View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Archivio istituzionale della ricerca - Università degli Studi di Udine

the final published version of the proceedings is available on IEEE Xplore.

Deep Generative Adversarial Residual Convolutional Networks for Real-World Super-Resolution

Rao Muhammad Umer Gian Luca Foresti Christian Micheloni
University of Udine, Italy.
engr.raoumer943@gmail.com, {gianluca.foresti, christian.micheloni}@uniud.it

Abstract

Most current deep learning based single image super-resolution (SISR) methods focus on designing deeper / wider models to learn the non-linear mapping between low-resolution (LR) inputs and the high-resolution (HR) outputs from a large number of paired (LR/HR) training data. They usually take as assumption that the LR image is a bicubic down-sampled version of the HR image. However, such degradation process is not available in real-world settings i.e. *inherent sensor noise, stochastic noise, compression artifacts, possible mismatch between image degradation process and camera device. It reduces significantly the performance of current SISR methods due to real-world image corruptions. To address these problems, we propose a deep Super-Resolution Residual Convolutional Generative Adversarial Network (SRResCGAN*[1]*) to follow the real-world degradation settings by adversarial training the model with pixel-wise supervision in the HR domain from its generated LR counterpart. The proposed network exploits the residual learning by minimizing the energy-based objective function with powerful image regularization and convex optimization techniques. We demonstrate our proposed approach in quantitative and qualitative experiments that generalize robustly to real input and it is easy to deploy for other down-scaling operators and mobile/embedded devices.*

1. Introduction

The goal of the single image super-resolution (SISR) is to recover the high-resolution (HR) image from its low-resolution (LR) counterpart. SISR problem is a fundamental low-level vision and image processing problem with various practical applications in satellite imaging, medical imaging, astronomy, microscopy imaging, seismology, remote sensing, surveillance, biometric, image compression, etc. In the last decade, most of the photos are taken using built-in smartphones cameras, where resulting LR image is inevitable and undesirable due to their physical limitations. It is of great interest to restore sharp HR images because some captured moments are difficult to reproduce. On the other hand, we are also interested to design low cost (limited memory and CPU power) camera devices, where the deployment of our deep network would be possible in practice. Both are the ultimate goals to the end users.

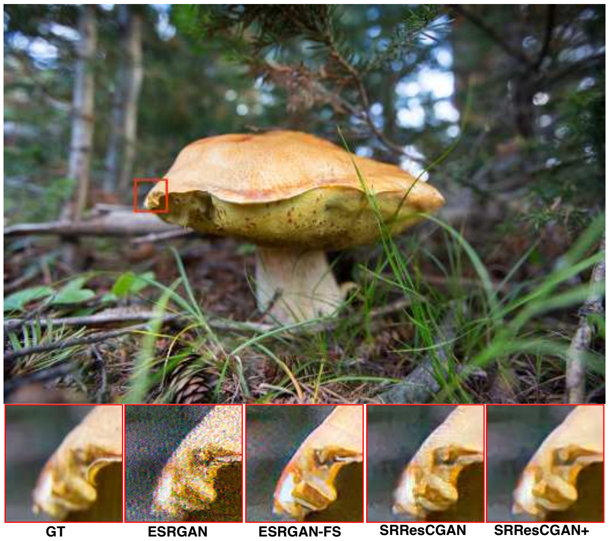


Figure 1: ×4 Super-resolution comparison of the proposed SRResCGAN method with the ESRGAN [27] and ESRGAN-FS [7] by the unknown artifacts for the '0815' image (DIV2K validation-set). Our method has better results to handle sensor noise and other artifacts, while the others have failed to remove these artifacts.

Usually, SISR is described as a linear forward observation model by the following image degradation process:

$$\mathbf{Y} = \mathbf{H}\tilde{\mathbf{X}} + \eta, \tag{1}$$

where $\mathbf{Y} \in \mathbb{R}^{N/s \times N/s}$ is an observed LR image (here $N \times N$ is typically the total number of pixels in an image), $\mathbf{H} \in \mathbb{R}^{N/s \times N/s}$ is a *down-sampling operator* (usually bicubic, circulant matrix) that resizes an HR image

[1] Our code and trained models are publicly available at https://github.com/RaoUmer/SRResCGAN

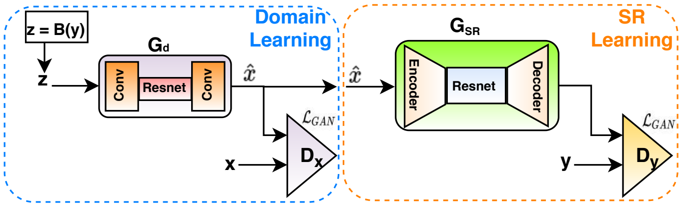


Figure 2: Visualizes the structure of the our proposed SR approach setup. In the Domain Learning part, we learn the domain distribution corruptions in the source domain ($\mathbf{x}$) by the network $\mathbf{G}_d$ (DSGAN structure), where our goal is to map images from $\mathbf{z}$ to $\mathbf{x}$, while preserving the image content. Here $\mathbf{B}$ denotes the bicubic downscaling operator which is applied on the clean HR target domain ($\mathbf{y}$) images. In the SR Learning part, we trained the network $\mathbf{G}_{SR}$ in a GAN framework by using generated LR ($\hat{\mathbf{x}}$) images from the $\mathbf{G}_d$ network with their corresponding HR images.

$\tilde{\mathbf{X}} \in \mathbb{R}^{N \times N}$ by a scaling factor s and η is considered as an additive white Gaussian noise with standard deviation σ. But in real-world settings, η also accounts for all possible errors during image acquisition process that include inherent sensor noise, stochastic noise, compression artifacts, and the possible mismatch between the forward observation model and the camera device. The operator $\mathbf{H}$ is usually ill-conditioned or singular due to the presence of unknown noises (η) that makes the SISR to highly ill-posed nature of inverse problems. Since, due to ill-posed nature, there are many possible solutions, regularization is required in order to select the most plausible ones.

Since there are many visible corruptions in the real-world images [20, 22], current state-of-art SISR methods often fail to produce convincing SR results as shown in Figure 1. Most of the existing SR methods rely on the known degradation operators such as bicubic with paired LR and HR images in the supervised training, while other methods do not follow the image observation (physical) model (refers to Eq. (1)). Three major problems arise in existing SR methods: **(1)** the first is to train the deeper/wider (lots of model's parameters) networks from a huge volume of training data, **(2)** the second is not to generalize well for natural image characteristics due to follow the known bicubic down-sampling degradation, and **(3)** it is not easy to deploy to current generations of smartphone cameras due to lots of network parameters and memory footprints. Therefore, we focus on a robust SISR method that is useful to improve the quality of images in such real-world settings.

