Invertible CNN-Based Super Resolution with Downsampling Awareness

Andrew Geiss and Joseph C. Hardin
Pacific Northwest National Laboratory
Richland WA, USA
andrew.geiss@pnnl.gov, joseph.hardin@pnnl.gov

Abstract

Single image super resolution involves artificially increasing the resolution of an image. Recently, convolutional neural networks have been demonstrated as very powerful tools for this problem. These networks are typically trained by artificially degrading high resolution images and training the neural network to reproduce the original. Because these neural networks are learning an inverse function for an image downsampling scheme, their high-resolution outputs should ideally re-produce the corresponding low-resolution input when the same downsampling scheme is applied. This constraint has not historically been explicitly and strictly imposed during training however. Here, a method for "downsampling aware" super resolution networks is proposed. A differentiable operator is applied as the final output layer of the neural network that forces the downsampled output to match the low resolution input data under 2D-average downsampling. It is demonstrated that appending this operator to a selection of state-of-the-art deep-learning-based super resolution schemes improves training time and overall performance on most of the common image super resolution benchmark datasets. In addition to this performance improvement for images, this method has potentially broad and significant impacts in the physical sciences. This scheme can be applied to data produced by medical scans, precipitation radars, gridded numerical simulations, satellite imagers, and many other sources. In such applications, the proposed method's guarantee of strict adherence to physical conservation laws is of critical importance.

1. Introduction

Single Image Super-Resolution (SISR) involves increasing the resolution of a single image beyond its native resolution and is a long-standing problem in the field of image processing. While several Super-Resolution (SR) frameworks exist that utilize additional information that can assist with the SR task, for instance multi-view imaging [32, 36] or video [18, 4], SISR schemes estimate sub-pixel scale values based only on a single coarsely resolved input image. The simplest approach to SISR is 2D-interpolation, and many interpolation schemes exist that produce High Resolution (HR) outputs of various qualities. These schemes apply the same operation everywhere in an image, and more sophisticated SISR schemes exist that produce high fidelity images by performing different operations at different locations depending on information contained in the local Low Resolution (LR) pixel data. [37] operates on image patches, and uses a dictionary of exemplars to infer likely sub-pixel scale features based on the local LR texture information for instance. [29] provides an overview of various SISR schemes.

1.1. Super Resolution with Neural Networks

Recently, deep Convolutional Neural Networks (CNNs) have become a popular tool for SISR and have been demonstrated to significantly outperform past algorithms under a variety of image quality metrics. Initially, a small (by current standards) 3-layer CNN was used to achieve state of the art SISR results [8], and the approach was quickly expanded to use significantly deeper CNNs [20]. The architecture of SISR CNNs has rapidly developed alongside research on CNNs designed for image classification and semantic segmentation. SISR networks are now built up of blocks of convolutional layers that include skip connections such as dense blocks [16], residual blocks [15], and channel attention blocks [6]. There has been a progression in the internal upsampling techniques used in these CNNs, from applying bicubic upsampling prior to processing [8], to the use of learned upsampling kernels [25], to the "pixel-shuffle" approach [34]. There has also been progression in the type of loss function used. While many approaches initially used pixel-wise errors, later work [22] used feature loss, based on the image feature representations learned by an image classification CNN, and adversarial loss, which allows the CNN to hallucinate plausible sub-pixel scale features. The current state of the art CNN SR schemes incorporate many of these concepts [3]. Seven recent CNN SR architectures

have been implemented for this study (Section 3.1). See [41] for a review of CNN-based SISR.

1.2. Invertible and Encoder/Decoder Networks

Deep CNNs have been a boon for SISR, and can produce sharp and visually appealing HR images. The way that they are typically implemented and trained means that they are not invertible however: there is no known mapping from their HR output back to the LR input. Usually training involves applying a degradation scheme to HR images and then asking the CNN to reconstruct them. Applying the same degradation scheme to the CNN output does not necessarily map back to the input LR image however, and while the outputs may be visually appealing, we know that the CNN has not produced the true HR image. Therefore, constraining the neural network output to the space of HR images that when degraded produce the input LR image may be desirable.

Several authors have done work related to inverting super resolution CNNs. In cases where the HR image is known but needs to be intentionally degraded (image compression for example) training two neural networks simultaneously, with one learning a downsampling operation and the other learning to invert it, achieves a significant performance improvement over other SR schemes [35]. This is not applicable when the HR image is unknown however: when increasing the resolution of an image beyond the capabilities of the sensor for instance. There are also methods to train CNNs for image classification where the operation learned by the CNN is invertible [19], though this is not directly applicable to SISR.

Most relevant to this study is: [28], who acknowledge that while Generative Adversarial Network (GAN) based SISR can produce very plausible HR outputs, the outputs do not necessarily reproduce the input when downsampled. To combat this, they add a term to the loss function that penalizes the CNN for producing outputs that do not downsample to the original input. [1] and [39] propose similar ideas. While this approach does not strictly force the HR outputs to downsample to the LR inputs, it does place a strong constraint on the space of possible output images for GANs. We discuss this method further in Section 4.3.

1.3. Contributions and Impacts

In this study, we introduce a method, referred to as "Downsampling Aware" (DA) super resolution, that strictly constrains a super resolution CNN's output to be *exactly* invertible under a known downsampling scheme. The DA SR method involves modification of the CNN rather than the loss function. A differentiable operator is derived that acts as the final layer of the CNN. This operator enforces that when 2D-average downsampling is applied to the CNN's HR output, the LR input is exactly reproduced. We demonstrate this method by applying it to seven different common CNN SISR architectures and find that it improves performance in every case, though only when the assumption of 2D-averaging as the downsampling operator is correct. In addition to this performance improvement in SISR for images from handheld cameras, our method has significant and broad implications for super-resolution in the physical sciences, where exact invertibility under 2D-averaging can be used to enforce physical conservation laws.

1.4. Applications in the Physical Sciences

There are many scientific fields where gridded data are used and CNN-based SR can be applied. In these applications, guaranteed physical consistency under 2D-averaging can ensure the SR scheme obeys physical conservation laws. This paper is focused on describing and benchmarking the proposed algorithm, but some possible applications in the physical sciences are described here.

Pixel intensities observed by most digital imagers are proportional to an average of subpixel spectral radiances. For most photos, breaking this property with a SISR scheme does not impact the visual fidelity of the result. This is a critical problem when super-resolving data from scientific imagers however. Sensors on Earth-observing satellites undergo rigorous calibration and validation to ensure that measured radiances are accurate for instance [26]. These radiances are then used to estimate physical properties such as land use and ocean and atmospheric conditions [31]. For a satellite imager, our algorithm can ensure that the SR process does not introduce non-physical radiances in the HR image, and some of these physics-based retrievals will still be applicable. Some imagers will require a weighted average because of their construction [30], but our algorithm should be easily modifiable in these cases.

Other possible applications of this method include data from ranging instruments such as radars [12], sonars [9], and lidars [24]. Weather radars can be used to estimate precipitation rates for instance [11], a physical quantity that should be conserved under spatial averaging. CNN-based SR schemes have been heavily used for medical imaging devices such as X-rays, CT-scanners, and MRIs [13], where strict conservation under 2D-averaging may be desirable. Finally, CNNs can be used to increase the resolution of output from gridded numerical simulations, like those from weather or computational fluid dynamics models [5] (this is often referred to as "downscaling"). In these simulations, physical conservation laws such as conservation of mass, momentum, or energy, are often enforced within the model, and ideally the downscaling scheme should not break these constraints. This is of particular importance if the SR scheme is applied during model integration, when even small errors can grow rapidly and cause instability in the model. In these applications, and many others, strict

conservation of large-scale statistical properties is often just as important as the visual fidelity of the HR output, and our method can ensure both.

