Super Resolution using Generative Adversarial Networks

Summer semester mini project report

By:

Avinash Yadav ITM2016004

Ankit Kumar IRM2016002

Neeraj Mishra IIM2016003

Under the supervision of

Prof. Shirshu Varma

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD

September, 2019

Candidate’s Declaration

We hereby declare that the project work entitled ”Super Resolution using Generative Adversarial Networks” submitted at Indian Institute of Information Technology, Allahabad, is the bonafide work of Avinash Yadav ITM2016004, Ankit Kumar IRM2016002 and Neeraj Mishra IIM2016003.
It is a genuine record of our study carried out from June 2019 till present under the guidance of Prof. Shirshu Varma. Due acknowledgements have been made in the text to all the materials used.

September 6, 2019 Prof. Shirshu Varma, IIIT Allahabad

Table of Contents

Abstract- ***Resolution has always been a problem since the term 'image' came into existence. There have been many researches on improving the resolution of an image but none could create very fast enough methods to get the perfect results that can manage both time and production of satisfactory outputs. So, in this report, we are going to explore about super resolution which has been a problem for many years using signal processing methods. But only now we are able to solve this issue using deep learning techniques. The method we propose here can turn a low resolution image to high resolution ones, upto a pleasant and satisfactory level. We aim to achieve this using GANs (Generative Adversarial Networks) at considerable upscaling factors.***

I. INTRODUCTION

Task of estimating high resolution image from a lower resolution image is referred as super resolution but image downscaling is a lossy process. However good an upscaling algorithm is, there will always be some amount of high frequency data lost from a downscale upscale function performed on an image.Even the best algorithms cannot effectively reconstruct data that does not exist.

Due to more advancement in the deep learning in the recent years especially in the field of multimedia like images, videos, especially depth maps or range images, digital elevation models(DEMS) and multispectral images.The main advancement in the Generative Adversarial Networks (2014) has lead to revolution in the field of super resolution.

Generative models that learn to discover the essence of data and find a best distribution to represent it with a generative model. We draw samples which are not in the training set but which follow the same distribution. Generative Adversarial Net (GAN) proposed in 2014, is able to generate better synthetic images than previous models, and since then it has become one of the most popular research areas.

A Generative Adversarial Net consists of two Neural networks, a Generator and a Discriminator, where the Generator tries to produce realistic images to fool the Discriminator, and the Discriminator tries to distinguish real images from generated ones. So in this project we are using SR-GANS which take low resolution image as an input and produce high resolution images.

II. MOTIVATION

General Techniques used for Super Resolution

Super Resolution can be done by simple algorithmic methods like Bicubic Interpolation, Nearest Neighbourhood interpolation, Bilinear interpolation (BI) et cetera .The report can be found here [1]. These methods either exploit internal similarities of the same image or learn mapping functions from external low and high-resolution exemplar pairs. How ever Deep Neural Networks have proven to outperform all these approaches.

CNNs

Convolutional neural networks date back decades and deep CNNs have recently shown an explosive popularity partially due to its success in image classification. In recent years, convolutional neural network (CNN) based models have achieved great performance in not only SISR task but also many image related tasks. CNN consists of multiple convolutional layers. Each Layer again has multiple filters to apply convolution with.The CNN learns to improve the outputs by updating the values of these filters after each batch. CNN has strong learning capacity and can automatically learn hierarchies feature from training data.The features in lower layers are primitive while those in upper layers are high-level abstract features made from combinations of lower-level features. CNN based models mostly use a pixel wise loss, such as L2 loss. Although the high resolution images constructed by these models are decent, they often tend to lack high-frequency details, especially at a large scaling factor. Furthermore CNNs tend to over smoothen the generated high resolution images.This is because of loss functions like Mean Square Error finds the averages of the pixel wise loss which encourages the model to learn to over smoothen the images to generate the outputs thinking that smoother images better represents the original high resolution images.The main problem with this method is that the sharper details and the high frequency details are lost in the output.The sharp edges are also smoothened.

Generative Adversarial Networks

As we have already seen that Generative Adversarial Networks showed promising results with generating new image data. We can reduce the over smoothening effect and other drawbacks of using SRCNNs and other methods by using a Deep Convolutional Generative Adversarial Network(DCGAN). This can be done by introducing the adversarial loss to the generator model.The loss function can be divided into two parts, the adversarial loss and the content loss .The adversarial loss encourages images that look natural (produce similar results to the distribution) and the content loss makes sure that the new image has similar features to the original low resolution image.

