# Single Image Super Resolution: An Analysis

GitHub(https://github.com/aba450/Super-Resolution)
Ammar bin Ayaz (aba450), Umar Farooq (uf247)

## 1 INTRODUCTION

Image super-resolution (SR), which refers to the process of recovering high- resolution (HR) images from low-resolution (LR) images, is an important class of image processing techniques in computer vision. In general, this problem is very challenging and inherently ill posed since there are always multiple HR images for a single LR image but with the rapid development of deep learning techniques, Super-resolution models based on Deep learning have been extensively explored and they often achieve state-of-the- art performance on different benchmarks of Super-Resolution.

## 2 MODELS

**SRCNN —** A Convolution neural network based approach of super resolving a low resolution image.

**SRGAN/SRResNet —** A residual block based approach with variations of using a GAN based architecture and a NoGAN based one.

**ESRGAN —** A variation in the architecture of SRGAN with introduction of Residual-in- Residual Dense blocks and relative discriminator.

## 3 LITERATURE REVIEW

### 3.1 SRCNN

Prior to deep neural networks to solve the problem of Super Resolution (SR) many external example-based methods [1], [2] were used. These methods firstly, densely crop the over-lapping patches from the input or the pre-processed image and are then encoded by a low-resolution dictionary. The sparse coefficients are passed into a high-resolution dictionary for reconstructing high resolution patches. The overlapping reconstructed patches are aggregated to produce the final output. This pipeline is equivalent to a deep convolutional neural network.

Convolutional neural networks are a class of deep neural networks that have shown explosive popularity partly due to their success in computer vision fields like image classification, face recognition and object detection. This is mainly due to their shift

invariance property based on their shared weights architecture and translation invariance characteristics.

Motivated by the above facts, the authors in [3] proposed a Super Resolution Convolutional Neural Network (SRCNN) that directly learns an end-to-end mapping between low- and high-resolution images. This model fundamentally differs from the example-based approaches as the dictionaries [1], [2] for modeling the patch space are learnt implicitly by the hidden layers. Patch extraction and aggregation are formulated as convolutional layers and the entire SR pipeline is achieved by learning.

SRCNN is a shallow deep learning model with only three layers. Layers used in the network are described below:

- Patch Extraction and representation: This layer extracts over-lapping patches from the low- resolution image and upscales it to the desired size using bicubic interpolation i.e., represents each patch as a high dimensional vector. These vectors comprise a set of feature maps.

- Non-linear mapping: This layer performs non-linear mapping of the high dimensional vectors(n1) into other high dimensional vectors(n2) where n1 > n2, something like PCA but in a non-linear way. Each mapped vector represents the high-resolution patch.

- Reconstruction: After the mapping, the high-resolution image is reconstructed by patch-wise aggregation of high-resolution patches. This reconstructed image is expected to be similar to the ground truth image.

### 3.2 SRGAN/SRResNet

SRGAN proposes an adversarial objective function that promotes super resolved image that lie close to the manifold of natural images.
The main highlight of the approach taken is multi-task loss formulation that consists of three main parts: 1) a MSE loss that encodes pixel wise similarity, 2) a perceptual similarity metric in terms of distance metric defined over high-level image representation, and 3) an adversarial loss that balances a min-max game between a generator and a discriminator.

#### 3.2.1 Architecture

Take the High Resolution images which is only available during training,  apply Gaussian filters to the HR images followed by a down sampling operation with a down sampling factor r, An image with tensor dimensions of W*H*C is downsampled to rW * rH * C. The end goal is to train a generator which estimates a high resolution image from its low resolution counter part.

The Architecture is based on the GAN approach where a discriminator network is trained which is optimized in an alternating manner with the generator network. With the approach we try to generate images which are almost same as real images and make it difficult for the discriminator to classify. This encourages a perceptually superior image residing in the manifold of the natural images. This kind of approach is totally different from the other SR solutions where they try to minimize the pixel wise measurement loss such as Mean Squared Error.

**Generator** Architecture
The core of the network is a number of residual blocks which have identical layout. There are two convolutional layers with small 3x3 kernels and 64 feature maps followed by batch normalization layers. The activation function used is ParamtericRELU. The architecture is depicted in the figure below.

