Kernel Modeling Super-Resolution on Real Low-Resolution Images

Ruofan Zhou
IC, EPFL
ruofan.zhou@epfl.ch

Sabine Süsstrunk
IC, EPFL
sabine.susstrunk@epfl.ch

Abstract

Deep convolutional neural networks (CNNs), trained on corresponding pairs of high- and low-resolution images, achieve state-of-the-art performance in single-image super-resolution and surpass previous signal-processing based approaches. However, their performance is limited when applied to real photographs. The reason lies in their training data: low-resolution (LR) images are obtained by bicubic interpolation of the corresponding high-resolution (HR) images. The applied convolution kernel significantly differs from real-world camera-blur. Consequently, while current CNNs well super-resolve bicubic-downsampled LR images, they often fail on camera-captured LR images.

To improve generalization and robustness of deep super-resolution CNNs on real photographs, we present a kernel modeling super-resolution network (KMSR) that incorporates blur-kernel modeling in the training. Our proposed KMSR consists of two stages: we first build a pool of realistic blur-kernels with a generative adversarial network (GAN) and then we train a super-resolution network with HR and corresponding LR images constructed with the generated kernels. Our extensive experimental validations demonstrate the effectiveness of our single-image super-resolution approach on photographs with unknown blur-kernels.

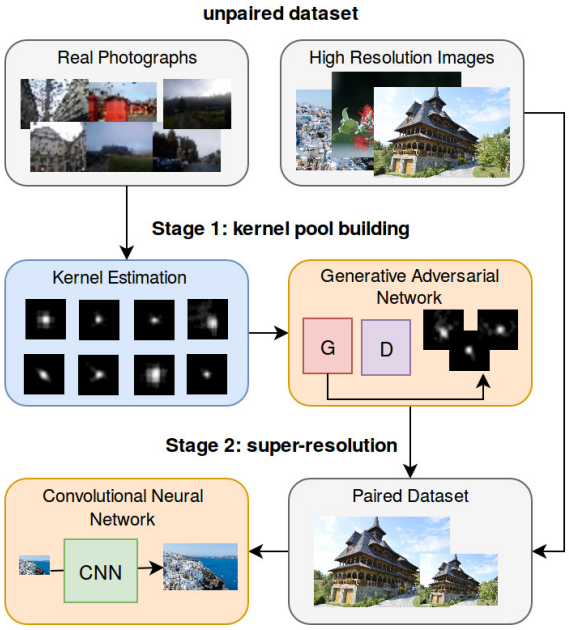


Figure 1: Illustration of our proposed kernel modeling super-resolution (KMSR) framework. The first stage consists of blur-kernel estimation from real photographs, which are used in training a GAN to generate a large pool of realistic blur-kernels. These generated blur-kernels are then utilized to create a paired dataset of corresponding HR and LR images for the training of a deep CNN.

1. Introduction

Single-image super-resolution methods aim to reconstruct a high-resolution (HR) image from a single, low-resolution (LR) image by recovering high-frequency details. Classic super-resolution (SR) algorithms [40, 41, 57] analytically model the blur-kernel and real-image properties in order to recover the HR images. In contrast, many modern SR methods [21, 45, 49] attempt to learn a mapping from LR images to HR images. Lately, several convolutional neural network (CNN) based SR models were developed [8, 17, 26, 31, 42, 55]. All these learning-based methods require large sets of paired LR and HR images for training.

It is non-trivial to obtain such paired LR and HR ground-truth images of real scenes. Current CNN-based SR networks thus rely on synthetically generated LR images [44]. The most common technique is to apply bicubic interpolation [25] to the HR image. However, the bicubic convolution kernel is different from real camera-blur [32]. The loss of high-frequency details in camera-captured images is due to several factors, such as optical blur, atmospheric blur, camera shake, and lens aberrations [34]. As a result, even though these CNN-based SR networks perform well on bicubic-downsampled LR images, their performance is limited on real photographs as they operate under a wrong

kernel assumption [11, 32]. Generative adversarial network (GAN) based methods [3, 30, 39, 47] can be extended to train SR networks on unpaired datasets, but they still rely on unrealistic blur-kernels. Super-resolution on real LR photographs with unknown camera-blur thus remains a challenging problem.

To generate synthetic LR images with real camera-blur, we can use kernel-estimation algorithms [28, 29, 35] to extract realistic blur-kernels from real LR photographs. However, as each camera, lens, aperture, and atmospheric-condition combination may result in a different blur-kernel, it is challenging to generate a sufficiently large and diverse dataset [28, 29] needed to train a SR network.

One approach is to generate synthetic LR images using many blur-kernels [36], which will improve the generalization ability of the SR network. Using a kernel estimator, we first extract blur-kernels from real photographs and use them for training a GAN. First proposed in [12], GANs are a class of neural networks that learn to generate synthetic samples with the same distribution as the given training data [2]. We thus augment the limited kernel-set we obtained using kernel estimation by leveraging the GAN's ability to approximate complex distributions [24, 30, 38, 50] to learn and generate additional blur-kernels.

Our Kernel Modeling Super-Resolution (KMSR) thus consists of two stages, as shown in Fig. 1. We first generate a GAN-augmented realistic blur-kernel pool by extracting real blur-kernels from photographs with a kernel estimation algorithm and by training a GAN to augment the kernel pool. We then construct a paired LR-HR training dataset with kernels sampled from the kernel pool, and train a deep CNN for SR.

