



**TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PURWANCHAL CAMPUS**

**COMPARISON OF DIFFERENT IMAGE SUPER RESOLUTION
TECHNIQUES**

BY:

ABHINAV JAISHI	(44002)
MEGHA THAKUR	(44021)
SANGHARSHA DAHAL	(44029)
KABIN POUDEL	(44047)

A FINAL REPORT TO THE DEPARTMENT OF ELECTRONICS AND
COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE
REQUIREMENT FOR THE BACHELOR'S DEGREE IN ELECTRONICS,
COMMUNICATION AND INFORMATION ENGINEERING

**DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
PURWANCHAL CAMPUS
DHARAN, NEPAL**

April 30, 2024

**COMPARISON OF DIFFERENT IMAGE SUPER RESOLUTION
TECHNIQUES**

BY:

ABHINAV JAISHI	(44001)
MEGHA THAKUR	(44021)
SANGHARSHA DAHAL	(44029)
KABIN POUDEL	(44047)

Project Supervisor
Asst. Prof. Pukar Karki

A project submitted to the Department of Electronics and Computer Engineering in
partial fulfillment of the requirements for the Bachelor's Degree in Electronics,
Communication and Information Engineering

Department of Electronics and Computer Engineering
Purwanchal Campus, Institute of Engineering
Tribhuvan University
Dharan, Nepal

April 30, 2024

©COPYRIGHT

The author has agreed that the Library, Department of Electronics and Computer Engineering, Purwanchal Campus, Institute of Engineering may make this report freely available for inspection. Moreover, the author has agreed that permission for extensive copying of this project report for scholarly purpose may be granted by the supervisor(s) who supervised the thesis work recorded herein or, in their absence, by the Head of the Department wherein the thesis report was done. It is understood that the recognition will be given to the author of this report and to the Department of Electronics and Computer Engineering, Purwanchal Campus, Institute of Engineering in any use of the material of this thesis report. Copying or publication or the other use of this report for financial gain without approval of the Department of Electronics and Computer Engineering, Purwanchal Campus, Institute of Engineering and author's written permission is prohibited.

Request for permission to copy or to make any other use of the material in this report in whole or in part should be addressed to:

Head

Department of Electronics and Computer Engineering

Purwanchal Campus, Institute of Engineering

Dharan , Sunsari

Nepal

DECLARATION

We declare that the work hereby submitted for Bachelors of Engineering in Electronics, Communication and Information at Institute of Engineering, Purwanchal Campus entitled "**“COMPARISON OF DIFFERENT IMAGE SUPER RESOLUTION TECHNIQUES”**" is our own work and has not been previously submitted by me at any university for any academic award.

We authorize Institute of Engineering, Purwanchal Campus to lend this report to other institution or individuals for the purpose of scholarly research.

ABHINAV JAISHI (44002)
MEGHA THAKUR (44021)
SANGHARSHA DAHAL (44029)
KABIN POUDEL (44047)

April 30, 2024

ACKNOWLEDGEMENT

We would like to take this opportunity to express our deepest gratitude and sincerest appreciation to all those who helped us to complete our project. We would like to give our special thanks to our project supervisor **Asst. Prof. Pukar Karki** whose help, suggestions, invaluable guidance and motivating feedback made the completion of the project a reality.

We would also like to thank **Asst. Prof. Pravin Sangroula**, Head of Department of Electronics and Computer Engineering for their regular support and co-operation. We would also like to acknowledge the help and co-operation of all the teaching and non-teaching staffs of the campus for the attainment of the project.

Lastly, we would like to express our deep appreciation and gratitude to our family, friends and well-wishers who have always helped us to keep up the morale and excitement in our work; without their encouragement and love we would not have achieved our goals.

ABHINAV JAISHI (44002)

MEGHA THAKUR (44021)

SANGHARSHA DAHAL (44029)

KABIN POUDEL (44047)

ABSTRACT

Image Super Resolution (ISR) techniques have been in development for a long time. It has witnessed substantial advancements in recent years, aiming to enhance the visual quality of low-resolution images. This study compares different ISR techniques based on deep learning and suggests which works best on different scenarios. This paper includes comparison of SISR techniques including SRGAN, SRResNet and SRCNN. Each technique has distinct architectural approaches and training methodologies, addressing the common goal of reconstructing high-resolution images from their low-resolution counterparts. The study aims to provide valuable insights for enthusiasts and researchers interested in using SISR techniques for high-quality image super-resolution. Each of these techniques has its strengths and weaknesses, and understanding them can help researchers and enthusiasts choose the most suitable method for their specific needs.

Keywords: *Super Resolution Generative Adversarial Network (SRGAN), Super Resolution Residual Network (SRResNet), Super Resolution Convolutional Neural Network (SRCNN), Single Image Super Resolution (SISR)*

TABLE OF CONTENTS

COPYRIGHT	i
DECLARATION	ii
ACKNOWLEDGEMENT	iii
ABSTRACT	iv
TABLE OF CONTENTS	vi
LIST OF FIGURES	vii
LIST OF TABLES	viii
1 INTRODUCTION	1
1.1 Statement of problem	1
1.2 Objectives	1
1.3 Scope and Applications	2
2 LITERATURE REVIEW	3
3 HARDWARE AND SOFTWARE REQUIREMENTS	5
3.1 Hardware Requirements	5
3.2 Software Requirements	5
4 RELATED THEORY	6
4.1 Architectural Explanations	6
4.2 General Explanations	9
5 METHODOLOGY	15
5.1 Datasets	15
5.2 Data Preprocessing	16
5.3 Training	16
5.4 Evaluation Metrics	17

6 RESULT AND CONCLUSION	19
6.1 Evaluation Metrics	19
6.2 Image Comparisons	23
6.3 Recommendation	26
6.4 Future Enhancements	26

List of Figures

1	SRCNN Architecture	6
2	SRResNet Architecture or Generator of SRGAN	7
3	Discriminator Architecture	8
4	A visual explanation of convolution.	10
5	Tanh curve	11
6	Relu curve	12
7	Leaky Relu curve	13
8	Parametric Relu curve	13
9	Pixel Shuffle	14
10	Residual block	14
11	COCO Set Images	15
12	Test Set Images(Set5, Set14, BSDS100)	15
13	SRResNet PSNR and SSIM Plot for Set5	20
14	SRGAN PSNR and SSIM Plot for Set5	21
15	SRCNN PSNR and SSIM Plot for Set5	22
16	Image of super resolved lizard with SRResNet, SRGAN and HR image .	23
17	Images of super resolved tiger with SRResNet, SRGAN and HR image .	24
18	Images of super resolved flowers with SRResNet, SRGAN and HR image	25

List of Tables

1	Evaluation Metrics in Test Datasets	19
---	---	----

1. INTRODUCTION

Image super resolution is a technique used to enhance the resolution and quality of image beyond its original size. It is useful technique in the field of image processing. It can be used to enhance the quality of image captured in night, shaky image etc. There are various techniques and algorithm used for image super resolution, including both traditional methods and deep learning-based approaches. Traditional super image resolution techniques are based on mathematics techniques. These techniques have been used for many years. Some of the traditional super resolution techniques include: Interpolation, Edge-Directed interpolation, Bayesian methods etc. Traditional methods have certain limitations compared to deep learning approaches. They may struggle to capture complex patterns and textures, and their performance is often constrained by the hand-crafted features and assumptions used in the algorithms. Deep learning based super resolution typically employs Convolution Neural Network (CNN) to learn the mapping between low-resolution and high-resolution images. The network is trained on large dataset of paired low-resolution and high- resolution images. During training, the networks learn to identify patterns and features that enables it to generate high-resolution details from low-resolution inputs.

