TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PURWANCHAL CAMPUS

COMPARISON OF DIFFERENT IMAGE SUPER RESOLUTION TECHNIQUES

BY:

ABHINAV JAISHI (PUR076BEI001)

MEGHA THAKUR (PUR076BEI021)

SANGHARSHA DAHAL (PUR076BEI029)

KABIN POUDEL (PUR076BEI047)

A MID TERM PROGRESS 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

January 11, 2024

ABSTRACT

Image Super Resolution (ISR) techniques have been developed 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 technique based on deep learning and suggest which works best on different scenarios.

This study conducts a comparative analysis of SRGAN which includes both SRResNet and SRGAN, aiming to assess its performance and characteristics in the context of image super-resolution. The investigation includes an exploration of the SRGAN architecture, training process, and quantitative/qualitative evaluation methodologies. Visual comparisons are made with ResNet and GAN techniques.The study aims to provide valuable insights for enthusiasts and researcher interested in using SRGAN for high-quality image super-resolution.

Keywords: *Super Resolution, GAN, SRGAN, SRResNet*

TABLE OF CONTENTS

List of Figures

List of Tables

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 different Super Resolution Algorithms.
- To compare them to findout their best usecases.

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:

1. Super resolution can be used to improve the resolution of old and degraded images, preserving historical photographs, artworks, and documents.

2. Super resolution can be used in digital cameras and smartphones to provide better zoom capabilities without significant loss of image quality.

3. Super resolution can enhance the resolution of individual frames in videos, leading to improved video quality and better extraction of information.

4. 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-ofthe-art methods of that time.[2]

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.[3]

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:

1. **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.
2. **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.
3. **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.
4. **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 SSResNet and SRGAN till now. Their architecture is explained below:

1. **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.

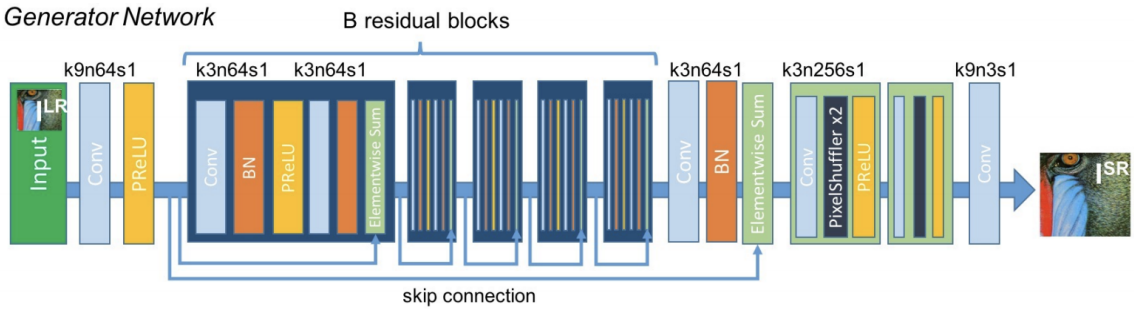

Figure 1: SRResNet Architecture or Generator of SRGAN

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.

2. **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.

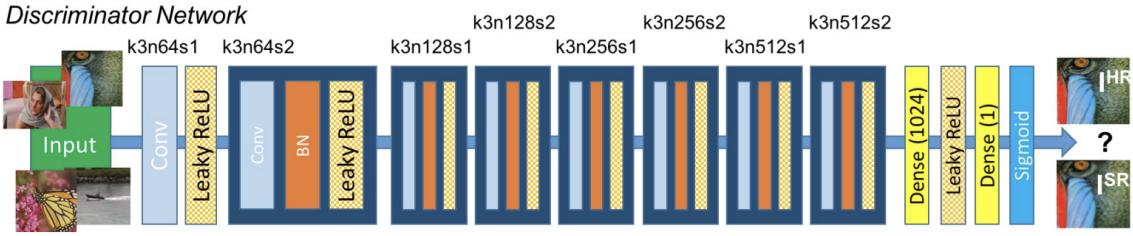

Figure 2: Discriminator Architecture

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 $-logP_{HR}$ when input image is real and $-logP_{SR}$ 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 $-logP_{HR}$ 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.

