KernelNet: A Blind Super-Resolution Kernel Estimation Network

Mehmet Yamac Baran Ataman Aakif Nawaz
Huawei Technologies Oy (Finland) Co. Ltd
{mehmet.yamac, baran.ataman, aakif.nawaz}@huawei.com

Abstract

Recently developed deep neural network methods have achieved remarkable performance in the Super Resolution problem when applied to Low Resolution (LR) images that are obtained from High Resolution (HR) images with ideal and predefined downsampling processing, i.e., convolution with a known blurring kernel that is followed by subsampling (e.g., Bicubic). However, when these algorithms are applied to real-world images whose downsampling pattern is unknown, unlike synthetically generated LR-HR image pairs, their performance drops drastically. Blind SR problem can be defined as real-world image SR when the downsampling blurring kernel (SR kernel) is unknown. The recent SR kernel estimation techniques like KernelGAN have shown promising results in this direction. However, their limited recovery performance and high computational complexity make them unsuitable for real-time usage, like for applications in mobile cameras. This paper proposes a modular and interpretable neural network structure, KernelNet, for the blind SR kernel estimation problem. The proposed model outperforms the state-of-the-art SR kernel estimator, KernelGAN, by a significant margin in SR kernel reconstruction accuracy. Moreover, to the best of our knowledge, the proposed algorithm is the first one that can estimate the SR kernel in real-time by performing O(1k) times faster than KernelGAN.

1. Introduction

Single image super-resolution, or SR task for short, involves generating a higher resolution image from the given low resolution (LR) one by enhancing the natural high-frequency details [4]. SR is a challenging ill-posed inverse problem, and a prior about desired HR solution (a solution in natural HR image space) is needed in order to avoid having infinitely many solutions [23]. The prior information can be satisfied via a predefined model-based approach [28, 34, 35] or directly learned from the training data.

Recent Convolutional Neural Network (CNN) based methods have shown the state-of-the-art performance on mapping from LR image to HR one [32, 25, 5, 20]. Most of these CNN-based methods have been trained with LR images that are degraded by convolution with a fixed blurring kernel (SR kernel, e.g., bicubic or Gaussian) followed by subsampling. They are also tested with similarly produced synthetic data; this is how they achieved the state-of-the-art result. However, their performances come short when it comes to real-world images whose degradation operation may be more complex. As an example, it is easily observable that one of the state-of-the-art CNN-based SR approaches, enhanced deep super-resolution network (EDSR) [25], does not show any significant improvement compared to the simple binary cubic interpolation [18] in the real world view as shown in Figure. 1.

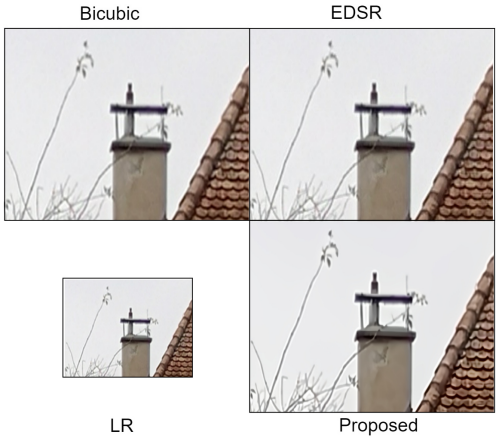

Figure 1. Comparison of the simple bicubic interpolation, one of the state of the art non-blind SR model, EDSR, and KernelNet together with a SR neural network model [40] in a real image from Huawei P20 camera.

Unlike synthetically generated LR images with a fixed SR kernel followed by decimation, when the SR kernel is unknown and possibly more complex, the SR problem turns into the blind SR problem. The literature that solely targets the blind SR problem is relatively short compared to extensive studies on the aforementioned non-blind SR approaches. The authors of [6], for the first time, argued that

an accurate prediction of the SR kernel is more important than any a priori about the image. Although there are several recent studies [39, 8] making HR to LR degradation unsupervised manner using a generative adversarial network (GAN) [9], in this paper, we focus on the works that consider "SR kernel" either in the preparation of the realistic training samples or use them explicitly during SR process. Among the SR methods that use GAN-based degradation to generate realistic training data, ESRGAN-FS [8] gives superior performance and is the winner of the "Real World SR Challenge (AIM 2019)" [27]. Having the generated training dataset, the authors of [8] also use another GAN-based method, ESRGAN [36] for SR. They also separate the low and high-frequency information and handle them separately.

Recent works [17, 44] use pre-estimated SR kernels or post-processed versions of them, e. g., data augmentation via generative methods, to obtain LR-HR pairs to be used in training. This way, they try to increase their CNN-based SR networks' generalization capabilities to achieve real-world SR. On the other hand, a relative but still non-blind direction is to create solutions that are able to work under any arbitrary SR kernels [40, 41]. These algorithms take GT or estimated kernels and noise level as inputs as well as LR image; then, they spit out an HR estimation. Whatever the research direction is, which can be preparing a training set with training samples degraded with a generalizable kernel pool or using an algorithm that can handle multiple degradations, estimating SR kernels accurately is still an open research question. The authors of [14] worked on parametric Gaussian models for SR kernel estimation. Later, the authors of [29] proposed a non-parametric solution to estimate the SR kernel. Few works first apply bicubic interpolation to LR image then estimate an approximate SR kernel using an existing blind deblurring algorithm [44]. Recently, the authors of [10] proposed an Iterative Kernel Correction (IKC) scheme that iteratively refines the SR kernel estimation and HR recovery. However, their analysis is restricted to isotropic Gaussian SR kernels with different variances. Moreover, the algorithm recovers the PCA coefficients of the SR kernels instead of SR kernels. Recently proposed Deep Alternating Network (DAN) [15] was also reported to have improved performance over IKC but still estimates PCA coefficients rather than SR kernels. On the other hand, the most recent work that purely focuses on SR kernel estimation is Blind Super-Resolution Kernel Estimation using an Internal-GAN (KernelGAN) [3]. Kernel GAN is an Internal-GAN [33] type network that produces downscaled versions of test images in training to learn image-specific SR kernels. Although KernelGAN provided state of the art result in SR kernel estimation, their SR kernel recovery performances are still limited, and there is still room for significant improvement in terms of kernel reconstruction accuracy. On the other hand, since they use an image-specific training for their CNN-based SR solution, it is not feasible to use it in real-time applications, such as SR on a mobile device.