Our major contributions in this paper are as follows: (1) we introduce KMSR to improve blind SR on real photographs by incorporating realistic blur-kernels in the framework, which improves the generalization capability of the network to unseen blur-kernels, (2) we show that a GAN can reliably generate realistic blur-kernels, and (3) we demonstrate with experiments on real images that the proposed KMSR achieves state-of-the-art results in terms of both visual quality and objective metrics.

2. Related Work

2.1. CNN-based Image Super-Resolution

Deep network architectures for super-resolution is an active research topic as they show good performance on synthetic LR images [44]. Dong *et al*. [8] adopt a 3-layer CNN to learn an end-to-end mapping from interpolated LR images to HR images. They achieve comparative results to conventional SR methods. Global [26] and local [17, 31] residual learning strategies can be employed to reduce the learning difficulty and simplify the training, and thus to optimize the performance of the SR networks. Shi *et al*. [42] suggest a sub-pixel upscaling that further increases the receptive field of the network; this provides more contextual information which helps to generate more accurate details.

All these networks are trained with paired LR-HR data, and often use a fixed down-sampling procedure for generating synthetic LR images. This leads to poor network generalization on real photographs as the actual image-acquisition does not correspond to the learned model. Some methods propose to capture real LR-HR image pairs by using different optical zoom [5, 54], but the networks trained on such datasets are limited to one specific camera model. Recent methods do propose to incorporate the degradation parameters including the blur-kernel into the network [14, 43, 51, 52, 53]. However, these methods rely on blur-kernel estimation algorithms only and thus have limited ability to handle arbitrary blur-kernels. In this paper, we solve the problem by modeling realistic kernels when creating the training dataset, which improves the practicality and generalization of the SR networks.

2.2. Blur-Kernel Estimation

In recent years, we have witnessed significant advances in single-image deblurring, as well as blur-kernel estimation. Efficient methods based on Maximum A Posteriori (MAP) formulations were developed with different likelihood functions and image priors [4]. In particular, heuristic edge-selection methods for kernel estimation [7] were proposed for the MAP estimation framework. To better recover the blur-kernel and better reconstruct sharp edges for image deblurring, some exemplar-based methods [16] exploit the information contained in both the blurred input and example images from an external dataset. More recently, the dark-channel prior [19] was used by Pan *et al*. [35] to simply and efficiently estimate the blur-kernel of natural images. As they achieve significant performance on deblurring tasks [29], we adopt their kernel-estimation algorithm for collecting blur-kernels of real images.

2.3. Generative Adversarial Network

GANs were proposed to approximate intractable probabilistic computations [24, 30, 38, 50, 59], and are used in some SR networks to improve visual quality [30, 39, 47]. However, training a GAN can be tricky and unstable, and it is rather hard to generate artifact-free HR images [24, 38, 50]. DCGAN [37] provides some useful guidelines for building and training GANs. WGAN [1, 15] further improves GAN training by overcoming the difficulties in maintaining training-balance between the generative network, the discriminative model, and the network architecture design. Several applications also demonstrate their ability to augment limited training data for deep learning [2, 6]. Therefore, we employ WGAN-GP [15], an im-

proved version of WGAN, to generate a large pool of kernels that are then employed to generate realistic LR images for the training of our KMSR network.

3. Proposed Method

This section introduces our kernel modeling super-resolution solution for real photos: KMSR. It is composed of a kernel-pool creation stage and a CNN-type SR network (See Fig. 1). We first introduce the imaging model under which we obtain LR and HR images. Then, we discuss the details of the kernel-pool generation and the SR network architecture.

3.1. Kernel Modeling Blind Super-Resolution

Let y be an HR image of size $r_1 \times r_2$ pixels, and let x be an LR observation of y of size $\lfloor r_1/s \rfloor \times \lfloor r_2/s \rfloor$, where $s > 1$ is the downsampling factor. The relation between x and y is expressed as in [11]:

$$x = (y * k) \downarrow^s + n, \tag{1}$$

where k denotes an unknown blur-kernel, $\downarrow^s$ denotes a decimation operator by a factor s, and n is the noise. We assume here that there is no noise in the LR image acquisition model, *i.e.*, $n = 0$.

We upscale the LR image to a coarse HR image x' with the desired size $r_1 \times r_2$ with traditional bicubic interpolation by the same factor s:

$$x' = (x * b_s), \tag{2}$$

where b_s is the bicubic-upscaling kernel with scale s. Thus we have

$$x' = ((y * k) \downarrow^s) * b_s, \tag{3}$$

Simplified,

$$x' = y * k' \tag{4}$$

where $k' = (k * b_s) \downarrow^s$.

To train a blind CNN SR network, we need paired training data y and x', obtained according to Eqn. 4 with different kernels k'. We adopt a GAN to help solve this problem. As discussed in Section 2.3, it is difficult to train a generative network to consistently recover HR images without artifacts. Thus, alternatively, our GAN is trained to produce blur-kernels rather than images.

3.2. Blur-Kernel Pool

Before building the paired training dataset, realistic blur-kernels need to be estimated from real photographs. These kernels are then used to better train the GAN for kernel modeling and kernel generation. The combination of the estimated kernels and the GAN-generated kernels forms the large kernel-pool used in building paired LR-HR training data.

