Flow-based Kernel Prior with Application to Blind Super-Resolution

Jingyun Liang[1] Kai Zhang[1,*] Shuhang Gu[1,2] Luc Van Gool[1,3] Radu Timofte[1]
[1] Computer Vision Lab, ETH Zurich, Switzerland
[2] The University of Sydney, Australia [3] KU Leuven, Belgium
{jinliang, kai.zhang, vangool, timofter}@vision.ee.ethz.ch shuhanggu@gmail.com
https://github.com/JingyunLiang/FKP

Abstract

Kernel estimation is generally one of the key problems for blind image super-resolution (SR). Recently, Double-DIP proposes to model the kernel via a network architecture prior, while KernelGAN employs the deep linear network and several regularization losses to constrain the kernel space. However, they fail to fully exploit the general SR kernel assumption that anisotropic Gaussian kernels are sufficient for image SR. To address this issue, this paper proposes a normalizing flow-based kernel prior (FKP) for kernel modeling. By learning an invertible mapping between the anisotropic Gaussian kernel distribution and a tractable latent distribution, FKP can be easily used to replace the kernel modeling modules of Double-DIP and KernelGAN. Specifically, FKP optimizes the kernel in the latent space rather than the network parameter space, which allows it to generate reasonable kernel initialization, traverse the learned kernel manifold and improve the optimization stability. Extensive experiments on synthetic and real-world images demonstrate that the proposed FKP can significantly improve the kernel estimation accuracy with less parameters, runtime and memory usage, leading to state-of-the-art blind SR results.

1. Introduction

Image super-resolution (SR) is a fundamental low-level vision task whose goal is to recover the high-resolution (HR) image from the low-resolution (LR) input. With the development of convolutional neural networks (CNN), CNN-based methods [10, 19, 25, 30, 38, 47, 56] have been gaining the popularity in solving image SR. However, most of existing works assume the blur kernel is fixed and known (*e.g.*, bicubic downsampling kernel), which tends to result in a dramatic performance drop in real-world applications. Hence, blind image SR that aims to deal with unknown blur kernels is becoming an active research topic.

Compared to non-blind SR, blind SR generally needs to additionally estimate the blur kernel and thus is more ill-posed. A popular line of work tries to decompose blind SR into two sub-problems, *i.e.*, kernel estimation and non-blind SR. As a preliminary step of non-blind SR, kernel estimation plays a crucial role. If the estimated kernel deviates from the ground-truth, the HR image reconstructed by the non-blind SR methods would deteriorate seriously [11, 17, 50]. In view of this, this paper focuses on the SR kernel estimation problem.

Recently, some kernel estimation methods, such as Double-DIP [15, 39] and KernelGAN [3], have shown promising results. Specifically, with two deep image priors (DIPs) [44], Double-DIP can be used to jointly optimize the HR image and blur kernel in the parameter space of untrained encoder-decoder networks by minimizing the LR image reconstruction error. Although DIP has shown to be effective for modeling natural images, whether it is effective to model blur kernel or not remains unclear. The main reason is that blur kernel usually has a small spatial size and has its own characteristics that differ from natural images. In [39], a fully-connected network (FCN) is used to model the kernel prior, which, however, lacks interpretability. With a different framework to Double-DIP, KernelGAN designs an internal generative adversarial network (GAN) for the LR image on the basis of image patch recurrence property [16, 35, 57]. It defines the kernel implicitly by a deep linear network, which is optimized by the GAN loss and five extra regularization losses such as sparsity loss. Obviously, these two methods do not make full use of the anisotropic Gaussian kernel prior which has been demonstrated to be effective enough for real image SR [11, 40, 50, 54, 55].

In this paper, we propose a flow-based kernel prior (FKP) for kernel distribution modeling and incorporate it into existing blind SR models. Based on normalizing flow, FKP consists of several batch normalization layers, permutation layers and affine coupling layers, which allow the model

*Corresponding author.

to capture the kernel distribution by learning an invertible mapping between the kernel space and the latent space (*e.g.*, high-dimensional Gaussian). FKP is optimized in an unsupervised way by minimizing the negative log-likelihood loss of the kernel. Once trained, it can be incorporated into existing blind SR models such as Double-DIP and KernelGAN for kernel estimation, in which FKP fixes its parameters and optimizes the latent variable in the network input space. Specifically, for Double-DIP, we jointly optimize DIP for HR image estimation and FKP for kernel estimation by minimizing the LR image reconstruction error. For KernelGAN, we blur the LR image with the kernel estimated by FKP rather than using a deep linear network, and then optimize it by adversarial training.

Using FKP as a kernel prior offers several advantages: 1) *Fewer parameters.* FKP model only has 143K parameters, whereas Double-DIP and KernelGAN involve 641K and 151K for kernel modeling, respectively. 2) *More stable convergence.* On the one hand, unlike Double-DIP that uses random noise input and KernelGAN that uses random network parameters for kernel initialization, FKP can explicitly initialize a reasonable kernel since it is a bijective function. On the other hand, the kernel is implicitly constrained to be in the learned kernel manifold during model optimization. 3) *Better kernel estimation.* With a learned kernel prior, the kernel estimation accuracy can be improved for several existing blind SR methods such as Double-DIP and KernelGAN.

The main contributions are summarized as follows:

1) We propose a kernel prior named FKP that is applicable for arbitrary blur kernel modeling. It learns a bijective mapping between the kernel and the latent variable. To the best of our knowledge, FKP is the first learning-based kernel prior.
2) By fixing its parameters and optimizing the latent variable, FKP traverses the learned kernel manifold and searches for the kernel prediction, ensuring reasonable kernels for initialization and along optimization.
3) With less parameters, runtime and memory usage, FKP improves the stability and accuracy of existing kernel estimation methods including Double-DIP and KernelGAN, leading to state-of-the-art blind SR performance.

2. Related Work

Kernel Estimation. Prior to the deep learning era, traditional kernel estimation methods typically utilize prior information of image patches or edges [2, 21, 35, 41, 46]. In the deep learning era, Gandelsman *et al*. [15] propose the Double-DIP based on the Deep Image Prior (DIP) [44], which uses untrained encoder-decoder networks with skip connections as image priors for image dehazing, image deconvolution, transparency separation, *etc*. Similarly, Ren *et al*. [39] propose a fully-connected network (FCN) as a kernel prior for image deconvolution. However, whether this idea works on blind SR kernel estimation or not is still an open problem, as blind SR is severely ill-posed due to downsampling. Different from above methods, Kligler *et al*. [3] propose KernelGAN to estimate kernel based on the image patch recurrence property [16, 35, 57]. They use a deep linear network as a generator to generate a re-downscaled image from the LR image, and a discriminator to ensure cross-scale patch similarity. The blur kernel is derived from the generator. Gu *et al*. [17] propose a predict-and-correct strategy to estimate kernel and HR image alternately. But it is highly dependent on training HR-LR image pairs and only estimates the feature of kernel.

