Unfolded Deep Kernel Estimation for Blind Image Super-resolution

Hongyi Zheng[1] and Hongwei Yong[1] and Lei Zhang[1,2,⋆]

[1]The Hong Kong Polytechnic University, [2]OPPO Research
{cshzheng,cshyong,cslzhang}@comp.polyu.edu.hk

Abstract. Blind image super-resolution (BISR) aims to reconstruct a high-resolution image from its low-resolution counterpart degraded by unknown blur kernel and noise. Many deep neural network based methods have been proposed to tackle this challenging problem without considering the image degradation model. However, they largely rely on the training sets and often fail to handle images with unseen blur kernels during inference. Deep unfolding methods have also been proposed to perform BISR by utilizing the degradation model. Nonetheless, the existing deep unfolding methods cannot explicitly solve the data term of the unfolding objective function, limiting their capability in blur kernel estimation. In this work, we propose a novel unfolded deep kernel estimation (UDKE) method, which, for the first time to our best knowledge, explicitly solves the data term with high efficiency. The UDKE based BISR method can jointly learn image and kernel priors in an end-to-end manner, and it can effectively exploit the information in both training data and image degradation model. Experiments on benchmark datasets and real-world data demonstrate that the proposed UDKE method could well predict complex unseen non-Gaussian blur kernels in inference, achieving significantly better BISR performance than state-of-the-art. The source code of UDKE is available at https://github.com/natezhenghy/UDKE.

Keywords: blind image super-resolution, blur kernel estimation, unfolding method

1 Introduction

Blind image super-resolution (BISR), which aims to reconstruct a high-resolution (HR) image from its low-resolution (LR) counterpart without knowing the degradation kernel and noise, is a very challenging computer vision problem [33]. The degradation process from an HR image to an LR image can be expressed as:

$$\mathbf{Y} = (\mathbf{K} \circledast \mathbf{X}) \downarrow_s + \mathbf{n} \tag{1}$$

⋆ Corresponding author.

⋆⋆This work is supported by the Hong Kong RGC RIF grant (R5001-18) and the PolyU-OPPO Joint Innovation Lab.

where $\mathbf{X}$ is the HR image, $\mathbf{Y}$ is its observed LR counterpart, $\mathbf{K}$ is the blur kernel, $\circledast$ is the 2D convolution operator, $\downarrow_s$ is the downsampling operator with scaling factor s, and $\mathbf{n}$ is the additive white Gaussian noise.

A variety of classical methods have been proposed to tackle the BISR problem [22,14,26]. The interpolation-based methods, such as bilinear and bicubic interpolation, are efficient to implement, whereas they have poor results for BISR. Model based methods employ a degradation model (*e.g.*, Eq. 1) to constrain the fidelity between the predicted SR image and the LR input, and exploit image priors to regularize the solution. Some representative methods include Maximum a Posterior [9], recurrence prior [12], *etc*. Learning based methods aim to learn image priors and mappings between the LR input and HR image from the training data, *e.g.*, dictionary learning [30] and patch based learning [4].

With the rapid development of deep learning, the deep neural network (DNN) based methods have become prevalent in the research of super-resolution (SR) and shown highly competitive performance [11]. However, most of the existing DNN based methods focus on the non-blind SR tasks, where the degradation process is simply assumed to be bicubic downsampling [38], or direct downsampling after blurred by fixed isotropic Gaussian kernels [37]. In real-world applications, however, the image degradation process is much more complex due to the unknown varying blur kernels and the corrupted noise, and these non-blind SR methods often fail. Therefore, DNN based BISR methods have been later proposed. Zhou *et al*. [40] analyzed blur kernels in real LR images by using the dark channel priors [23], and built a BISR blur kernel dataset. The KGAN (Kernel-GAN) [5] employs a generative adversarial network (GAN), which is trained online during the inference stage, to estimate the blur kernel with some presumed priors, *e.g.*, Gaussian prior. However, these methods rely heavily on training data and they do not consider the LR image degradation process. They often fail to handle images degraded with unseen kernels during inference.

To address the limitations of the above purely data-driven methods, some deep unfolding methods have been proposed to encode the degradation model into the learning process. With the input LR image $\mathbf{Y}$, the objective function of deep unfolding methods can be generally depicted as:

$$\min_{\{\mathbf{K},\mathbf{X}\}} \frac{1}{2\sigma^2}\|(\mathbf{K}\circledast\mathbf{X})\downarrow_s - \mathbf{Y}\|_2^2 + \lambda_{\mathbf{X}}\boldsymbol{\psi}(\mathbf{X}) + \lambda_{\mathbf{K}}\boldsymbol{\phi}(\mathbf{K}) \tag{2}$$

where $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ represent the priors on $\mathbf{X}$ and $\mathbf{K}$, $\lambda_{\mathbf{X}}$ and $\lambda_{\mathbf{K}}$ are the balance parameters, and σ is the noise level. Eq. 2 can be divided into two components, namely data term ($\|(\mathbf{K}\circledast\mathbf{X})\downarrow_s - \mathbf{Y}\|_2^2$) and prior term ($\lambda_{\mathbf{X}}\boldsymbol{\psi}(\mathbf{X}) + \lambda_{\mathbf{K}}\boldsymbol{\phi}(\mathbf{K})$). Typical deep unfolding methods include IKC [13] and DAN [20], which employ an iterative framework to unfold Eq. 2 and perform BISR. Nevertheless, the data term is difficult to solve under the deep learning framework, and these methods do not solve the data term explicitly in kernel estimation, which limits their BISR performance (please refer to Sec. 3.1 for more discussions).

In this work, we propose a novel unfolded deep kernel estimation method, namely UDKE, by explicitly solving the data term under the deep learning framework. Based on UDKE, we implement a BISR framework, which, to our

best knowledge, is the first deep unfolding framework that fully unfolds the objective function in Eq. 2. UDKE effectively and efficiently encodes the knowledge of the image degradation model into the DNN architecture and learns priors of images and blur kernels jointly in an end-to-end manner. By explicitly solving the objective function of BISR with learned priors during inference, it can efficiently estimate the unseen complex non-Gaussian blur kernels, surpassing existing kernel estimation methods by a large margin.

We extensively evaluate the proposed UDKE based BISR framework on multiple BISR benchmarks as well as real-world data. It records new state-of-the-art BISR performance, while costs only 1% the inference time of leading online-learning based methods (*e.g.*, DIP-FKP [17]).

2 Related Work

Traditional BISR methods. Traditional BISR methods can be categorized into model-based methods [14,22,26,19,29] and learning based methods [4,10]. The former adopts an image degradation model and image priors to estimate the desired HR image. He *et al*. [14] proposed a soft Maximum a Posteriori based method to alternatively perform blur kernel estimation and HR image reconstruction. Michaeli *et al*. [22] proposed a non-parametric BISR model that exploits the inherent recurrence property of image patches. Shao *et al*. [26] employed the convolution consistency prior to estimate the blur kernel. These methods follow the constraints of the degradation model and have good interpretability; however, their BISR performance is usually limited because of the relatively weak handcrafted priors.

Equipped with a training dataset, learning based methods aim to learn from it more effective image priors and/or LR-to-HR image mappings. Begin *et al*. [4] designed a framework to estimate camera parameters from the LR image, and estimate the HR image. Corduneanu *et al*. [10] proposed a spatial-variant BISR method, which learns a set of linear blur filters from the neighboring pixels. Liu *et al*. [19] developed a sparse representation based method for BISR, which utilizes the image self-similarity prior to learn an over-complete dictionary to represent the HR image. These methods, by learning from external training data, exhibit better BISR performance than model-based methods; however, their performance will drop a lot when the degradation parameters (*e.g.*, blur kernel) of test data are much different from that of the training data.

