Image Super-Resolution
ECE 176

Achyut Pillai
ECE
apillai@ucsd.edu
PID: A16646336

Abdullah Ashfaq
HDSI
aashfaq@ucsd.edu
PID: A59026296

Abstract

The study aims to qualitatively and quantitatively compare various deep learning models for image super-resolution (SR) task. The dataset that we use is DIV2K which is diverse and specifically curated for SR task. We compared traditional (bicubic interpolation), CNN-based (SRCNN) and GAN-based approaches (SRGAN and ESRGAN) in their ability to transform a low-resolution images into a high-resolution images accurately. We found that the GAN-based approaches greatly surpass other methods, in particular, ESRGAN performed the best on all quantitative metrics as well as qualitative observations.

1 Introduction

Image Super-Resolution (SR) plays an important role in enhancing image quality for a variety of applications, including medical imaging, satellite imaging, video streaming, and restoration of historical images. This paper aims to reconstruct high resolution images from low resolution inputs where certain details may have been lost. Through the use of deep learning our approach intends to not only restore an overall structure but recover finer texture details.

We implemented three main architectures in our approach: the Super-Resolution Convolutional Neural Network (SRCNN), the Super-Resolution Generative Adversarial Network (SRGAN), and the Enhanced SRGAN (ESRGAN). The SRCNN provides a baseline deep learning model, optimized with mean squared error (MSE), while the SRGAN and ESRGAN are deeper GAN models that use adversarial and perceptual losses to generate more photorealistic images. Training these models on the DIV2K dataset allowed us to compare their performance with bicubic interpolation as a baseline using objective methods like PSNR, SSIM, LPIPS, and FID as well as qualitative human assessments on different upscaling factors.

Our results demonstrate that while the GAN models are more computationally expensive, they offer superior texture recovery and produce a better overall image compared to the SRCNN and the bicubic interpolation.

2 Related Work

The topic of image Super-Resoulution has seen significant progress in recent years, of which the papers on SRCNN, SRGAN, and ESRGAN inspired this project.

The Super-Resolution Convolutional Neural Network (SRCNN) introduced by Dong et al. [1] demonstrated that a simple convolutional network could learn an end-to-end mapping between low-resolution and high-resolution images.

The Super-Resolution Generative Adversarial Network (SRGAN) presented by Ledig et al. [2] incorporated adversarial loss alongside perceptual loss to generate photorealistic images with enhanced texture details.

Building on the ideas of SRGAN, Wang et al. [3] proposed the Enhanced SRGAN, which further refined the network architecture. ESRGAN introduced techniques such as residual-in-residual dense blocks, significantly improving the recovery of fine details.

3 Method

Each of our models employs its own unique strategy for constructing SR images from low-resolution (LR) inputs.

3.1 Model Architectures

3.1.1 SRCNN

We implemented two variants of the SRCNN:

- **SRCNN_v1:** This model has three convolutional layers with the first layer using a kernel of size 9. It is followed by two more convolutional layers with kernels of size 5. Each convolutional layer is followed by a ReLU activation to introduce non linearity into the model.
- **SRCNN_v2:** This variant increases the model capacity by using a larger number of filters in the first convolutional layer (128 filters) and adding an extra convolutional layer with kernel size 3 and ReLU activation before the final layer.

3.1.2 SRGAN

The SRGAN improves upon the SRCNN structure by incorporating adversarial training:

- **Generator:** The generator, based on the SRResNet architecture from [2], includes an initial convolutional layer with a large kernel followed by a series of residual blocks. Each residual block contains two convolutional layers with batch normalization and PReLU activations. After these blocks, another convolutional layer with batch normalization combines the new features with the original ones. Upsampling is then performed on the image using sub-pixel convolution and PReLU, and a final convolutional layer produces the high-resolution output.
- **Discriminator:** The discriminator is a deep convolutional network that distinguishes between real HR images and those produced by the generator. It consists of multiple convolutional layers with increasing feature dimensions, batch normalization, and LeakyReLU activations, followed by average pooling and dense layers to yield a probability score.
- **Perceptual Loss:** A VGG-based perceptual loss is computed by comparing feature representations (extracted from a layer of a pre-trained VGG network) of the generated and ground truth HR images. This loss encourages the generator to produce images that are perceptually similar to real images.