Normalizing Flow. Normalizing flows [8, 9, 22, 24, 28, 29, 32, 37] are invertible generative models that deform the complex data distribution to a simple and tractable distribution. Dinh *et al*. [8] propose to stack non-linear additive coupling and other transformation layers as the flow model NICE. Inspired by NICE, Dinh *et al*. [9] propose RealNVP, which upgrades additive coupling to affine coupling without loss of invertibility and achieves better performance. After that, Kingma *et al*. [28] propose 1×1 convolution to replace the fixed permutation layer in RealNVP and succeed to synthesize realistic-looking images. Normalizing flows have also been successfully applied in generating other types of data, such as audio data [26] and point cloud data [51].

3. Flow-based Kernel Prior

Generally, the classical degradation model of image SR [12, 13, 31] assumes the LR image $\mathbf{y}$ is obtained via a composition of blurring and downsampling from the HR image $\mathbf{x}$. Mathematically, it is formulated as

$$\mathbf{y} = (\mathbf{x} \otimes \mathbf{k}) \downarrow_s + \mathbf{n}, \tag{1}$$

where $\mathbf{x} \otimes \mathbf{k}$ denotes the convolution between $\mathbf{x}$ and blur kernel $\mathbf{k}$, $\downarrow_s$ represents the downsampling operation with scale factor s, and $\mathbf{n}$ is the noise. Particularly, blind SR [3, 17, 21, 44] aims to estimate the HR image and blur kernel simultaneously. According to the *Maximum A Posteriori* (MAP) framework, it can be solved as

$$\mathbf{x}^*, \mathbf{k}^* = \arg\min_{\mathbf{x},\mathbf{k}} \|\mathbf{y} - (\mathbf{x} \otimes \mathbf{k}) \downarrow_s \|^2 + \lambda\Phi(\mathbf{x}) + \gamma\Omega(\mathbf{k}), \tag{2}$$

where $\|\mathbf{y} - (\mathbf{x} \otimes \mathbf{k}) \downarrow_s \|^2$ is the data fidelity term, $\Phi(\mathbf{x})$ denotes the image prior, $\Omega(\mathbf{k})$ represents the kernel prior, λ and γ are trade-off parameters. It has been well-studied that poor kernel estimation would cause a severe performance drop for HR image estimation [11, 17, 55]. However, while various image priors have been proposed to describe natural image statistics [5, 6, 7, 20, 36, 44], little attention has been paid on designing the kernel prior.

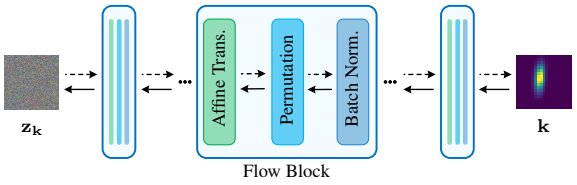


Figure 1: The schematic illustration of the flow-based kernel prior (FKP) network. FKP learns an invertible mapping between the kernel $\mathbf{k}$ and the latent variable $\mathbf{z_k}$ by several flow blocks, each of which is a succession of batch normalization, permutation and affine transformation layers.

In view of this, we aim to learn a kernel prior based on normalizing flow in this paper. Formally, let $\mathbf{k} \in K$ denotes the kernel variable and $\mathbf{z_k} \in Z$ denotes the corresponding latent variable. $\mathbf{k}$ and $\mathbf{z_k}$ obey probability distributions p_K and p_Z, respectively. We define a bijection $f_{\boldsymbol{\theta}} : K \rightarrow Z$ with parameter $\boldsymbol{\theta}$. For kernel $\mathbf{k}$, it can be encoded as a latent variable $\mathbf{z_k} = f_{\boldsymbol{\theta}}(\mathbf{k})$ in the latent space. Inversely, $\mathbf{k}$ could be exactly reconstructed by the inverse mapping: $\mathbf{k} = f_{\boldsymbol{\theta}}^{-1}(\mathbf{z_k})$. According to the *change of variable* formula [8], the probability of $\mathbf{k}$ is computed as

$$p_K(\mathbf{k}) = p_Z(f_{\boldsymbol{\theta}}(\mathbf{k})) \left| \det\left(\frac{\partial f_{\boldsymbol{\theta}}(\mathbf{k})}{\partial \mathbf{k}}\right) \right|, \tag{3}$$

where $\frac{\partial f_{\boldsymbol{\theta}}(\mathbf{k})}{\partial \mathbf{k}}$ is the Jacobian of $f_{\boldsymbol{\theta}}$ at $\mathbf{k}$. Generally, p_Z is a simple tractable distribution, such as multivariate Gaussian distribution. $f_{\boldsymbol{\theta}}$ is often composed of a sequence of invertible and tractable transformations: $f_{\boldsymbol{\theta}} = f_{\boldsymbol{\theta}}^1 \circ f_{\boldsymbol{\theta}}^2 \circ \cdots \circ f_{\boldsymbol{\theta}}^N$, and we have $\mathbf{h}^n = f_{\boldsymbol{\theta}}^n(\mathbf{h}^{n-1})$ for $n \in \{1, ..., N\}$. The input and output $\mathbf{h}^0$ and $\mathbf{h}^N$ of $f_{\boldsymbol{\theta}}$ are $\mathbf{k}$ and $\mathbf{z_k}$, respectively. Under maximum likelihood estimation, $\boldsymbol{\theta}$ can be optimized by minimizing the negative log-likelihood (NLL) loss

$$\mathcal{L}(\mathbf{k}; \boldsymbol{\theta}) = -\log p_Z(f_{\boldsymbol{\theta}}(\mathbf{k})) - \sum_{n=1}^{N} \log \left| \det\left(\frac{\partial f_{\boldsymbol{\theta}}^n(\mathbf{h}^{n-1})}{\partial \mathbf{h}^{n-1}}\right) \right|. \tag{4}$$

More specifically, we build FKP by stacking invertible flow layers. As shown in Fig. 1, it consists of several flow blocks, and each block includes three consecutive layers: batch normalization layer, permutation layer and affine transformation layer [9]. For affine transformation layer, we use small fully-connected neural networks (FCN) for scaling and shifting, in which each FCN stacks fully-connected layers and *tanh* activation layers alternately.

