Super-Resolution landscape

Ajinkya Indulkar
UMass Amherst
140 Governors Drive
Amherst, MA 01003
aindulkar@cs.umass.edu

Sarvesh Upadhyay
UMass Amherst
140 Governors Drive
Amherst, MA 01003
supadhyay@cs.umass.edu

Abstract

Super-resolution refers to the task of increasing the dimensions of an image from a Low resolution image to a High resolution image. This problem has been an active research problem for the past couple of decades. Earlier techniques used interpolation in the form of bicubic interpolation, bi-linear interpolation etc. Recently people have used CNNs, GANs and Autoencoders to do image super-resolution with convincing results. More recently researchers used randomly-initialized deep neural nets as priors to achieve promising results. In this project, we compare two modern revolutionary techniques used for Super-resolution: LapSRN and Deep Image Prior. Furthermore, we experiment Charbonnier Loss with Deep Image Prior. We also experiment with weighted intermediate losses in LapSRN instead of an unweighted loss.

1. Introduction

Super-resolution task refers to converting a Low resolution image to a High resolution image using upscaling. It is a standard inverse problem and is comparable to denoising in that sense. Super-resolution is ill-posed because there are multiple High resolution images that map to a single Low resolution image.

Traditionally, super-resolution has been performed by using standard interpolation techniques such as Bicubic Interpolation [6]. Such techniques have a downside that the upscaling performed by them is general to all images and is not dataset specific. Because of that, it fails to preserve the original statistics of the image. Additionally, the upscaled images look blurry and have little aesthetic value to humans. Newer Deep Learning based techniques result in aesthetically better looking images with sharp edges and better peak signal to noise ratio.

Dont et al. [3] introduced a Super-resolution Convolutional Neural Network (SRCNN) that learns a mapping from a LR image to a HR image using a Deep Convolutional Neural Network. One downside of this paper was that they used traditional upscaling methods such as bicubic interpolation to upscale the image and then perform super-resolution. This results in artificially smooth images and result in unnatural results. Newer techniques use a transposed convolution or a deconvolution to learn a filter that maps from a downscaled image to a upscaled image.

Traditional methods also used ℓ_2 loss during optimization which results in blurry images. LapSRN replaces the ℓ_2 loss with Charbonnier Loss [2] which results in sharper images. Most older methods also perform a single upsampling using a predefined interpolation which works well for low-scale super-resolution but doesn't work well with high-scale super-resolution. LapSRN fixes this by using a transposed convolution instead of traditional interpolation based methods. LapSRN also does progressive predictions at intermediate scales. The entire network is differentiable and therefore can be trained end-to-end without any supervision.

Some recent techniques [10] used Generative Adversarial Network to learn an Adversarial network to do super-resolution of low-res images.

Another technique proposed by Ulyanov et al. [13] used image priors or regularizers to learn image super-resolution as opposed to network weights.

It is apparent that there are numerous techniques in the current super-resolution landscape that use different ideas for super-resolution.

Our contribution. We compare different modern Deep Learning based techniques to some traditional techniques such as interpolation. We also apply Charbonnier penalty in Deep Image Prior to see the effects. Charbonnier loss results in sharper images in case of LapSRN, so our hypothesis is that it should result in better performance. In LapSRN, the losses at each level are simply summed up. We experiment with a weighted sum of losses.

2. Related Work

SRCNN[3] was one of the earliest CNN based Super-resolution techniques. It had one downside as mentioned

earlier. Additionally, inference was slow and couldn't be used in real-time.

Super-Resolution Using Very Deep Convolutional Networks (VDSR) [7] introduced the idea of up-scaling the image in the first step and learning a residual image that when added to this upscaled image results in a super-resolved image. This methods was orders of magnitude faster than SRCNN[3] and also had better PSNR performance than SRCNN[3]. Using residuals also result in sharper lines compared to directly learning the HR image.

Deeply-recursive convolutional network (DRCN)[8] tries to increase depth of paper by sharing parameters and using recursion. In all the above methods, features are extracted from the HR image. This makes training significantly harder since the number of parameters increases.

Fast Super-Resolution Convolutional Neural Networks (FSRCNN)[4] built upon SRCNN by adding a hourglass structure. FSRCNN learns coarse-grained features from LR images and then uses a deconvolution operator to upscale the feature maps in the final layer. This significantly speeds up inference since most of the network operates on the LR image and the up-scaling only happens at later layers. In fact, FSRCNN achieves a 40X speedup over SRCNN. This makes it suitable for real-time super-resolution at 24fps on a non-expensive CPU.