2. Method

The training paradigm typically used for CNN-based SISR schemes involves artificial degradation of a high resolution image. The CNN is provided LR images as inputs and tasked to produce the original HR images. That is: if I_{HR} and I_{LR} are the high- and low-resolution images respectively, $D\{*\}$ is the image downsampling operator and $SR\{*\}$ is the super resolution scheme, typical CNN-based SR schemes try to find $SR\{*\}$ such that: $I_{HR} \approx SR\{I_{LR}\}$, and during training: $I_{HR} \approx SR\{D\{I_{HR}\}\}$. Because the super-resolution operator is meant to invert the downsampling operator, it seems reasonable to desire the SR operator to also be invertible by the downsampling operator: $I_{LR} = D\{SR\{I_{LR}\}\}$. Past CNN-based SR schemes do not enforce this relation however, and here we derive a Downsampling Aware (DA) operator that can be incorporated into to most common SR CNN architectures that strictly enforces this relation.

2.1. Derivation of Downsampling Aware Operator

Here we use the notation that C_P is an $N \times N$ pixel block in I_{HR}, where $1/N$ is the downsampling ratio and the shorthand: $\sum_{x_i \in C_P}$ is a 2D-summation over all HR pixels in the block. P will represent the values of the single pixel in the low resolution input image corresponding to C_P in the output. x_i will represent the value of a pixel in the high-resolution image that is output by the second-to-last layer of the neural network, and $f(x_i)$ is the pixel value after a correction is applied (the last layer of the CNN). In this study, we assume 2D-average downsampling, which allows for a simpler formulation of f, the downsampling aware operator. P and x_i are assumed to have pixel intensities bounded by -1 and 1. We would like $f(x_i)$ to have several properties:

$$\frac{1}{N^2} \sum_{x_i \in C_P} f(x_i) = P \tag{1}$$

$$f(x_i) \in [-1, 1] \tag{2}$$

$$x_i > x_j \rightarrow f(x_i) \geq f(x_j) \tag{3}$$

$$f(x_i) \text{ is piecewise differentiable} \tag{4}$$

(1) ensures that the SR scheme is invertible under a 2D-average downsampling scheme. (2) enforces that the values output by f are within the input image's dynamic range of $[-1, 1]$. (3) requires that the order of the output pixels' intensity is maintained, *e.g.* the brightest pixel remains the brightest and the darkest remains the darkest. Finally, (4): $f(x_i)$ will be included as a part of the CNN during training and must be differentiable for backpropagation to work.

Short proofs that $f(x_i)$, defined below, satisfies these conditions are given in Appendix Section 2. We start by considering the case that:

$$\bar{x} < P \qquad \text{where:} \quad \bar{x} = \frac{1}{N^2} \sum_{x_i \in C_P} x_i \tag{5}$$

Here, the average of the pixel intensities in C_P in the HR output is less than the intensity of the corresponding input pixel P and the HR pixel intensities need to be adjusted upwards. $x_i \in [-1, 1]$ and $f(x_i)$ can be formulated:

$$f(x_i) = x_i + \alpha(1 - x_i), \qquad \alpha \in [0, 1] \tag{6}$$

Here, α is a parameter that can be solved for as a function of P and $\bar{x}$. It can be seen that when $\alpha = 1, f(x_i) = 1$, and when $\alpha = 0, f(x_i) = x_i$. α can be found by enforcing (1):

$$\alpha = (P - \bar{x})/(1 - \bar{x}) \tag{7}$$

In the case that $\bar{x} > P$:

$$f(x_i) = x_i - \alpha(1 + x_i), \qquad \alpha \in [0, 1] \tag{8}$$

$$\alpha = (\bar{x} - P)/(1 + \bar{x}) \tag{9}$$

This adjusts the HR pixel intensities downwards so that when $\alpha = 1$, $f(x_i) = -1$, and when $\alpha = 0$, $f(x_i) = x_i$. $f(x_i)$ can then be defined as a piecewise function by:

$$f(x_i) = \begin{cases} x_i + \left(\frac{P-\bar{x}}{1-\bar{x}}\right)(1 - x_i) & \bar{x} < P \\ x_i & \bar{x} = P \\ x_i + \left(\frac{P-\bar{x}}{1+\bar{x}}\right)(1 + x_i) & \bar{x} > P \end{cases} \tag{10}$$

And finally, condensed to a single line:

$$f(x_i) = x_i + (P - \bar{x})\left(\frac{\sigma + x_i}{\sigma + \bar{x}}\right), \ \sigma = \text{sign}(\bar{x} - P) \tag{11}$$

2.2. Interpretation of $f(x_i)$ and α

The DA operator in Equation (11) has an intuitive physical interpretation: it operates on CNN outputs that can be interpreted as images (the last layer of the CNN prior to applying (11) is assumed to have 3-channel RGB output and have applied a $tanh$ transfer function that bounds the output to $[-1, 1]$). (11) is a correction to this output image that is applied independently to each channel. (11) assumes that each $N \times N$ block of pixels (C_P) in the output corresponds to one input pixel P, and it strictly enforces that the average intensity over this block should match the intensity of P. When the input pixel intensity (P) exceeds the average over the corresponding HR output pixels (x_i), the remaining unused output pixel intensity $(1 - x_i)$ is computed for each output pixel and a constant fraction of it, α, is added to the output pixel values. A similar approach is applied

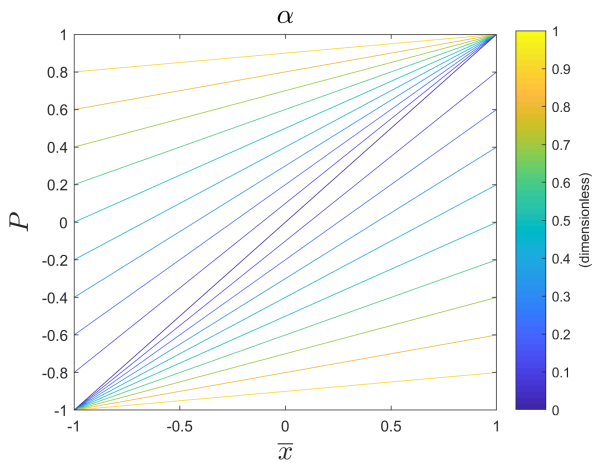

Figure 1. Visualization of the parameter α as a function of P, the input low resolution pixel intensity, and $\bar{x}$, the 2D-average of the corresponding pixels in the high-resolution output image before correction.

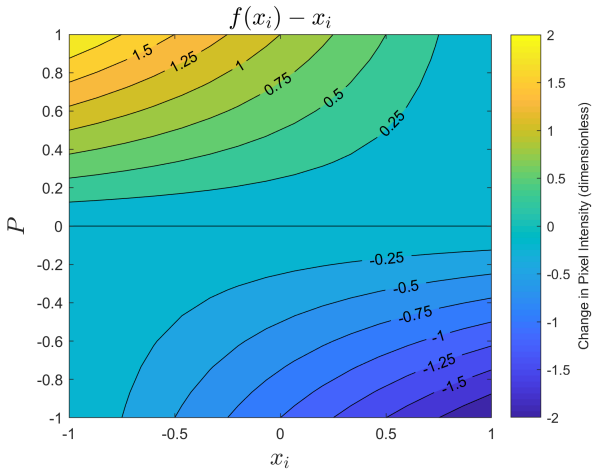

Figure 2. Visualization of the correction $f(x_i) - x_i$ as a function of varying P. A sample input of 16 x_i's ranging from -1 to 1 with a mean of 0 is used.

when $\bar{x} > P$. The magnitude of the correction and importance of α in fully trained CNNs is analyzed in more detail in Section 4.4. Figure 1 shows a visualization of how α depends on P and $\bar{x}$. It shows that $\alpha = 0$ along the one-to-one line where $\bar{x} = P$ and no correction is needed and it increases towards 1 near the top and bottom of the plot where $P = \pm 1$ and all of the unused (used) pixel intensity range must be added (subtracted) from the output. Figure 2 shows the magnitude of the correction $(f(x_i) - x_i)$ when the DA operator is applied to a hypothetical block of output pixels for a range of LR input pixel values (P). The example output pixel intensities (x_i) are evenly distributed between -1 and 1 and have a mean of 0. This plot demonstrates that the correction term varies smoothly with respect to P and x_i, a desirable feature because gradients will need to be backpropagated through this layer during training. It also demonstrates that $f(x_i)$ applies the largest corrections to the pixels with intensity farthest from P however, which may not be optimal from an image quality standpoint.