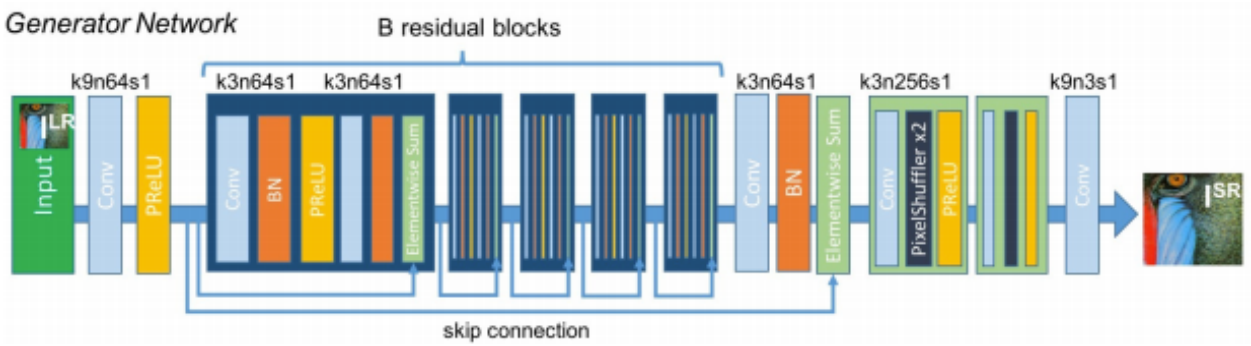



Fig 1 SRGAN generator

**Discriminator** Architecture
It contains eight convolutional layers with an increasing number of 3x3 kernels which is similar to a VGG network. The increment is by a factor of 2 from 64 to 512 kernels. Strides are used to reduce resolution when the number of features is doubles. The activation used in the beginning is leakyRELU and in the end the network has two dense layers with a final sigmoid activation function. The architecture is depicted in the figure below.

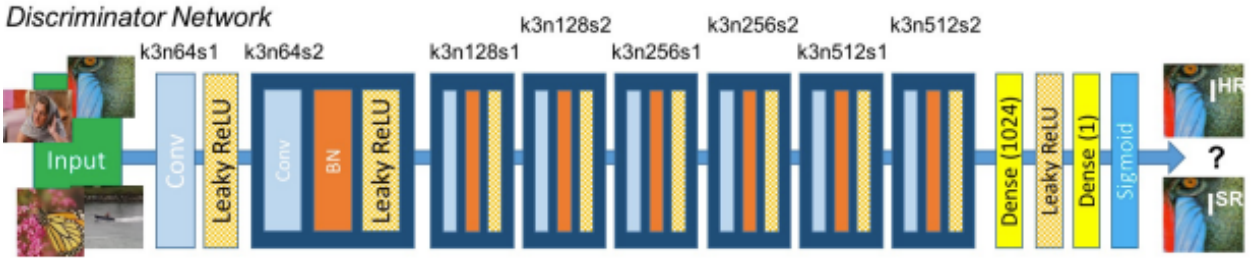



Fig 2 SRGAN discriminator

### 3.2.2 Loss Function(Perceptual)

One of the main focus of the approach is the definition of the perceptual loss function which played a major role in enhancing the performance of the generator network. The loss is a sum of two loss functions content loss and adversarial loss. The formula for the same is given below:

$$l^{SR} = \underbrace{\underbrace{l^{SR}_{\mathrm{X}}}_{\text{content loss}} + \underbrace{10^{-3} l^{SR}_{Gen}}_{\text{adversarial loss}}}_{\text{perceptual loss (for VGG based content losses)}}$$

Let's talk about content loss. The most commonly used loss function for SR is pixel wise MSE loss. However it lacks the frequency content which in turn results in perceptually unsatisfactory solutions. The loss used in the paper is closer to the perceptual similarity. It is basically VGG loss which is the euclidean distance between the feature representations of the a reconstructed image and the reference image as in HR image.

With this we also add the generative loss to the network to get the overall perceptual loss. The loss is defined based on the probabilities of the discriminator over all the training examples.

### 3.2.3 Key Takeaways

- New state of the art with SRResNet in terms of PSNR/SSIM ratings which is basically a no GAN version of the network proposed in [4] where it just uses the generator with MSE loss function.
- With the output of the images and MOS(Mean Opinion Score) , the paper [4] has highlighted some limitations of PSNR/SSIM metrics for evaluation.
- SRGAN architecture which uses a GAN architecture with residual blocks and augments the content loss with adversarial loss.
- One of the main things is that the MOS (Mean Opinion Score) scores of SRGAN for large up scaling factors is a new state of the art as compared to the reference methods mentioned in the paper [4].

