# SINGLE IMAGE SUPER RESOLUTION (SISR)

CO543 IMAGE PROCESSING - FINAL PROJECT REPORT


TEAM

E/16/078: THILINI DESHIKA

E/16/134: YASITHA HERATH

*Deaprtment of Computer Engineering*

*Unviersity of Peradeniya*


2021

# Contents

**Abstract**

Image Super-Resolution is a widely used digital image processing technique being applied in many areas including medical imaging, remote sensing and intelligent surveillance. Most of the time, a Low-Resolution image is captured from the originally observed High-Resolution image in order to achieve efficient transmission and storage. Super-Resolution is performed on the captured downgraded image before using it for the intended task. The captured downgraded image may be affected due to noise and lighting. Therefore, it is essential to recover the image from those issues when obtaining the Super-Resolution image. Image Super-Resolution can be defined as the process of reconstructing the High-Resolution images from a given set of Low-Resolution images while minimizing the ill conditions that occurred in image acquisition.

Deep Neural Network models show a significant performance in Super-Resolution. However, the performance of a neural network model depends on various factors including the model architecture, optimization methods used and performance metrics considered. For this project, the DIV2K dataset with downgrading factor bicubic_x4 is used. First, a pre trained Enhanced deep Residual Network (EDSR) on the DIV2K dataset considering the Mean Absolute Error (MAE) as the loss function and considering Peak Single to Noise Ratio (PSNR) and Structural Similarity Index (SSIM) values as the performance metrics is taken as the generator. Then, the result obtained from the EDSR model is fine-tuned using a Generative Adversarial Network for image Super Resolution (SRGAN) considering the Perceptual Loss as the performance metric.

# 1 Introduction

Image Super-Resolution (SR) is the task of obtaining a high-resolution image of a scene given a low-resolution image(s) of the scene. SR is a field being developed for decades from classical interpolation to the deep learning approach. SR has a variety of its applications and some of them are mentioned below.

- When obtaining MRI images, it can be difficult to capture a high-resolution image because of limitations in scan time and spatial coverage. SR can be used to obtain a high-resolution image from a captured low-resolution image.

- SR can be used in media file transmission, as sending a low-resolution file and then upscaling can reduce server costs.

- Face recognition can be performed on low-resolution images obtained from security cameras used in surveillance applications.

The relationship between a low-resolution image and its high-resolution image depends on the situation. There are many practical concerns that affect image downgrading including blur, noise and decimation. In this project, it is assumed that image downgrading has happened as bicubic downsampling of the high-resolution image and then the low-resolution image is obtained.

The DIV2K dataset obtained from TensorFlow datasets is used to perform the Super Resolution task. The aim of the proposed solution is to increase the image resolution by 4 times. Therefore, the DIV2K dataset with bicubic downgrading method by 4 times (div2k/bicubic_x4) is considered here.

A supervised learning approach is taken to perform SR on the DIV2K dataset with a known downgrading method of down-sampling with a scale of 4. The presented solution here is to reconstruct a high-resolution image from a single low-resolution image using Enhanced Deep Residual Networks for Single Image Super-Resolution (EDSR) [1] as the generator for SRGAN [2]. The EDSR model learns the end to end mapping between the high-resolution image and its corresponding downsampled low-resolution image. The mapping is represented as a deep Convolutional Neural Network (CNN) that will be able to output the upsampled high-resolution image for a given low-resolution image. Then the EDSR model is fine-tuned using a Generative Adversarial Network for image Super Resolution (SRGAN). [2].

# 2 Related Work

Super Resolution is a task evolving more than three decades because of its diversity in applications. Among them, some of the popular techniques used to perform the task are described below.

## 2.1 Pre-Upsampling Super Resolution

In Pre-Upsampling Super Resolution, bicubic interpolation methods are used along with a deep learning approach. SRCNN and VDSR are examples for this method.

### 2.1.1 SRCNN

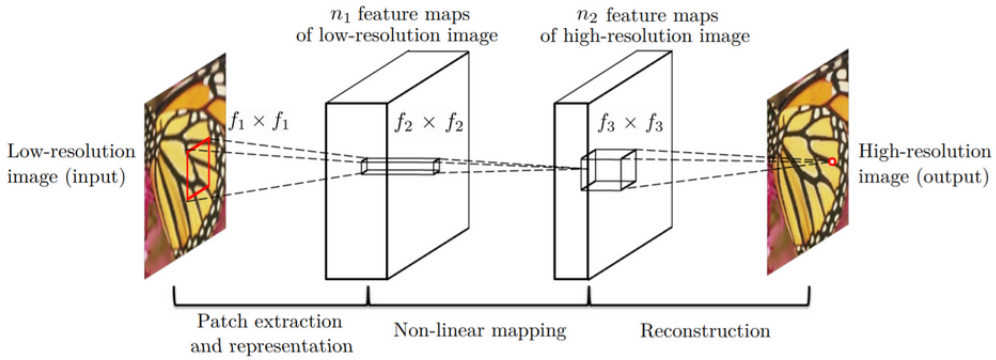


Figure 1: SRCNN

[3]

SRCNN [3] is a simple CNN architecture consisting of three layers; patch extraction, non-linear mapping and reconstruction (Fig. 1). The patch extraction layer is used to extract dense patches from the input, and to represent them using convolutional filters. The non-linear mapping layer consists of $1 \times 1$ convolutional filters used to change the number of channels and add non-linearity. The final reconstruction layer reconstructs the high resolution image. The MSE loss function is used to train the SRCNN while PSNR is used to evaluate the results.