3.1.3 ESRGAN

The ESRGAN further refines the SRGAN by introducing a different generator:

- **Generator:** The ESRGAN generator uses a series of Residual in Residual Dense Blocks (RRDBs). It starts with an initial convolutional layer to extract features, followed by a stack of RRDBs. Each RRDB stacks together multiple Residual Dense Blocks (RDBs) that use dense connections and local feature fusion. After the RRDB stack, a convolutional layer fuses the features, which are then added to the initial features. Upsampling is performed through multiple stages using convolutional layers, PixelShuffle operations, and LeakyReLU activations, followed by a final convolutional layer that reconstructs the HR image.
- **Discriminator:** The discriminator architecture is identical to that in SRGAN.
- **Perceptual Loss:** The same VGG-based perceptual loss in SRGAN is used here.

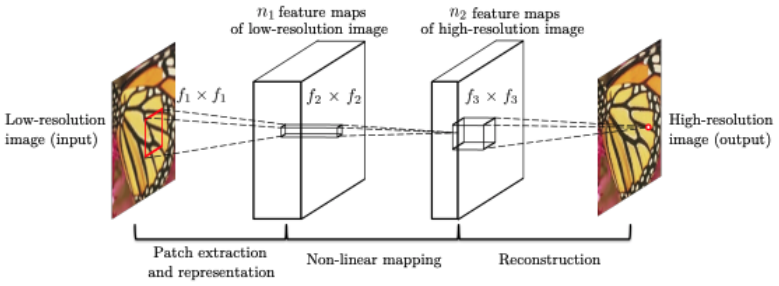

Figure 1: Original SRCNN architecture from [1].

3.2 Comparison with Original Architectures

3.2.1 SRCNN

Our work includes two variants of SRCNN:

- **SRCNN_v1:** This implementation closely follows the original design presented in Figure 1 and [1].
- **SRCNN_v2:** In contrast to the original, this variant increases model capacity by using double the number of filters in the first convolutional layer (128) and introducing an extra convolutional layer with a smaller kernel size.

3.2.2 SRGAN

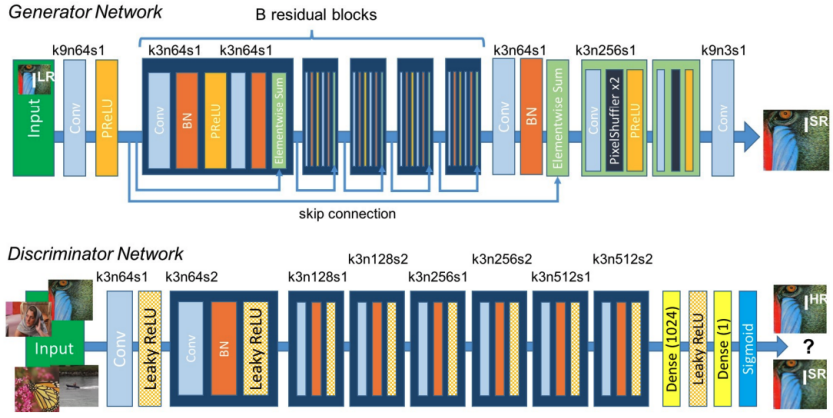

Figure 2: Original SRGAN architecture from [2].

Our implementation maintains the core framework of a generator with residual blocks, sub-pixel convolution for upsampling, and a discriminator with deep convolutional layers found in [2] (Figure 2). Our modifications include:

- **Perceptual Loss Modification:** Unlike the original SRGAN, which typically uses a VGG network without batch normalization, our implementation employs VGG19 with batch normalization. This change was made to improve the stability of the training and to see if feature extraction is improved.
- **Learning Rate Scheduling:** We use a Cosine Annealing Learning Rate Scheduler to improve convergence and stability during training.

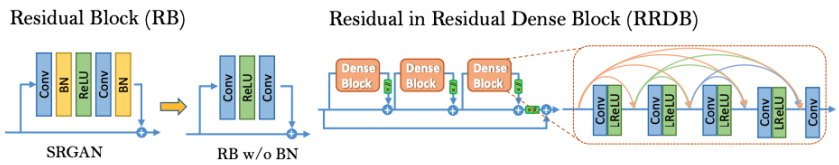

Figure 3: Original ESRGAN architecture from [3].