3. Experimental Design

To evaluate the "downsampling aware" approach to SISR, we have implemented a selection of SISR-CNNs from the recent literature and trained them under identical conditions both with and without the DA operator. The CNN's and training/evaluation procedure are described here and results are discussed in Section 4.

3.1. Neural Networks

Here, we have reproduced seven different CNN architectures from the recent SISR literature. For unbiased comparison, we have altered each model slightly so they all have a similar number of trainable parameters: 5×10^6 (except in the case of SRCNN and LapSRN which have fewer). Aside from this, we have kept their implementations as true to their description in the corresponding papers as possible. All of these CNNs were implemented in Keras with a Tensorflow backend. We provide a more detailed overview of the CNNs and our implementations in the Appendix Section 1, and the exact implementation can be found on github[1], along with diagrams from Keras's "plot_model" function. The CNNs are:

SR-CNN [8]: specifically the "9-5-5" version of this model. This a relatively simple CNN, but is the first application of a CNN to SISR.

Lap-SRN [21]: Is a design based on Laplacian pyramid upsampling. The CNN consists of a feature extraction branch that infers HR features from the LR image, and a image reconstruction branch, that generates images at the intermediate resolutions between the LR input and HR output based on these features.

Dense U-Net (DUN): Is a U-net style architecture [33] where each convolution block has been replaced with a "dense" block [16] (this CNN has previously been used for applying super-resolution to weather radar data [12]).

Deep Back Projection Network (DBPN) [14]: Is composed of blocks of convolutional layers that repeatedly upsample and downsample the input image between the input and target resolutions using transposed and strided convolutions (respectively).

[1] https://github.com/avgeiss/invertible_sr

Dense SR Net (DSRN) [38]: This CNN directly extends concepts from [16] to super resolution, and consists of a series of densely connected blocks of convolutional layers. The blocks operate at low spatial resolution and build up a feature representation of the image which is fed to an upsampling module composed of two transposed convolutions at the end of the network.

Enhanced Deep Residual Network (EDRN) [23]: This CNN is based on ideas from [15]. It is composed of residual blocks, which pass their inputs through several convolutional layers and then add the result to the original input. "Enhanced" refers to removing several aspects of the residual blocks that were used in [15] that are not beneficial for super resolution.

Residual Dense Network (RDN) [43]: The RDN combines aspects of residual and densely connected networks. It is composed of "dense" blocks [16] that have residual connections [15] between their input and output.

3.2. Training and Testing

Downsampling scheme: This study uses 2D-average downsampling for image degradation. Bicubic downsampling with anti-aliasing is more common in the CNN-SISR literature; specifically the Matlab image resizing scheme, but 2D-averaging was assumed in deriving (11). As a short aside: the Matlab bicubic downsampling scheme is not currently open-source, and using an open-source option for future SR studies would be more conducive to reproducible research. More discussion on image downsampling algorithms is provided in Section 4.2. The CNNs here perform 4x SISR: they take 48x48 pixel inputs and produce 192x192 outputs. Pixel intensities are converted to a range of [-1,1] and a $tanh$ activation is applied to the output. In the DA cases the $tanh$ activation is applied *before* applying (11).

Training Data: The CNNs are trained on the Div2k [2] dataset. This is a collection of 800 high resolution RGB images commonly used for SISR research, with a 100-image test set. The first 790 images are used for training and and the last 10 images are used to compute validation scores, as in [23].

Loss Function and Validation Metrics: Pixel-wise MSE is used as a loss function:

$$\mathcal{L} = \overline{(x - \hat{x})^2} \tag{12}$$

where the overbar is an average over the pixels in the batch and x and $\hat{x}$ are output and ground truth HR pixel values. Peak Signal to Noise Ratio (PSNR) is evaluated throughout training. PSNR is a standard metric for evaluating CNN-based SISR schemes [41], and here is computed in the same way as [8]: by first converting the CNN's output from the RGB to the YCbCr color space and then computing PSNR on the intensity (Y) channel.

Training: For fair inter-comparison, the same training method is used for each CNN even if it does not match the technique used in the original paper. Each network is trained for 300 epochs, with the learning rate reduced by a factor of 10 after the 200th epoch. Epochs are 1000 batches of 16 image chips selected randomly from the training set with random flips and rotations. We use the Adam optimizer with an initial learning rate of 10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-7}$. This approach is similar to [23] and [43]. PSNR is computed on the 10-image validation set after each epoch (Figure 3). During validation and testing, each LR image is broken into 48x48 pixel chips using a 24-pixel stride. This over-samples the image (the chips overlap) and the output HR image is constructed by tiling the 96x96 center portions of each of the HR outputs. PSNR and Structural Similarity Index (SSIM) [40] are calculated for each output image and these values are averaged over the validation/test set.

4. Results and Discussion

4.1. Performance

Figure 3 (a-g) shows the training loss for each CNN with (red lines) and without (black lines) the downsampling aware operator. The DA versions of the CNNs perform comparably or better than the conventional CNNs, with the largest performance advantage early during training. The difference also tends to be the largest for CNNs that have the lowest overall performance. The best validation scores achieved during training are indicated by the two markers on the vertical axis. The downsampling aware operator has a positive impact on the CNN performance and training time regardless of the CNN architecture used.

Example outputs from EDRN both with and without DA are shown in Figure 4, where each row contains a sample image from one of the test datasets. Outputs from the other CNNs can be found in Appendix Section 1. These figures are included as a demonstration that the SISR schemes are able to produce reasonably high quality outputs (the small differences in PSNR shown in Figure 3 are unlikely to relate to noticeable differences in the no-DA and DA cases). Because the CNNs were trained with MSE loss, they struggle to reproduce fine textures, but are particularly good at maintaining sharp edges.

Table 1 summarizes final performance on several standard benchmark datasets. Two metrics are shown: PSNR computed in the same manner as [8], and the Structural

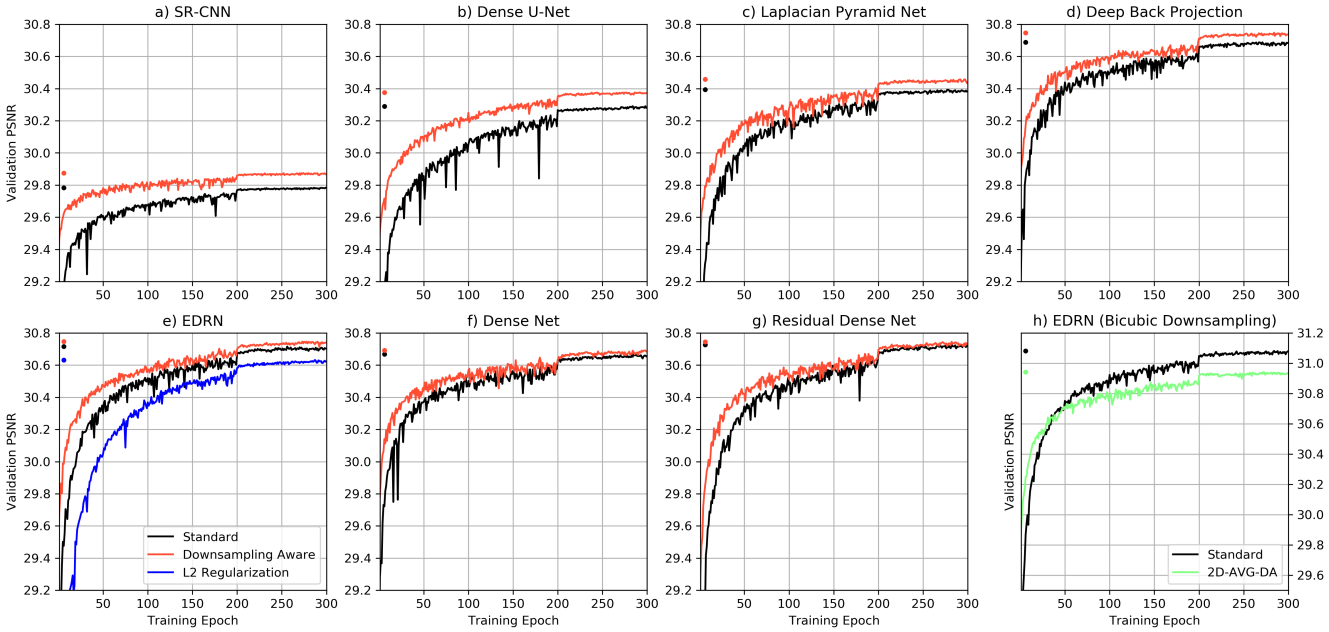

Figure 3. (a-g): Validation set PSNR during training for the seven CNN architectures with (red) and without (black) Downsampling Awareness (DA). The blue line in (e) uses a regularization term instead of DA. (h): Shows how incorrect assumptions about downsampling change results, and applies conventional CNN-SISR (black) and DA SISR that assumes 2D-average downsampling (green) to bicubic-downsampled images.