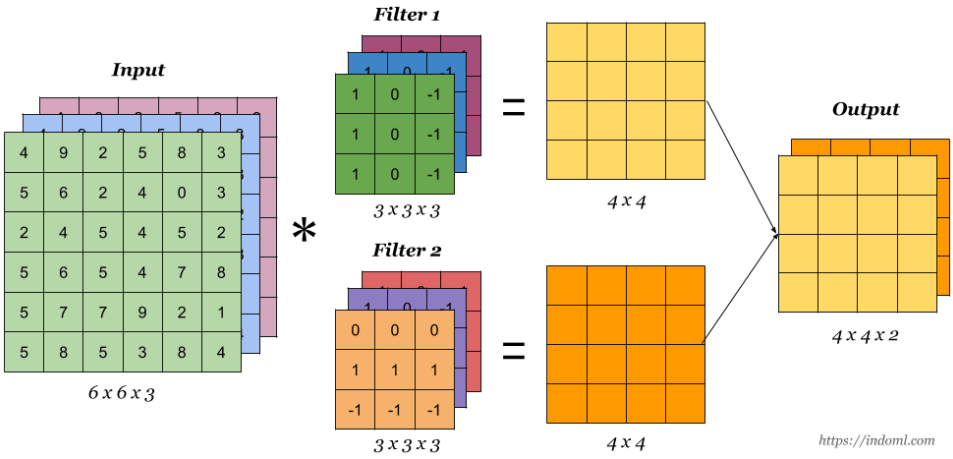

Figure 3: A visual explanation of convolution.

- **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.
- **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 size. 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:

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

   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.

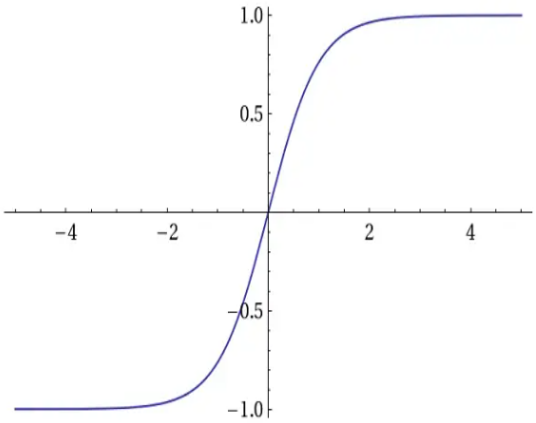

Figure 4: Tanh curve

- **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.

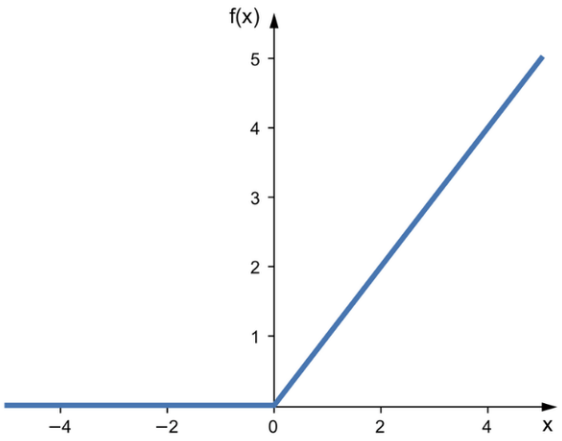

Figure 5: Relu curve

Mathematically it can be represented as:

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

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:

$$f(x) = \begin{cases} x & if\ x \geq 0 \\ 0 & if\ \alpha * x < 0 \end{cases}$$

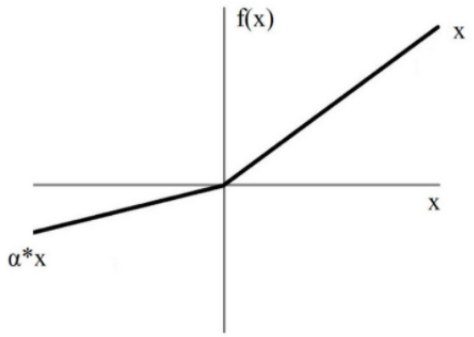

Figure 6: Leaky Relu curve

- **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.

