A Comparative Study of PnP and Learning Approaches to Super-Resolution in a Real-World Setting

Samim Zahoor Taray[1], Sunil Prasad Jaiswal[1(✉)], Shivam Sharma[1,2], Noshaba Cheema[2,3], Klaus Illgner-Fehns[1], Philipp Slusallek[2,3], and Ivo Ihrke[1,4]

[1] K|Lens GmbH, Saarbrücken, Germany
sunil.jaiswal@k-lens.de
[2] Saarland Informatics Campus, Saarbrücken, Germany
[3] German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany
[4] Center for Sensor Systems (ZESS), University of Siegen, Siegen, Germany

Abstract. Single-Image Super-Resolution has seen dramatic improvements due to the application of deep learning and commonly achieved results show impressive performance. Nevertheless, the applicability to real-world images is limited and expectations are often disappointed when comparing to the performance achieved on synthetic data. For improving on this aspect, we investigate and compare two extensions of orthogonal popular techniques, namely plug-and-play optimization with learned priors, and a single end-to-end deep neural network trained on a larger variation of realistic synthesized training data, and compare their performance with special emphasis on model violations. We observe that the end-to-end network achieves a higher robustness and flexibility than the optimization based technique. The key to this is a wider variability and higher realism in the training data than is commonly employed in training these networks.

Keywords: Deconvolution · Generalizability · Degradations · Super-resolution

1 Introduction

Single-image super-resolution (SISR) techniques, in particular, learning based or learning supported techniques, have evolved to a point where impressive image improvements can be achieved on a wide variety of scenes [8,9,22,32,41]. However, their applicability in real-world scenarios is reduced by an often inadequate modeling of the image formation in real imaging systems [25]. In particular, spatially varying blur is often ignored and the noise characteristics of real sensors, modified by the non-linear operations of image signal processors, are commonly inadequately treated as Gaussian processes. This limitation can be traced back to synthesized data being used for training. The general approach is to use images from existing data sets and to synthesize low resolution images by using bicubic downsampling.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-030-92659-5_20.

C. Bauckhage et al. (Eds.): DAGM GCPR 2021, LNCS 13024, pp. 313–327, 2021.
https://doi.org/10.1007/978-3-030-92659-5_20

This, in conjunction with the required generalizability to different camera and lens combinations, poses a challenge for the real-world application of SISR techniques. Our goal is to improve on this situation. For this purpose, we follow two orthogonal approaches: 1) we generalize a method based on the Plug-and-Play (PnP) framework [35] to handle spatially varying blur kernels. 2) we explore the potential of Deep Neural Networks (DNNs) [8] to model mappings of input images that are subject to a variety of possible degradations, including different blur kernels of the expected spatial variation due to the lens system, as well as different noise levels and compositions, i.e. different ratios of Gaussian and Poisson noise as well as correlations introduced by image processing. We compare the performance of these two approaches and assess their robustness against model violations.

Common knowledge holds that PnP techniques are more resilient against the outlined problems, whereas DNNs quickly deteriorate in performance when run on data outside the characteristics of the training set. The latter is often assumed to include training data for a single image formation model and a modest variability in the noise settings. Our investigations show that the training data set can include a sufficient variety of image formation settings such that the performance of the trained DNN approaches an effectively blind deconvolution setting.

In summary, our contributions are as follows:

- we generalize PnP optimization techniques for SISR to handle spatially varying blur kernels,
- we propose a data synthesis approach for generating realistic input data for DNN training that incorporates the major sources of variation in real-world data,
- we train a state-of-the-art DNN [12,36] on a suitably synthesized training data set that includes expected real-world image formation variations (varying blur and realistic noise distributions),
- experimentally compare the resulting techniques on different scenarios with special emphasis on a violation of the image formation model.

We find that the end-to-end learning technique yields superior results and robustness as compared to the extended plug-and-play technique, as well as compared against the state-of-the-art.

2 Related Work

The goal of SISR is to generate a high-resolution (HR) image from a single low-resolution (LR) image. Most methods are intended for photographic content and aim to hallucinate details and textures that fit nicely with the input LR image while producing a realistic looking higher resolution image.

Example-based methods can be divided into two categories namely Internal and External Example-based methods.

Internal Example-based Methods [13,17] rely on the assumption that a natural image has repetitive content and exploit this recurrence by replacing each patch within the LR image with a higher resolution version. Shocher et al. [33] propose Zero Shot Super-Resolution wherein they train a small CNN using patches taken only from the test image.

External Example-based methods rely on an external database of paired LR/HR images. Freeman et al. [11] first generate a dictionary of LR and HR patches that is then used similarly to the previously discussed techniques in applications. Chang et al. [6] extended this method using k nearest dictionary neighbors whereas Zeyde et al. [38] learn sparse representations of LR and HR patches. Dictionary-based methods were superseded by Convolutional Neural Network (CNN) based methods which have been very successful and continue to dominate standard SISR benchmarks.

