Single Image Super Resolution with GANs

Dacheng Wang
Heinz College of Information Systems and Public Policy
Carnegie Mellon University
Pittsburgh, PA 15213
dacheng3@andrew.cmu.edu

Shimin Zhang
Heinz College of Information Systems and Public Policy
Carnegie Mellon University
Pittsburgh, PA 15213
shiminz@andrew.cmu.edu

Abstract

In this article, we work on the problem of Single Image Super Resolution, where the aim is to recover a high resolution image from the corresponding low resolution image. Super Resolution is a classic Image Processing problem with a wide range of application such as surveillance and medical diagnosis. In this reports, we will discuss the related works around this area in recent years, and the challenges for image super-resolution. Moreover, an exhaustive comparison among different implementations has been presented. The article also proposes a few experiments in attempt to improve the implementation. Lastly, the article discusses the current obstacles for future works.

1 Introduction

The pursuit of higher resolution has been a constant theme in both professional industry and consumer electronics segments in recent years. Not only does it help professionals including security and doctors to better perform their works, it enhances our enjoyment in digital contents from day to day.

The display technologies have been through a burst of breakthroughs in recent years, and high-definition displays (4K and 8K) have became the mainstream in TV, monitors, and even our phones. However, the amount of 4K and 8K contents are disproportional in comparison. Not only is it expensive to record/capture high-definition contents, the infrastructure required (networking bandwidth for streaming, etc.) can become burdensome to both consumer and content suppliers. As a result, an alternative solution is required: one requires minimum amount of changes in the hardware and infrastructure, but could deliver a better digital experience to the consumers.

Super-resolution (SR) naturally becomes the focus in this area. The ability to transform low-resolution (LR) contents to high-resolution (HR) contents is more valuable than ever. This article will mainly focus on the first step: Single Image Super Resolution (SISR).

2 Literature review

Early approaches to SISR include interpolation-based and prediction-based methods[4]. However, these approaches oversimplify the SISR problem since the generated images contain much fewer details comparing to the high resolution images. Later methods such as example-based methods often designed to suit only domain specific tasks [2].

The recent deep learning-based methods can provide superior accuracy. The Super-Resolution Convolutional Neural Network (SRCNN) surpasses the bicubic baseline with just a few training iterations and outperforms the sparse-code-based methods (SC) [14], a representative external example-based SR method [2]. SRCNN directly learns an end-to-end mapping, which is represented as a deep CNN, between the low/high-resolution images. Kim et al. [6] present a deeply-recursive convolutional network (DRCN), which has a very deep recursive layer that can improve performance without introducing new parameters for additional convolutions. Lim et al. [8] further deploy an enhanced deep super-resolution network (EDSR) and a multi-scale deep super-resolution system with performance exceeding state-of-the-art PSNR-based methods by removing unnecessary batchnorm layers.

Nevertheless, PSNR-based generated images appear overly smooth since minimizing MSE leads to finding pixel-wise averages. Perceptual-driven approaches and GAN-based models, on the other hand, are able to provide more visually-appealing results. SRGAN, a generative adversarial network (GAN) for image superresolution (SR), recover photo-realistic textures from heavily downsampled images [7]. SRGAN uses a perceptual loss function which consist of a adversarial loss and a content loss. The content loss is motivated by perceptual similarity instead of pixel similarity, and the adversarial loss pushes the result to the natural image manifold.

To further improve the image quality, Wang et al. [11] derives an Enhanced SRGAN (ESRGAN). They introduce the Residual-in-Residual Dense Block without batchnorm as the basic block and utilize relativistic GAN [5] for discriminator. In addition, they modified the perceptual loss by using features before activation so that the picture brightness and texture recovery are more consistent.

3 Dataset

We use the DIV2K dataset for training [1], and focus on x4 upscaling. For test data, we use the given large_test dataset.

4 Evaluation metrics

To evaluation the model performance, we calculated peak signal-to-noise ratio (PSNR) and structural index similarity (SSIM) between our generated super resolution images and their corresponding high resolution images. These two metrics are widely used in image quality assessment. All the PSNR and SSIM mentioned in this report are the results we get on test data.