### 2.1.2 VDSR

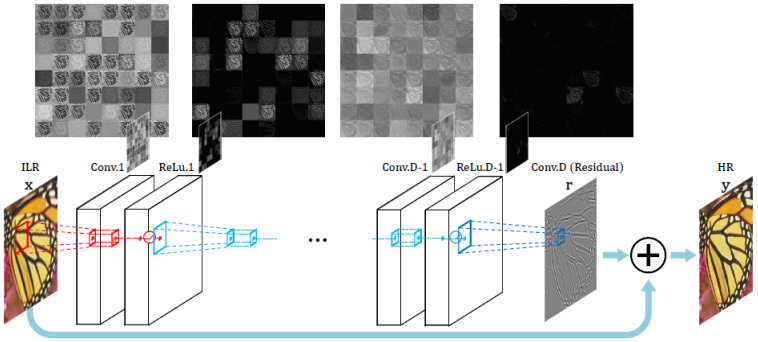


Figure 2: VDSR

[4]

Very Deep Super Resolution (VDSR) [4] is an improvement on SRCNN improved by following features. A deep network with small 3×3 convolutional filters is used instead of a smaller network with large convolutional filters based on the VGG architecture. The network tries to learn the residuals of the output image and the interpolated input, rather than learning the direct mapping. The initial low resolution image is added to the network output to get the final HR output. Gradient clipping is used to train the deep network with higher learning rates.

## 2.2 Post-Upsampling Super Resolution

In pre-upsampling Super Resolution, the computational power requirement is high because of its feature extraction process. This issue is solved in Post-Upsampling Super Resolution, as there feature extraction is done in the lower resolution space and then upsampling is done at the end. Here, a learned upsampling in the form of deconvolution/sub-pixel convolution is used instead of using simple bicubic interpolation for upsampling making the network trainable end-to-end. FSRCNN and ESPCN are examples for this method.

### 2.2.1 FSRCNN

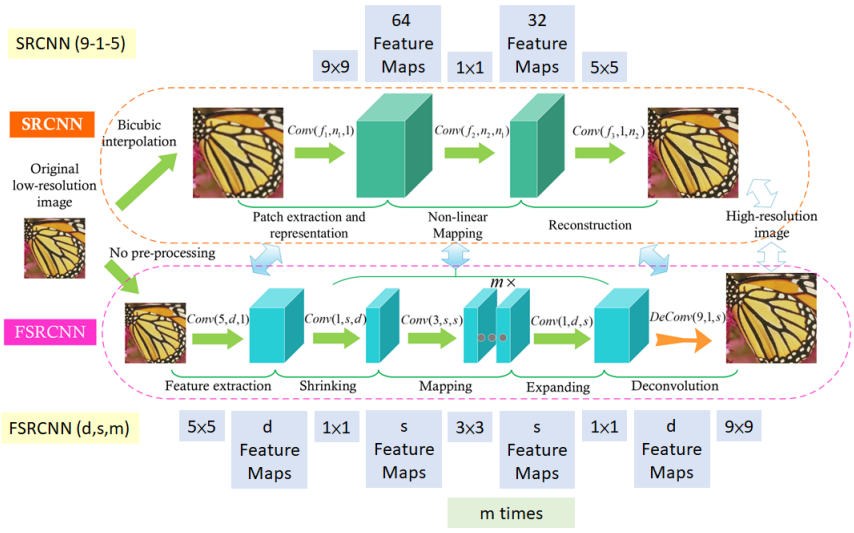


Figure 3: FSRCNN

[5]

In FSRCNN [5] there is no pre-processing or upsampling at the beginning and the feature extraction took place in the low resolution space. A $1\times1$ convolution is used after the initial $5\times5$ convolution to reduce the number of channels, and hence lesser computation and memory is used. Multiple $3\times3$ convolutions are used to simplify the architecture to reduce the number of parameters. Upsampling is done by using a learned deconvolutional filter.

### 2.2.2 ESPCN

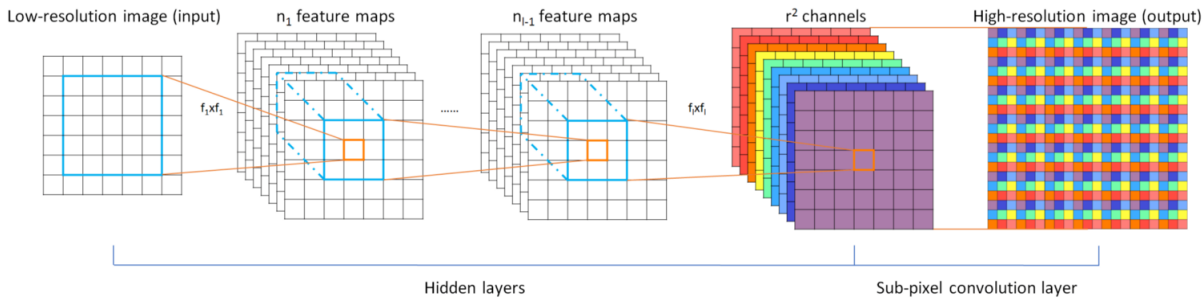


Figure 4: ESPCN

[3]

ESPCN [3] introduces the concept of sub-pixel convolution to replace the deconvolutional layer for upsampling. It resolves the checkerboard issue in deconvolution, which occurs due to the overlap operation of convolution. Sub-pixel convolution works by converting depth to space, as seen in the (Fig. 4). Pixels from multiple channels in a low resolution image are rearranged to a single channel in a high resolution image.

