through G to make the generated data $G(z)$ as close to the real data x as possible. The goal of D is to distinguish the real data x from the generated data $G(z)$ as much as possible. The ultimate goal of a GAN is that G can be used to fool D while it cannot distinguish between $G(z)$ and x through D. The vanilla GAN architecture is shown in Fig. 1(a). The objective function of the vanilla GAN is formulated as:

$$\min_{G}\max_{D} V(G,D) = E_{x\sim P_x}[\log D(x)] + E_{z\sim P_z}[\log(1-D(G(z)))] \ , \tag{2}$$

where P_x and P_z denote the distribution of real data and input noise, respectively, and E represents mathematic expectation. The mathematical description of the adversarial process is shown as: fix G and maximize $V(G, D)$ $V(G, D)$ through adjusting the parameters of D; fix D and minimize $V(G, D)$ through adjusting the parameters of G.

However, the distribution model does not need to be presupposed for generated data through the vanilla GAN, whose generation process is too free to keep it stable as an unsupervised learning network. For the complex distribution of data or high-resolution images, the training of the vanilla GAN becomes uncontrollable and prone to mode collapse as well as explosion gradients [29]. Through a CGAN, the constraint condition y, such as class labels or images, is introduced into the generator and discriminator, providing guidance for data generation under specific conditions [30]. When comparing a CGAN with the vanilla version, the unsupervised learning model is changed into a supervised one for the training. The CGAN architecture is shown in Fig. 1(b). The objective function of the CGAN is written as:

$$\min_{G}\max_{D} V(G,D) = E_{x\sim P_x}[\log D(x \mid y)] + E_{z\sim P_z}[\log(1-D(G(z \mid y)))] \ . \tag{3}$$



(a) The vanilla GAN architecture

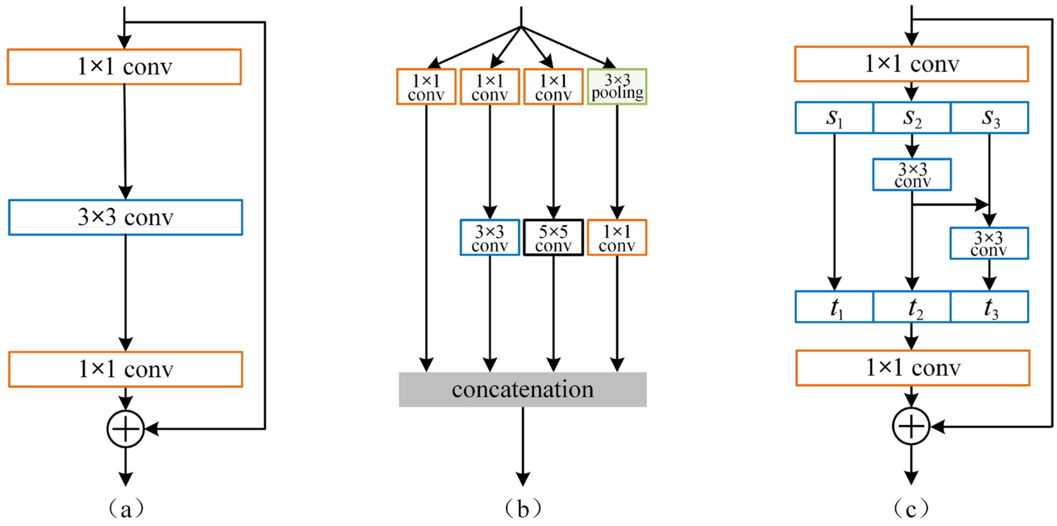
(b) The CGAN architecture

Fig. 1. The vanilla GAN and the CGAN architecture

2.3 Multi-scale Feature Extraction

In computer vision processing tasks, the expression of multi-scale features requires receptive fields of different sizes to extract feature information at different scales. Because of their inherent feature extraction ability, CNNs can be used to extract multi-scale features of an image from coarse to fine by stacking convolutional layers. By increasing the depth and width of CNN network layers to extract richer and more abstract image feature information, this strategy has been widely used, which has achieved good results, such as AlexNet [31] and VGGNet [32]. However, simply stacking network layers or performing multi-scale feature extraction in a multi-layer network cascade will not only cause an increase in running time and memory as well as the overfitting of models, at the same time, redundant network layers can result in the loss of image features during transmission.

Many efficient multi-scale feature extraction approaches have been proposed: the residual learning structure is used to solve the problem of degradation in a deep network architecture through ResNet [33], in which the feature extraction ability of a CNN can be made full use of by stacking convolution layers. In inception networks [34-37], convolutional layers of different kernel sizes are used in parallel to extract multi-scale features of images, and then local features are fused through concatenation to obtain global features. In DenseNet [26], short connections are added between any two network layers, through each of which the current feature-maps are delivered to all of the subsequent layers for reusing features at different layers. Based on the idea of image translation, Kupyn et al [19] proposed using CGAN to perform image deblurring tasks, and the reconstruction effect of images was further improved. Ye et al [38] proposed a scale-iterative upscaling network for image deblurring, and the super-resolution structure was introduced to restore images. Recently, Gao et al. [20] proposed a more effective multi-scale residual block, called Res2Net. Based on the classic bottleneck block of the ResNet model, Res2Net has replaced the middle 3×3 filter with hierarchical residual-like and smaller filter groups, which increase the number of receptive fields at a fine-grained level to enhance the multi-scale feature extraction capability. The architecture differences among ResNet (with bottleneck block) [33], Inception-v1 (with dimension reductions) [34] and Res2Net (with four filter groups) [20] are shown in Fig. 2.