In this work, we propose SR learning method (SRResCGAN) that strictly follows the image observation (physical) model (refers to Eq. (1)) to overcome the challenges of real-world super-resolution and is inspired by powerful image regularization and large-scale optimization techniques to solve general inverse problems (*i.e.* easy to deployable for other down-scaling operators). The visualization of our proposed SISR approach setup is shown in Figure 2. Due to the unavailability of paired (LR/HR) data, we train firstly the down-sampling (DSGAN) [7] network ($\mathbf{G}_d$) to generate LR images with same characteristics as the corrupted source domain ($\mathbf{x}$). We aim to learn the distribution (real-world) mapping from bicubically down-sampled images ($\mathbf{z}$) of HR images ($\mathbf{y}$) to the source domain images ($\mathbf{x}$), while preserving the image content. In the second part, the SR network ($\mathbf{G}_{SR}$) is trained in a GAN framework [9] by using generated LR ($\hat{\mathbf{x}}$) images with their corresponding HR images with pixel-wise supervision in the clean HR target domain ($\mathbf{y}$).

We evaluate our proposed SR method on multiple datasets with synthetic and natural image corruptions. We use the Real-World Super-resolution (RWSR) dataset [22] to show the effectiveness of our method through quantitative and qualitative experiments. Finally, we also participated in the NTIRE2020 RWSR challenges (track-1 and track-2) associated with the CVPR 2020 workshops. Table 2 shows the final testset results of the track-1 of our method (**MLP-SR**) with others, while we only provide the visual comparison of the track-2 since no ground-truth (GT) is available (refers to Fig. 5), and the quantitative results of the track-2 are in the challenge report [22].

2. Related Work

2.1. Image Super-Resolution methods

Recently, the numerous works have addressed the task of SISR using deep CNNs for their powerful feature representation capabilities. A preliminary CNN-based method to solve SISR is a super-resolution convolutional network with three layers (SRCNN) [4]. Kim *et al.* [11] proposed a very deep SR (VDSR) network with residual learning approach. The efficient sub-pixel convolutional network (ESPCNN) [24] was proposed to take bicubicly LR input and introduced an efficient sub-pixel convolution layer to upscale the LR feature maps to HR images at the end of the network. Lim *et al.* [18] proposed an enhanced deep SR (EDSR) network by taking advantage of the residual learning. Zhang *et al.* [29] proposed iterative residual convolutional network (IRCNN) to solve SISR problem by using a plug-and-play framework. Zhang *et al.* [30] proposed a deep CNN-based super-resolution with multiple degradation (SRMD). Yaoman *et al.* [17] proposed a feedback network (SRFBN) based on feedback connections and recurrent neural network like structure. In [26], the authors proposed SRWDNet to solve the joint deblurring and super-resolution task. These methods mostly rely on the PSNR-based metric by optimizing the $\mathcal{L}_1/\mathcal{L}_2$ losses with blurry results, while they do not preserve the visual quality with respect to human perception. Moreover, the above mentioned methods are deeper or wider CNN networks to learn non-linear mapping from LR to HR with a large number of training samples, while neglecting the real-world settings.

2.2. Real-World Image Super-Resolution methods

For the perception SR task, a preliminary attempt was made by Ledig *et al.* [27] who proposed the SRGAN method to produce perceptually more pleasant results. To further enhance the performance of the SRGAN, Wang *et al.* [27] proposed the ESRGAN model to achieve the state-of-art perceptual performance. Despite their success, the previously mentioned methods are trained with HR/LR image pairs on the bicubic down-sampling and thus limited performance int real-world settings. More recently, Lugmayr *et al.* [20] proposed a benchmark protocol for the real-wold image corruptions and introduced the real-world challenge series [21] that described the effects of bicubic downsampling and separate degradation learning for super-resolution. Later on, Fritsche *et al.* [7] proposed the DSGAN to learn degradation by training the network in an unsupervised way, and also modified the ESRGAN as ESRGAN-FS to further enhance it performance in the real-world settings. However, the above methods still suffer unpleasant artifacts (see the Figures 1, 4 and 6, and the Table 1). Our approach takes into account the real-world settings by greatly increase its applicability.

3. Proposed Method

3.1. Problem Formulation

By referencing to the equation (1), the recovery of $\mathbf{X}$ from $\mathbf{Y}$ mostly relies on the variational approach for combining the observation and prior knowledge, and is given as the following objective function:

$$\hat{\mathbf{E}}(\mathbf{X}) = \arg\min_{\mathbf{X}} \frac{1}{2}\|\mathbf{Y} - \mathbf{H}\mathbf{X}\|_2^2 + \lambda \mathbf{R}_W(\mathbf{X}), \quad (2)$$

where $\frac{1}{2}\|\mathbf{Y} - \mathbf{H}\mathbf{X}\|_2^2$ is the data fidelity (*i.e.* log-likelihood) term that measures the proximity of the solution to the observations, $\mathbf{R}_W(\mathbf{X})$ is regularization term that is associated with image priors, and λ is the trade-off parameter that governs the compromise between data fidelity and regularizer terms.

A generic form of the regularizers in the literature [3, 15] is given as below:

$$\mathbf{R}_W(\mathbf{X}) = \sum_{k=1}^{K} \rho_k(\mathbf{L}_k\mathbf{X}), \quad (3)$$

where $\mathbf{L}$ corresponds to the first or higher-order differential linear operators such as gradient, while $\rho(.)$ denotes a potential functions such as ℓ_p vector or matrix norms that acts on the filtered outputs [13]. Thanks to the recent advances of deep learning, the regularizer (*i.e.* $\mathbf{R}_W(\mathbf{X})$) is employed by deep convolutional neural networks (ConvNets), whose parameters are denoted by $\mathbf{W}$, that have the powerful image priors capabilities.

Besides the proper selection of the regularizer and formulation of the objective function, another important aspect of the variational approach is the minimization strategy that will be used to get the required solution. Interestingly, the variational approach has direct link to Bayesian approach and the derived solutions can be described by either as penalized maximum likelihood or as maximum a posteriori (MAP) estimates [1, 6].