III. LITERATURE REVIEW

Super resolution can be achieved in multiple ways. Early prediction models include interpolation-based methods such as bilinear, bicubic, and Lanczos resampling [5]. Though these methods are fast, they oversimplify the problem and produce overly smooth outputs [4]. Some edge-based methods have been proposed but even they are less effective against high frequency structures such as textures [6]. In 2014, Yang et al. [6] have researched all these techniques and arrived at the conclusion that example-based methods yield the best results. These are machine learning approaches which rely on training data to find a complex mapping from low resolution images to high resolution images.
Recently, Dong et al. [7] have shown that convolutional neural networks (CNN) perform even better. Also, learning upscaling filters proved to be beneficial in terms of accuracy and speed [8]. However, these methods rely on pixel-wise loss functions such as MSE which encourages finding pixel-wise averages of plausible solutions which are typically overly-smooth and thus have poor perceptual quality [9].
In [9] and [4], the authors have overcome the above problem by employing Generative Adversarial Networks [1] to generate images. Ledig et al. [4] propose SRGAN model that uses perceptual loss and adversarial loss to favor outputs residing on the manifold of natural images. The results were very photo-realistic and hence our project aims at performing super resolution using GANs.

1. In 2013, Rujul R. Makwana and Nita D. Mehta published their paper “Survey on Single image Super Resolution Techniques” [2] where they provide a comprehensive review of existing algorithmic super resolution techniques such as Bicubic Interpolation, Nearest Neighbourhood interpolation, Bilinear interpolation (BI) etc.
   From this paper we got familiarized with the different algorithmic super resolution techniques.

2. In 2016, Chao Dong et al. published a paper “Image Super-Resolution Using Deep Convolutional Networks” [3] where they used deep convolutional neural network that takes the low-resolution image as the input and outputs the high-resolution one by learning the end-to-end mapping between the two.

3. In 2014, Ian J. Goodfellow et al. published their paper “Generative Adversarial Nets” [1] in which they proposed a new framework for estimating generative models via an adversarial process where the

generative model is pitted against an adversary: a discriminative model that learns to determine whether a sample is from the model distribution or the data distribution.
This is where we got the idea of implementing Generative Adversarial Networks.

4. In 2017, Christian Ledig et al. in their paper “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network” [4] in which they proposed an SRGAN and claim that their results are closer to those of the original high-resolution images than to those obtained with any state-of-the-art method.

IV. Problem definition

Super resolution has various practical applications like medical imaging [10], satellite imagery [11] and face identification [12] where rich details are greatly desired. Whichever method is adopted, it must be efficient as well as effective. CNN models give a decent result but they are based on pixel-wise loss methods which fail to build multimodal distribution. This problem can be solved by using Generative Adversarial Networks. Hence, in this work, we propose a GAN-based method to produce photo-realistic super resolution on images.

The major tasks or the components of our project are:
- Preprocessing(Processing the low and high resolution images and preparing the training data).
- Building the Discriminator model which discriminates real and generated images.
- Building the Generator model which tries to generates the high resolution images and integrating it with the Discriminator model.

The Discriminator is going to be a deep convolutional model which tries to classify the images as Real or Generated(fake).The Model consists of an input layer,an output layer along with multiple hidden layers. Usually these hidden layers consists of convolutional layers, pooling layers, fully connected layers and normalization layers.Convolution is a process of reducing the input size to a smaller one by convolving the image with a filter with a particular size and a particular stride.The size of the output of a hidden layer or the size of the input of next hidden layer is determined by the filter size and stride size.

The Generation model is again a deep convolutional model which tries to generate a High resolution image from the given low resolution image.This model also has an input layer,an output layer and several hidden layers.We use some convolution layers initially and then resize the image and again further apply the convolution layers to it to generate the High resolution image.The Initial convolution layers applies the main features to the image like the colours and other features and the later convolution layers which comes after the resizing probably applies sharper and important details to the image helping it to gain all the important details.