In this work, we introduce a real-time SR kernel estimator network, KernelNet, that surpasses the state-of-the-art SR kernel estimator, KernelGAN, by a significant gap in terms of both SR kernel reconstruction accuracy and computational complexity. Compared to model-based or internal GAN-based solutions, KernelNet provides a non-iterative solution, making it feasible to be used in real-time SR solution when plugged in an SR solution that can work in arbitrary any SR kernel [40]. The other merit of the proposed neural network model is its explainable and efficient design in which each module's responsibility is easily explainable either mathematically or empirically.

2. Problem Definition and Related Work

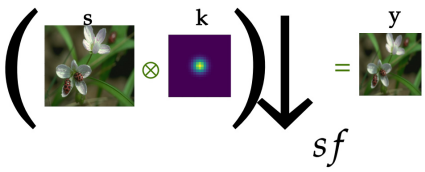

Figure 2. A pictorial representation of the realistic SR degradation process with an example SR kernel, $\mathbf{k}$, and a HR image, $\mathbf{s}$.

In the literature [6, 2, 7], the conventional single image SR problem is defined as blurring with an SR kernel than followed by a decimation operation. Mathematically speaking, let $\mathbf{s}$ be our unknown HR image (the latent image) and $\mathbf{k}$ be the unknown SR kernel, then the degradation from HR to LR with a scale factor sf can be defined via

$$\mathbf{y} = (\mathbf{s} \otimes \mathbf{k})_{\downarrow_{sf}} + \mathbf{n}, \tag{1}$$

where $\mathbf{y}$ is the observation, $\downarrow_{sf}$ standard decimation operation with scale factor sf, $\mathbf{k}$ is the SR kernel and $\mathbf{n}$ is the possible additive noise (e.g., AWGN). A pictorial representation of the defined degradation process with an example of $(\mathbf{s}, \mathbf{k}, \mathbf{y})$ triple can be shown in Figure. 2. Unless otherwise stated, throughout this article, we assume that the degradation model is noise-free, e.g., $\mathbf{n} = 0$. Although there are a few other aforementioned model-based solutions that directly deal with SR kernel estimation, their source code is not available, and KernelGAN has made a significant gap in terms of both reconstruction accuracy and computational time these methods [3]. Therefore, in addition to KernelGAN, we have also used a model-based SR kernel estimation technique [44], which has recently been proposed to obtain the realistic kernel pool and compare the performance of our proposed solution, KernelNet. In [44], they

Figure 3. Model based approximate SR kernel estimation.

first obtain a coarse estimation of the HR image and then obtain an approximate estimation of SR kernel using an existing blind deblurring algorithm. In the sequel, we will explain these two types of strategies in more detail.

2.1. Model Based Approach

The degradation model in a deblurring problem can be defined as

$$\mathbf{y_b} = \mathbf{s} \otimes \mathbf{k_b} + \mathbf{n}, \tag{2}$$

which is slightly different from the SR degradation model in Eq. (1) i.e., there is no decimation process [19]. Compared to blind SR, the blind deblurring problem is well studied in the literature [21, 11, 12]. Most of the literature works are model-based and iteratively estimate both blur kernel and latent images in an alternating manner [11, 12, 31]. A model-based deblurring problem can be defined as

$$\min_{\mathbf{k_b},\mathbf{s}} \Phi\left(\mathbf{k_b}, \mathbf{s}\right) \text{ subject to } \Upsilon\left(\mathbf{y_b}\right) \tag{3}$$

where $\min_{\mathbf{k_b},\mathbf{s}} \Phi\left(\mathbf{k_b}, \mathbf{s}\right)$ is the regularization function and $\Upsilon\left(\mathbf{y_b}\right) = \{(\mathbf{s}, \mathbf{k_b}) : \mathbf{s} \otimes \mathbf{k_b} = \mathbf{y_b}\}$ in the noise free case. In the case of observation is also corrupted by an additive noise, the optimization problem can be relaxed to $\Upsilon\left(\mathbf{y_b}\right) = \left\{(\mathbf{s}, \mathbf{k_b}) : \|\mathbf{s} \otimes \mathbf{k_b} - \mathbf{y_b}\|_p \leq \epsilon\right\}$ where ϵ is a small enough constant and $\|.\|_p$ is ℓ_p-norm, e.g., ℓ_2-norm.

The recent study [44] has proposed the idea of estimating realistic SR kernels by first using bicubic interpolation to observation then leveraging a state-of-the-art model-based blurring technique. Mathematically speaking, they first apply bicubic interpolation by scale factor sf to LR observation, $\mathbf{y}$ to obtain a coarse estimation of HR image, i.e.,

$$\mathbf{y_b} = \mathbf{y} \otimes \mathbf{b_{sf}}, \tag{4}$$

where $\mathbf{b_{sf}}$ is the bicubic interpolation kernel with scale factor sf. Therefore, the relationship between the coarse estimation of the latent image, $\mathbf{y_b}$, and ground truth HR image, $\mathbf{s}$, can be formulated via

$$\mathbf{y_b} = (\mathbf{s} \otimes \mathbf{k})_{\downarrow_{sf}} \otimes \mathbf{b_{sf}} = \mathbf{s} \otimes \mathbf{k}', \tag{5}$$

where $\mathbf{k}' = (\mathbf{k} \otimes \mathbf{b_{sf}})_{\downarrow_{sf}}$. They used the model [30], which jointly estimate the blurring kernel and the latent image using dark channel prior [13],

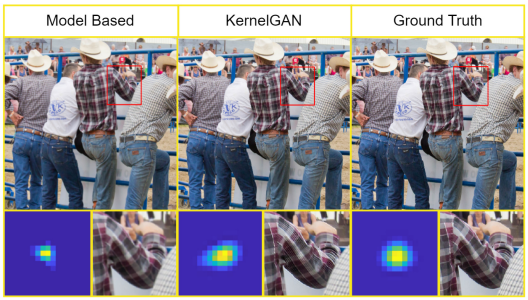

Figure 4. Kernel mismatch problem. Inaccurate estimatation of of SR kernel leads to visible artifacts on HR estimation.

$$min_{\mathbf{s},\mathbf{k}'} \|\nabla\mathbf{s} \otimes \mathbf{k}' - \nabla\mathbf{y}\|_2 + \lambda_1 \|\mathbf{k}'\|_2^2 + \lambda_2 \|\nabla\mathbf{s}\|_0 + \lambda_3 \left\|\nabla\mathbf{s}^{dark}\right\|_0, \tag{6}$$

where $\mathbf{s}^{dark}$ is the dark channel of the image. The pictorial representation of this approximate SR kernel estimation can be seen in Figure. 3.