Direct DNN based BISR methods. DNN based methods have become the mainstream of BISR research, outperforming the traditional learning-based methods by a large margin. Many DNN based BISR methods directly perform BISR without performing blur kernel estimation. CinCGAN [34] converts the LR image with unknown degradation into the bicubic degradation domain and then performs non-blind SR. Degradation GAN [8] learns the degradation process implicitly via a GAN to assist SR task. DASR [28] learns abstract representations of various degradations, and then adopts a DNN to perform the SR task. Kligler *et al*. [5] proposed the KGAN method, which trains a GAN on the LR

image to estimate the blur kernel based on Gaussian prior and patch recurrence property. Liang *et al*. [17] enhanced KGAN with flow-based prior, which works well when the blur kernel follows Gaussian assumption. These methods do not consider the image degradation process and largely rely on the training dataset in model learning. Their performance would deteriorate when encounter unseen degradation parameters (*e.g*., blur kernel) in inference.

Deep unfolding BISR methods. To address the limitations of direct DNN based BISR methods, a few deep unfolding BISR methods have been proposed. These methods share an iterative framework to unfold the objective function in Eq. 2. By alternatively estimating the blur kernel and the super-resolved image, they aim to utilize the image degradation model to assist the BISR task. The early deep unfolding methods [3,25] utilize the maximum a posterior framework to perform the image denoising task. Zhang *et al*. [35] proposed a deep unfolding framework for non-blind SR by using the Half Quadratic Splitting algorithm to unfold the objective function. For BISR, Gu *et al*. [13] proposed the IKC method, which adopts a DNN to iteratively correct the blur kernel estimation in an implicit dimension-reduced space. Luo *et al*. [20] proposed a DAN approach, which iteratively estimates the blur kernel and super-resolved image with the help of conditional residual block.

However, all the previous unfolding deep methods do not explicitly solve the data term in kernel estimation, thus they do not fully unfold the objective. This limits their capability to estimate complex unseen kernels, and they fail to address the limits of direct DNN based methods properly. Actually, their performance might be even worse than those direct DNN based methods when encounter unseen kernels during inference (*e.g*., IKC and DAN only work on Gaussian kernels). We propose an effective and efficient kernel estimation method by explicitly solving the data term and hence truly unfolding the whole objective function under the deep learning framework. Our proposed method can estimate more complex unseen non-Gaussian blur kernels in inference.

3 Methodology

3.1 Problems of previous deep unfolding BISR methods

As described in Eq. 2, the objective function of deep unfolding methods can be divided into the data term ($\|(\mathbf{K}\circledast\mathbf{X})\downarrow_s-\mathbf{Y}\|_2^2$) and the prior term ($\lambda_{\mathbf{X}}\boldsymbol{\psi}(\mathbf{X})+\lambda_{\mathbf{K}}\boldsymbol{\phi}(\mathbf{K})$). According to [35], the data term enforces physical constraints on the image degradation process, and it should be solved explicitly to enable an unfolding method to estimate unseen (different from those in training) blur kernels during inference. However, all the previous deep unfolding methods [13,20] employ DNNs to estimate the blur kernel implicitly without solving the data term explicitly. Therefore, they do not fully unfold the objective function to utilize the information embedded in the image degradation model. As a result, most of these methods simply assume Gaussian blur kernels in BISR and they have limited generalization capability to more complex non-Gaussian kernels.

The major reason that the previous methods do not explicitly unfold the data term lies in that the available solutions, which are developed in traditional unfolding methods, are hard to be incorporated into the deep learning framework. In traditional methods, the data term can be solved by either numerical methods or analytical methods. The numerical methods such as the Alternating Direction Method of Multipliers [7] solve the data term iteratively. Such iterative methods work well in traditional unsupervised BISR methods; however, they are too time-consuming to use in deep learning framework, which requires training on a large amount of data. On the other hand, the analytical methods such as the Least Squares Method (LSM)[1] can provide an analytical solution of the data term. However, the image-to-column (im2col) operation required in LSM would increase the memory-overhead by thousands of times, which is not acceptable in deep learning framework. One way to waive the im2col operation is to transform the original problem into the frequency domain by the Fast Fourier Transform. Such methods have been used in the non-blind SR task [35], where the blur kernel is known. In BISR, however, the support set of the unknown blur kernels is much smaller than that of images, making the explicit solution in frequency domain hard to achieve.

In this work, we investigate deeply this challenging problem, and propose an effective yet efficient method to explicitly solve the data term with minimal memory overhead under the deep learning framework.

3.2 Unfolded deep kernel estimation based BISR framework

The framework of our proposed unfolded deep kernel estimation (UDKE) based BISR method is shown in Fig. 1. Suppose we are given N training triplets $\{\mathbf{Y}_i,\mathbf{Y}_i^{gt},\mathbf{K}_i^{gt}\}$, where $\mathbf{Y}_i$ is the i^{th} observed LR image, and $\mathbf{Y}_i^{gt}$ and $\mathbf{K}_i^{gt}$ are the ground-truth HR image and ground-truth blur kernel, respectively. In order to accommodate the unfolding objective in Eq. 2 into an end-to-end training framework, we rewrite it into a bi-level optimization problem as follows:

$$\min_{\{\theta_{\boldsymbol{\psi}},\theta_{\boldsymbol{\phi}}\}} \frac{1}{N}\sum_{i=1}^{N} L_{\mathbf{X}}(\mathbf{X}_i,\mathbf{Y}_i^{gt})+\gamma L_{\mathbf{K}}(\mathbf{K}_i,\mathbf{K}_i^{gt}) \tag{3a}$$

$$\text{s.t. } \{\mathbf{K}_i,\mathbf{X}_i\}=\operatorname{argmin}_{\mathbf{K},\mathbf{X}} \frac{1}{2\sigma_i^2}\|(\mathbf{K}\circledast\mathbf{X})\downarrow_s-\mathbf{Y}_i\|_2^2+\lambda_{\mathbf{K}}\boldsymbol{\phi}(\mathbf{K})+\lambda_{\mathbf{X}}\boldsymbol{\psi}(\mathbf{X}) \tag{3b}$$

where $\mathbf{X}_i$ and $\mathbf{K}_i$ are the predicted HR image and blur kernel; $L_{\mathbf{X}}(\cdot,\cdot)$ and $L_{\mathbf{K}}(\cdot,\cdot)$ are the loss functions, γ is a trade-off parameter; $\theta_{\boldsymbol{\psi}}$ and $\theta_{\boldsymbol{\phi}}$ denote the parameters of deep priors (*i.e.*, DNNs) $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$.