Similarity Index (SSIM) [40]. SSIM is a metric designed to be more representative of the perceptual quality of an image than pixel-wise metrics. It scales from -1 to 1 and higher values are better. The test datasets are: SET5 [7], SET14 [42], BSDS100 [27], Manga109 [10], Urban 100 [17] and the 100-image Div2k validation set [2]. These are all collections of images frequently used for benchmarking SISR schemes. Manga 109 are illustrated images, Urban 100 contains photographs of urban scenes, and the other datasets are collections of miscellaneous photographs. These datasets are common in the SISR literature, and [41] provide a more detailed overview. Table 1 demonstrates that appending the DA operator to a CNN almost always provides a marginal performance improvement (note that SET5 and SET14 are the smallest test sets and their results are thus more susceptible to statistical variance), while ensuring strict preservation of the statistics of the input data.

4.2. Limitations

Using 2D-average downsampling makes derivation of (11) easier, but it is not always used to train SISR CNNs. To evaluate how DA affects SISR when the downsampling assumption is wrong, we trained the EDRN CNN using Matlab's bicubic downsampling scheme, both with and without DA that assumes 2D-averaging. The training performance is shown in Figure 3h. The conventional EDRN outperforms the DA version except early during training. For the 100-image div2k test set the conventional scheme had a PSNR/SSIM of: 30.26/0.8393 while the DA EDRN scored: 30.13/0.8356. This is unsurprising because the relation between the HR and LR images enforced by the DA operator is not correct in this case. This reduction in skill is small however, and there are use cases where a small reduction in perceptual quality may be of secondary importance to a guarantee of physical consistency.

While 2D-averaging is assumed in this study, similar operators could be derived for other downsampling schemes, and modifying (10,11) to include the weights of a bicubic downsampling kernel seems straightforward. The derivation of (10,11) assumed that each HR pixel influences only one LR pixel. This is not true for all downsampling schemes (anti-aliasing breaks this assumption for instance). Derivation of a DA operator may be much more difficult in these cases. In any case, while 2D-averaging may not be the best assumption for images, it is often the correct assumption for applications to physical data like those discussed in Section 1.4 to ensure that conservation laws are enforced.

Finally, some of the outputs from the DA CNN contain faint checkerboard artifacts; in the trees in the BSDS100 sample image in Figure 4 for instance. This also occurs to some degree for the no-DA cases, so we hypothesize that these patterns are at least partially a result of using 2D-average downsampling without anti-aliasing. In some initial experiments however, we have observed that the problem is

Table 1. Evaluation of several super resolution CNN architectures, both with and without Downsampling Awareness (DA), applied to standard test datasets for image super resolution. Entries show Peak Signal to Noise Ratio / Structural Similarity Index (PSNR/SSIM), with higher scores in bold.

	SET5	SET14	BSDS100	Manga-109	Urban-100	Div2k
SR-CNN	32.20/.8914	27.07/.7468	26.35/.7155	27.43/.8452	24.23/.7247	29.05/.8090
w/ DA	**32.47/.8959**	**27.19/.7506**	**26.41/.7179**	**27.63/.8513**	**24.36/.7316**	**29.14/.8132**
Lap-SRN	33.47/.9075	27.85/.7676	26.88/.7353	29.58/.8825	25.74/.7823	29.99/.8334
w/ DA	**33.50/.9081**	**27.89/.7687**	**26.91/.7366**	**29.70/.8856**	**25.88/.7875**	**30.06/.8352**
DUN	**33.30**/.9040	27.57/.7588	26.66/.7272	28.63/.8668	24.99/.7561	29.56/.8224
w/ DA	33.26/**.9044**	**27.67/.7616**	**26.72/.7289**	**28.86/.8726**	**25.13/.7629**	**29.66/.8256**
DBPN	33.17/.9034	27.63/.7603	26.70/.7287	28.86/.8707	25.14/.7607	29.66/.8247
w/ DA	**33.36/.9055**	**27.69/.7618**	**26.75/.7301**	**28.94/.8725**	**25.31/.7670**	**29.73/.8268**
DSRN	**33.62**/.9081	**27.90/.7678**	26.87/.7350	29.54/.8822	25.72/.7825	29.98/.8332
w/ DA	33.56/**.9086**	27.85/.7673	**26.88/.7360**	**29.63/.8849**	**25.80/.7859**	**30.00/.8344**
EDRN	33.52/.9072	**27.92**/.7677	26.89/.7359	29.60/.8821	25.81/.7853	30.03/.8344
w/ DA	**33.59/.9089**	27.91/**.7690**	**26.91/.7373**	**29.67/.8853**	**25.88/.7883**	**30.08/.8361**
RDN	**33.49**/.9072	27.88/.7669	26.88/.7353	29.60/.8813	25.86/.7859	30.04/.8342
w/ DA	33.49/**.9078**	**27.91/.7689**	**26.92/.7369**	**29.67/.8836**	**25.94/.7896**	**30.06/.8351**

more pronounced when the DA operator is used for larger upsampling ratios (x16 upsampling), and corresponds to the regions that the DA operator is applied to. Preliminary work indicates that an approach in which 2X super-resolution is applied multiple times with the DA operator applied after each resolution increase significantly reduces this pattern. This seems to be a limitation of our algorithm for large resolution increases, but an in-depth exploration of the problem and possible mitigation strategies are left as future research.

4.3. Other Methods

While no existing methods strictly enforce invertibility of CNN SR under a known downsampling scheme, past work [28, 1, 39] has proposed adding a term to the loss function of GANs to help approximate invertibility. [28] compute an MSE term between the LR input image and the downsampled output from their GAN-based SR scheme and add it to the adversarial loss function used by the GAN. This results in more physically plausible outputs. This method is unlikely to improve CNNs that optimizing pixel-wise MSE however, because it effectively imposes the same loss function twice at different resolutions. We confirm this by training the EDRN-CNN with the loss function:

$$\mathcal{L} = \overline{(x - \hat{x})^2} + \lambda \overline{(D\{x\} - D\{\hat{x}\})^2} \tag{13}$$

where $D\{*\}$ is a 4×4 averaging downsampling operator and λ is a weighting coefficient (here, $\lambda = 16$). Validation PSNR throughout training is shown in Figure 3e as a blue line. The PSNR/SSIM computed on the Div2k test set was: 29.95/0.8327. This is lower than the scores for EDRN in Table 1 and the additional loss term decreased performance. It is easy to see why: the loss function already optimizes pixel-wise PSNR which is directly related to MSE and adding terms is then unlikely to improve PSNR. While this approach is not applicable in our case, [28] is one of the few studies that has demonstrated methods to enforce that the HR-outputs from SR-CNNs downsample to the LR-input.

4.4. Importance of α

In Section 2.2, $f(x_i)$ (11) is interpreted as a correction applied to an intermediate output from the CNN that can itself be interpreted as a HR image. During training, does the quality of this intermediate output improve, or does the CNN learn to rely on the correction? The magnitude of the parameter α can be interpreted as the magnitude of the correction (no correction is applied when $\alpha = 0$ and $\alpha = 1$ applies the maximum correction). An equivalent question is: is $\overline{\alpha}$ (the mean of α) near zero after training? We investigate using the EDRN architecture.