1.1. Statement of problem

The need of super image resolution arises when we encounter low-resolution images that lack the desired level of detail and clarity that may be due to various factors such as limitations in capturing hardware, resizing operations, or compression from social media. There is not only need of Super Resolution, We also need to know which method works best for which usecase.

1.2. Objectives

- To implement Image Super Resolution methods like SRCNN, SRResNet and SRGAN.
- To compare their performance in different scenarios with metrics like PSNR and SSIM.

1.3. Scope and Applications

Image super resolution has numerous practical applications in field like photography, surveillance, and digital art where enhanced image quality is crucial for accurate decision-making, analysis, and interpretation. The development of efficient and accurate super-resolution method involves exploring traditional signal processing techniques, as well as cutting-edge deep learning approaches, to achieve optimal results while considering computational efficiency and resource constraints. We can understand that not every technique will work well in every scenarios. So, There is a need for understanding the techniques to use them where they perform best.

Super image resolution can be used for following things:

- Super resolution can be used to improve the resolution of old and degraded images, preserving historical photographs, artworks, and documents.
- Super resolution can be used in digital cameras and smartphones to provide better zoom capabilities without significant loss of image quality.
- Super resolution can enhance the resolution of individual frames in videos, leading to improved video quality and better extraction of information.
- In surveillance systems, low resolution camera feeds can be upscaled to improve the ability to identify faces, license plates, or other critical details, helping in investigations.

2. LITERATURE REVIEW

In the timeline of image scaling process, the first used methods were different interpolation techniques and sparse representation-based methods. But with the advent of deep-learning based image scaling techniques interpolation methods were less commonly used in image enhancement.

D. Han (2013) “**Comparison of commonly used image interpolation methods**” discusses multiple interpolation methods like nearest neighbor, bilinear, bicubic, etc. Although interpolation is fast but have some disadvantages. But using these methods was not optimal for our use case as they may lead to the loss of fine details and sharpness in image and can amplify noise present in image which may not be accurate in real-world images.[1]

C. Dong, et. al, “**Image super-resolution using deep convolutional networks**” is a seminal paper on use of Deep Learning for the use of for the task of SR. It only consists of three layers and requires the LR image to be up-sampled using bicubic interpolation prior to being processed by the network, but it was shown to outperform the state-of-the-art methods of that time.[2]

W. Yang et al., "Deep Learning for single image super resolution" is an article that discusses the challenges of SISR and how deep learning has been applied to address them. The authors review representative deep learning-based SISR methods, categorizing them by their contributions to two key aspects: efficient neural network architectures and effective optimization objectives. They analyze the strengths and weaknesses of these methods. Some limitations of current methods include the use of bicubic interpolation as input, the lack of efficient upsampling modules, and the difficulty of handling unknown degradations. [3]

C. Ledig et al., “**Photo-realistic single image super-resolution using a generative adversarial network**” discusses about a Generative Adversarial Network (GAN) for image super-resolution (SR). It is the first framework capable of inferring photo-realistic natural images for 4x upscaling factors using GAN. To achieve this, it uses a perceptual loss function which consists of an adversarial loss and a content loss. The adversarial loss pushes the solution to the natural image manifold using a discriminator network that is trained to differentiate between the super-resolved images and original photo-realistic

images. In addition, it uses a content loss motivated by perceptual similarity in VGG space instead of similarity in pixel RGB space. This deep residual network is able to recover photo-realistic textures from heavily down sampled images on public benchmarks. An extensive mean-opinion-score (MOS) test shows hugely significant gains in perceptual quality using SRGAN. The MOS scores obtained with SRGAN are closer to those of the original high-resolution images than to those obtained with any state-of-the-art method at the time. Both SRResNet and SRGAN are explained in related theory portion of the study.[4]

3. HARDWARE AND SOFTWARE REQUIREMENTS

3.1. Hardware Requirements

We used the system with following specifications for training:

- PC with Windows 11
- 16 GB RAM
- Nvidia Geforce GTX 1050(4GB)
- 256 GB storage
- Processor: Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz

3.2. Software Requirements

The software requirements for developing, training, evaluating, and deploying the model effectively are listed below:

- **Integrated Development Environment (IDE):** We used Visual Studio Code (VS Code) as IDE for training, testing and documentation. It is a popular feature rich code editor. It is available on Windows, macOS, and Linux.
- **Version Control System:** We used Git and Github. Git is a distributed version control system. It allows multiple developers to collaborate on projects, keeping track of modifications, managing different versions of files, and merging changes.
- **Web framework:** We simply used Streamlit framework. Streamlit is an open-source Python library that makes it remarkably easy to create and share beautiful, custom web apps for machine learning and data science.
- **Deep-Learning Frameworks:** We used PyTorch. It is an open-source, Python-based deep learning library. It allows to build and train neural networks for various tasks like computer vision, natural language processing, and more.

4. RELATED THEORY

4.1. Architectural Explanations

Image Super Resolution upscales a Low-Resolution Image to High Resolution Image, filling in the missing pixels with the different techniques. There are other simpler methods to upscale images like Linear or Bi-cubic interpolation, but they do not generate any new information based on the environment and hence are not super useful to upscale an LR image. Deep Learning based methods require huge amounts of data so that the model is not overfitted, the dataset needs to contain an HR and LR version of the same image that are perfectly aligned to each other. We need to synthetically create LR images from HR images.

SR aims to then reverse whichever degradation process is considered, to retrieve the original underlying high-fidelity image. We have implemented SRCNN, SSResNet and SRGAN. Their architecture is explained below:

1. **SRCNN:** Super-Resolution Convolutional Neural Network, is a deep learning architecture devised for single-image super-resolution tasks. Developed by **Dong et al. in 2014** [2], SRCNN revolutionized image enhancement by learning complex mappings between low-resolution and high-resolution images directly from data. Its architecture consists of multiple layers of convolutional neural networks (CNNs), which process input images in a hierarchical manner, gradually extracting high-level features. The architecture of SRCNN is given below:

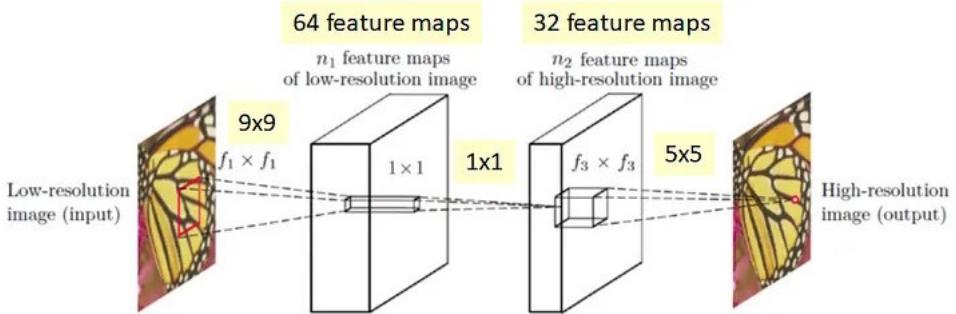


Figure 1: SRCNN Architecture

We converted image from training database into Y,Cb,Cr format and used only Y (Luminance) component for training the model. The primary loss function used

in SRCNN is Mean Squared Error (MSE). We used Adam optimizer, optimizer is responsible for updating the network’s parameters during the training process in order to minimize the chosen loss function.