3.2.3 ESRGAN

For ESRGAN, our implementation builds on the Residual in Residual Dense Block (RRDB) structure described in [3] (Figure 3). The differences from the original include:

- **Hyperparameter Tuning**: Certain hyperparameters, such as growth rate and the residual scaling factors in the RRDBs, have been adjusted to provide better results on our dataset.
- **Perceptual Loss Modification:** Similar to our SRGAN implementation, we use a pretrained VGG19 with batch normalization for computing the perceptual loss.

3.3 Training Strategy

We employed slightly different training strategies for the SRCNN and the GAN-based models.

Common Loss Components: All models are trained to balance pixel-level accuracy and perceptual quality using a combination of loss functions. Adversarial and Perceptual Loss are only present in the GAN models:

- **Pixel-wise Loss:** For SRCNN, this is the Mean Squared Error (MSE) between the generated image and the ground truth HR image. In the GAN-based models, MSE is used during pretraining and L1 loss is used after pretraining.
- **Adversarial Loss:** In SRGAN and ESRGAN, a binary cross-entropy (BCE) loss is used to force the generator to produce images that the discriminator classifies as real.
- **Perceptual Loss:** This is computed as the MSE between feature representations of the generated and ground truth HR images, extracted from the pre-trained VGG network (VGG19 with batch normalization).

SRCNN Training: The training loop for SRCNN is straightforward. For each epoch:

1. **Data Loading:** Low-resolution and high-resolution image pairs are loaded.
2. **Forward Pass:** The SRCNN model processes the LR images, and the MSE is computed.
3. **Backward Pass:** Gradients are computed via backpropagation and the Adam optimizer updates the model weights.
4. **Metrics Calculation:** The PSNR is calculated over the validation set to monitor image quality.

SRGAN/ESRGAN Training: Training for the GAN-based models involves a two-step approach:

1. **Pretraining the Generator:** For a fixed number of epochs (20), the generator is trained using only the pixel-wise loss (MSE) to obtain an optimal starting point for the generator.
2. **Full GAN Training:** After pretraining, both the generator and discriminator are trained simultaneously. In each iteration:
   - **Generator Update:** The generator minimizes a composite loss that includes:
     - Pixel Loss (L1): Measures the difference between generated and HR images.
     - Perceptual Loss: Computed from VGG-based features.
     - Adversarial Loss: Encourages the generator to produce images that fool the discriminator.

The adversarial loss weight is gradually increased over a fraction of the GAN training epochs to stabilize the training process and to ensure that the generator is able to learn to reconstruct the images first.

- **Discriminator Update:** The discriminator is trained to distinguish between real HR images and generated images using BCE loss.

3. **Optimization and Scheduling:** The Adam optimizer is used for the generator and discriminator with separate learning rates. A Cosine Annealing Learning Rate Scheduler is applied during the full GAN training phase.
4. **Metrics Calculation:** The PSNR is calculated over the validation set to monitor image quality.

3.4 Evaluation Metrics

We assess the performance of our models using the following quantitative metrics, in addition to visual assessments:

- **Peak Signal-to-Noise Ratio (PSNR):** Evaluates reconstruction quality, with higher scores indicating greater similarity to the ground truth.
- **Structural Similarity Index Measure (SSIM):** Measures perceptual similarity between generated and ground-truth images, where a score of 1 indicates identical images.
- **Learned Perceptual Image Patch Similarity (LPIPS):** Quantifies perceptual similarity by comparing deep feature activations from a pre-trained network, with lower scores indicating more similar images.
- **Fréchet Inception Distance (FID):** Compares the distributions of real and generated images, with lower scores reflecting a closer resemblance.

4 Experiments

4.1 DIV2K Dataset

The DIV2K dataset [4, 5] is a diverse collection of high resolution images, specifically curated for image super-resolution tasks.

- **Data Format**: Images are in PNG format with 2K resolution.
- **Dataset Split**: 800 Training Images, 100 Validation Images

4.2 Data Preparation and Augmentation

We utilize custom dataset classes to load HR images from the DIV2K dataset. For each image:

- **Random Cropping:** A random crop of fixed size (e.g., 96×96) of each HR image is taken to generate training samples.
- **Bicubic Downsampling:** The cropped HR image is downscaled by a fixed factor (2, 4, or 8) using bicubic interpolation to produce the corresponding LR image.
- **Data Augmentation:** Random horizontal/vertical flips and rotations are applied to improve training robustness.
- **Tensor Conversion:** Images are converted to tensors and scaled to match the network's expected input range.