4.1 Peak signal-to-noise ratio (PSNR)

PSNR can be most easily defind via the mean squred error (MSE) between two images. Given two $m \times n$ monochrome image, MSE between image *x* and image *y* can be defined as:

$$MSE = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (x_{ij} - y_{ij})^2$$

m	number of rows in image
n	number of columns in image
x_{ij}	pixel value from image x
y_{ij}	pixel value from image y

$$PSNR(x, y) = \frac{10 \log_{10}[\max(\max(x), \max(y))]^2}{MSE}$$

In our case, the data are color images with three RGB values per pixel. The definition of PSNR is the same, except that the MSE is the sum over all squared value differences for each color divided by image size ($m \times n$) and by three [13].

PSRN is commonly used to measure the quality of image reconstruction. Higher PSNR generally means higher quality of reconstruction. However, PSNR can be a measuring tool that has poor

performance when used to predict image fidelity and quality. For instance, the value of PSNR will not change if the the signs of the error change. In Figure 1, an original image (left) is distorted by adding a positive constant (top right) and by adding the same constant, but with random signs (bottom right). The PSNR/MSE between the original and any of the right images are the same, but the right images exhibit drastically different visual distortions. [12].

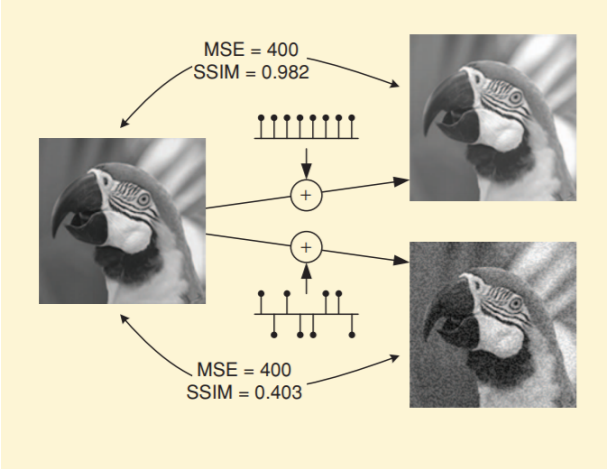


Figure 1: An original image (left) is distorted by adding a positive constant (top right) and by adding the same constant with random signs. [12]

4.2 Structural index similarity (SSIM)

SSIM is a newer measurement tool that is designed based on three factors i.e. luminance, contrast, and structure to better suit the human visual system [10]. The formula of SSIM is shown in Figure 2. Suppose that x and y are local image patches (taken from the same location) of two images. The local SSIM index measures the similarities between the two patches. Specifically, the similarity $l(\mathrm{x}, \mathrm{y})$ of luminances, the similarity $c(\mathrm{x}, \mathrm{y})$ of contrasts, and the similarity $s(\mathrm{x}, \mathrm{y})$ of structures.

$$S(\mathrm{x}, \mathrm{y}) = l(\mathrm{x}, \mathrm{y}) \cdot c(\mathrm{x}, \mathrm{y}) \cdot s(\mathrm{x}, \mathrm{y}) = \left(\frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \right) \cdot \left(\frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \right) \cdot \left(\frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \right),$$

Figure 2: SSIM formula [12].

In the formula, μ_x and μ_y are (respectively) the local sample means of x and y, σ_x and σ_y are (respectively) the local sample standard deviations of x and y, and σ_{xy} is the sample cross correlation of x and y after removing their means [12]. The items C1, C2, and C3 are small positive constants that serves as bias terms. To compute the SSIM value for the whole image, a sliding window is applied to first compute the local patch SSIM then average the SSIM values across the image. The range of SSIM values is between -1 and 1, and it equals 1 if the two images are identical.

Although SSIM might be a better metrics than PSNR, it has its drawback. It is very sensitive to geometric modifications, such as spatial scaling, spatial shift and rotation. This is contradictory to the fact that SSIM measures structural distortion since geometric modifications are oftentimes non-structural distortion. Shown in Figure 3 - 5, the SSIM of Figure 4 - 5 is very low, though there is no structural distortion.

In conclusion, PSNR and SSIM are both not good at predicting human visual response to image quality, but these two metrics are always reported in SR papers and can be served as benchmarks. Thus, we still use PSNR and SSIM to measure the model performance, but we also manually judge the reconstruction to ensure quality.