CNN-based methods were introduced by Dong et al. [8] who proposed the shallow convolutional network SRCNN. Kim et al. [20] show significant performance gains by using deeper networks. Shi et al. [32] propose the Efficient Sub-Pixel Convolution Layer (ESPCN) to include an up-scaling step into the network that had previously been implemented as a pre-process. Typical training losses are the L_1 and L_2 losses. These loss functions tend to create unattractive structures in the super-resolved (SR) images [19,22]. Thus, several recent works [19,22,36] have aimed to devise loss functions which correlate better with human perception. Ledig et al. [22] propose SRGAN which uses a combination of three loss functions to achieve photo-realistic super-resolution. Wang et al. [36] introduce improvements by comprehensively studying the factors affecting the performance of SRGAN, creating the ESRGAN technique. It should be noted that SRGAN and ESRGAN tend to produce images with textures even in flat image regions, creating perceptually unpleasing artifacts. To overcome this problem, Fritsche et al. [12] propose frequency separation, i.e. the GAN cannot synthesize in low-frequency image regions. We base our end-to-end technique, Sect. 5 on a combination of ESRGAN trained with a perceptual loss and frequency separation.

Recently, some algorithms have been proposed to overcome the challenges of real-world super-resolution mentioned in the introduction [18,21,24,27,31,42]. Zhou and Süsstrunk [42] construct a synthetic data set by simulating degradation on LR and HR pairs and train a CNN to perform SR, whereas Lugmayr et al. [24] propose unsupervised learning for data set generation while employing a supervised network for SR. Some methods [21,27] exploit blind kernel estimation to generalize better to real images while maintaining a non-blind general approach, whereas Ji et al. [18] propose to use two differently trained networks with a final output fusion stage to solve the realistic image super-resolution problem.

Plug-and-Play (PnP) Approaches are, in contrast, based on an optimization formulation of the SISR problem [7,15,26,34,35,43]. Typically, convex optimization has been employed for this purpose in image restoration [7,35,43] and, due to its flexibility, the idea has been extended to different applications [15,26,34], including super-resolution [4,5]. Most of these methods are limited to Gaussian noise assumptions. Zhang et al. [39] proposed to include a learned CNN prior in the iterative solution using the MAP framework. We build on their technique in Sect. 4.

3 Overview

The important components of an imaging system are its optics, its sensor and its processing unit, respective the algorithm running on this unit. These components work

together to give images the characteristics that we associate with natural real-world images, i.e. non-uniform lens blur and non-Gaussian noise afflicts real-world images. Optimization-based algorithms must properly model this image formation and we extend the PnP technique DPSR [39] to handle spatially varying blur in Sect. 4. Learning techniques rely on large amounts of training data that are most suitably generated synthetically. We describe the generation of suitable data with the outlined real-world characteristics and give details of the training process of the state-of-the-art network ESRGAN-FS [12,36] in Sect. 5. In Sect. 6, we compare the performance of the two approaches, first in synthetic experiments and later on actual real-world images.

4 Plug and Play Framework

Let the observed LR image be denoted by $\mathbf{y} \in R^{MN}$. It is related to the desired HR image $\mathbf{x} \in R^{s^2MN}$ via blurring, sampling and the addition of noise. Here M and N are the height and width of the LR image and $s \geq 1$ is the scaling factor. The effect of spatially varying blur is modeled by a linear operator H that operates on $\mathbf{x}$, while the sampling of the blurred image is modeled by the linear operator D. In this section, the sampled image is degraded by additive noise denoted by vector $\mathbf{n} \in R^{MN}$. In our implementations, the operators are realized as function evaluations rather than being discretized as sparse matrices.

According to Zhang et al. [39], it is advantageous, mathematically, to switch the procedures of blurring and sampling, because it enables a simple algorithmic decomposition that allows neural networks to be used as priors for super resolution.

$$\mathbf{y} = HD\mathbf{x} + \mathbf{n}. \tag{1}$$

The image formation model of Eq. 1, while physically incorrect, is an approximation that enables network-encoded priors to be easily incorporated into proximal optimization algorithms [3].

The resulting optimization problem is

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2\sigma^2}\|\mathbf{y} - HD\mathbf{x}\|^2 + \lambda\Phi(\mathbf{x}), \tag{2}$$

where the first term is the data-fidelity term and $\Phi(\mathbf{x})$ is the regularization term encoding the prior information, that is network-encoded in our setting. Solving the optimization problem via proximal algorithms consists of a variable splitting, that, in the case of the ADMM method [28] applied to Eq. 2 leads to the following iterative scheme:

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \lambda\Phi(\mathbf{x}) + \frac{\rho}{2}\|D\mathbf{x} - \mathbf{z}^k + \mathbf{u}^k\|_2^2, \tag{3}$$

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} \frac{1}{2\sigma^2}\|\mathbf{y} - H\mathbf{z}\|_2^2 + \frac{\rho}{2}\|D\mathbf{x}^{k+1} - \mathbf{z} + \mathbf{u}^k\|_2^2, \tag{4}$$

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \left(D\mathbf{x}^{k+1} - \mathbf{z}^{k+1}\right). \tag{5}$$

Here, $D\mathbf{x} = \mathbf{z}$ is the variable splitting, $\mathbf{u}$ is the dual variable, σ is the noise level under a Gaussian white noise assumption, λ is the regularization parameter and ρ is the penalty

parameter of the associated augmented Lagrangian L_ρ [3]. For a detailed derivation, please see the supplemental material. Please observe that the linear operators H and D now appear in different sub-expressions and not, as before, in combination. This enables the plug-and-play nature of the algorithm.