(a) The bottleneck block of ResNet (b) Inception-v1 network with dimension reductions (c) Model Res2Net with four filter groups

Fig. 2. Comparison among multi-scale feature extraction models

3 Proposed Methods

The architecture of our proposed image deblurring network is shown in Fig. 4. The input blurred image is restored to a sharp image in an adversarial pattern between the generator and discriminator.

3.1 Multi-scale Residual Model MSR2B

To better restore the edge structures and texture details of the latent sharp image, the efficient MSR2B proposed in this paper is used for the generator, as the backbone network to fully extract image features at different scales and the potential features. The MSR2B architecture is shown in Fig. 3. The MSR2B model consists of Part A and Part B. A hierarchical residual-like style is used in Part A while the split-transform-merge strategy [35] is used in Part B, and multi-scale representation ability with multiple fine-grained receptive fields is obtained based on the entire model.

In Part A, the number of input feature maps is decreases at a 1×1 bottleneck convolutional layer, where the number of parameters decreases. Then the feature maps are split into four equally-sized groups, denoted by s_i, where $i \in \{1, 2, 3, 4\}$ and the spatial size of each s_i is the same, in which feature channels account for one-fourth of the input channels. The corresponding 3×3 convolution operation in each group s_i is represented by $\mathcal{H}_i$, except for s_1. t_i represents the corresponding output feature map of s_i, where $i \in \{1, 2, 3, 4\}$. Thus, t_i can be defined as:

$$t_i = \begin{cases} s_i & i = 1; \\ \sigma\left(\mathcal{H}_i\left(s_i\right)\right) & i = 2; \\ \sigma\left(\mathcal{H}_i\left(s_i + t_{i-1}\right)\right) & i = 3,4 \end{cases} , \quad \textbf{(4)}$$

where $\sigma(x)$ represents the rectified linear unit (ReLU) function, and $\sigma(x) = max(0, x)$. The 3×3 convolution in s_1 is deleted for reusing features and reducing parameters in the entire MSR2B. Through the current 3×3 convolutional layer $\mathcal{H}_i (2 \le i \le 4)$, features are extracted from s_i, and then the output features of $\mathcal{H}_i$ together with the input features of s_{i+1} are sent to the next convolutional layer $\mathcal{H}_{i+1}$. Repeat this process until the feature maps of all the four groups are processed. Each time when s_i is processed through a 3×3 convolutional layer, the receptive fields of output can be enlarged once, so we obtain different receptive field sizes at t_i $(2 \le i \le 4)$ layers.

In *Part B*, we construct a two-bypass structure, which contains two convolutional layers: a 3×3 convolution and a 5×5 convolution. The features of t_i $(2 \le i \le 4)$ are concatenated and processed through a the two-bypass structure, so that we can further extract image features at different scales. The function of *Part B* can be expressed as:

$$W_1 = \sigma\left(\mathcal{K}_1([t_2, t_3, t_4])\right) , \quad \textbf{(5)}$$

$$W_2 = \sigma\left(\mathcal{K}_2([t_2, t_3, t_4])\right) , \quad \textbf{(6)}$$

$$W_{out} = \mathcal{K}_b\left([W_0, W_1, W_2]\right) , \quad \textbf{(7)}$$

where $\mathcal{K}_1$, $\mathcal{K}_2$ and $\mathcal{K}_b$ denote the 3×3 convolution, 5×5 convolution and the rearward 1×1 bottleneck convolution operation, respectively, while $[t_2, t_3, t_4]$ and $[W_0, W_1, W_2]$ represent the concatenation operation. Finally, the feature maps of *Part A* and *Part B* are concatenated and processed through a 1×1 bottleneck convolutional layer, which recovers 256 feature maps. Consequently, the change in the number of feature-maps based on the overall MSR2B model is constant, which can be described as $C_{n-1} = C_n = 256$, where C_{n-1} and C_n denote the input and output feature maps of a MSR2B, respectively. It is based on this architecture that we can stack multiple MSR2Bs in the generator structure. A shortcut connection and element-wise addition are applied in the MSR2B, which can be described as:

$$C_n = C_{n-1} + W_{out} . \quad \textbf{(8)}$$

With our MSR2B model, we use the split and merge feature map method to enrich the receptive field sizes and extract image features at different scales. The residual learning makes the entire network more efficient.