V. Proposed methodology

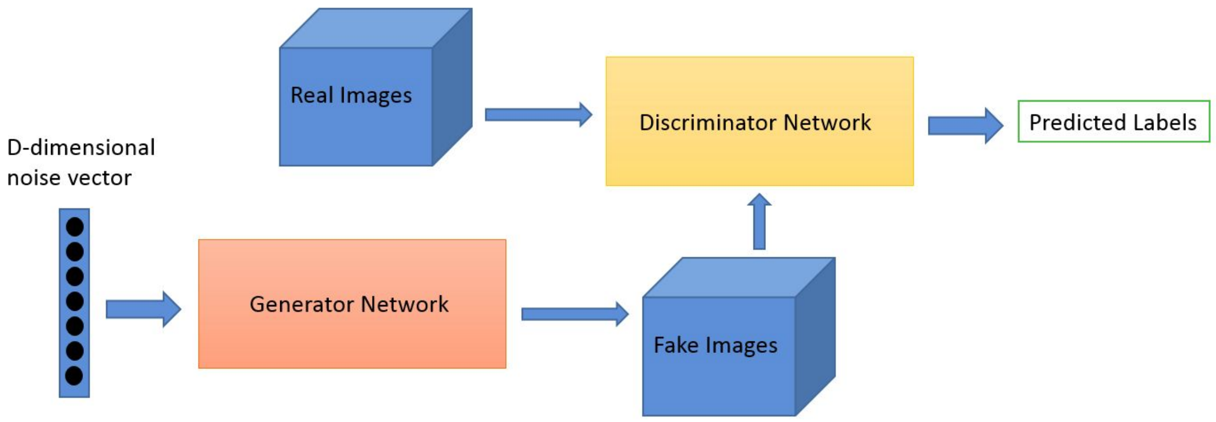
Generative Adversarial Network

A Generative Adversarial Network mainly consists of two components:
- The Generator (G)
- The Discriminator (D)

The Generator: Given a dataset of similar images, the function of the generator is to produce new similar looking images. It takes in random noise as input and produces a corresponding output with help of a deconvolutional neural network based on some randomly initialized weights and biases. Depending on whether the output is valid or not, the generator adjusts its parameters.
But this requires us to manually provide it the feedback whether the generated image is a plausible output or not. This is where the discriminator comes in and automates the process

The Discriminator: The discriminative network takes in inputs from both the original dataset as well as the generated images and tries to distinguish the real images from the generated images. This can be achieved using a simple classifier consisting of a convolutional neural network.

We initially train the discriminator (D) upto some accuracy and then alternate between generated images and real images. The objective of the generator (G) is to maximize the error rate of D.



The key idea is to backpropagate gradients from the results of D's classification to G, so that G gets better at producing fabricated images that can fool D and D gets better at flagging generated images.

Using GANs for Single Image Super Resolution

As we have already seen how Generative Adversarial Nets work,we are going to apply this method to our model for performing Single Image Super Resolution.

The clear description of how actually the model is going to look like is shown in the Figure below. In the Figure below LR stands for Low Resolution Image, HR for corresponding original High resolution Image and SR stands for the Super Resolution output image which is the output of the Generator(G). As we can see, the final loss of the discriminator which is the sum of its loss of the Discriminator Network(D) for both HR and SR images. This final loss is used to update the weights of both the Generator(G) and the Discriminator(D) Networks.

Architecture
Following is the architecture of the model that we used for performing super-resolution