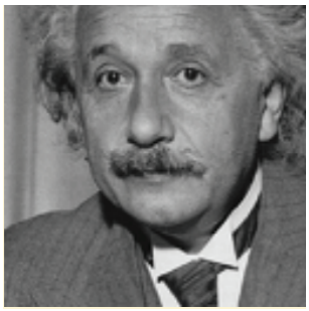


Figure 3: Original Image (SSIM = 1) [12].

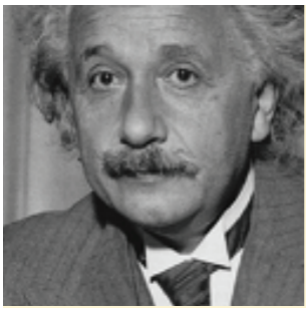


Figure 4: Spatial Shift (SSIM = 0.399) [12].

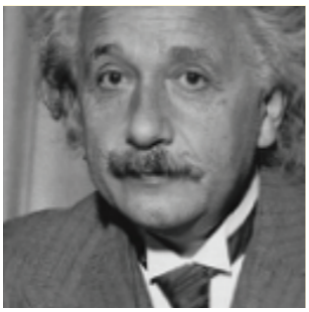


Figure 5: Rotation (SSIM = 0.551) [12].

5 Baselines

There are a total of three models we use as baselines: Autoencoder, SRGAN, and ESRGAN. While comparing the performance and results among them, we keep everything outside of the architecture the same (data, hyper-parameters, number of epochs, etc.) to ensure control. In the section below, we will discuss about the observation for each of the baseline model, and then compare the best outputs to determine the baseline model for experiments and enhancements.

5.1 Autoencoder

5.1.1 Description

The Autoencoder we build is based on Deep-Learning Super-resolution Image Reconstruction (DSIR) proposed by Duarte [3]. We modified the re-scaling factor and number of input and output channels to match our data. The model has a 3-layer Encoder and a 4-layer Decoder. Each Encoder layer consists of a convolutional layer, a batchnorm layer, a relu activation, and a maxpooling layer. Each Decoder layer consists of a transposed convolutional layer, a batchnorm layer and a relu activation. The overall model structure is shown in Figure 6.

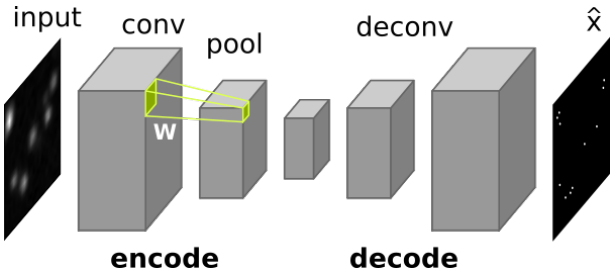


Figure 6: Autoencoder structure [3].

5.1.2 Observation

We trained the Autoencoder for 50 epochs. After 50 epochs, it achieves a surprisingly high PSNR of 21.50 and SSIM of 0.53. Since the two metrics are not objective when judging image quality, we manually look at the images and find the reconstruction quality to be poor (see Figure 8b). This further demonstrates that PSNR and SSIM are not always reliable.

5.2 SRGAN

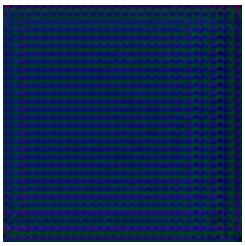
5.2.1 Description

Basing on the SRGAN paper from Ledig in 2017 [7], we implement an almost identical architecture as our baseline. The only two modifications are the re-scaling factor and replacing the output layer

from Dense-Leakly ReLU-Dense-Sigmoid to a convolutional layer. We found those two modifications serve our dataset better with identical performance to the original paper.

5.2.2 Observation

Our SRGAN model performs better than the Autoencoder. After 26 epochs of training, it yields a PSNR of 18.37 and SSIM of 0.48. Albeit the metrics seems to be lower than the Autoencoder, we found the result images have more details (Figure 8c). This will be explored more in Sec. 5.4. However, we find out it learns slower than the Autoencoder for the first few epochs, and it shows the BN artifacts as the original paper proposed. After extensive training, the artifacts become rare but not eliminated (see Figure 7b)



(a) SRGAN at Epoch 2

(b) SRGAN at Epoch 10 - shows BN artifact

Figure 7: SRGAN Issues.