3.2.1 Blur-Kernel Estimation

To generate a set of realistic blur-kernels $K' = \{k'_1, k'_2, \ldots, k'_e\}$, we first randomly extract a patch p of size $d \times d$ from the bicubic-upscaled LR image (or coarse HR image) x'. We then estimate the blur-kernel k' of size 25×25 from p using the blur-kernel estimation algorithm of [35]. Their standard formulation for image deblurring, based on the dark-channel prior [19], is as follows:

$$\min_{p,k'} \|\nabla p * k' - \nabla p\| + \theta \|k'\|_2^2 + \mu \|\nabla p\|_0 + \left\|\nabla p^{dark}\right\|_0 \tag{5}$$

p is the extracted patch from x', and p^{dark} is the dark channel [19] of the patch. Coordinate descent is used to alternatively solve for the latent patch p and the blur-kernel k'. The details can be found in [35]. To eliminate patches that are lacking high-frequency details (such as patches extracted from the sky, walls, etc.) in which the blur-kernel estimation algorithm might fail, we define constraints for p as follows:

$$|Mean(p) - Var(p)| \geq \alpha \cdot Mean(p) \tag{6}$$

where $Mean(p)$ and $Var(p)$ calculate the mean intensity and the variance, respectively, and $\alpha \in (0, 1)$. If the constraint is satisfied, p will be regarded as a valid patch and the estimated blur-kernel k' from p is added to the set K'.

We extract 5 patches from each bicubic-upscaled LR image x'. We set the patch size $d = 512$ and $\alpha = 0.003$.

3.2.2 Kernel Modeling with GAN

In practice, input LR images may be hard to obtain and limited to a few camera models. In addition, the kernel-estimation algorithm [35] is computationally expensive. As such, the quantity and diversity of kernels collected in the last subsection may be limited, and the results of training a deep CNN only with these kernels will not suffice. We thus propose to model the blur-kernel distribution over the estimated kernel set K', and to generate a larger blur-kernel pool K^+ that contains more examples of realistic blur-kernels with more diversity. We use a GAN to generate such realistic blur-kernels.

We use WGAN-GP [15], which is an improved version of WGAN [1], for the objective function of our GAN:

$$L = \mathop{\mathbb{E}}_{\tilde{f} \sim \mathbb{P}_g}[D(\tilde{f})] - \mathop{\mathbb{E}}_{f \sim \mathbb{P}_r}[D(f)]] + \lambda \mathop{\mathbb{E}}_{\hat{f} \sim \mathbb{P}_{\hat{f}}}[(\left\|\nabla D(\hat{f})\right\|_2 - 1)^2] \tag{7}$$

where D is the discriminative network, $\mathbb{P}_r$ is the distribution over K', and $\mathbb{P}_g$ is the generator distribution. $\mathbb{P}_{\hat{f}}$ is defined as a distribution sampling uniformly along straight lines between pairs of points sampled from $\mathbb{P}_r$ and $\mathbb{P}_g$. f, $\tilde{f}$, $\hat{f}$ are the random samples following the distribution $\mathbb{P}_r$, $\mathbb{P}_g$ and $\mathbb{P}_{\hat{f}}$, respectively. For more details, see [15].

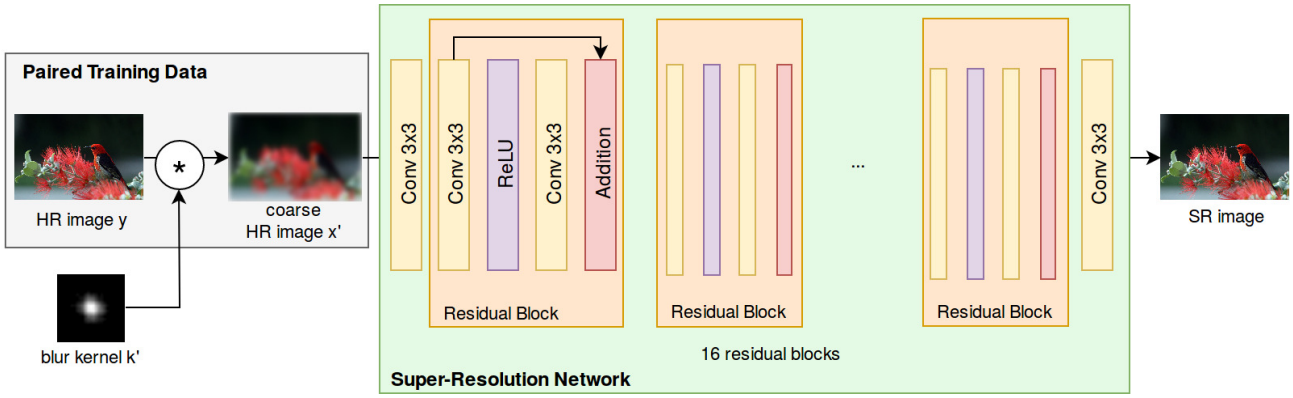


Figure 2: The Convolutional Neural Network architecture of KMSR. We convolve the HR image y with a blur-kernel k' randomly chosen from the blur-kernel pool K^+ to generate the coarse HR image x'. The other units each have 64 filters except for the last unit, where the filter number is equal to the number of output channels.