FKP is trained by the NLL loss given training kernel samples. When it is plugged into existing kernel estimation models as a kernel prior, we first randomly sample a latent variable $\mathbf{z_k}$, which corresponds to a random kernel as shown in Fig. 2. Then, we fix the model parameters and update $\mathbf{z_k}$ by gradient back-propagation under the guidance of kernel estimation loss. Instead of starting with random initialization and slowly updating the kernel, FKP moves along the learned kernel manifold and generates reliable kernels $f_{\boldsymbol{\theta}}^{-1}(\mathbf{z_k})$ during the update of $\mathbf{z_k}$. In addition, when $\mathbf{z_k}$ follows multivariate Gaussian distribution, most of the mass of distribution is near the surface of a sphere of radius $\sqrt{D}$ [34, 45], where D is the dimension of $\mathbf{z_k}$. Therefore we optimize $\mathbf{z_k}$ on the sphere surface by restricting its Euclidean norm as $\|\mathbf{z_k}\|_2 = \sqrt{D}$ after every update, avoiding optimizing in the entire latent space.

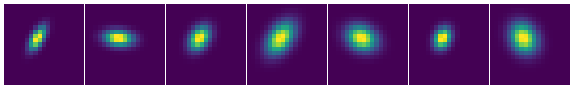


Figure 2: Diverse kernel samples generated by randomly sampling from the latent space of the flow-based kernel prior (FKP). Here, FKP is trained on anisotropic Gaussian kernels with scale factor 4.

4. Incorporating FKP to Double-DIP

4.1. Original Double-DIP

DIP [44] is an image prior in the form of a randomly initialized encoder-decoder network $\mathcal{G}$, in which the network structure captures low-level image statistics. By optimizing network parameter $\boldsymbol{\theta}_{\mathcal{G}}$, natural-looking image $\mathbf{x} = \mathcal{G}(\mathbf{z_x}; \boldsymbol{\theta}_{\mathcal{G}})$ is reconstructed from the fixed random noise input $\mathbf{z_x}$. To model different image components, Double-DIP [15] couples two DIPs for image decomposition tasks such as image segmentation. This framework is also exploited in image deconvolution by replacing one DIP with a fully-connected network (FCN) [39]. If Double-DIP and its variants are used for blind SR, they can be formulated as

$$\begin{cases} \boldsymbol{\theta}_{\mathcal{G}}^*, \boldsymbol{\theta}_{\mathcal{K}'}^* = \underset{\boldsymbol{\theta}_{\mathcal{G}}, \boldsymbol{\theta}_{\mathcal{K}'}}{\arg\min} \|\mathbf{y} - (\mathcal{G}(\mathbf{z_x}; \boldsymbol{\theta}_{\mathcal{G}}) \otimes \mathcal{K}'(\mathbf{z_k}; \boldsymbol{\theta}_{\mathcal{K}'})) \downarrow_s \|^2 \\ \mathbf{x}^* = \mathcal{G}(\mathbf{z_x}; \boldsymbol{\theta}_{\mathcal{G}}^*), \quad \mathbf{k}^* = \mathcal{K}'(\mathbf{z_k}; \boldsymbol{\theta}_{\mathcal{K}'}^*) \end{cases} \tag{5}$$

where $\mathcal{K}'(\mathbf{z_k}; \boldsymbol{\theta}_{\mathcal{K}'})$ is a kernel prior based on untrained neural networks such as DIP and FCN. In training, random noise inputs $\mathbf{z_x}$ and $\mathbf{z_k}$ are fixed, while randomly initialized network parameter $\boldsymbol{\theta}_{\mathcal{G}}$ and $\boldsymbol{\theta}_{\mathcal{K}'}$ are optimized to minimize the LR image reconstruction error.

However, applying above Double-DIP framework to blind SR kernel estimation is non-trivial. On the one hand, unlike images, kernels are spatially small and have no natural image properties such as self-similarity. Designing a DIP-like network based on convolution layers or fully-connected layers and then using the network architecture as a kernel prior may not be a good choice. On the other hand, due to the downsampling operation, blind SR is extremely ill-posed compared with other image restoration tasks. The knowledge incorporated by untrained networks may not be sufficient for estimating the HR image and kernel simultaneously. As will be shown in experiments, untrained neural networks fail to generate reasonable kernel estimations.

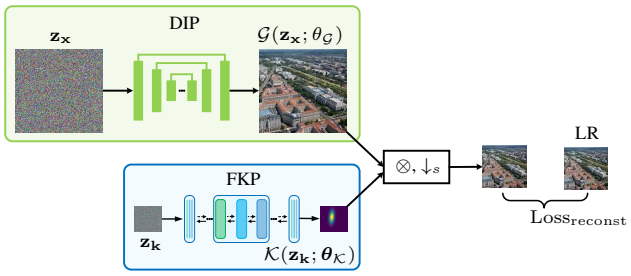


Figure 3: The schematic illustration of DIP-FKP. DIP estimates SR image $\mathcal{G}(\mathbf{z_x};\theta_\mathcal{G})$ from $\mathbf{z_x}$, while FKP estimates kernel $\mathcal{G}(\mathbf{z_x};\theta_\mathcal{G})$ from $\mathbf{z_k}$. $\otimes$ and $\downarrow_s$ denote blur and downsampling with scale factor s, respectively. The model is optimized by minimizing the LR image reconstruction error.

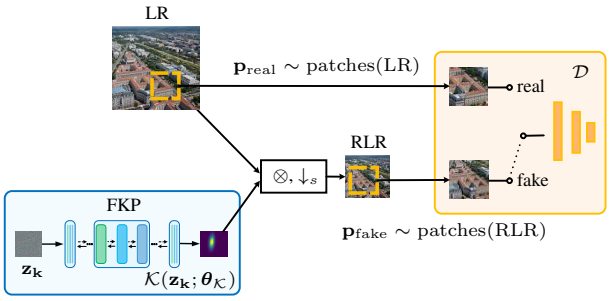


Figure 4: The schematic illustration of KernelGAN-FKP. Kernel $\mathcal{K}(\mathbf{z_k};\boldsymbol{\theta}_\mathcal{K})$ is generated by FKP and is then used to generate the re-downscaled LR (RLR) image from the LR image. $\otimes$ and $\downarrow_s$ denote blur and downsampling with scale factor s, respectively. The model is optimized similar to GAN: FKP tries to fool the discriminator $\mathcal{D}$ to believe that image patch $\mathbf{p}_{\text{real}}$ extracted from the LR image and patch $\mathbf{p}_{\text{fake}}$ extracted from the RLR image share the same distribution.