2.2. Kernel GAN

KernelGAN [3] is a recently proposed state of the art SR kernel estimation network. It is an Internal-GAN [33] technique that only uses the given test image in image-specific training and does not require additional training data. KernelGAN composes of one generator and one discriminator network. Given the test image to be upscaled, the generator generates a lower scale image by degrading and downscaling the test image. The discriminator tries to distinguish whether the generated LR image has the same patch distribution as the original one. The discriminator and generator are trained by using the crops from the test image in an alternating manner; the downscaled generator is trained to fool the discriminator in each iteration. After convergence, the generator can be used as the SR degradation model. The generator has five fully convolutional layers and one downsampled layer with scale factor sf. Therefore, the impulse response of the convolutional layers produces the SR kernel estimation.

2.3. Drawbacks of Current Solutions

2.3.1 Kernel Mismatch Problem

We designed a small experiment to demonstrate the importance of accurate SR kernel prediction. An arbitrarily selected GT HR image from the Div2k Dataset [1] was degraded with a previously known SR kernel as in Equation 1 to obtain the LR image. Later, the SR kernel was estimated from this LR image by using both KernelGan and the model-based scheme explained in Section 1. Together with these estimated kernels, the LR image was given to an

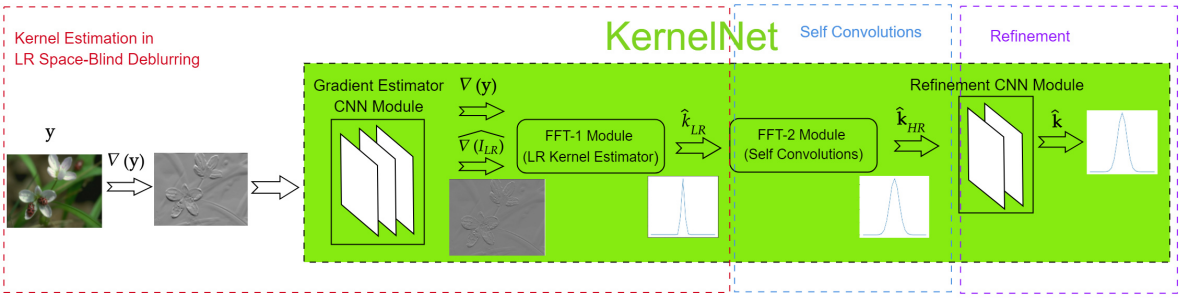

Figure 5. Proposed SR Kernel Estimator Network, KernelNet.

SR solution [40] that can work on any SR kernel. In Figure 4, we can see the kernels that were estimated by these two algorithms. Fig 4. reveals how sensitive the SR solution is to inaccuracies in the SR kernel estimation; inaccurate kernel estimation yields visible artifacts. Let the estimated kernels by KernelGAN and model-based are $\hat{\mathbf{k}}_{\mathbf{kerGan}}$ and $\hat{\mathbf{k}}_{\mathbf{dark}}$, respectively. With a closer look, the estimated kernel of the model-based scheme, $\hat{\mathbf{k}}_{\mathbf{dark}}$, causes over-smoothing because the width of $\hat{\mathbf{k}}_{\mathbf{dark}}$ is smaller than the one of GT kernel, $\mathbf{k}$. On the other hand, when it comes to kernelGAN estimation SR kernel, $\hat{\mathbf{k}}_{\mathbf{kerGan}}$, its width is larger than the width of GT, which results in oversharpening and ringing artifacts. By using inaccurate SR kernel in any SR solution, including training, the CNN based solutions with inaccurate kernels may cause a similar complication, which is known as the kernel mismatch problem [10].

2.3.2 Computational Complexity

As discussed, the state of the art SR kernel estimation technique, KernelGAN, and aforementioned model-based scheme are iterative algorithms like other existing ones [14, 29]. Therefore, the current SR kernel estimation algorithms are relatively slow and can not be used in a real-time solution together with an SR solution [40] that requires the SR kernel estimation. For example, the estimations of SR kernel by model-based solution and KernelGAN are shown in Figure 4, whose estimation process took approximately 1000 and 70 seconds, respectively.

3. Proposed Approach

Emprical Observation: As in [42], if we ease the SR degradation problem to the deblurring one in low scale as

$$\mathbf{y} = (\mathbf{s} \otimes \mathbf{k})_{\downarrow_{sf}} + \mathbf{n} = \mathbf{s}_{\downarrow_{sf}} \otimes \mathbf{k_{LR}} + \mathbf{n}, \qquad (7)$$

where $\mathbf{k_{LR}}$ is the effective blurring kernel in low scale (original scale of the observation, $\mathbf{y}$), then one can easily use an existing blind debluring algorithm to estimate $\mathbf{k_{LR}}$. Although working on low scale blurring kernel is easier as deblurring is a well-studied problem, unfortunately, obtaining $\mathbf{k}$ from $\mathbf{k_{LR}}$ is a non-trivial task. Many times, the literature has observed that the lower scale blurring kernel width is narrower than SR kernel [40]. We observe from the estimated realistic SR kernels that by self convolutions on the estimated $\mathbf{k_{LR}}$ we can achieve a coarse estimation of SR kernel, i.e., the coarse estimation kernel resembles ground-truth SR kernel in shape, and it has the same width with the one of the ground-truth. Particularly, we claim that the following coarse estimation can be refined for more accurate SR kernel estimation,

$$\widehat{\mathbf{k_{HR}}} = \overbrace{\widehat{\mathbf{k_{LR}}} \otimes \widehat{\mathbf{k_{LR}}} \otimes ... \otimes \widehat{\mathbf{k_{LR}}}}^{2^{sf}}. \qquad (8)$$