## 2.3 Residual Networks

EDSR, MDSR and CARN are some of the examples for Residual Networks. Among them, the EDSR will be described in Section 3.5. MDSR and CARN are described below.

### 2.3.1 MDSR

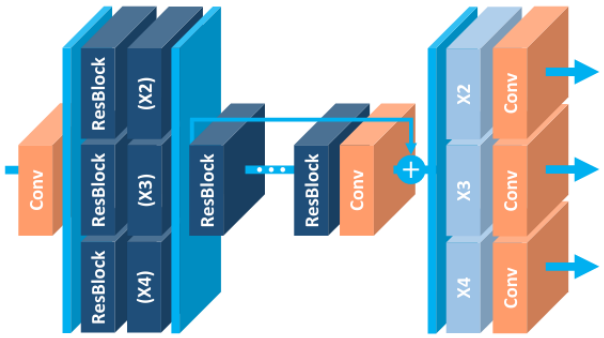


Figure 5: MDSR

[1]

MDSR [1] is an extension of EDSR with multiple input and output modules that give corresponding resolution outputs at 2x, 3x, and 4x. At the beginning, the pre-processing modules for scale-specific input are present consisting of two residual blocks with 5×5 kernels. A large kernel is used for the pre-processing layers to keep the network shallow while still achieving a high receptive field. At the end of the scale-specific pre-processing modules, shared residual blocks are there which is a common block for data of all resolutions. After the shared residual blocks, scale-specific upsampling modules are present. Although the overall depth of MDSR is 5x compared to single-scale EDSR, the number of parameters is only 2.5x due to the shared parameters.

### 2.3.2 CARN

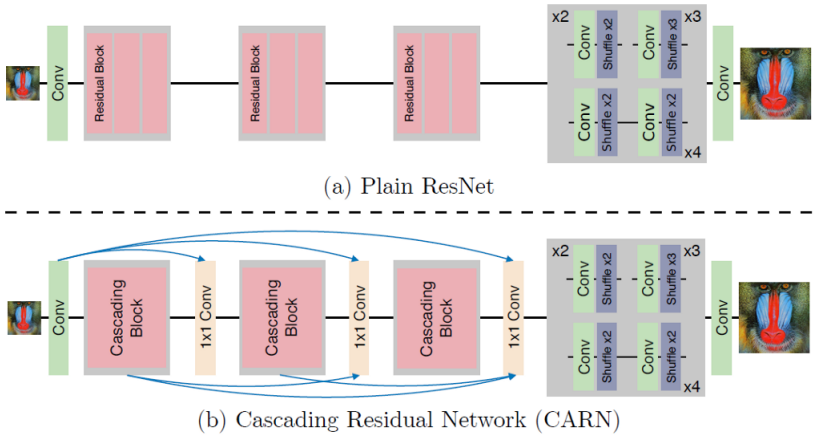


Figure 6: CARN

[6]

CARN [6] is an enhancement of residual networks where a cascading mechanism at both the local and global level incorporates features from multiple layers and gives the network the ability to receive more information. In addition to CARN, a smaller CARN-M is also proposed to have a lighter architecture with the help of recursive network architecture. The global connections in CARN are visualized in Fig. 6. The culmination of each cascading block with a $1\times1$ convolution receives inputs from all the previous cascading blocks and the initial input, thus resulting in an effective transfer of information.

## 2.4 Generative Models

In the previously described methods, the model is trained by considering the pixel difference between the input LR images and output SR images. However, the pixel loss does not optimize the perceptual quality of the output image. Generative models are designed considering the perceptual loss making the output image more realistic to the human eye view. SRGAN, EnhanceNet and ESRGAN are some of the examples for the generative models.

### 2.4.1 SRGAN

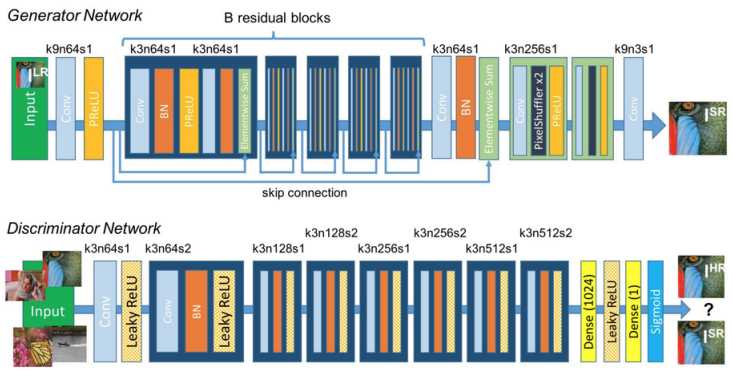


Figure 4: Architecture of Generator and Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) indicated for each convolutional layer.