The first update step, Eq. 3 is a super-resolution scheme with a network-based prior encoded in $\Phi(\mathbf{x})$ as can be seen by ignoring the dual variable $\mathbf{u}$ that tends to zero as the splitting equality $D\mathbf{x} = \mathbf{z}$ is approached by the iteration. We follow the procedure of Zhang et al. [39] to obtain $\mathbf{x}^{k+1}$ via a pre-trained CNN. We use the same architecture as [39], i.e. SRResnet+, and train it for variable levels of additive white Gaussian noise in the range $[0, 50]$ for joint denoising and super-resolution. We denote the network by $\mathcal{SR}_D$ to note that the network is trained for a fixed downsampling operator D. To adapt a single network for various noise levels, a noise level map of the same size as the input image is created and concatenated with the input image. The image and the noise level map concatenated together become the input to the network which outputs a super-resolved and denoised image $\mathbf{x}^{k+1}$, i.e. Eq. 3 is replaced by

$$\mathbf{x}^{k+1} = \mathcal{SR}_D\left(\mathbf{z}^k, \rho\right). \tag{6}$$

The second update step, Eq. 4 can be interpreted as a deblurring step at the size of the LR image, which results in an intermediate deblurred LR image $\mathbf{z}$. Previous work considers the blur to be spatially invariant [39]. However, we consider the more general problem where the blur is spatially varying. We discuss this minimization problem in detail in Sect. 4.1 because it contains our extension of the algorithm [39] to spatially varying blur kernels.

The third update step given in Eq. 5 is simply the update of the dual variable required in ADMM.

We set $\lambda = 0.3$ and vary ρ to affect regularization for all our experiments. In practice, we increase ρ on an exponential scale from $\rho = 1/2500$ to $\rho = 2/\sigma$, where σ is the noise level of the image (assumed to be in the range $[0, 255]$).

4.1 Spatially Varying De-blurring

In this section we describe our approach to tackle the minimization problem of Eq. 4 which can be re-written in the following way:

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} \underbrace{\|\mathbf{y} - H\mathbf{z}\|_2^2 + \sigma^2\rho\|\tilde{\mathbf{x}}^{k+1} - \mathbf{z}\|_2^2}_{G(\mathbf{z})}. \tag{7}$$

For ease of notation, we have used $\tilde{\mathbf{x}}^{k+1} = D\mathbf{x}^{k+1} + \mathbf{u}^k$. Recall that $\mathbf{y}$ is the input LR image that we wish to super-resolve and σ is the noise level of each pixel in $\mathbf{y}$. As mentioned above, H now represents the spatially varying blur operator. The objective $G(\mathbf{z})$ is smooth and convex. To minimize the objective, we rely on the Gradient Descent with Momentum [30] method with constant step size and momentum parameters given by

$$\mathbf{w}_{t+1} = \mathbf{z}_t + \beta(\mathbf{z}_t - \mathbf{z}_{t-1}) \tag{8}$$

$$\mathbf{z}_{t+1} = \mathbf{w}_{t+1} - \alpha\nabla_{\mathbf{z}} G(\mathbf{w}_{t+1}), \tag{9}$$

Algorithm 1. $\mathcal{AG}$ solver

Input Operators H and H^T. Constant quantities $\mathbf{y}$, $\tilde{\mathbf{x}}^{k+1}$, ρ and σ. Parameters α and β and total iterations $N_{\mathcal{AG}}$

1: Initialize $\mathbf{z}_{-1} = \mathbf{y}$ and $\mathbf{z}_0 = \mathbf{0}$
2: **while** $t < N_{\mathcal{AG}}$ **do**
3: $\quad \mathbf{w}_{t+1} = \mathbf{z}_t + \beta(\mathbf{z}_t - \mathbf{z}_{t-1})$
4: $\quad \mathbf{z}_{t+1} = \mathbf{w}_{t+1} - 2\alpha\left(H^T H \mathbf{w}_{t+1} + \rho\sigma^2 \mathbf{w}_{t+1} - H^T y - \rho\sigma^2 \tilde{\mathbf{x}}^{k+1}\right)$

Output $\mathbf{z}_{N_{\mathcal{AG}}}$

where $\nabla_{\mathbf{z}}$ denotes the gradient operator with respect to $\mathbf{z}$ and $\mathbf{w}$ is an intermediate vector introduced to shorten the notation. We set $\alpha = 0.2$ and $\beta = 0.1$ for all our experiments. The details of the solver are given in Algorithm 1.

Note that the solver only relies on matrix vector multiplications of the form $H\mathbf{z}$ and $H^T\mathbf{z}$. We implement these efficiently using the *Filter Flow Framework* of Hirsch et al. [16]. Explicit formulas for evaluating H and H^T are given in the supplement. For convenience, let us denote the solver by $\mathcal{AG}$ which allows us to write Eq. 4 as:

$$\mathbf{z}^{k+1} = \mathcal{AG}(H, H^T, \tilde{\mathbf{x}}^{k+1}, \mathbf{y}, \sigma, \rho). \tag{10}$$

The solver $\mathcal{AG}$ takes as inputs the operators H and H^T along with the vectors $\mathbf{y}$, $\tilde{\mathbf{x}}^k$ and the constants ρ and σ, and outputs the deblurred image $\mathbf{z}^{k+1}$. Finally, the overall scheme to solve the original SISR problem of Eq. 2 is presented in Algorithm 2.