Firstly, $\overline{\alpha}$ for the Div2k test set was 0.29, so even for the fully trained network, the intermediate HR image requires a correction to ensure that it will downsample to the input image. Figure 5 panels a and b, show the intermediate output and the corrected output for an example HR image chip from the Div2k test set respectively (the original image is shown in panel e). The intermediate output has noticeably incorrect color and intensity and there is a checkerboard pattern corresponding to the 4x4 pixel regions that the correction is applied to. None of these appear after the correction is applied however. The magnitude of the correction can be reduced by adding a regularization term to the loss function:

$$\mathcal{L} = \overline{(x - \hat{x})^2} + \lambda \overline{\alpha^2} \tag{14}$$

Here, $\overline{(x - \hat{x})^2}$ is the pixel-wise MSE, and $\lambda\overline{\alpha^2}$ is the weighted mean squared value of alpha over each batch (here, $\lambda = 10$). We compute MSE on pixel values between 0-255 and $\alpha \in [0, 1]$, so in practice the contribution

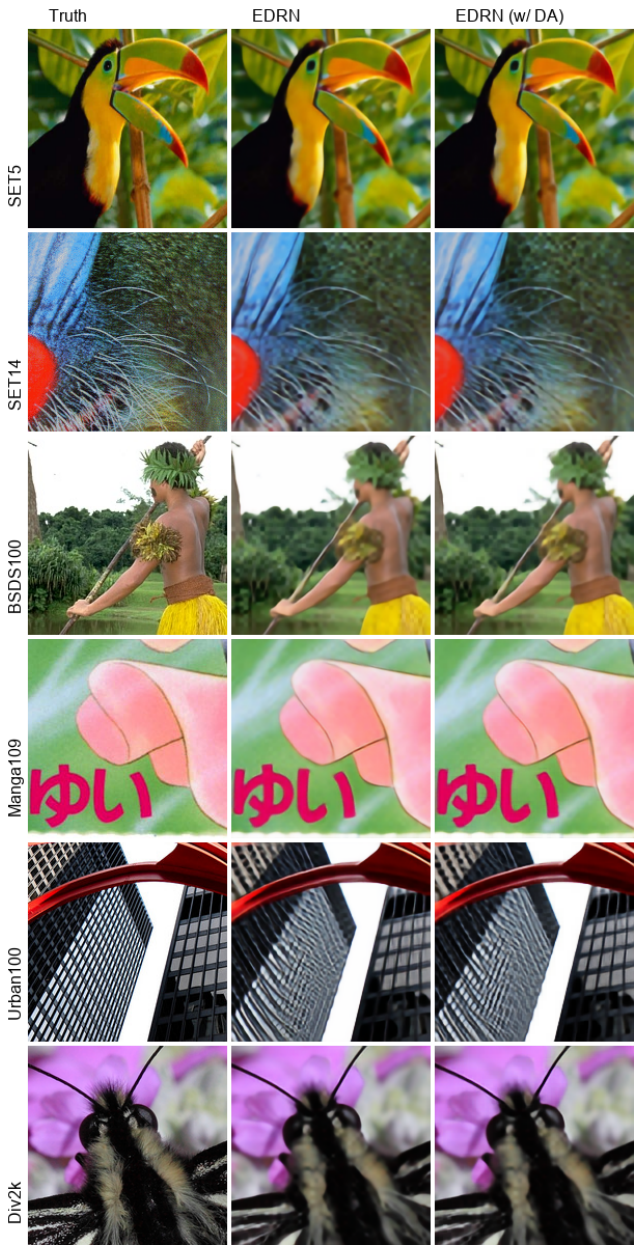

Figure 4. 4x super resolution for sample images from each of the test datasets from our implementation of the Enhanced Deep Residual Network with and without the downsampling awareness.

of α to the loss is much smaller than the MSE component. Nonetheless, after training with this regularization term, $\overline{\alpha}$ on the Div2k test set was 9.8×10^{-8}. This is demonstrated in panels c and d of Figure 1, where the intermediate output (c) is now a near perfect match for the final output (d). Finally, the overall performance of the SISR scheme was not substantially altered and the CNN had a PSNR of 30.06 and a SSIM of 0.8357 on the Div2k test set, only marginally lower than the results without regularization.

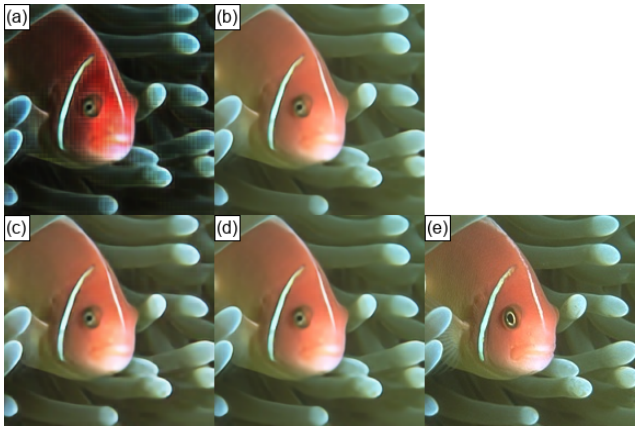

Figure 5. Examination of intermediate outputs from our downsampling aware implementation of EDRN [23] for a sample image from BSDS100 [27] before and after the DA-operator is applied. (a): output prior to DA layer; (b): final output after DA layer; (c) and (d): same as (a) and (b) but with regularization on α (the magnitude of the correction); (e): ground truth image for reference.

5. Conclusions

Here, we demonstrated a new method to ensure that the output from any super-resolution-CNN, when downsampled with 2D averaging, exactly reproduces the low resolution input. In addition to providing physical consistency between the input and output data, this approach improves the CNN performance for many different super resolution architectures. The method involves constructing the CNN with "awareness" of the downsampling scheme, and does not require any modifications to the data, training procedure, or loss function. While the approach used here focuses on downsampling with 2D-averaging, we believe that it can be modified to work with other downsampling schemes, and found that even when the assumption of 2D-averaging is incorrect, the skill of the CNN is not dramatically reduced.

The datasets used here mostly consist of images from handheld cameras, where the perceptual quality of the HR output is the primary goal. CNN-based super resolution is applicable to other types of imagery and gridded data however, where a guarantee that the statistics of the LR data are preserved is much more impactful. For example, this approach could be used to: super resolve radar data while preserving rain rates; generate high resolution satellite imagery without introducing non-physical radiances; or downscale coarse resolution output from a numerical model without breaking physical conservation laws, conservation of mass, momentum, or energy for instance. In these applications preserving the LR image statistics in the HR image is paramount, and the technique presented here can deliver the high visual fidelity provided by CNN-based super resolution schemes without sacrificing physical consistency.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image 2 style-gan: How to embed images into the style-gan latent space? *International Conference on Computer Vision (ICCV)*, 2019. 2, 7

[2] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. *CVPRW*, 2017. 5, 6

[3] Saeed Anwar and Nick Barnes. Densely residual laplacian super-resolution. *arXiv*, 1906.12021, 2019. 1

[4] S. Baker and T. Kanade. Super resolution optical flow. *Tech. Report, CMU-RI-TR-99-36, Robotics Institute, Carnegie Mellon University*, 1999. 1

[5] Jorge Bano-Medina, Rodrigo Manzanas, and Jose Manuel Gutierrez. Configuration and intercomparison of deep learning neural models for statistical downscaling. *Geosci. Model Dev.*, 13:2109–2124, 2020. 2

[6] A. A. Bastidas and H. Tang. Channel attention networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 881–888, 2019. 1

[7] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. *BMVC*, 2012. 6

[8] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016. 1, 4, 5, 11

[9] Peter Feldens. Super resolution by deep learning improves boulder detection in side scan sonar backscatter mosaics. *Remote Sensing*, 12(14):2248, 2020. 2

[10] A. Fujimoto, T. Ogawa, K. Yamamoto, Y. Matsui, T. Yamasaki, and K. Aizawa. Manga109 dataset and creation of metadata. *MANPU*, 2016. 6

[11] Richard A. Fulton, Jay P. Breidenbach, Dong-Jun Seo, Dennis A. Miller, and Timothy O'Bannon. The wsr-88d rainfall algorithm. *Weather Forecasting*, 13(2):377–395, 1998. 2