2. **SRResNet:** The SRResNet is a fully convolutional network designed for 4x super-resolution. It incorporates residual blocks with skip connections to increase the optimizability of the network despite its significant depth. The SRResNet is trained and used as a standalone network and provides a nice baseline for the SRGAN – for both comparison and initialization. We use the following architecture for this network.

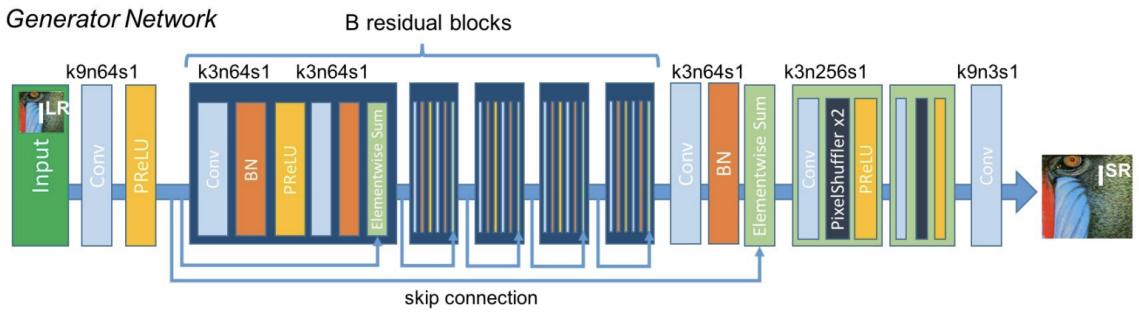


Figure 2: SRResNet Architecture or Generator of SRGAN

The SRResNet first contains convolution block of large Kernel Size 9x9, stride of 1 and 64 channels with PreLU activation. There are 16 residual blocks with convolution layer of Kernel size 3x3 followed by batch normalization, PreLU same conv layer again and batch normalization again. The output then is passed through same 3x3 conv layer and batch normalized. Two subpixel convolution blocks are used followed by PreLU activation each of which provides two times upscaling. Finally, a convolution with large kernel size of 9x9 and 1 stride with 3 out channels for RGB is done with Tanh activation to get super resolved image. In Forward Pass, SRResNet produces a 4x upscaled image from provided low-res image using the above architecture. Mean-Squared Error (MSE) is used as the loss function to compare the upscaled image and original high-quality image. MSE is a type of content loss but here it only looks in RGB space of predicted and target images. Minimizing the MSE by changing the parameters of the network will make the model produce images closer to the original images.

3. **SRGAN**: Super-Resolution Generative Adversarial Network consist of two adversary networks Generator and Discriminator which are trained in tandem. The goal of the Generator is to learn to super-sample an image such that Discriminator can't tell difference between artificial and natural origins. The interplay between these two networks leads to the improvement of both over time.

The Generator learns not only by minimizing content loss, as in the case of the SRResNet but also by spying on the Discriminator's methods. By providing the Generator access to the Discriminator's inner workings in the form of the gradients produced therein when backpropagating from its outputs, the Generator can adjust its parameters in a way that alters the Discriminator's outputs in its favour. As the Generator produces more realistic high-resolution images, we use these to train the Discriminator, improving its discriminating abilities.

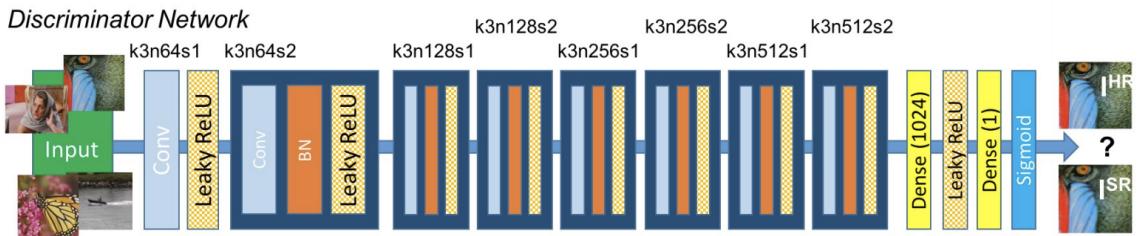


Figure 3: Discriminator Architecture

The Generator has same architecture as SRResNet. Training Discriminator is not different from any binary image classifier. It is trained on both real images and generated images from generator. In Forward Pass, it outputs the probability score P_{HR} . If the provided image was real, we desire the P_{HR} to be as high as possible towards 1. And for generated image as low as possible towards 0. We use Binary Cross Entropy loss $-\log P_{HR}$ when input image is real and $-\log P_{SR}$ when image is generated. Here $P_{SR} = 1 - P_{HR}$. Minimizing these losses by changing the parameters will make discriminator predict higher probability for real images and lower probability for generated images.

We use better content loss than that of SRResNet. In ResNet we use MSE loss in RGB space which produces overly smooth images with no finer details. There are a lot of possibilities of similar pixel combination that can be formed from low resolution image patch. When we use content loss in RGB space, it averages the

output rather than choosing one of the combinations which would produce better result as an overly smooth “averaged” prediction will always have lower MSE.

We can use CNN trained to classify images to find deeper meaning of patterns in images. This new representation space is more suitable for calculating content loss and can hallucinate new details showing creativity. We specifically use the VGG19 network as recommended in the paper. We use MSE-based content loss in this VGG space to compare the images. The use of content loss is only one component of the generator update, we use adversarial loss obviously. The super-resolved image is passed through the Discriminator with its weight frozen not to update the discriminator but to get the probability score P_{HR} with misleading label and use the BCE loss $-\log P_{HR}$ and resulting gradient information to update the Generator’s weights such that it produces images closer to its natural origin. GAN is trained in an interleaved fashion, where the generator and discriminator are alternatively trained for short periods of time just once before making the switch in this case.

4.2. General Explanations

1. **Conv2d:** Conv2D stands for two-dimensional convolution, a fundamental operation in convolutional neural networks (CNNs). A convolutional layer applies a series of filters or kernels to an input image. Each filter detects specific patterns or features, like edges, textures, or shapes. The convolution operation involves sliding these filters over the input image, performing element-wise multiplication between the filter and the overlapping portions of the image, and then summing up the results to create a feature map.
 - **Kernel:** A kernel describes a filter that we are going to pass over an input image. The kernel will move over the whole image, from left to right, from top to bottom by applying a convolutional product. The output of this operation is called a filtered image.
 - **Stride:** Stride refers to the number of steps the convolutional filter moves horizontally or vertically during convolutional operation. For example, a stride of 2 means the filter moves two pixels at a time.