3.2. Objective Function Minimization Strategy

The proper optimization strategy is employed to find $\mathbf{W}$ that minimizes the energy-based objective function to get the required latent HR image. So, we want to recover the underlying image $\mathbf{X}$ as the minimizer of the objective function in Eq. (2) as:

$$\hat{\mathbf{X}} = \arg\min_{\mathbf{X}\in\tilde{\mathbf{X}}} \hat{\mathbf{E}}(\mathbf{X}), \quad (4)$$

By referencing the Eqs. (2) and (3), we can write it as:

$$\hat{\mathbf{X}} = \arg\min_{\mathbf{X}} \frac{1}{2}\|\mathbf{Y} - \mathbf{H}\mathbf{X}\|_2^2 + \lambda \sum_{k=1}^{K} \rho_k(\mathbf{L_k}\mathbf{X}), \quad (5)$$

Since it is reasonable to require constraints on the image intensities such as non-negativity values (*i.e.* $\alpha = 0, \beta = +\infty$) that arise in the natural images, Eq. (5) can be rewritten in a constrained optimization form:

$$\hat{\mathbf{X}} = \arg\min_{\alpha\leq\mathbf{X}\leq\beta} \frac{1}{2}\|\mathbf{Y} - \mathbf{H}\mathbf{X}\|_2^2 + \lambda \sum_{k=1}^{K} \rho_k(\mathbf{L_k}\mathbf{X}), \quad (6)$$

To solve the Eq. (6), there are several modern convex-optimization schemes for large-scale machine learning problems such as HQS method [8], ADMM [2], and Proximal methods [23]. In our work, we solve the under study problem in (6) by using the Proximal Gradient Method (PGM) [23], which is a generalization of the gradient descent algorithm. PGM deals with the optimization of a function that is not fully differentiable, but it can be split into a smooth and a non-smooth part. To do so, we rewrite the problem in (6) as:

$$\hat{\mathbf{X}} = \arg\min_{\mathbf{X}} \underbrace{\frac{1}{2}\|\mathbf{Y} - \mathbf{H}\mathbf{X}\|_2^2 + \lambda \sum_{k=1}^{K} \rho_k(\mathbf{L_k}\mathbf{X})}_{\mathbf{F}(\mathbf{X})} + \iota_c(\mathbf{X}), \quad (7)$$

where ι_c is the indicator function on the convex set $\mathbb{C} \in \{\mathbf{X} \in \mathbb{R}^{N\times N} : \forall k, \alpha \leqslant \mathbf{x}_k \leqslant \beta\}$. In [15], Lefkimmiatis proposed a trainable projection layer that computes the proximal map for the indicator function as:

$$\iota_c(\mathbf{X},\ \varepsilon) = \begin{cases} 0 \quad , & if \quad \|\mathbf{X}\|_2 \leqslant \varepsilon \\ +\infty \, , & otherwise \end{cases}$$

where $\epsilon = e^{\alpha}\sigma\sqrt{C \times H \times W - 1}$ is the parametrized threshold, in which α is a trainable parameter, σ is the noise level, and $C \times H \times W$ is the total number of pixels in the image.

Thus, the solution of the problem in (7) is given by the PGM by the following update rule:

$$\mathbf{X}^t = \text{Prox}_{\gamma^t \iota_c}\left(\mathbf{X}^{(t-1)} - \gamma^t \nabla_{\mathbf{X}} \mathbf{F}(\mathbf{X}^{(t-1)})\right), \qquad (8)$$

where γ^t is a step-size and $\text{Prox}_{\gamma^t \iota_c}$ is the proximal operator [23], related to the indicator function ι_c, that is defined as:

$$\mathbf{P}_{\mathbb{C}}(\mathbf{Z}) = \arg\min_{\mathbf{X} \in \mathbf{C}} \frac{1}{2\sigma^2}\|\mathbf{X} - \mathbf{Z}\|_2^2 + \iota_c(\mathbf{X}). \qquad (9)$$

The gradient of the $\mathbf{F}(\mathbf{X})$ is computed as:

$$\nabla_{\mathbf{X}} \mathbf{F}(\mathbf{X}) = \mathbf{H}^T(\mathbf{H}\mathbf{X} - \mathbf{Y}) + \lambda \sum_{k=1}^{K} \mathbf{L_k}^T \phi_k(\mathbf{L_k}\mathbf{X}), \quad (10)$$

where $\phi_k(.)$ is the gradient of the potential function (ρ_k). By combining the Eqs. (8), (9) and (10), we have the final form of our solution as:

$$\begin{aligned} \mathbf{X}^t = \mathbf{P}_{\mathbb{C}}\Big((1 - \gamma^t \mathbf{H}^T\mathbf{H})\mathbf{X}^{(t-1)} + \gamma^t \mathbf{H}^T\mathbf{Y} - \\ \lambda\gamma^t \sum_{k=1}^{K} \mathbf{L_k}^T \phi_k(\mathbf{L_k}\mathbf{X}^{(\mathbf{t-1})})\Big), \end{aligned} \qquad (11)$$

The formulation in (11) can be thought as performing one proximal gradient descent inference step at starting points $\mathbf{Y}$ and $\mathbf{X}^{(0)} = 0$, which is given by:

$$\mathbf{X} = \mathbf{P}_{\mathbb{C}}\left(\mathbf{H}^T\mathbf{Y} - \alpha \sum_{k=1}^{K} \mathbf{L_k}^T \phi_k(\mathbf{L_k}\mathbf{Y})\right), \qquad (12)$$

where $\alpha = \lambda\gamma$ corresponds to the projection layer trainable parameter, $\mathbf{L_k}^T$ is the adjoint filter of $\mathbf{L_k}$, and $\mathbf{H}^T$ represents the up-scaling operation.

Thus, we design the generator network ($\mathbf{G}_{SR}$, refers to Fig. 3-(a)) according to Eq. (12), where $\phi_k(.)$ corresponds to a point-wise non-linearity (*i.e. PReLU*) applied to convolution feature maps. It can be noted that most of the parameters in Eq. (12) are derived from the prior term of Eq. (2), which leads to the proposed generator network as representing the most of its parameters as image priors. In order to learn valid weights of regularization parameters, the weights should be zero-mean and fixed-scale constraints. To tackle this, we use the same parametrization technique proposed in [15]. Our generator network structure can also be described as the generalization of one stage TNRD [3] and UDNet [15] that have good reconstruction performance for image denoising problem.