[12] A. Geiss and J. C. Hardin. Radar super resolution using a deep convolutional neural network (accepted). *Journal of Atmospheric and Ocean Technology*, 2020. 2, 4, 11

[13] Mariana-Iuliana Georgescu, Radu Tudor Ionescu, and Nicolae Verga. Convolutional neural networks with intermediate loss for 3d super-resolution of ct and mri scans. *IEEE Access*, 8:49112–49124, 2020. 2

[14] M. Haris, G. Shakhnarovich, and N. Ukita. Deep back-projection networks for super-resolution. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1664–1673, 2018. 4, 11

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1, 5, 11, 12

[16] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. 1, 4, 5, 11

[17] J.-B. Huang, A. Singh, and N. Ahuja. Single image super resolution from transformed self-exemplars. *CVPR*, 2015. 6

[18] Y. Huang, W. Wang, and L. Wang. Video super-resolution via bidirectional recurrent convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):1015–1028, 2018. 1

[19] Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. i-revnet: Deep invertible networks. *arXiv*, 1802.07088, 2018. 2

[20] J. Kim, J. Lee, and K. Lee. Accurate image super-resolution using very deep convolutional networks. *arXiv*, 1511.04587, 2016. 1

[21] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Fast and accurate image super-resolution with deep laplacian pyramid networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5835–5843, 201y. 4, 11

[22] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 105–114, 2017. 1

[23] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee. Enhanced deep residual networks for single image super-resolution. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1132–1140, 2017. 5, 8, 11

[24] G. Liu, J. Ke, and E. Y. Lam. Cnn-based super-resolution full-waveform lidar. *Imaging and Applied Optics Congress, The Optical Society of America*, JW2A.29, 2020. 2

[25] J. Long, E. Shelhamer, and T. Darell. Fully convolutional networks for semantic segmentation. *arXiv*, 1411.4038, 2014. 1

[26] B. Markham, J. Barsi, G. Kvaran, L. Ong, E. Kaita, S. Biggar, J. Czapla-Myers, N. Mishra, and D. Helder. Landsat-8 operational land imager radiometric calibration and stability. *Remote Sens.*, 6:12275–12308, 2014. 2

[27] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. *ICCV*, 2001. 6, 8

[28] Sachit Menon, Alexandru Damian, Shijia Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. *arXiv*, 2003.03808, 2020. 2, 7

[29] K. Nasrollahi and T. Moeslund. Super-resolution: a comprehensive survey. *Machine Vision and Applications*, 25:1423–1468, 2014. 1

[30] Mash Nishihama, Robert Wolfe, David Solomon, Frederick Patt, Jeffrey Blanchette, Albert Fleig, and Edward Masuoka. Modis level 1a earth location:algorithm theoretical basis document version 3.0. *NASA*, 1997. 2

[31] NOAA. Goes-r product algorithm theoretical basis documents. https://www.goes-r.gov/resources/docs.html. Accessed: 2020-10-25. 2

[32] A. Richard, I. Cherabier, M. R. Oswald, V. Tsiminaki, M. Pollefeys, and K. Schindler. Learned multi-view texture super-resolution. *arXiv*, 2001.04775, 2020. 1

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, 2015. 4, 11

[34] Wenzhe Shi, Jose Caballero, Ferenc Huszár, 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. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016. 1, 12

[35] Wanjie Sun and Zhenzhong Chen. Learned image downscaling for upscaling using content adaptive resampler. *IEEE Transactions on Image Processing*, 29:4027–4040, 2020. 2

[36] Y. Tao and J-P. Muller. Super-resolution restoration of misr images using the ucl magigan system. *Remote Sensing*, 11, 2018. 1

[37] R. Timofte, V. De Smet, and L. Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. *12th Asian Conference on Computer Vision, Singapore*, pages 111–126, 2015. 1

[38] T. Tong, G. Li, X. Liu, and Q. Gao. Image super-resolution using dense skip connections. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4809–4817, 2017. 5, 11

[39] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempit-sky. Deep image prior. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 7

[40] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 4:600–612, 2004. 5, 6

[41] Z. Wang, J. Chen, and S. Hoi. Deep learning for image super-resolution: A survey. *arXiv*, 1902.06068v2, 2020. 2, 5, 6

[42] R. Zeyde, M. Elad, and M. Protter. On single image scaleup using sparse-representations. *International Conference on Curves and Surfaces*, 2010. 6

[43] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu. Residual dense network for image super-resolution. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2472–2481, 2018. 5, 12

APPENDIX

1. Individual CNN Descriptions and Sample Outputs

SR-CNN

[8] is an early application of convolutional neural networks to the task of image super resolution. The authors use a relatively simple CNN with only three convolutional layers and ReLU transfer functions to map a LR image to its HR counterpart. Here, we apply the so-called "9-5-5" version of this model. It consists of a 3-layers with 64-9x9 filters, 32-5x5 filters, and 3-5x5 filters in that order. As implemented here, it has 6.9×10^4 trainable parameters. Unsurprisingly, it is outperformed by all of the other CNNs, all of which have much more sophisticated design and many more trainable parameters.

Lap-SRN

[21] design a CNN based on Laplacian pyramid upsampling. While some earlier approaches [8] first upsampled images using an interpolation scheme, and then processed the upsampled image with a CNN, [21] instead operate directly on the LR image, and progressively upsample it using transposed convolutions. Their CNN consists of two branches, a feature extraction branch, that infers HR residual features from the LR image, and a image reconstruction branch, that generates images at the intermediate resolutions between the LR input and the HR output by adding the residuals produced by the feature extraction branch. Our implementation of the Lap-SRN scheme uses 8.4×10^5 trainable parameters. It is outperformed by all of the CNNs but SR-CNN and the Dense U-Net. It is also one of the earlier implementations of CNN-based SISR.

Dense U-Net (DUN)

This CNN combines concepts from [33] and [16]. It is a U-net style architecture where each of the 2-3 convolution blocks in the U-net have been replaced with a block of densely connected convolutions. This architecture has previously been used for applying super-resolution to weather radar data [12]. The U-net architecture was originally developed for image segmentation, but is particularly good at combining feature information at different spatial scales [33], and so seems appropriate for the SISR task. The implementation used here uses a growth rate of 38 (see [16]), downsamples the input to 1/4 of its original resolution at the lowest level of the U-net, and uses 3 convolutions per densely connected block. It has 5.3×10^6 trainable parameters.

Deep Back Projection Network (DBPN)

The deep back projection network [14] uses a series of blocks that repeatedly upsample and then downsample the input image between the input and target resolutions using transposed and strided convolutions (respectively). The high- and low-resolution feature representations from prior blocks are concatenated and fed into subsequent blocks, much like in a densely connected network [16]. In our implementation for 4x SR we use a filter size of 8x8 for the up- and down-sampling layers. We use a total of eight blocks where each block consists of four convolutional layers organized as either: up-down-up-down or down-up-down-up, with the order depending on the input resolution. Each of the convolutional layers use 64 filters. This results in 5.6×10^6 trainable parameters.

Dense SR Net (DSRN)

This CNN directly extends concepts from [16] to super resolution [38]. The network consists of a series of densely connected blocks of convolutional layers. Dense blocks concatenate the feature outputs of all previous convolutional layers before feeding them into the next one. The intuition behind this is that while the number of features in each individual layer is smaller than other CNN architectures, the CNN can learn to use combinations of different features from diffrent parts of the network. It also provides more direct paths between the input and output of the network, which helps reduce the vanishing gradient problem. The blocks operate at low spatial resolution and build up a feature representation of the image. At the end of the network there is an upsampling module composed of two transposed convolutions. In our implementation, there are 8 dense blocks composed of 6 layers each with a growth rate of 16. The upsampling module uses 256 features. This results in 5.8×10^6 trainable parameters.