Algorithm 2. PnP Optimization with DNN prior for SISR

Input LR image $\mathbf{y}$, noise level σ, operators H and H^T total iterations T

1: Initialize $\mathbf{z}^0 = \mathbf{0}$ and $\mathbf{u}^0 = \mathbf{0}$
2: **while** $k < N_{\mathcal{SR}}$ **do**
3: $\quad \mathbf{x}^{k+1} = \mathcal{SR}_D\left(\mathbf{z}^k, \rho\right)$ [Joint SR and denoising]
4: $\quad \mathbf{z}^{k+1} = \mathcal{AG}(H, H^T, \mathbf{x}^{k+1}, \mathbf{y}, \sigma, \rho)$[Deblurring Step]
5: $\quad \mathbf{u}^{k+1} = \mathbf{u}^k + \left(D\mathbf{x}^{k+1} - \mathbf{z}^{k+1}\right)$ [Dual var. update]

Output $\mathbf{x}^{N_{\mathcal{SR}}}$

5 End-to-End Learning with Input Variants

In the above section, we generalize PnP optimization techniques and present a non-blind learning-supported optimization algorithm to handle spatially varying blur kernels. Our goal in the current section is to train the state-of-the-art SISR end-to-end network ESRGAN [36] to include variations in expected input data both in terms of realistic noise models and levels as well as in expected blur variations. We use the Residual-in-Residual Dense Block Network (RRDBNet) architecture of [36] with 23 RRDBs, and train the network using either L_1 loss or the perceptually motivated L_{per} loss [22]. We refer to the respective training loss in discussing our experimental results

in Sect. 6. We also employ the frequency separation technique of Fritsche et al. [12] to suppress texture generation in smooth image areas. The images are cropped to a size of 128×128 during training and data augmentation consisting of random horizontal and vertical flips is also applied for better generalization. We implement the network in PyTorch and use the built in Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$ and an initial learning rate of 2×10^{-4} for optimization. We set the batch size to 24 and train the network for 500 epochs. Training the network on one Nvidia Quadro 6000 RTX takes around 10 h.

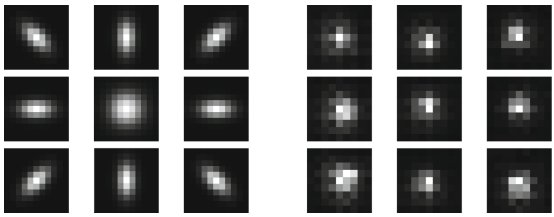

Fig. 1. Simulated spatially varying blur kernels used for the synthetic experiments (left). Measured spatially varying blur kernels of the target optical system (right). The spatial layout indicates the image regions affected by the corresponding kernels. The kernels are smoothly blended across the simulated images.

5.1 Training Data Synthesis

In order to generate a large amount of training data, we rely on LR image synthesis. To achieve realism, we aim to match the blurring and noise characteristics found in real world images. We use a data set containing high quality images [1,23] as a basis. Since the images in these data sets can contain residual noise, we clean these images prior to the synthesis operation by Gaussian filtering and downsampling, resulting in the clean HR image. Let this image be denoted by $\mathbf{x}$. The HR image is blurred to obtain a blurred image denoted by $\mathbf{x}_{\text{blurred}}$ by convolving $\mathbf{x}$ with a filter $\mathbf{h}$. This gives the clean but blurred image $\mathbf{x}_{\text{blurred}}$ which is then downsampled by the desired scale factor s giving the downsampled image denoted by $\mathbf{y}_s$ where s denotes, as before, the factor by which we ultimately wish to super-resolve the images. The filter $\mathbf{h}$ is obtained by randomly selecting one filter from a filter bank. The filter bank consists of filters obtained as explained below.

- We synthesize a set of filters consisting of 9×9, 7×7 and 5×5 Gaussian filters of different major axes and orientations. A subset of these filters are shown in Fig. 1 (left).
- We also measure PSFs by imaging an illuminated pinhole of $30\,\mu\text{m}$ diameter in a darkened photographic studio and add these to the filter bank. These are shown in Fig. 1 (right).

– In order to increase the size and diversity of the filter bank, we extract blur kernels from real-world images taken with the target optical system[1] using KernelGAN [21] and add them to the filter bank. We extracted around 3000 kernels using this approach and a few of them are shown in Fig. 2.

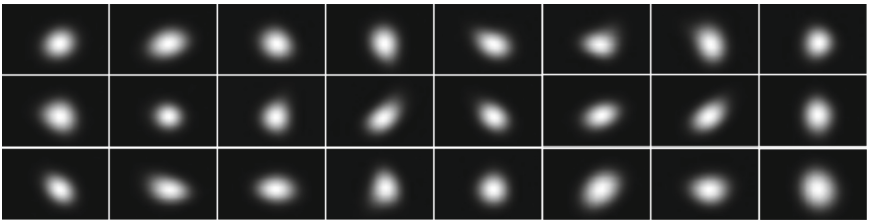

Fig. 2. Blur kernels extracted using KernelGAN [21] on 24 real-world images.