4.3 Results

4.3.1 SRCNN

Both versions of SRCNN were trained at three scale factors ($\times2$, $\times4$, and $\times8$) and until the loss stopped decreasing. The models trained fairly quickly within 100 epochs. Then we computed metrics on the validation dataset. Tables 1, 2, 3 show that the modification in the original structure that

Table 1: FID Scores for Bicubic, SRCNN, SRGAN, and ESRGAN at Different Scales on Validation set

Scale	Bicubic	SRCNN_{v1}	SRCNN_{v2}	SRGAN	ESRGAN
2	13.37	67.94	105.98	6.41	5.00
4	37.16	79.91	79.93	30.02	25.85
8	63.55	97.52	100.06	50.20	47.60

Table 2: Average PSNR Scores for Bicubic, SRCNN, SRGAN, and ESRGAN at Different Scales on Validation set

Scale	Bicubic	SRCNN_{v1}	SRCNN_{v2}	SRGAN	ESRGAN
2	30.99	25.67	21.26	32.83	33.53
4	26.64	24.38	24.39	27.19	27.24
8	23.09	22.07	22.00	23.19	23.43

we introduced (SRCNN_v2) didn't work and performed worse than the original architecture. And SRCNN for our task performed no better than the traditional approaches. Figure 9 visualizes SRCNN results. We hypothesize that tweaking with the optimizer, weight initialization, etc can improve SRCNN beyond bicubic interpolation but not by much. So we shifted focus on the GAN-based approaches.

4.3.2 GANs

Both GAN architectures were evaluated at three scale factors ($\times 2$, $\times 4$, and $\times 8$), resulting in three distinct generator–discriminator pairs for each of the SRGAN and ESRGAN models. The SRGAN models were trained for 1000 epochs, with the first 20 dedicated exclusively to pretraining the generator, while the ESRGAN models were trained for 500 epochs, also using the initial 20 epochs for generator pretraining.

To assess the performance of the models, quantitative metrics such as the PSNR, SSIM, LPIPS, and FID are important to consider.

Table 1 shows that both SRGAN and ESRGAN outperform the Bicubic baseline at every scale, with the ESRGAN consistently achieving the lowest FID scores. This indicates that the GAN-based approaches, particularly the ESRGAN with its enhanced generator architecture, are more effective at generating images with improved texture and overall photo-realism, especially at higher scaling factors, where the score differences become more pronounced.

To complement these quantitative findings, we can qualitatively compare the outputs of Bicubic interpolation, SRGAN, and ESRGAN against their corresponding ground-truth HR images, with some more image specific quantitative metrics. Figures 4, 5, and 6 display an example image from the DIV2K validation set at scaling factors $\times 2$, $\times 4$, and $\times 8$, respectively. Each figure presents a row with the full images of the LR image, Bicubic baseline, SRGAN, ESRGAN, and the ground-truth) along with a patch cropped from the center (size 96×96 pixels for HR images and $(96/\text{scale}) \times (96/\text{scale})$ for LR images) to highlight the finer differences.

As shown in Figure 5, the SRGAN produces sharper results than the Bicubic baseline, but the ESRGAN is able to recover even more realistic textures and finer structures, especially around edges. Quantitatively, the ESRGAN usually achieves slightly higher PSNR and SSIM values, and

Table 3: Average SSIM Scores for Bicubic, SRCNN, SRGAN, and ESRGAN at Different Scales on Validation set

Scale	Bicubic	SRCNN_{v1}	SRCNN_{v2}	SRGAN	ESRGAN
2	0.9260	0.8005	0.4037	0.9432	0.9517
4	0.8551	0.7700	0.7678	0.8572	0.8622
8	0.8001	0.7237	0.7103	0.7908	0.8018

