Deep Laplacian Pyramid Networks for Fast and Accurate Super-Resolution

Wei-Sheng Lai[1] Jia-Bin Huang[2] Narendra Ahuja[3] Ming-Hsuan Yang[1]
[1]University of California, Merced [2]Virginia Tech [3]University of Illinois, Urbana-Champaign
http://vllab.ucmerced.edu/wlai24/LapSRN

Abstract

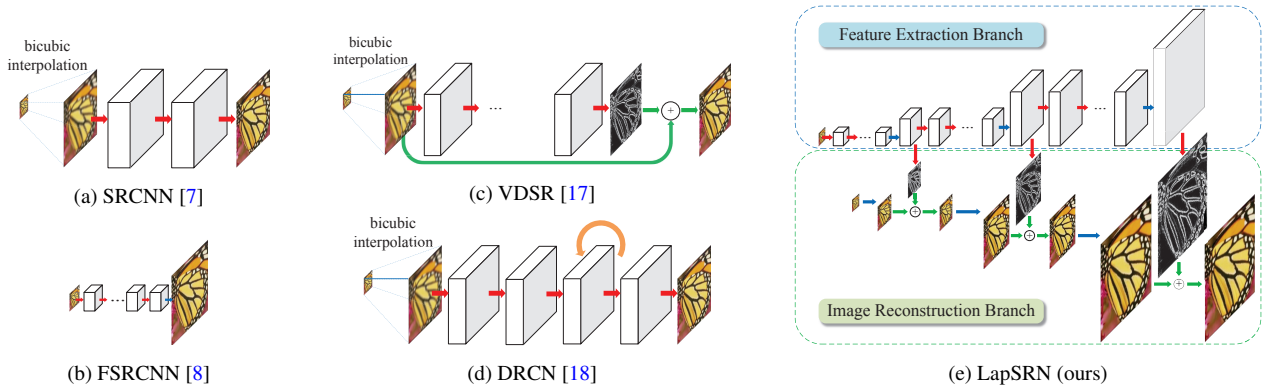
Convolutional neural networks have recently demonstrated high-quality reconstruction for single-image super-resolution. In this paper, we propose the Laplacian Pyramid Super-Resolution Network (LapSRN) to progressively reconstruct the sub-band residuals of high-resolution images. At each pyramid level, our model takes coarse-resolution feature maps as input, predicts the high-frequency residuals, and uses transposed convolutions for upsampling to the finer level. Our method does not require the bicubic interpolation as the pre-processing step and thus dramatically reduces the computational complexity. We train the proposed LapSRN with deep supervision using a robust Charbonnier loss function and achieve high-quality reconstruction. Furthermore, our network generates multi-scale predictions in one feed-forward pass through the progressive reconstruction, thereby facilitates resource-aware applications. Extensive quantitative and qualitative evaluations on benchmark datasets show that the proposed algorithm performs favorably against the state-of-the-art methods in terms of speed and accuracy.

1. Introduction

Single-image super-resolution (SR) aims to reconstruct a high-resolution (HR) image from a single low-resolution (LR) input image. In recent years, example-based SR methods have demonstrated the state-of-the-art performance by learning a mapping from LR to HR image patches using large image databases. Numerous learning algorithms have been applied to learn such a mapping, including dictionary learning [37, 38], local linear regression [30, 36], and random forest [26].

Recently, Dong et al. [7] propose a Super-Resolution Convolutional Neural Network (SRCNN) to learn a nonlinear LR-to-HR mapping. The network is extended to embed a sparse coding-based network [33] or use a deeper structure [17]. While these models demonstrate promising results, there are three main issues. First, existing methods use a pre-defined upsampling operator, e.g., bicubic interpolation, to upscale input images to the desired spatial resolution before applying the network for prediction. This pre-processing step increases unnecessary computational cost and often results in visible reconstruction artifacts. Several algorithms accelerate SRCNN by performing convolution on LR images and replacing the pre-defined upsampling operator with sub-pixel convolution [28] or transposed convolution [8] (also named as deconvolution in some of the literature). These methods, however, use relatively small networks and cannot learn complicated mappings well due to the limited network capacity. Second, existing methods optimize the networks with an ℓ_2 loss and thus inevitably generate blurry predictions. Since the ℓ_2 loss fails to capture the underlying multi-modal distributions of HR patches (i.e., the same LR patch may have many corresponding HR patches), the reconstructed HR images are often overly-smooth and not close to human visual perception on natural images. Third, most methods reconstruct HR images *in one upsampling step*, which increases the difficulties of training for large scaling factors (e.g., $8\times$). In addition, existing methods cannot generate intermediate SR predictions at multiple resolutions. As a result, one needs to train a large variety of models for various applications with different desired upsampling scales and computational loads.

To address these drawbacks, we propose the Laplacian Pyramid Super-Resolution Network (LapSRN) based on a cascade of convolutional neural networks (CNNs). Our network takes an LR image as input and progressively predicts the sub-band residuals in a coarse-to-fine fashion. At each level, we first apply a cascade of convolutional layers to extract feature maps. We then use a transposed convolutional layer for upsampling the feature maps to a finer level. Finally, we use a convolutional layer to predict the sub-band residuals (the differences between the upsampled image and the ground truth HR image at the respective level). The predicted residuals at each level are used to efficiently reconstruct the HR image through upsampling and addition operations. While the proposed LapSRN consists of a set of cascaded sub-networks, we train the network with a robust Charbonnier loss function in an end-to-end fashion (i.e., without stage-wise optimization). As depicted in Fig-



(a) SRCNN [7] (c) VDSR [17]

(b) FSRCNN [8] (d) DRCN [18] (e) LapSRN (ours)

Figure 1: Network architectures of SRCNN [7], FSRCNN [8], VDSR [17], DRCN [18] and the proposed LapSRN. Red arrows indicate convolutional layers. Blue arrows indicate transposed convolutions (upsampling). Green arrows denote element-wise addition operators, and the orange arrow indicates recurrent layers.

ure 1(e), our network architecture naturally accommodates deep supervision (i.e., supervisory signals can be applied simultaneously at each level of the pyramid).