The next step consists of injecting realistic noise into $\mathbf{y}_\mathrm{s}$ to finally arrive at the desired low resolution image. To achieve this we first convert $\mathbf{y}_\mathrm{s}$ from sRGB to RAW space, yielding $\mathbf{r}_s$. This is in order to apply noise in the linear response regime of the sensor rather than in the non-linearly distorted processed image domain. Next, we generate noise according to the Poisson-Gaussian noise model of Foi et al. [10] by sampling from a heteroskedastic Gaussian distribution [14], i.e.

$$\mathbf{n}(\mathbf{r}_s) \sim \mathcal{N}(0, \sigma^2(\mathbf{r}_s)), \tag{11}$$

where $\sigma^2(\mathbf{r}_s)$ denotes the variance of the Gaussian distribution which is a function of the RAW image and is given by $\sigma^2(\mathbf{r}_s) = a\mathbf{r}_s + b$. The parameter a determines the amount of the Poisson and b the amount of the Gaussian noise component. The values of a and b are determined by the amount of noise present in the target images that we aim at super-resolving. As shown by Guo et al. [14], deep convolutional neural networks can effectively denoise images containing noise with different noise levels if the network is trained for these different noise levels. Therefore, we generate a variety of images containing different levels of Poisson and Gaussian noise by sampling a and b from a sensible range of values [14]. The last step consists of converting the noisy RAW image back into sRGB space which finally results in the noisy low resolution sRGB image $\mathbf{y}$ corresponding to the high resolution image $\mathbf{x}$. Figures 4 and 5 show examples of ground truth (GT) HR images and the corresponding synthesized LR images (LR input) with realistic blur and noise characteristics with this method. For the conversion steps, i.e. to convert the downsampled sRGB image to RAW space and then back to the sRGB space, we use CycleISP [37]. We note that our method would work with other methods for sRGB to RAW conversion.

[1] This way the SR scheme is tailored towards a particular type of optical system. In order to gain generality w.r.t. manufacturing differences for the optical system, the kernels have been extracted from pictures taken with different lenses of the same model.

6 Experimental Results

In the following, we evaluate and compare the two methods of Sect. 4 and Sect. 5 in terms of their performance with respect to real-world imperfections, i.e. realistic noise components and spatially varying blur. We first perform a set of quantitative synthetic experiments on a subset of 10 images taken from the DIV2K [1] validation set in Sect. 6.1 and finally show the qualitative performance on real-world data where no ground truth is available in Sect. 6.2.

6.1 Synthetic Experiments

Effect of Noise: To study and understand the effect of noise on the quality of SR we perform an experiment in which we synthesize test data only with different noise settings of a and b in Eq. 11 including the ISP-simulation (realistic noise, but no image formation model) and study the performance of the different algorithms. The results are shown in Fig. 3. Our test candidates are two CNN variants according to Sect. 5, trained with L_1 (red) and with L_{per} (green) loss and the PnP method of Sect. 4 (blue). We train the networks with training data up to the noise parameters $a = 0.03, b = 0.025$. The maximum training noise is marked by a dashed black vertical bar in Fig. 3. The figure shows three different performance metrics averaged over 10 test images. PSNR (dotted) and SSIM (dashed) curves refer to the right axis labels of the plots, whereas the perceptual LPIPS (solid) metric [40] uses the left axis labels. PSNR is plotted as relative PSNR which is a fraction of the maximum of 32.0 throughout the paper. Figure 4 shows visual examples on one of the test images. The red vertical dashed line in Fig. 3 indicates the noise settings used in Fig. 4.

As expected, with rising noise levels, the performance decreases for all methods and measures, both for Gaussian and Poissonian noise components. The L_1 trained models dominate the PSNR and SSIM scores in both cases while PnP is outperformed for PSNR. For SSIM, the L_{per} trained CNN and PnP compete in performance: for lower noise levels, the L_{per} trained model has advantages, whereas for larger levels, PnP performs better. In terms of Poisson noise, this trend is large, whereas for the Gaussian part (parameter b), the two methods behave very similarly for larger amounts of noise. In terms of the perceptual LPIPS score, a lower graph is better. Here, the L_{per} trained CNN has the best performance for low to medium noise levels in both cases, however, a cross-over is observed with the L_1 trained model. For the Poisson component (parameter a), the cross-over occurs, as expected, close to the maximum trained noise level. For the Gaussian component it occurs much earlier due to unknown reasons. The PnP method is not competitive in this metric which can be traced to its generation of flat structures in the images, see also Fig. 4.

Effect of Spatially Varying Blur: In this experiment, we test the importance of modeling the spatially varying blur in an imaging system. For this, we synthesize test images with known spatially varying blur kernels according to the methodology described in Sect. 4.1. An example of spatially varying blur kernels is provided in Fig. 1. For a fair comparison, the PnP method should run on Gaussian noise, whereas the CNNs should run on realistic noise. We solve this problem by fixing a realistic noise level

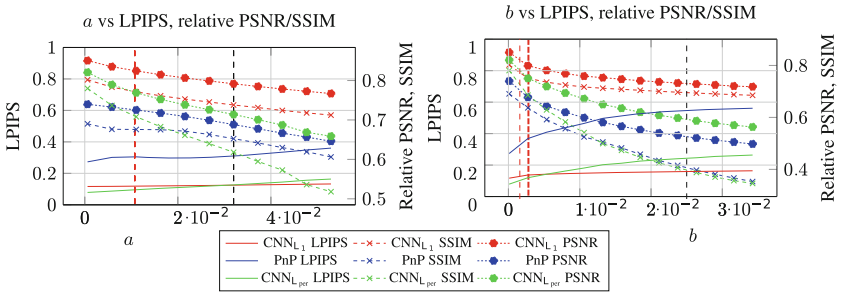

Fig. 3. Model performance against noise parameter variations of a Poisson-Gaussian model with ISP simulation. For PSNR and SSIM, larger values are better, for LPIPS, lower values indicate better performance. Curves are only to be compared for like scores (same decorator). The details of the plots are explained in the main text. (Color figure online)

Fig. 4. Visual examples of the noise experiment on one of the test images degraded with the varying blur kernels shown in Fig. 1 (left) and realistic noise with noise levels given by $a = 0.011, b = 0.0001$.