Figure 7: SRGAN

[2]

SRGAN [2] is a GAN based architecture that uses the SRResnet network as the generator. The loss function of SRGAN consists of three terms; MSE loss capturing pixel similarity, perceptual similarity loss used to capture high-level information by using a deep network and the adversarial loss from the discriminator. Although SRGAN gives lower PSNR, it gives a perceptual quality for the resulting images.

### 2.4.2 EnhanceNet

EnhanceNet [7] uses a Fully Convolutional Network (FCN) with residual learning that employs an extra term in the loss function to capture finer texture information. Addition to the losses considered in SRGAN, a texture loss is also included in the EnhanceNet.

### 2.4.3 ESRGAN

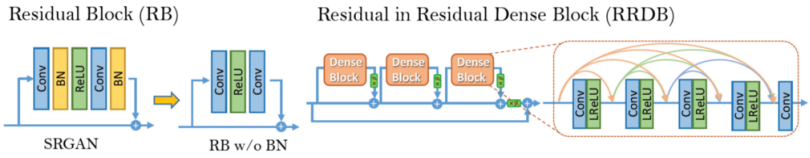


Figure 8: ESRGAN

[8]

ESRGAN [8] is designed as an improvement of SRGAN by adding a relativistic discriminator. It makes real images look less real compared to the generated images, thus helping to fool the discriminator. Batch normalization in SRGAN is also removed in ESRGAN, and Dense Blocks are used for better information flow.

# 3 Approach

The approach is to fine tune an EDSR model using an SRGAN on the div2k dataset with the bicubic_x4 downgrading factor.

## 3.1 DIV2K Dataset

The DIV2K dataset [9] contains DIVerse 2K resolution high-quality RGB images and their corresponding downgraded images with a large diversity of contents. The dataset contains a set of images with several downgrading factors and they are bicubic_x2, bicubic_x3, bicubic_x4, bicubic_x8, unknown_x2, unknown_x3, unknown_x4, realistic_mild_x4, realistic_difficult_x4 and realistic_wild_x4. Here, we have used the bicubic_x4 downgrading factor.

The div2k/bicubic_x4 is taken from TensorFlow datasets. The dataset contains 800 train images and 100 validation images. All the images come as a pair of high-resolution image and its corresponding low-resolution image.

## 3.2 Data Augmentation

The DIV2K dataset consists of only 800 training images and 100 validation images. To train a neural network, this small number of data will not be enough. Therefore, below mentioned data augmentation techniques will be used to get different varieties of the images. This helps to give a more generalized result as the model can learn from a wide variety of features that comes from a large number of images.

### 3.2.1 Random Crop

A random part of the size 96x96 is cropped from the high-resolution image, and then the corresponding part is cropped from the low-resolution image. As downsampling factor 4 is used, the cropped low-resolution image will be of size 24x24.

### 3.2.2 Random Flip

Here, a random flip operation is performed. A randomly generated value is obtained from the tf.random.normal method and flip operation will be performed only if the randomly generated number is less than 0.5. Here, both the low-resolution image and the high-resolution image are flipped left and right.

### 3.2.3 Random Rotate

Here, a random rotation operation is performed. A random value is generated from the tf.random.uniform method with a maximum value of 4. Then, both the low resolution and high-resolution images are rotated 90 degrees according to the times of the generated random number.

## 3.3 Data Preprocessing

### 3.3.1 Train Data Preprocessing

The training dataset is augmented with the random crop, random flip and random rotate. The batch size 4 and infinite repeat are assigned for the training data preprocessing. Then train data is prefetched in order to make the training process faster. The preprocessed train data contains pairs of low-resolution image segments of shape (24, 24, 3) and corresponding high-resolution image segments of shape (96, 96, 3).