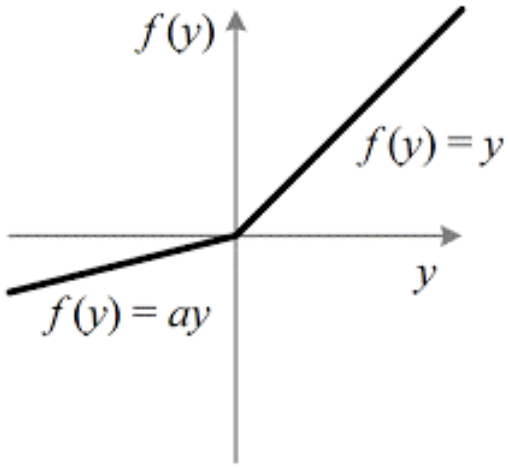

Figure 7: Parametric Relu curve

Mathematically it can be represented as:

$$R(x) = \begin{Bmatrix} x & if\ x \geq 0 \\ 0 & if\ a * x < 0 \end{Bmatrix}$$

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).

   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

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.

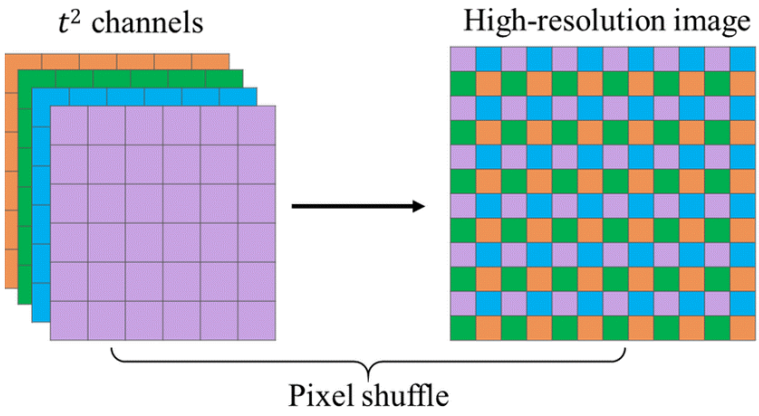

Figure 8: Pixel Shuffle

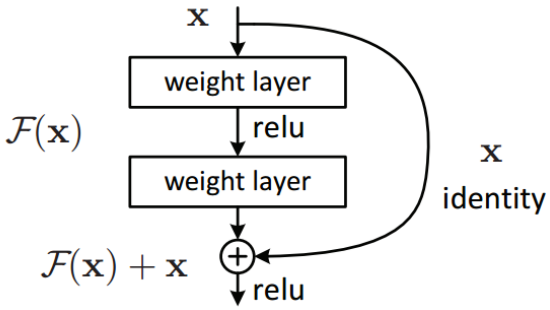

Figure 9: Residual block

5. METHODOLOGY

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

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.

   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 superesolved images.

Figure 10: COCO Set Images

Figure 11: Test Set Images(Set5, Set14, BSDS100)

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.

   (a) Cropping: We need to crop a random patch of image every time. So We create a fucntion 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.

   (b) 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.

3. **Training:** The training process is explained below

   (a) For SRResNet:

      i. Input: Downsampled images of size 24x24.

      ii. Optimizer: Adam optimizer with a learning rate of 1e-4.

      iii. Loss fucntion:Mean Squared Error (MSE) loss between the generated images and the original high-resolution images.

   (b) For SRGAN

      i. Input: Downsampled images of size 24x24.

      ii. Optimizer: Adam optimizer with a learning rate of 1e-4.

      iii. Loss Function:

      Perceptual Loss = Content Loss + 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.