Table 1. Numerical results for spatially varying blur. For the experimental setting, see the main text.

Method	Varying blur, Gaussian noise		
	PSNR↑	SSIM↑	LPIPS↓
PnP	23.20739	0.67589	0.3049
CNN L_1 (invariant)	26.3152	0.7699	0.281
CNN L_1	**26.5405**	**0.781**	0.266
CNN L_{per}	24.4716	0.7068	**0.0955**

of ($a = 0.01, b = 0.001$), synthesizing LR images as input for the CNN technique and estimating the standard deviation. We then synthesize another set of LR images as input for the PnP technique that uses this estimated Gaussian noise level. The numerical performance is shown in Table 1.

We again report averages over 10 test images synthesized with the fixed set of kernels in Fig. 1 (left). In addition to the previous contestants, we add an L_1 trained CNN that was only learned on a spatially invariant kernel (center kernel in Fig. 1 left) to demonstrate the importance of proper modeling. The PnP method is used with known

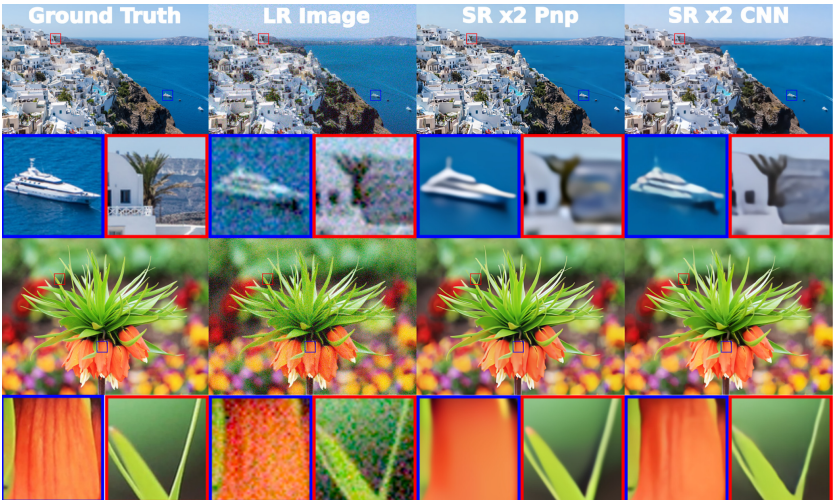

Fig. 5. Qualitative comparison of super-resolved images produced by the PnP approach with the L_1-trained CNN (spatial variation). The results from the CNN have sharper details and more realistic textures.

blur kernels (non-blind setting). Visual results for this experiment are shown in Fig. 5 for the L_1 trained CNN and the PnP method. PnP favors smooth regions with sharp edges, similar but not as noticable as in a standard Total Variation regularized setting. The variation-trained CNN recovers meaningful structures, but avoids hallucination of detail in smooth images regions thanks to frequency separation.

Model Violations. In this numerical experiment, we analyze the effect of mismatches in model assumptions. The PnP method of Sect. 4 has the strongest assumptions since it is a non-blind technique and is based on a Gaussian error assumption as well as resting on a non-physical image formation model, Eq. 1, whereas the network family of Sect. 5 is adjusted by the training set and has blind estimation capabilities due to training on a variety of kernels and noise settings.

We synthesize test data for two image formation models: space-variant blurring followed by downsampling (the physically correct model), and the non-physical inverse order of operations, Eq. 1. In addition, we compare performance in the assumed Gaussian noise setting vs. the realistic noise setting of Sect. 5.

6.2 Real-World Images

We perform a comparison of our methods (CNN L_{per}, blind = "Ours ESRGAN-FS", PnP with manually tuned kernels and noise settings = "Ours PnP") with recent state-of-the-art methods for real-world SR on a variety of real-world images taken using different cameras, lenses and under different settings. The methods we use for comparison are ESRGAN [36] which is trained on the default DF2K data set, i.e. with clean LR images. We also use the combination of CycleISP [37] + ESRGAN. CycleISP is the

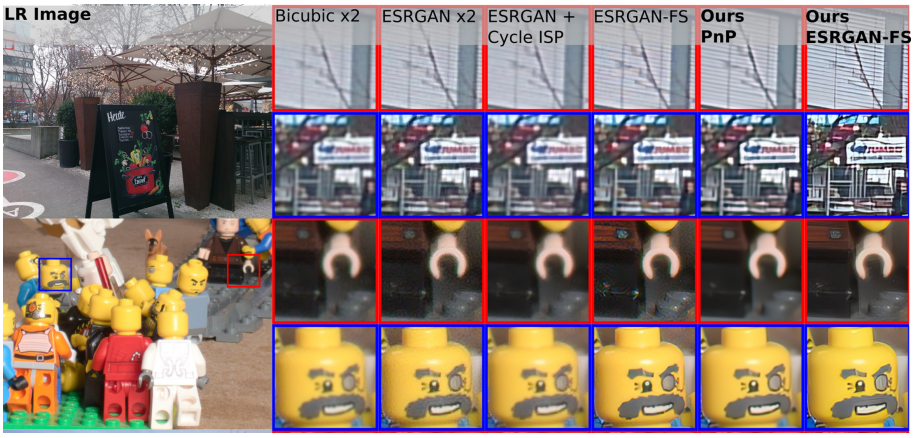

Fig. 6. Comparison with state of the art for a variation of scenes. Zoom into the digital version for details. The test images show a variation of artificial/man made and natural structures as well as smooth and highly textured imaged regions. We also include structures close to the sampling limit.