We adopt a similar network architecture to DCGAN [37]. The generative network G takes $z \sim N(0, 1)$, a vector of length 100 and generates a blur-kernel sample. It contains 4 fractionally-strided convolutions [10] of filter size 4×4, with batch normalization [23], ReLU [33], and a final convolution layer of filter size 8×8. The filter number of G from the second to the last unit is 1025, 512, 256, 1, respectively. The discriminative network D takes a kernel sample as input and identifies if it is fake, it contains 3 convolution layers with instance batch-normalization [46] and leaky ReLU [48]. The filter number of D from the first to the third unit is 256, 512, 1024, respectively.

The trained GAN model G is used to generate blur-kernel samples for augmenting K' until the final kernel pool $K^+ = K' \cup \{G(z_1), G(z_2), G(z_3), \dots\}$ is obtained. Like the normalization of kernels in [35], we apply sum-to-one and non-negative constraints on the generated kernels.

3.3. Super-Resolution with CNN

Previous approaches [8, 17, 31] propose to solve the SR problem by training a CNN with large datasets, and these methods have achieved impressive results on synthetic data. Deep neural networks implicitly learn the latent model from the paired training dataset, and thus do not require explicit knowledge of image priors. Hence, we utilize a CNN in our SR framework.

We create the training dataset in the following manner: the HR images are divided into small patches of size $m \times m$, which form the set $Y = \{y_1, y_2, \dots, y_t\}$. Blur-kernels in K^+ obtained in Section 3.2.1 are randomly chosen to convolve with patches in Y to obtain $X' = \{x'_1, x'_2, \dots, x'_t\}$, where $x'_j = y_j * k'_l$. The sets X' and Y form a paired training dataset $\{X', Y\}$.

The network structure of the CNN, which consists of 16 residual blocks [20], is illustrated in Fig. 2. Zero padding is adopted to ensure consistent input and output dimension. The objective function of our network is L1 enabling the network to obtain better performance [56].

4. Experiments

4.1. Implementation Details

We utilize the DPED [22] images to build the realistic blur-kernel set K'. DPED [22] is a large-scale dataset that consists of over 22K real photos captured with 3 different low-end phone models. We separate the dataset into two parts, *DPED-training* and *DPED-testing*, according to the camera models. *DPED-training* consists of photos taken with the Blackberry Passport and Sony Xperia Z, and serves as reference real-photography LR set for extracting the realistic blur-kernels k'_e in Sec. 3.2.1. *DPED-testing* consists of photos captured with the iPhone3GS, and is used as a validation dataset. We collect 1000 realistic blur-kernels $K' = \{k'_1, k'_2, \dots, k'_{1000}\}$ from *DPED-training* by using the kernel estimation codes from [35]. We use these kernels in the training of the kernel modeling GAN G. We set the batch size as 32 and $\lambda = 10$ for the loss function (see Eqn. 5). G is trained for 20,000 epochs. The extended blur-kernel pool K^+ is obtained by generating 1,000 kernels using the trained G and adding them to K'.

We use the training set of DIV2K [44] as HR images, from which we extract patches of size 128×128. We build the paired dataset $\{X', Y\}$ during training of the SR network: in each epoch, each HR patch is convolved with a kernel k' randomly chosen from K^+ to obtain a coarse HR patch. We train our SR network with ADAM optimizer [27]. We set the batch size to 32. The learning rate is initialized as 10^{-4} and is halved at every 10 epochs. The source code is publicly available online[1].

[1]https://github.com/IVRL/Kernel-Modeling-Super-Resolution

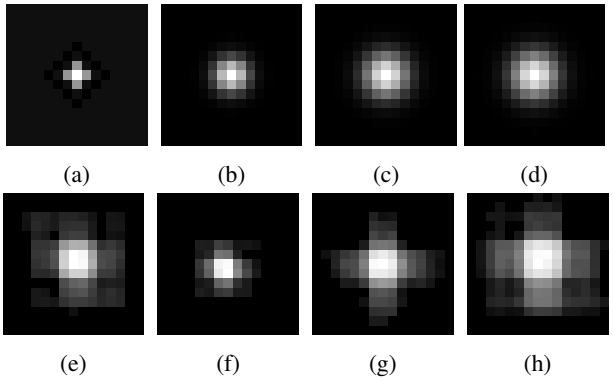


Figure 3: Visualization of different blur-kernels for scale $s = 2$ ($\times 2$ SR). To better visualize the kernels, we only show a 15×15 patch cropped from the center. (a) the bicubic kernel [25] with anti-aliasing implemented in Matlab [13]; (b), (c) and (d) three isotropic Gaussian kernels $g_{1.25}$, $g_{1.6}$ and $g_{1.7}$, respectively, which are widely used in $\times 2$ SR [9, 18, 58]. (e), (f) two kernel samples k'_e estimated from real photos, (g) and (h) two blur-kernels $G(z_i)$ generated with the KMSR GAN.