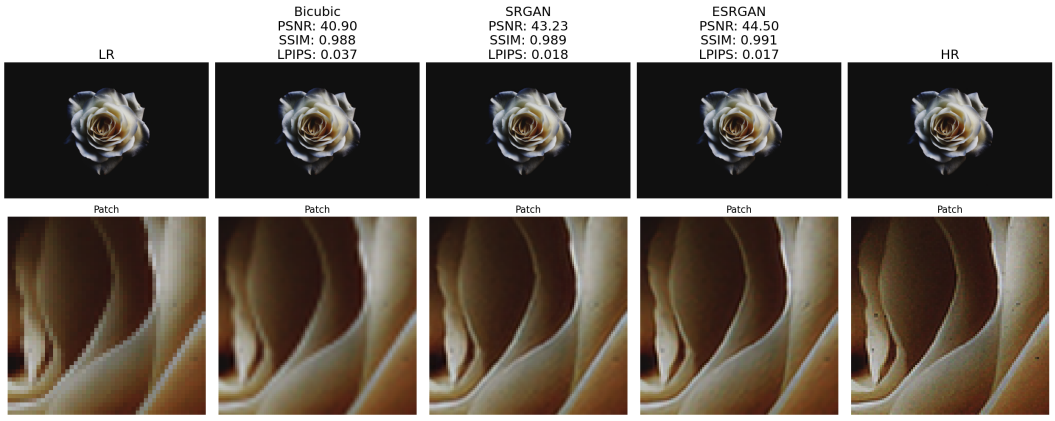

Figure 4: SRGAN/ESRGAN Results on image 0843 with scale factor 2.

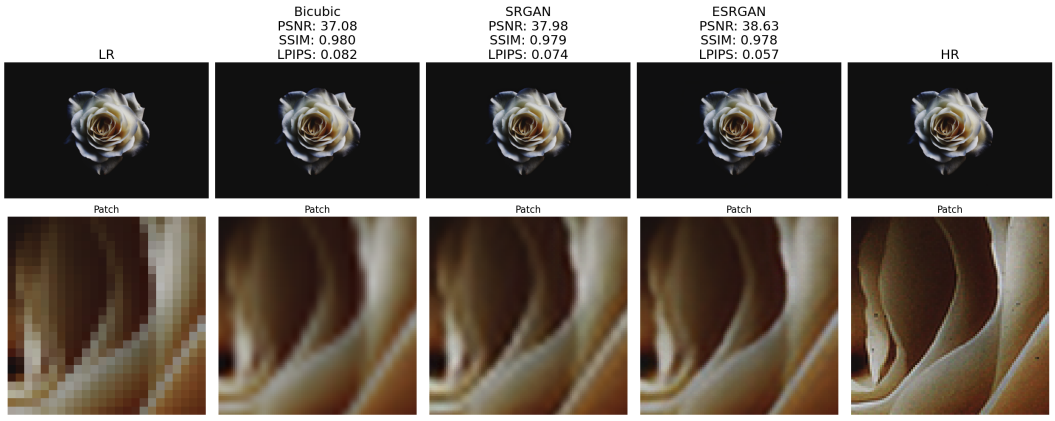

Figure 5: SRGAN/ESRGAN Results on image 0843 with scale factor 4.

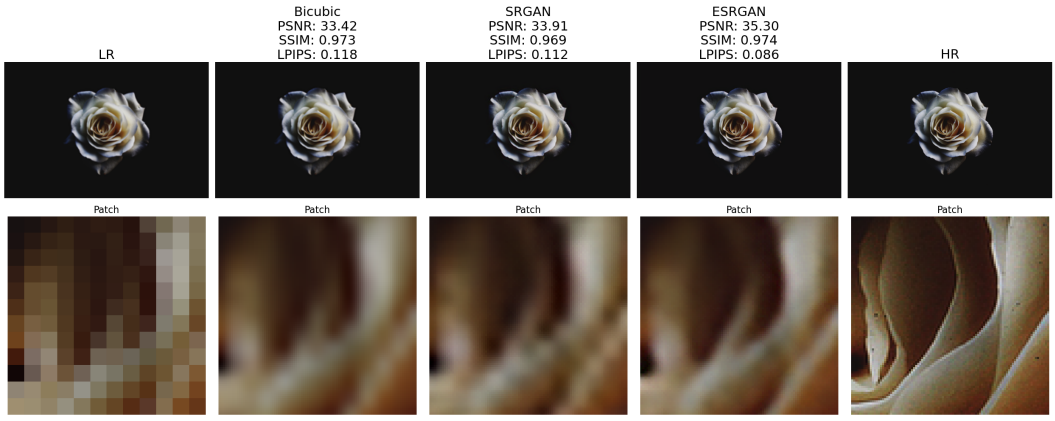

Figure 6: SRGAN/ESRGAN Results on image 0843 with scale factor 8.

lower LPIPS (indicating better perceptual similarity) than the SRGAN. For example, at an $\times 8$ scale in Figure 6, ESRGAN's PSNR increases by approximately 1-2 dB compared to the SRGAN while maintaining high SSIM and low LPIPS. In some cases, the GAN models have slightly worse quantitative metrics, but in every case when examining the patches, both the SRGAN and ESRGAN reconstruct sharper edges and more natural color transitions, with the ESRGAN outperforming the SRGAN.