3.3. Domain Learning

To learn the domain distribution corruptions from the source domain ($\mathbf{x}$), we train the network $\mathbf{G}_d$ (see in the Fig. 2) in a GAN framework [9] as done in DSGAN [7] with the following loss function:

$$\mathcal{L}_{\mathbf{G}_d} = \mathcal{L}_{color} + 0.005 \cdot \mathcal{L}_{tex} + 0.01 \cdot \mathcal{L}_{per} \qquad (13)$$

where, $\mathcal{L}_{color}, \mathcal{L}_{tex}, \mathcal{L}_{per}$ denote the color loss (*i.e.* $\mathcal{L}_1$ loss focus on the low frequencies of the image), texture/GAN loss (*i.e.* focus on the high frequencies on the image), and perceptual (VGG-based) loss, respectively.

Network architectures: The generator network ($\mathbf{G}_d$) consists of 8 *Resnet* blocks (two *Conv* layers and PReLU activations in between) that are sandwiched between two *Conv* layers. All *Conv* layers have 3 × 3 kernel support with 64 feature maps. Finally, *sigmoid* non-linearity is applied on the output of the $\mathbf{G}_d$ network. While, the discriminator network ($\mathbf{D_x}$) consists of a three-layer convolutional network that operates on a patch level [10, 16]. All *Conv* layers have 5 × 5 kernel support with feature maps from 64 to 256 and also applied Batch Norm and Leaky ReLU (LReLU) activations after each *Conv* layer except the last *Conv* layer that maps 256 to 1 features.

Training description: We train the $\mathbf{G}_d$ network with image patches 512×512, which are bicubically downsampled with MATLAB *imresize* function. We randomly crop source domain images ($\mathbf{x}$) by 128 × 128 as do in [7]. We train the network for 300 epochs with a batch size of 16 using Adam optimizer [12] with parameters $\beta_1 = 0.5$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$ without weight decay for both generator and discriminator to minimize the loss in (13). The learning rate is initially set to 2.10^{-4} for first 150 epochs and then linearly decayed to zero after remaining (*i.e.* 150) epochs as do in [7].

3.4. Super-Resolution Learning

3.4.1 Network Losses

To learn the super-resolution for the target domain, we train the proposed ($\mathbf{G}_{SR}$) network in a GAN framework [9] with the following loss functions:

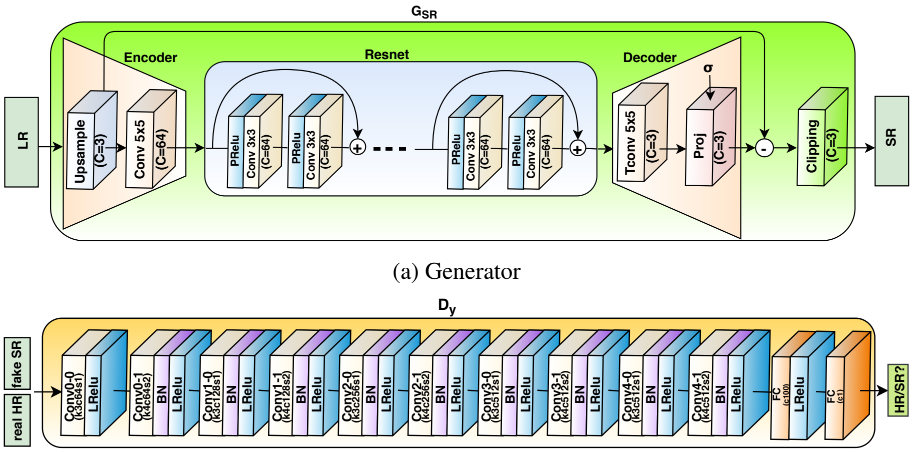
$$\mathcal{L}_{G_{SR}} = \mathcal{L}_{\text{per}} + \mathcal{L}_{\text{GAN}} + \mathcal{L}_{tv} + 10 \cdot \mathcal{L}_1 \qquad (14)$$

where, these loss functions are defined as follows:

Perceptual loss ($\mathcal{L}_{\text{per}}$): It focuses on the perceptual quality of the output image and is defined as:

$$\mathcal{L}_{\text{per}} = \frac{1}{N}\sum_{i}^{N} \mathcal{L}_{\text{VGG}} = \frac{1}{N}\sum_{i}^{N} \|\phi(\mathbf{G}_{SR}(\hat{\mathbf{x}}_i)) - \phi(\mathbf{y}_i)\|_1 \qquad (15)$$

where, ϕ is the feature extracted from the pretrained VGG-19 network at the same depth as ESRGAN [27].



(a) Generator

(b) Discriminator.

Figure 3: The architectures of Generator and Discriminator networks. The k, c, s denotes kernel size, number of filters, and stride size.

Texture loss ($\mathcal{L}_{\text{GAN}}$): It focuses on the high frequencies of the output image and is defined as:

$$\begin{aligned}\mathcal{L}_{\text{GAN}} = \mathcal{L}_{\text{RaGAN}} = &- \mathbb{E}_{\mathbf{y}}\left[\log\left(1 - \mathbf{D}_{\mathbf{y}}(\mathbf{y}, \mathbf{G}_{SR}(\hat{\mathbf{x}}))\right)\right] \\ &- \mathbb{E}_{\hat{\mathbf{y}}}\left[\log\left(\mathbf{D}_{\mathbf{y}}(\mathbf{G}_{SR}(\hat{\mathbf{x}}), \mathbf{y})\right)\right]\end{aligned} \tag{16}$$

where, $\mathbb{E}_{\mathbf{y}}$ and $\mathbb{E}_{\hat{\mathbf{y}}}$ represent the operations of taking average for all real ($\mathbf{y}$) and fake ($\hat{\mathbf{y}}$) data in the mini-batches respectively. We employed the relativistic discriminator used in the ESRGAN [27] that provides the relative GAN score of real HR and fake SR image patches and is defined as:

$$\mathbf{D}_{\mathbf{y}}(\mathbf{y}, \hat{\mathbf{y}})(C) = \sigma(C(\mathbf{y}) - \mathbb{E}[C(\hat{\mathbf{y}})]) \tag{17}$$

where, C is the raw discriminator output (see in the Fig. 3-(b)) and σ is the sigmoid function.
Content loss ($\mathcal{L}_1$): It is defined as:

$$\mathcal{L}_1 = \frac{1}{N}\sum_{i}^{N} \|\mathbf{G}_{SR}(\hat{\mathbf{x}}_i) - \mathbf{y}_i\|_1 \tag{18}$$

where, N is represents the size of mini-batch.
TV (total-variation) loss ($\mathcal{L}_{tv}$): It focuses to minimize the gradient discrepancy and produce sharpness in the output image and is defined as:

$$\begin{aligned}\mathcal{L}_{tv} = \frac{1}{N}\sum_{i}^{N} (&\|\nabla_h \mathbf{G}_{SR}\left(\hat{\mathbf{x}}_i\right) - \nabla_h\left(\mathbf{y}_i\right)\|_1 + \\ &\|\nabla_v \mathbf{G}_{SR}\left(\hat{\mathbf{x}}_i\right) - \nabla_v\left(\mathbf{y}_i\right)\|_1\end{aligned} \tag{19}$$

where, ∇_h and ∇_v denote the horizontal and vertical gradients of the images.

3.4.2 Network Architectures

Figure 3 shows the network architectures of both Generator ($\mathbf{G}_{SR}$) and Discriminator ($\mathbf{D}_{\mathbf{y}}$).
Generator ($\mathbf{G}_{SR}$): We design the generator network according to the optimization update formula in (12). In the $\mathbf{G}_{SR}$ network (refers to Fig. 3-(a)), both *Encoder (Conv, refers to $\mathbf{L_k}$ filters)* and *Decoder (TConv, refers to ${\mathbf{L_k}}^T$ filters)* layers have 64 feature maps of 5×5 kernel size with $C \times H \times W$ tensors, where C is the number of channels of the input image. Inside the *Encoder*, LR image ($\mathbf{Y}$) is upsampled by the Bilinear kernel with *Upsample* layer (refers to the operation $\mathbf{H}^T\mathbf{Y}$), where the choice of the upsampling kernel is arbitrary. *Resnet* consists of 5 residual blocks with two Pre-activation *Conv* layers, each of 64 feature maps with kernels support 3×3. The pre-activations (refers to the learnable non-linearity functions $\phi_k(.)$) are the parametrized rectified linear unit (PReLU) with 64 out feature channels support. The trainable projection (*Proj*) layer [15] (refers to the proximal operator $\mathbf{P}_{\mathbb{C}}$) inside *Decoder* computes the proximal map with the estimated noise standard deviation σ and handles the data fidelity and prior terms. Moreover, the *Proj* layer parameter α is fine-tuned during the training via a back-propagation. The noise realization is estimated in the intermediate *Resnet* that is sandwiched between *Encoder* and *Decoder*. The estimated residual image after *Decoder* is subtracted from the LR input image. Finally, the clipping layer incorporates our prior knowledge about the valid range of image intensities and enforces the pixel values of the reconstructed image to lie in the range $[0, 255]$. Reflection padding is also used before all *Conv* layers to ensure slowly-varying changes at the boundaries of the input images.

Dataset (HR/LR pairs)	SR methods	#Params	PSNR↑	SSIM↑	LPIPS↓	Artifacts
Bicubic	EDSR [18]	$43M$	24.48	0.53	0.6800	Sensor noise ($\sigma = 8$)
Bicubic	EDSR [18]	$43M$	23.75	0.62	0.5400	JPEG compression (quality=30)
Bicubic	ESRGAN [27]	$16.7M$	17.39	0.19	0.9400	Sensor noise ($\sigma = 8$)
Bicubic	ESRGAN [27]	$16.7M$	22.43	0.58	0.5300	JPEG compression (quality=30)
CycleGAN [20]	ESRGAN-FT [20]	$16.7M$	22.42	0.55	0.3645	Sensor noise ($\sigma = 8$)
CycleGAN [20]	ESRGAN-FT [20]	$16.7M$	22.80	0.57	0.3729	JPEG compression (quality=30)
DSGAN [7]	ESRGAN-FS [7]	$16.7M$	22.52	0.52	0.3300	Sensor noise ($\sigma = 8$)
DSGAN [7]	ESRGAN-FS [7]	$16.7M$	20.39	0.50	0.4200	JPEG compression (quality=30)
DSGAN [7]	SRResCGAN (ours)	$380K$	25.46	0.67	0.3604	Sensor noise ($\sigma = 8$)
DSGAN [7]	SRResCGAN (ours)	$380K$	23.34	0.59	0.4431	JPEG compression (quality=30)
DSGAN [7]	SRResCGAN+ (ours)	$380K$	26.01	0.71	0.3871	Sensor noise ($\sigma = 8$)
DSGAN [7]	SRResCGAN+ (ours)	$380K$	23.69	0.62	0.4663	JPEG compression (quality=30)
DSGAN [7]	SRResCGAN (ours)	$380K$	25.05	0.67	0.3357	unknown (validset) [22]
DSGAN [7]	SRResCGAN+ (ours)	$380K$	25.96	0.71	0.3401	unknown (validset) [22]
DSGAN [7]	ESRGAN-FS [7]	$16.7M$	20.72	0.52	0.4000	unknown (testset) [21]
DSGAN [7]	SRResCGAN (ours)	$380K$	24.87	0.68	0.3250	unknown (testset) [22]

Table 1: Top section: $\times 4$ SR quantitative results comparison of our method over the DIV2K validation-set (100 images) with added two known degradation *i.e.* sensor noise ($\sigma = 8$) and JPEG compression ($quality = 30$) artifacts. Middle section: $\times 4$ SR results with the unknown corruptions in the RWSR challenge track-1 (validation-set) [22]. Bottom section: $\times 4$ SR comparison with the unknown corruptions in the RWSR challenge series [21, 22]. The arrows indicate if high ↑ or low ↓ values are desired.

Discriminator ($\mathbf{D_y}$): The Figure 3-(b) shows the architecture of discriminator network that is trained to discriminate real HR images from generated fake SR images. The raw discriminator network contains 10 convolutional layers with kernels support 3×3 and 4×4 of increasing feature maps from 64 to 512 followed by Batch Norm (BN) and leaky ReLU as do in SRGAN [14].