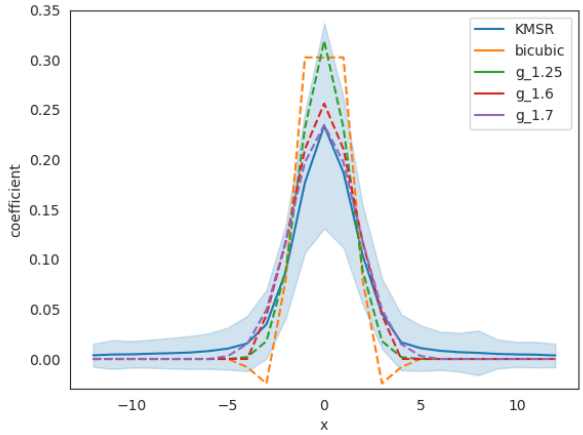


Figure 4: Plot of different blur-kernels. The solid line shows the mean kernel shape from the blur-kernel pool K^+ generated with KMSR. The shadow area illustrates the variance. The dashed lines show the shape of the bicubic kernel [25] and three Gaussian kernels that are commonly used in synthesizing LR images [9, 18, 58].

4.2. Estimated Kernels

We first study the distributions of blur-kernels. We show examples of kernels k'_e generated with KMSR in Fig 3 and Fig. 4. We also visualize the Matlab bicubic-kernel and three isotropic Gaussian kernels g_{sigma} with sigmas ($g_{1.25}$, $g_{1.6}$ and $g_{1.7}$) that are commonly used to synthesize LR images in $\times 2$ SR. Note that the bicubic-kernel is band-pass compared to the low-pass shape of the other kernels. The bicubic-kernel is designed to keep the sharpness of the image and to avoid aliasing during the down-sampling operation [25]. As stated in [11], the bicubic-kernel is *not* a proper approximation of the real blur-kernel in image acquisition, as camera-blur is low-pass and often attenuates the high-frequency information of the scene more. In Fig. 4, also notice that the kernels generated by by KMSR encompass a wide range of distributions, including the Gaussian kernels that are a better approximation of the real camera-blur [34] than the bicubic-kernel. KMSR is thus able to generate very diverse coarse HR images.

4.3. Experiments on Bicubic and Gaussian Blur-Kernels

In this section, we evaluate KMSR and other CNN-based SR networks on synthetic LR images by applying different blur-kernels to the validation set of the DIV2K [44] dataset.

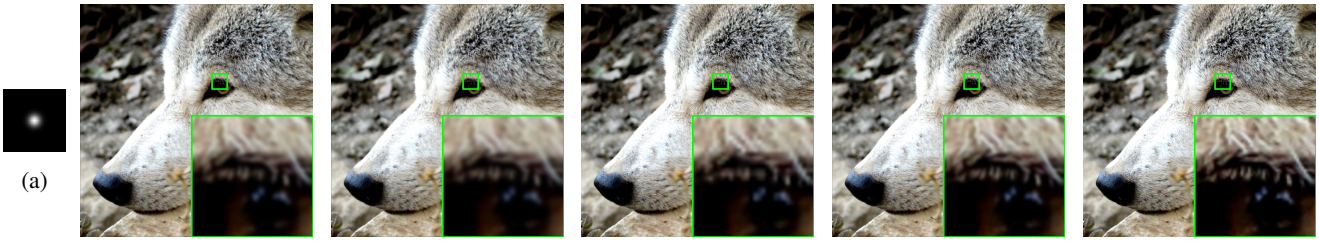
We test on two upscaling factors, $s = 2$ ($\times 2$ SR) and $s = 4$ ($\times 4$ SR) and on four synthetic LR datasets that are generated using four different kernels on the DIV2K [44] validation set. We include the anti-aliasing bicubic kernel, as it is used by many algorithms even though it is not a physically feasible camera-blur for real images [11]. We also test on 3 isotropic Gaussian kernels, $g_{1.25}$ [58], $g_{1.6}$ [9] and $g_{1.7}$ [18]; they are commonly used as blur-kernels in the generation of synthetic LR images [34]. The four kernels are visualized in the first row of Fig. 3.

We compare our proposed KMSR with the state-of-the-art CNN-based SR methods: SRCNN [8] (we use the 9-5-5 model), VDSR [26], EDSR [31] and DBPN [17]. We use the published codes and models from the respective authors. Note that these four networks are trained using only the bicubic-kernel in the generation of corresponding LR images from HR images.

The quantitative results of the different SR networks on the different LR datasets are provided in Table 1. Although KMSR produces worse results on LR images generated with the bicubic kernel, it outperforms all other networks on all other experimental settings on both upscaling factors $s = 2$ and $s = 4$. We can also observe that the performance of SR networks that are trained using only bicubic LR images is limited when the bicubic kernel deviates from the true blur-kernel. These networks gain less than 0.4dB improvement in PSNR compared to simple bicubic interpolation (column 3 in Table 1). Even with deeper layers, EDSR [31] and DBPN [17] do not outperform shallow networks SRCNN [8] and VDSR [26]. By modeling realistic kernels, our KMSR outperforms them all by up to 1.91dB. A visual comparison using $g_{1.6}$ as blur-kernel and $s = 2$ as upscaling factor is given in Fig. 5. Note that KMSR produces results that visually appear sharper than other methods, as it is trained using more realistic blur-kernels.