Since SR degradation is a linear process, one may expect that the lower scale blurring kernel's width will be inversely proportional to the SR kernel's width due to the decimation. In ideal case, if $\mathbf{k_{LR}}$ is the Gaussian kernel with variance σ^2_{LR}, then 2^{sf} times self convolutions yield also a Gaussian function with variance $2^{sf} \times \sigma^2_{LR}$. Therefore, if the width $\mathbf{k_{LR}}$ is w_{LR} then the width of $\mathbf{k_{HR}}$ will be $\sqrt{2^{sf}} \times w_{LR}$ which completes the informal proof for the ideal Gaussian SR kernel case. For instance, when the scale factor is 2, the width of SR will be $2 \times w_{LR}$; for the case of scale factor 4, it will be $4 \times w_{LR}$.

Although the statement is straightforward to prove for the ideal Gaussian function, in the following section, we will empirically show that for the realistic SR kernels, the aforementioned self-convolutions idea can be utilized to find a coarse estimation SR kernel. For instance, in Figure 6, we plot an estimated low scale kernel, $\widehat{\mathbf{k_{LR}}}$, the output of self convolutions, $\widehat{\mathbf{k_{HR}}}$, and the ground truth SR kernel, $\mathbf{k}$.

3.1. The Proposed SR Kernel Estimation Scheme

In light of the empirical observation explained above, we propose an SR kernel estimation pipeline that consists of three main steps. In the first step, we can estimate the effective blurring kernel in the low scale, $\mathbf{k_{LR}}$. After having the estimation, $\widehat{\mathbf{k_{LR}}}$, a coarse estimation of SR kernel can be

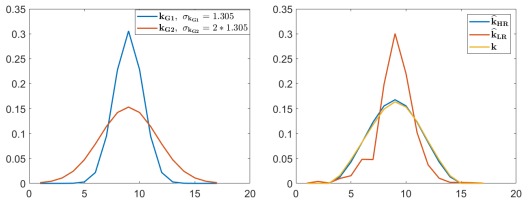

Figure 6. Ideal Gaussian kernel vs. realistic SR kernel. (a) Ideal Gaussian kernel with variance, $\sigma_{\mathbf{kG1}} = 1.305$, and its corresponding higher resolution kernel for $sf = 2$, which yields the Gaussian kernel, $\mathbf{k_{G2}}$, with variance $\mathbf{k_{G2}}$, $\sigma_{\mathbf{kG2}} = 2 * 1.305$ ($\sqrt{2^{sf}} = \sqrt{2^2} = 2$). (b) A realistic SR kernel, $\mathbf{k}$, from our kernel pool, estimated effective blurring kernel in low resolution, $\widehat{\mathbf{k}}_{\mathbf{LR}}$, and its corresponding $\widehat{\mathbf{k}}_{\mathbf{HR}}$ after self-convolutions.

obtained by convolving $\widehat{\mathbf{k}}_{\mathbf{LR}}$ with itself 2^{sf} times. And finally, a finer estimation of SR kernel, $\widehat{\mathbf{k}}$, can be obtained by refining this coarse estimation, $\widehat{\mathbf{k}}_{\mathbf{LR}}$. The proposed pipeline is shown in Figure 7.

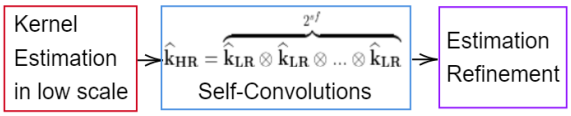

Figure 7. Pictoral representation of the proposed SR kernel estimation pipeline.

3.2. KernelNet

In order to realize the proposed SR kernel estimation pipeline with a real-time algorithm, we developed a modular neural network, KernelNet. KernelNet has four models; the first two are responsible from estimating the effective blurring kernel in low scale, $\widehat{\mathbf{k}}_{\mathbf{LR}}$. Then it is followed by the *Self-Convolutions Module* (later we call it FFT-2 Module) to up-sample the kernel, and finally, the finer estimation of $\mathbf{k}$ is received as an output of *Refinement Module*. In this section, we will explain each module in detail.

3.2.1 Gradient Estimation Module

In blind deblurring literature, working on the gradient domain instead of the spatial domain is a common practice [37, 24], as taking gradient filters out the unrelated information and keeps only the edges that are more informative [22]. In this manner, our low scale blurring kernel estimation part consists of two modules: The first of these is responsible for the sharp gradient estimation. Then, kernel estimation is done by a close form solution in FFT-1 Module. A possible Gradient Estimator Module is composed of fully convolutional layers, and as input it takes a two-channel input of gradients of observed image, $(\partial_h \mathbf{y}, \partial_v \mathbf{y})$. Then, it produces the corresponding two channel estimation that includes estimated sharp gradients in h and v directions (horizontal and vertical). As it gives slightly better performance than its competitors (see Ablation Study Section of the supplementary material) in blurring kernel estimation, our sharp gradient estimation module is inspired by [38]. The module consists of six convolutional layers. The five hidden layers have 128 neurons, and we use ReLu activation functions for these layers. For the output layer, Tanh is used. The kernel sizes are, 9×9, 1×1, 3×3, 5×5, 1×1 and 3×3, respectively. The first three layers are for sustaining the main structure, whereas the last layers are used to enhance the sharp edges [38].