In the above bi-level optimization problem, Eq. 3a describes its backward pass, where the parameters (of DNNs) $\theta_{\boldsymbol{\psi}}$ and $\theta_{\boldsymbol{\phi}}$ implicitly embed the deep priors and they are updated based on the losses $L_{\mathbf{X}}$ and $L_{\mathbf{K}}$. Eq. 3b describes the forward pass of the framework, which takes the LR observation $\mathbf{Y}_i$ as input to estimate the blur kernel $\mathbf{K}_i$ and the HR image $\mathbf{X}_i$ with the learned deep priors. For the convenience of expression, we omit the subscript "i" in the following development. Eq. 3b can be split into the following two sub-problems by the

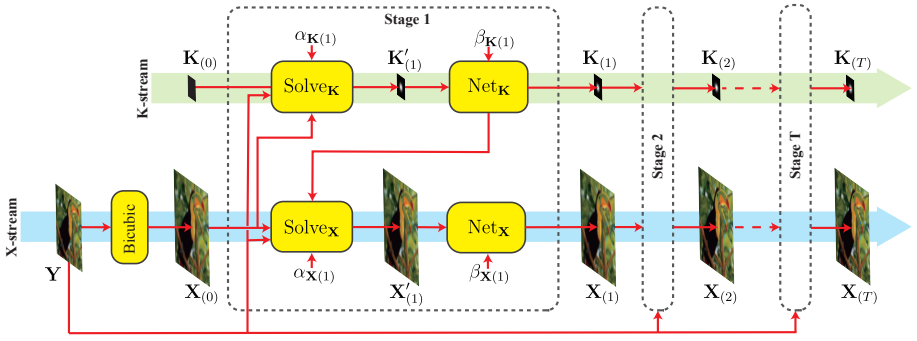


Fig. 1: The overall architecture of our UDKE based BISR framework.

Half Quadratic Splitting (HQS) algorithm:

$$\min_{\{\mathbf{X},\mathbf{X}'\}} \frac{1}{2\sigma^2}\|(\mathbf{K}\circledast\mathbf{X}')\downarrow_s-\mathbf{Y}\|_2^2+\lambda_{\mathbf{X}}\psi(\mathbf{X})+\frac{\mu_{\mathbf{X}}}{2}\|\mathbf{X}-\mathbf{X}'\|_2^2 \tag{4a}$$

$$\min_{\{\mathbf{K},\mathbf{K}'\}} \frac{1}{2\sigma^2}\|(\mathbf{K}'\circledast\mathbf{X})\downarrow_s-\mathbf{Y}\|_2^2+\lambda_{\mathbf{K}}\phi(\mathbf{K})+\frac{\mu_{\mathbf{K}}}{2}\|\mathbf{K}-\mathbf{K}'\|_2^2 \tag{4b}$$

where $\mathbf{X}'$ and $\mathbf{K}'$ are auxiliary variables; $\mu_{\mathbf{X}}$ and $\mu_{\mathbf{K}}$ are penalty parameters.

Eqs. 4a and 4b can be solved iteratively. Particularly, in the t-th iteration we can solve the auxiliary variables $\mathbf{K}'_{(t)}$ and $\mathbf{X}'_{(t)}$ with analytical solutions, while the two mapping DNNs, denoted by Net$_{\mathbf{K}}$ and Net$_{\mathbf{X}}$, map the pre-priors $\mathbf{K}'_{(t)}$ and $\mathbf{X}'_{(t)}$ to post-priors $\mathbf{K}_{(t)}$ and $\mathbf{X}_{(t)}$ with the implicitly embedded priors:

$$\begin{aligned}\mathbf{K}'_{(t)}&=\text{Solve}_{\mathbf{K}}(\mathbf{Y},\mathbf{K}_{(t\text{-}1)},\mathbf{X}_{(t\text{-}1)},\alpha_{\mathbf{K}}) \\ &=\text{argmin}_{\mathbf{K}^\star}\tfrac{1}{2}\|(\mathbf{K}^\star\circledast\mathbf{X}_{(t\text{-}1)})\downarrow_s-\mathbf{Y}\|_2^2+\tfrac{\alpha_{\mathbf{K}}}{2}\|\mathbf{K}^\star-\mathbf{K}_{(t\text{-}1)}\|_2^2\end{aligned} \tag{5a}$$

$$\mathbf{K}_{(t)}=\text{Net}_{\mathbf{K}}(\mathbf{K}'_{(t)},\beta_{\mathbf{K}})=\text{argmin}_{\mathbf{K}^\star}\ \phi(\mathbf{K}^\star)+\tfrac{\beta_{\mathbf{K}}}{2}\|\mathbf{K}'_{(t)}-\mathbf{K}^\star\|_2^2 \tag{5b}$$

$$\begin{aligned}\mathbf{X}'_{(t)}&=\text{Solve}_{\mathbf{X}}(\mathbf{Y},\mathbf{K}_{(t)},\mathbf{X}_{(t\text{-}1)},\alpha_{\mathbf{X}}) \\ &=\text{argmin}_{\mathbf{X}^\star}\tfrac{1}{2}\|(\mathbf{K}_{(t)}\circledast\mathbf{X}^\star)\downarrow_s-\mathbf{Y}\|_2^2+\tfrac{\alpha_{\mathbf{X}}}{2}\|\mathbf{X}^\star-\mathbf{X}_{(t\text{-}1)}\|_2^2\end{aligned} \tag{5c}$$

$$\mathbf{X}_{(t)}=\text{Net}_{\mathbf{X}}(\mathbf{X}'_{(t)},\beta_{\mathbf{X}})=\text{argmin}_{\mathbf{X}^\star}\ \psi(\mathbf{X}^\star)+\tfrac{\beta_{\mathbf{X}}}{2}\|\mathbf{X}'_{(t)}-\mathbf{X}^\star\|_2^2 \tag{5d}$$

where $\{\alpha_{\mathbf{K}},\alpha_{\mathbf{X}},,\beta_{\mathbf{K}}\beta_{\mathbf{X}}\}=\{\mu_{\mathbf{K}}\sigma^2,\mu_{\mathbf{X}}\sigma^2,\frac{\mu_{\mathbf{K}}}{\lambda_{\mathbf{K}}},\frac{\mu_{\mathbf{X}}}{\lambda_{\mathbf{X}}}\}$.

The architecture of our UKDE based BISR framework is built from the unfolded equations Eqs. 5a∼5d. It has two branches. The kernel estimation branch corresponds to Eqs. 5a∼5b and is represented as the K-stream in Fig. 1, which will be discussed in Section. 3.3. Eqs. 5c∼5d super-resolve the HR image with the estimated kernel, and they are represented as the X-stream in Fig. 1, which will be discussed in Section 3.4.

3.3 K-stream: unfolded explicit kernel estimation

In this section, we elaborate in detail the proposed novel kernel estimation method, which corresponds to the K-stream in Fig. 1. The first step is to explicitly solve Eq. 5a (data term), which is represented as Solve$_{\mathbf{K}}$ in Fig. 1. It takes

$\mathbf{Y}$ and the estimations of $\mathbf{K}$ and $\mathbf{X}$ in previous stage as inputs, then explicitly solves Eq. 5a to update $\mathbf{K}'$. As the dimension of $\mathbf{K}$ is much lower than that of $\mathbf{X}$, this system is over-determined and can be solved by the LSM method [1]. Denote by $\boldsymbol{U}_a$ the im2col operator with block size a, and by $\boldsymbol{P}_{\frac{a-1}{2}}$ the circular padding operator with padding size of $\frac{a-1}{2}$. Let $\mathfrak{X}=\boldsymbol{U}_k(\boldsymbol{P}_{\frac{a-1}{2}}(\mathbf{X}))$ and $\mathfrak{Y}=\boldsymbol{U}_k(\mathbf{Y})$, then Eq. 5a can be written as:

$$\text{argmin}_{\mathbf{k}^\star}\ \tfrac{1}{2}\|\mathfrak{M}_s\mathfrak{X}\mathbf{k}-\mathfrak{Y}\|_2^2+\tfrac{\alpha_\mathbf{K}}{2}\|\mathbf{k}^\star-\mathbf{k}\|_2^2 \tag{6}$$

where $\mathbf{k}^\star=\text{vec}(\mathbf{K}^\star)$, $\mathbf{k}=\text{vec}(\mathbf{K})$, $\text{vec}(\cdot)$ is the vectorization operator, and $\mathfrak{M}_s$ is the matrix representation of the downsampling operator $\downarrow_s$ with scale factor s.