4.2. Proposed DIP-FKP

Instead of using a DIP-like untrained network as the kernel prior, we propose to incorporate FKP into the Double-DIP framework, which we refer to as DIP-FKP. The HR image and kernel are jointly estimated as

$$\begin{cases} \boldsymbol{\theta}^*_\mathcal{G}, \mathbf{z}^*_\mathbf{k} = \arg\min_{\boldsymbol{\theta}_\mathcal{G}, \mathbf{z_k}} \|\mathbf{y} - (\mathcal{G}(\mathbf{z_x};\boldsymbol{\theta}_\mathcal{G}) \otimes \mathcal{K}(\mathbf{z_k};\boldsymbol{\theta}_\mathcal{K})) \downarrow_s \|^2 \\ \mathbf{x}^* = \mathcal{G}(\mathbf{z_x};\boldsymbol{\theta}^*_\mathcal{G}), \quad \mathbf{k}^* = \mathcal{K}(\mathbf{z}^*_\mathbf{k};\boldsymbol{\theta}_\mathcal{K}) \end{cases} \tag{6}$$

where $\mathcal{K}(\mathbf{z_k};\boldsymbol{\theta}_\mathcal{K})$ is the incorporated FKP. The corresponding schematic illustration is shown in Fig. 3.

In DIP-FKP, we optimize the kernel latent variable $\mathbf{z_k}$, rather than network parameter $\boldsymbol{\theta}_\mathcal{K}$, which is fixed as it has modeled the kernel prior. More specifically, in forward propagation, FKP generates a kernel prediction $\mathcal{K}(\mathbf{z_k};\boldsymbol{\theta}_\mathcal{K})$ to blur the SR image $\mathcal{G}(\mathbf{z_x};\boldsymbol{\theta}_\mathcal{G})$ produced by DIP, in order to obtain the LR image prediction. The mean squared error between the resulting LR prediction and LR image are used as the loss function. In back propagation, gradients are back-propagated from the loss function to kernel prediction, and then to the latent variable $\mathbf{z_k}$.

With FKP, DIP-FKP embeds kernel prior into the network effectively by moving the kernel prediction along the learned kernel manifold, which enables accurate and stable kernel estimation. Therefore, without massive training data and long training time, DIP-FKP can estimate SR image and blur kernel simultaneously during testing phase. It is noteworthy that while DIP-FKP is able to estimate kernels accurately, it has limited performance on SR image reconstruction as it is self-supervised. For this reason, we use non-blind model USRNet [53] to generate the final SR result based on the kernel estimation.

5. Incorporating FKP to KernelGAN

5.1. Original KernelGAN

In a single image, small image patches tend to recur across different scales [16, 57]. It is further observed in [35] that when the LR image is degraded with a blur kernel to obtain a re-downscaled low-resolution (RLR) image, the kernel between the HR and LR image maximizes the internal patch distribution similarity between the LR and RLR image. According to this observation, KernelGAN [3] trains an internal generative adversarial network (GAN) on a single LR image in order to estimate the blur kernel. It consists of a deep linear generator $\mathcal{G}$ that downscales the LR image by several learnable convolution layers and a discriminator $\mathcal{D}$ that distinguishes between patch distributions of the LR and RLR image. The blur kernel is derived from $\mathcal{G}$ in every iteration. Formally, KernelGAN is optimized as

$$\begin{cases} \boldsymbol{\theta}^*_\mathcal{G}, \boldsymbol{\theta}^*_\mathcal{D} = \arg\min_{\boldsymbol{\theta}_\mathcal{G}} \max_{\boldsymbol{\theta}_\mathcal{D}} \left\{ (\mathcal{D}(\mathbf{p}) - 1)^2 + (\mathcal{D}(\mathcal{G}(\mathbf{p})))^2 + \mathcal{R} \right\} \\ \mathbf{k}^* \leftarrow \boldsymbol{\theta}^*_\mathcal{G} \end{cases} \tag{7}$$

where $\mathbf{p}$ is a patch randomly extracted from the LR image $\mathbf{y}$, *i.e.*, $\mathbf{p} \sim \text{patches}(\mathbf{y})$, and $\mathcal{R}$ represents extra regularization on the kernel $\mathbf{k}$. More specifically, $\mathcal{R}$ includes the mean squared error between $\mathbf{k}$ and the bicubic kernel, as well as other regularization constraints on kernel pixel sum, boundary, sparsity and centrality. However, the performance of KernelGAN is unstable. It also suffers from the burden of hyper-parameter selection due to multiple regularization terms.

5.2. Proposed KernelGAN-FKP

The instability of KernelGAN may come from the weak patch distribution of some images, *i.e.*, there are multiple kernels that could generate the RLR image with similar patch distribution to the LR image. In this case, the discriminator cannot distinguish between different patch distributions, thus leading to wrong kernel prediction. To alleviate the problem, we propose to incorporate the proposed FKP into KernelGAN in order to constrain the optimization

space. We refer to this method as KernelGAN-FKP which can be formulated as

$$\begin{cases} \mathbf{z}_{\mathbf{k}}^{*}, \boldsymbol{\theta}_{\mathcal{D}}^{*} = \arg\min_{\mathbf{z}_{\mathbf{k}}} \max_{\boldsymbol{\theta}_{\mathcal{D}}} \{(\mathcal{D}(\mathbf{p}) - 1)^2 \\ \qquad\qquad + (\mathcal{D}((\mathbf{p} \otimes \mathcal{K}(\mathbf{z}_{\mathbf{k}}; \boldsymbol{\theta}_{\mathcal{K}})) \downarrow_s))^2\} \\ \mathbf{k}^{*} = \mathcal{K}(\mathbf{z}_{\mathbf{k}}^{*}; \boldsymbol{\theta}_{\mathcal{K}}) \end{cases} \tag{8}$$

As illustrated schematically in Fig. 4, KernelGAN-FKP directly generates the kernel from the latent variable $\mathbf{z_k}$ and uses it to degrade the LR image instead of using a deep linear network. In optimization, $\mathbf{z_k}$ is optimized to fool the discriminator, which equals to traversing in the kernel space to find a kernel that the discriminator cannot distinguish. This constrains the optimization space of the generator and ensures kernel generation quality, which allows more stable convergence than the original KernelGAN, even without extra regularization terms. Similar to DIP-FKP, we adopt USRNet [53] for non-blind SR after kernel prediction.