3.2.2 FFT-1 Module

We define the estimated sharp gradient as follows,

$$\widehat{\nabla I_{LR}} = \begin{bmatrix} \widehat{\frac{\partial I_{LR}}{\partial_h}} \\ \widehat{\frac{\partial I_{LR}}{\partial_v}} \end{bmatrix} = \begin{bmatrix} \partial_h g \\ \partial_v g \end{bmatrix}, \tag{9}$$

where $\widehat{\frac{\partial I_{LR}}{\partial_h}}$ is the gradient in h dimension and $\widehat{\frac{\partial I_{LR}}{\partial_v}}$ is the gradient in v dimension. Having the sharp edge estimation from gradient estimator module, in order to obtain the effective blurring kernel in the low scale, $\mathbf{k_{LR}}$, we can solve the following optimization problem,

$$\widehat{\mathbf{k}}_{\mathbf{LR}} = \arg\min_{\mathbf{k_{LR}}} \left\| \mathbf{y} - \widehat{\nabla I_{LR}} \otimes \mathbf{k_{LR}} \right\|_2^2 + \lambda \left\| \mathbf{k_{LR}} \right\|_2^2, \tag{10}$$

where λ is a positive tradeoff parameter. The function to be minimized in (10) is quadratic and a closed form solution. However, by diagonalizing the gradient operator in Fourier domain [43], we can handle the optimization problem computationally efficiently,

$$\widehat{\mathbf{k}}_{\mathbf{LR}} = \mathcal{F}^{-1}\left(\frac{\overline{\mathcal{F}(\partial_h \mathbf{g})} \odot \mathcal{F}(\partial_h \mathbf{y}) + \overline{\mathcal{F}(\partial_v \mathbf{g})} \odot \mathcal{F}(\partial_v \mathbf{y})}{\mathcal{F}(\partial_h \mathbf{g})^2 + \mathcal{F}(\partial_v \mathbf{g})^2 + \lambda} \right), \tag{11}$$

where $\mathcal{F}(.)$, $\mathcal{F}^{-1}(.)$, $\overline{\mathcal{F}(.)}$, $\odot$ are Fourier transform, the inverse of Fourier transform, the complex conjugate of Fourier transform, and the element-wise multiplication, respectively. This module does not include any trainable parameters. We used PyTorch methods `torch.rfft` and `torch.irfft` for FFT and inverse FFT operations. After obtaining $\widehat{\mathbf{k}}_{\mathbf{LR}}$ via Equation (11), its negative values are thresholded out and then it is normalized to have sum one.

3.2.3 FFT-2 Module

Like the FFT-1 Module, the self convolutions can be handled in the Fourier domain computationally effectively as it can be done with sf times element-wise multiplication operations. As it is done after FFT-1 Module, the negative elements of the estimation $\widehat{\mathbf{k}}_{\mathbf{HR}}$ are zeroed out, and then it is

normalized to have sum one. The proposed FFT-2 Module is summarized in Algorithm 1, where $[k]_+ = \max\{k, 0\}$.

Algorithm 1: FFT-2 Module

Result: $\widehat{\mathbf{k}}_{\mathbf{HR}}$

initialization: $\mathbf{V} = \mathcal{F}\left(\widehat{\mathbf{k}}_{\mathbf{LR}}\right)$;

for $i = 1; i \leq sf, i = i + 1$ **do**

  $\mathbf{V} = \mathbf{V} \odot \mathbf{V}$;

end

$\widehat{\mathbf{k}}_{\mathbf{HR}} = \mathcal{F}^{-1}(\mathbf{V})$;

$\widehat{\mathbf{k}}_{\mathbf{HR}} = \left[\widehat{\mathbf{k}}_{\mathbf{HR}}\right]_+$;

$\widehat{\mathbf{k}}_{\mathbf{HR}} = \frac{\widehat{\mathbf{k}}_{\mathbf{HR}}}{\|\widehat{\mathbf{k}}_{\mathbf{HR}}\|_1}$;

3.2.4 Estimation Refinement Module

Our Estimation Refinement Module is also composed of fully convolution layers, and it takes the coarse estimation $\widehat{\mathbf{k}}_{\mathbf{HR}}$ as input. Then, it produces a finer and final estimation, $\hat{\mathbf{k}}$. Our network module structure is as follows: It is a compact network consisting of only three CNN layers. The hidden layers use 64 neurons. The kernel sizes are 3×3, 3×3 and 1×1, respectively. We use sigmoid activation functions after each layer. After the CNN layers, the estimation's negative elements are zeroed out; then, the estimation is also normalized to have sum 1, as it is done in FFT-1 and FFT-2 Modules.

3.3. Kernel Pool Generation

The recent works [17, 44] have proposed to estimate the realistic SR kernels from a specific camera dataset to create an SR kernel pool. Then, they used these kernel pools to degrade HR images to obtain a training set consisting of realistic LR-HR image pairs. In that way, they increased their CNN-based SR solutions' generalization capabilities. The study [44] used a model-based solution explained in Section 2.1, and [17] used KernelGAN. Among them, Real-world SR via kernel estimation and noise injection (RealSR) [17] achieves state-of-the-art performance on blind super-resolution and is the winner of "CVPR NTIRE 2020 Challenge on Real-World Super-Resolution" [26].

As explained in Section 2.3, the existing SR kernel estimation solutions have limited performances, which causes the kernel mismatch problem. In this study, we used an extended SR kernel pool that consists of the estimation of different alternative SR kernel estimation algorithms.

Similarly, we used the idea of SR kernel pool generation to train our KernelNet. From the Huawei P20 Dataset [16], we estimated SR kernels for scale factor two from 752 and 347 different images using the model-based solution and KernelGAN, respectively. The kernels are plotted in Figure 8. As can be observed from the plot, the estimated kernel pools for two different algorithms show slightly different features; model-based estimations are slightly narrower than the estimations of KernelGAN (see Section 2.3 about how it can affect the HR image estimations).

Namely, for scale factor two, we collected 748 (five very bad estimations are filtered out from 752) from model-based solution, 344 (three of them were filtered out from 347) from KernelGAN. We also created isotropic and anisotropic Gaussian kernels with standard deviations randomly drawn over interval $[1.2, 2]$. Their numbers are 420 and 204 for isotropic and anisotropic cases, respectively.