Some methods[5, 1, 11] have used self similarity in images to model similar looking patches inside an image.

Above mentioned approaches relies on training a network end to end from random initialization or fine tuning the network and later running inference on the network. Deep Image Prior [13], a diametrically opposite approach to solve super resolution problem, does not require any training of the generator network. Deep Image Prior attempts to improve resolution of image over number of network iterations. This is done by postulating that low-level image statistics required for image generation problem is encoded by the structure of generator network.

3. Approach

In this project we concentrate on the problem of super-resolution. It refers to the task of increasing the resolution of an image from a Low Resolution (LR) image to a High Resolution image (HR).

3.1. Baseline

We use simple bicubic interpolation as a baseline. Although, this baseline will just serve as a comparison for older methods and is not representative of today's landscape. Any recent model will surely achieve better PSNR.

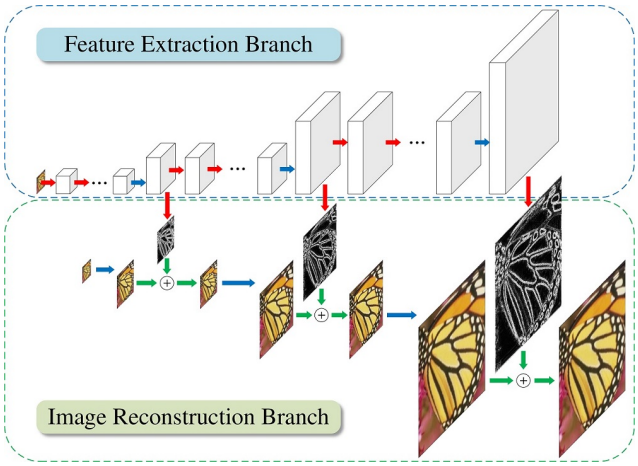


Figure 1: LapSRN architecture. Red arrows=Conv, Blue arrows=Deconv, Green=Elemwise Sum.

3.2. Laplacian Pyramid Super-Resolution Network (LapSRN)[9]

LapSRN[9] as the name implies uses a pyramid architecture as shown in Figure 1. It consists of multiple *levels*. *Levels* are defined as a group of layers after which we make an intermediate (or final) prediction. After each level we scale the image by 2X. So to scale the image by a factor of S, we have $\log_2 S$ *levels*. For 8X super-resolution we will have 3 *levels*. The network consists of two branches:

Feature extraction Each level s has of d conv layers and one transposed convolutional layer that upscales the residual. The output of this transposed convolutional layer is used as input to the next level and is also used to construct a residual image for the Image reconstruction branch. Each layer learns coarse-grained features and converts it to fine-grained feature maps. This results in a significant speedup since we are learning features from the LR image.

Image reconstruction. In each level, the input image is upscaled by 2X with a transposed convolutional layer. The upscaling layer is initialized so that it performs bilinear upscaling. This is then summed element-wise to the residual image from the feature extraction branch. This image is an intermediate super-resolved image. It can also be used as an input to later layers to perform more scaling.

We also use the robust Charbonnier Loss function. [2] as described in Equation:

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

where $\rho(x) = \sqrt{x^2 + \varepsilon^2}$ where ε is set to a small number like $1e-3$. L is the number of *layers* and N is the number

of data cases. This loss is more robust than ℓ_2 loss and handles outliers well. The output at each level computes a loss and the total loss is the sum of these losses. For example, in case of 8X super-resolution, the total loss is the sum of loss from the 2X, 4X and 8X super-resolved image. This is a very important property in real-life where you might have devices having limited computational resources. Additionally, different devices have different needs. A mobile phone has limited screen resolution and doing super-resolution at a high scale might not make sense. In that case we can just stop the forward pass after a few *layers* and get a HR image.

For the case of 4X super-resolution the final loss will be the sum of losses of the 2X image and the 4X image. Let's call them $\mathscr{L}_1$ and $\mathscr{L}_2$. The final loss $\mathscr{L}$ is an unweighted sum of these losses.

$$\mathscr{L} = \mathscr{L}_1 + \mathscr{L}_2$$

We try out different weight combinations by weighing each loss $\mathscr{L}_i$ by w_i. So, our final loss becomes:

$$\mathscr{L} = w_1\mathscr{L}_1 + w_2\mathscr{L}_2 \tag{2}$$

Lai et. al. [9] go into detail and show how the three main highlights of LapSRN: (1) Pyramid structure (2) Robust Loss (3) Residual Learning are important for the network and without these the model performance degrades significantly.