Our algorithm differs from existing CNN-based methods in the following three aspects:

(1) **Accuracy**. The proposed LapSRN extracts feature maps directly from LR images and jointly optimizes the upsampling filters with deep convolutional layers to predict sub-band *residuals*. The deep supervision with the Charbonnier loss improves the performance thanks to the ability to better handle outliers. As a result, our model has a large capacity to learn complicated mappings and effectively reduces the undesired visual artifacts.

(2) **Speed**. Our LapSRN embraces both fast processing speed and high capacity of deep networks. Experimental results demonstrate that our method is faster than several CNN based super-resolution models, e.g., SRCNN [7], SCN [33], VDSR [17], and DRCN [18]. Similar to FSRCNN [8], our LapSRN achieves real-time speed on most of the evaluated datasets. In addition, our method provides significantly better reconstruction accuracy.

(3) **Progressive reconstruction**. Our model generates multiple intermediate SR predictions in *one* feed-forward pass through progressive reconstruction using the Laplacian pyramid. This characteristic renders our technique applicable to a wide range of applications that require resource-aware adaptability. For example, the same network can be used to enhance the spatial resolution of videos depending on the available computational resources. For scenarios with limited computing resources, our $8\times$ model can still perform $2\times$ or $4\times$ SR by simply bypassing the computation of residuals at finer levels. Existing CNN-based methods, however, do not offer such flexibility.

2. Related Work and Problem Context

Numerous single-image super-resolution methods have been proposed in the literature. Here we focus our discussion on recent example-based approaches.

SR based on internal databases. Several methods [9, 12] exploit the self-similarity property in natural images and construct LR-HR patch pairs based on the scale-space pyramid of the low-resolution input image. While internal databases contain more relevant training patches than external image databases, the number of LR-HR patch pairs may not be sufficient to cover large textural variations in an image. Singh et al. [29] decompose patches into directional frequency sub-bands and determine better matches in each sub-band pyramid independently. Huang et al. [15] extend the patch search space to accommodate the affine transform and perspective deformation. The main drawback of SR methods based on internal databases is that they are typically slow due to the heavy computational cost of patch search in the scale-space pyramid.

SR based on external databases. Numerous SR methods learn the LR-HR mapping with image pairs collected from external databases using supervised learning algorithms, such as nearest neighbor [10], manifold embedding [2, 5], kernel ridge regression [19], and sparse representation [37, 38, 39]. Instead of directly modeling the complex patch space over the entire database, several methods partition the image database by K-means [36], sparse dictionary [30] or random forest [26], and learn locally linear regressors for each cluster.

Convolutional neural networks based SR. In contrast to modeling the LR-HR mapping in the *patch* space, SRCNN [7] jointly optimize all the steps and learn the non-linear mapping in the *image* space. The VDSR network [17] demonstrates significant improvement over SRCNN [7] by increasing the network depth from 3 to 20 convolutional layers. To facilitate training a deeper model with a fast

Table 1: Comparisons of CNN based SR algorithms: SRCNN [7], FSRCNN [8], SCN [33], ESPCN [28], VDSR [17], and the proposed LapSRN. The number of layers includes both convolution and transposed convolution. Methods with direct reconstruction performs one-step upsampling (with bicubic interpolation or transposed convolution) from LR to HR images, while progressive reconstruction predicts HR images in multiple steps.

Method	Network input	#Layers	Residual learning	Reconstruction	Loss function
SRCNN [7]	LR + bicubic	3	No	Direct	L2
FSRCNN [8]	LR	8	No	Direct	L2
SCN [33]	LR + bicubic	5	No	Progressive	L2
ESPCN [28]	LR	3	No	Direct	L2
VDSR [17]	LR + bicubic	20	Yes	Direct	L2
DRCN [18]	LR + bicubic	5 (recursive)	No	Direct	L2
LapSRN (ours)	LR	27	Yes	Progressive	Charbonnier

convergence speed, VDSR trains the network to predict the residuals rather the actual pixel values. Wang et al. [33] combine the domain knowledge of sparse coding with a deep CNN and train a cascade network (SCN) to upsample images to the desired scale factor progressively. Kim et al. [18] propose a shallow network with deeply recursive layers (DRCN) to reduce the number of parameters.

To achieve real-time performance, the ESPCN network [28] extracts feature maps in the LR space and replaces the bicubic upsampling operation with an efficient sub-pixel convolution. The FSRCNN network [8] adopts a similar idea and uses a hourglass-shaped CNN with more layers but fewer parameters than that in ESPCN. All the above CNN-based SR methods optimize networks with an ℓ_2 loss function, which often leads to overly-smooth results that do not correlate well with human perception. In the context of SR, we demonstrate that the ℓ_2 loss is less effective for learning and predicting sparse residuals.

We compare the network structures of SRCNN, FSRCNN, VDSR, DRCN and our LapSRN in Figure 1 and list the main differences among existing CNN-based methods and the proposed framework in Table 1. Our approach builds upon existing CNN-based SR algorithms with three main differences. First, we jointly learn residuals and upsampling filters with convolutional and transposed convolutional layers. Using the learned upsampling filters not only effectively suppresses reconstruction artifacts caused by the bicubic interpolation, but also dramatically reduces the computational complexity. Second, we optimize the deep network using a robust Charbonnier loss function instead of the ℓ_2 loss to handle outliers and improve the reconstruction accuracy. Third, as the proposed LapSRN progressively reconstructs HR images, the same model can be used for applications that require different scale factors by truncating the network up to a certain level.

Laplacian pyramid. The Laplacian pyramid has been used in a wide range of applications, such as image blending [4], texture synthesis [14], edge-aware filtering [24] and semantic segmentation [11, 25]. Denton et al. propose a generative model based on a Laplacian pyramid framework (LAPGAN) to generate realistic images in [6], which is the most related to our work. However, the proposed LapSRN differs from LAPGAN in three aspects.