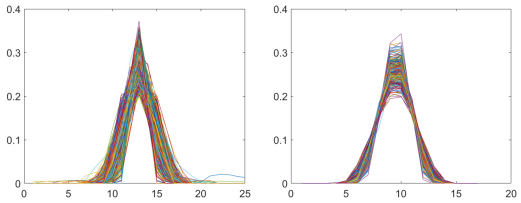

Figure 8. Realistic SR kernels are estimated from Huawei P20 Dataset [16] using two SR kernel estimation algorithms ($sf = 2$): (a) SR kernel estimations from 752 different images from the dataset using model based scheme. (b) Estimated SR kernels from 347 images using KernelGAN.

3.4. Kernel Pool for Scale 3 and Scale 4

Unfortunately, KernelGAN is not optimized for odd scale factors. Therefore we used only the model-based solution to estimate realistic SR kernels for $sf = 3$. However, our self-convolutions module (with one self-convolution) can also be applied to KernelGAN estimations obtained for $sf = 2$ to extend the pool. Namely, the SR kernel pool for $sf = 3$ consists of 883 kernels. Among, 305 kernels are estimated via model-based solution, 347 are self-convolved KernelGAN estimations (estimations for $sf = 2$), 232 are anisotropic, and 90 are isotropic Gaussian kernels with a standard deviation randomly drawn from the interval $\left[\sqrt{2} * 1.2, \sqrt{2} * 2\right]$. The kernel pool for $sf = 4$ was obtained similarly. Therefore, we leave the details to the supplementary material for the sake of brevity.

4. Training

4.1. Pre-training of Gradient Estimator Module

The gradient estimation module tries to produce the sharp gradient images in h and v directions from the gradients of the observed image, $\mathbf{y}$. For more stable end-to-end training of KernelNet, we first pre-trained this module as follows: A total of 13811 grayscale image patches of size 512×512 were cropped from the DIV2K Dataset.

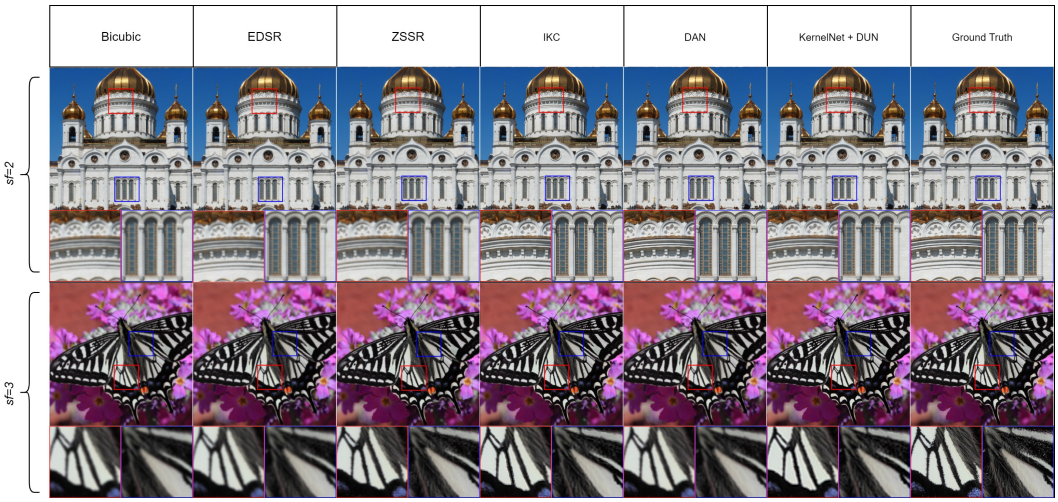

Figure 9. Examples from Synthetic Test Data.

Table 1. The performance comparison of the state-of-the-art SR solutions on synthetic test data.

	Scale	Bicubic	EDSR	IKC	KernelGAN + ZSSR	ZSSR	DAN	ESRGAN-FS	RealSR	KernelNet +DUN
PSNR /SSIM	**2x**	24.54/0.819	24.57/0.823	24.80/0.862	23.93/0.829	24.54/0.823	23.84/0.831	25.01/0.844	-	29.31/0.921
	3x	23.86/0.793	24.06/0.804	24.37/0.826	–	24.02/0.803	24.18/0.821	-	-	27.55/0.879
	4x	21.86/0.748	20.70/0.730	20.55/0.728	18.67/0.674	19.21/0.701	20.82/0.732	18.91/0.674	22.44/0.774	26.11/0.841

We obtained LR ones using randomly chosen SR kernels in the SR degradation model from these high-resolution images. To avoid aliasing during the ground truth LR sharp gradient images, instead of decimation operation, bicubic downsampling was applied to the original high-resolution images, i.e., $\mathbf{g_{GT}} = \nabla\left((\mathbf{s}) \downarrow_{bi,sf}\right)$. In order to compensate for this additional bicubic blurring, just for this pre-training, LR input images are also obtained via bicubic downsampling following the kernel blurring with randomly selected SR kernel from the pool, i.e., $\mathbf{y} = \nabla\left((\mathbf{s} \otimes \mathbf{k}) \downarrow_{bi,sf}\right)$.

From 13811 image patches, 12943 were randomly selected for training, and 868 were for validation. This network module was trained for 60 epochs. The batch size was 48, and the learning rate was $1e-4$. We used ℓ_1-norm as the loss function.

4.2. End-to-End Training of KernelNet

Having the pre-trained gradient estimation network, we end-to-end trained the whole kernel estimation network. Kernel-Net takes the gradient of the LR image as input and produces the estimation of SR kernel as output. LR images were obtained from HR images by using the SR degradation process given in Equation (1). The same training and validation datasets from DIV2K were used as HR image datasets. For $sf = 2$, 1368 SR kernels from the kernel pool were used for training. A more detailed explanation about the SR kernel training pool is given in the supplementary document. The learning rate was $1e-4$, batch size was 32, and the training of the end-to-end system reached convergence at epoch 15. During the training, the ℓ_1-norm was used as the loss function.

Table 2. Kernel estimation performance of the competing algorithms. The kernel estimation errors are reported in ℓ_1-norm. When the estimated kernels are used in blind SR algorithms, DUN, and DPIR, SR performances are also reported (in PSNR).