Blur-Kernel	Scale	Bicubic	SRCNN [8]	VDSR [26]	EDSR [31]	DBPN [17]	KMSR
bicubic	×2	29.94	31.89	32.63	33.58	33.84	33.52
$g_{1.25}$		26.14	26.56	26.54	26.58	26.60	27.94
$g_{1.6}$		25.49	25.72	25.72	25.69	25.70	27.63
$g_{1.7}$		25.11	25.30	25.34	25.28	25.28	27.15
bicubic	×4	26.28	27.89	28.04	28.95	29.03	27.99
$g_{2.3}$		24.71	24.83	24.91	25.10	25.18	26.14
$g_{2.5}$		24.34	24.30	24.34	24.39	24.42	25.64
$g_{2.7}$		24.11	24.14	24.05	24.27	24.23	25.33

Table 1: Comparison on DIV2K [44] in terms of PSNR in the evaluation of bicubic and Gaussian blur-kernels. We highlight the best results in red color and the second best in blue color. Note that our proposed KMSR outperforms other state-of-the-art SR networks by up to 1.91dB on Gaussian kernels.



(a) (b) SRCNN [8] (c) VDSR [26] (d) EDSR [31] (e) DBPN [17] (f) **KMSR**

Figure 5: Qualitative comparison of ×2 SR on image 0805 from DIV2K [44], using a Gaussian blur-kernel $g_{1.6}$ (a) as the blur-kernel and $s = 2$ as upscaling factor. Visual results of ×4 SR are in the supplementary material.

4.4. Experiments on Realistic Kernels

To validate the capability of the proposed KMSR on images with real unknown kernels, we conduct experiments on synthesizing LR images with unseen realistic blur-kernels on ×2 and ×4 SR. We collect 100 blur-kernels from the LR images in the *DEPD-testing* dataset (i.e., the iPhone3GS images), which is not seen in the training of KMSR. We then apply these blur kernels to generate coarse HR images using the DIV2K [44] validation set. Table 2 shows the resulting PSNR and SSIM of the different SR networks. As before, the performance of the SR networks trained using only the bicubic-kernel is limited on these images. This highlights the sensitivity of CNN-based SR networks to wrong kernels in the creation of the training dataset. Blur-kernel modeling is a promising venue for improving SR networks if the algorithm is to be applied to real camera data.

We present qualitative results in Fig. 6. KMSR successfully reconstructs the detailed textures and edges in the HR images and produces better outputs.

Method	Scale	PSNR	SSIM
bicubic interpolation	×2	25.06	0.72
SRCNN [8]		25.30	0.74
VDSR [26]		25.29	0.74
EDSR [31]		25.28	0.74
DBPN [17]		25.30	0.75
KMSR		27.52	0.79
bicubic interpolation	×4	23.32	0.69
SRCNN [8]		23.42	0.69
VDSR [26]		23.39	0.69
EDSR [31]		23.49	0.69
DBPN [17]		23.51	0.70
KMSR		25.13	0.74

Table 2: Comparison on DIV2K [44] in the evaluation of realistic blur-kernels estimated from *DEPD-testing*. We highlight the best results in red color and the second best in blue color.

4.5. Experiments on Real Photographs

We also conduct ×2 SR experiments on real photographs. Fig. 7 illustrates the KMSR output on one photograph captured by the iPhone3GS in the *DEPD-testing* dataset. Perceptual-driven SR methods usually recover more detailed textures and achieve better visual quality than previous SR networks. In addition to the four SR methods we compare to, we also show the output from the perceptually-optimized SR network ESRGAN [47]. It is noticeable that the networks trained using only the bicubic-downsampled LR images tend to produce overly smooth

(a) (b) SRCNN [8] (c) VDSR [26] (d) EDSR [31] (e) DBPN [17] (f) **KMSR**

Figure 6: Qualitative comparison on $\times 2$ SR on image 0847 from DIV2K [44], using a realistic blur-kernel (a) estimated from *DPED-testing*. Visual results of $\times 4$ SR are in the supplementary material.

(a) VDSR [26] (b) EDSR [31] (c) DBPN [17] (d) ESRGAN [47] (e) **KMSR**

Figure 7: $\times 2$ SR qualitative comparison of different SR networks on image 83 from *DPED-testing*. Better viewed on screen.

	EDSR [31]	DBPN [17]	KMSR
#preference	3	1	44

Table 3: Results of the psychovisual experiment. #preference shows the number of SR results from the specific method that are chosen as "the clearest and the sharpest image" by more than 50% of the participants. For 44 out of 50 images, results from our KMSR are favored over the other two methods.

images, whereas KMSR can recover a sharp image with better details.

As there are no reference HR images for this experiment, we validate our methods with a psychovisual experiment on a crowd-sourcing website[2]. We only compare to EDSR [31] and DBPN [17] as they are the state-of-the-art CNN-based SR networks. Note that because of the resolution limitations of display devices, we could not show full-resolution images. We randomly select 50 images from *DEPD-testing* and crop patches of size 500×500 from each image. For each patch, we show the participants the SR results from EDSR [31], DBPN [17] and our KMSR. We ask them to choose the clearest and the sharpest image among them[3]. To avoid bias, the order of the three SR images are randomly shuffled. In total, 35 users participated in the experiment with each of them labeling all 50 images. The results of the psychovisual experiment are in Table 3. For 44 of the 50 images, the output from KMSR are preferred over the two other methods, which suggests that KMSR is able to produce visually better results than the other two SR networks.