First, LAPGAN is a *generative model* which is designed to synthesize diverse natural images from random noise and sample inputs. On the contrary, our LapSRN is a super-resolution model that predicts a particular HR image based on the given LR image. LAPGAN uses a cross-entropy loss function to encourage the output images to respect the data distribution of training datasets. In contrast, we use the Charbonnier penalty function to penalize the deviation of the prediction from the ground truth sub-band residuals.

Second, the sub-networks of LAPGAN are *independent* (i.e., no weight sharing). As a result, the network capacity is limited by the depth of each sub-network. Unlike LAPGAN, the convolutional layers at each level in LapSRN are *connected* through multi-channel transposed convolutional layers. The residual images at a higher level are therefore predicted by a deeper network with shared feature representations at lower levels. The feature sharing at lower levels increases the non-linearity at finer convolutional layers to learn complex mappings. Also, the sub-networks in LAPGAN are *independently trained*. On the other hand, all the convolutional filters for feature extraction, upsampling, and residual prediction layers in the LapSRN are *jointly trained* in an end-to-end, deeply supervised fashion.

Third, LAPGAN applies convolutions on the upsampled images, so the speed depends on the size of HR images. On the contrary, our design of LapSRN effectively increases the size of the receptive field and accelerates the speed by extracting features from the LR space. We provide comparisons with LAPGAN in the supplementary material.

Adversarial training. The SRGAN method [20] optimizes the network using the perceptual loss [16] and the adversarial loss for photo-realistic SR. We note that our LapSRN can be easily extended to the adversarial training framework. As it is not our contribution, we provide experiments on the adversarial loss in the supplementary material.

3. Deep Laplacian Pyramid Network for SR

In this section, we describe the design methodology of the proposed Laplacian pyramid network, the optimization using robust loss functions with deep supervision, and the details for network training.

3.1. Network architecture

We propose to construct our network based on the Laplacian pyramid framework, as shown in Figure 1(e). Our model takes an LR image as input (rather than an upscaled version of the LR image) and progressively predicts residual images at $\log_2 S$ levels where S is the scale factor. For example, the network consists of 3 sub-networks for super-resolving an LR image at a scale factor of 8. Our model has two branches: (1) feature extraction and (2) image reconstruction.

Feature extraction. At level s, the feature extraction branch consists of d convolutional layers and one transposed convolutional layer to upsample the extracted features by a scale of 2. The output of each transposed convolutional layer is connected to two different layers: (1) a convolutional layer for reconstructing a residual image at level s, and (2) a convolutional layer for extracting features at the finer level $s+1$. Note that we perform the feature extraction at the *coarse* resolution and generate feature maps at the *finer* resolution with only one transposed convolutional layer. In contrast to existing networks that perform all feature extraction and reconstruction at the fine resolution, our network design significantly reduces the computational complexity. Note that the feature representations at lower levels are shared with higher levels, and thus can increase the non-linearity of the network to learn complex mappings at the finer levels.

Image reconstruction. At level s, the input image is upsampled by a scale of 2 with a transposed convolutional (upsampling) layer. We initialize this layer with the bilinear kernel and allow it to be jointly optimized with all the other layers. The upsampled image is then combined (using element-wise summation) with the predicted residual image from the feature extraction branch to produce a high-resolution output image. The output HR image at level s is then fed into the image reconstruction branch of level $s+1$. The entire network is a cascade of CNNs with a similar structure at each level.

3.2. Loss function

Let x be the input LR image and θ be the set of network parameters to be optimized. Our goal is to learn a mapping function f for generating a high-resolution image $\hat{y} = f(x;\theta)$ that is close to the ground truth HR image y. We denote the residual image at level s by r_s, the upscaled LR image by x_s and the corresponding HR images by y_s. The desired output HR images at level s is modeled by $y_s = x_s + r_s$. We use the bicubic downsampling to resize the ground truth HR image y to y_s at each level. Instead of minimizing the mean square errors between y_s and $\hat{y}_s$, we propose to use a robust loss function to handle outliers. The overall loss function is defined as:

$$\begin{aligned} \mathscr{L}(\hat{y}, y; \theta) &= \frac{1}{N}\sum_{i=1}^{N}\sum_{s=1}^{L}\rho\left(\hat{y}_s^{(i)} - y_s^{(i)}\right) \\ &= \frac{1}{N}\sum_{i=1}^{N}\sum_{s=1}^{L}\rho\left((\hat{y}_s^{(i)} - x_s^{(i)}) - r_s^{(i)}\right), \end{aligned} \tag{1}$$

where $\rho(x) = \sqrt{x^2+\varepsilon^2}$ is the Charbonnier penalty function (a differentiable variant of ℓ_1 norm) [3], N is the number of training samples in each batch, and L is the number of level in our pyramid. We empirically set ε to $1e-3$.

In the proposed LapSRN, each level s has its loss function and the corresponding ground truth HR image y_s. This multi-loss structure resembles the deeply-supervised nets for classification [21] and edge detection [34]. However, the labels used to supervise intermediate layers in [21, 34] are the *same* across the networks. In our model, we use *different* scales of HR images at the corresponding level as supervision. The deep supervision guides the network training to predict sub-band residual images at different levels and produce multi-scale output images. For example, our $8\times$ model can produce $2\times$, $4\times$ and $8\times$ super-resolution results in *one* feed-forward pass. This property is particularly useful for resource-aware applications, e.g., mobile devices or network applications.

3.3. Implementation and training details

In the proposed LapSRN, each convolutional layer consists of 64 filters with the size of 3×3. We initialize the convolutional filters using the method of He et al. [13]. The size of the transposed convolutional filters is 4×4 and the weights are initialized from a bilinear filter. All the convolutional and transposed convolutional layers (except the reconstruction layers) are followed by leaky rectified linear units (LReLUs) with a negative slope of 0.2. We pad zeros around the boundaries before applying convolution to keep the size of all feature maps the same as the input of each level. The convolutional filters have small spatial supports (3×3). However, we can achieve high non-linearity and increase the size of receptive fields with a deep structure.