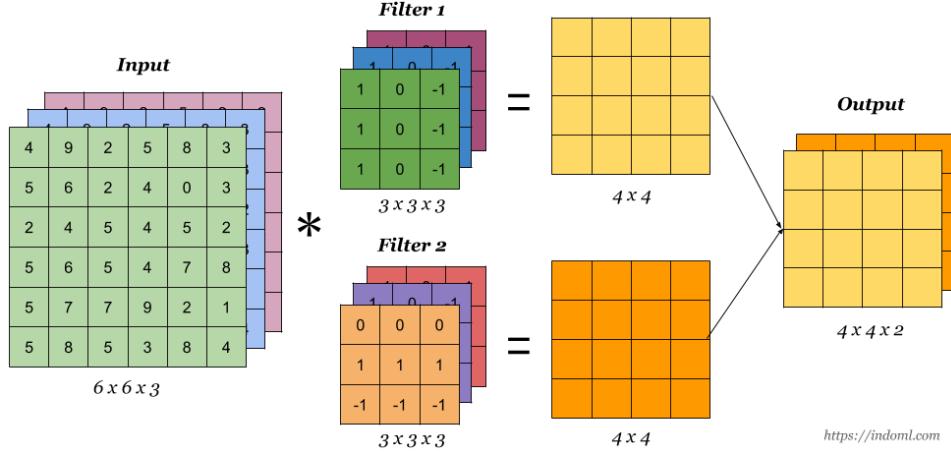


Figure 4: A visual explanation of convolution.

- **Padding:** Padding describes the number of pixels added to the sides of the input channels before their convolutional filtering. Usually padding pixels are set to zero. This is particularly useful when we want the size of the output channels to be equal to the size of the input channels. When the kernel size is 3*3 then the output channel size is decreased by one on each side. To overcome this problem, we can use padding of 1.
- **In channel:** In channel are used to describe how many channels are present in the input. For instance, if the input is an RGB image, then it has three channels. When using a Conv2d layer, we can specify the number of in channel, and the convolution operation is performed independently on each channel of the input. This allows the network to learn different features from different input channels, enabling it to detect various patterns within each channel.
- **Out Channel:** Out Channel specifies the number of output channels produced by the convolutional layer. Each output channel corresponds to a distinct convolutional filter (or kernel) applied to the input.

2. **Batch Normalization:** Batch normalization is a technique that normalizes the inputs to each layer in a network by adjusting and scaling to have zero mean and unit variance. This helps to reduce “internal covariate shift” problem which can hinder training. An internal covariant shift occurs when there is a change in input distribution to the network. When the input distribution changes, hidden layers try

to learn to adapt to the new distribution. This slows down the training process. If a process slows down, it takes a long time to converge to a global minimum. Batch normalization transform the data so that its values have similar scale, making it easier for the network to learn and converge faster.

Formula for batch normalization:

$$y = \frac{x - E[x]}{\sqrt{Var[x] + \epsilon}} + \gamma + \beta \quad (1)$$

where, β and γ are learnable parameter vector.

X represent the value of the feature map/channel.

$Var[X]$ is the variance of the mini batch for the feature map/channel.

ϵ is small constant.

3. **Activation Function:** Activation function decides whether a neuron should be activated or not. This means that it will decide whether the neuron's input to the network is important or not in the process of prediction. The primary role of the activation function is to transform the summed weighted input from the node into an output value to be fed to the next hidden layer or as output. Different types of activation function used in our project is listed below:

- **Tanh:** Tanh function is similar to the sigmoid activation function, and even has the S-shaped with the difference in output range of -1 to 1. In Tanh, the larger the input (more positive), the closer the output value will be to 1.0, whereas the smaller the input (more negative), the closer the output will be to -1.0.

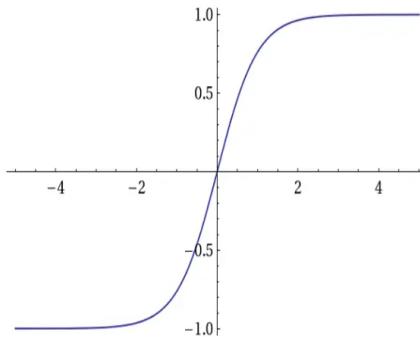


Figure 5: Tanh curve

- **ReLU:** ReLU stands for Rectified Linear Unit. Although it gives an impression of a linear function, ReLU has a derivative function and allows for backpropagation while simultaneously making it computationally efficient. The main catch is that ReLU function does not activate all the neurons at the same time. The neurons will only be deactivated if the input to the ReLU function is less than zero. [5]

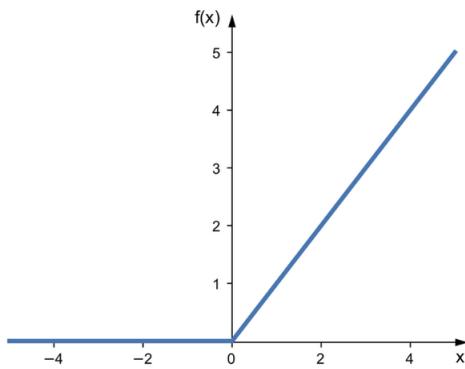


Figure 6: Relu curve

Mathematically it can be represented as:

$$R(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad (2)$$

The major limitation of ReLU is Dying ReLU problem. The negative side of the graph makes the gradient value zero. Due to this reason, during the backpropagation process, the weights and biases for some neurons are not updated. This can create dead neuron which never get activated.

- **Leaky Relu:** : Leaky ReLU is an improved version of ReLU function to solve the Dying ReLU problem as it has small positive slope in the negative area. The advantage of Leaky ReLU is same as ReLU, in addition to the fact that it does enable backpropagation, even for negative input values. By making this minor modification for negative input values, the gradient of the left side of the graph comes out to be a non-zero value.

The limitation faced by Leaky ReLU is the prediction may not be consistent for negative input values.

Mathematically it can be represented as:

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad (3)$$

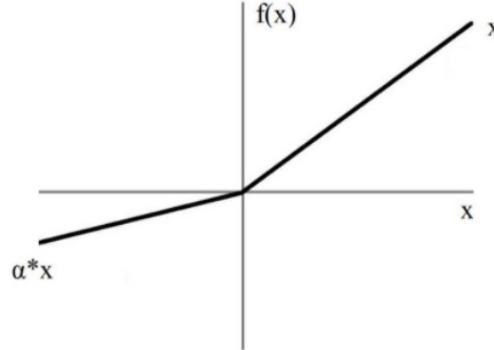


Figure 7: Leaky Relu curve

- **Prelu:** Parametric ReLU is another variant of ReLU that aims to solve the problem of gradient's becoming zero for the left half axis. This function provides the slope of the negative part of the function as an argument.[6]

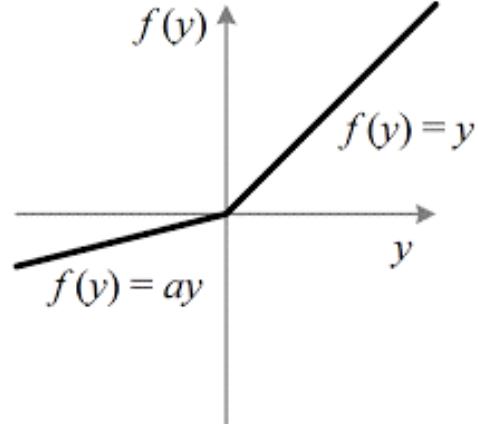


Figure 8: Parametric Relu curve

Mathematically it can be represented as:

$$R(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad (4)$$

where ‘a’ is the slope parameter for negative values. Prelu can learn slope parameter using backpropagation at a negligible increase in the cost of training.