By taking the derivative of Eq. 6 w.r.t. $\mathbf{k}^\star$ and letting the derivative be zero, we can obtain the closed-form solution of $\mathbf{K}'$ as follows:

$$\mathbf{K}'=\text{vec}^{-1}\{(\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}+\alpha_\mathbf{K}\mathbf{I})^{-1}(\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{Y}+\alpha_\mathbf{K}\mathbf{k})\} \tag{7}$$

where $\text{vec}^{-1}(\cdot)$ reverses the vectorization operator $\text{vec}(\cdot)$. However, $\mathfrak{X}$ has a size of $C\times h\times w\times k\times k$, which is much larger than $\mathbf{X}$ of size $C\times h\times w$, where C, h, w, and k are channel number, height, width and blur kernel size of the super-resolved image. It is too memory-consuming to directly compute Eq. 7 in practice. In the rest of this section, we will elaborate the proposed memory-efficient solution to tackle this problem.

It can be seen that the size of $\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}$ is $C\times k\times k\times k\times k$ and $k\ll h$, $k\ll w$. We propose an efficient solution to calculate $\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}$ from $\mathbf{X}$ without storing $\mathfrak{X}$ or $\mathfrak{Y}$, reducing significantly the memory consumption. (Note that $\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{Y}$ can be regarded as a special case of $\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}$, where $\mathfrak{Y}=\mathfrak{M}_s\mathfrak{X}$). Denote by $\boldsymbol{U}_k$ the im2col operator with block size k, and by $\boldsymbol{P}_{\frac{k-1}{2}}$ the circular padding operator with padding size $\frac{k-1}{2}$. The element at (x,y) in $\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}$ can be calculated through dilated convolution between the x^{th} and the y^{th} im2col blocks in $\boldsymbol{U}_k\circ\boldsymbol{P}_{\frac{k-1}{2}}(\mathbf{X})$, where $\circ$ is the notation of function composition.

Unfortunately, calculating $h\times w$ elements in $\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}$ requires $h\times w$ convolution operations, and each of them is based on a unique pair of kernel and feature maps, which is too time-consuming. Thus, we have to convert them into parallel operations to utilize the modern parallel computing library such as CUDA. This can be done by convolving $\boldsymbol{P}_{k-1}(\mathbf{X})$ with a series of dilated feature maps, each has a unique dilation pattern. Generally speaking, with the scale factor s, there are s^2 dilation patterns. We use 2-dimensional indices to arrange these dilated feature maps, denoted by $\hat{\mathbf{X}}^{(i,j)}$, by using the following rule:

$$\hat{\mathbf{X}}^{(i,j)}:\begin{cases}\hat{\mathbf{X}}^{(i,j)}_{(x,y)}=\mathbf{X}_{(x,y)} & x\%i=0\ \&\ y\%j=0\\ \hat{\mathbf{X}}^{(i,j)}_{(x,y)}=0 & otherwise\end{cases} \tag{8}$$

where $\%$ is the modulo operator, $i=\{0,1...s-1\}$ and $j=\{0,1...s-1\}$. Convolution operations between $\boldsymbol{P}_{k-1}(\mathbf{X})$ and $\hat{\mathbf{X}}^{(i,j)}$ result in s^2 feature maps. Then we merge them into a single feature map, denoted by $\mathbf{F}$, with the help of a mapping

function $\boldsymbol{f}$. $\boldsymbol{f}$ and $\mathbf{F}$ are defined as follows:

$$\boldsymbol{f}{:}(x,y)\rightarrow((\lfloor\frac{k-1}{2}\rfloor-x)\%s,(\lfloor\frac{k-1}{2}\rfloor-y)\%s) \tag{9a}$$

$$\mathbf{F}{:}\mathbf{F}_{(x,y)}=(\hat{\mathbf{X}}^{\boldsymbol{f}(x,y)}\circledast\boldsymbol{P}_{k-1}(\mathbf{X}))_{(x,y)} \tag{9b}$$

Eq. 9 costs s^2 operations to solve, which is still time-consuming when s is large. With the help of pixel-shuffle operation, Eq. 9 can be further reduced into a constant number of operations as follows:

$$\boldsymbol{g}{:}(x,y)\rightarrow x\times s+y \tag{10a}$$

$$\mathbf{F}=\boldsymbol{S}_s^{-1}\circ\boldsymbol{M}_{\boldsymbol{g}\circ\boldsymbol{f}\circ\boldsymbol{g}^{-1}}\{\boldsymbol{S}_s(\mathbf{X})\circledast\boldsymbol{S}_s\circ\boldsymbol{P}_{k\text{-}1}(\mathbf{X})\} \tag{10b}$$

where $\boldsymbol{g}$ is a mapping function, $\boldsymbol{S}_s$ and $\boldsymbol{S}_s^{-1}$ are the pixel shuffle/un-shuffle operations with scale factor s, and $\boldsymbol{M}$ reorders channels of a matrix according to mapping $\boldsymbol{g}\circ\boldsymbol{f}\circ\boldsymbol{g}^{-1}$. The elements in the x^{th} row of $\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}$ will reside in the x^{th} im2col block of $\boldsymbol{U}_k(\mathbf{F})$. Finally, $\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}$ can be computed as:

$$\mathfrak{X}^T\mathfrak{M}_s^T\mathfrak{M}_s\mathfrak{X}=\boldsymbol{R}\circ\boldsymbol{U}_k(\mathbf{F}) \tag{11}$$

where $\boldsymbol{R}(\mathbf{A})$ flips each row of matrix $\mathbf{A}$.

The memory-efficient solution described in Eqs. 8∼11 can reduce the memory consumption of solving blur kernels by a factor of $\frac{h\times w}{k\times k}$. For example, to super-resolve an image to 2K resolution (2048×1024) with k=11, it can save over 17000× memory overhead. The second step in the K-stream is to solve Eq. 5b to map pre-prior $\mathbf{K}'$ to post-prior $\mathbf{K}$, which is done by a DNN ($\text{Net}_{\mathbf{K}}$ in Fig. 1). The $\text{Net}_{\mathbf{K}}$ consists of 3 blocks, each of which is composed of two Convolutional (Conv) layers and one LeakyReLU layer. All Conv layers have 16 channels and all LeakyReLU layers have a negative slope of 0.01. A trailing ReLU layer is added to restrict the output estimation to be positive. The architecture graph is provided in the **Supplementary File**.

3.4 X-stream: super-resolved image estimation

The X-stream solves Eqs. 5c∼5d to estimate the super-resolved image. Given the blur kernel estimated by UDKE, it reduces into a non-blind SR problem, which can be easily done in two steps. The first step takes $\mathbf{Y}$ and the estimations of $\mathbf{K}$ and $\mathbf{X}$ as inputs, then solves Eq. 5c to update $\mathbf{X}'$. According to [35], the closed-form solution of $\mathbf{X}'$ in Eq. 5c can be derived by the Fast Fourier Transform (FFT):

$$\mathbf{X}'=\tfrac{1}{\alpha_{\mathbf{X}}}\mathscr{F}^{-1}\{\mathcal{Z}-\mathcal{K}\odot(\frac{(\bar{\mathcal{K}}\odot\mathcal{Z})}{\alpha_{\mathbf{X}}+(\bar{\mathcal{K}}\odot\mathcal{K})})\} \tag{12}$$

where $\mathscr{F}(\cdot)$ and $\mathscr{F}^{-1}(\cdot)$ denote the 2D FFT and its inverse, $\mathcal{K}=\mathscr{F}(\mathbf{K})$, $\mathcal{X}^\star=\mathscr{F}(\mathbf{X}^\star)$, $\mathcal{Y}=\mathscr{F}(\mathbf{Y})$, $\mathcal{X}=\mathscr{F}(\mathbf{X})$, $\mathcal{Z}=\mathcal{K}\circ\mathcal{Y}+\alpha_{\mathbf{X}}\mathcal{X}$, $\bar{\mathcal{K}}$ is the complex conjugate of $\mathcal{K}$, $\odot$ and $\dot{-}$ are the 2D Hadamard product and division, respectively.