Training details. Convolutional layers have 64 filters of size 3×3. All non linearities are Leaky ReLUs with a slope of 0.2. The transposed convolutional layer has size 4×4. We use a batch size of 16 and randomly crop a 128×128 patch. We also randomly do data augmentation by performing three transformations on training images: (1) Scale image randomly between 0.5 and 1. (2) Rotate images randomly (3) Flip images randomly. We use Adam optimizer with a weight decay of $1e-4$ and a learning rate starting at 0.001 that is halved every 10 epochs. We use a depth $d = 5$.

3.3. Deep Image Prior

Image super-resolution can also be formulated as an optimization task or energy minimization task of the form

$$x^* = \min_x E(x; x_0) + R(x)$$

where x is the HR image and x_0 is the LR image. The first term is the data term and the second term can be seen as a handcrafted prior or a regularization term.

For the purposes of super-resolution, we set the data term to:

$$E(x; x_0) = ||d(x) - x_0||^2$$

where $d(.) : \mathbb{R}^{3\times tH\times tW} \rightarrow \mathbb{R}^{3\times H\times W}$ is a down-sampling operator that maps HR image to LR image. i.e. $d(.)$ scales-down an image by a factor of t. We observe that $d(.)$ is not an onto function since there can be many HR images x that map to a LR image x_0. To fix this surjective function, we have the regularizer $R(x)$ that helps the system to select the most plausible HR images.

The Deep Image Prior paper uses a generator network $x = f_\theta(z)$ to generate x and uses gradient descent to optimize θ. This network can be used as an inherent regularizer and the optimization problem can now be reformulated as follows:

$$\theta^* = arg\min_\theta E(f_\theta(z); x_0), \quad x^* = f_{\theta^*}(z)$$

Here, $z \in \mathbb{R}^{C'\times H\times W}$ is the latent vector and $x^* = f_{\theta^*}(z) \in \mathbb{R}^{3\times H\times W}$. The authors used a uniformly distributed code vector.

It might seem that the generator should over-fit on the noisy image but the authors of the paper showed that the network offer high impedance to noise and low impedance to image signal. This can be characterized as "reluctance" of network to learn LR image and it first learns HR image and then learns LR images. So, stopping the training process early should result in the generator producing HR images. Priors restricts the generated image to resemble natural image and not noise. Randomly initialized UNet architecture have an implicit "prior" and they resist generation of noisy images and have a bias towards natural images. Stating in a different way, the structure of the network is enough to encode most of low-level image statistics required for image generation. Therefore, HR images can be generated without any prior training of the generator network.

The authors have used a hourglass (or decoder-encoder architecture) based UNet with skip connections network 2. The network can be modularized into three components down-sampler, up-sampler and skip-connection. Latent vector z has same spatial size as of x with 32 feature maps and z is initialized with uniform noise. UNet has five up-sampling and down-sampling layers. Number of kernel in each layer is 128 and size of each kernel is 3. Each skip connections have 4 filters of size 1. LeakyRelu is used to introduce non-linearity in the network. To down-sample an image, network perform convolution with appropriate stride (as decided by factor size t). ADAM optimizer was used to perform optimization. The UNet architecture was optimized using both MSE loss function and Charbonnier Loss function. [2]. The decimation operator $d(.)$ here is differentiable down-sampling operator lanczos.

4. Experiment

We build our contributions on top of existing Deep Image Prior[1] and LapSRN[2] implementations. Our code can be

[1]https://github.com/DmitryUlyanov/deep-image-prior
[2]https://github.com/BUPTLdy/Pytorch-LapSRN