Enhanced Deep Residual Network (EDRN)

[23] is based on ideas from [15]. This network is composed of a series of residual blocks. These blocks pass the input to the block through several convolutional layers and then add the result to the original input. The intuition here is that

these residual skip connections provide a more direct path between the input and output of the network, and allow the neural network to learn which residual blocks and which features to use as it trains, again combating the vanishing gradient problem. "Enhanced" refers to removing several aspects of the residual blocks that were used in [15] that are not beneficial for super resolution. Batch normalization for instance. Our implementation has 16 residual blocks with 128-filter internal convolutional layers. Like DSRN, this approach operates at low resolution and then upsamples as one of the final operations in the network architectures. Here the upsampling is done using a pixel shuffle however [34]. The implementation used here has 5.2×10^6 trainable parameters.

Residual Dense Network (RDN)

The residual dense network [43], as the name suggests, combines aspects of both residual and densely connected networks. Like the last two CNNs discussed, it operates at low spatial resolution and builds up a feature representation of the image before upsampling it in the final layers of the network. Internally, it is composed of densely connected blocks, each block however is followed by a feature compression layer (1x1 convolution) that allows the input to the block to be added to the output (a residual connection). Finally, the network includes a long skip connection between the first layers and the last. Upsampling is performed with a pixel shuffle. Our implementation uses 10 blocks of 9 layers each with 32 channels per layer. This yields 5.3×10^6 parameters.

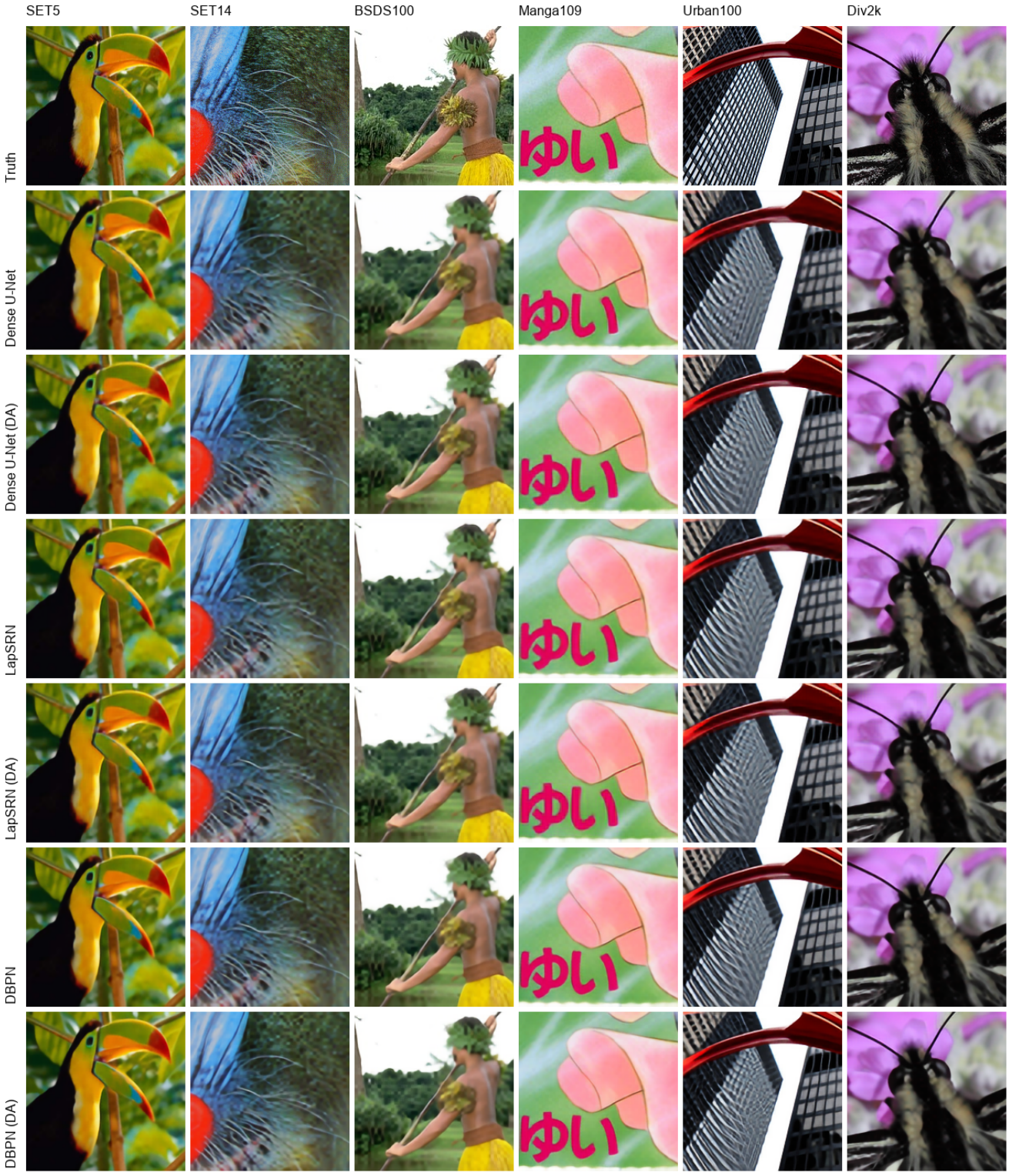

Figure 6. An example of applying super resolution to an image chip from each of the training sets for three of the CNNs, both with and without downsampling awareness.

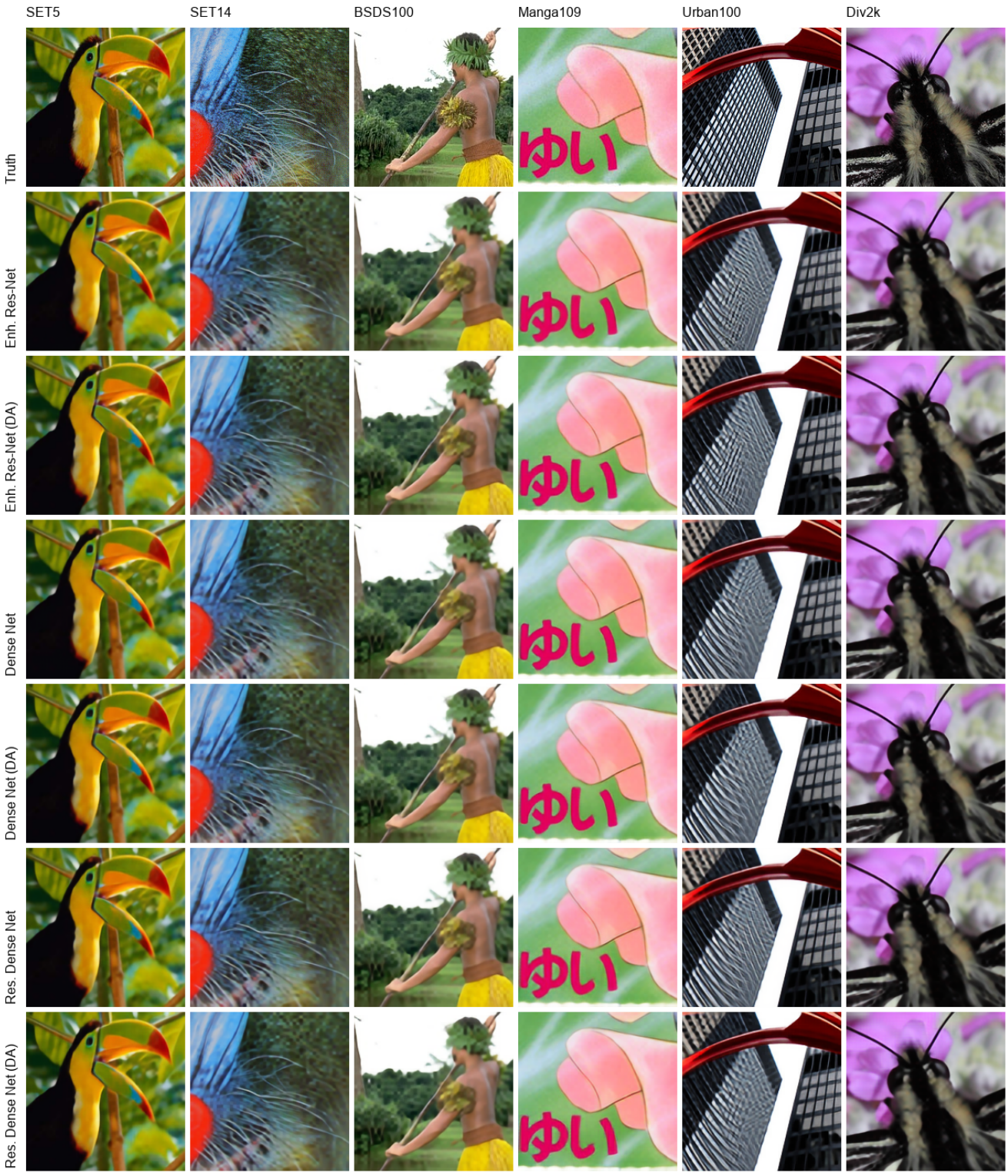

Figure 7. An example of applying super resolution to an image chip from each of the training sets for three of the CNNs not shown in Figure 6, both with and without downsampling awareness.