Algorithm 1: Overall unfolding process of our UDKE based BISR framework

Input : LR image $\mathbf{Y}$, stages no. T, kernel size k, scale factor s, noise level σ
Output : Predicted HR image $\mathbf{X}^{pred}$, predicted blur kernel $\mathbf{K}^{pred}$
$\mathbf{X}_0 = \text{bic}_s(\mathbf{Y})$, $\mathbf{K}_0 = \frac{1}{k^2}$;
for $t = 1,...,T$ **do**
  $\{\alpha_{\mathbf{X}(t)}, \alpha_{\mathbf{K}(t)}, \beta_{\mathbf{X}(t)}, \beta_{\mathbf{K}(t)}\} = \text{HyperNet}_{(t)}(s, \sigma)$;
  $\mathbf{K}'_{(t)} = \text{Solve}_{\mathbf{K}}(\mathbf{Y}, \mathbf{K}_{(t\text{-}1)}, \mathbf{X}_{(t\text{-}1)}, \alpha_{\mathbf{K}}(t))$;
  $\mathbf{K}_{(t)} = \text{Net}_{\mathbf{K}}(\mathbf{K}'_{(t)}, \beta_{\mathbf{K}(t)})$;
  $\mathbf{X}'_{(t)} = \text{Solve}_{\mathbf{X}}(\mathbf{Y}, \mathbf{K}_{(t)}, \mathbf{X}_{(t\text{-}1)}, \alpha_{\mathbf{X}(t)})$;
  $\mathbf{X}_{(t)} = \text{Net}_{\mathbf{X}}(\mathbf{X}'_{(t)}, \beta_{\mathbf{X}(t)})$;
$\mathbf{X}^{pred} = \mathbf{X}_T$;
$\mathbf{K}^{pred} = \mathbf{K}_T$;

The second step solves Eq. 5d and map pre-prior $\mathbf{X}'$ to post-prior $\mathbf{X}$. From the Bayesian perspective [35], this step can be exactly interpreted as a denoising problem and can be solved by a DNN $\text{Net}_{\mathbf{X}}$. For efficiency consideration, we adopt the U-Net [24] as the DNN following previous non-blind SR method [35]. $\text{Net}_{\mathbf{X}}$ consists of 7 blocks. The first 3 blocks downsample the feature maps through strided convolution, and the last 3 blocks upsample the feature maps by transposed convolution. Each block consists of 4 residual units, while each of them consists of 2 Conv layers with ReLU and a skip connection. The channel numbers of Conv layers in the first 4 blocks are 64, 128, 256, 512, respectively. The architecture graph is provided in the **Supplementary File**.

3.5 Summary of the unfolding process

The K-stream and X-stream work alternatively to estimate the blur kernel $\mathbf{K}$ and HR image $\mathbf{X}$ for T stages. The determination of T will be discussed in Section 4.1. In the first stage, the input $\mathbf{X}_0$ is initialized by $\text{bic}_s(\mathbf{Y})$, where bic_s is the bicubic upsampling operation with scale factor s, while all the elements of $\mathbf{K}_0$ are initialized to $\frac{1}{k^2}$. For each stage, a tiny 2-layer fully connected network, called HypaNet, which takes σ and s as inputs, is introduced to predict the hyperparameters. The architecture graph of HypaNet and is provided in the **Supplementary File**. Algorithm 1 depicts the overall unfolding process.

4 Experiments

4.1 Implementation details

Training data and kernel pool. Following previous BISR works [35,20,28], we use the DIV2K [2] and Flickr2K [18] datasets to train our UDKE based framework. The ground truth images $\mathbf{Y}^{gt}$ are obtained by randomly cropping patches of size 128×128 from the original images, and the LR images $\mathbf{Y}$ are obtained by randomly selecting kernels from the DEPD-training kernel pool [40]

Table 1: 2× BISR results (PSNR/SSIM). Best results are in red.

Datasets	Set5			BSD100			Urban100		
σ	0	2.55	7.65	0	2.55	7.65	0	2.55	7.65
Bicubic	26.56/.8196	26.53/.8111	25.87/.7569	24.95/.7071	24.79/.6402	24.49/.6402	22.18/.7032	22.07/.6895	21.86/.6327
RCAN	27.35/.8578	27.04/.8153	24.73/.6113	25.70/.7620	25.29/.7170	23.69/.5345	23.14/.7650	22.84/.7192	21.75/.5442
ZSSR	27.03/.8803	26.53/.8143	25.41/.5840	25.31/.7498	25.10/.7122	23.29/.5449	22.66/.7443	22.34/.7121	21.89/.5579
DASR	27.48/.8788	27.00/.8188	24.95/.6310	25.82/.7624	25.27/.7143	23.81/.5442	23.37/.7803	22.80/.7181	21.86/.5611
KGAN	25.70/.8448	24.56/.6748	19.50/.3765	24.02/.7540	22.67/.5776	18.56/.3149	21.85/.7575	21.14/.6007	18.02/.3706
KFKP	17.28/.5279	14.55/.2766	12.25/.1829	21.16/.6608	19.79/.4934	16.27/.2890	19.56/.6725	18.61/.5361	16.02/.3260
DFKP	27.42/.8795	26.16/.7633	21.44/.4464	25.60/.7968	24.79/.6925	19.99/.3687	22.91/.7860	22.63/.7155	19.76/.4414
IKC	27.02/.8807	26.36/.8095	24.81/.6199	25.42/.8029	25.26/.7141	23.75/.5412	23.28/.8030	22.86/.7168	21.79/.5539
DAN	27.40/.8705	27.01/.8163	24.93/.6277	25.70/.7620	25.29/.7179	23.79/.5438	23.14/.7647	22.84/.7201	21.79/.5500
Ours	30.50/.8964	30.11/.8786	28.93/.8416	27.68/.8282	27.29/.8017	26.46/.7516	25.51/.8341	25.19/.8138	24.57/.7748
UBound	34.65/.9424	33.01/.9145	31.07/.9004	30.02/.8840	29.11/.8754	27.94/.8433	28.01/.8468	27.45/.8371	25.87/.8210

Table 2: 4× BISR results (PSNR/SSIM). Best results are in red.