5.3 ESRGAN

5.3.1 Description

The main benefits of ESRGAN over SRGAN will be explored in Sec. 6. We use the same architecture implemented in the ESRGAN paper [11] with minimal changes including re-scaling to meet our dataset requirement.

5.3.2 Observation

Overall, ESRGAN performs the best comparing to the other two baseline models both in convergence speed and final result (Figure 8d), and ends up becoming our pick of baseline for further experiments. With various training stage epoch allocation setup, the PSNR/SSIM stabilizes around 20.3 PSNR and 0.51 SSIM after about 30 epochs (see Sec. 7.1.2 for details).

5.4 Comparison Among Baselines

We see a drastically different result when comparing the metrics and the generated images. Table 1 indicates that the Autoencoder baseline performans the best. However, looking at the SR images comparison (Figure 8), it is easy to conclude that ESRGAN > SRGAN > Autoencoder.

Table 1: Metrics Comparison Among Baselines

Baseline	PSNR	SSIM
Autoencoder	**21.50**	**0.53**
SRGAN	18.37	0.48
ESRGAN	20.30	0.51

6 Model description

The majority of experiments and results are built upon the state-of-the-art ESRGAN model from Wang et al. [11], which is derived from SRGAN [7]. In this section, we will briefly discuss the enhancements ESRGAN implemented comparing to the original SRGAN, which can be summarized

(a) Original LR Image

(b) Autoencoder SR Image

(c) SRGAN SR Image

(d) ESRGAN SR Image

(e) Ground Truth HR Image

Figure 8: Baselines Comparison.

into two main improvements: improvement in residual blocks design and improvement in loss calculation.

6.1 Improvement In Residual Blocks Design

One of the major advantages of ESRGAN comparing to SRGAN is that the generated images are less likely to have the artifacts (see Figure 9). This is mainly due to the removal of batch normalization layers within the residual blocks. Another main difference within the design of residual blocks in ESRGAN is the usage of dense block. Similar to the motivation of DenseNets, the dense connections helps ESRGAN to achieve a higher network capacity.

6.2 Improvement In Loss Calculation

In the original SRGAN implementation, the discriminator is tasked with estimating the probability of an input image being real [7]. The authors of ESRGAN introduced a different way of thinking: instead of only using the generated results, we can estimate how much is the real image (label) relatively more realistic than the generated image. The benefit of this Relativistic average Discriminator RaD is that it not only uses the generated data but also the real data within the adversarial training [11]. This leads to sharper image overall (Figure 9)

In addition, Wang et al. also improved upon the calculation of perceptual loss. In SRGAN, the perceptual loss is composed as the weighted sum of the content loss and the adversarial loss where the content loss is simply an MSE loss (pixel-wise error between high resolution image and super resolution image. The adversarial component is a VGG loss (euclidean distance between the feature representations of reconstructed image and the reference image) [7].

SRGAN uses results after the activation layer from a pre-trained VGG19 model [7]. Instead, ESRGAN uses results before the activation layer to counter two shortcomings from the SRGAN's design:

- Post-activation features tend to be sparse, which leads to weaker supervision and worse performance in a deep neural network.
- The reconstructed image using post-activation features tend to have different brightness from the ground truth. This will be discussed in more details in later section where we compare results from the three baseline models (Autoencoder, SRGAN, and ESRGAN).

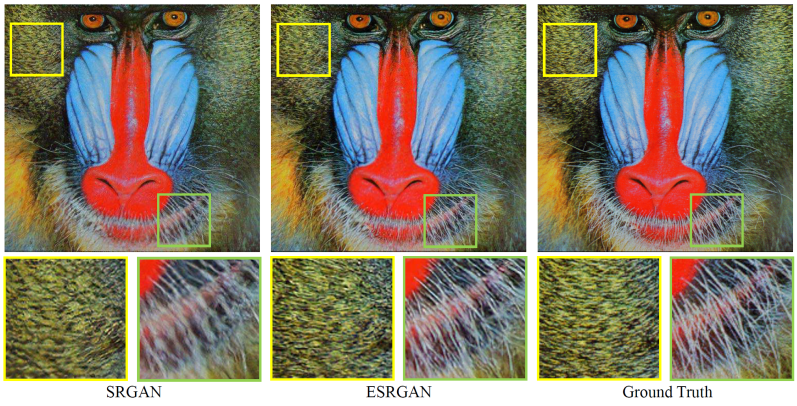


Figure 9: Comparison between the super-resolution results between SRGAN and ESRGAN. We can see that ESRGAN's result is sharper with less artifacts. [11]

7 Experiments and results

We have mainly done two types of experiments: experiments with data preprocessing and training methods, and experiments with model architecture. All the experiments have been done with the ESRGAN model.