3.4.3 Training description

At the training time, we set the input LR patch size as 32×32. We train the network for 51000 training iterations with a batch size of 16 using Adam optimizer [12] with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$ without weight decay for both generator and discriminator to minimize the loss in (14). The learning rate is initially set to 10^{-4} and then multiplies by 0.5 after 5K, 10K, 20K, and 30K iterations. The projection layer parameter σ is estimated according to [19] from the input LR image. We initialize the projection layer parameter α on a log-scale values from $\alpha_{max} = 2$ to $\alpha_{min} = 1$ and then further fine-tune during the training via a back-propagation.

4. Experiments

4.1. Training data

We use the source domain data ($\mathbf{x}$: 2650 HR images) that are corrupted with unknown degradation e.g. sensor noise, compression artifacts, etc. and target domain data ($\mathbf{y}$: 800 clean HR images) provided in the NTIRE2020 Real-World Super-resolution (RWSR) Challenge track1 [22]. We use the source and target domain data for training the $\mathbf{G}_d$ network to learn the domain corruptions, while due to unavailability of paired LR/HR data, we train the $\mathbf{G}_{SR}$ network (refers to section-3.4) with generated LR data ($\hat{x}$) from the $\mathbf{G}_d$ network (refers to section-3.3) with their corresponding HR target ($\mathbf{y}$) images.

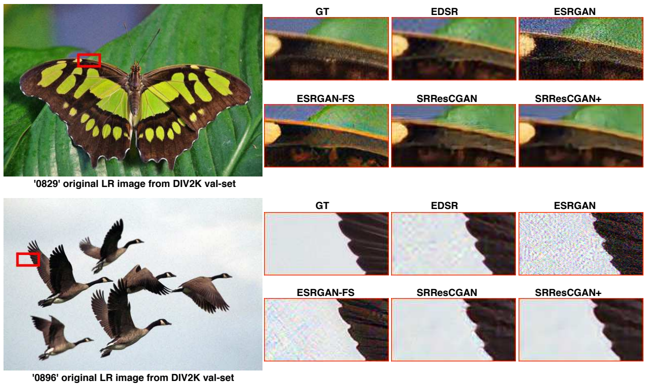

GT
EDSR
ESRGAN
ESRGAN-FS
SRResCGAN
SRResCGAN+
'0829' original LR image from DIV2K val-set
GT
EDSR
ESRGAN
ESRGAN-FS
SRResCGAN
SRResCGAN+
'0896' original LR image from DIV2K val-set

Figure 4: Visual comparison of our method with other state-of-art methods on the NTIRE2020 RWSR (track-1) validation set [22] at the $\times 4$ super-resolution.

4.2. Data augmentation

We take the input LR image patches as generated by the domain learning $\mathbf{G}_d$ network (refers to section 3.3) with their corresponding HR image patches. We augment the training data with random vertical and horizontal flipping, and 90° rotations. Moreover, we also consider another effective data augmentation technique, called *MixUp* [28]. In *Mixup*, we take randomly two samples $(\mathbf{x}_i, \mathbf{y}_i)$ and $(\mathbf{x}_j, \mathbf{y}_j)$ in the training LR/HR set $(\tilde{\mathbf{X}}, \mathbf{Y})$ and then form a new sample $(\tilde{\mathbf{x}}, \mathbf{y})$ by interpolations of the pair samples by following the same degradation model (1) as do in [5]. This simple technique encourages our network to support linear behavior among training samples.

4.3. Technical details

We implemented our method with Pytorch. The experiments are performed under Windows 10 with i7-8750H CPU with 16GB RAM and on NVIDIA RTX-2070 GPU with 8GB memory. It takes about 28.57 hours to train the model. The run time per image (on GPU) is 0.1289 seconds at the testset. In order to further enhance the fidelity, we use a self-ensemble strategy [25] (denoted as SRResCGAN+) at the test time, where the LR inputs are flipped/rotated and the SR results are aligned and averaged for enhanced prediction.

4.4. Evaluation metrics

We evaluate the trained model under the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM), and LPIPS [31] metrics. The PSNR and SSIM are distortion-based measures that correlate poorly with actual perceived similarity, while LPIPS better correlates with human perception than the distortion-based/handcrafted measures. As LPIPS is based on the features of pretrained neural networks, so we use it for the quantitative evaluation with features of AlexNet [31]. The quantitative SR results are evaluated on the RGB color space.

4.5. Comparison with the state-of-art methods

We compare our method with other state-of-art SR methods including EDSR [18], ESRGAN [27], ESRGAN-FT [20], and ESRGAN-FS [7]. Table 1 shows the quantitative results comparison of our method over the DIV2K validation-set (100 images) with two known degradation (*i.e.* sensor noise, JPEG compression) as well as unknown degradation in the RWSR challenge series [21, 22]. Our method results outperform in term of PSNR and SSIM compared to other methods, while in case of LPIPS, we are slightly behind the ESRGAN-FS (*i.e.* sensor noise ($\sigma = 8$), JPEG compression ($quality = 30$)), but ESRGAN-FS has the worst PSNR and SSIM values. We have much better LPIPS ($+0.08$) than the ESRGAN-FS (winner of AIM2019 RWSR challenge [21]) with unknown artifacts. The ESRGAN-FT has a good LPIPS value, but it achieved the worst PSNR and SSIM scores. Despite that, the parameters of the proposed $\mathbf{G}_{SR}$ network are much less (*i.e.* $\times 44$) than the other state-of-art SISR networks, which makes it suitable for deployment in mobile/embedded devices where memory storage and CPU power are limited as well as good image reconstruction quality.

Regarding the visual quality, Fig. 4 shows the qualitative comparison of our method with other SR methods on the $\times 4$ upscaling factor (validation-set). In contract to the existing state-of-art methods, our proposed method produce very good SR results that is reflected in the PSNR/SSIM/LPIPS values, as well as the visual quality of the reconstructed images with almost no visible corruptions.