## 3.3 ESRGAN

Single Image Super resolution has gained a lot of traction in the deep learning community recently. Once the pioneer work of SRCNN [3] was proposed, many other deep convolutional neural networks have brought prosperous development. The main focus of these approaches are to improve PSNR(Peak signal to noise ratio), but we have seen in the SRGAN [4] paper that this metric fundamentally disagrees with the subjective evaluation of human observers.

The amazing work implemented by SRGAN paper which proposes a GAN based approach and perceptual loss function encourages the network to favor the solutions that are more photo realistic or more like natural images. Having said that there is still some clear difference between the generated images and the natural images.

### 3.3.1 Architecture

**Generator** Architecture

The main changes in the structure of the generator of SRGAN are:

1) Remove all the Batch Normalization(BN) layers
2) Replace the original basic block with the proposed Residual-in-Residual Dense Block(RDDB).

Removing the BN layers made a huge impact as reported in the paper [5].
It increased performance and reduced computational complexity in different PSNR oriented tasks including super resolution and debarring. The reason for this can be that BN layers normalize the features using means and variance in a batch during training and use estimated mean and variance of the whole training set during testing. If there is lot of difference in the training and testing set, BN layers introduce unpleasant artifact and hamper generalization. One observation was that BN layers bring artifacts for deep networks and if it trained over GAN framework.

The high level architecture is similar to SRGAN [4], with changes like using RRDB dense block and removing BN layers. Both the architectures are depicted in the figure below.

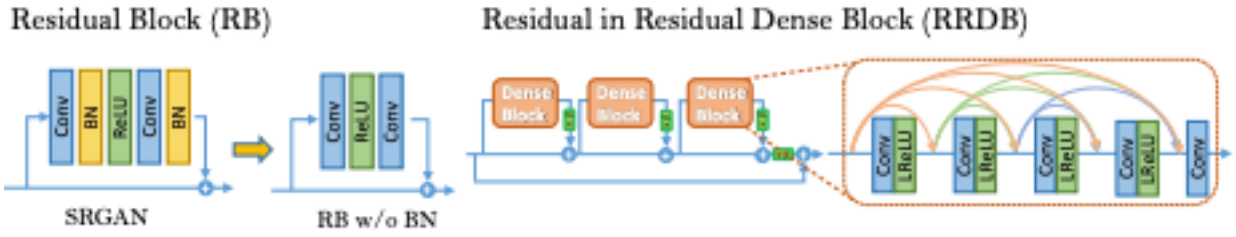
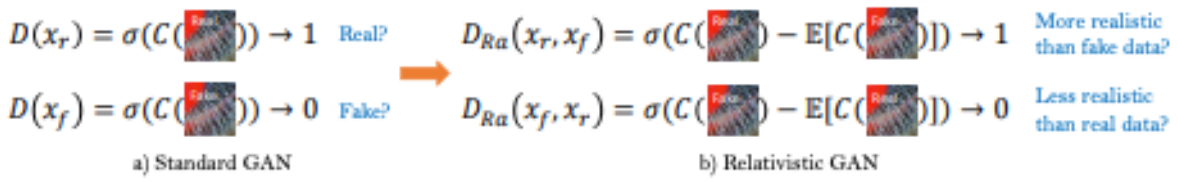



Fig 3 RRDB block ESRGAN

**Discriminator** Architecture

Apart from making changes in the generator architecture they [5] also changed the structure of the discriminator by introducing relativistic GAN, which focusses on identifying the relative difference between a realistic image and a fake one. The below figure depicts the difference of this from a standard discriminator.

Fig 4 Relative Discriminator ESRGAN



### 3.3.2 Perceptual Loss

Paper [5] proposes more effective perceptual loss by constraining on features before activation rather than after activation as practiced in SRGAN. The advantage of using the features before activation overcomes drawbacks in the original design like having sparse activated features in a very deep network, using features after activation also causes inconsistent reconstructed brightness compared with the ground truth image. Fine tuning the VGG network for material recognition also helped in identifying textures which is very critical in case of Super Resolution.

### 3.3.3 Network Interpolation

They [5] also did network interpolation which helped in removing unpleasant noise and keeping intact the perceptual quality of the image. To achieve this the approach is to first train the PSNR-oriented network and then obtain a GAN-based network by fine tuning. It helps in producing meaningful results and maintains the perceptual quality and fidelity without the need to retrain the model.