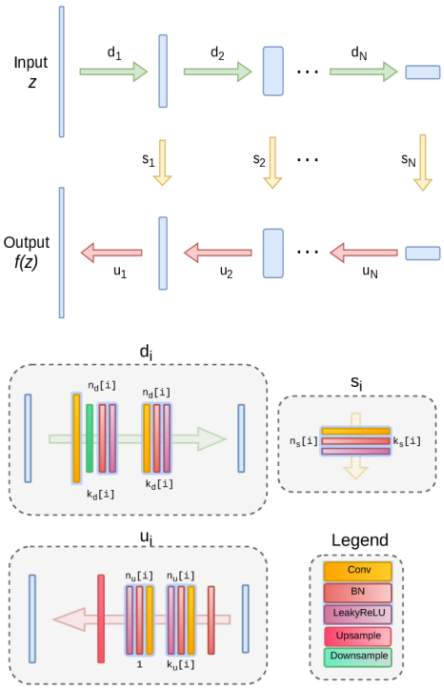


Figure 2: UNet Architecture with skip connections. $n_d[i]$, $n_u[i]$, $n_s[i]$ correspond to number of filters at depth i for down-sampling, up-sampling and skip connections respectively and $k_d[i]$, $k_u[i]$, $k_s[i]$ correspond to filter sizes.

found here[3]. For all experiments we use Python 3.6, PyTorch 0.4, CUDA 9.0, cudNN 7.0 and we train them on NVIDIA K80, GTX 1050Ti and GTX 1050 GPUs.

To solve the problem of super-resolution, we are using BSD300 image dataset [12] that consists of images of natural scenes. It consists of 200 Training images and 100 Testing images. Although most modern works perform 8X super-resolution, in this work, we only perform 4X super-resolution due to computational resource constraints. Additionally, the Deep Image Prior takes several minutes to generate a single super-resolved image. For this reason, we only do a comparison on 10 images from the testing dataset.

In order to quantitatively, measure the performance of the model we are using PSNR (Peak Signal to Noise Ratio) score which is defined as:

$$PSNR = 20\log_{10}(255) - 10\log_{10}(MSE)$$

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(I(i,j) - F(i,j))^2$$

where $F(i,j)$ is the upscaled image and $I(i,j)$ is the ground truth high resolution image.

[3]https://github.com/svh2811/Super-Resolution

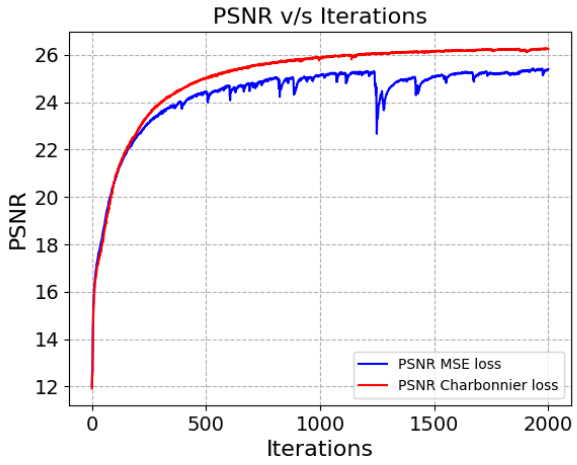


Figure 3: Plot of PSNR vs Iterations for Deep Image Prior for two different loss functions.

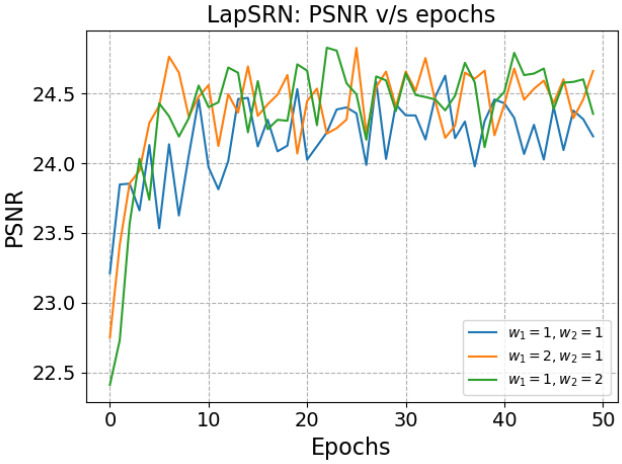


Figure 4: PSNR v/s epochs for LapSRN for different settings of w_1

Additionally, we also use SSIM (Structured Similarity) [14] to quantitatively compare performance.

LapSRN. We train the LapSRN for 50 epochs with a batch size of 16 and where each epoch consists of 30 iterations. After every epoch, we compute the entire test PSNR. Figure 4 shows the trend of PSNR v/s epochs for different values of w_1 and w_2. It appears that there is no significant difference apparent in the trend. The difference between the PSNR in Figure 4 and Table 1 is because the PSNR values in Figure 4 are computed on the entire test set whereas the results in Table 1 are computed on a subset of the Test set.