2. Proofs of Eqs. 1-4:

PROOF OF (1):

Given Eqn. 11:

$$f(x_i) = x_i + (P - \bar{x})\left(\frac{\sigma + x_i}{\sigma + \bar{x}}\right), \quad \text{where:} \quad \sigma = \text{sgn}(\bar{x} - P), \quad \text{and:} \quad \bar{x} = \frac{1}{N^2}\sum_{x_i \in C_P} x_i$$

$$\frac{1}{N^2}\sum_{x_i \in C_P} f(x_i) = \frac{1}{N^2}\sum_{x_i \in C_P}\left[x_i + (P - \bar{x})\left(\frac{\sigma + x_i}{\sigma + \bar{x}}\right)\right]$$

$$= \frac{1}{N^2}\sum_{x_i \in C_P} x_i + \left(\frac{P - \bar{x}}{\sigma + \bar{x}}\right)\left[\frac{1}{N^2}\sum_{x_i \in C_P}\sigma + \frac{1}{N^2}\sum_{x_i \in C_P} x_i\right]$$

$$= \bar{x} + \frac{(P - \bar{x})}{(\sigma + \bar{x})}(\sigma + \bar{x}) = \bar{x} + P - \bar{x} = P$$

PROOF OF (2):

The remaining proofs require the piecewise definition on $f(x_i)$ given in (10):

$$f(x_i) = \begin{cases} x_i + \left(\frac{P - \bar{x}}{1 - \bar{x}}\right)(1 - x_i) & \bar{x} < P \\ x_i & \bar{x} = P \\ x_i + \left(\frac{P - \bar{x}}{1 + \bar{x}}\right)(1 + x_i) & \bar{x} > P \end{cases}$$

Additionally the constraint that $P, x_i, \bar{x} \in [-1, 1]$ is needed.

For the $\bar{x} < P$ case: defining $\alpha = (P - \bar{x})/(1 - \bar{x})$ then if $\alpha \in [0, 1]$, $f(x_i) \in [x_i, 1] \in [-1, 1]$.

$$\text{Starting from:} \quad \bar{x} < P \leq 1$$

$$0 < P - \bar{x} \leq 1 - \bar{x}$$

$$\text{noting that:} \quad \bar{x} < 1 \rightarrow -\bar{x} > -1 \rightarrow 1 - \bar{x} > 0$$

$$0 < \frac{P - \bar{x}}{1 - \bar{x}} \leq 1 \quad \therefore \alpha \in [0, 1] \quad \text{and} \quad f(x_i) \in [-1, 1]$$

The $\bar{x} = P$ case is trivial because $f(x_i) = x_i$ and since $x_i \in [-1, 1]$, $f(x_i) \in [-1, 1]$.

For the $\bar{x} > P$ case: defining $\alpha = (\bar{x} - P)/(\bar{x} + 1)$ then $f(x_i) = x_i - \alpha(1 + x_i)$ and if $\alpha \in [0, 1]$, $f(x_i) \in [-1, x_i] \in [-1, 1]$.

$$\text{Starting from:} \quad -1 \leq P < \bar{x}$$

$$1 \geq -P > -\bar{x}$$

$$\bar{x} + 1 \geq \bar{x} - P > 0$$

$$\text{noting that:} \quad \bar{x} > -1 \rightarrow 1 + \bar{x} > 0$$

$$1 \geq \frac{\bar{x} - P}{\bar{x} + 1} > 0 \quad \therefore \alpha \in [0, 1] \quad \text{and} \quad f(x_i) \in [-1, 1]$$

PROOF OF (3):

We would like $f(x_i)$ to have the property that $x_i > x_j \rightarrow f(x_i) \geq f(x_j)$. Note that here x_i and x_j are high resolution pixel value outputs from the algorithm that each correspond to the same input pixel P. This means that their corresponding values of P and $\bar{x}$ are the same. Again consider Eqn. 11:

For the $\bar{x} < P$ case: taking $\alpha = (P - \bar{x})/(1 - \bar{x})$, $f(x_i) = x_i + \alpha(1 - x_i)$:
We know from the proof of (2) that for $\bar{x} < P$: $\alpha \in [0, 1]$ so $(1 - \alpha) \in [0, 1]$ and:

$$\text{if} \quad x_i > x_j$$

$$x_i(1 - \alpha) \geq x_j(1 - \alpha)$$

$$x_i + \alpha - \alpha x_i \geq x_j + \alpha - \alpha x_j$$

$$x_i + \alpha(1 - x_i) \geq x_j + \alpha(1 - x_j)$$

$$f(x_i) \geq f(x_j)$$

For the $\bar{x} = P$ case: this is trivial because $f(x_i) = x_i$ and $f(x_j) = x_j$ so if $x_i > x_j \rightarrow f(x_i) > f(x_j)$.

For the $\bar{x} > P$ case: taking $\alpha = (\bar{x} - P)/(\bar{x} + 1)$, and $f(x_i) = x_i - \alpha(1 + x_i)$:
We know from the proof of (2) that for $\bar{x} > P$: $\alpha \in [0, 1]$ so $(1 - \alpha) \in [0, 1]$ and:

$$\text{if} \quad x_i > x_j$$

$$x_i(1 - \alpha) \geq x_j(1 - \alpha)$$

$$x_i - \alpha - \alpha x_i \geq x_j - \alpha - \alpha x_j$$

$$x_i - \alpha(1 + x_i) \geq x_j - \alpha(1 + x_j)$$

$$f(x_i) \geq f(x_j)$$

PROOF OF (4):

Here, we use the piecewise definition of $f(x_i)$ given in (10) and demonstrate its differentiability by finding the partial derivatives with respect to x_i. Note that $\bar{x}$ is a function of x_i so $\partial\bar{x}/\partial x_i = N^{-2}$:
For $\bar{x} < P$:

$$f(x_i) = x_i + \frac{P - \bar{x}}{1 - \bar{x}}(1 - x_i) = \frac{x_i - x_i\bar{x}}{1 - \bar{x}} + \frac{P - Px_i - \bar{x} + \bar{x}x_i}{1 - \bar{x}} = \frac{x_i + P - Px_i - \bar{x}}{1 - \bar{x}}$$

$$\frac{\partial f}{\partial x_i} = \left[(1 - \bar{x})(1 - P - N^{-2}) + N^{-2}(x_i + P - Px_i - \bar{x})\right]/(1 - \bar{x})^2$$

$$= \left(1 - P - N^{-2} - \bar{x} + P\bar{x} + N^{-2}\bar{x} + N^{-2}x_i + N^{-2}P - N^2Px_i - N^{-2}\bar{x}\right)/(1 - \bar{x})^2$$

$$= (P - 1)(\bar{x} - 1 + (1 - x_i)/N^2)(1 - \bar{x})^{-2}$$

We omit the derivation of the other derivative as it is very similar, the full derivative of $f(x_i)$ is:

$$\frac{\partial f(x_i)}{\partial x_i} = \begin{cases} (P - 1)(\bar{x} - 1 + (1 - x_i)N^{-2})(1 - \bar{x})^{-2} & \bar{x} < P \\ 1 & \bar{x} = P \\ (P + 1)(\bar{x} + 1 - (1 + x_i)N^{-2})(1 + \bar{x})^{-2} & \bar{x} > P \end{cases}$$