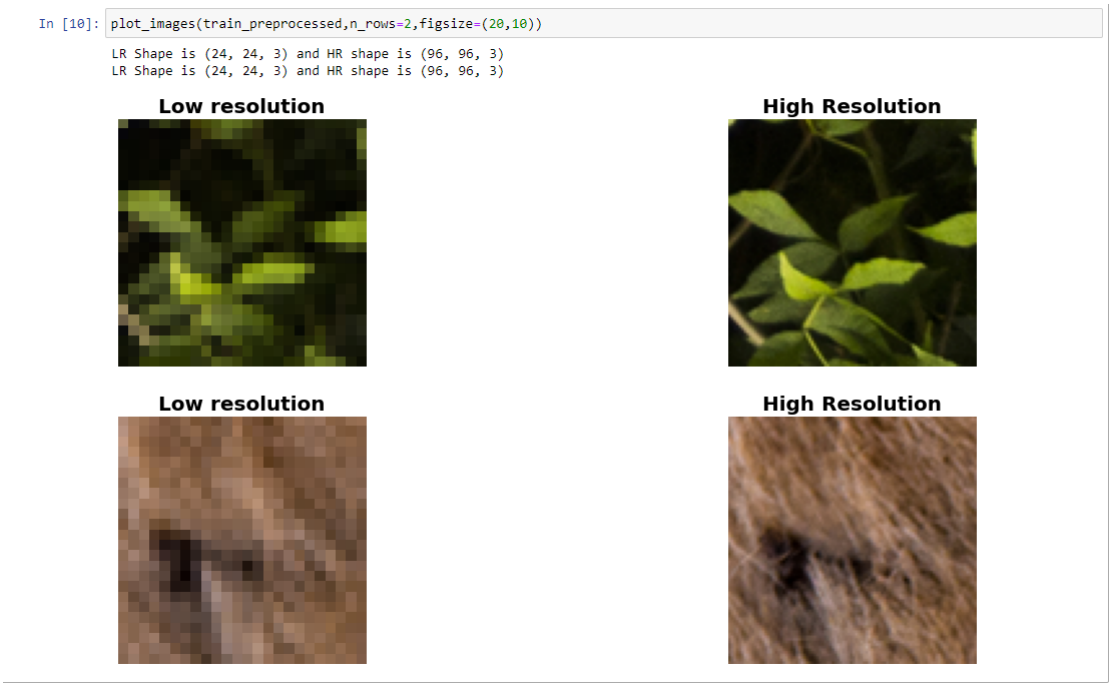


Figure 9: Preprocessed Training Data

### 3.3.2 Validation Data Preprocessing

For the validation dataset, any data augmentation transformations are not performed as it is aimed to get the resulting super-resolution image for the whole image. There, batch size 1 and repeat count 1 are assigned as the preprocessing of validation data.

The preprocessed validation data contains pairs of low-resolution image and a corresponding high-resolution image of the shape as same as the original image taken from the dataset.


Figure 10: Preprocessed Validation Data

## 3.4 Residual Networks

It is required for a super-resolution model to preserve information that comes in the low-resolution image when resulting in the high-resolution image as well. Residual networks are designed to learn the residuals between low resolution and high-resolution images. The identified information is conveyed to the higher layers via skip connections. The skip connections in ResBlocks (Residual Blocks) (Fig. 11) help to make a deeper network by optimizing the network. ResNet [10], SRResNet [2] and EDSR [1] are variations of residual network architectures.

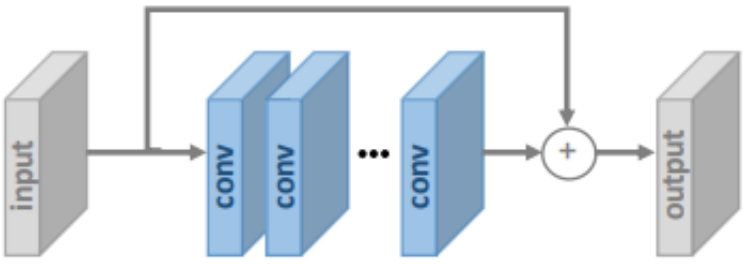


Figure 11: Residual Blocks

[11]

### 3.5 EDSR model

The SRResNet [2] architecture is designed based on the ResNet (Residual Networks) architecture by removing the final ReLU activation layer from the basic ResBlock. The EDSR model [1] is a further development of SRResNet [2] architecture where Batch Normalization layers are removed from the residual blocks.

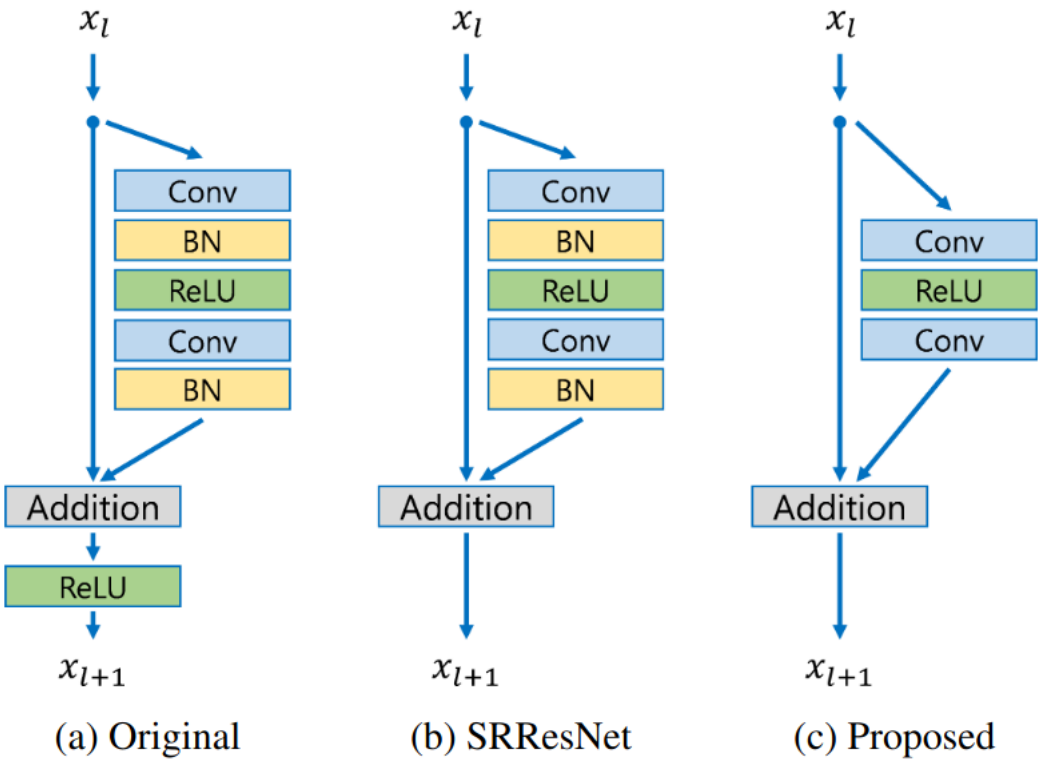


Figure 12: Comparison of residual blocks in original ResNet, SRResNet, and EDSR [1]

The removal of Batch Normalization layers has improved the model by memory reduction and increasing the accuracy. The range of flexibility of networks is removed when features are normalized thus keeping the batch normalization layers can affect the model accuracy. The batch normalization layers cost a considerable memory requirement. When the batch normalization layers are removed, the memory usage is saved approximately 40% in EDSR when compared to SRResNet [1]. This helps to design a model with better performance under limited computational resources.