current state-of-the-art method for real-world denoising [29] and we use it to clean up the LR images first. Then, we use ESRGAN to perform SR. We also compare against the method of Fritsche et al. [12] (ESRGAN-FS), which is the current state of the art in real-world SR [25]. Since ground truth HR images are not available for these real-world images, we have to rely on a qualitative comparison. The results of our experiment are shown in Fig. 6.

ESRGAN can produce sharp images, however, it also enhances the noise and cannot effectively deal with blur in real world images. The CycleISP + ESRGAN method, on the other hand, is able to remove some noise but the resulting images look smooth and lack detailed textures. The results from the method of Fritsche et al. are sharp and contain a good amount of detail. However, it also enhances noise in parts of the images significantly. For "Ours PnP", we manually tuned the blur kernels and noise settings for $\approx$1 h per image and show the best result. As in the synthetic test, the contrast is good, but fine detail is smoothed out. "Ours ESRGAN-FS" produces the best results overall. The noise is effectively suppressed in all parts of the super-resolved image. The generated images are sharp and contain a good amount of texture and details. The images from this method are perceptually the most pleasing out of the methods compared.

7 Discussion and Conclusions

Our experiments show that a single end-to-end CNN for blind SISR trained on a suitably varied data set can match the performance of a non-blind learning-supported optimization algorithm that has full knowledge of both the PSFs and the noise level. In terms of PSNR and SSIM, the performance is slightly worse, but in the perceptually more relevant LPIPS metric [40], the blind end-to-end network out-performs the optimiza-

Table 2. Comparison of results on test images first blurred and then downsampled by a scale factor of 2. CNN L_1 achieves better results for realistic noise, whereas the PnP approach achieves better results for additive white Gaussian noise in terms of PSNR and SSIM. However, CNN L_1 achieves better results in terms of LPIPS score.

Method	Realistic noise			Gaussian noise		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
	Blurring followed by downsampling					
CNN L_1	**26.6646**	**0.7841**	**0.1212**	23.6885	0.6673	**0.1681**
PnP	23.0467	0.6845	0.3115	**25.9489**	**0.7617**	0.1820
	Downsampling followed by blurring					
CNN L_1	**25.1422**	**0.7342**	**0.1503**	24.4181	0.7029	**0.1438**
PnP	23.3231	0.6706	0.3080	**25.8148**	**0.7408**	0.1743

tion algorithm. This observation holds even when the image formation model of the optimization algorithm is matched exactly in the synthetic experiment.

The optimization scheme of Sect. 4 can, for our purposes, be interpreted as an upper bound on the performance that a blind learning-supported optimization scheme would be able to achieve. The conclusion from our experiments is then that the end-to-end network is more flexible and results in perceptually more meaningful super-resolved images. This experimental observation suggests that end-to-end networks are competitive for *blind* denoising, deblurring, and super-resolution tasks (Table 2).

SISR CNNs are mappings from an input manifold of blurred and noisy patches to an output manifold of clean super-resolved images. Training the network to generalize to a variation of possible input data involves the approximation of a many-to-one mapping: differently blurred versions of the same real-world patch (different noise levels being strictly analogous) are supposed to map to the same clean super-resolved image patch. This implies that several points that are well separated on the input manifold map to points that are very close to each other on the output manifold.

This is the defining property of an ill-posed problem, when considering the SISR network in reverse. In order to analyze this behavior more concretely, future work could use invertible neural networks [2] to visualize the posterior distribution of blurred patches that map to a clean patch. Given posterior-sampled blurred and corresponding clean patches, the associated blur kernels could be visualized and thus analyzed, possibly illuminating the blind super-resolution capabilities of networks as proposed here.

In summary, we have experimentally shown the ability of CNNs to generalize to blind settings in the context of SISR where they can meet or surpass the performance of dedicated non-blind schemes, depending on the employed quality measure. We thus believe that training on data variations can find application in other domains involving more complex image formation models such as microscopy, light field imaging, computed tomography and more. Key to success is the faithful modeling of the forward image formation process, including both the optical and the digital parts of the process. The required large amounts of training data to cover the input and output manifolds adequately can then be synthesized from commonly available data sources.

Acknowledgement. This work was partially funded by the German Ministry for Education and Research (BMBF) under the grant PLIMASC.