6. Experiments

6.1. Experimental Setup

Data Preparation. Since the blur kernels of real LR images are usually unimodal [35, 41] and can typically be modeled by a Gaussian [40], most existing blind SR works [3, 17, 21, 40, 42, 43, 49, 54, 55] assume the SR kernel is isotropic or anisotropic Gaussian kernel. Following this widely-adopted assumption, we conduct experiments on anisotropic Gaussian kernels, even though FKP is trained in an unsupervised way and can be used to model arbitrary kernel estimation given kernel samples. For scale factor $s \in \{2, 3, 4\}$, the kernel sizes and width ranges are set to $(4s + 3) \times (4s + 3)$ and $[0.175s, 2.5s]$, respectively. The rotation angle range is $[0, \pi]$ for all s. We blur and downsample images with random kernels to generate testing sets based on Set5 [4], Set14 [52], BSD100 [33], Urban100 [23] and the validation set of DIV2K [1]. Following KernelGAN [3] and USRNet [53], the blur kernel is shifted and the upper-left pixels are kept in downsampling to avoid subpixel misalignments. For evaluation, we compare kernels by PSNR, and compare SR images by PSNR and SSIM [48] on Y channel in the YCbCr space.

FKP. Since FKP is a bijection, the latent variable dimension, the FCN input and output dimensions are the same as the kernel size. To capture kernel variety, the dimension of FCN hidden layers is set to $5(s + 1)$ for scale factor s. The total number of flow blocks and FCN depth are set to 5 and 3, respectively. We randomly generate anisotropic Gaussian kernels as training data and use the Adam optimizer [27] to optimize the model for 50,000 iterations. The batch size and learning rate are set to 100 and 1e-4, respectively. Due to page limit, ablation study on FKP is provided in the supplementary.

DIP-FKP. The DIP architecture is same as in [44]. The model is optimized by the Adam optimizer for 1,000 iterations with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rates for FKP and DIP are set to 0.1 and 0.005, respectively.

KernelGAN-FKP. Small patches of size 64×64 are randomly cropped from the LR image for GAN training. We use the WGAN-GP loss [18] and set the gradient penalty to 0.1 to increase training stability. The learning rate is 5e-4 and the batch size is 64. For scale factors 2 and 4, we train the model for 1,000 and 4,000 iterations, respectively, by the Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$.

6.2. Experiments on DIP-FKP

6.2.1 Quantitative Results

Comparison with state-of-the-arts. Average PSNR and SSIM results of different methods are shown in Table 1. We compare the proposed DIP-FKP with bicubic interpolation, RCAN [56], DIP [44], Double-DIP [15, 39] and the upper bound model (non-blind USRNet [53] given ground-truth kernels). Specifically, RCAN is one of the representative bicubic SR oriented models. When the kernel deviates from the predefined bicubic kernel, its performance deteriorates seriously. Double-DIP tries to remedy kernel mismatch by incorporating an untrained fully-connected network (FCN) as the kernel prior. However, its performance is unsatisfactory and worse than DIP that has no kernel prior. In contrast, DIP-FKP incorporates FKP as the kernel prior and improves the performance of DIP by significant margins across all datasets and scale factors. After applying the estimated kernels of DIP-FKP for non-blind SR, the performance could be further improved due to accurate kernel estimation.

Comparison with other kernel priors. Table 2 shows the results on kernel PSNR and image PSNR/SSIM (before and after non-blind SR) when using different kernel priors. "DIP+Softmax" uses a Softmax layer as the kernel prior, which is used to meet the non-negative and sum-to-one constraints on kernel. However, such a simple kernel prior gives rise to poor performance. Double-DIP adds two fully-connected layers before the Softmax layer, but its performance is still unsatisfactory. This actually accords with our analysis in Sec. 4.1 that an untrained network may not be a good kernel prior. For the special case of Gaussian kernel, it is also possible to use a parametric Gaussian generation model as a kernel prior (denoted as "DIP+Parametric Prior"). As we can see, before non-blind SR, using FKP achieves similar image PSNR as using the parametric prior. This is because the quality of generated images is heavily dependent on DIP. However, FKP significantly outperforms the parametric prior in terms of kernel estimation, which leads to better image PSNR after non-blind SR. It is worth pointing out that, unlike the parametric prior, FKP can be

Table 1: Average PSNR/SSIM of different methods on various datasets. Note that due to GPU memory constraints, we crop 960×960 center image patches for DIV2K in kernel estimation. The best and second best results are highlighted in red and blue colors, respectively.

Method	Scale	Set5 [4]	Set14 [52]	BSD100 [33]	Urban100 [23]	DIV2K [1]
Bicubic Interpolation	$\times 2$	26.58/0.8010	24.85/0.6939	25.19/0.6633	22.35/0.6503	26.97/0.7665
RCAN [56]	$\times 2$	26.80/0.8004	24.83/0.6945	25.21/0.6619	22.30/0.6499	26.99/0.7666
DIP [44]	$\times 2$	26.82/0.7518	25.40/0.6868	24.71/0.6508	23.29/0.6749	-
Double-DIP [15]	$\times 2$	24.71/0.6423	22.21/0.5626	23.31/0/5681	21.03/0.5701	-
DIP-FKP (ours)	$\times 2$	30.16/0.8637	27.06/0.7421	26.72/0.7089	24.33/0.7069	-
DIP-FKP + USRNet [53] (ours)	$\times 2$	32.34/0.9308	28.18/0.8088	28.61/0.8206	26.46/0.8203	30.13/0.8686
GT + USRNet [53] (upper bound)	$\times 2$	36.37/0.9508	32.56/0.8945	31.34/0.8772	29.97/0.8954	34.59/0.9268
Bicubic Interpolation	$\times 3$	23.38/0.6836	22.47/0.5884	23.17/0.5625	20.37/0.5378	24.50/0.6806
RCAN [56]	$\times 3$	23.56/0.6802	22.31/0.5801	23.04/0.5506	20.14/0.5247	24.32/0.6712
DIP [44]	$\times 3$	28.14/0.7687	25.19/0.6581	25.25/0.6408	23.22/0.6512	-
Double-DIP [15]	$\times 3$	23.21/0.6535	20.20/0.5071	20.38/0.4499	19.61/0.4993	-
DIP-FKP (ours)	$\times 3$	28.82/0.8202	26.27/0.6922	25.96/0.6660	23.47/0.6588	-
DIP-FKP + USRNet [53] (ours)	$\times 3$	30.78/0.8840	27.76/0.7750	27.29/0.7484	24.84/0.7510	29.03/0.8354
GT + USRNet [53] (upper bound)	$\times 3$	33.95/0.9199	29.91/0.8283	28.82/0.7935	27.22/0.8274	31.79/0.8754
Bicubic Interpolation	$\times 4$	21.70/0.6198	20.86/0.5181	21.95/0.5097	19.13/0.4729	23.01/0.6282
RCAN [56]	$\times 4$	21.86/0.6174	20.37/0.4940	21.71/0.4935	18.60/0.4465	22.69/0.6128
DIP [44]	$\times 4$	27.34/0.7465	25.03/0.6371	24.92/0.6030	22.55/0.6128	-
Double-DIP [15]	$\times 4$	20.99/0.5578	18.31/0.4426	18.57/0.3815	18.15/0.4491	-
DIP-FKP (ours)	$\times 4$	27.77/0.7914	25.65/0.6764	25.15/0.6354	22.89/0.6327	-
DIP-FKP + USRNet [53] (ours)	$\times 4$	29.29/0.8508	26.70/0.7383	25.97/0.6902	23.89/0.7078	27.44/0.7859
GT + USRNet [53] (upper bound)	$\times 4$	31.91/0.8894	28.30/0.7742	27.33/0.7277	25.47/0.7635	29.99/0.8272

Figure 5: Visual results of different methods on synthetic and real-world images for scale factor 4. Estimated/ground-truth kernels are shown on the top right of images. More visual results are provided in the supplementary.