7.1 Data/training Focused Experiments

7.1.1 Crop vs. No Crop

In the original SRGAN paper [7], the authors proposed cropping each image into 16 random 96 x 96 sub images for each mini-batch during training. In our initial attempt for both SRGAN and ESRGAN, we follow this procedure and are generating 32 x 32 sub images for both inputs and labels. This means our 800 training images become 121,856 sub images (with batch size of 16, that is 7,616 batches in total). Even though this method performs extremely well in validation (Figure 10), it does not perform as well with the testing dataset (Figure 11a), where the input image resolution is 1/16 of the training input image resolution. To overcome this problem, we decide to scale down both the training input and label images 4 times to match the image size of testing data, and train one image at a time (800 data point per epoch). Comparing to the over-sharpening images generated from ESRGAN with cropping, the images from ESRGAN without cropping looks more natural (Figure 11b). In addition, since we decrease our batch number per epoch from 7,616 to 800, our training time per epoch decrease from two hours to 8 mins.

Figure 10: ESRGAN With Cropping Validation. Hardly distinguishable between SR (left) and HR (right).

(a) ESRGAN With Cropping Testing: over-sharpening

(b) ESRGAN With Scaling-down Testing: more natural

Figure 11: Cropping vs. Scaling-down.

7.1.2 Training Stage Epoch Allocation

In the ESRGAN paper, the training process is divided into two stages. The authors first train a PSNR-oriented model with the L1 loss and then employ the trained PNSR-oriented model as the initialization for the generator for fine-tuning. However, the paper does not specify what a good ratio between training time (number of epochs) for stage 1 and stage 2 would be.

We experiment with various ratio between the two training stages and find that the best ratio is shown below:

$$\frac{numEpochs\_Stage1}{numEpochs\_Stage2} \approx 2.5 \tag{1}$$

When *numEpochs_Stage1* = 20, we achieve the best PSNR at epoch 28 (the 8th epoch of Stage 2). When *numEpochs_Stage1* = 30, we achieve the best PSNR at epoch 43 (the 13th epoch of Stage 2). When *numEpochs_Stage1* = 100, we achieve the best PSNR/SSIM at epoch 140 (the 40th epoch of Stage 2). See Table 3.

Table 2: Training Stage Epoch Allocation Experiment

numEpochs_Stage1 = 20		
Stage1	Stage2	PSNR/SSIM
20 epochs	5 epochs	18.15/0.44
20 epochs	8 epochs	**19.78/0.46**
20 epochs	20 epochs	19.57/0.47
numEpochs_Stage1 = 30		
Stage1	Stage2	PSNR/SSIM
30 epochs	5 epochs	19.25/0.45
30 epochs	13 epochs	**20.08/0.51**
30 epochs	30 epochs	19.82/0.50
numEpochs_Stage1 = 100		
Stage1	Stage2	PSNR/SSIM
100 epochs	20 epochs	19.32/0.47
100 epochs	40 epochs	**20.47/0.51**
100 epochs	60 epochs	20.11/0.50
100 epochs	80 epochs	19.87/0.48

7.2 Model Focused Experiments

7.2.1 Gaussian Noise

Inspired by Rakotonirina et al. [9], we add noise to the generator, which is an approach that is used in human face generation that based on GANs. Gaussian noise is added to the output of each residual dense block of ESRGAN with a learned scaling factor γ, as Figure 12 shows. The stochastic variation only has localized effect, leaving the overall high-level aspects of the image intact. The generated images have more natural textures due to generalization. See Figure 13, 14, 15. The test PSNR/SSIM improves from 18.098/0.443 to 19.861/0.493 with 10 epochs for Stage 1 training and 10 epochs for Stage 2 training. The experiment is done with the training data with no cropping.