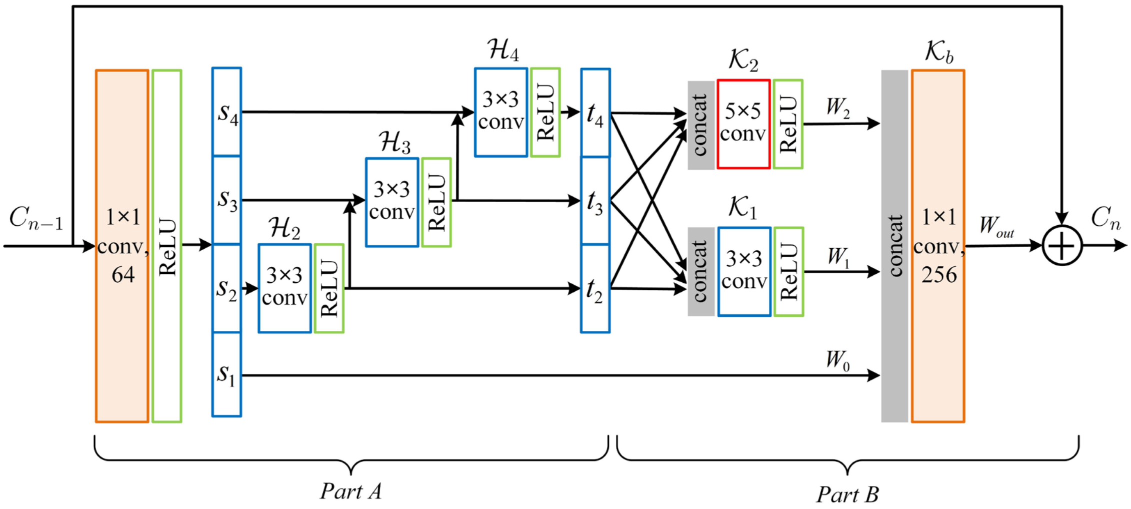


Fig. 3. Architecture of multi-scale Res2Net block (MSR2B)

3.2 Network Architecture

The generator CNN architecture in this paper is constructed with a MSR2B as the main body, whose structure is shown in Fig. 4. The generator network contains one convolutional layer with Stride 1 and a kernel size of 7×7, two stride convolutional layers with Stride 2 and a kernel size of 3×3, which are used to downsample the images and increase the number of feature channels. There are nine MSR2Bs to extract features and a bottleneck convolutional layer with 1×1 kernels, which are used for extracting and fusing the features of all of the output of the nine MSR2Bs. We then use two transposed convolutional layers with Stride 2 and 3×3 kernels, and then an ordinary convolutional layer with 1×1 kernels, for upsampling and recovering the identical number of channels in the input images. Each convolutional layer is followed by a ReLU activation layer to enhance the nonlinear expression ability of the network. In addition, we add global and local skip connections in the generator architecture to strengthen its feature learning ability. Through the global skip connection, the residual I_R (denoted by the restored image) of I_B (denoted by the blurred image) is learned through the generator to reduce the complexity of end-to-end learning from I_B to I_R, thereby making the network more efficient. This residual learning process can be formulated as:

$$I_S = I_B + I_R \ , \qquad \textbf{(9)}$$

where I_S denotes the actual sharp image. To prevent the loss of features during the transmission process and make full use of the feature information extracted by the MSR2Bs, a bottleneck convolution is added at the end of the ninth MSR2B, meanwhile local skip connections are constructed between the MSR2Bs and the bottleneck convolution layer. The InstanceNorm (IN) layer is removed from the generator network, because with the addition of the IN layer in image deblurring tasks, the ability to characterize details will be weakened while bringing artifacts and checkerboard effect to the generated images [39].

The discriminator network in this paper is identical to PatchGAN, whose structure is shown in Fig. 4. Unlike the widely-used two-classifiers, which a scalar value is directly output after convolutions to classify an image as real or fake, a matrix X with the size of $N \times N$ size is output via PatchGAN. The value of neuron X_{ij} in X represents the probability that the corresponding receptive field patch ij is real or fake, and through the PatchGAN network, an assumption is made that the pixels separated by more than a patch diameter are independent, thereby modeling an image as the Markov random field [27]. The average probability value of all the patches in an image is taken as the integral probability value through PatchGAN, which is used to classify whether the image is real or fake. Compared with the ordinary two classifiers, more attention is paid to the high-frequency features of the image through PatchGAN by identifying each patch, focused on characterizing and extracting local features, which is more conducive to recovering sharp images from complex blurred images. PatchGAN can be regarded as the texture loss for training [27]. In this paper, the discriminator architecture contains five convolutional layers with a kernel size of 4×4. The front four layers, with Stride 2, are followed by an IN layer and LeakyReLU function with $\alpha = 0.2$, and at the last layer with Stride 1, the matrix X is output as 30×30×1, with a size of 70×70 for each patch.

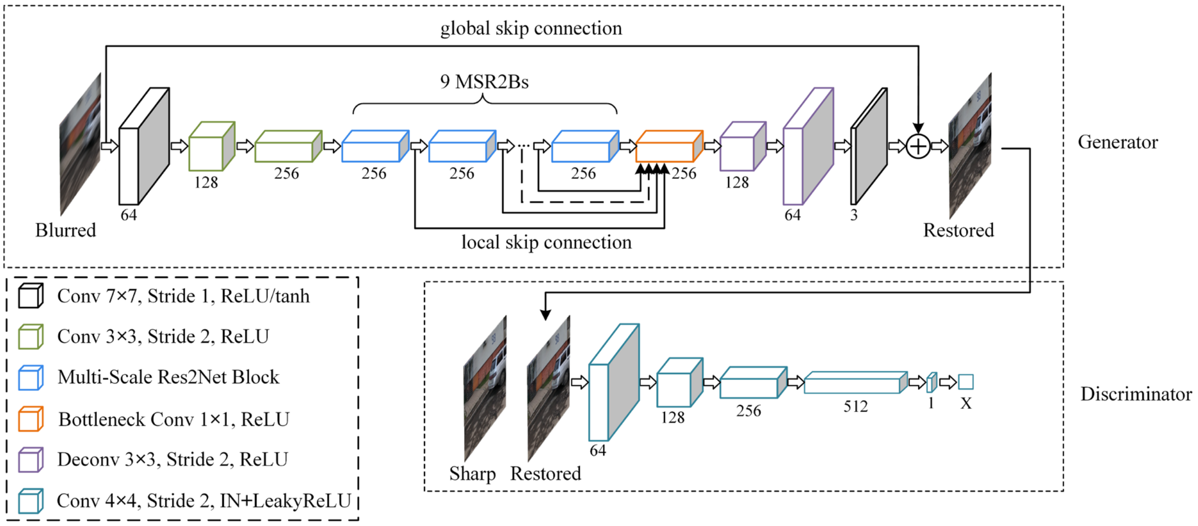


Fig. 4. Architecture of the proposed deblurring network

3.3 Loss Function

The loss function in this paper is composed of two parts: adversarial loss and content loss. The total loss function is expressed as:

$$L_{total} = L_{adv} + \lambda L_{pers} \quad . \tag{10}$$

The vanilla GAN has problems such as mode collapse and unstable convergences during the training process. It was pointed out that the problems WGAN [40] were caused by the application of the Jensen-Shannon and Kullback-Leibler divergence as the optimization strategy, which were proposed instead of using the earth-mover (EM) distance to measure the distance between the real and generated sample distribution, thus optimizing the sample quality. However, WGAN enforced the Lipschitz constraint on the discriminator by clipping weight, which resulted in vanishing or exploding gradients. To solve this problem, gradient penalty was used for WGAN-GP [41] instead of weight clipping, and better performance was obtained compared with WGAN. The objective function of WGAN-GP is

$$L_{adv} = E_{x\sim P_x}[D(x)] - E_{\tilde{x}\sim P_{\tilde{x}}}[D(\tilde{x})] + \mu E_{\hat{x}\sim P_{\hat{x}}}[(\|\nabla_x D(\hat{x})\|_2 - 1)^2] \ , \tag{11}$$

where $\tilde{x} = G(z)$, x and $\tilde{x}$ denote the sharp image data and restored data respectively, P_x and $P_{\tilde{x}}$ denote the sharp image data distribution and generator model distribution, respectively. $\hat{x}$ is the random sample between x and $\tilde{x}$, and $P_{\hat{x}}$ is defined as the sampling along straight lines between P_x and $P_{\tilde{x}}$. μ represents the penalty coefficient. In this paper, we define L_{adv} as the adversarial loss function, which helps remain the training process stable and easy to converge.

To define the content loss function, we use perceptual loss [42] instead of the classic L1 or L2 loss, through which the content consistency between the generated and sharp images can be improved. The perceptual loss L_{pers} is defined by the feature map parameters within the pre-trained VGG19 network [32], which is defined as:

$$L_{pers} = \frac{1}{W_{ij} H_{ij}} \sum_{x=1}^{W_{ij}} \sum_{y=1}^{H_{ij}} \left[\phi_{ij}(S)_{x,y} - \phi_{ij}\left(G(B)\right)_{x,y} \right]^2 , \tag{12}$$

where S and $G(B)$ represent the sharp and restored image respectively. ϕ_{ij} represents the feature map generated by the jth convolutional layer before the ith max-pooling layer in the pretrained VGG19 network. W_{ij} and H_{ij} denote the sizes of feature maps. Because the feature maps in lower layers represent more texture features compared with deeper layers, both coefficients i, j are set as 3 respectively. The perceptual loss is helpful for generating more realistic images.

4 Experiments

4.1 Datasets

The GOPRO dataset was proposed by Nah et al, who used a GoPro4 Hero Black high-speed camera to shoot videos of multiple scenes at a frame rate of 240 fps, and averaged the consecutive frames in videos to generate blurred images. These blurred images simulated the realistic blurred scenes such as camera shake, defocus and relative displacement of objects. The GOPRO dataset included 3214 sharp-blurred image pairs, of which 2103 were selected as the training set, and the other 1111 were used as the test set.

The Kohler dataset was obtained by Kohler et al. [43], through recording and replaying the motion trajectory of a 6D camera. It is used as a standard dataset for evaluating the blind image deblurring algorithms and contains 4 sharp images, each of which corresponds to 12 blurred images with different blur kernels. However, the Kohler dataset is small, which can only be used as a test set.