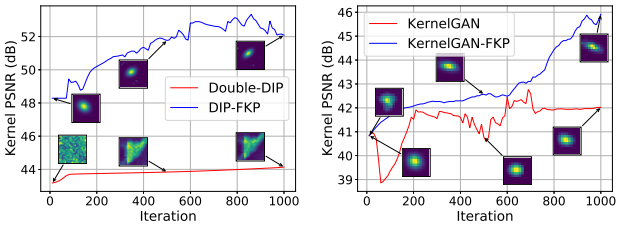
used to model arbitrary kernel distributions as it is trained in an unsupervised way.

Robustness to non-Gaussian kernel and image noise. We add noise to kernels and LR images to test the model robustness to non-Gaussian kernels and image noises, respectively. For non-Gaussian kernels, we apply uniform multiplicative noise (40% of the maximum kernel pixel value, *i.e.*, 0.4) on the kernel and then normalize it to meet the sum-to-one constraint. For image noise, we add noise of level 10 (3.92% of the maximum image pixel value) to the image after blurring and downsampling. As one can see from Table 2, even under heavy kernel corruption, DIP-FKP still produces comparable results, showing good robustness to non-Gaussian kernels. When the image is corrupted by noise, DIP-FKP has a moderate performance drop, but it still outperforms its competitors by a large margin. In this case, we argue that the kernel estimation performance is mainly limited by DIP rather than the proposed FKP.

Model parameter, runtime and memory usage. The total number of parameters of kernel priors in DIP-FKP

Table 2: Average PSNR/SSIM of using different kernel priors. Experiments are conducted on BSD100 [33] when scale factor is 2. Results on non-Gaussian kernel and image noise are also provided.

Method	Kernel PSNR	Image PSNR/SSIM	
		Before Non-Blind SR	After Non-Blind SR
×2			
DIP [44] + Softmax	32.67	23.62/0.5587	23.76/0.5783
Double-DIP [15, 39]	39.98	23.31/0.5681	18.47/0.4441
DIP [44] + Parametric Prior	34.99	**26.76/0.7091**	28.00/0.7682
DIP-FKP (ours)	**46.79**	26.72/0.7089	**28.61/0.8206**
×2, with Non-Gaussian Kernel			
DIP [44] + Softmax	32.84	23.69/0.5629	23.81/0.5877
Double-DIP [15, 39]	39.36	23.29/0.5693	18.25/0.4364
DIP [44] + Parametric Prior	34.50	26.74/0.7096	27.77/0.7631
DIP-FKP (ours)	**44.27**	**26.76/0.7097**	**27.88/0.8019**
×2, with Image Noise of Level 10 (3.92%)			
DIP [44] + Softmax	32.47	23.06/0.5314	23.67/0.5846
Double-DIP [15, 39]	39.89	22.73/0.5322	21.95/0.6011
DIP [44] + Parametric Prior	31.98	26.61/0.6939	27.11/0.7118
DIP-FKP (ours)	**45.20**	**26.66/0.6946**	**27.67/0.7403**



(a) (b)

Figure 6: The intermediate kernel results of Double-DIP, DIP-FKP, KernelGAN and KernelGAN-FKP during optimization. The testing images in (a) and (b) are "*156065*" in BSD100 [33] and "*0812*" in DIV2K [1], respectively.

and Double-DIP are 143K and 641K, respectively. With a lightweight FKP, runtime and memory usage of DIP-FKP on a Tesla V100 GPU for generating a HR image of size $1,024 \times 1,024$ are about 280 seconds and 10.6GB, while Double-DIP needs about 300 seconds and 11.2GB memory for the same setting.

6.2.2 Visual Results

Comparison with state-of-the-arts. The visual results of different methods on synthetic and real-world images are shown in Fig. 5. As one can see, the results of RCAN tends to be blurry, while DIP tends to generate noise-like images. When different kernel priors are incorporated to DIP, both "DIP+Softmax" and Double-DIP fail to generate reasonable kernels, resulting in obvious artifacts such as over-smooth patterns and ringings. In contrast, DIP-FKP produces kernels that are very close to the ground-truths and generates the most visually-pleasant SR results for both synthetic and real-world images.

Intermediate results during optimization. Fig. 6(a) provides an example to show the intermediate kernel results of Double-DIP and DIP-FKP. It can be observed that Double-DIP is randomly initialized and only has a slight improvement during optimization. In contrast, with the incorporation of FKP, DIP-FKP has a good kernel initialization and the kernel is constrained to be in the learned kernel manifold, thereby converging better than Double-DIP.

Table 3: Average PSNR/SSIM of different methods on DIV2K [1]. Note that KernelGAN is not applicable for scale factor 3. Results on small-image datasets are omitted due to weak patch recurrence.

Method	Kernel PSNR	Non-blind PSNR/SSIM
×2		
Bicubic Interpolation	-	26.97/0.7665
RCAN [56]	-	26.99/0.7666
KernelGAN [44]+USRNet [53]	44.95	27.59/0.8162
KernelGAN-FKP + USRNet [53] (ours)	**47.78**	**28.69/0.8567**
GT + USRNet [53] (upper bound)	-	34.59/0.9268
×4		
Bicubic Interpolation	-	23.20/0.6329
RCAN [56]	-	23.20/0.6310
KernelGAN [44]+USRNet [53]	57.26	23.69/0.6539
KernelGAN-FKP + USRNet [53] (ours)	**60.61**	**25.46/0.7229**
GT + USRNet [53] (upper bound)	-	29.46/0.8069
×2, with Non-Gaussian Kernel		
Bicubic Interpolation	-	26.96/0.7662
RCAN [56]	-	26.98/0.7663
KernelGAN [44]+USRNet [53]	43.27	27.00/0.8030
KernelGAN-FKP + USRNet [53] (ours)	**44.72**	**27.40/0.8334**
GT + USRNet [53] (upper bound)	-	34.59/0.9272
×2, with Image Noise of Level 10 (3.92%)		
Bicubic Interpolation	-	26.65/0.7258
RCAN [56]	-	26.22/0.6627
KernelGAN [44]+USRNet [53]	44.55	28.53/0.8281
KernelGAN-FKP + USRNet [53] (ours)	**47.13**	**29.58/0.8303**
GT + USRNet [53] (upper bound)	-	31.27/0.8497