Overall, the results demonstrate that the ESRGAN consistently outperforms SRGAN and Bicubic interpolation at multiple scale factors in terms of both perceptual quality and quantitative metrics. In particular, at higher upscaling factors like $\times 8$, the benefits of ESRGAN become more pronounced, as shown by the increased clarity and reduced artifacts in the patch regions.

4.4 Summary

For the task of image-superresolution, ESRGAN provides the best qualitative and quantitative results with the SRGAN following closely behind. The SRCNN, while computationally efficient, provides only decent results, not any better than bicubic interpolation. For all models, the lower the scale factor the better the results.

5 Supplementary Material

Figure 7 provides a comparison of Bicubic, SRGAN, and ESRGAN on another image. Figure 8 provides a comparison of Bicubic, SRGAN, and ESRGAN at scale 8 on an expanded image to see the differences in the models. Figure 9 provides the results for the SRCNN_{v1} on an image.

Codebase can be found at this Github Repo.

Presentation Video can be found here.

References

[1] Dong, C., Loy, C., He, K. and Tang, X. (2014). *Image Super-Resolution Using Deep Convolutional Networks*. In European Conference on Computer Vision (ECCV).

[2] Ledig, C., Theis, L., Huszár, F., Caballero, J., Aitken, A. P., Tejani, A., Totz, J., Wang, Z. and Shi, W. (2017). *Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network*. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[3] Wang, X., Yu, K., Wu, S., Gu, C., Liu, Y., Dong, C., Qiao, Y. and Change Loy, C. (2018). *ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks*. In European Conference on Computer Vision (ECCV) Workshops.

[4] Agustsson, E. and Timofte, R. (2017, July). NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

[5] Timofte, R., Agustsson, E., Van Gool, L., Yang, M.-H., Zhang, L., Lim, B., & others. (2017, July). NTIRE 2017 Challenge on Single Image Super-Resolution: Methods and Results. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