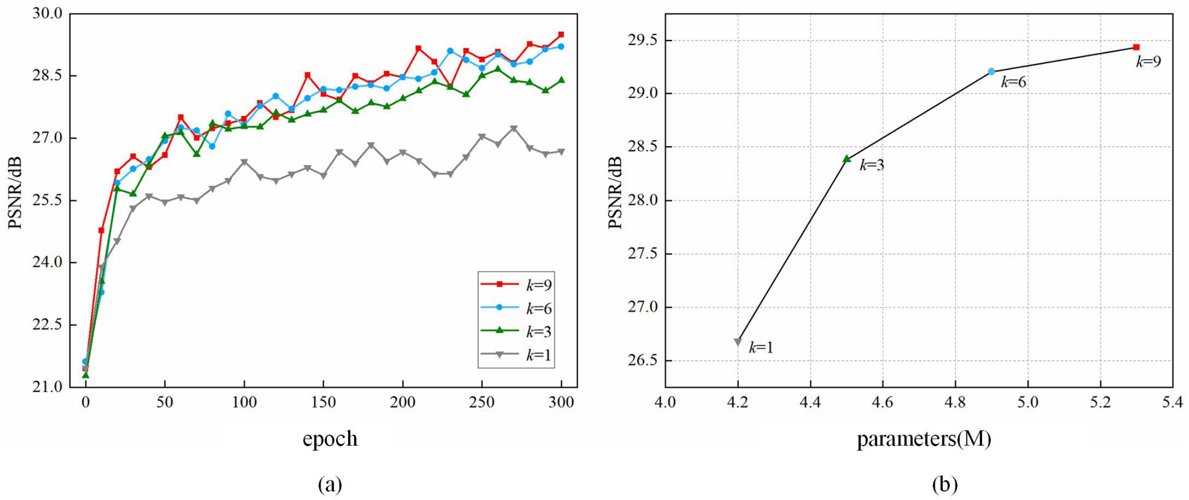
4.2 Training Details

We implemented our deblurring network based on the Pytorch deep learning framework using a desktop computer with an Intel i7-6900K CPU and a Nvidia Titan X GPU. An Adam optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$ is used for the network in the training process. In our experiments, the model converged after 300 epochs, and the learning rate of the generator and discriminator was set as 10^{-4} for the initial 200 epochs, which then decayed linearly to 10^{-7} for the next 100 epochs. We set batch size as 1 and the weight coefficient λ of L_{pers} as 100 during training. We treated the ground-truth sharp images as the constraint condition y, and used the randomly-cropped images containing 256×256 pixels as the training input.

To accurately evaluate image quality and the effectiveness of the deblurring model proposed in this paper, peak signal to noise ratio (PSNR) and structural similarity (SSIM) were used as the objective evaluation criteria. PSNR reflected the degree of difference among corresponding pixels, and the evaluation result was expressed in dB. The similarity of images was measured through SSIM from three aspects: brightness, contrast and structure. A higher PSNR or SSIM value meant a better image quality.

4.3 Effectiveness of the MSR2B

Firstly, the influence of the number of MSR2B modules k on the network performance was verified. As shown in Fig. 5, it can be seen that at the beginning, with the increase of MSR2B modules, the network deblurring effect improves rapidly. However, continuing to increase the number of MSR2B modules will lead to the increase of network parameters, and the efficiency of performance improvement is not obvious. Therefore, in order to balance the computational complexity and deblurring performance, the final number of MSR2B modules k was set as 9.



(a) With the increase of training epochs, the network performance comparison

(b) The performance and parameter trade-off curve

Fig. 5. The impact of different numbers of MSR2Bs on network performance

To verify the effectiveness of the MSR2B proposed in this paper, we compared this structure with the baseline models ResNet and Res2Net. Although we could have increased the number of MSR2Bs to deepen the network and improve its performance, in order to balance the network performance and complexity, we ended up selecting nine MSR2Bs. For a fair comparison, we replaced them with the same number of ResNet and Res2Net blocks in the generator CNN architecture, and kept the same experimental parameters.

The quantitative results of the deblurring networks with three residual learning models based on GOPRO evaluation dataset are shown in Table 1, and the visual comparison is shown in Fig. 6. We can conclude that model MSR2B performs much better than the classic model ResNet, improving the average PSNR and SSIM by 1.02 dB and 0.017 respectively meanwhile achieving a further improvement in deblurring tasks compared with Res2Net. From the visual comparison results, we can see that ResNet can be used to recover the edge structures from severely-blurred input images, but the texture details are still not clear enough, and it is difficult to recognize valid content information from the restored images; Res2Net is used to recover clearer images for its hierarchical ability to extract features, but some characters are still distorted. The deblurring performance of MSR2B shows a significantly more improvement than other models, and richer texture details as well as clearer edge structure characters are restored.

Table 1. Quantitative results of different residual blocks

Models	ResNet	Res2Net	MSR2B
PSNR/dB	28.11	28.90	29.13
SSIM	0.944	0.951	0.961

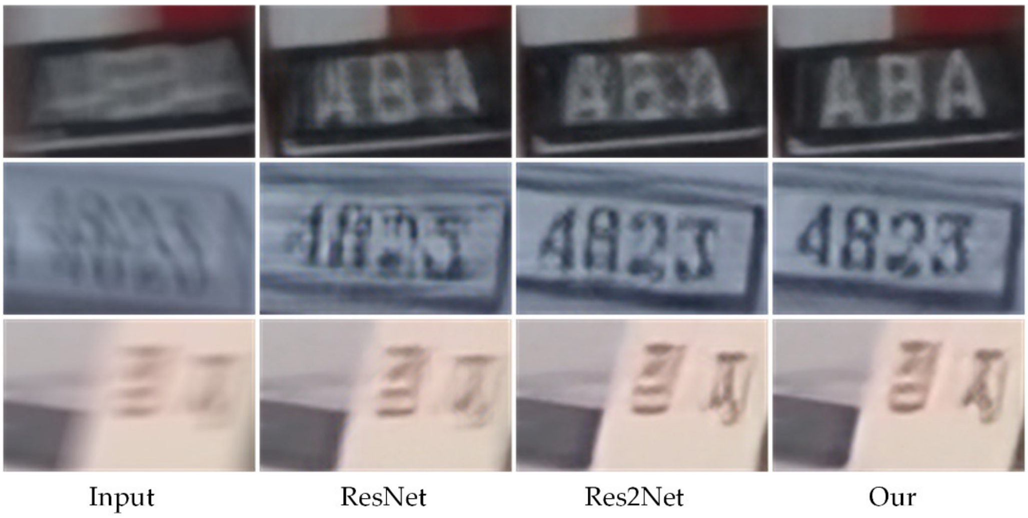


Fig. 6. Visual comparison of different residual blocks
(From left to right: results of ResNet, Res2Net and our proposed MSR2B)

4.4 Comparison with Other Methods

We compared our method with state-of-the-art deblurring methods on the GOPRO and Kohler evaluation dataset, and the results or official codes were directly taken from their papers. Robust kernels were estimated based on Reference [24] by removing outliers from image edges, which we chose as the representative traditional non-uniform deblurring algorithm. A CNN was used in Reference [10] to estimate blur kernels, through which the latent sharp image was recovered via deconvolution. A multi-scale CNN was used for image deblurring in Reference [14] without the process of estimating blur kernels. A GAN and an image-to-image translation strategy were applied to an end-to-end image deblurring task in Reference [19], where a deep residual network was used to extract abstract features. The model also has a faster inference speed than a multi-scale CNN.