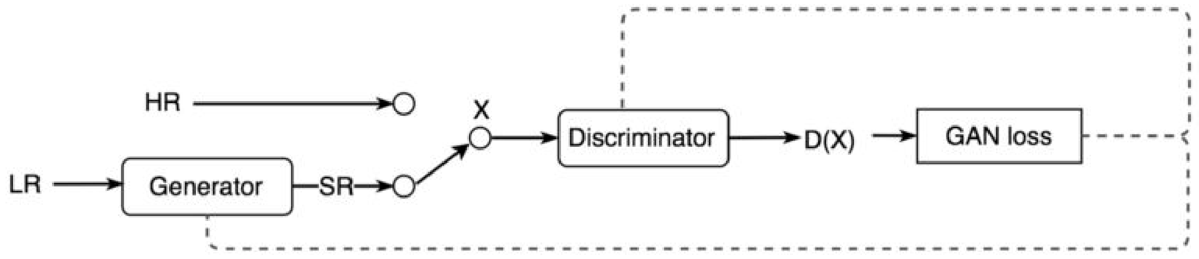


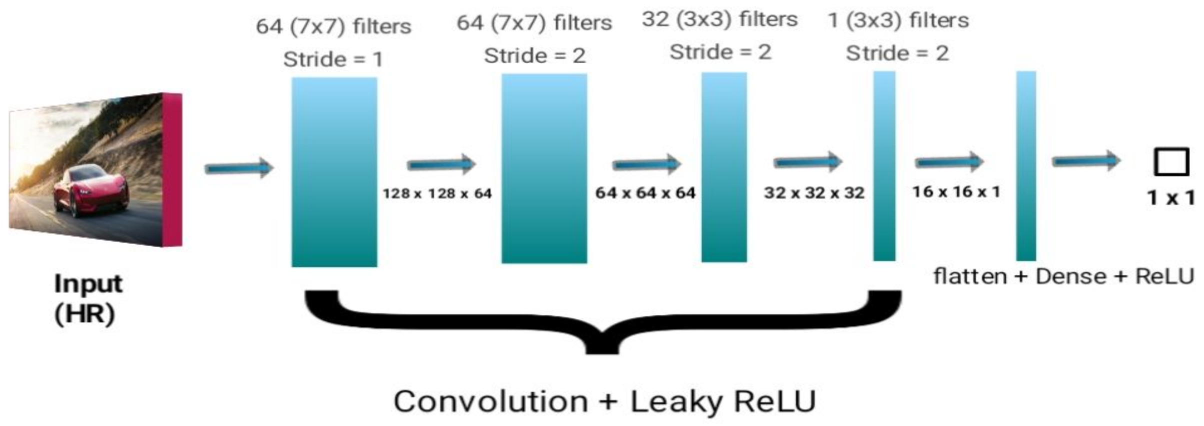
Table 1: Generator

Layer	Filters	Kernel Size	Activation Function
Convolutional Layer 1	32	3x3	Leaky Rectified Linear Unit
Convolutional Layer 2	128	3x3	Leaky Rectified Linear Unit
Convolutional Layer 3	128	3x3	Leaky Rectified Linear Unit
Resize Image			
Convolutional Layer 4	128	3x3	Leaky Rectified Linear Unit
Convolutional Layer 5	64	3x3	Leaky Rectified Linear Unit
Convolutional Layer 6	3	3x3	Sigmoid Activation Layer

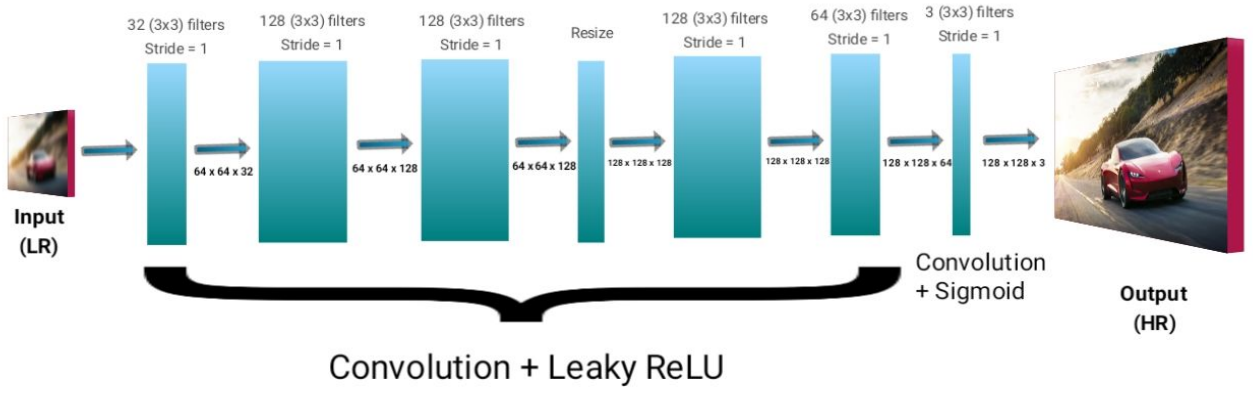
Table 2: Discriminator

Layer	Filters	Kernel Size	Activation Function
Convolutional Layer 1	64	7x7	Leaky Rectified Linear Unit
Convolutional Layer 2	64	7x7	Leaky Rectified Linear Unit
Convolutional Layer 3	32	3x3	Leaky Rectified Linear Unit
Convolutional Layer 4	1	3x3	Leaky Rectified Linear Unit
Densely Connected Layer	-	-	Rectified Linear Unit

As already discussed previously, the discriminator converts the 128x128x3 image(3 implies the RGB channels of the image) into a 16x16x1 matrix which is used to determine if the image is real or generated.We are using leaky ReLU as out activation function for all the hidden layers except for the last layer which is a fully connected dense layer.The leaky ReLU activation function is defined as $max\ \{x, threshold * x\}$ .For our model we set the threshold as 0.01 it is simply a design choice. The architecture of our discriminator model can be seen below.



The generator model on the other hand converts the low resolution 64x64x3 images to 128x128x3.This model has 3 convolution layers, then we resize the outputs of the third layer and then further apply 3 more convolution layers on the resized image.The first three layers applies the important and generic details to the image and the last three layers probably add the sharper details like edges and sharp changes in the colors to the resized image.We are using Leaky ReLU as the activation function for all the hidden convolutional layers except the last layer for which we are using sigmoid activation function.The sigmoid activation function is defined as S(x) = $\frac{1}{1 + e^{-x}}$ .The main reason we are using sigmoid function is that it maps the values of the layer to be between (-1, 1). The architecture of our generator model can be seen below.



5.2 Why Does This Work?

It is now a fact that the filters(kernels) are the main reason behind the working of any CNNs.For the classification tasks,the filters are trained to detect the key features of the input datasets.However in the generative tasks the filters are trained in such a way that when convolution is done with them,they adds more features to already existing inputs.What exactly each filter in the Neural Network does to the input is still not clear.Filters are still a black box in CNNs.The best we can do is to visualize what each filter looks like.We can not predict what exactly each filter is going to learn.However some studies show that the features in lower layers are primitive while those in upper layers are high-level features made from combinations of lower-level features.