Deep Image Prior. This paper by Ulyanov et al. [13] takes a very different approach to doing SR. Instead of

learning a distribution that maps from LR images to HR images, they focus on the structure of the network to encode information about the statistics of an image. The neural network generates a High resolution image x which is downscaled to get LR image representation of generated HR image $d(x)$. Loss is calculated w.r.t to $d(x)$ and actual low resolution image x_0. The process is repeated until the PSNR value starts to saturate, for our experiment we stop after 2000 iterations. Figure 3 shows the trend of PSNR v/s iterations for Mean squared error loss function and Charbonnier loss function.

4.1. Results

Figure 5 shows the High Resolution images generated by the models that we used. LapSRN produces images that are very close to the Ground Truth image. Deep Image Prior with Charbonnier loss also generates very sharp features which are impressive considering there was no training. Figure 6 shows parts of images zoomed in to get a magnified look at parts of the image. We can see that Deep Image Prior and LapSRN generate significantly better looking images compared to Bicubic interpolation.

Figure 8 also shows the comparison of the image generated by Deep Image Prior using MSE and Charbonnier penalty. The model using Charbonnier penalty generates better pictures than the one using MSE. Similarly we also compare images generated by LapSRN for different weight configurations in Figure 9. The images look very similar and are indistinguishable.

We can see in Figure 3 that the Charbonnier loss converges significantly faster than MSE and also has less variance compared to MSE. It confirms our earlier hypothesis that it performs well with Deep Image Prior.

The final test set performance is shown in Table 1 and it shows that Deep Image Prior with Charbonnier Loss outperforms all other models by a significant margin in both metrics: PSNR and SSIM.

4.2. Limitations

In the zebra image in Figure 6, it can be seen that some fine details cannot be generated well by LapSRN or Deep Image Prior. This is a common problem with Super-resolution models. A limitation with the Deep Image Prior is that we have little insight into the actual reasoning behind why random networks are better priors. Trained networks such as LapSRN are harder to train but are very efficient at inference. In fact LapSRN can be used to perform real-time super-resolution. Deep Image Prior on the other hand requires several minutes to generate a super-resolved image which is not ideal for real-time applications.

Table 1: Quantitative evaluation of SR algorithms. **Red** indicates best and Blue indicates second best.

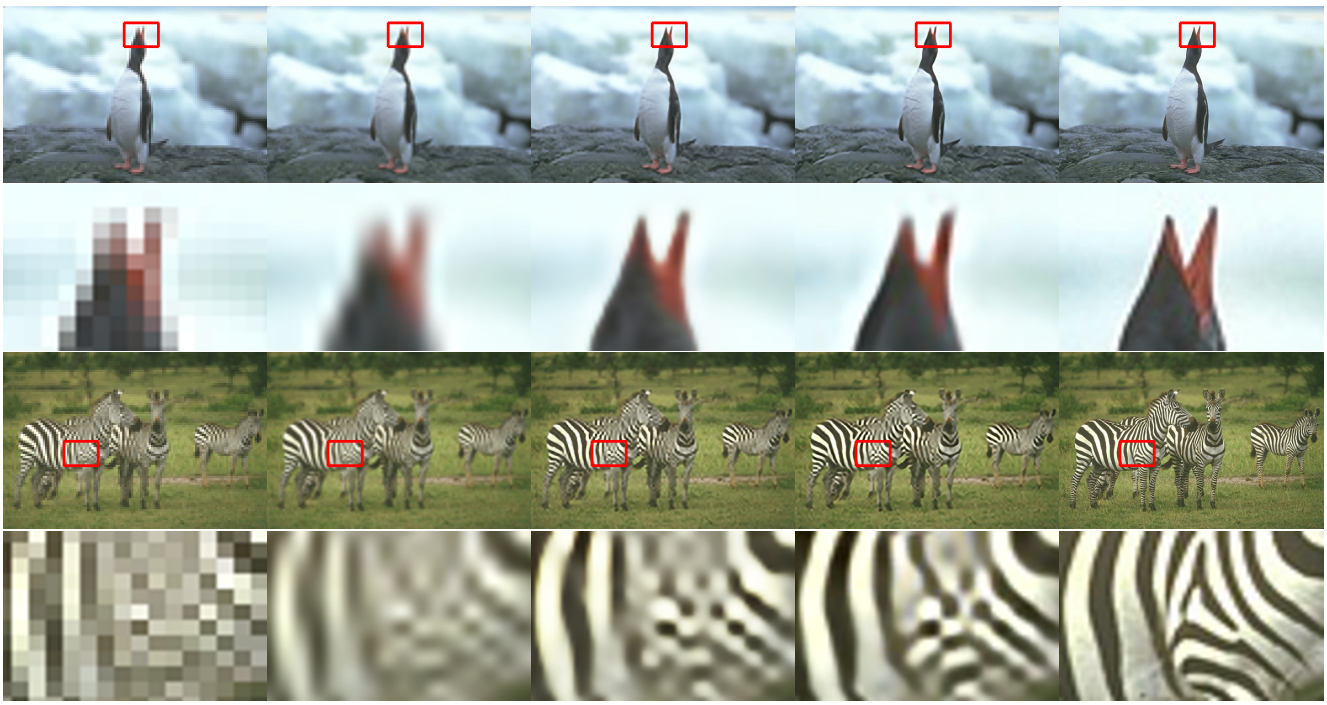
Algorithm	PSNR	SSIM
Bicubic	25.09	0.8055
LapSRN $wR_1 = 1, w_2 = 1$	26.10	0.7649
Deep Image Prior with MSE	25.45	0.8036
LapSRN $w_1 = 2, w_2 = 1$ **(Ours)**	26.03	0.7600
LapSRN $w_1 = 1, w_2 = 2$ **(Ours)**	25.99	0.7605
Deep Image Prior with Charbonnier (Ours)	**26.28**	**0.8340**