The quantitative results of different methods based on GOPRO and Kohler evaluation dataset are shown in Table 2, where the highest PSNR and SSIM value among all the state-of-the-art methods based on both datasets is achieved through our deblurring method. Our method also has the fastest deblur speed for 720p images on GOPRO dataset, averaging 0.45s for each image, which is 24% less than the inference time of Kupyn et al. On Kohler dataset, the PSNR and SSIM of our method increase by 2.0% and 2.5% compared with Nah et al. and Kupyn et al. respectively. Fig. 7 shows the visualization results of Nah et al., Kupyn et al. and our method on GOPRO test dataset. The results are generated based on models trained only on the GOPRO training dataset. The visual comparisons show that a good deblurring effect has been achieved in the results of Nah et al. and Kupyn et al., but the texture and detail features of the input images still fail to be restored, such as the color of the signboard, the frame outline of the window and the text on the billboard. By comparison, through our method, the edges and textures can be restored more clearly and faithfully. The visualization results of Nah et al., Kupyn et al., and our method on the Kohler dataset are shown in Fig. 8. Compared with Nah et al., through our method, the images can be restored more clearly, while compared with Kupyn et al., the artifacts in the input images can be reduced through our method, which shows a good generalization ability. These visual comparisons show that our proposed method visually performs much better than other state-of-the-art methods.

Table 2. Quantitative results of different residual blocks

Methods	GOPRO dataset		Kohler dataset		Inference time
	PSNR/dB	SSIM	PSNR/dB	SSIM	
Pan et al. [24]	23.50	0.836	25.47	0.811	26 min
Sun et al. [10]	24.64	0.843	25.22	0.773	20 min
Nah et al. [14]	28.93	0.910	25.74	0.804	7.21 s
Kupyn et al. [19]	28.70	0.958	25.86	0.802	0.85 s
Ours	29.13	0.961	26.62	0.836	0.45 s

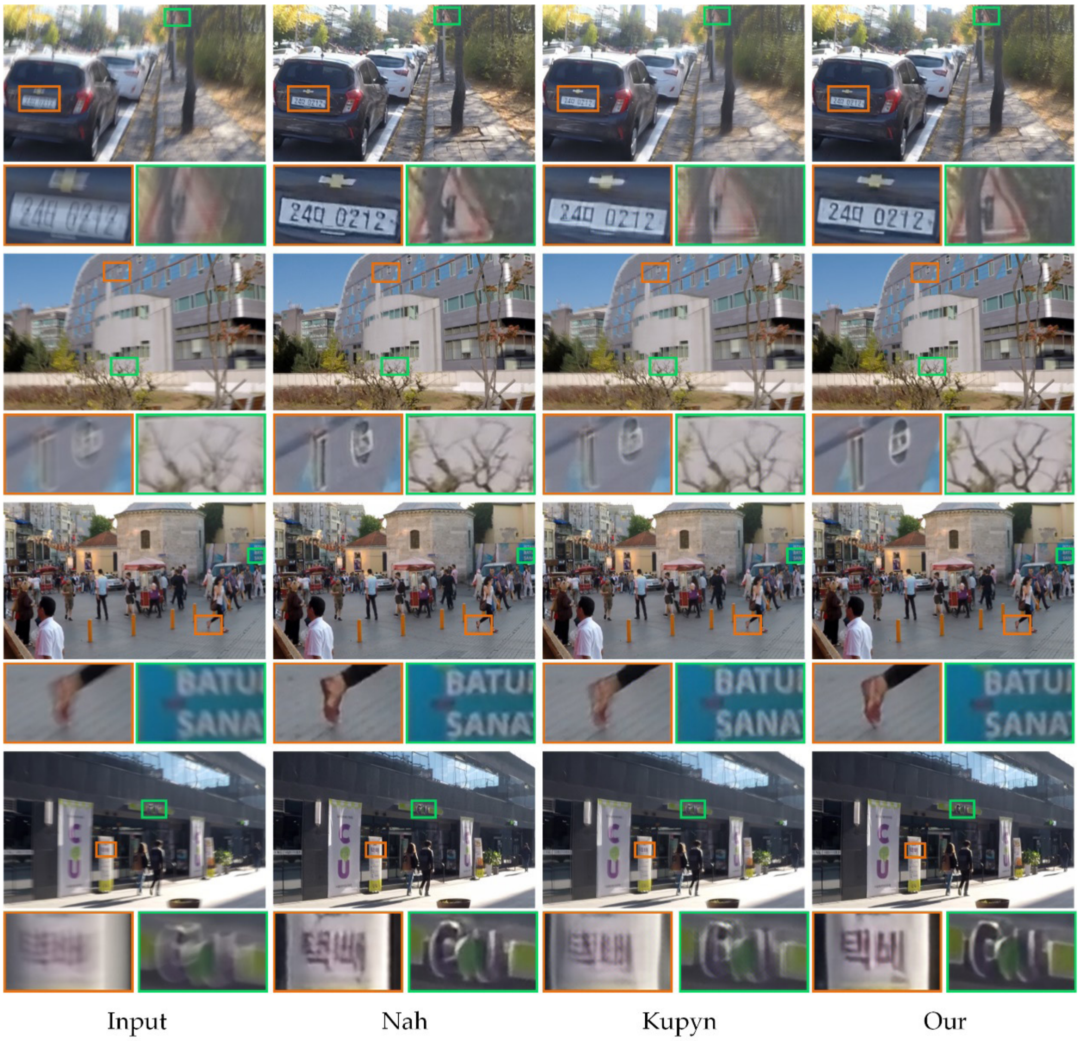


Fig. 7. Visual comparison on GOPRO test dataset
(From left to right: input blurred images, results of Nah et al., Kupyn et al. and our method)