### 3.3.4 Key Takeaways

The results shown in the paper [5] present that ESRGAN model achieves consistently better perceptual quality than previous SR methods. Important features of the model are listed below:

1) Network structure introduces Residual-in-Residual Dense Block (RRDB), which has higher capacity and easier to train.
2) Removal of Batch normalization layers, and using residual scaling and smaller initialization to help training a very deep network.
3) Introduction of RaGAN (Relativistic average GAN), which focusses on identifying more realistic images, rather than if an image is fake or real.
4) Improvement in perceptual loss by using VGG features before activation, which was done after activation in SRGAN [4].

## 4 TRAINING DETAILS

### 4.1 SRCNN

- Training Dataset = T-91 image dataset
- Downsampling factor = 2x
- Epochs = 400 on the whole dataset
- Optimizer = Adam with rate 1e-4
- Batch Size = 16

### 4.2 SRGAN/SRResNet

- Batch Size = 16
- Number of batch iterations = 150,000 (SRResNet) 15,000(SRGAN)
- Target Size = (96 * 96)
- Optimizer = Adam with rate of 1e-4
- Downsampling factor = 4x
- Training Dataset = Flickr8k

### 4.3 ESRGAN

- Batch Size = 16
- Number of batch iterations = 15,000
- Optimizer = Adam with rate 1e-4
- Downsampling factor = 4x
- Training Dataset = DIV2K LR and HR images
- Target Size = (128 * 128)

## 5 RESULTS

The table depicted in figure 6 does a comparative analysis of the PSNR scores for different models we have explored. The numbers given in small brackets are the PSNR scores reported in the corresponding papers. In case of ESRGAN I have used the inference [] of the author of the paper [5] and the scores have improved with some fine tuning done by the authors of the paper [5].
From the table we can see that the psnr score of SRGAN and SRResNet [4] are low in our training and the reason for that is that we have used a smaller dataset and trained for less number of epochs due to the limitations of google collar. But we have seen an increasing trend as we increase the iterations and same goes for the size of data.
One thing that we have noticed is that although the scores for SRCNN or SRResNet are comparable with GAN based models but the perceptual quality of the images are much better for SRGAN and ESRGAN. This has been reported in both the papers [4] [5]. They have also reported MOS(Mean Opinion Score) which was beyond the scope of this project.
The difference in the perceptual quality can be seen in the pictures we are sharing below in figure 5. We have added all the output images for Set5 and Set14 images in git repository. In the output images we can see that the best images is generated using ESRGAN model. SRCNN also does a good job considering the size of the training dataset, training time and complexity of the model.

Input image

ESRGAN output

Ground-truth image

SRGAN

SRCNN

Input image

Ground-truth

SRCNN

SRGAN ESRGAN

Fig 5 Sample Output images for different models

| Method | Training dataset | Set5 | Set14 | BSD 100 |
|---|---|---|---|---|
| SRCNN (2x) | T-91 | 36.99 (36.66) | 32.34 (32.45) | 32.04 |
| SRCNN (4x) | T-91 | 31.80 (30.49) | 28.10 (27.50) | 28.15 |
| SRResNet (4x) | Flickr8k | 28.50 (32.05) | 25.52 (28.49) | 25.49 (27.58) |
| SRGAN (4x) | Flickr8k | 28.20 (29.40) | 25.33 (26.02) | 24.52 (25.16) |
| ESRGAN (4x) | DIV2K | 28.97 (32.73) | 28.28 (28.99) | 28.12 (27.85) |
| ESRGAN Pretrained (4x) | DIV2K | 32.48 | 31.21 | 30.35 |

Fig 6 PSNR scores for the models

We have also provided the functionality of uploading an image inside the notebook for esrgan model. Sample input and output image is depicted in the figure 7 below. The notebook for esrgan inference has this functionality.


Fig 7 Upload image esrgan

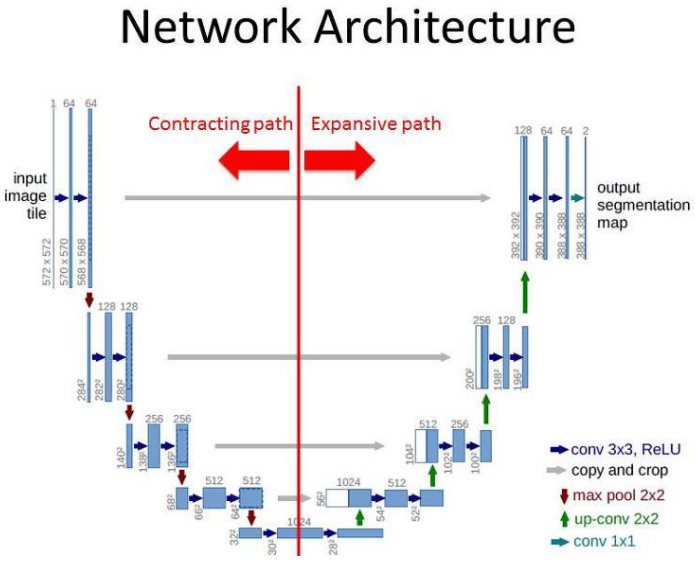
# 6 Additional Experiments

## 6.1 UNet+ResNet+FastAI

This technique was discussed in the Practical Deep learning course by fastai. The main components of the approach are listed below:

- UNet architecture for Super resolution of low resolution images. The name of the architecture says it all and is perfect for use in super resolution, most of the networks decrease the image size and increase the number of features, In UNet the encode-decode networks transforms the input into features and then tries to get back to its original form. The architecture is depicted in the figure below.

Fig 8 UNet Architecture



- Although the task of super resolution can be done by just using UNet, but in case of images we need a more comprehensive loss and not just look at the numerical loss. To achieve this we use ResNet feature loss.
- Fastai [10] library has been built on top of pytorch and provides multiple functionalities out of the box like finding learning rate, data augmentation etc.

### 6.1.1 Training Details

- Batch Size = 32
- Low resolution = (128 * 128)
- Upsampling factor = 4x
- Training Dataset = Oxford Pets
- Number of epochs = 20 on the whole dataset
- Learning rate = 1e-3
- Weight decay = 1e-3

### 6.1.2 Super Resolved Images

This technique [9] took much less training time and still produced some very fascinating super resolved images which are shown below. We will explore the technique in future to give a comparative analysis. But it looks very promising.

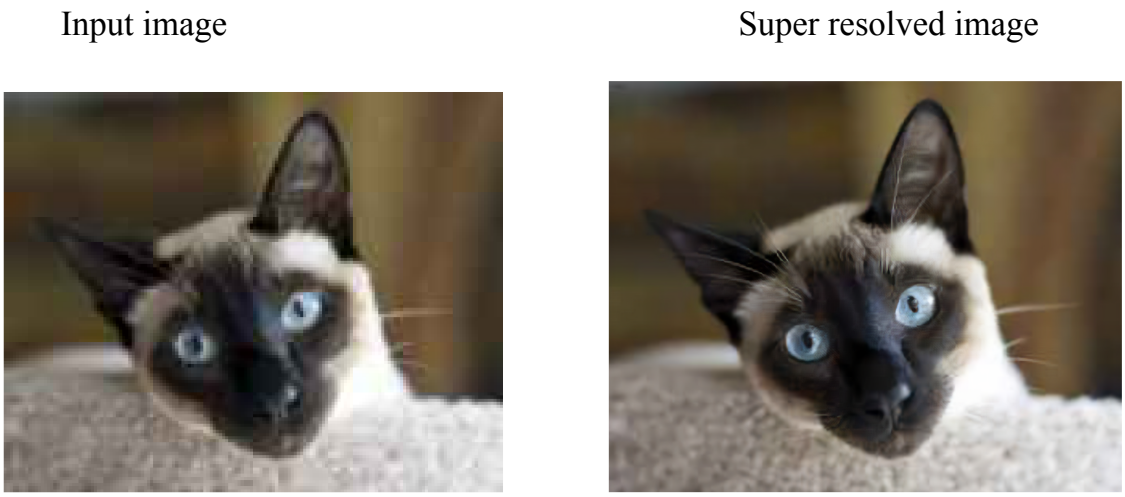



Fig 9 Super resolution unet+resnet+fastai

### 6.2 Super resolution WebApp

We have also added a basic **flask** based web app where we can upload our own images and then it would generate super resolved images. We are using srcnn weights to generate the output. The instructions to run the web app is provided in the separate readme of the super-webapp in the git repo. Sample look of the app is showcased in the figure below. The web app is closely based on a medium blog [8].