4. Pixel Shuffle: The Pixel shuffle is a technique used in image super resolution to upscale an image by a factor of r . It works by rearranging the pixels in a feature map of size $(W, H, C * r^2)$ to form a super-pixel feature map of size $(r*W, r*H, C)$. [7]

The pixel shuffle operation can be described as follows:

- For each pixel in the input feature map, split its value into r^2 sub-pixels.
- Arrange the sub-pixels in grid-like pattern, with each sub-pixels occupying a position in a feature map.

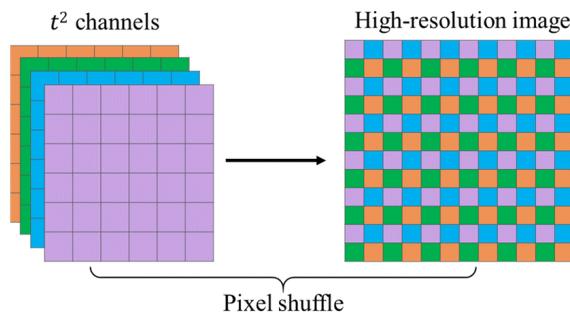


Figure 9: Pixel Shuffle

5. Residual Block: A residual block is a building block that contains one or more convolutional layers and a skip connection. The skip connection allows gradient to flow through the network more efficiently, improving the training process and avoiding the vanishing gradient problem.

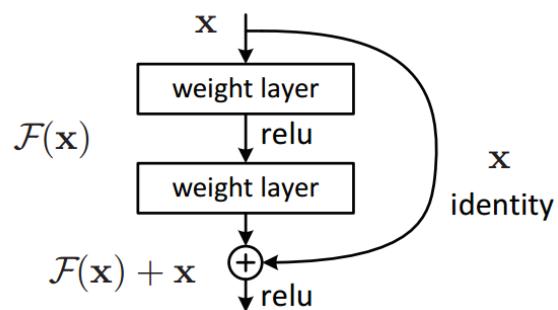


Figure 10: Residual block

5. METHODOLOGY

In this chapter we will discuss about Data Collection, Data Preprocessing , Metrics used to evaluate the results.

5.1. Datasets

We use different datasets for training and testing purposes. We tried CelebA dataset, Outdoor Scene Test(OST) datasets and finally landed upon Common Objects in Context dataset(COCO) for training purposes.



Figure 11: COCO Set Images

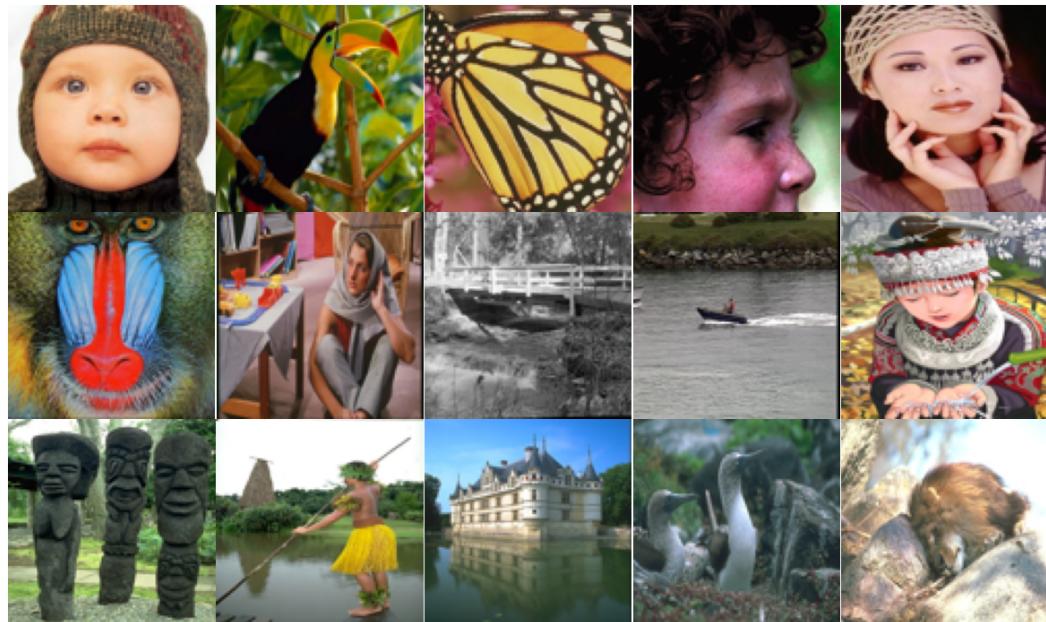


Figure 12: Test Set Images(Set5, Set14, BSDS100)

We are using 2014 variant of COCO dataset. It contains 83k train images and 41k validation images. We are using both of these sets as training set images in our case. We

have a total of 124k training images. We have used three famous testing datasets used in Super Resolution application Set5, Set14 and BSD100. They are industry standard testing data sets. We are using to these test datasets to calculate PSNR and SSIM and superesolve them to compare between different superresolved images.

5.2. Data Preprocessing

We can't process the images directly as images are very resource intensive, We need to train on the cropped patches of images. And to train generator we need high resolution and low resolution version of same patch. In Data Preprocessing step we crop and downsample the images.

- Cropping: We need to crop a random patch of image every time. So We create a function to give valid left, top, right, bottom postion of images for a certain patch size. In this case we use 96x96 patch of an image. We then use crop function of the PIL library to crop a patch of image by passing those postions.
- Downsampling: We use Bicubic downsampling to downsample the cropped patch of image by a scaling factor of 4 in this case. Bicubic Downsampling is a method used to reduce the resolution of an image by averaging pixel values in a specific way. It is commonly employed in image processing and computer graphics when it's necessary to create a lower-resolution version of an image.

5.3. Training

The training process is explained below

1. For SRCNN:

- Input: Downsampled images of size 24x24 for 4x scaling and 32x32 for 3x scaling only in Y-channel.
- Optimizer: Adam optimizer with a learning rate of 1e-4.
- Loss function: Mean Squared Error (MSE) loss between the generated images and the original high-resolution images.

2. For SRResNet:

- Input: Downsampled images of size 24x24.
- Optimizer: Adam optimizer with a learning rate of 1e-4.
- Loss function: Mean Squared Error (MSE) loss between the generated images and the original high-resolution images.