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 :

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

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:

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

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 DISCUSSION

6.1. Current Result

During Training We calculated PSNR and SSIM for Set5 datasets at every epoch for both SRResNet and SRGAN. These metrics don't actually match with human perception but

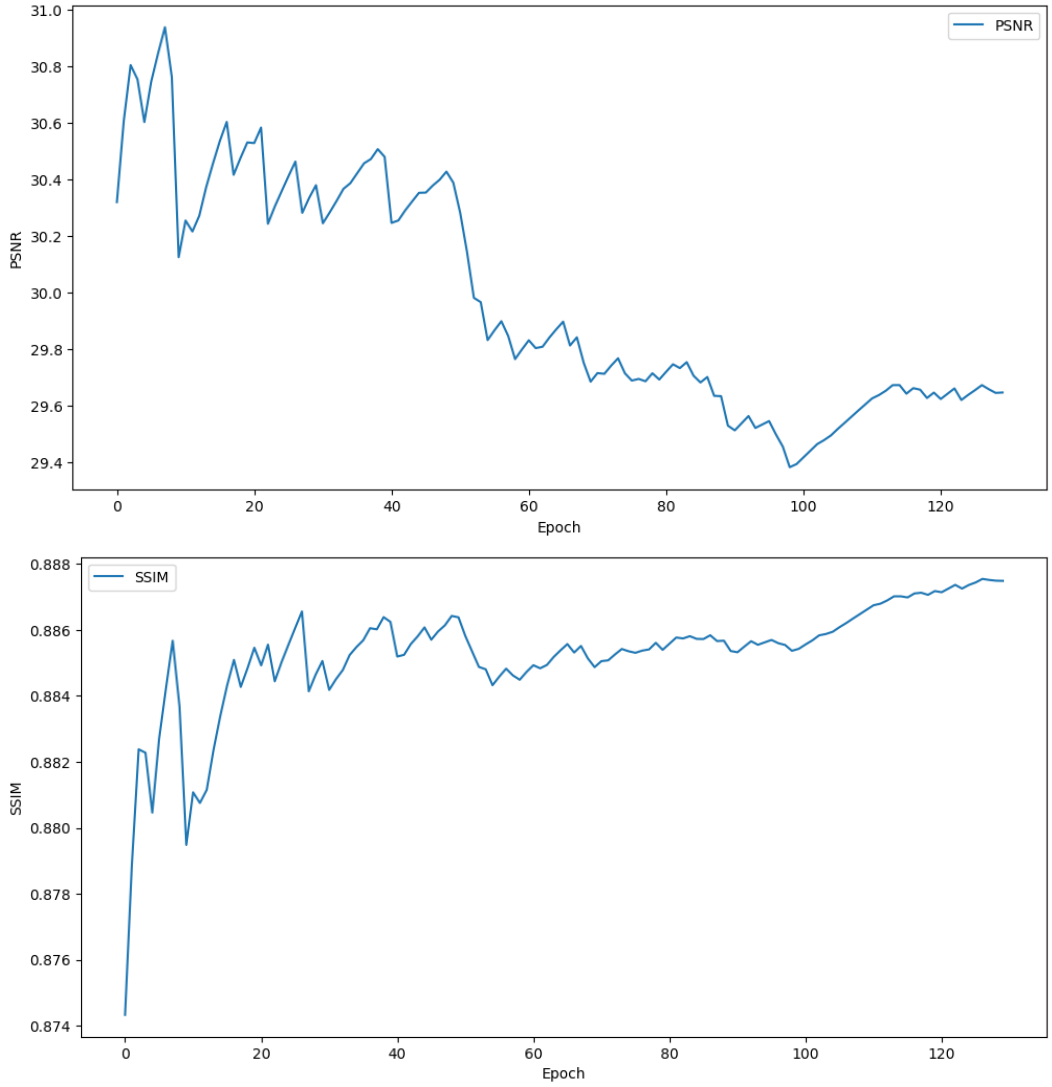

Figure 12: SRResNet PSNR and SSIM Plot for Set5

even if PSNR and SSIM is decreasing they are becoming more stable and we see model get better after each epoch. We have also calculated PSNR and SSIM for all the test datasets Set5, Set14 and BSDS100 after the training was completed.