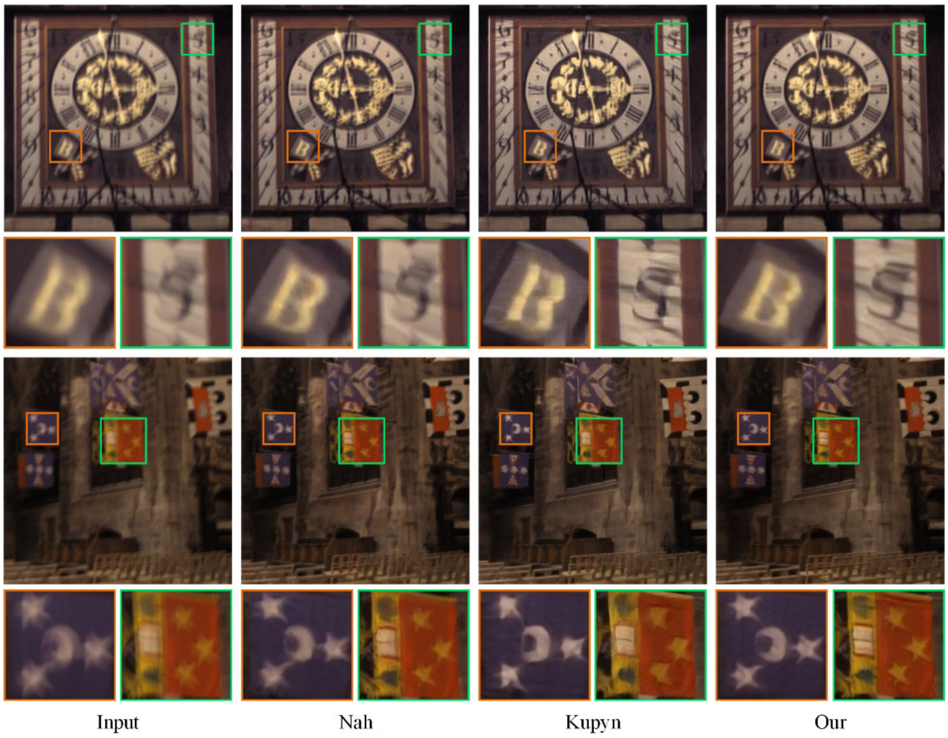


Fig. 8. Visual comparison on Kohler test dataset
(From left to right: input blurred images, results of Nah et al., Kupyn et al. and our method)

5 Conclusions

In this paper, we describe a multi-scale CGAN structure for using a multi-scale feature extraction approach in end-to-end image deblurring. In a generator network, we add global and local skip connections to enhance the learning efficiency and an efficient representation of multi-scale features. We also use the multi-scale feature extraction block MSR2B, in which the split and merge feature map method is applied to the extraction of multi-scale features, which achieves better performance than the baseline residual blocks ResNet and Res2Net. In the discriminator network, the PatchGAN is used as a critical network, because it is more conducive to characterizing and extracting local features, which we treat as a form of texture loss. The total loss function consists of adversarial loss optimized by WGAN-GP and perceptual loss. The former helps make the training more stable, while the latter contributes to the generation of more realistic images. Experimental results show that the method proposed outperforms previous representative deblurring methods.

Our MSR2B module can be integrated with existing deblurring methods with no effort, and the global and local skip connections can effectively enhance the feature extraction ability of convolutional neural networks, but both of them have the disadvantage of increasing computational parameters and cost, which is the next step to be improved.

6 Conclusions

This research is supported by Qingyang Science and Technology Planning Projects (No. QY-STK-2022A-086, No. QY-STK-2022A-020).

References

[1] L. Xu, J. Jia, Two-phase kernel estimation for robust motion deblurring, in: Proc. 2010 European Conference on Computer Vision, 2010.

[2] J. Pan, R. Liu, Z. Su, X. Gu, Kernel estimation from salient structure for robust motion deblurring, Signal Processing: image Communication 28(9)(2013) 1156-1170.

[3] A. Levin, Y. Weiss, F. Durand, W.T. Freeman, Efficient marginal likelihood optimization in blind deconvolution, in: Proc. 2011 IEEE Conference on Computer Vision and Pattern Recognition, 2011.

[4] D. Krishnan, T. Tay, R. Fergus, Blind deconvolution using a normalized sparsity measure, in: Proc. 2011 IEEE Conference on Computer Vision and Pattern Recognition, 2011.

[5] L. Xu, S. Zheng, J. Jia, Unnatural l0 sparse representation for natural image deblurring, in: Proc. 2013 IEEE Conference on Computer Vision and Pattern Recognition, 2013.

[6] X. Chen, J. Yang, Q. Wu, Image deblur in gradient domain, Optical Engineering 49(11)(2010) 792-796.

[7] T. Ma, T. Huang, X. Zhao, Y. Lou, Image deblurring with an inaccurate blur kernel using a group-based low-rank image prior, Information Sciences 408(2017) 213-233.