The main comparison to be made is between CNNs and GANs in the task of super resolution.The main differenceis that the Generative Adversarial Networks use a better loss function than simple CNNs.It can be said that it is a better loss function as it not only includes the content loss like in CNNs but also has a loss called Adversarial Loss.This is the loss that comes from the adversary of the generator which is discriminator.This simply implies that the generator does not directly learn from the images but also from what the discriminator thinks about the generated output.This makes the generator generate images which are not only closes to the original images in terms of pixel wise difference but also tries to generate images which are acceptable by the discriminator network.Further discussion about the Loss functions is done below

Generator Loss Function

The loss function of the generator is divided into two parts :

1. Content Loss
2. Adversarial Loss

Content Loss

The Content loss mean of absolute value of the pixel wise difference between the generated image and the original high resolution image.This is simply calculating pixel wise difference and then computing the mean on absolute values of all the pixel difference.Mathematically,

$$Content\ loss\ =\ mean(abs(g - \ highres))$$

where,

g is the generated image ,and

Highres is the actual high resolution image

Adversarial Loss

Now to the above content loss,we add another term to it

$$Adversarial\ Loss\ =\ 0.1\ \times\ mean(z\ *\ -log(sigmoid(x))\ +\ (1\ -\ z)\ *\ -log(1\ -\ sigmoid(x)))$$

Where,

$x = d_{fake}$, and

$z\ = d_{label}$

The term d_{fake} is the output of the discriminator when a generated input is given to it. d_{label} is 1. The multiplication factor 0.1 is again a value of choice. It can be anything but we chose to take it 0.1

Total Loss

Let x be d_{fake} which is the output of the discriminator when a generated(fake) image is given to it.Let z' be the sum of adversarial loss and content loss .The term $z * -log(sigmoid(x)) + (1 - z) * -log(1 - sigmoid(x))$ is also called as Cross entropy with Sigmoid activation function.

$$x = d_{fake}$$
$$z = d_{label}$$
$$Generator\ loss = Adversarial\ loss + Content\ loss$$

$$\begin{aligned}
&Generator\ loss \\
&= 0.1 \times mean(z * -log(sigmoid(x)) + (1 - z) * -log(1 - sigmoid(x))) + mean(abs(g - highres)) \\
&= 0.1 \times mean(z * -log(\tfrac{1}{1+e^{-x}}) + (1 - z) * -log(\tfrac{e^{-x}}{1+e^{-x}})) + mean(abs(g - highres)) \\
&= 0.1 \times mean(z * log(1 + e^{-x}) + (1 - z) * (- log(e^{-x}) + log(1 + e^{-x}) + mean(abs(g - highres)) \\
&= 0.1 \times mean(z * log(1 + e^{-x}) + (1 - z) * (x + log(1 + e^{-x}) + mean(abs(g - highres)) \\
&= 0.1 \times mean((1 - z) * x + log(1 + e^{-x})) + mean(abs(g - highres))
\end{aligned}$$

Discriminator Loss function

Total Loss

The overall discriminator loss is again sum of two losses $d_{LossReal}$ and $d_{LossFake}$ .Let d_{real} be the output of the discriminator when the input is a real(High resolution image) and d_{fake} is the output of the discriminator when the input was a generated image.Let $d_{LabelReal}$ be 1 $d_{LabelFake}$ be 0.

$$d_{LossReal} = mean(d_{LabelReal} * -log(sigmoid(d_{real})) + (1 - d_{LabelReal}) * -log(1 - sigmoid(d_{real})))$$
$$d_{LossFake} = mean(d_{LabelFake} * -log(sigmoid(d_{fake})) + (1 - d_{LabelFake}) * -log(1 - sigmoid(d_{fake})))$$
$$\text{Discriminator loss} = d_{LossReal} + d_{LossFake}$$

Experiments

Similarity Metrics

There are several metrics for quantifying the similarity of two images. The PSNR (Peak Signal to Noise Ration) and SSIM (Structural Similarity Index) are two popular techniques to compare two images.PSNR(Peak Signal To Noise Ratio) computes the mean squared reconstruction error after denoising. Higher PSNR means more noise removed. However, as a least squares result, it is slightly biased towards over smoothed or blurry results i.e. This algorithm removes the noise and also a part of the textures will have a good score.

$$PSNR = 10log_{10}(\tfrac{R^2}{MSE})$$

Where R is the maximum possible pixel value in the image. For example, when the pixels are represented using 8 bits per sample, this is 255 .($2^{\text{number of bits}}$ - 1).MSE is the Mean Square Error.

On the other hand SSIM has been developed to have a quality reconstruction metric that also takes into account the similarity of the edges (high frequency content) between the denoised image and the ideal one. To have a good SSIM measure, an algorithm needs to remove noise while preserving the edges of the objects.
Hence in this paper, we choose SSIM to compare the different super-resolution techniques.