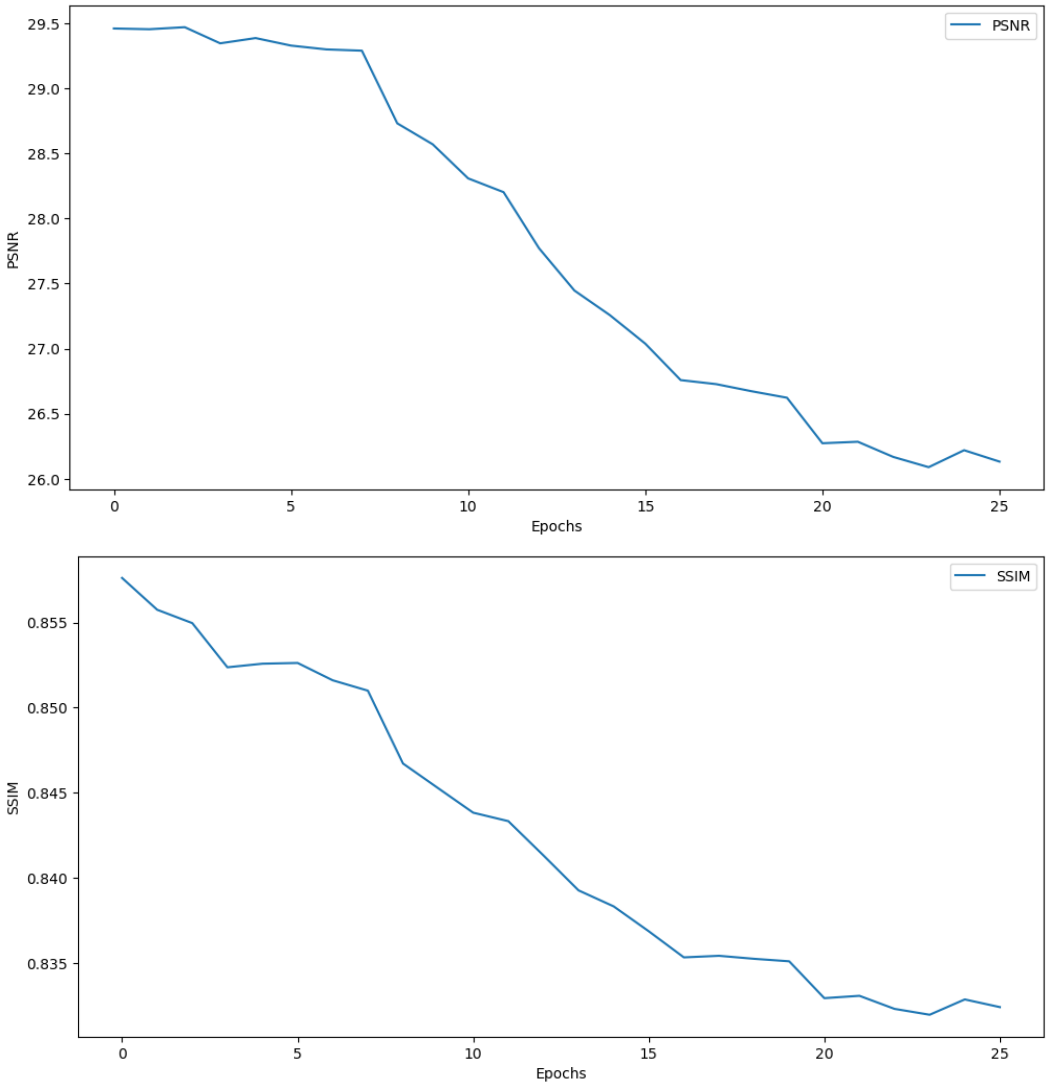

Figure 13: SRGAN PSNR and SSIM Plot for Set5

	PSNR	SISM	PSNR	SISM	PSNR	SISM
ResNet	29.83	0.887	28.275	0.795	27.203	0.751
SRGAN	23.946	0.821	25.112	0.715	24.631	0.662
	Set5		Set14		BSD100	