6.3. Experiments on KernelGAN-FKP

6.3.1 Quantitative Results

Comparison with state-of-the-arts. In Table 3, we compare the average kernel and image PSNR/SSIM of the proposed KernelGAN-FKP with bicubic interpolation, RCAN [56], KernelGAN [3] and the upper bound model (non-blind SR model USRNet [53] given ground-truth kernels). As one can see, RCAN has similar performance to naive bicubic interpolation due to kernel mismatch. KernelGAN is able to deal with different kernels and achieves better results than RCAN. Compared with KernelGAN, KernelGAN-FKP obtains further kernel PSNR gains of 2.83dB and 3.35dB for scale factors 2 and 4, respectively, which incur 1.1dB and 1.77dB image PSNR improvements after non-blind SR. The improvements can be attributed to the incorporation of FKP.

Robustness to non-Gaussian kernel and image noise. Table 3 also shows the results of different methods when given non-Gaussian kernels or noisy images. The experiment details are similar to DIP-FKP. As one can see, although all methods suffer from performance drops,

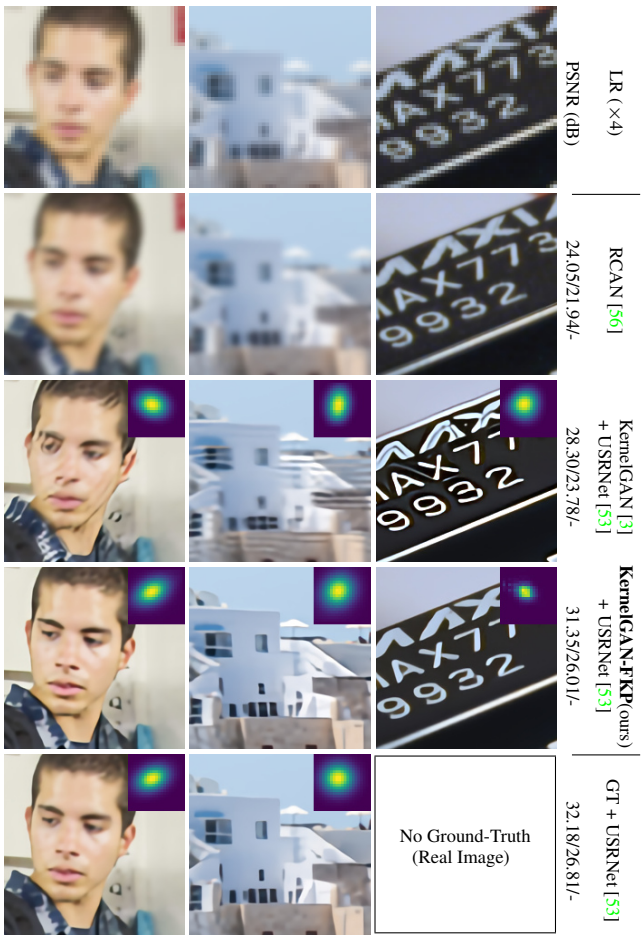


Figure 7: Visual results of different methods on synthetic and real-world images for scale factor 4. Estimated/ground-truth kernels are shown on the top right of images. More visual results are provided in the supplementary.

KernelGAN-FKP still outperforms other methods by substantial margins. In particular, KernelGAN-FKP delivers comparable performance in handling image noise. The underlying reason might be that noise injection in the generator of GAN can help to circumvent over-fitting [14].

Model parameter, runtime and memory usage. From KernelGAN to KernelGAN-FKP, the deep linear network is replaced by the proposed FKP, which reduces the total number of generator parameters from 151K to 143K. On a Tesla V100 GPU, runtime and memory usage of KernelGAN are about 93 seconds and 1.3GB, respectively, which are independent of images sizes. As for KernelGAN-FKP, it requires 90 seconds and 1.5GB memory.

6.3.2 Visual Results

Comparison with state-of-the-arts. Fig. 7 shows the visual comparison of different methods on synthetic and real-world images. It can be seen that RCAN tends to generate blurry images that are only slightly better than LR images. This is because the assumed bicubic kernel is sharper than the ground-truth kernels. Instead, KernelGAN tends to produce smoother kernels, leading to over-sharpen edges. In comparison, KernelGAN-FKP generates more accurate blur kernels and results in less artifacts for both synthetic and real-world images.

Intermediate results during optimization. The intermediate kernel results of KernelGAN-FKP and KernelGAN are shown in Fig. 6(b). As one can see, KernelGAN-FKP converges more stably and better than KernelGAN, which oscillates during optimization. This suggests that the incorporation of FKP can increase the training stability and improve kernel estimation performance.

6.4. DIP-FKP v.s. KernelGAN-FKP

While DIP-FKP and KernelGAN-FKP outperform Double-DIP and KernelGAN, respectively, it is interesting to compare their differences. Since DIP-FKP jointly estimates the kernel and HR image, it requires more memory. Also, it generally generates better kernel estimations and has more stable convergence for small images. On the contrary, KernelGAN-FKP requires much less memory as it only needs to optimize the kernel, but it does not perform well for small images and large scale factors because it needs to re-downscale the LR image.

7. Conclusion

In this paper, we propose a flow-based kernel prior (FKP) for kernel distribution modeling and incorporate it into existing blind SR methods for better kernel and image estimation performance. FKP learns an invertible mapping between the complex kernel distribution and a tractable latent variable distribution based on normalization flow blocks. Its training is unsupervised and thus FKP is applicable for arbitrary kernel assumptions. When used as a kernel prior, FKP freezes its parameters and optimizes the latent variable in the network input space. Therefore, reasonable kernels are guaranteed for initialization and along optimization. FKP can be easily incorporated into existing kernel estimation models, such as Double-DIP and KernelGAN, by replacing their kernel modeling modules. Extensive experiments on synthetic LR images and real-world images demonstrate that FKP significantly improves the accuracy of kernel estimation and thus leads to state-of-the-art blind SR results.