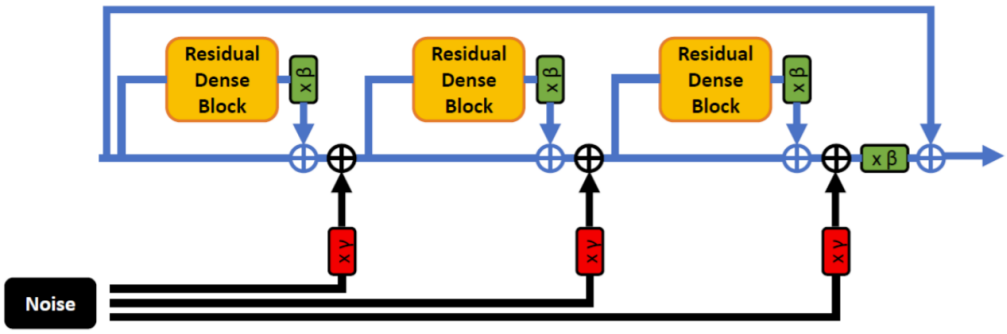


Figure 12: Gaussian noise is added after each residual block. [9]

Figure 13: Without Noise (18.098/0.443)

Figure 14: With Noise (19.861/0.493)

Figure 15: Ground Truth of Picture 43074 from Large Test Set

8 Conclusion and discussion

By comparing the three state-of-the-art models (AE, SRGAN, and ESRGAN), we learned more about each of them and how they would behave differently with similar setting in data and hyperparameters. Unsurprisingly, ESRGAN outperforms the other two baseline models. To further improve the performance of ESRGAN, a variety of techniques were introduced including re-scaling, stage 1 / stage 2 allocation, and adding Gaussian noise. All three have improved the model in certain ways and should be explored further in the future.

9 Future work

With the three main experiments conducted in this article, we would like to know if the improvement can be generalized outside of the setting and dataset we have selected. Furthermore, there could be interconnection between them. Can an allocation of more stage 1 (psnr-oriented) training in combination of less Gaussian Noise outperform an allocation of more stage 2 training in combination of more Gaussian Noise, given stage-2 training tends to over-sharp the generated image?

In addition, the main obstacle in SISR, finding a deterministic evaluation metric, has not yet been overcame. This article discusses that both PSNR and SSIM, the two main metrics used in this area, do not necessarily align with what human perceive as clear images. We would like to revisit some sections including the experiments and comparison in this article if a better metric comes out in the future.

10 Division of work

Dacheng Wang: ESRGAN baseline model training, SRGAN baseline model training, metrics calculation, experiments with data preprocessing, report writing.

Shimin Zhang: ESRGAN baseline model training, Autoencoder baseline model training, experiments with training stage, experiments with gaussian noise, report writing.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1122–1131, 2017.

[2] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014.

[3] Alex Duarte. Deep-learning super-resolution image reconstruction (dsir). `https://github.com/leaxp/Deep-Learning-Super-Resolution-Image-Reconstruction-DSIR/blob/master/LICENSE`, 2018.

[4] Claude E Duchon. Lanczos filtering in one and two dimensions. *Journal of Applied Meteorology and Climatology*, 18(8):1016–1022, 1979.

[5] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard gan. *arXiv preprint arXiv:1807.00734*, 2018.

[6] Jiwon Kim, Jung Kwon Lee, and Kyoung 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.

[7] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[8] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[9] Nathanaël Carraz Rakotonirina and Andry Rasoanaivo. Esrgan+: Further improving enhanced super-resolution generative adversarial network. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3637–3641. IEEE, 2020.

[10] De Rosal Igantius Moses Setiadi. Psnr vs ssim: imperceptibility quality assessment for image steganography. *Multimedia tools and applications*, 80(6):8423–8444, 2021.

[11] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[12] Zhou Wang and A.C Bovik. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE signal processing magazine*, 26(1):98–117, 2009.

[13] Wikipedia contributors. Peak signal-to-noise ratio — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Peak_signal-to-noise_ratio&oldid=1021991316, 2021. [Online; accessed 11-May-2021].

[14] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE transactions on image processing*, 19(11):2861–2873, 2010.

A Appendix

Table 3: Model Performance of the Model Submitted on Canvas

Stage1	Stage2	PSNR/SSIM
100 epochs	20 epochs	19.32/0.47
100 epochs	40 epochs	**20.47/0.51**
100 epochs	60 epochs	20.11/0.50
100 epochs	80 epochs	19.87/0.48

*This model is a ESRGAN model, trained with data without cropping