3. For SRGAN

- (a) Input: Downsampled images of size 24x24.
- (b) Optimizer: Adam optimizer with a learning rate of 1e-4.
- (c) Loss Function:

$$\text{Perceptual Loss} = \text{Content Loss} + \text{Adversarial Loss}$$

- Content loss: MSE loss but computed in the feature space of a VGG19 network. The VGG19 extracts features from both the super-resolved image and the original image, and the MSE is computed on these feature representations.
- Adversarial loss: Binary Cross Entropy (BCE) with Logit Loss. The super-resolved images are fed into the discriminator, and the output of the discriminator is used to calculate the adversarial loss.

5.4. Evaluation Metrics

We use PSNR and SSIM as two evaluation Metrics to measure the performance of the models. Although, Images perception is very different for humans than these metrics. It provides some theoretical value to compare. The author of SRGAN paper suggested Mean opinion score as a better alternative but it is very difficult to conduct such survey.

- **PSNR (Peak Signal-to-Noise Ratio)** is a measure of the quality of a reconstructed or processed image. It compares the peak signal power to the noise power that affects the quality of the image. It can be calculated as :

$$PSNR = 10 * \log_{10}\left(\frac{MAX^2}{MSE}\right) \quad (5)$$

where MAX is the maximum possible pixel value of the image and MSE (Mean Squared Error) is the average squared difference between the original and the

reconstructed images. Higher PSNR values generally indicate better image quality, as it implies lower distortion or noise in the reconstructed image.

- **SSIM (Structural Similarity Index)** is a metric that measures the structural similarity between two images. It takes into account luminance, contrast, and structure information to provide a more comprehensive assessment of image quality. It can be calculated by using three terms luminance(l), contrast(c) and structure(s) as:

$$SSIM(x, y) = \frac{(2 * \mu_x * \mu_y + C_1)(2 * \sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (6)$$

where μ is the mean, σ is the standard deviation, and the σ_{xy} is the covariance between x and y. C_1 and C_2 are constants to avoid instability when the denominator is close to zero.

SSIM values range from -1 to 1, where 1 indicates perfect similarity. Higher SSIM values imply better structural similarity between the original and the reconstructed images.

6. RESULT AND CONCLUSION

Here, We will discuss how different models performed when compared to each other. We compare them using PSNR and SSIM in different test sets. Since, these metrics don't tell the complete picture we have provided the comparison images from different models here.

6.1. Evaluation Metrics

We calculated PSNR and SSIM for all the test datasets Set5, Set14 and BSDS100 after the training was completed with each model.

	PSNR	SISM	PSNR	SISM	PSNR	SISM
SRCNN(3x)	31.02	0.9	27.668	0.812	26.939	0.781
SRCNN(4x)	28.751	0.844	25.72	0.734	25.44	0.696
ResNet	29.83	0.887	28.275	0.795	27.203	0.751
SRGAN	26.132	0.821	25.684	0.716	24.969	0.665
	Set5		Set14		BSD100	

Table 1: Evaluation Metrics in Test Datasets

During Training We calculated PSNR and SSIM for Set5 datasets at every epoch for both SRResNet and SRGAN.