Table 1: Evaluation Metrics in Test Datasets

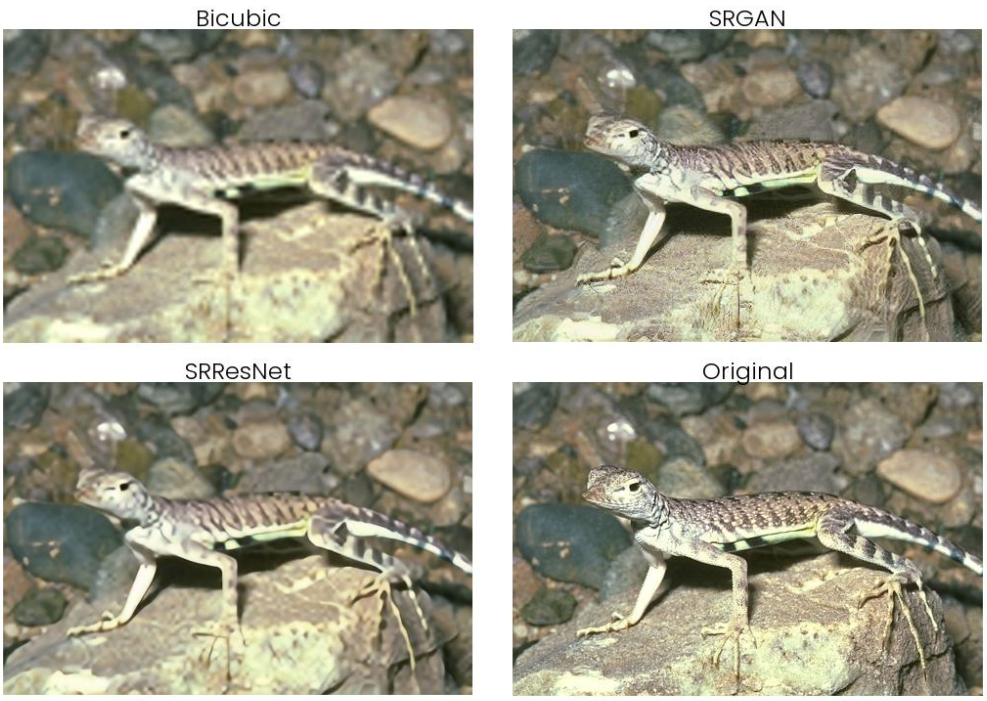

Figure 14: Image of super resolved lizard with SRResNet, SRGAN and HR image

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

Figure 15: Images of super resolved flowers and tiger with SRResNet, SRGAN and HR image

6.2. Work Completed:

As we are planning on comparison of different methods. We have selected few methods like SRCNN, SRGAN, and ESRGAN. SRGAN involves residual network SRResNet which will be also used as baseline for SRGAN. We have trained and tested SRGAN on various sets of data involving CelebA, DIV2k, Flickr2k, OST and COCO(2014). CoCo has a large and diverse set of data and it perfromed well thats why we are finalizing this dataset for training further methods.

6.3. Work Remaining

1. We have not trained models for SRCNN and ESRGAN yet.
2. We don't have a polished website to showcase our results.

6.4. Limitations

Few limitations related to SRGAN that we have trained are:

1. The implementation and training of SRGAN demand substantial computational resources, posing challenges in terms of the time and hardware needed.
2. The effectiveness of SRGAN relies heavily on the availability of a sizable and diverse training dataset, presenting challenges in obtaining and curating such data.
3. There is a lot of subjectivity in evaluating the images as quantitative metrics can't identify human perception.
4. The study lacks comparison with real-world benchmarks or scenarios.
5. Due to constraints, we couldn't explore tinkering hyperparameters.
6. The study lacks a direct comparison with the latest state-of-the-art super-resolution techniques.

6.5. Work Schedule:

Action plan	2023							2024		
	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar
Research										
Data Acquisition										
Model Development and Evaluation										
Model Deployment										
Prepare the report and presentation										

References

[1] Dianyuan 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.

[2] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks, 2015.

[3] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, abs/1609.04802, 2016.