We use 91 images from Yang et al. [38] and 200 images from the training set of Berkeley Segmentation Dataset [1] as our training data. The same training dataset is used in [17, 26] as well. In each training batch, we randomly sample 64 patches with the size of 128×128. An epoch has $1,000$ iterations of back-propagation. We augment the training data in three ways: (1) *Scaling*: randomly downscale between $[0.5, 1.0]$. (2) *Rotation*: randomly rotate image by 90°, 180°, or 270°. (3) *Flipping*: flip images horizontally or vertically with a probability of 0.5. Following

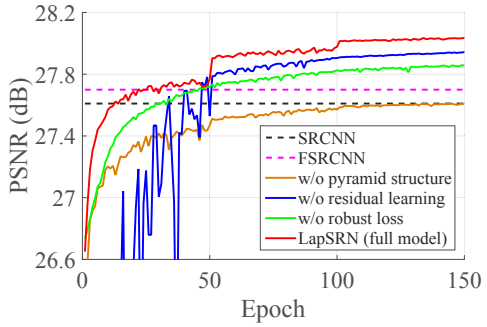


Figure 2: Convergence analysis on the pyramid structure, loss functions and residual learning. Our LapSRN converges faster and achieves improved performance.

Table 2: Ablation study of pyramid structures, loss functions, and residual learning. We replace each component with the one used in existing methods, and observe performance (PSNR) drop on both SET5 and SET14.

Residual	Pyramid	Loss	SET5	SET14
✓		Robust	30.58	27.61
	✓	Robust	31.10	27.94
✓	✓	ℓ_2	30.93	27.86
✓	✓	Robust	**31.28**	**28.04**

the protocol of existing methods [7, 17], we generate the LR training patches using the bicubic downsampling. We train our model with the MatConvNet toolbox [31]. We set momentum parameter to 0.9 and the weight decay to $1e-4$. The learning rate is initialized to $1e-5$ for all layers and decreased by a factor of 2 for every 50 epochs.

4. Experiment Results

We first analyze the contributions of different components of the proposed network. We then compare our LapSRN with state-of-the-art algorithms on five benchmark datasets and demonstrate the applications of our method on super-resolving real-world photos and videos.

4.1. Model analysis

Residual learning. To demonstrate the effect of residual learning, we remove the image reconstruction branch and directly predict the HR images at each level. Figure 2 shows the convergence curves in terms of PSNR on the SET14 for $4\times$ SR. The performance of the "non-residual" network (blue curve) converges slowly and fluctuates significantly. The proposed LapSRN (red curve), on the other hand, outperforms SRCNN within 10 epochs.

Loss function. To validate the effect of the Charbonnier loss function, we train the proposed network with the ℓ_2 loss function. We use a larger learning rate ($1e-4$) since the gradient magnitude of the ℓ_2 loss is smaller. As illustrated in Figure 2, the network optimized with ℓ_2 loss (green curve) requires more iterations to achieve comparable performance with SRCNN. In Figure 3(d), we show that the network trained with the ℓ_2 loss generates SR results with more ringing artifacts. In contrast, the SR images reconstruct by the proposed algorithm (Figure 3(e)) contain relatively clean and sharp details.



Figure 3: Contribution of different components in the proposed network. (a) HR image. (b) w/o pyramid structure (c) w/o residual learning (d) w/o robust loss (e) full model (f) ground truth.

Table 3: Trade-off between performance and speed on the depth at each level of the proposed network.

Depth	SET5		SET14	
	PSNR	Second	PSNR	Second
3	31.15	0.036	27.98	0.036
5	31.28	0.044	28.04	0.042
10	31.37	0.050	28.11	0.051
15	31.45	0.077	28.16	0.071

Pyramid structure. By removing the pyramid structure, our model falls back to a network similar to FSRCNN but with the residual learning. To use the same number of convolutional layers as LapSRN, we train a network with 10 convolutional layers and one transposed convolutional layer. The quantitative results in Table 2 shows that the pyramid structure leads to moderate performance improvement (e.g. 0.7 dB on SET5 and 0.4 dB on SET14).

Network depth. We train the proposed model with different depth, $d = 3, 5, 10, 15$, at each level and show the trade-offs between performance and speed in Table 3. In general, deep networks perform better shallow ones at the expense of increased computational cost. We choose $d = 10$ for our $2\times$ and $4\times$ SR models to strike a balance between performance and speed. We show that the speed of our LapSRN with $d = 10$ is faster than most of the existing CNN-based SR algorithms (see Figure 6). For $8\times$ model, we choose $d = 5$ because we do not observe significant performance gain by using more convolutional layers.

4.2. Comparisons with the state-of-the-arts

We compare the proposed LapSRN with 8 state-of-the-art SR algorithms: A+ [30], SRCNN [7], FSRCNN [8], SelfExSR [15], RFL [26], SCN [33], VDSR [17] and DRCN [18]. We carry out extensive experiments using 5 datasets: SET5 [2], SET14 [39], BSDS100 [1], URBAN100 [15] and MANGA109 [23]. Among these datasets,