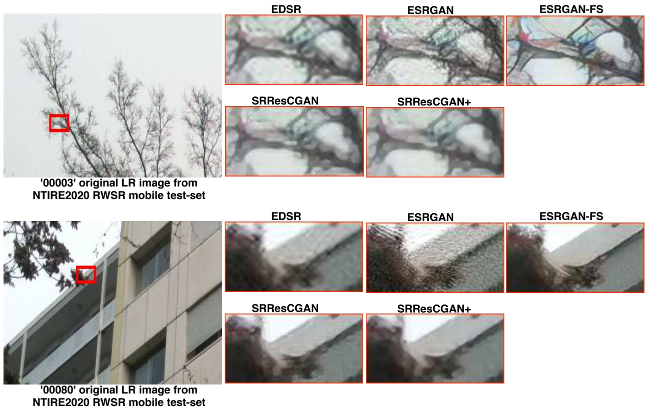

EDSR
ESRGAN
ESRGAN-FS
SRResCGAN
SRResCGAN+
'00003' original LR image from NTIRE2020 RWSR mobile test-set
EDSR
ESRGAN
ESRGAN-FS
SRResCGAN
SRResCGAN+
'00080' original LR image from NTIRE2020 RWSR mobile test-set

Figure 5: Visual comparison of our method with other state-of-art methods on the NTIRE2020 RWSR (track-2: Smartphone Images) test set [22] at the $\times 4$ super-resolution.

4.5.1 Visual comparison on the Real-World smartphone images

We also evaluate our proposed method on the real-world images captured from the smartphone provided in the RWSR challenge track-2 [22] (testset). We use the our pretrained model (refers to section-3.4) without any fine-tuning from the source domain data of the smartphone images for getting the SR results. Since there are no GT images available, we only compare the visual comparison as shown in the Fig. 5. ESRGAN still produces strong noise presence artifacts, while the EDSR produce less noisy, but more blurry results due the PSNR-based metric. ESRGAN-FS produces the sharp images and less amount of corruptions due to fine-tuning of the source domain images (*i.e.* extra training efforts). In contract, our method has still produced satisfying results by reducing the visible corruptions without any extra fine-tuning effort.

4.5.2 The NTIRE2020 RWSR Challenge

We also participated in the NTIRE2020 Real-World Super-Resolution (RWSR) Challenge [22] associated with the CVPR 2020 workshops. The goal of this challenge is to super-resolve ($\times 4$) images from the Source Domain (corrupted) to the Target Domain (clean). We train firstly the domain learning model ($\mathbf{G}_d$) on the corrupted source domain dataset to learn visible corruptions, and after that train the SR learning model ($\mathbf{G}_{SR}$) on the clean target domain dataset with their correspond generated LR pairs from the ($\mathbf{G}_d$) model (refers to the sections-3.3 and 3.4 for more details). Table 2 provides the final $\times 4$ SR results for track-1 (testset) of our method (**MLP-SR**) with others. The final ranking is based on the Mean Opinion Score (MOS) [22]. Our method remains among the top 7 best solutions. We also provide the visual comparison of our method with others on the track-1 testset in the Fig. 6. Our method produces

Team	PSNR↑	SSIM↑	LPIPS↓	MOS↓
Impressionism	24.67 (16)	0.683 (13)	0.232 (1)	2.195
Samsung-SLSI-MSL	25.59 (12)	0.727 (9)	0.252 (2)	2.425
BOE-IOT-AIBD	26.71 (4)	0.761 (4)	0.280 (4)	2.495
MSMers	23.20 (18)	0.651 (17)	0.272 (3)	2.530
KU-ISPL	26.23 (6)	0.747 (7)	0.327 (8)	2.695
InnoPeak-SR	26.54 (5)	0.746 (8)	0.302 (5)	2.740
ITS425	27.08 (2)	0.779 (1)	0.325 (6)	2.770
MLP-SR	24.87 (15)	0.681 (14)	0.325 (7)	2.905
Webbzhou	26.10 (9)	0.764 (3)	0.341 (9)	-
SR-DL	25.67 (11)	0.718 (10)	0.364 (10)	-
TeamAY	27.09 (1)	0.773 (2)	0.369 (11)	-
BIGFEATURE-CAMERA	26.18 (7)	0.750 (6)	0.372 (12)	-
BMIPL-UNIST-YH-1	26.73 (3)	0.752 (5)	0.379 (13)	-
SVNIT1-A	21.22 (19)	0.576 (19)	0.397 (14)	-
KU-ISPL2	25.27 (14)	0.680 (15)	0.460 (15)	-
SuperT	25.79 (10)	0.699 (12)	0.469 (16)	-
GDUT-wp	26.11 (8)	0.706 (11)	0.496 (17)	-
SVNIT1-B	24.21 (17)	0.617 (18)	0.562 (18)	-
SVNIT2	25.39 (13)	0.674 (16)	0.615 (19)	-
AITA-Noah-A	24.65 (-)	0.699 (-)	0.222 (-)	2.245
AITA-Noah-B	25.72 (-)	0.737 (-)	0.223 (-)	2.285
Bicubic	25.48 (-)	0.680 (-)	0.612 (-)	3.050
ESRGAN Supervised	24.74 (-)	0.695 (-)	0.207 (-)	2.300

Table 2: Final testset results for the RWSR challenge Track-1. The top section in the table contains ours (**MLP-SR**) with other methods that are ranked in the challenge. The middle section contains participating approaches that deviated from the challenge rules, whose results are reported for reference but not ranked. The bottom section contains baseline approaches. Participating methods are ranked according to their Mean Opinion Score (MOS).

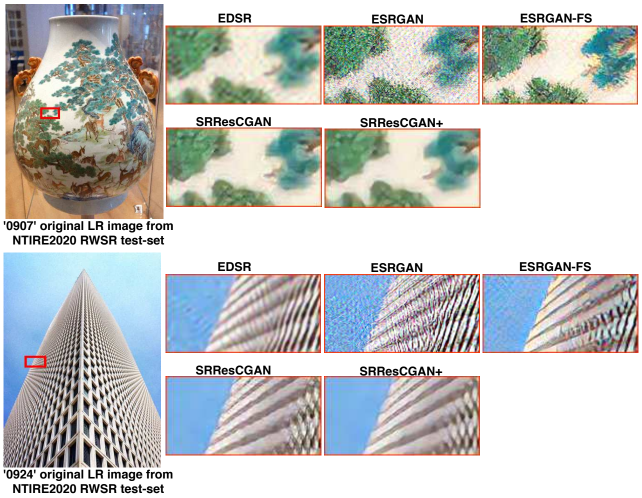


Figure 6: Visual comparison of our method with other state-of-art methods on the NTIRE2020 RWSR (track-1: Image Processing Artifacts) testset [22] at the ×4 super-resolution.

sharp images without any visible corruptions, while the others suffer image corruptions.