Training details and parameters

We trained our models on an NVIDIA GeForce GTX 920M on a random sample of 10000 images from the imagenet dataset.We resized all the images to dimensions 128x128x3 which we uses as the high resolution images for training our models.We resized these 128x128x3 images to 64x64x3 and used them as inputs to our generator model.We have used the loss functions as described in the previous sections.For both the generator and discriminator models we used Adam optimizer with beta1 as 0.9 and learning rates 0.0001 as the optimizing function for both the generator and the discriminator models.We trained our models for 10 epochs with a batch size of 9 images.The training took approximately 9 hours on the GPU.The generator loss went from around 0.506074429 at the start of the training to For testing we are using images from several benchmark datasets like SET5,SET14,URBAN100 etc.We compute the average value of SSIM of these test images as the metric of super resolution.

Performance of the final networks

Table 3: Average SSIM values on different datasets by various algorithms and SRGAN in case of 2x scaling

Dataset	Bilinear	Bicubic	Lanczos	Nearest	SRGAN
BSD100	0.8398	0.875	0.887	0.853	**0.9093**
SET14	0.8399	0.8787	0.8916	0.8519	**0.9101**
SET5	0.8886	0.9207	0.9313	0.8884	**0.9363**

Table 4: Average SSIM values on different datasets by various algorithms and SRGAN in case of 3x scaling

Dataset	Bilinear	Bicubic	Lanczos	Nearest	SRGAN
BSD100	0.7399	0.7657	0.7784	0.7355	**0.7976**
SET14	0.7535	0.7832	0.7972	0.7439	**0.809**
SET5	0.8266	0.8531	**0.8652**	0.8055	0.865

This is the table comparing input image size vs Time taken to perform super resolution with an upscaling factor of 2x.Note: These are the results for running the model on a CPU and not a GPU.The values might differ in different CPUs as well.This table is given just to show the increasing time for performing super resolution with the increase in the size of input image

Table 5: Input size vs. Execution time

Input Size	Time Taken(secs)
50x50	0.502456665
100x100	1.098718166
150x150	2.181693316
200x200	3.908487797
250x250	5.777350664
300x300	8.004456758
350x350	10.86392927
400x400	14.07040524

How can Super Resolution be Applied to Videos

Video super resolution can be done by adding a small extension to the single image super resolution.We divide the video(Low Resolution Video) into frames and record the frame rate of the video by computing frames per second of the video.Then we apply single Image super resolution on each and every frame and generate the SR for each frame.Then we simply stitch the SR frames at a frame rate that we recorded before splitting the video into frames.

Advantages and Disadvantages of GANs

Advantages

- Many Convolutional Neural Networks use only content loss to updates the weights of their models. As GANs use both content loss and adversarial loss for this work, they are better.
- GANs do not require huge datasets or Labeled datasets. It's easy to find datasets for GANs.
- They don't try to over smoothen the images, they generate the image sharp and crisp.

Disadvantages

- For training GANs we need to find the Nash Equilibrium of a game. Sometimes gradient descent can find it and sometimes it may not. Still a good algorithm for this isn't developed yet.
- It becomes difficult for GANs to generate data when it is given a discrete dataset
- As GANs require two models namely Discriminative model and Generative model, it is memory heavy to train these two models at the same time.
- Time is also an important disadvantage in GANs, because as said before, we have two models to train, it may take many several hours to be completed.

Datasets

Training Dataset

We used The Validation Set from ILSVRC2012(Image Recognition Challenge).This Set Contains 50000 randomly selected Images from Imagenet dataset.We are only using 10000 of these images for our model.

Testing Dataset

Our Testing dataset contains images from various datasets like

- Set5
- Set14
- BSD100
- Urban100
- Manga109
- Historical

VI. HARDWARE, SOFTWARE REQUIREMENTS

Software Requirements:

1. TensorFlow-GPU :

Tensorflow is an software library for numerical computation. First we define the nodes of the computation graph, then inside a session, the actual computation takes place. TensorFlow is widely used in Machine Learning.

2. Open CV:

OpenCV(Open Source Computer Vision) is an open source library of programming functions used for real-time computer-vision. It is mainly used for image processing, video capture and analysis for features like face and object recognition.

3. Pillow(PIL):

Pillow is a fork of PIL (Python Image Library).This library can be used to manipulate images quite easily.PIL is widely used for Image Processing and manipulation along with openCV.

4. SciPy:

SciPy is an open-source software for mathematics, science, and engineering. The SciPy ecosystem includes general and specialised tools like Numpy, Matplotlib, IPython, Sympy for data management and computation, productive experimentation and high-performance computing.