Datasets	Set5			BSD100			Urban100		
σ	0	2.55	7.65	0	2.55	7.65	0	2.55	7.65
Bicubic	23.05/.6844	22.93/.6782	22.70/.6335	22.60/.5680	22.58/.5622	22.35/.5231	19.70/.5524	19.69/.5452	19.56/.5043
RCAN	24.01/.7196	23.87/.7132	21.78/.4960	23.11/.5969	22.50/.5740	20.33/.4853	19.88/.5849	19.48/.5742	19.00/.4513
ZSSR	24.33/.7369	22.88/.6563	22.50/.5063	23.03/.6070	22.79/.5933	20.21/.5024	20.95/.6048	20.19/.5824	19.37/.4769
DASR	24.33/.7243	24.16/.7008	22.31/.5321	23.23/.6490	23.12/.6171	21.14/.5192	21.43/.6360	21.32/.6031	19.43/.4902
KGAN	22.65/.6613	22.03/.6325	17.4/.3820	22.12/.6032	21.76/.5832	17.65/.3326	18.23/.6032	17.65/.4723	14.25/.3027
KFKP	15.32/.4859	14.21/.3332	12.75/.2199	18.23/.3765	17.65/.3321	15.64/.2818	17.64/.4083	17.53/.3125	15.13/.3022
DFKP	24.15/.7318	23.83/.6927	19.34/.4467	23.12/.6530	22.88/.6311	18.21/.3831	20.23/.6320	20.13/.5923	18.02/.3445
IKC	24.03/.7152	22.96/.6938	21.39/.5062	23.93/.6123	23.11/.6035	21.12/.4767	21.33/.6265	21.14/.6125	19.34/.4532
DAN	24.25/.7165	24.13/.6817	22.14/.5157	23.14/.6254	23.01/.5963	20.98/.4943	21.10/.6030	21.01/.5977	19.29/.4433
Ours	27.33/.8118	27.22/.8048	26.51/.7638	24.99/.6813	24.92/.6738	24.46/.6412	22.47/.6937	22.42/.6872	22.13/.6589
UBound	31.01/.8810	30.43/.8793	29.43/.8603	26.85/.7543	26.21/.7432	25.92/.7136	24.98/.7632	24.39/.7412	23.88/.6960

and applying the degradation model in Eq. 1. The DEPD kernel pool adopts dark channel priors [23] to analyze BISR kernels from real-world LR images captured by low-end camera phones. Specifically, the DEPD-training subset consists of 1,000 BISR kernels analyzed from photos captured by Blackberry Passport phone and Sony Xperia Z. The DEPD-evaluation subset consists of 300 BISR kernels analyzed from photos captured by iPhone 3GS.

Training details. The L_1 loss is used as the loss function for each stage of UDKE. Following [39], the weight on the loss of the last stage is set as 1, and the weights on all the other stages are set as $\frac{1}{T-1}$. The Adam optimizer [16] is used for updating the parameters of $\text{Net}_{\mathbf{K}}$ and $\text{Net}_{\mathbf{X}}$. The batch size is 32. We train the network for 10^5 iterations. The learning rate starts from 10^{-4} and decays by a factor of 0.5 for every 2^4 iterations. In order to speed up and stabilize the training process, we first train a 1-stage model and reload its weights into the T-stage model for fine-tuning. All the T stages share the same parameters. The size k for BISR kernel is set to 11 and the scale factor s is set to {2,3,4}, following previous BISR methods [35,20,28]. For each scale factor, we train a model for all noise levels, which are set to {0,2.55,7.65} as in [35].

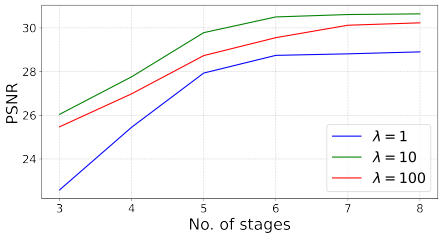


Table 3: Kernel estimation results (PSNR). Best results are in red.

Scale	Datasets	Set5			BSD100			Urban100		
	σ	0	2.55	7.65	0	2.55	7.65	0	2.55	7.65
×2	KGAN	41.0	40.9	39.2	40.4	40.2	39.7	40.3	40.2	39.9
	KFKP	37.8	38.0	37.5	38.7	38.9	38.5	38.6	38.7	38.5
	DFKP	39.4	39.6	39.4	39.0	39.3	38.9	38.6	38.8	38.5
	Ours	51.0	49.7	48.9	49.3	47.2	47.3	48.8	47.7	47.0
×4	KGAN	40.1	39.1	37.1	39.0	38.2	37.2	38.7	37.1	36.9
	KFKP	37.7	37.2	36.8	37.3	36.4	36.1	37.0	36.0	35.8
	DFKP	39.1	38.4	37.7	37.8	37.8	37.1	37.1	36.8	36.2
	Ours	45.1	45.5	42.2	43.8	43.6	43.5	44.4	44.7	44.0

Fig. 2: Ablation study on λ and T.

The selection of λ and T. There are mainly two parameters to set in our method, the trade-off parameter λ and the no. of stages T. We perform ablation studies to select them. The ablation study is done on the Set5 [6] dataset with s=2 and σ=0. The PSNR results w.r.t. λ and T are illustrated in Fig. 2. It can be seen that the PSNR index increases with the increase of T, and the highest PSNR is achieved when λ=10. However, the improvement becomes minor when T=6. Hence, we set λ=10 and T=6 in all the following experiments.

4.2 Comparison with state-of-the-arts

In this section, we compare the proposed UDKE based framework with state-of-the-art BISR methods. Three sets of experiments are conducted to comprehensively evaluate UDKE. First, we evaluate UDKE by using the DEPD-evaluation kernel pool, which is more complex than the widely used Gaussian kernels and more similar to the degradation in real-world images [40]. Second, we compare UDKE, which does not impose presumptions on blur kernels, with the methods that presume Gaussian blur kernels, by using just Gaussian kernels. Third, we evaluate UDKE on real-world images whose degradation kernels can be more complex than Gaussian kernels and those in the DEPD-evaluation kernel pool.

Experiments with DEPD-evaluation kernel pool. Following prior works [17,20], the widely used Set5, BSD100 [21] and Urban100 [15] datasets are used in the experiments. Testing LR images are obtained by applying randomly selected BISR kernels from the DEPD-evaluation pool to HR images according to the degradation process described in Eq. 1. We compare our UDKE based framework with bicubic interpolation, SOTA non-blind SR method RCAN [38], SOTA direct DNN based BISR methods, including ZSSR [27], DASR [28], KGAN [5], DFKP (DIP-FKP) [17] and KFKP (KernelGAN-FKP) [17], and SOTA deep unfolding BISR methods, including IKC [13] and DAN [20]. Note that some methods (*e.g.*, KernelNet [32]) are inapplicable for comparison. This is because that their source code is not disclosed and we cannot reproduce them; and they reported performance on non-standard testing sets so that we cannot compare with the reported results either.

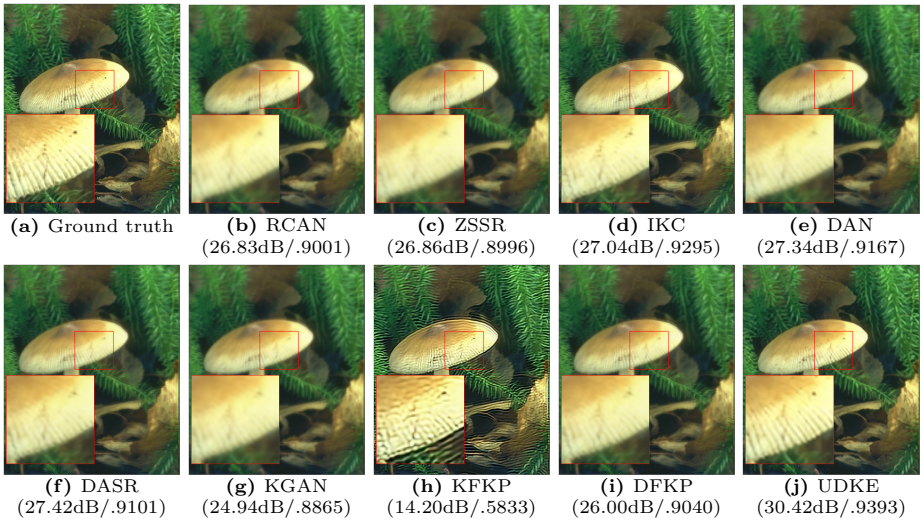
The experimental results of ×2 and ×4 BISR are shown in Tables 1∼2, respectively. The experimental results of ×3 BISR are provided in the **Sup-**

plementary File. We also provide an "upper bound" (UBound) of the BISR methods, which is obtained by feeding the ground truth kernel to the non-blind SISR method USRNet [35], yielding a non-blind "upper bound" as a reference for evaluating those blind BISR methods. Besides the PSNR/SSIM of the reconstructed image, for those BISR methods that estimate the blur kernels (*i.e.*, KGAN [5], DFKP [17], KFKP [17] and our UDKE), we also compute the PSNR of the estimated kernel (Kernel-PSNR), and list the results in Table 3.