5. Conclusion

In our project we explore LapSRN and Deep Image Prior - two different approaches to performing super-resolution. We compare their architectures and discuss their pros and cons. Furthermore, we adopt the Charbonnier Loss Penalty from LapSRN in Deep Image Prior to achieve significantly better performance. This is a promising direction for further exploration. Moreover, human inspection of the images generated by Deep Image Prior using Charbonnier Loss reveal that they generated High resolution images look much better aesthetically than the other models. Another experiment we performed was using a weighted sum of the intermediate losses of LapSRN instead of a direct unweighted sum. Unfortunately, it didn't give any significant improvements. A future idea would be to use the Charbonnier Loss with the datasets used in the Deep Image Prior paper and compare the performance.

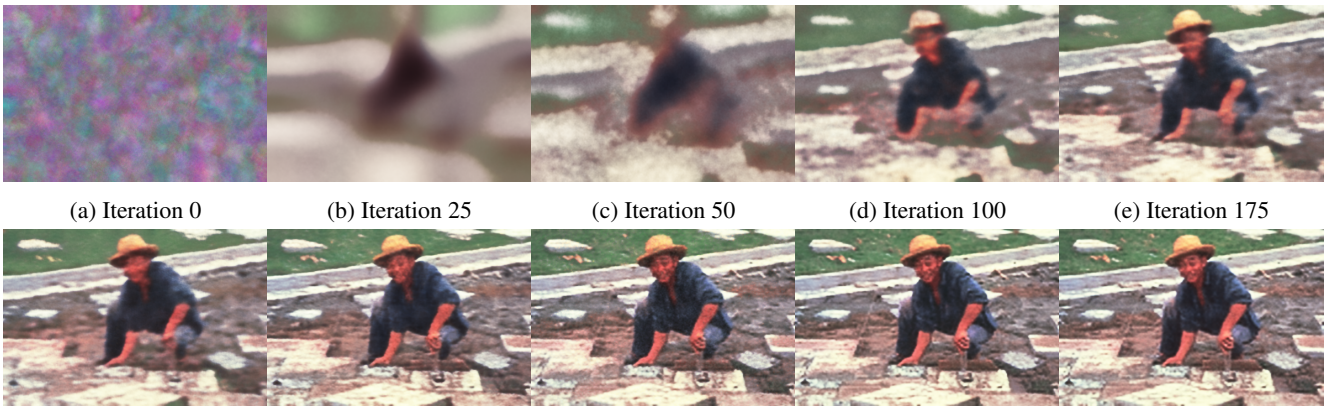
(a) Low Res Image (b) Bicubic (c) Deep Image Prior with Charbonnier loss (d) LapSRN (e) High Res Image