Method	Scale	Ker. Est. Error	PSNR/ SSIM (with DUN)	PSNR /SSIM (with DPIR)	Ker. Est. Time (sec)
Fixed Kernel (Gaussian)	2x	0.319	26.61/0.890	27.99/0.911	–
	3x	0.359	27.35/0.872	26.92/0.863	
	4x	0.311	21.79/0.790	24.09/0.805	
Model Based	2x	0.495	21.99/0.776	24.33/0.840	783.23
	3x	0.368	22.18/0.786	23.73/0.816	
	4x	0.392	18.32/0.652	21.50/0.731	
KernelGAN	2x	0.428	25.35/0.850	26.37/0.870	73.66
	3x	–	–	–	
	4x	0.570	16.96/0.620	20.41/0.707	
KernelNet	2x	**0.245**	**29.31**/0.918	29.20/**0.921**	**0.05**
	3x	**0.242**	**28.07/0.890**	27.55/0.879	
	4x	**0.221**	**26.11/0.841**	25.55/0.824	

4.3. Training Procedure for Scale 3 and Scale 4

We first trained the sharp gradient estimation network for 30 epochs. The batch size was 48, and the learning rate was $1e-4$. A total of 21255 grayscale images with resolution 510x510 were extracted from the DIV2K training dataset. Then, among them, 16753 were randomly selected for the training, and 4502 were selected for the validation. This pre-trained gradient estimator model is inserted into

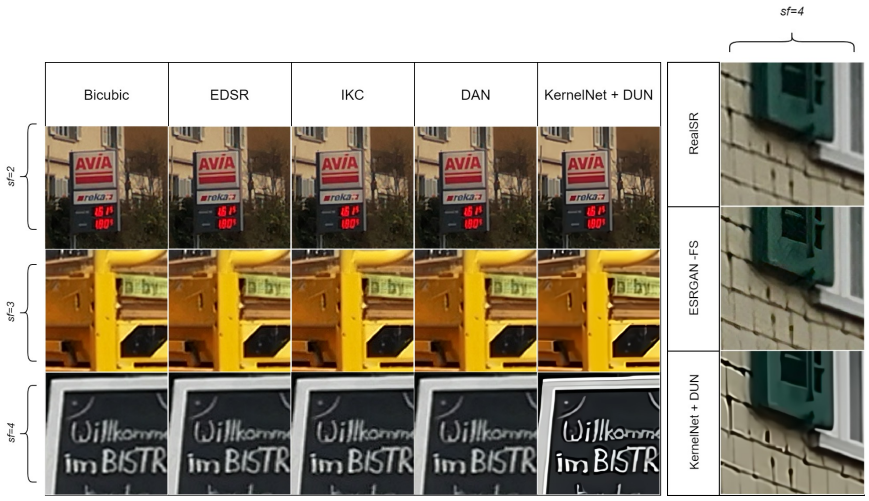

Figure 10. Visual Comparison on real images. On the right, copmarison with state-of-the-art GAN-based techniques for $sf = 4$ is given.

the end-to-end KernelNet system, and then the training of the whole system was started. The learning rate was $1e-4$, batch size was 32. The training of the end-to-end system reached convergence at epoch 15. For $sf = 3$, 749 kernels are used for training, while for $sf = 4$, 638 SR kernels were used (see the supplementary material for details).

5. Experimental Evaluation

5.1. Kernel Estimation on Synthetic Data

Synthetic data were generated as follows: We chose 100 images cropped with size 1024×1024 from the DIV2K test dataset. Then these images are randomly selected and degraded by SR degradation model in Equation (1). To degrade these images, we also created a test SR kernel pool. The test SR kernels were selected as subsets of SR pools for different scales. For instance, for $sf = 2$, 368 kernels from the kernel pool were separated as the test set. The details about test SR pools for each scale factor can be seen in the supplementary document.

From the synthetically generated LR images, the SR kernels were estimated using three different algorithms; KernelGAN, model-based approach, and KernelNet. As performance metric, we used ℓ_1-norm of SR kernel estimation error, i.e., $\left\|\widehat{\mathbf{k}} - \mathbf{k}\right\|_1$. Moreover, one may question whether to use a fixed kernel instead of estimating SR kernel. To discuss this possibility, we also used a fixed Gaussian kernel with standard deviation 1.6 (for only $sf = 4$, it is updated to 3). Therefore, also average $\left\|\widehat{\mathbf{k_G}} - \mathbf{k}\right\|_1$ over test set was also reported. Having estimation of SR kernel, we used Deep Unfolding Network (DUN) [40] or Plug-and-play image restoration with deep Denoiser PRior (DPIR) [41] to estimate HR image. In Table 2, we can see the comparison of all competing SR kernel estimation techniques. We also compared our proposed solution with the state-of-the-art SR algorithms. Namely, we selected simple bicubic interpolation, EDSR, ZSSR, IKC, DAN, and two GAN-based state-of-the-art blind SR solutions, EDSR-FS and RealSR, as competing algorithms. The comparison of the state-of-the-art SR solutions on synthetic test data is given in Table 1. In Figure 9, a visual comparison of blind SR algorithms on examples from the synthetic test data can be seen. Please note that EDSR-FS and RealSR are only suitable for even scale factors. These two algorithms were trained by using the same training set that KernelNet used, while for other algorithms, the original training procedures were used.

5.2. Visual Comparison on Real Images

For visual comparison on real images, the images from the Huawei P20 dataset were up-scaled using all competing SR methods. The noise level σ for the network DUN was chosen as $\sigma = 10$, which was 0 for the synthetic dataset. The visual comparison with state-of-the-art SR solutions is given in Figure 10. More visual examples and ablation study can be found in the supplemental material.

6. Conclusion

In this paper, we introduced KernelNet, a novel, modular, and interpretable neural network used to reconstruct the SR kernel accurately at breakneck speeds. Combined with a non-blind SR solution such as Deep Unfolding Network, we were able to surpass most of the well-known state-of-the-art blind SR techniques both in various quality metrics as well as visually on real images. The proposed real SR scheme demonstrates tremendous promise towards improving the SR capability of current mobile phones.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017.