4.6. Ablation Study

For our ablation study, we compare the different combinations of losses of the proposed SR learning model ($\mathbf{G}_{SR}$). We consider the LPIPS measure for its better visual correlation with the human perception. Table 3 shows the quantitative results of our method over the DIV2K validation-set (track-1) [22] with unknown degradation. We first train the SR model with combination of the losses ($\mathcal{L}_{\text{per}}, \mathcal{L}_{\text{GAN}}, \mathcal{L}_1$) similar to ESRGAN. After that, we add $\mathcal{L}_{tv}$ to the previous combinations, and train the model again, we obtain little a bit better LPIPS with sharp SR images. Finally, when we apply the high-pass filter ($\mathbf{w}_H$) weights to the output image to compute the GAN loss ($\mathcal{L}_{\text{GAN}}$) focus on the high-frequencies with the previous combinations during training the network, we get the best LPIPS value (*i.e.* $+0.01$ improvement to the previous variants) with more realistic SR images. Therefore, we opt the last one as the final combination of loss functions for our model ($\mathbf{G}_{SR}$) training and also used for evaluation in the section 4.5.

		unknown artifacts		
SR method	SR Generator Loss combinations ($\mathcal{L}_{G_{SR}}$)	PSNR↑	SSIM↑	LPIPS↓
SRResCGAN	$\mathcal{L}_{\text{per}} + \mathcal{L}_{\text{GAN}} + 10 \cdot \mathcal{L}_1$	25.48	0.69	0.3458
SRResCGAN	$\mathcal{L}_{\text{per}} + \mathcal{L}_{\text{GAN}} + \mathcal{L}_{tv} + 10 \cdot \mathcal{L}_1$	25.40	0.69	0.3452
SRResCGAN	$\mathcal{L}_{\text{per}} + \mathbf{w}_H * \mathcal{L}_{\text{GAN}} + \mathcal{L}_{tv} + 10 \cdot \mathcal{L}_1$	25.05	0.67	0.3357

Table 3: This table reports the quantitative results of our method over the DIV2K validation set (100 images) with unknown degradation for our ablation study. The arrows indicate if high ↑ or low ↓ values are desired.

5. Conclusion

We proposed a deep SRResCGAN method for the real-world super-resolution task by following the image observation (physical / real-world settings) model. The proposed method solves the SR problem in a GAN framework by minimizing the energy-based objective function with the discriminative and residual learning approaches. The proposed method exploits the powerful image regularization and large-scale optimization techniques for image restoration. Our method achieves very good SR results in terms of the PSNR/SSIM/LPIPS values as well as visual quality compared to the existing state-of-art methods. The proposed method follows the real-world settings for limited memory storage and CPU power requirements (*i.e.* 44 times less number of parameters than the others) for the mobile/embedded deployment.

Acknowledgement

This work is supported by EU H2020 MSCA through Project ACHIEVE-ITN (Grant No 765866).

References

[1] Mario Bertero and Patrizia Boccacci. *Introduction to inverse problems in imaging*. CRC press, 1998. 3

[2] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, pages 1–122, 2011. 3

[3] Yunjin Chen and Thomas Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1256–1272, 2017. 3, 4

[4] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. *ECCV*, pages 184–199, 2014. 2

[5] Ruicheng Feng, Jinjin Gu, Yu Qiao, and Chao Dong. Suppressing model overfitting for image super-resolution networks. *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 0–0, 2019. 6

[6] Mário Figueiredo, José M Bioucas-Dias, and Robert D Nowak. Majorization–minimization algorithms for wavelet-based image restoration. *IEEE Transactions on Image processing*, 16(12):2980–2991, 2007. 3

[7] Manuel Fritsche, Shuhang Gu, and Radu Timofte. Frequency separation for real-world super-resolution. *ICCV workshop*, 2019. 1, 2, 3, 4, 6, 7

[8] D. Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing (TIP)*, pages 932–946, July 1995. 3

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems (NIPS)*, pages 2672–2680, 2014. 2, 4

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, pages 1125–1134, 2017. 4

[11] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. *CVPR*, pages 1646–1654, 2016. 2

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, pages 1–15, 2015. 4, 6

[13] Dilip Krishnan and Rob Fergus. Fast image deconvolution using hyper-laplacian priors. *NIPS*, 2009. 3

[14] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *CVPR*, pages 4681–4690, 2017. 6

[15] Stamatios Lefkimmiatis. Universal denoising networks: A novel cnn architecture for image denoising. *CVPR*, pages 3204–3213, 2018. 3, 4, 5

[16] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. *ECCV*, pages 702–716, 2016. 4

[17] Yaoman Li, Jinglei Yang, Zheng Liu, Xiaomin Yang, Gwanggil Jeon, and Wei Wu. Feedback network for image super-resolution. *CVPR*, 2019. 2

[18] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. *CVPRW*, pages 1132–1140, 2017. 2, 6, 7

[19] Xinhao Liu, Masayuki Tanaka, and Masatoshi Okutomi. Single-image noise level estimation for blind denoising. *IEEE Transactions on Image Processing (TIP)*, pages 5226–5237, 2013. 6

[20] Andreas Lugmayr, Martin Danelljan, and Radu Timofte. Unsupervised learning for real-world super-resolution. *ICCV workshops*, 2019. 2, 3, 6, 7

[21] Andreas Lugmayr, Martin Danelljan, Radu Timofte, et al. Aim 2019 challenge on real-world image super-resolution: Methods and results. In *ICCV Workshops*, 2019. 3, 6, 7

[22] Andreas Lugmayr, Martin Danelljan, Radu Timofte, et al. Ntire 2020 challenge on real-world image super-resolution: Methods and results. *CVPR Workshops*, 2020. 2, 6, 7, 8

[23] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and Trends in Optimization*, pages 127–239, 2014. 3, 4

[24] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016. 2

[25] Radu Timofte, Rasmus Rothe, and Luc Van Gool. Seven ways to improve example-based single image super resolution. In *CVPR*, pages 1865–1873, 2016. 7

[26] Rao Muhammad Umer, Gian Luca Foresti, and Christian Micheloni. Deep super-resolution network for single image super-resolution with realistic degradations. In *ICDSC*, pages 21:1–21:7, September 2019. 2

[27] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. ESRGAN: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 1, 3, 4, 5, 6, 7

[28] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. *International Conference on Learning Representations (ICLR)*, 2018. 6

[29] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2808–2817, 2017. 2

[30] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Learning a single convolutional super-resolution network for multiple degradations. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3262–3271, 2018. 2

[31] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018. 7