5. Numpy:

NumPy is the fundamental package for scientific computing with Python.Numpy arrays are faster and occupies less space than regular python lists.Since deep learning is a memory heavy job,using Numpy is a must.We mainly use Numpy while converting the input data into tensors which is required for training in Tensorflow.

6. Scikit-image:

Scikit-image is an image processing Python package that works with numpy arrays it is an efficient tool for image processing.

Hardware Requirement:

GPU(Dedicated Memory of more than 2GB) :

Tensorflow runs on more efficiently on a GPU compared to a CPU. As we are working on image data and it requires a lot of space and all the tensors and the numpy arrays are stored in the GPU, a minimum dedicated memory of 2GB is required.

VII. Implementation Plan & Timeline

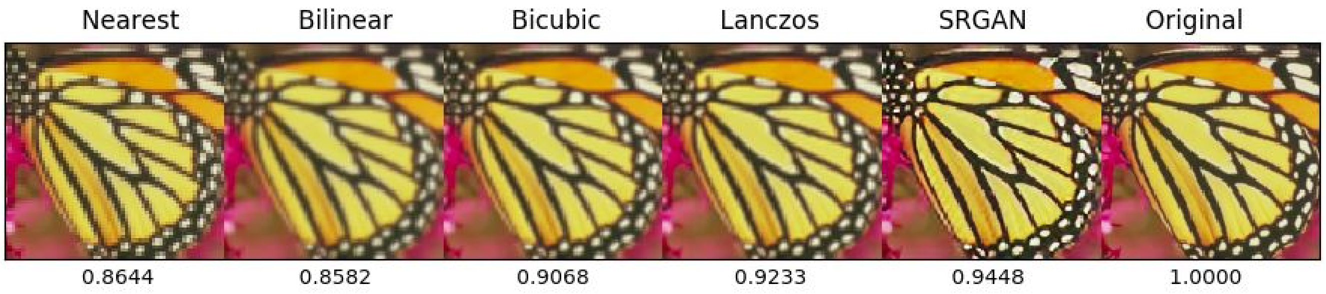
Table 6: Project Timeline

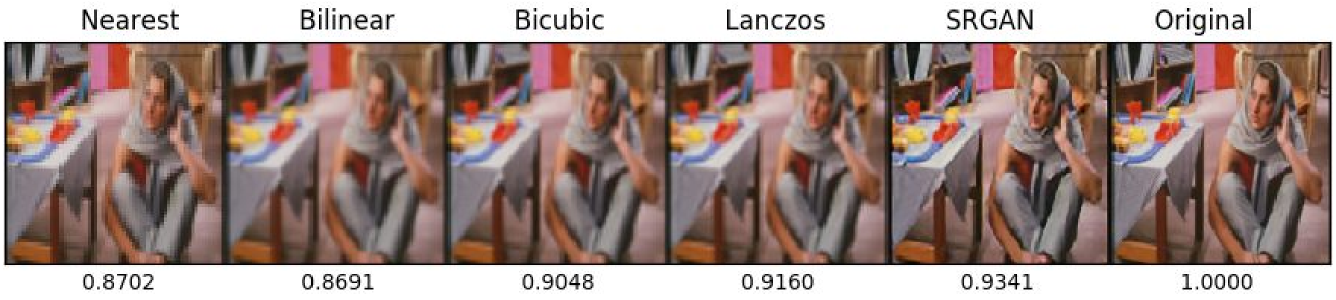
Project Activity	Duration
Literature Review	Week 1 - Week 2
Familiarization with GAN and finding the dataset	Week 3 - Week 4
Built the Discriminator (D) and Generator(G)	Week 5 - Week 7
Trained the model with various inputs and parameters	Week 8 - Week 10

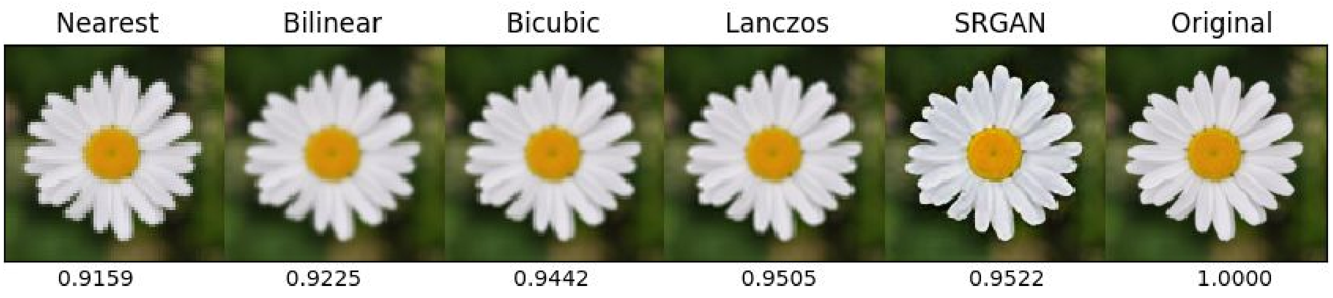
1. We Acquired the datasets and preporsessed the inputs as numpy array files.
2. We created the models for discriminator and generator.
3. We trained the models on the training dataset several times changing the training parameters till we achieved required results.
4. We then tested our model on the images from the testing datasets and generated quantitative results using the similarity metrics.