4.6. Experiments on Zoom-in Super-Resolution

To further verify the performance of the proposed KMSR, we conduct experiments on images captured with the same camera, but different focal lengths. We use a 24-70mm zoom lens to capture photo-pairs. The 35mm focal length photo serves as LR image, and the photo taken at the same position with the 70mm focal length serves as the reference HR image for a $\times 2$ SR of the LR image. We capture all the photos with a small aperture (f/22) to minimize the depth-of-field differences. We crop patches of size 250×250 from the LR image and patches of size 500×500 from the reference HR image. To align the patches, we do a grid search for horizontal and vertical alignments, then we apply the different SR networks on the LR patch.

[2] www.clickworker.com

[3] Experiment webpage: https://ivrlwww.epfl.ch/ruofan/exp/index.html

(a) SRCNN [8] (b) VDSR [26] (c) EDSR [31] (d) DBPN [17] (e) **KMSR** (f) reference

Figure 8: Qualitative comparison of different SR networks on $\times 2$ zoom-in. (a)-(e) The SR results on the LR image taken with a 35mm focal length. (f) the reference HR image taken with a 70mm focal length. More examples are shown in the supplementary material.

Method	**PSNR**	**SSIM**
bicubic interpolation	26.93	0.79
SRCNN [8]	27.07	0.80
VDSR [26]	27.11	0.80
EDSR [31]	27.45	0.81
DBPN [17]	27.42	0.81
KMSR	29.13	0.84

Table 4: Average PSNR and SSIM of different SR networks on the $\times 2$ zoom-in dataset. The evaluation is performed only on the luminance channel to alleviate the effect of bias caused by the color variations of the two images. We highlight the best results in red color and the second best in blue color.

Table 4 shows the results of different SR networks on this zoom-in task. KMSR outperforms all other SR networks by a large margin both in PSNR and SSIM. A visual result is shown in Fig. 8. KMSR is capable of generating a sharper image than the other SR networks.

4.7. Ablation studies

To demonstrate the effectiveness of using realistic kernels and also to show the precision of the kernel estimation algorithm [35] that we use, we train and test another version of the proposed network, KMSR$_{A1}$, without collecting the realistic kernels. In building the kernel pool K'_{A1} for KMSR$_{A1}$, we use the bicubic-downsampled HR images as LR images, i.e. we estimate the blur kernels k'_{A1} on the bicubic-downsampled, bicubic-upscaled coarse HR images X'_{A1}. We then follow the same procedure as KMSR. We train a GAN on K'_{A1} and generate the larger kernel pool K^+_{A1} used to train KMSR$_{A1}$. We test KMSR$_{A1}$ on different experimental settings, the quantitative results are shown in Table 5. For the Gaussian and realistic kernels, KMSR$_{A1}$ achieves comparative results with the state-of-the-art SR networks (see Table 1), which implies that KMSR$_{A1}$ is capable of learning the mapping from bicubic-downsampled LR images to HR images. The results also shows that we achieve significant performance gains with KMSR that is trained with the realistic kernels of K^+ (last column in Table 1).

Blur-Kernel	KMSR$_{A1}$	KMSR$_{A2}$	KMSR
bicubic	33.66	33.28	33.52
$g_{1.25}$	26.47	27.42	27.94
$g_{1.6}$	25.62	27.02	27.63
$g_{1.7}$	25.28	27.90	27.15
realistic	25.29	27.10	27.52

Table 5: Evaluation of KMSR on $\times 2$ SR in different training setting.

To test the contribution of the GAN in improving generalization, we trained KMSR$_{A2}$, which is KMSR without using the GAN but with simple data augmentation to expand the kernel pool. In this case, KMSR$_{A2}$ is only trained on K'_{A2} which contains the original estimated kernels k' and their rotated, flipped, and scaled versions. Results are shown in Table 5 On average, KMSR obtains 0.5dB improvements on KMSR$_{A2}$. leading us to believe that using a GAN to augment the kernel pool results in a more diverse representation than simple data augmentation. This further validates the effectiveness of incorporating a GAN in order to augment the realistic kernel-pool.

5. Conclusion

We improve the performance of CNN-based SR networks on real LR images by modeling realistic blur-kernels. In contrast to existing methods that use a bicubic-kernel in the imaging model to obtain LR training images, we generate the SR training dataset by employing a set of realistic blur-kernels estimated from real photographs. We further augment the blur-kernel pool by training a GAN to output additional realistic kernels. Our KMSR is able to produce visually plausible HR images, demonstrated by both quantitative metrics, qualitative comparisons, and a psychovisual experiment. KMSR offers a feasible solution toward practical CNN-based SR on real photographs.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, 2017.

[2] Christopher Bowles, Liang Chen, Ricardo Guerrero, Paul Bentley, Roger Gunn, Alexander Hammers, David Alexander Dickie, Maria Valdés Hernández, Joanna Wardlaw, and Daniel Rueckert. GAN augmentation: augmenting training data using generative adversarial networks. *arXiv preprint arXiv:1810.10863*, 2018.

[3] Adrian Bulat, Jing Yang, and Georgios Tzimiropoulos. To learn image super-resolution, use a gan to learn how to do image degradation first. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 185–200, 2018.

[4] Jian-Feng Cai, Hui Ji, Chaoqiang Liu, and Zuowei Shen. Framelet-based blind motion deblurring from a single image. *IEEE Transactions on Image Processing*, 21(2):562–572, 2012.

[5] Chang Chen, Zhiwei Xiong, Xinmei Tian, Zha Zheng-Jun, and Feng Wu. Camera lens super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[6] Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang. Image blind denoising with generative adversarial network based noise modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3155–3164, 2018.