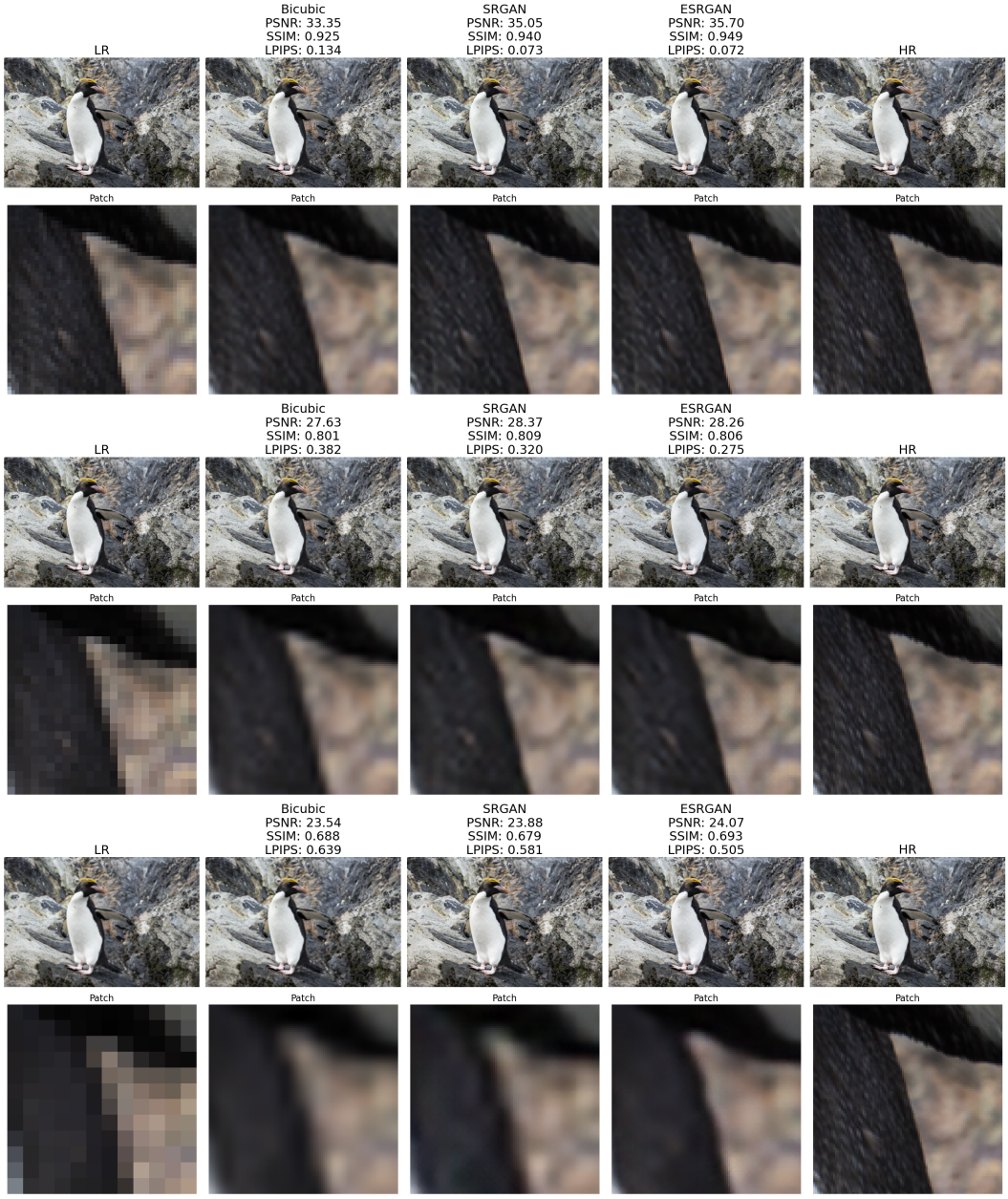

Figure 7: SRGAN/ESRGAN Results on image 0801 for scale factors from top down of 2, 4, and 8.

Figure 8: SRGAN/ESRGAN Results on image 0847 for scale factor 8.

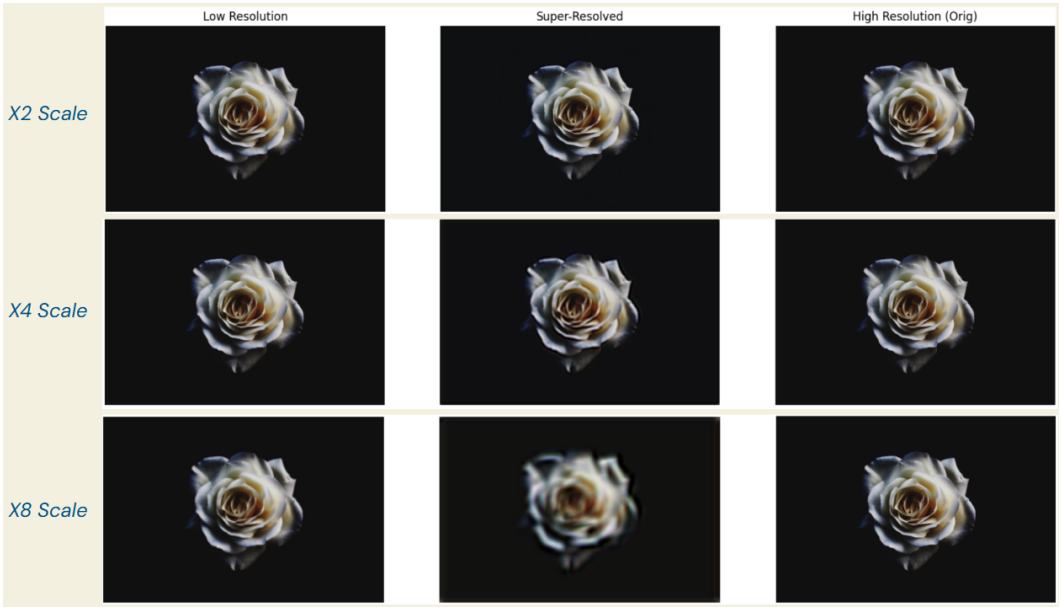

Figure 9: SRCNN_v1 Results on image 0843 for scale factor 2,4,8