Figure 5: Visual comparative analysis of different super resolution algorithm



(a) Low Res Image (b) Bicubic (c) Deep Image Prior with Charbonnier loss (d) LapSRN (e) High Res Image

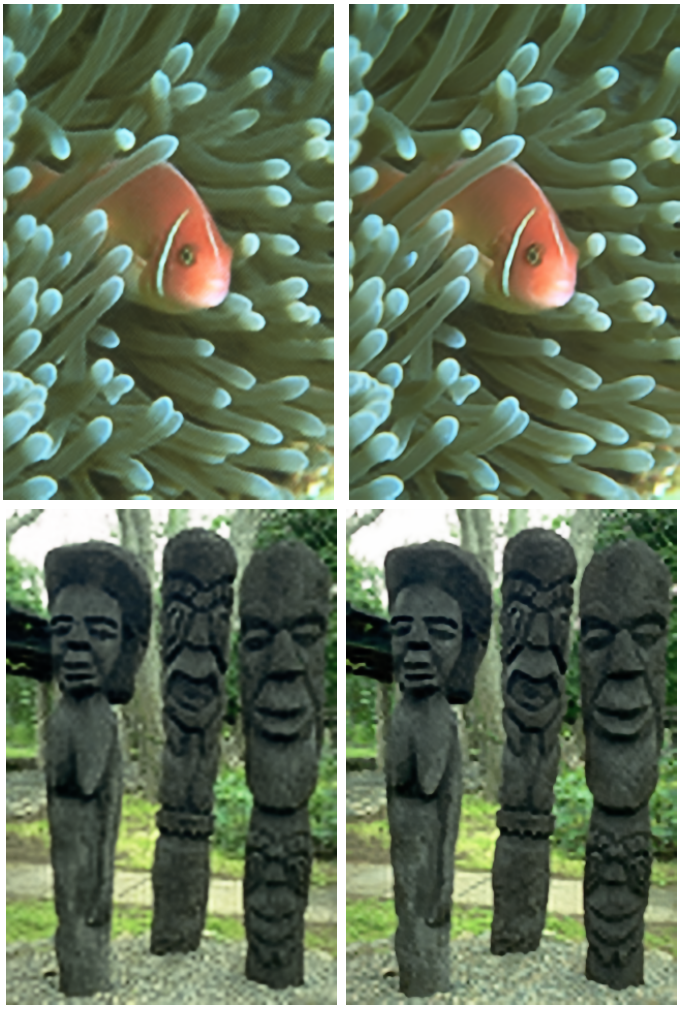
Figure 6: Zoomed Image-Clip for Highlighted region



(a) Iteration 0 (b) Iteration 25 (c) Iteration 50 (d) Iteration 100 (e) Iteration 175

(f) Iteration 250 (g) Iteration 500 (h) Iteration 1000 (i) Iteration 1500 (j) Iteration 2000

Figure 7: Images generated by UNet across Iterations



(a) Deep Image Prior with MSE Loss (b) Deep Image Prior with Charbonnier Loss

Figure 8: Visual comparative analysis of different loss function used for Deep Image Prior

(a) LapSRN with w_1=1, w_2=1 (b) LapSRN with w_1=1, w_2=2 (c) LapSRN with w_1=2, w_2=1

Figure 9: Visual comparative analysis of different weight setting used for LapSRN

References

[1] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 60–65. IEEE, 2005.

[2] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud. Deterministic edge-preserving regularization in computed imaging. *IEEE Transactions on image processing*, 6(2):298–311, 1997.

[3] 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.

[4] C. Dong, C. C. Loy, and X. Tang. Accelerating the super-resolution convolutional neural network. In *European Conference on Computer Vision*, pages 391–407. Springer, 2016.

[5] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 349–356. IEEE, 2009.

[6] R. Keys. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, 29(6):1153–1160, 1981.

[7] J. Kim, J. Kwon Lee, and K. Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, 2016.

[8] J. Kim, J. Kwon Lee, and K. Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1637–1645, 2016.

[9] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 624–632, 2017.

[10] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint*, 2016.

[11] S. Lefkimmiatis. Non-local color image denoising with convolutional neural networks. *arXiv preprint*, 2017.

[12] 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. 2:416–423, July 2001.

[13] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Deep image prior. *arXiv preprint arXiv:1711.10925*, 2017.

[14] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 2, pages 1398–1402. Ieee, 2003.