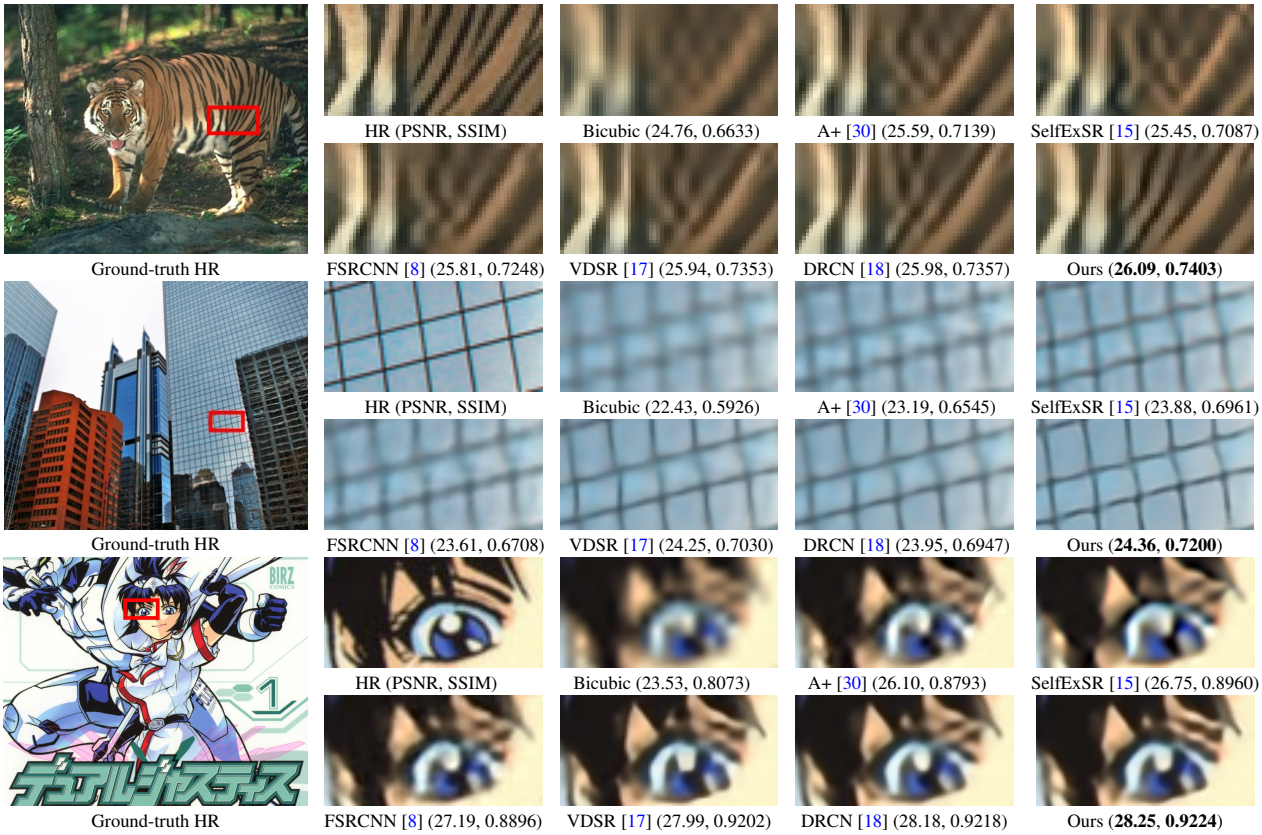


Figure 4: Visual comparison for 4× SR on BSDS100, URBAN100 and MANGA109.

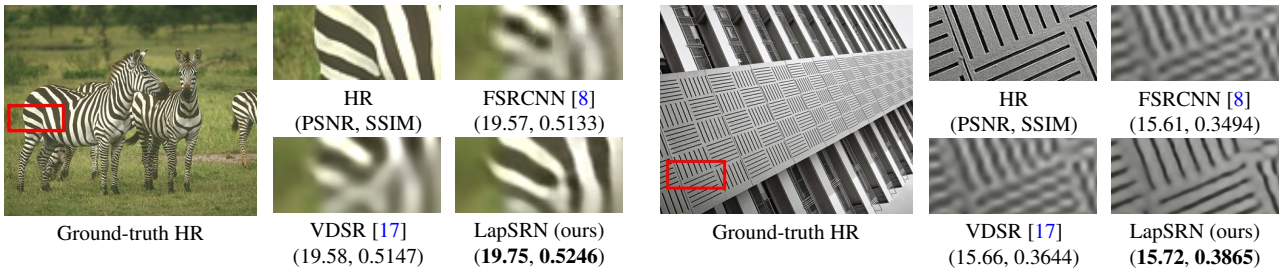


Figure 5: Visual comparison for 8× SR on BSDS100 and URBAN100.

SET5, SET14 and BSDS100 consist of natural scenes; URBAN100 contains challenging urban scenes images with details in different frequency bands; and MANGA109 is a dataset of Japanese manga. We train the LapSRN until the learning rate decreases to $1e-6$ and the training time is around three days on a Titan X GPU.

We evaluate the SR images with three commonly used image quality metrics: PSNR, SSIM [32], and IFC [27]. Table 4 shows quantitative comparisons for 2×, 4× and 8× SR. Our LapSRN performs favorably against existing methods on most datasets. In particular, our algorithm achieves higher IFC values, which has been shown to be correlated well with human perception of image super-resolution [35]. We note that the best results can be achieved by training with specific scale factors (Ours 2× and Ours 4×). As the intermediate convolutional layers are trained to minimize the prediction errors for both the corresponding level and higher levels, the intermediate predictions of our 8× model are slightly inferior to our 2× and 4× models. Nevertheless, our 8× model provides a competitive performance to the state-of-the-art methods in 2× and 4× SR.

In Figure 4, we show visual comparisons on URBAN100, BSDS100 and MANGA109 with the a scale factor of 4×. Our method accurately reconstructs parallel straight lines and grid patterns such as windows and the stripes on tigers. We observe that methods using the bicubic upsampling for pre-processing generate results with noticeable artifacts [7, 17, 26, 30, 33]. In contrast, our approach effectively suppresses such artifacts through progressive reconstruction and the robust loss function.

Table 4: Quantitative evaluation of state-of-the-art SR algorithms: average PSNR/SSIM/IFC for scale factors 2×, 4× and 8×. **Red** text indicates the best and blue text indicates the second best performance.