References

1. Agustsson, E., Timofte, R.: NTIRE 2017 challenge on single image super-resolution: dataset and study, July 2017
2. Ardizzone, L., et al.: Analyzing inverse problems with invertible neural networks. arXiv preprint arXiv:1808.04730 (2018)
3. Boyd, S., Boyd, S.P., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
4. Brifman, A., Romano, Y., Elad, M.: Turning a denoiser into a super-resolver using plug and play priors, pp. 1404–1408 (2016)
5. Chan, S.H., Wang, X., Elgendy, O.A.: Plug and-play ADMM for image restoration: fixed-point convergence and applications. IEEE Trans. Comput. Imaging **3**, 84–98 (2017)
6. Chang, H., Yeung, D.Y., Xiong, Y.: Super-resolution through neighbor embedding, vol. 1. IEEE (2004)
7. Danielyan, A., Katkovnik, V., Egiazarian, K.: Image deblurring by augmented Lagrangian with BM3D frame prior. In: Workshop on Information Theoretic Methods in Science and Engineering, pp. 16–18 (2010)
8. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a deep convolutional network for image super-resolution. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8692, pp. 184–199. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10593-2_13
9. Dong, C., Loy, C.C., Tang, X.: Accelerating the super-resolution convolutional neural network. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 391–407. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_25
10. Foi, A., Trimeche, M., Katkovnik, V., Egiazarian, K.: Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data. IEEE Trans. Image Process. **17**(10), 1737–1754 (2008)
11. Freeman, W.T., Jones, T.R., Pasztor, E.C.: Example-based super-resolution. IEEE CG&A **22**(2), 56–65 (2002)
12. Fritsche, M., Gu, S., Timofte, R.: Frequency separation for real-world super-resolution. In: Proceedings of ICCV Workshops, pp. 3599–3608. IEEE (2019)
13. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image, pp. 349–356. IEEE (2009)
14. Guo, S., Yan, Z., Zhang, K., Zuo, W., Zhang, L.: Toward convolutional blind denoising of real photographs, pp. 1712–1722 (2019)
15. Heide, F., et al.: FlexISP: a flexible camera image processing framework. ACM Trans. Graph. (ToG) **33**, 1–13 (2014)
16. Hirsch, M., Sra, S., Schölkopf, B., Harmeling, S.: Efficient filter flow for space-variant multiframe blind deconvolution, pp. 607–614. IEEE (2010)
17. Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars, pp. 5197–5206 (2015)
18. Ji, X., Cao, Y., Tai, Y., Wang, C., J. Li, F.H.: Real-world super-resolution via kernel estimation and noise injection. In: CVPRW (2020)
19. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 694–711. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_43

20. Kim, J., Kwon Lee, J., Mu Lee, K.: Accurate image super-resolution using very deep convolutional networks, pp. 1646–1654 (2016)
21. Kligler, S., Shocher, A., Irani, M.: Blind super-resolution kernel estimation using an internal-GAN, pp. 284–293 (2019)
22. Ledig, C., et al.: Photo-realistic single image super-resolution using a generative adversarial network, pp. 4681–4690 (2017)
23. Lim, B., Son, S., Kim, H., Nah, S., Lee, K.M.: Enhanced deep residual networks for single image super-resolution, July 2017
24. Lugmayr, A., Danelljan, M., Timofte, R.: Unsupervised learning for real-world super-resolution (2019)
25. Lugmayr, A., et al.: Aim 2019 challenge on real-world image super-resolution: methods and results, pp. 3575–3583. IEEE (2019)
26. Meinhardt, T., Moller, M., Hazirbas, C., Cremers, D.: Learning proximal operators: using denoising networks for regularizing inverse imaging problems, pp. 1781–1790 (2017)
27. Michaeli, T., Irani, M.: Nonparametric blind super-resolution, pp. 945–952 (2013)
28. Parikh, N., Boyd, S.: Proximal algorithms. Found. Trends Optim. **1**(3), 127–239 (2014)
29. Plotz, T., Roth, S.: Benchmarking denoising algorithms with real photographs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1586–1595 (2017)
30. Qian, N.: On the momentum term in gradient descent learning algorithms. Neural Netw. **12**(1), 145–151 (1999)
31. Ren, H., Kheradmand, A., El-Khamy, M., Wang, S., Bai, D., Lee, J.: Real-world super-resolution using generative adversarial networks. In: CVPRW, pp. 1760–1768 (2020)
32. Shi, W., et al.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network, pp. 1874–1883 (2016)
33. Shocher, A., Cohen, N., Irani, M.: Zero-shot super-resolution using deep internal learning, pp. 3118–3126 (2018)
34. Tirer, T., Giryes, R.: Image restoration by iterative denoising and backward projections. IEEE Trans. Image Process. **28**, 1220–1234 (2019)
35. Venkatakrishnan, S.V., Bouman, C.A., Wohlberg, B.: Plug-and-play priors for model-based reconstruction, pp. 945–948. IEEE (2013)
36. Wang, X., et al.: ESRGAN: enhanced super-resolution generative adversarial networks. In: Leal-Taixé, L., Roth, S. (eds.) ECCV 2018. LNCS, vol. 11133, pp. 63–79. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11021-5_5
37. Zamir, S.W., et al.: CycleISP: real image restoration via improved data synthesis, pp. 2696–2705 (2020)
38. Zeyde, R., Elad, M., Protter, M.: On single image scale-up using sparse-representations. In: Boissonnat, J.-D., et al. (eds.) Curves and Surfaces 2010. LNCS, vol. 6920, pp. 711–730. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27413-8_47
39. Zhang, K., Zuo, W., Zhang, L.: Deep plug-and-play super-resolution for arbitrary blur kernels, pp. 1671–1681 (2019)
40. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric, pp. 586–595 (2018)
41. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 294–310. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_18
42. Zhou, R., Süsstrunk., S.: Kernel modeling superresolution on real low-resolution images, pp. 2433–2443 (2019)
43. Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration, pp. 497–486 (2011)