[2] Simon Baker and Takeo Kanade. Limits on super-resolution and how to break them. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1167–1183, 2002.

[3] Sefi Bell-Kligler, Assaf Shocher, and Michal Irani. Blind super-resolution kernel estimation using an internal-gan. In *Advances in Neural Information Processing Systems*, pages 284–293, 2019.

[4] Hong Chang, Dit-Yan Yeung, and Yimin Xiong. Super-resolution through neighbor embedding. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I. IEEE, 2004.

[5] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014.

[6] 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.

[7] Michael Elad and Arie Feuer. Restoration of a single super-resolution image from several blurred, noisy, and undersampled measured images. *IEEE transactions on image processing*, 6(12):1646–1658, 1997.

[8] Manuel Fritsche, Shuhang Gu, and Radu Timofte. Frequency separation for real-world super-resolution. In *IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, 2019.

[9] 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.

[10] 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.

[11] Ankit Gupta, Neel Joshi, C Lawrence Zitnick, Michael Cohen, and Brian Curless. Single image deblurring using motion density functions. In *European Conference on Computer Vision*, pages 171–184. Springer, 2010.

[12] Stefan Harmeling, Hirsch Michael, and Bernhard Schölkopf. Space-variant single-image blind deconvolution for removing camera shake. In *Advances in Neural Information Processing Systems*, pages 829–837, 2010.

[13] 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, 2010.

[14] Yu He, Kim-Hui Yap, Li Chen, and Lap-Pui Chau. A soft map framework for blind super-resolution image reconstruction. *Image and Vision Computing*, 27(4):364–373, 2009.

[15] Yan Huang, Shang Li, Liang Wang, Tieniu Tan, et al. Unfolding the alternating optimization for blind super resolution. *Advances in Neural Information Processing Systems*, 33, 2020.

[16] Andrey Ignatov, Radu Timofte, Sung-Jea Ko, Seung-Wook Kim, Kwang-Hyun Uhm, Seo-Won Ji, Sung-Jin Cho, Jun-Pyo Hong, Kangfu Mei, Juncheng Li, et al. Aim 2019 challenge on raw to rgb mapping: Methods and results. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3584–3590. IEEE, 2019.

[17] Xiaozhong Ji, Yun Cao, Ying Tai, Chengjie Wang, Jilin Li, and Feiyue Huang. Real-world super-resolution via kernel estimation and noise injection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 466–467, 2020.

[18] Robert Keys. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, 29(6):1153–1160, 1981.

[19] Deepa Kundur and Dimitrios Hatzinakos. Blind image deconvolution. *IEEE signal processing magazine*, 13(3):43–64, 1996.

[20] 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.

[21] Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman. Understanding and evaluating blind deconvolution algorithms. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1964–1971. IEEE, 2009.

[22] Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *CVPR 2011*, pages 2657–2664. IEEE, 2011.

[23] Yongbo Li, Weisheng Dong, Xuemei Xie, Guangming Shi, Jinjian Wu, and Xin Li. Image super-resolution with parametric sparse model learning. *IEEE Transactions on Image Processing*, 27(9):4638–4650, 2018.

[24] Yuelong Li, Mohammad Tofighi, Junyi Geng, Vishal Monga, and Yonina C Eldar. Deep algorithm unrolling for blind image deblurring. *arXiv preprint arXiv:1902.03493*, 2019.

[25] 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.

[26] Andreas Lugmayr, Martin Danelljan, and Radu Timofte. Ntire 2020 challenge on real-world image super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 494–495, 2020.

[27] Andreas Lugmayr, Martin Danelljan, Radu Timofte, Manuel Fritsche, Shuhang Gu, Kuldeep Purohit, Praveen Kandula,

Maitreya Suin, AN Rajagoapalan, Nam Hyung Joon, et al. Aim 2019 challenge on real-world image super-resolution: Methods and results. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3575–3583. IEEE, 2019.

[28] Antonio Marquina and Stanley J Osher. Image super-resolution by tv-regularization and bregman iteration. *Journal of Scientific Computing*, 37(3):367–382, 2008.

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

[30] 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.

[31] Qi Shan, Jiaya Jia, and Aseem Agarwala. High-quality motion deblurring from a single image. *Acm transactions on graphics (tog)*, 27(3):1–10, 2008.

[32] 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.

[33] Assaf Shocher, Shai Bagon, Phillip Isola, and Michal Irani. Ingan: Capturing and retargeting the. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4492–4501, 2019.

[34] Jian Sun, Zongben Xu, and Heung-Yeung Shum. Image super-resolution using gradient profile prior. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[35] Yu-Wing Tai, Shuaicheng Liu, Michael S Brown, and Stephen Lin. Super resolution using edge prior and single image detail synthesis. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2400–2407. IEEE, 2010.

[36] 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 (ECCV) Workshops*, pages 0–0, 2018.

[37] Li Xu and Jiaya Jia. Two-phase kernel estimation for robust motion deblurring. In *European conference on computer vision*, pages 157–170. Springer, 2010.

[38] Xiangyu Xu, Jinshan Pan, Yu-Jin Zhang, and Ming-Hsuan Yang. Motion blur kernel estimation via deep learning. *IEEE Transactions on Image Processing*, 27(1):194–205, 2017.

[39] Yuan Yuan, Siyuan Liu, Jiawei Zhang, Yongbing Zhang, Chao Dong, and Liang Lin. Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 701–710, 2018.

[40] Kai Zhang, Luc Van Gool, and Radu Timofte. Deep unfolding network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3217–3226, 2020.

[41] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *arXiv preprint arXiv:2008.13751*, 2020.

[42] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Deep plug-and-play super-resolution for arbitrary blur kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1671–1681, 2019.

[43] Ningning Zhao, Qi Wei, Adrian Basarab, Nicolas Dobigeon, Denis Kouamé, and Jean-Yves Tourneret. Fast single image super-resolution using a new analytical solution for ℓ_2–ℓ_2 problems. *IEEE Transactions on Image Processing*, 25(8):3683–3697, 2016.

[44] Ruofan Zhou and Sabine Susstrunk. Kernel modeling super-resolution on real low-resolution images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2433–2443, 2019.