The architecture of the EDSR model is as follows.

1. Normalize

   The input is normalized by subtracting the DIV2K RGB mean.

2. Conv2d

   A Conv2d layer of 64 filters is applied with a kernel of size 3.

3. ResBlock

   There are 16 ResBlocks used in the model.

4. Conv2d

   A Conv2d layer of 64 filters is applied with a kernel of size 3.

5. Add

   ResBlock output and the original Input are added.

6. Upsampling

   In the upsampling, a Conv2d layer is applied first. Then pixel shuffle is performed on the output of the Conv2d layer to reshape it. The result of the upsampling layer is a scaled output of the given scale value of 4.

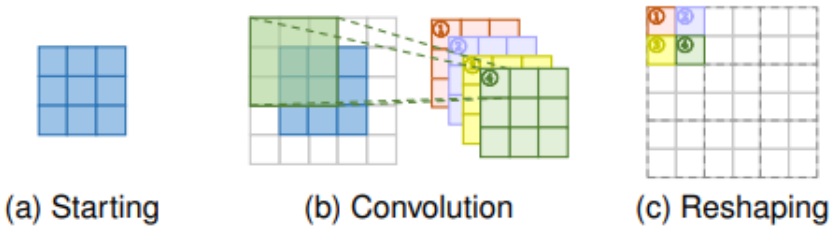


Figure 13: Upsampling

[11]

7. Conv2d

   A Conv2d layer of 3 filters is applied with a kernel of size 3.

8. Denormalize

   The output of the Conv2d layer is denormalized in order to get the original from the back.

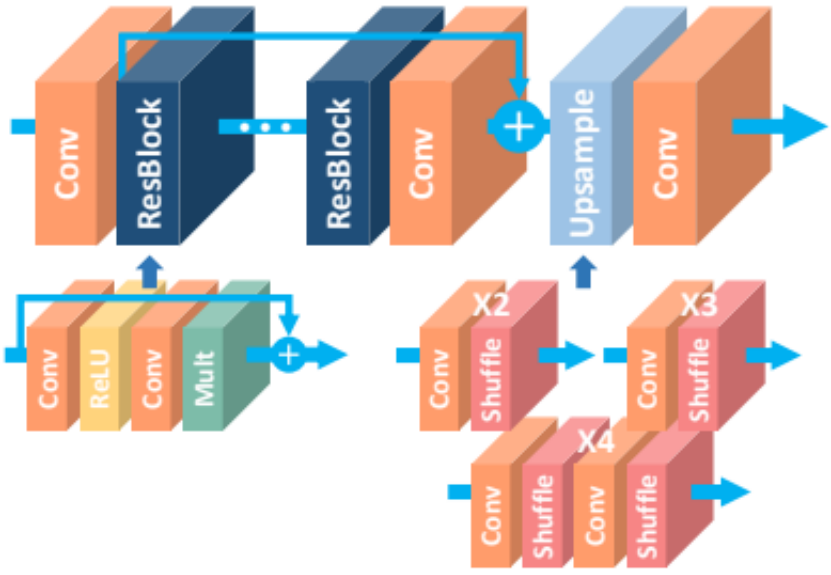


Figure 14: EDSR Model

[1]

### 3.6 Model Training

Due to the limitations of computational resources, we have used weights of pertained EDSR model from https://github.com/krasserm/super-resolution

### 3.7 Fine Tuning with SRGAN

The results obtained from the EDSR model is fine-tuned by training the SRGAN with perceptual loss and discrimination loss. This gives a more realistic texture to the obtained super-resolution image.