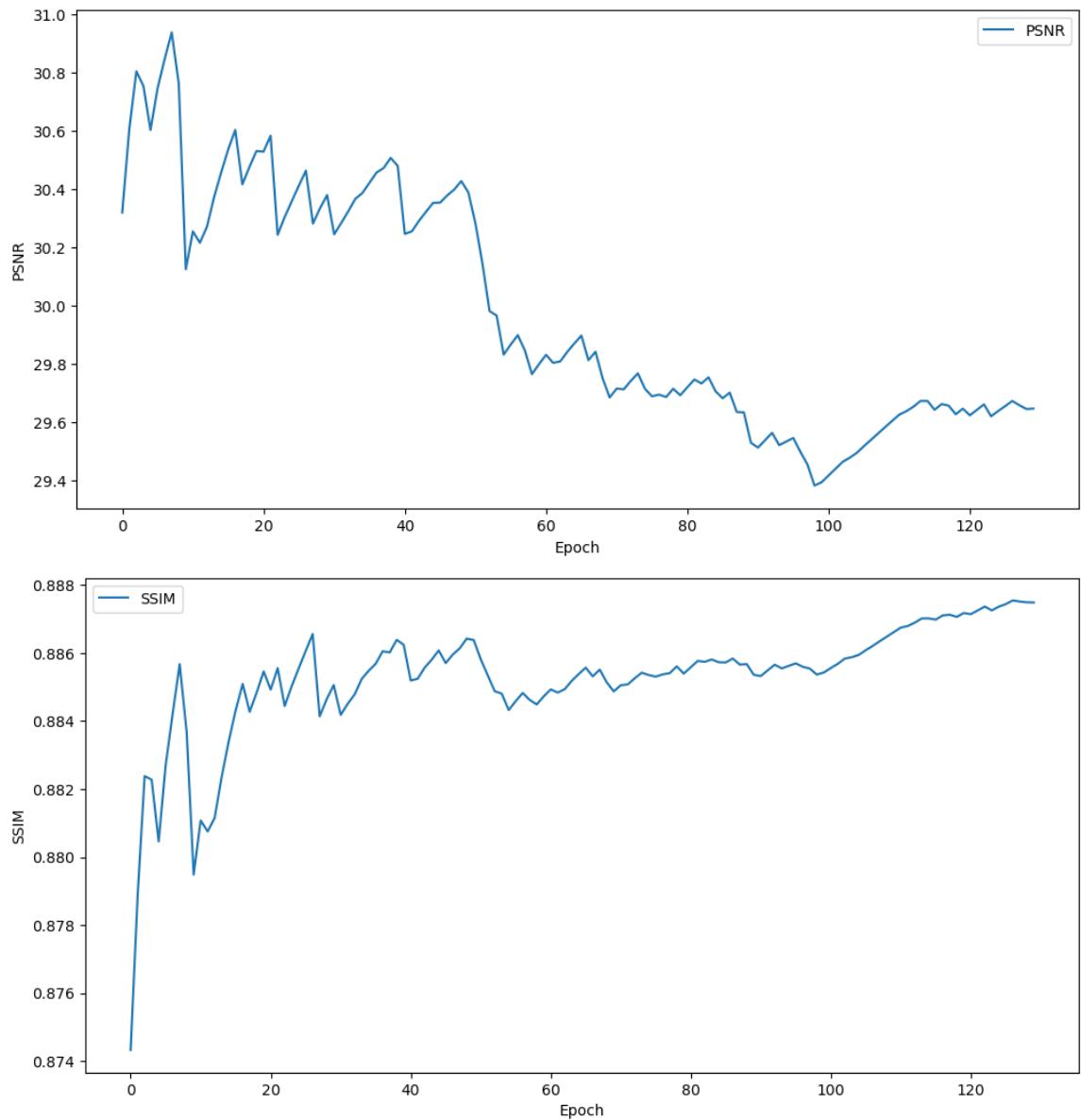


Figure 13: SRResNet PSNR and SSIM Plot for Set5

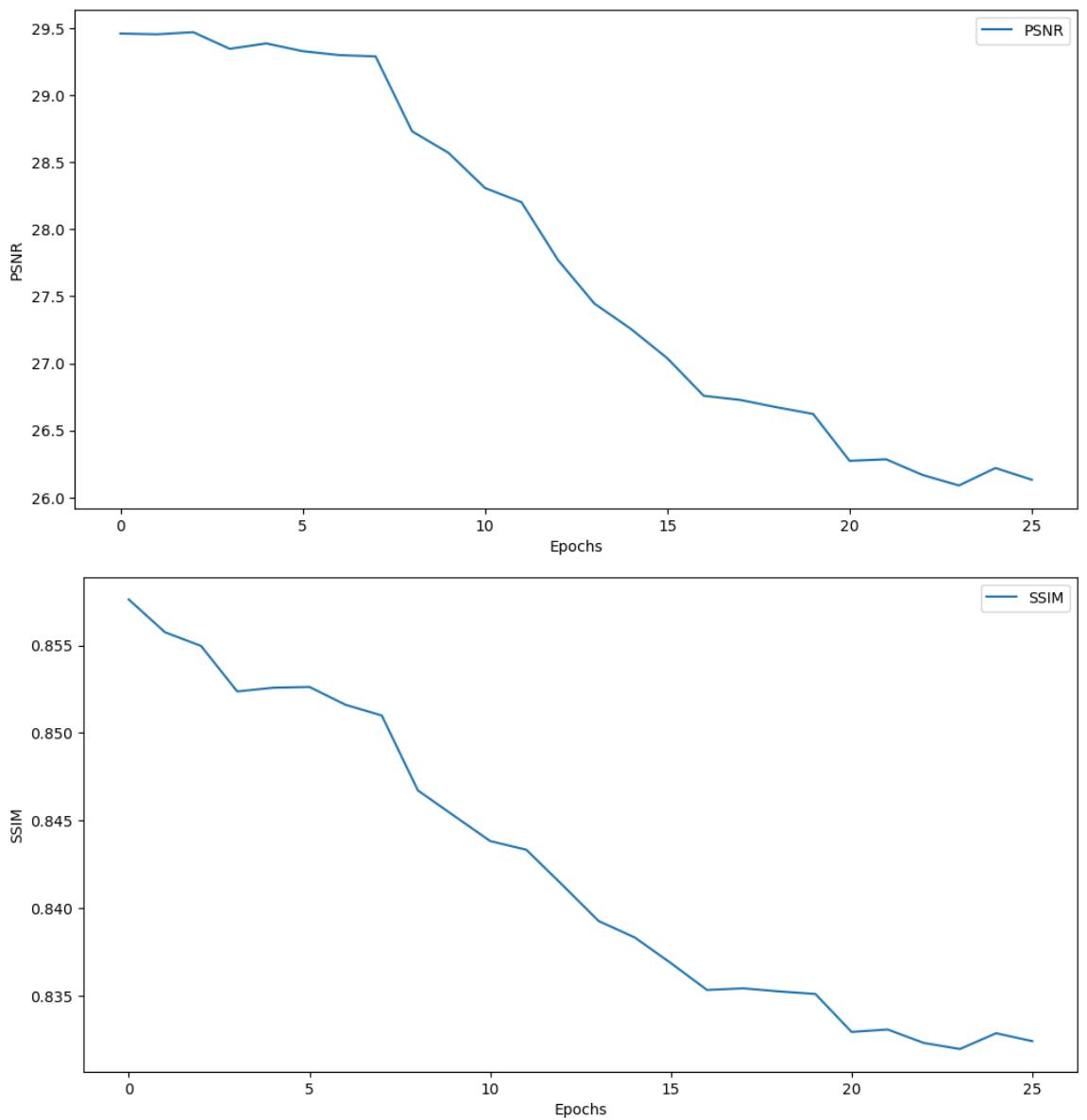


Figure 14: SRGAN PSNR and SSIM Plot for Set5

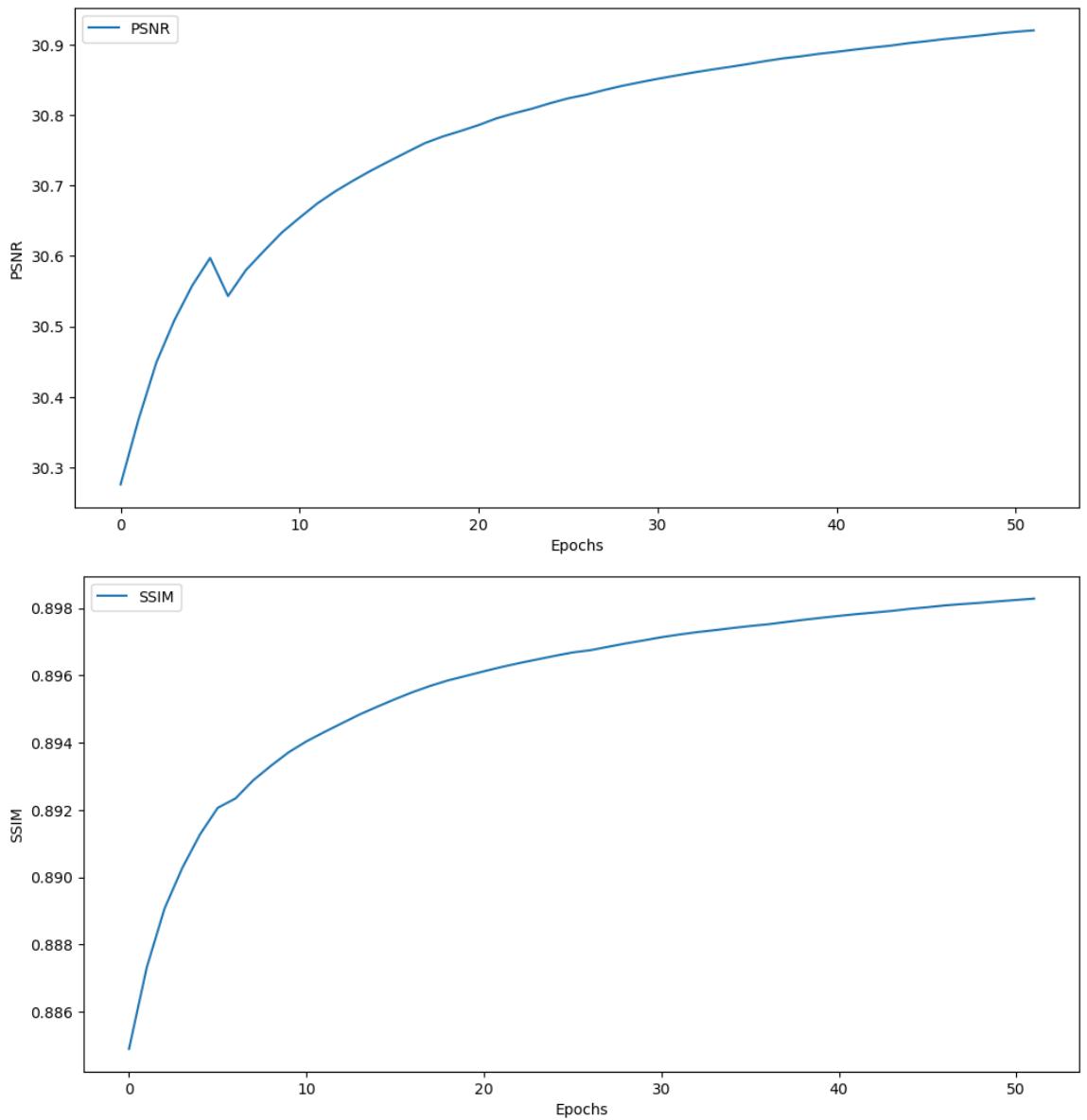


Figure 15: SRCNN PSNR and SSIM Plot for Set5

These metrics don't actually match with human perception but even if PSNR and SSIM is decreasing they are becoming more stable and we see model get better after each epoch.