It can be seen that UDKE based framework achieves the best PSNR, SSIM and Kernel-PSNR results under all experiment settings, significantly outperforming the competing methods. Both the direct DNN based methods (ZSSR, DASR, KGAN, KFKP, DFKP) and deep unfolding methods (IKC and DAN) do not surpass the non-blind method RCAN. This is mainly because these methods cannot encode effectively the information of the degradation process in their models, and hence they can not estimate the HR images with unseen degradation parameters during inference. In contrast, our proposed UDKE can estimate the degradation kernel very well (please see the Kernel-PSNR in Table 3) and consequently reproduce the original image with better quality. To further demonstrate the performance of UDKE, we also build several testing sets, each of them is built with a specific kernel from the DEPD-evaluation pool. The results are provided in the **Supplementary File** due to the limit of space.

Fig. 3 visualizes the BISR results on an image by the competing methods. It can be seen that the image reconstructed by UDKE has the best visual quality with sharp edges and rich textures. Other methods generate blurry results as they fail to well estimate and exploit the BISR kernel information. KFKP, which enforces strong Gaussian priors on kernel estimation and employs unstable adversarial training, results in distorted image textures due to kernel mismatching. Fig. 4 visualizes the predicted kernels by those BISR methods with explicit kernel estimation. It can be seen that only UDKE predicts the kernel with high fidelity, while others yield unreliable results because they generally impose the Gaussian-shape assumption on the kernels. More visualization results can be found in the **Supplementary File**.

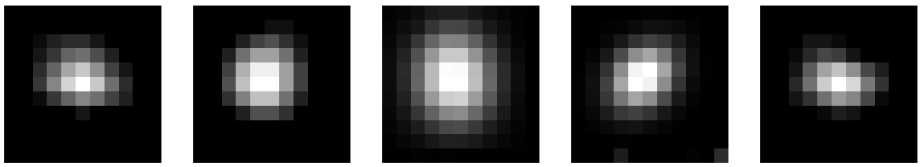
Experiments with Gaussian kernels. Although UDKE is not designed and trained for Gaussian kernel degradation, we also test its effectiveness on Gaussian kernels in comparison with those BISR methods designed and trained for Gaussian blur kernels, *i.e.*, DAN [20], KGAN [5], KFKP [17], and DFKP [17]. The Gaussian kernels are obtained by the process adopted in Gaussian BISR methods [5,17], which randomly samples the length and rotation angle to generate Gaussian kernels. The experiment is done on BSD100 set. Table 4 shows the BISR results on anisotropic Gaussian degradation. The isotropic Gaussian BISR results are provided in the **Supplementary File**. One can see that UDKE achieves competitive results with those Gaussian BISR methods but without hardcoding any Gaussian priors. With the increase of noise level, it can even surpass DFKP, showing better robustness to noise.



(a) Ground truth **(b)** RCAN (26.83dB/.9001) **(c)** ZSSR (26.86dB/.8996) **(d)** IKC (27.04dB/.9295) **(e)** DAN (27.34dB/.9167)

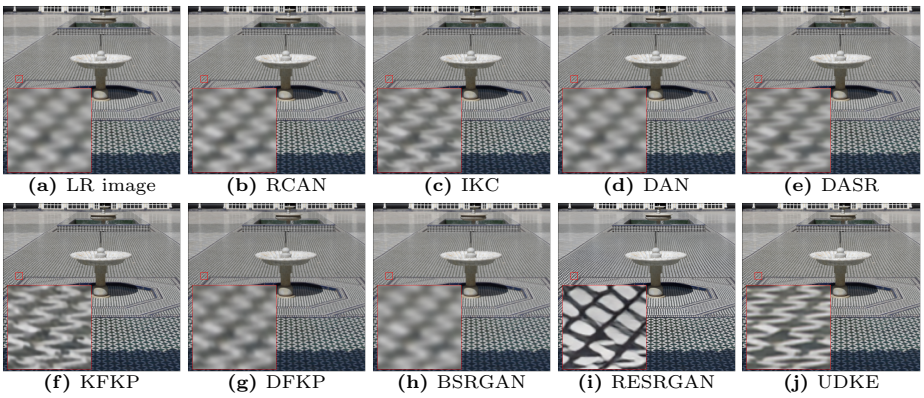
(f) DASR (27.42dB/.9101) **(g)** KGAN (24.94dB/.8865) **(h)** KFKP (14.20dB/.5833) **(i)** DFKP (26.00dB/.9040) **(j)** UDKE (30.42dB/.9393)

Fig. 3: Results on "208001" in BSD100 with s=2 and σ=0 (PSNR/SSIM).



(a) Ground truth **(b)** KGAN (43.7dB) **(c)** KFKP (39.6dB) **(d)** DFKP (43.3dB) **(e)** UDKE (47.6dB)

Fig. 4: Kernel estimation results on "208001" in BSD100 with s=2 and σ=0.



(a) LR image **(b)** RCAN **(c)** IKC **(d)** DAN **(e)** DASR

(f) KFKP **(g)** DFKP **(h)** BSRGAN **(i)** RESRGAN **(j)** UDKE

Fig. 5: Result on a real-world image (better viewed on screen).

Table 4: Anisotropic Gaussian BISR results (PSNR/SSIM/Kernel-PSNR).

s		σ=0	σ=2.55	σ=7.65
2	DAN	28.12/.8557/-	27.13/.8028/-	26.35/.7443/-
	KGAN	26.33/.7751/44.3	26.01/.7432/43.8	25.79/.7026/42.1
	KFKP	27.62/.8512/49.1	27.11/.8010/46.9	26.34/.7450/45.2
	DFKP	28.42/.8863/49.8	27.32/.8218/47.2	26.71/.7570/45.8
	Ours	27.56/.8112/48.5	27.10/.8003/46.9	26.73/.7700/46.1
4	DAN	24.98/.6813/-	24.42/.6348/-	23.01/.5575/-
	KGAN	23.97/.6211/43.2	23.70/.5987/42.1	23.01/.5575/41.8
	KFKP	24.87/.6612/44.2	24.49/.6353/43.0	24.01/.5801/42.7
	DFKP	25.47/.7233/46.4	24.62/.6891/44.0	24.10/.6013/44.0
	Ours	24.89/.6612/44.3	24.52/.6348/43.1	24.32/.6187/44.0

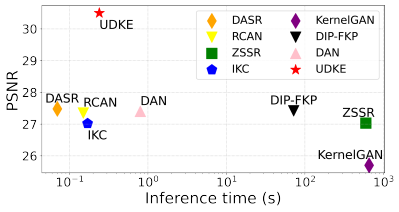


Fig. 6: Inference time v.s. PSNR.