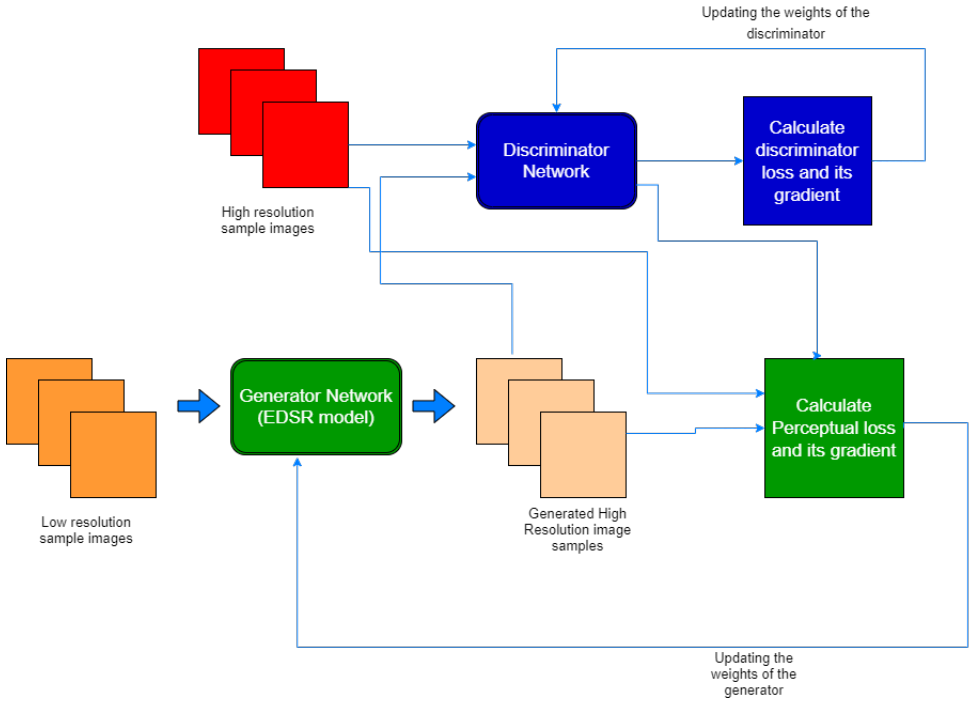


Figure 15: Fine Tuning with SRGAN

## 4 Experiment

### 4.1 Loss Function

In the presented solution, a supervised learning approach is taken where the low-resolution images are mapped to their high-resolution image. Also, the super-resolution task works on the pixels of an image. Therefore, Mean Absolute Error (MAE) is used as the loss function, where it is convenient to calculate the average of the absolute errors between the pixel mapping of the low-resolution image and the high-resolution image. MAE is an estimate of how the obtained image has deviated from the original image.

### 4.2 Performance Metrics

Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index (SSIM) were used as performance metrics.

### 4.2.1 PSNR

$$PSNR(f,g) = 10\log_{10}\left(255^2 / MSE(f,g)\right)$$

where

$$MSE(f,g) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(f_{ij} - g_{ij}\right)^2$$

Figure 16: PSNR Definition

[12]

PSNR [12] is an estimate of how much the resulting image is deviated from the original image. In Fig. 16, it can be observed that PSNR value approaches infinity when MSE approaches zero. This denotes that having a high value for PSNR is better as our aim is to minimize the MSE error.

### 4.2.2 SSIM

$$SSIM(f,g) = l(f,g)c(f,g)s(f,g)$$

where

$$\begin{cases} l(f,g) = \dfrac{2\mu_f\mu_g + C_1}{\mu_f^2 + \mu_g^2 + C_1} \\ c(f,g) = \dfrac{2\sigma_f\sigma_g + C_2}{\sigma_f^2 + \sigma_g^2 + C_2} \\ s(f,g) = \dfrac{\sigma_{fg} + C_3}{\sigma_f\sigma_g + C_3} \end{cases}$$

[12]

Figure 17: SSIM Definition

SSIM [12] is used to measure the similarity between the resulting image and the original image. SSIM is designed by modeling any image distortion as a combination of the factors; loss of correlation, luminance distortion and contrast distortion (Fig. 4.2.2). Hence it is more correlated with the quality perception of the human visual system (HVS) than the other performance metrics.

In Fig. 16 and Fig. 4.2.2, the terms comes in the equation are as below.

- f - The matrix that represents the data of original image
- g - The matrix that represents the data of resulting image
- M - The numbers of rows of pixels of the images
- i - The index of the considered row
- N - The number of columns of pixels of the image

- j - The index of the considered column

## 4.3 Pixel Loss

In PSNR [12], pixel-wise L2 loss is considered and aimed to optimize it in order to achieve better performance. However, considering only the pixel loss can lead to poor perceptual quality and may give results with lack of realistic texture. Therefore, it is needed to consider the perceptual loss also when obtaining the Super Resolution image.

## 4.4 Perceptual Loss

$$l^{SR} = \underbrace{\underbrace{l_{\mathrm{X}}^{SR}}_{\text{content loss}} + \underbrace{10^{-3} l_{Gen}^{SR}}_{\text{adversarial loss}}}_{\text{perceptual loss (for VGG based content losses)}} \quad [2]$$

Figure 18: Perceptual Loss

In SRGAN [2], a perceptual loss function composed of a content loss and an adversarial loss is used as the performance metric (Fig. 4.4). The content loss compares deep features extracted from SR and HR images. The generative component is added to the perceptual loss in order to favor the network on manifolds of natural images, by trying to fool the discriminator network.

## 4.5 Results

The EDSR model was trained with a pixel wise loss considering PSNR as the performance metric. Then it was fine tuned with a perceptual loss using an SRGAN in order to obtain more realistic textures in the Super Resolution image.

PSNR, SSIM and UQI values obtained for a randomly selected image in the dataset are mentioned below. The Low Resolution image and obtained corresponding Super Resolution image are also visualized. (Fig. 4.5)