Acknowledgements This work was partially supported by the ETH Zurich Fund (OK), a Huawei Technologies Oy (Finland) project, the China Scholarship Council and a Microsoft Azure grant. Special thanks goes to Yijue Chen.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017. 5, 6, 7

[2] Isabelle Begin and FR Ferrie. Blind super-resolution using a learning-based approach. In *IEEE Conference on International Conference on Pattern Recognition*, pages 85–89, 2004. 2

[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. 1, 2, 4, 5, 7, 8

[4] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie line Alberi Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *British Machine Vision Conference*, pages 135.1–135.10, 2012. 5, 6

[5] Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. Enhancing sparsity by reweighted $\ell 1$ minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905, 2008. 2

[6] Tony F Chan and Chiu-Kwong Wong. Total variation blind deconvolution. *IEEE Transactions on Image Processing*, 7(3):370–375, 1998. 2

[7] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007. 2

[8] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 2, 3

[9] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 2, 3

[10] 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, 2014. 1

[11] Netalee Efrat, Daniel Glasner, Alexander Apartsin, Boaz Nadler, and Anat Levin. Accurate blur models vs. image priors in single image super-resolution. In *IEEE Conference on International Conference on Computer Vision*, pages 2832–2839, 2013. 1, 2

[12] 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. 2

[13] Sina Farsiu, Dirk Robinson, Michael Elad, and Peyman Milanfar. Advances and challenges in super-resolution. *International Journal of Imaging Systems and Technology*, 14(2):47–57, 2004. 2

[14] Ruili Feng, Deli Zhao, and Zhengjun Zha. On noise injection in generative adversarial networks. *arXiv preprint arXiv:2006.05891*, 2020. 8

[15] Yossi Gandelsman, Assaf Shocher, and Michal Irani. Double-dip": Unsupervised image decomposition via coupled deep-image-priors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 11026–11035, 2019. 1, 2, 3, 5, 6, 7

[16] Daniel Glasner, Shai Bagon, and Michal Irani. Super-resolution from a single image. In *IEEE Conference on International Conference on Computer Vision*, pages 349–356, 2009. 1, 2, 4

[17] Jinjin Gu, Hannan Lu, Wangmeng Zuo, and Chao Dong. Blind super-resolution with iterative kernel correction. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1604–1613, 2019. 1, 2, 5

[18] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017. 5

[19] Yong Guo, Jian Chen, Jingdong Wang, Qi Chen, Jiezhang Cao, Zeshuai Deng, Yanwu Xu, and Mingkui Tan. Closed-loop matters: Dual regression networks for single image super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5407–5416, 2020. 1

[20] 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. 2

[21] 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. 2, 5

[22] Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. Neural autoregressive flows. *arXiv preprint arXiv:1804.00779*, 2018. 2

[23] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015. 5, 6

[24] Priyank Jaini, Kira A Selby, and Yaoliang Yu. Sum-of-squares polynomial flow. *arXiv preprint arXiv:1905.02325*, 2019. 2

[25] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, 2016. 1

[26] Sungwon Kim, Sang-gil Lee, Jongyoon Song, Jaehyeon Kim, and Sungroh Yoon. Flowavenet: A generative flow for raw audio. *arXiv preprint arXiv:1811.02155*, 2018. 2

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[28] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018. 2

[29] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751, 2016. 2

[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 *IEEE Conference on Computer Vision*

and Pattern Recognition, pages 4681–4690, 2017. 1

[31] Ce Liu and Deqing Sun. On bayesian adaptive video super resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(2):346–360, 2013. 2

[32] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Srflow: Learning the super-resolution space with normalizing flow. In *European Conference on Computer Vision*, pages 715–732, 2020. 2

[33] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *IEEE Conference on International Conference on Computer Vision*, pages 416–423, 2001. 5, 6, 7

[34] Sachit Menon, Alexandru Damian, Shijia Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2437–2445, 2020. 3

[35] Tomer Michaeli and Michal Irani. Nonparametric blind super-resolution. In *IEEE Conference on International Conference on Computer Vision*, pages 945–952, 2013. 1, 2, 4, 5

[36] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy, and Ping Luo. Exploiting deep generative prior for versatile image restoration and manipulation. *arXiv preprint arXiv:2003.13659*, 2020. 2

[37] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, pages 2338–2347, 2017. 2

[38] Yajun Qiu, Ruxin Wang, Dapeng Tao, and Jun Cheng. Embedded block residual network: A recursive restoration model for single-image super-resolution. In *IEEE Conference on International Conference on Computer Vision*, pages 4180–4189, 2019. 1

[39] Dongwei Ren, Kai Zhang, Qilong Wang, Qinghua Hu, and Wangmeng Zuo. Neural blind deconvolution using deep priors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3341–3350, 2020. 1, 2, 3, 5, 6, 7

[40] Gernot Riegler, Samuel Schulter, Matthias Ruther, and Horst Bischof. Conditioned regression models for non-blind single image super-resolution. In *IEEE Conference on International Conference on Computer Vision*, pages 522–530, 2015. 1, 5

[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, 2015. 2, 5

[42] Assaf Shocher, Nadav Cohen, and Michal Irani. "zero-shot" super-resolution using deep internal learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3118–3126, 2018. 5

[43] Jae Woong Soh, Sunwoo Cho, and Nam Ik Cho. Meta-transfer learning for zero-shot super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3516–3525, 2020. 5

[44] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018. 1, 2, 3, 5, 6, 7

[45] Roman Vershynin. *Random Vectors in High Dimensions*, page 38–69. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018. 3

[46] Qiang Wang, Xiaoou Tang, and Harry Shum. Patch based blind image super resolution. In *IEEE Conference on International Conference on Computer Vision*, pages 709–716, 2005. 2

[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 *European Conference on Computer Vision Workshops*, pages 701–710, 2018. 1

[48] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 5

[49] Yu-Syuan Xu, Shou-Yao Roy Tseng, Yu Tseng, Hsien-Kai Kuo, and Yi-Min Tsai. Unified dynamic convolutional network for super-resolution with variational degradations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 12496–12505, 2020. 5

[50] Chih-Yuan Yang, Chao Ma, and Ming-Hsuan Yang. Single-image super-resolution: A benchmark. In *European Conference on Computer Vision*, pages 372–386, 2014. 1

[51] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *IEEE Conference on International Conference on Computer Vision*, pages 4541–4550, 2019. 2

[52] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International Conference on Curves and Surfaces*, pages 711–730, 2010. 5, 6

[53] Kai Zhang, Luc Van Gool, and Radu Timofte. Deep unfolding network for image super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3217–3226, 2020. 4, 5, 6, 7, 8

[54] Kai Zhang, Jingyun Liang, Luc Van Gool, and Radu Timofte. Designing a practical degradation model for deep blind image super-resolution. *arXiv preprint arXiv:2103.14006*, 2021. 1, 5

[55] 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. 1, 2, 5

[56] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *European Conference on Computer Vision*, pages 286–301, 2018. 1, 5, 6, 7, 8

[57] Maria Zontak and Michal Irani. Internal statistics of a single natural image. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 977–984, 2011. 1, 2, 4