Experiments on real-world images. We further test UDKE on real-world LR images whose degradation kernels can be more complex than Gaussian as well as the DEPD-evaluation pool. We add two state-of-the-art real-world SISR methods, *i.e.*, RESRGAN (Real-ESRGAN) [31] and BSRGAN [36], for more comprehensive comparison. The results are shown in Fig. 5. It can be seen that UDKE produces the best result, with not only superior perceptual quality but also more accurate fidelity (see the ground tiles). KFKP generates many noisy artifacts and distorted edges, while RESRGAN generates false patterns on the ground tiles. All the other competing methods output blurry results. This experiment demonstrates the robust kernel estimation capability of UDKE in real-world scenarios. More examples can be found in the **Supplementary File**.

Inference time. We further compare the inference time of UDKE and the competing methods. The experiments are conducted on Set5 (s=2 and σ=0) with a GTX 2080Ti GPU. The results are shown in Fig. 6. It can be seen that UDKE is slightly slower than DASR and has a similar speed to RCAN and IKC. It is over $\times$100 times faster than the leading online training based BISR method DFKP, which trains a deep model during the inference, and $\times$1000 times faster than ZSSR and KGAN. Meanwhile, UDKE achieves significantly higher PSNR (about 3dB) than all competing methods.

5 Conclusion

We proposed a novel unfolded deep kernel estimation method, namely UDKE, to explicitly solve the data term of the unfolding objective function for effective BISR. Equipped with the designed memory-efficient algorithm, UDKE is free of the im2col operation required in traditional Least Squares Method and hence saves over 17000$\times$ memory overhead, providing an efficient solution to explicitly solve the data term under the deep learning framework. UDKE addresses the challenging BISR problem by efficiently utilizing the information of degradation model during inference. Extensive experiments on both synthetic and real-world images validated that UDKE could faithfully predict non-Gaussian blur kernels, and reproduce high quality images with sharp structures and rich textures, surpassing existing BISR methods by a large margin. Meanwhile, UDKE has good efficiency, making it an attractive choice for BISR in practice.

References

1. Abdi, H., et al.: The method of least squares. Encyclopedia of Measurement and Statistics. CA, USA: Thousand Oaks (2007)
2. Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: Dataset and study. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (July 2017)
3. Barbu, A.: Training an active random field for real-time image denoising. IEEE Transactions on Image Processing **18**(11), 2451–2462 (2009)
4. Begin, I., Ferrie, F.: Blind super-resolution using a learning-based approach. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. vol. 2, pp. 85–89. IEEE (2004)
5. Bell-Kligler, S., Shocher, A., Irani, M.: Blind super-resolution kernel estimation using an internal-gan. arXiv preprint arXiv:1909.06581 (2019)
6. Bevilacqua, M., Roumy, A., Guillemot, C., Alberi-Morel, M.L.: Low-complexity single-image super-resolution based on nonnegative neighbor embedding (2012)
7. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine learning **3**(1), 1–122 (2011)
8. Bulat, A., Yang, J., Tzimiropoulos, G.: To learn image super-resolution, use a gan to learn how to do image degradation first. In: Proceedings of the European conference on computer vision (ECCV). pp. 185–200 (2018)
9. Capel, D., Zisserman, A.: Super-resolution enhancement of text image sequences. In: Proceedings 15th International Conference on Pattern Recognition. ICPR-2000. vol. 1, pp. 600–605. IEEE (2000)
10. Corduneanu, A., Platt, J.C.: Learning spatially-variable filters for super-resolution of text. In: IEEE International Conference on Image Processing 2005. vol. 1, pp. I–849. IEEE (2005)
11. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a deep convolutional network for image super-resolution. In: European conference on computer vision. pp. 184–199. Springer (2014)
12. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: 2009 IEEE 12th international conference on computer vision. pp. 349–356. IEEE (2009)
13. Gu, J., Lu, H., Zuo, W., Dong, C.: Blind super-resolution with iterative kernel correction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1604–1613 (2019)
14. He, Y., Yap, K.H., Chen, L., Chau, L.P.: A soft map framework for blind super-resolution image reconstruction. Image and Vision Computing **27**(4), 364–373 (2009)
15. Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5197–5206 (2015)
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
17. Liang, J., Zhang, K., Gu, S., Van Gool, L., Timofte, R.: Flow-based kernel prior with application to blind super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10601–10610 (2021)
18. Lim, B., Son, S., Kim, H., Nah, S., Mu Lee, K.: Enhanced deep residual networks for single image super-resolution. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 136–144 (2017)

19. Liu, Y., Sun, W.: Blind super-resolution for single remote sensing image via sparse representation and transformed self-similarity. In: Journal of Physics: Conference Series. vol. 1575, p. 012115. IOP Publishing (2020)
20. Luo, Z., Huang, Y., Li, S., Wang, L., Tan, T.: Unfolding the alternating optimization for blind super resolution. arXiv preprint arXiv:2010.02631 (2020)
21. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int'l Conf. Computer Vision. vol. 2, pp. 416–423 (July 2001)
22. Michaeli, T., Irani, M.: Nonparametric blind super-resolution. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 945–952 (2013)
23. Pan, J., Sun, D., Pfister, H., Yang, M.H.: Blind image deblurring using dark channel prior. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1628–1636 (2016)
24. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
25. Samuel, K.G., Tappen, M.F.: Learning optimized map estimates in continuously-valued mrf models. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 477–484. IEEE (2009)
26. Shao, W.Z., Elad, M.: Simple, accurate, and robust nonparametric blind super-resolution. In: International Conference on Image and Graphics. pp. 333–348. Springer (2015)
27. Shocher, A., Cohen, N., Irani, M.: "zero-shot" super-resolution using deep internal learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3118–3126 (2018)
28. Wang, L., Wang, Y., Dong, X., Xu, Q., Yang, J., An, W., Guo, Y.: Unsupervised degradation representation learning for blind super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10581–10590 (2021)
29. Wang, Q., Tang, X., Shum, H.: Patch based blind image super resolution. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1. vol. 1, pp. 709–716. IEEE (2005)
30. Wang, S., Zhang, L., Liang, Y., Pan, Q.: Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2216–2223. IEEE (2012)
31. Wang, X., Xie, L., Dong, C., Shan, Y.: Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In: International Conference on Computer Vision Workshops (ICCVW)
32. Yamac, M., Ataman, B., Nawaz, A.: Kernelnet: A blind super-resolution kernel estimation network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 453–462 (2021)
33. Yang, W., Zhang, X., Tian, Y., Wang, W., Xue, J.H., Liao, Q.: Deep learning for single image super-resolution: A brief review. IEEE Transactions on Multimedia **21**(12), 3106–3121 (2019)
34. Yuan, Y., Liu, S., Zhang, J., Zhang, Y., Dong, C., Lin, L.: Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 701–710 (2018)

35. Zhang, K., Gool, L.V., Timofte, R.: Deep unfolding network for image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3217–3226 (2020)
36. Zhang, K., Liang, J., Van Gool, L., Timofte, R.: Designing a practical degradation model for deep blind image super-resolution. In: IEEE International Conference on Computer Vision. pp. 4791–4800 (2021)
37. Zhang, K., Zuo, W., Zhang, L.: Learning a single convolutional super-resolution network for multiple degradations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3262–3271 (2018)
38. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: Proceedings of the European conference on computer vision (ECCV). pp. 286–301 (2018)
39. Zheng, H., Yong, H., Zhang, L.: Deep convolutional dictionary learning for image denoising. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 630–641 (2021)
40. Zhou, R., Susstrunk, S.: Kernel modeling super-resolution on real low-resolution images. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2433–2443 (2019)