[8] J. Pan, D. Sun, H. Pfister, M. Yang, Blind image deblurring using dark channel prior, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[9] Z. Zha, X. Zhang, Y. Wu, Q. Wang, X. Liu, L. Tang, X. Yuan, Non-convex weighted ℓp nuclear norm based ADMM framework for image restoration, Neurocomputing 311(2018) 209-224.

[10] J. Sun, W. Cao, Z. Xu, J. Ponce, Learning a convolutional neural network for non-uniform motion blur removal, in: Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[11] D. Gong, J. Yang, L. Liu, Y. Zhang, I. Reid, C. Shen, A.V.D. Hengel, Q. Shi, From motion blur to motion flow: a deep learning solution for removing heterogeneous motion blur, in: Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[12] A. Chakrabarti, A neural approach to blind motion deblurring, in: Proc. 2016 European Conference on Computer Vision, 2016.

[13] C.J. Schuler, M. Hirsch, S. Harmeling, B. Scholkopf, Learning to deblur, IEEE Transactions on Pattern Analysis and Machine Intelligence 38(7)(2016) 1439-1451.

[14] S. Nah, T.H. Kim, K.M. Lee, Deep multi-scale convolutional neural network for dynamic scene deblurring, in: Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[15] X. Tao, H. Gao, X. Shen, J. Wang, J. Jia, Scale-recurrent network for deep image deblurring, in: Proc. 2018 IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[16] M. Noroozi, P. Chandramouli, P. Favaro, Motion deblurring in the wild, in: Proc. 2017 German Conference on Pattern Recognition, 2017.

[17] J. Zhang, J. Pan, J. Ren, Y. Song, L. Bao, R.W.H. Lau, M.-H. Yang, Dynamic scene deblurring using spatially variant recurrent neural networks, in: Proc. 2018 IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[18] O. Kupyn, T. Martyniuk, J. Wu, Z. Wang, DeblurGAN-v2: Deblurring (orders-of-magnitude) faster and better, in: Proc. 2019 IEEE International Conference on Computer Vision, 2019.

[19] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, J. Matas, DeblurGAN: blind motion deblurring using conditional adversarial networks, in: Proc. 2018 IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[20] S. Gao, M. Cheng, K. Zhao, X. Zhang, M. Yang, P.H.S. Torr, Res2Net: a new multi-scale backbone architecture, IEEE Transactions on Pattern Analysis and Machine Intelligence 43(2)(2021) 652 - 662.

[21] C. Li, M. Wand, Precomputed real-time texture synthesis with markovian generative adversarial networks, in: Proc. 2016 European Conference on Computer Vision, 2016.

[22] R. Fergus, B. Singh, A. Hertzmann, S.T. Roweis, W.T. Freeman, Removing camera shake from a single photograph, in: Proc. SIGGRAPH06: Special Interest Group on Computer Graphics and Interactive Techniques Conference, 2006.

[23] S. Cho, S. Lee, Fast motion deblurring, in: Proc. ACM Transactions on Graphics (SIGGRAPH Asia 2009), 2009.

[24] J. Pan, Z. Lin, Z. Su, M. Yang, Robust kernel estimation with outliers handling for image deblurring, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[25] S. Ramakrishnan, S. Pachori, A. Gangopadhyay, S. Raman, Deep generative filter for motion deblurring, in: Proc. 2017 IEEE International Conference on Computer Vision, 2017.

[26] G. Huang, Z. Liu, L.V. Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[27] P. Isola, J. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Proc. 2014 Advances in Neural Information Processing Systerms, 2014.

[29] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training GANs, in: Proc. 2016 Advances in Neural Information Processing Systerms, 2016.

[30] M. Mirza, S. Osindero, Conditional generative adversarial nets. <https://arxiv.org/abs/1411.1784>, 2014 (accessed 06.11.2014).

[31] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Proc. 2012 Advances in Neural Information Processing Systerms, 2012.

[32] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Proc. 3rd International Conference on Learning Representations, ICLR 2015, 2015.

[33] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[35] C. Szegedy, S. Ioffe, V. Vanhoucke, A.A. Alemi, Inception-v4, Inception-ResNet and the impact of residual connections on learning, in: Proc. 2017 National Conference of the American Association for Artificial Intelligence, 2017.

[36] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[37] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in: Proc. 2015 32nd International Conference on Machine Learning, 2015.

[38] M. Ye, D. Lyu, G. Chen, Scale-iterative upscaling network for image deblurring, IEEE Access 8(2020) 18316-18325.

[39] W. Shao, Y.-Y. Liu, L.-Y. Ye, L.-Q. Wang, Q. Ge, B.-K. Bao, H.-B. Li, DeblurGAN+: revisiting blind motion deblurring using conditional adversarial networks, Signal Processing 168(2020) 107338.

[40] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, in: Proc. 2017 34th International Conference on Machine Learning, 2017.

[41] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A. Courville, Improved training of wasserstein GANs, in: Proc. 2017 Advances in Neural Information Processing Systerms, 2017.

[42] J. Johnson, A. Alahi, F.-F. Li, Perceptual losses for real-time style transfer and super-resolution, in: Proc. 2016 European Conference on Computer Vision, 2016.

[43] R. Kohler, M. Hirsch, B.J. Mohler, B. Scholkopf, S. Harmeling, Recording and playback of camera shake: benchmarking blind deconvolution with a real-world database, in: Proc. 2012 European Conference on Computer Vision, 2012.