Algorithm	Scale	SET5 PSNR / SSIM / IFC	SET14 PSNR / SSIM / IFC	BSDS100 PSNR / SSIM / IFC	URBAN100 PSNR / SSIM / IFC	MANGA109 PSNR / SSIM / IFC
Bicubic	2	33.65 / 0.930 / 6.166	30.34 / 0.870 / 6.126	29.56 / 0.844 / 5.695	26.88 / 0.841 / 6.319	30.84 / 0.935 / 6.214
A+ [30]	2	36.54 / 0.954 / 8.715	32.40 / 0.906 / 8.201	31.22 / 0.887 / 7.464	29.23 / 0.894 / 8.440	35.33 / 0.967 / 8.906
SRCNN [7]	2	36.65 / 0.954 / 8.165	32.29 / 0.903 / 7.829	31.36 / 0.888 / 7.242	29.52 / 0.895 / 8.092	35.72 / 0.968 / 8.471
FSRCNN [8]	2	36.99 / 0.955 / 8.200	32.73 / 0.909 / 7.843	31.51 / 0.891 / 7.180	29.87 / 0.901 / 8.131	36.62 / 0.971 / 8.587
SelfExSR [15]	2	36.49 / 0.954 / 8.391	32.44 / 0.906 / 8.014	31.18 / 0.886 / 7.239	29.54 / 0.897 / 8.414	35.78 / 0.968 / 8.721
RFL [26]	2	36.55 / 0.954 / 8.006	32.36 / 0.905 / 7.684	31.16 / 0.885 / 6.930	29.13 / 0.891 / 7.840	35.08 / 0.966 / 8.921
SCN [33]	2	36.52 / 0.953 / 7.358	32.42 / 0.904 / 7.085	31.24 / 0.884 / 6.500	29.50 / 0.896 / 7.324	35.47 / 0.966 / 7.601
VDSR [17]	2	37.53 / 0.958 / 8.190	32.97 / **0.913** / 7.878	**31.90** / **0.896** / 7.169	**30.77** / **0.914** / 8.270	37.16 / 0.974 / 9.120
DRCN [18]	2	**37.63** / 0.959 / 8.326	32.98 / 0.913 / 8.025	31.85 / 0.894 / 7.220	30.76 / 0.913 / 8.527	**37.57** / 0.973 / **9.541**
LapSRN (ours 2×)	2	37.52 / **0.959** / **9.010**	**33.08** / 0.913 / **8.505**	31.80 / 0.895 / **7.715**	30.41 / 0.910 / **8.907**	37.27 / **0.974** / 9.481
LapSRN (ours 8×)	2	37.25 / 0.957 / 8.527	32.96 / 0.910 / 8.140	31.68 / 0.892 / 7.430	30.25 / 0.907 / 8.564	36.73 / 0.972 / 8.933
Bicubic	4	28.42 / 0.810 / 2.337	26.10 / 0.704 / 2.246	25.96 / 0.669 / 1.993	23.15 / 0.659 / 2.386	24.92 / 0.789 / 2.289
A+ [30]	4	30.30 / 0.859 / 3.260	27.43 / 0.752 / 2.961	26.82 / 0.710 / 2.564	24.34 / 0.720 / 3.218	27.02 / 0.850 / 3.177
SRCNN [7]	4	30.49 / 0.862 / 2.997	27.61 / 0.754 / 2.767	26.91 / 0.712 / 2.412	24.53 / 0.724 / 2.992	27.66 / 0.858 / 3.045
FSRCNN [8]	4	30.71 / 0.865 / 2.994	27.70 / 0.756 / 2.723	26.97 / 0.714 / 2.370	24.61 / 0.727 / 2.916	27.89 / 0.859 / 2.950
SelfExSR [15]	4	30.33 / 0.861 / 3.249	27.54 / 0.756 / 2.952	26.84 / 0.712 / 2.512	24.82 / 0.740 / 3.381	27.82 / 0.865 / 3.358
RFL [26]	4	30.15 / 0.853 / 3.135	27.33 / 0.748 / 2.853	26.75 / 0.707 / 2.455	24.20 / 0.711 / 3.000	26.80 / 0.840 / 3.055
SCN [33]	4	30.39 / 0.862 / 2.911	27.48 / 0.751 / 2.651	26.87 / 0.710 / 2.309	24.52 / 0.725 / 2.861	27.39 / 0.856 / 2.889
VDSR [17]	4	31.35 / 0.882 / 3.496	28.03 / 0.770 / 3.071	27.29 / 0.726 / 2.627	25.18 / 0.753 / 3.405	28.82 / 0.886 / 3.664
DRCN [18]	4	31.53 / 0.884 / 3.502	28.04 / 0.770 / 3.066	27.24 / 0.724 / 2.587	25.14 / 0.752 / 3.412	28.97 / 0.886 / 3.674
LapSRN (ours 4×)	4	**31.54** / **0.885** / **3.559**	**28.19** / **0.772** / **3.147**	**27.32** / **0.728** / **2.677**	**25.21** / **0.756** / **3.530**	**29.09** / **0.890** / **3.729**
LapSRN (ours 8×)	4	31.33 / 0.881 / 3.491	28.06 / 0.768 / 3.100	27.22 / 0.724 / 2.660	25.02 / 0.747 / 3.426	28.68 / 0.882 / 3.595
Bicubic	8	24.39 / 0.657 / 0.836	23.19 / 0.568 / 0.784	23.67 / 0.547 / 0.646	20.74 / 0.515 / 0.858	21.47 / 0.649 / 0.810
A+ [30]	8	25.52 / 0.692 / 1.077	23.98 / 0.597 / 0.983	24.20 / 0.568 / 0.797	21.37 / 0.545 / 1.092	22.39 / 0.680 / 1.056
SRCNN [7]	8	25.33 / 0.689 / 0.938	23.85 / 0.593 / 0.865	24.13 / 0.565 / 0.705	21.29 / 0.543 / 0.947	22.37 / 0.682 / 0.940
FSRCNN [8]	8	25.41 / 0.682 / 0.989	23.93 / 0.592 / 0.928	24.21 / 0.567 / 0.772	21.32 / 0.537 / 0.986	22.39 / 0.672 / 0.977
SelfExSR [15]	8	25.52 / 0.704 / 1.131	24.02 / 0.603 / 1.001	24.18 / 0.568 / 0.774	21.81 / 0.576 / 1.283	22.99 / 0.718 / 1.244
RFL [26]	8	25.36 / 0.677 / 0.985	23.88 / 0.588 / 0.910	24.13 / 0.562 / 0.741	21.27 / 0.535 / 0.978	22.27 / 0.668 / 0.968
SCN [33]	8	25.59 / 0.705 / 1.063	24.11 / 0.605 / 0.967	24.30 / 0.573 / 0.777	21.52 / 0.559 / 1.074	22.68 / 0.700 / 1.073
VDSR [17]	8	25.72 / 0.711 / 1.123	24.21 / 0.609 / 1.016	24.37 / 0.576 / 0.816	21.54 / 0.560 / 1.119	22.83 / 0.707 / 1.138
LapSRN (ours 8×)	8	**26.14** / **0.738** / **1.302**	**24.44** / **0.623** / **1.134**	**24.54** / **0.586** / **0.893**	**21.81** / **0.581** / **1.288**	**23.39** / **0.735** / **1.352**