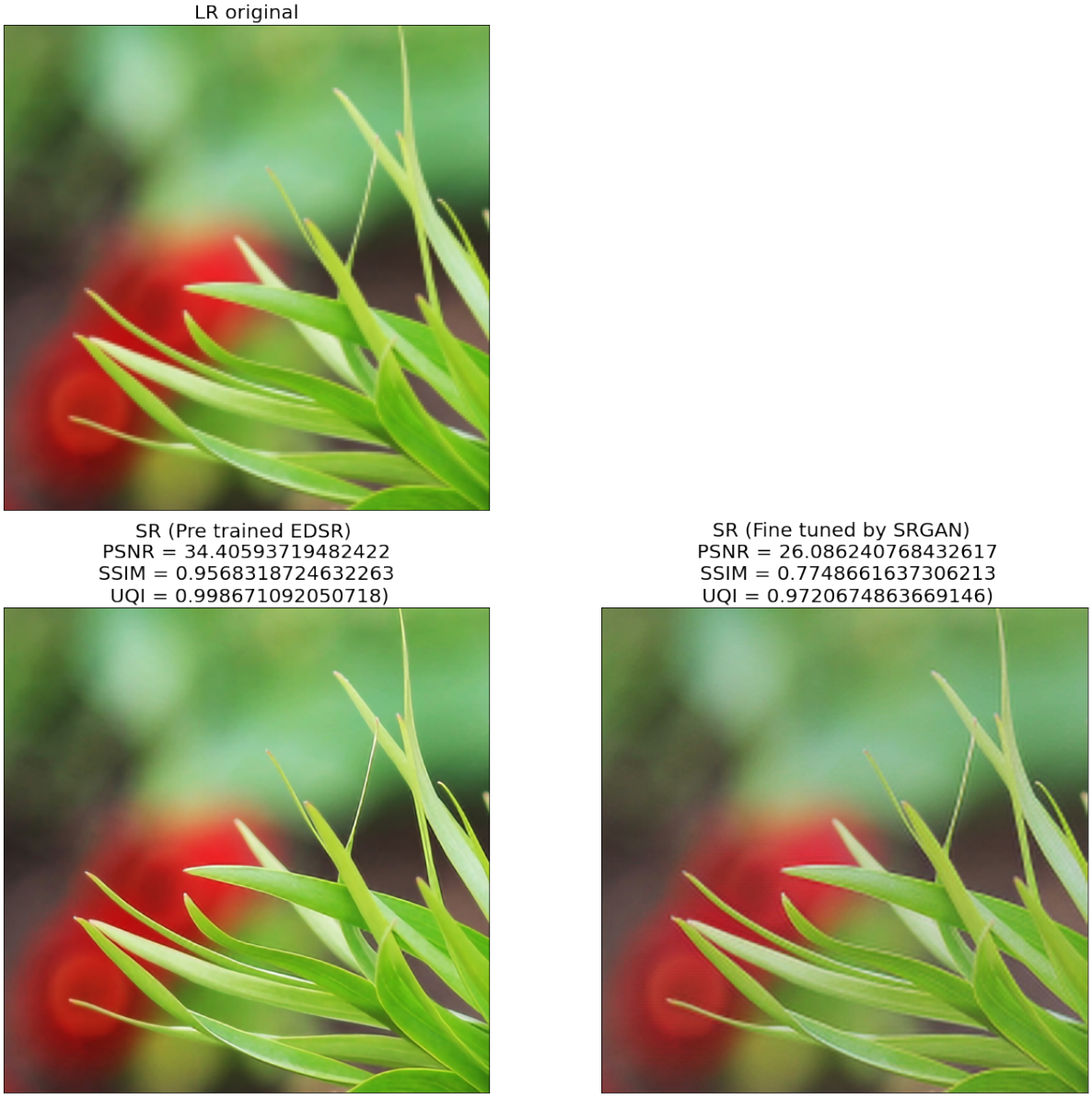


Figure 19: Results for an image from dataset

Results obtained for an unseen image are mentioned below. The Low Resolution image and obtained corresponding Super Resolution image are visualized, with the values obtained for performance metrics. (Fig. 4.5)


Figure 20: Results for an unseen image

# 5 Conclusion

Most of the time, Super Resolution models are trained considering PSNR as the performance metric. In PSNR the pixel-wise loss is considered always for the loss function. Even if we train a model to minimize the pixel loss and come up with a good PSNR value, it is not sufficient to target only the PSNR value when training the model. Because, some Super Resolution images may be there with high PSNR value, yet have a more unrealistic view for the human eye. For this case, it is needed to consider other performance metrics aimed at human eye views such as PIQE (Perception Based Image Quality Evaluator) and UQI (Universal Image Quality Index). However, the best model and performance metric to be applied should be decided based on the application of the Super-Resolution task. Further, the model can be improved more by training the model on varieties of data with mixed downgrading factors.

# 6 Contribution

- Literature review - E/16/078, E/16/134
- Data preprocessing - E/16/078
- Data augmentation - E/16/078
- Model architecture designing - E/16/078, E/16/134
- Work on EDSR model - E/16/078, E/16/134
- Fine-tune EDSR with SRGAN - E/16/134
- Model training - E/16/134
- Model evaluation - E/16/078, E/16/134

# References


[1] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution, 2017.

[2] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network, 2017.

[3] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network, 2016.

[4] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks, 2016.

[5] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network, 2016.

[6] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network, 2018.

[7] Mehdi S. M. Sajjadi, Bernhard Schölkopf, and Michael Hirsch. Enhancenet: Single image super-resolution through automated texture synthesis, 2017.

[8] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. Esrgan: Enhanced super-resolution generative adversarial networks, 2018.

[9] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[11] Martin Krasser. Single image super-resolution with deep neural networks, 2019. http://krasserm.github.io/2019/09/04/super-resolution/.

[12] Alain Horé and Djemel Ziou. Image quality metrics: Psnr vs. ssim. pages 2366–2369, 08 2010.