6.2. Image Comparisons

These are the few results from comparing Bicubic, SRResNet and SRGAN super resolved images with the original HR images.

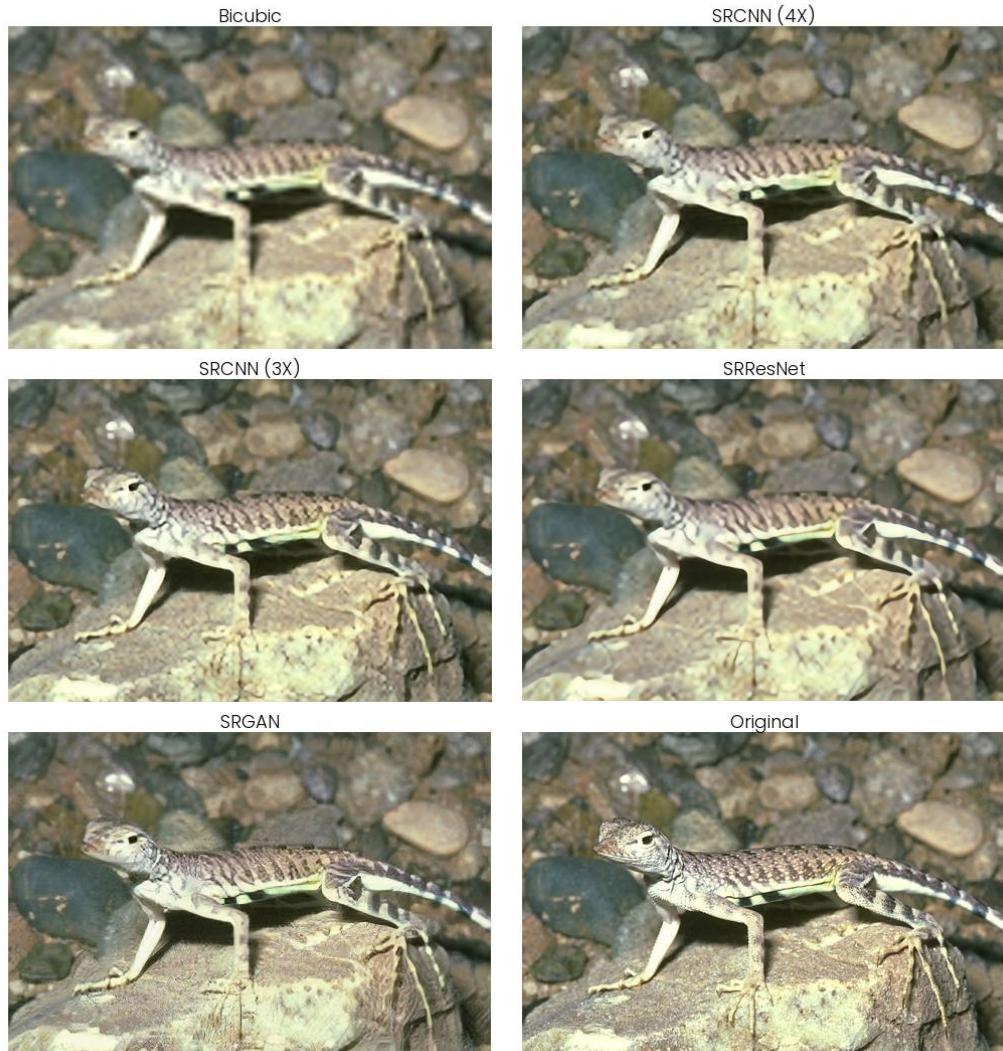


Figure 16: Image of super resolved lizard with SRResNet, SRGAN and HR image



Figure 17: Images of super resolved tiger with SRResNet, SRGAN and HR image



Figure 18: Images of super resolved flowers with SRResNet, SRGAN and HR image

6.3. Recommendation

From our observations, when one needs to do 2x upscaling there is no need for a NN model, We can simply use Bicubic Interpolation if there is enough details present in low resolution image. We found NN models more important and useful when upscaling 3x and 4x. SRCNN models are simple convolutional neural networks, They can't perform in the league of SRResNet and SRGAN when 4x upscaling. But SRCNN model use very less resources and We recommend use SRCNN for upto 3x upscaling. And for SRResNet and SRGAN they are both resource intensive and SRGAN takes very long to train even more than SRResNet. They perform well in 4x upscaling also specially SRGAN. We recommend the use of SRGAN for scaling above 3x.

6.4. Future Enhancements

We can improve our project by doing following things:

1. We only used Y-channel for training SRCNN model, We can use pretrained model in Y-channel and train it in all 3 YCbCr channels for better result.
2. Data is always a limitation for deep learning models, The models will always perform better if we include more diverse images.
3. There is a lot of subjectivity in evaluating the images as quantitative metrics can't identify human perception, We can use Mean Opinion Score like metric that can be used to compare different models and their performance.
4. We can implment ESRGAN and transformer model like xx for better comparison.

References

- [1] D. Han, “Comparison of commonly used image interpolation methods,” in *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013)*. Atlantis Press, 2013/03. [Online]. Available: <https://doi.org/10.2991/iccsee.2013.391>
- [2] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” *CoRR*, vol. abs/1501.00092, 2015.
- [3] W. Yang, X. Zhang, Y. Tian, W. Wang, J.-H. Xue, and Q. Liao, “Deep learning for single image super-resolution: A brief review,” *IEEE Transactions on Multimedia*, vol. 21, no. 12, p. 3106–3121, Dec. 2019. [Online]. Available: <http://dx.doi.org/10.1109/TMM.2019.2919431>
- [4] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, “Photo-realistic single image super-resolution using a generative adversarial network,” *CoRR*, vol. abs/1609.04802, 2016. [Online]. Available: <http://arxiv.org/abs/1609.04802>
- [5] A. F. Agarap, “Deep learning using rectified linear units (relu),” 2019.
- [6] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” 2015.
- [7] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, “Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network,” 2016.
- [8] C. Galea, “Image super-resolution: An overview of the current state of research,” 2022. [Online]. Available: <https://towardsdatascience.com/image-super-resolution-an-overview-of-the-current-state-of-research-94294a77ed5a>