For 8× SR, we re-train the model of A+, SRCNN, FSRCNN, RFL and VDSR using the publicly available code[1]. Both SelfExSR and SCN methods can handle different scale factors using progressive reconstruction. We show 8× SR results on BSDS100 and URBAN100 in Figure 5. For 8× SR, it is challenging to predict HR images from bicubic-upsampled images [7, 17, 30] or using one-step upsampling [8]. The state-of-the-art methods do not super-resolve the fine structures well. In contrast, the LapSRN reconstructs high-quality HR images at a relatively fast speed. We present SR images generated by all the evaluated methods in the supplementary material.

4.3. Execution time

We use the original codes of state-of-the-art methods to evaluate the runtime on the same machine with 3.4 GHz Intel i7 CPU (64G RAM) and NVIDIA Titan X GPU (12G Memory). Since the codes of SRCNN and FSRCNN for testing are based on CPU implementations, we reconstruct these models in MatConvNet with the same network weights to measure the run time on GPU. Figure 6 shows the trade-offs between the run time and performance (in terms of PSNR) on SET14 for 4× SR. The speed of the proposed LapSRN is faster than all the existing methods except FSRCNN. We present detailed evaluations on run time of all evaluated datasets in the supplementary material.

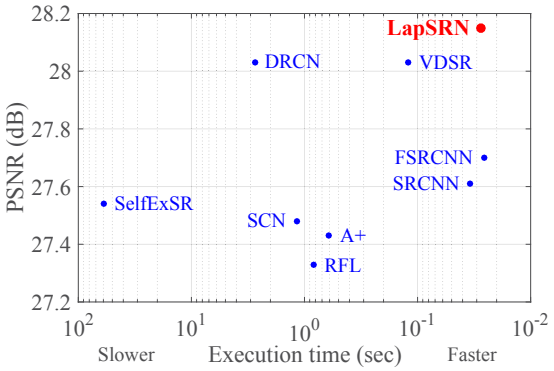


Figure 6: Speed and accuracy trade-off. The results are evaluated on SET14 with the scale factor 4×. The LapSRN generates SR images efficiently and accurately.

[1] We do not re-train DRCN because the training code is not available.

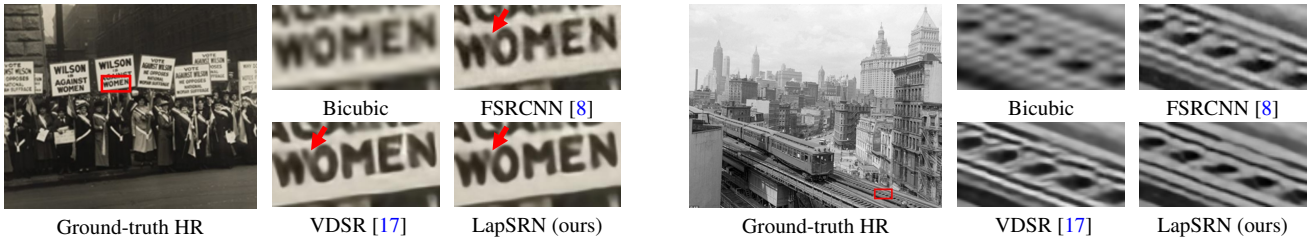


Figure 7: Comparison of real-world photos for 4× SR. We note that the ground truth HR images and the blur kernels are not available in these cases. On the left image, our method super-resolves the letter "W" accurately while VDSR incorrectly connects the stroke with the letter "O". On the right image, our method reconstructs the rails without the ringing artifacts.



Figure 8: Visual comparison on a video frame with a spatial resolution of 1200 × 800 for 8× SR. Our method provides more clean and sharper results than existing methods.

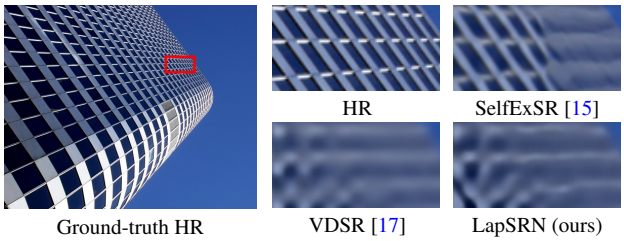


Figure 9: A failure case for 8× SR. Our method is not able to hallucinate details if the LR input image does not consist of sufficient amount of structure.

4.4. Super-resolving real-world photos

We demonstrate an application of super-resolving historical photographs with JPEG compression artifacts. In these cases, neither the ground-truth images nor the downsampling kernels are available. As shown in Figure 7, our method can reconstruct sharper and more accurate images than the state-of-the-art approaches.

4.5. Super-resolving video sequences

We conduct frame-based SR experiments on two video sequences from [22] with a spatial resolution of 1200 × 800 pixels.[2] We downsample each frame by 8×, and then apply super-resolution frame by frame for 2×, 4× and 8×, respectively. The computational cost depends on the size of *input* images since we extract features from the LR space. On the contrary, the speed of SRCNN and VDSR is limited by the size of *output* images. Both FSRCNN and our approach achieve real-time performance (i.e., over 30 frames per second) on all upsampling scales. In contrast, the FPS is 8.43 for SRCNN and 1.98 for VDSR on 8× SR. Figure 8 visualizes results of 8× SR on one representative frame.