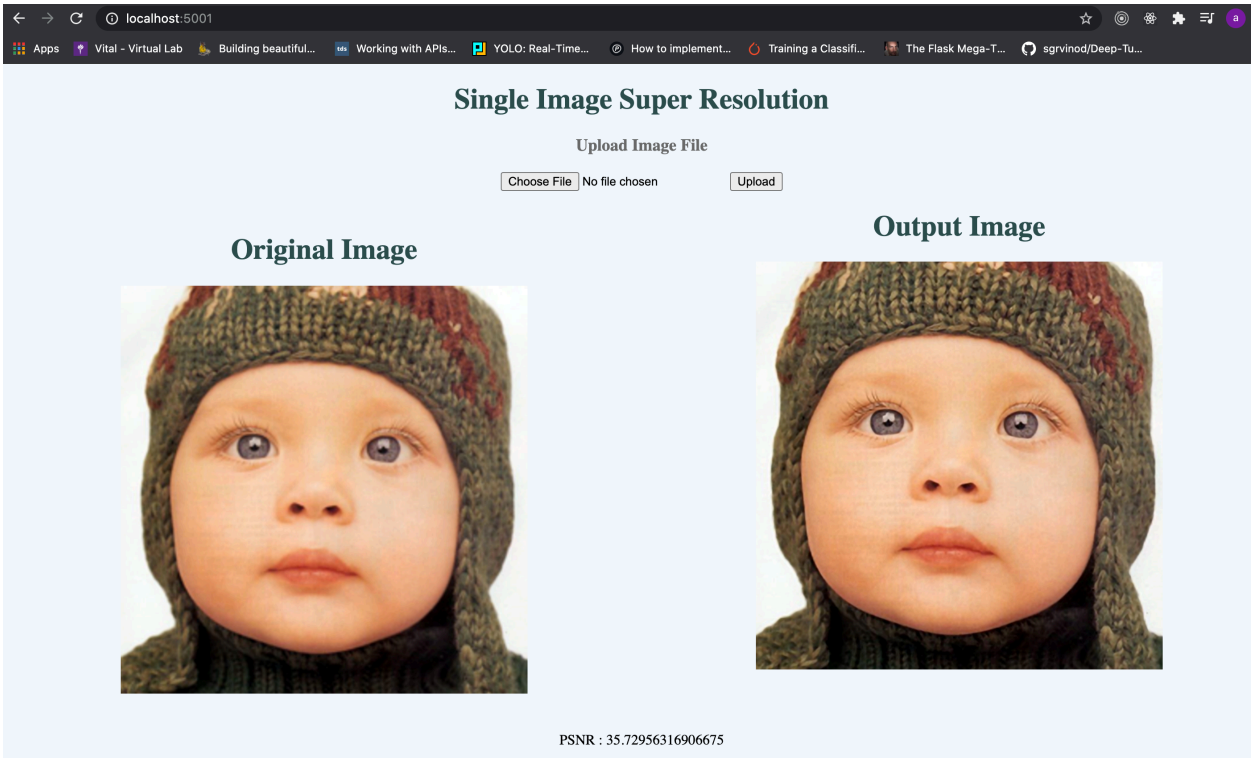



Fig 10 Super resolution web app

## 7 Conclusion

We have presented a comparative analysis of different category of models with the task of Super-Resolving a low resolution image. Looking at the output images generated we feel that ESRGAN performs the best in terms of the perceptual quality of the images generated. In future we would like to extend this analysis to videos, and explore new state of the art models in this domain. We are planning to go further and provide this as a one stop place where people can explore the domain of Super resolution in detail.

## 8 References


[1] Yang, J., Wright, J., Huang, T., Ma, Y.: Image super-resolution as sparse representation of raw image patches. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8 (2008)

[2] Yang, J., Wright, J., Huang, T.S., Ma, Y.: Image super-resolution via sparse representation. IEEE Transactions on Image Processing 19(11), 2861–2873 (2010)

[3] Chao Dong, Chen Change Loy, Kaiming He, Xiaoou Tang: Image Super-Resolution Using Deep Convolutional Networks. arXiv preprint arXiv:1501.00092 (2014)

[4] C. Ledig *et al*., "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 105-114, doi: 10.1109/CVPR.2017.19.

[5] Wang, Xintao & Yu, Ke & Wu, Shixiang & Gu, Jinjin & Liu, Yihao & Dong, Chao & Loy, Chen Change & Qiao, Yu & Tang, Xiaoou. (2018). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks.

[6] Xintao Wang, Ke Yu, Kelvin C.K. Chan and Chao Dong, & Chen Change Loy. (2020). BasicSR. https://github.com/xinntao/BasicSR.

[7] Julien Guillaumin https://github.com/JGuillaumin/SuperResGAN-keras

[8] Aanisha Bhattacharyya, https://medium.com/towards-artificial-intelligence/building-a-super-resolution-image-web-app-57e26886cb45

[9] Fast AI Course, https://github.com/fastai/course-v3/blob/master/nbs/dl1/lesson7-superres.ipynb

[10] Fast ai documentation, https://docs.fast.ai/