[7] Sunghyun Cho and Seungyong Lee. Fast motion deblurring. *ACM Transactions on Graphics*, 28(5):145, 2009.

[8] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016.

[9] Weisheng Dong, Lei Zhang, Guangming Shi, and Xin Li. Nonlocally centralized sparse representation for image restoration. *IEEE Transactions on Image Processing*, 22(4):1620–1630, 2013.

[10] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

[11] Netalee Efrat, Daniel Glasner, Alexander Apartsin, Boaz Nadler, and Anat Levin. Accurate blur models vs. image priors in single image super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2832–2839, 2013.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[13] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[14] Jinjin Gu, Hannan Lu, Wangmeng Zuo, and Chao Dong. Blind super-resolution with iterative kernel correction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1604–1613, 2019.

[15] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville Courville. Improved training of wasserstein GANs. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[16] Yoav Hacohen, Eli Shechtman, and Dani Lischinski. Deblurring by example using dense correspondence. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2384–2391, 2013.

[17] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1664–1673, 2018.

[18] Hu He and Lisimachos P. Kondi. A regularization framework for joint blur estimation and super-resolution of video sequences. In *IEEE International Conference on Image Processing*, volume 3, pages III–329. IEEE, 2005.

[19] Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2341–2353, 2011.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[21] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015.

[22] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. Dslr-quality photos on mobile devices with deep convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3277–3285, 2017.

[23] Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR.

[24] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[25] Robert Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(6):1153–1160, 1981.

[26] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, 2016.

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[28] Rolf Köhler, Michael Hirsch, Betty Mohler, Bernhard Schölkopf, and Stefan Harmeling. Recording and playback of camera shake: benchmarking blind deconvolution with a real-world database. In *Proceedings of the European Conference on Computer Vision*, pages 27–40. Springer, 2012.

[29] Wei-Sheng Lai, Jia-Bin Huang, Zhe Hu, Narendra Ahuja, and Ming-Hsuan Yang. A comparative study for single image blind deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1709, 2016.

[30] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4681–4690, 2017.

[31] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 136–144, 2017.

[32] Tomer Michaeli and Michal Irani. Nonparametric blind super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 945–952, 2013.

[33] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, pages 807–814, 2010.

[34] Kamal Nasrollahi and Thomas B. Moeslund. Super-resolution: a comprehensive survey. *Machine Vision and Applications*, 25(6):1423–1468, 2014.

[35] Jinshan Pan, Deqing Sun, Hanspeter Pfister, and Ming-Hsuan Yang. Blind image deblurring using dark channel prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1628–1636, 2016.

[36] Clément Peyrard, Moez Baccouche, and Christophe Garcia. Blind super-resolution with deep convolutional neural networks. In *International Conference on Artificial Neural Networks*, pages 161–169. Springer, 2016.

[37] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. 2016.

[38] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, 2016.

[39] Mehdi SM Sajjadi, Bernhard Scholkopf, and Michael Hirsch. Enhancenet: Single image super-resolution through automated texture synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4491–4500, 2017.

[40] Jordi Salvador and Eduardo Perez-Pellitero. Naive bayes super-resolution forest. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 325–333, 2015.

[41] Wen-Ze Shao and Michael Elad. Simple, accurate, and robust nonparametric blind super-resolution. In *International Conference on Image and Graphics*, pages 333–348. Springer, 2015.

[42] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016.

[43] Assaf Shocher, Nadav Cohen, and Michal Irani. zero-shot super-resolution using deep internal learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3118–3126, 2018.

[44] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, Lei Zhang, Bee Lim, et al. NTIRE 2017 challenge on single image super-resolution: Methods and results. In *The IEEE Conference on Computer Vision and Pattern Recognition Workshops*, July 2017.

[45] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: adjusted anchored neighborhood regression for fast super-resolution. In *Proceedings of the Asian Conference on Computer Vision*, pages 111–126. Springer, 2014.

[46] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: the missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[47] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. ESRGAN: enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision*, pages 63–79. Springer, 2018.

[48] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[49] Jianchao Yang, Zhaowen Wang, Zhe Lin, Scott Cohen, and Thomas Huang. Coupled dictionary training for image super-resolution. *IEEE Transactions on Image Processing*, 21(8):3467–3478, 2012.

[50] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. Stack-GAN: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5907–5915, 2017.

[51] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep CNN denoiser prior for image restoration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3929–3938, 2017.

[52] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Learning a single convolutional super-resolution network for multiple degradations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3262–3271, 2018.

[53] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Deep plug-and-play super-resolution for arbitrary blur kernels. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[54] Xuaner Zhang, Qifeng Chen, Ren Ng, and Vladlen Koltun. Zoom to learn, learn to zoom. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[55] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision*, pages 286–301, 2018.

[56] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging*, 3(1):47–57, 2017.

[57] Xiaole Zhao, Yadong Wu, Jinsha Tian, and Hongying Zhang. Single image super-resolution via blind blurring estimation and anchored space mapping. *Computational Visual Media*, 2(1):71–85, 2016.

[58] Xiaole Zhao, Yadong Wu, Jinsha Tian, and Hongying Zhang. Single image super-resolution via blind blurring estimation and anchored space mapping. *Computational Visual Media*, 2(1):71–85, 2016.

[59] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017.