4.6. Limitations

While our model is capable of generating clean and sharp HR images on a large scale factor, e.g., 8×, it does not "hallucinate" fine details. As shown in Figure 9, the top of the building is significantly blurred in the 8× downscaled LR image. All SR algorithms fail to recover the fine structure except SelfExSR [15], which explicitly detects the 3D scene geometry and uses self-similarity to hallucinate the regular structure. This is a common limitation shared by parametric SR methods [7, 8, 17, 18]. Another limitation of the proposed network is the relative large model size. To reduce the number of parameters, one can replace the deep convolutional layers at each level with recursive layers.

[2]Our method is *not* a video super-resolution algorithm as temporal coherence or motion blur are not considered.

5. Conclusions

In this work, we propose a deep convolutional network within a Laplacian pyramid framework for fast and accurate single-image super-resolution. Our model progressively predicts high-frequency residuals in a coarse-to-fine manner. By replacing the pre-defined bicubic interpolation with the learned transposed convolutional layers and optimizing the network with a robust loss function, the proposed LapSRN alleviates issues with undesired artifacts and reduces the computational complexity. Extensive evaluations on benchmark datasets demonstrate that the proposed model performs favorably against the state-of-the-art SR algorithms in terms of visual quality and run time.

Acknowledgments

This work is supported in part by the NSF CAREER Grant #1149783, gifts from Adobe and Nvidia. J.-B. Huang and N. Ahuja are supported in part by Office of Naval Research under Grant N00014-16-1-2314.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 33(5):898–916, 2011. 4, 5

[2] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *BMVC*, 2012. 2, 5

[3] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. *IJCV*, 61(3):211–231, 2005. 4

[4] P. J. Burt and E. H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, 1983. 3

[5] H. Chang, D.-Y. Yeung, and Y. Xiong. Super-resolution through neighbor embedding. In *CVPR*, 2004. 2

[6] E. L. Denton, S. Chintala, and R. Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015. 3

[7] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *TPAMI*, 38(2):295–307, 2015. 1, 2, 3, 5, 6, 7, 8

[8] C. Dong, C. C. Loy, and X. Tang. Accelerating the super-resolution convolutional neural network. In *ECCV*, 2016. 1, 2, 3, 5, 6, 7, 8

[9] G. Freedman and R. Fattal. Image and video upscaling from local self-examples. *ACM TOG (Proc. of SIGGRAPH)*, 30(2):12, 2011. 2

[10] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE, Computer Graphics and Applications*, 22(2):56–65, 2002. 2

[11] G. Ghiasi and C. C. Fowlkes. Laplacian pyramid reconstruction and refinement for semantic segmentation. In *ECCV*, 2016. 3

[12] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *ICCV*, 2009. 2

[13] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 4

[14] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *SIGGRAPH*, 1995. 3

[15] J.-B. Huang, A. Singh, and N. Ahuja. Single image super-resolution from transformed self-exemplars. In *CVPR*, 2015. 2, 5, 6, 7, 8

[16] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 3

[17] J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, 2016. 1, 2, 3, 4, 5, 6, 7, 8

[18] J. Kim, J. K. Lee, and K. M. Lee. Deeply-recursive convolutional network for image super-resolution. In *CVPR*, 2016. 2, 3, 5, 6, 7, 8

[19] K. I. Kim and Y. Kwon. Single-image super-resolution using sparse regression and natural image prior. *TPAMI*, 32(6):1127–1133, 2010. 2

[20] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 3

[21] C. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets, 2015. In *International Conference on Artificial Intelligence and Statistics*, 2015. 4

[22] R. Liao, X. Tao, R. Li, Z. Ma, and J. Jia. Video super-resolution via deep draft-ensemble learning. In *ICCV*, 2015. 8

[23] Y. Matsui, K. Ito, Y. Aramaki, T. Yamasaki, and K. Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, pages 1–28, 2016. 5

[24] S. Paris, S. W. Hasinoff, and J. Kautz. Local laplacian filters: Edge-aware image processing with a laplacian pyramid. *ACM TOG (Proc. of SIGGRAPH)*, 30(4):68, 2011. 3

[25] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *ECCV*, 2016. 3

[26] S. Schulter, C. Leistner, and H. Bischof. Fast and accurate image upscaling with super-resolution forests. In *CVPR*, 2015. 1, 2, 4, 5, 6, 7

[27] H. R. Sheikh, A. C. Bovik, and G. De Veciana. An information fidelity criterion for image quality assessment using natural scene statistics. *TIP*, 14(12):2117–2128, 2005. 6

[28] W. Shi, J. Caballero, F. Huszar, J. Totz, A. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016. 1, 3

[29] A. Singh and N. Ahuja. Super-resolution using sub-band self-similarity. In *ACCV*, 2014. 2

[30] R. Timofte, V. De Smet, and L. Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *ACCV*, 2014. 1, 2, 5, 6, 7

[31] A. Vedaldi and K. Lenc. MatConvNet: Convolutional neural networks for matlab. In *ACM MM*, 2015. 5

[32] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *TIP*, 13(4):600–612, 2004. 6

[33] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang. Deep networks for image super-resolution with sparse prior. In *ICCV*, 2015. 1, 2, 3, 5, 6, 7

[34] S. Xie and Z. Tu. Holistically-nested edge detection. In *CVPR*, 2015. 4

[35] C.-Y. Yang, C. Ma, and M.-H. Yang. Single-image super-resolution: a benchmark. In *ECCV*. 2014. 6

[36] C.-Y. Yang and M.-H. Yang. Fast direct super-resolution by simple functions. In *ICCV*, 2013. 1, 2

[37] J. Yang, J. Wright, T. Huang, and Y. Ma. Image super-resolution as sparse representation of raw image patches. In *CVPR*, 2008. 1, 2

[38] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image super-resolution via sparse representation. *TIP*, 19(11):2861–2873, 2010. 1, 2, 4

[39] R. Zeyde, M. Elad, and M. Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces*. 2010. 2, 5