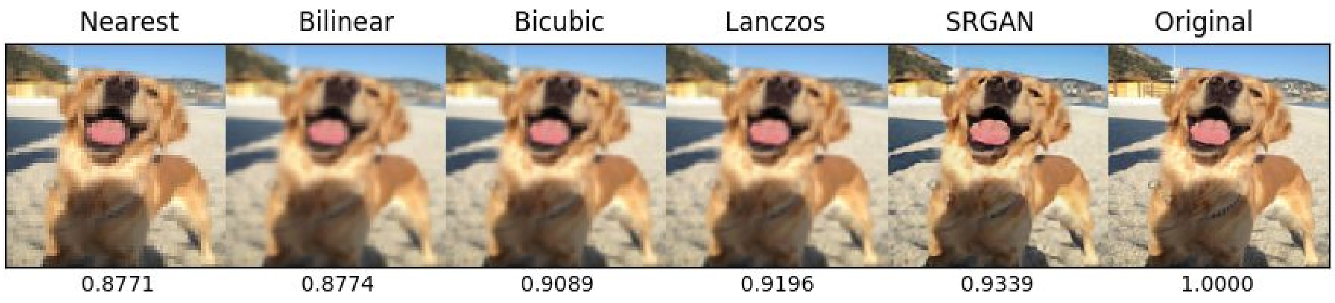
VIII. Results

The results obtained are photo-realistic and are better than interpolation techniques such as nearest neighbour, bilinear, bicubic and lanczos. Following is an example for the same:









SSIM scores for different interpolation techniques

These results illustrate that super-resolution achieved using GANs are clearly better than the current algorithmic interpolation techniques like bicubic, lanczos etc.

This model can also be applied on videos. We split the video into frames and store them in a temporary location and perform super resolution on each of these frames and stitch the higher resolution frames back to create a video file.

IX. CONCLUSION

- In this project, we have proposed a Super Resolution GAN which produces photo-realistic images for 2x scaling and our model performs better than current approaches
- We confirmed the superior perceptual performance of GANs for the task of Super Resolution using Similarity Metrics like SSIM.
- GANs are very powerful and have a lot of potential. We can improve image quality of higher scaling factors by tweaking the model and training on more powerful GPUs

REFERENCES

[1] Ian J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. *Generative adversarial nets*. In Advances in Neural Information Processing Systems (NIPS), pages 2672–2680, 2014.

[2] Rujul R. Makwana, Nita D. Mehta "Survey on Single image Super Resolution Techniques" IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)e-ISSN: 2278-2834,p- ISSN: 2278-8735.Volume 5, Issue 5

[3] Dong, C., Loy, C.C., He, K.,et al.:'Image super-resolution using deep convolutional networks', Trans. Pattern Anal. Mach. Intell., 2016, 38, (2), pp. 295–307, doi: 10.1109/tpami.2015.2439281

[4] L.C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z.Wang, et al. *Photo-realistic single image super-resolution using a generative adversarial network*. arXiv preprint arXiv:1609.04802,2016.

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

[6] Yang, C.Y., Ma, C., Yang, M.H.: Single-image super-resolution: A benchmark. In: European Conference on Computer Vision, pp.372–386 (2014)

[7] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(2):295–307, 2016.

[8]. Dong, C. C. Loy, and X. Tang. Accelerating the super-resolution convolutional neural network. In European Conference on Computer Vision (ECCV), pages 391–407. Springer, 2016.

[9] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In International Conference on Learning Representations (ICLR), 2016

[10] W. Shi, J. Caballero, C. Ledig, X. Zhuang, W. Bai, K. Bhatia, A. M. S. M. de Marvao, T. Dawes, D. ORegan, and D. Rueckert. Cardiac image super-resolution with global correspondence using multi-atlas patchmatch. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 9–16. Springer, 2013.

[11] M. Thornton, P. M. Atkinson, and D. Holland. Sub-pixel mapping of rural land cover objects from fine spatial resolution satellite sensor imagery using super-resolution pixel-swapping. International Journal of Remote Sensing, 27(3):473–491, 2006

[12] E. Bilgazyev, B. Efraty, S. K. Shah, and I. A. Kakadiaris. Improved face recognition using super-resolution. In Biometrics (IJCB), 2011 International Joint Conference on, pages 1–7. IEEE, 2011.

Suggestions by the Board