MINOR FINAL PROJECT REPORT
15B19CI691
Supervisor : **Aparajita Nanda**

DCSRGAN

Dimensionally Invariant Color Bounded Super-Resolution Generative Adversarial Networks

GROUP G117 - RAJAT GOEL 18103108 B3, SARTHAK SHARMA 18103112 B3, SHREYA GROVER 18103113 B3

Deployed at github.com/grajat90/ResampleGAN

INTRODUCTION

Image resampling is the procedure of converting a sampled image from any coordinate structure to another structure. When forgeries are introduced in digital images, generally the operations like rotation, resizing, skewing etc., are included to form it relational with reference to the adjacent original area. So there is a recognisable loss within the standard of the image and it will become an important signature of manipulated images.

Image resampling finds uses in many fields. In computer graphics, it allows texture to be applied to surfaces in a CG imagery, without the need to explicitly model the texture. In medical and remotely sensed imagery it allows an image to be registered with some standard coordinate system.This is primarily done to organize for further processing.

The highly challenging task of estimating a high- resolution (HR) image from its low-resolution (LR) counterpart is mentioned as super-resolution (SR). SR received substantial attention from within the pc vision research community and features a wide selection of applications.

In Spite of varied studies with accurate and faster single image super-resolution using deep convolutional neural networks, there still remains a problem: How can we recover the finer texture details once we super-resolve at large upscaling factors? The behaviour of optimisation-based super-resolution methods is principally driven by the selection of the target function. Recent works are largely focused on minimizing the mean squared reconstruction error which ends up in high peak signal-to-noise ratios, but they're often lacking high-frequency details.

In this work, we propose Dimensionally Invariant Colour Bounded Super-Resolution Generative Adversarial Networks (DCSRGAN) that we employ a deep residual network. Our deep residual network is in a position to recover photo-realistic textures from heavily downsampled images on public benchmarks. The adversarial loss pushes our solution to the natural image manifold employing a discriminator network that's trained to differentiate between the super-resolved images and original photo-realistic images. We've used a perpetual loss using high-level feature maps of the VGG network combined with a discriminator that encourages solutions perceptually hard to differentiate from the HR reference images. Different from previous works, we introduce a novel perpetual loss, colour loss to assist recover finer texture details.

This loss works to bound the mutations and learnings the network can do during the training period. It was observed during the training of the traditional SRGAN network that it has a tendency to diverge away from the original colours used to some different colours. Such a mutation can arise due to the fact that certain colour palettes can appear “real” to the discriminator and the VGG network might still be able to recognise similar features in such images. Hence a colour loss bound where the network can apply the gradients at the time of training. In our case, the colour loss is the binary cross-entropy error between the original image and the generated one.

Deep learning-based methods have recently pushed the state-of-the-art on the matter of Single Image Super-Resolution. However, in this project, we also revisit the more traditional interpolation-based methods and compare them against our solution to achieve the best results.

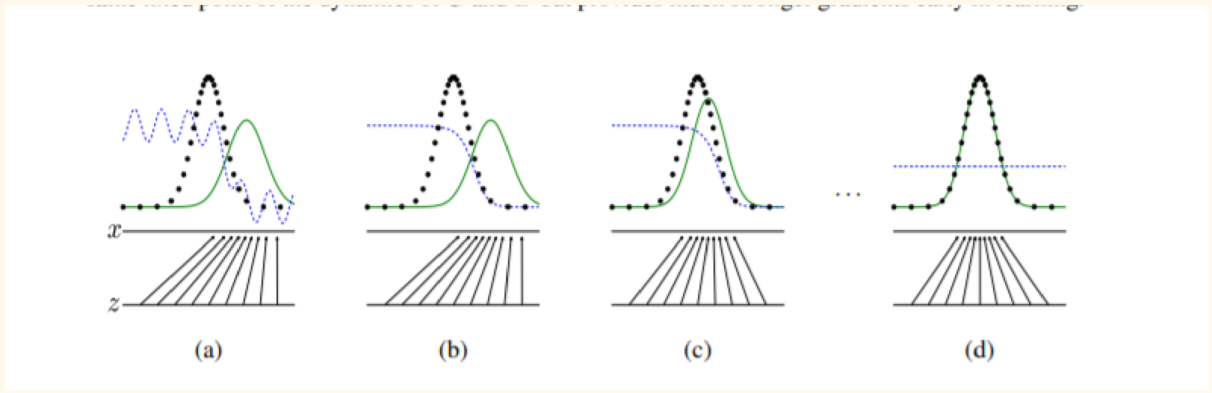
LITERATURE REVIEW

We reviewed 6 different research papers to combine their methods, inspiration,ideas and outcomes to build up our project. The different literature reviewed is summarised sequentially.

1.Generative Adversarial Nets by Ian J. Goodfellow [2]

In adversarial nets framework, 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 framework can yield specific training algorithms for many kinds of model and optimization algorithm. The authors have explored the special case when the generative model generates samples by passing random noise through a multilayer perceptron, and the discriminative model is also a multilayer perceptron. We refer to this special case as adversarial nets.

- **Adversarial nets**:To learn the generator's distribution pg over data x, we define a prior on input noise variables pz(z), then represent a mapping to data space as G(z; θg), where G is a differentiable function represented by a multilayer perceptron with parameters θg. We also define a second multilayer perceptron D(x; θd) that outputs a single scalar. D(x) represents the probability that x came from the data rather than pg. We train D to maximize the probability of assigning the correct label to both training examples and samples from G. ***We simultaneously train G to minimize log(1 − D(G(z)))***: 2 In other words, D and G play the following two-player minimax game with value function V (G, D): min G max D
  - **V (D, G) = Ex~pdata(x) [log D(x)] + Ez~pz(z) [log(1 − D(G(z)))]**



Generative adversarial nets are trained by simultaneously updating the discriminative distribution (D, blue, dashed line) so that it discriminates between samples from the data generating distribution (black, dotted line) px from those of the generative distribution pg (G) (green, solid line). The lower horizontal line is the domain from which z is sampled, in this case uniformly. The horizontal line above is part of the domain of x. The upward arrows show how the mapping x = G(z) imposes the non-uniform distribution pg on transformed samples. G contracts in regions of high density and expands in regions of low density of pg. (a) Consider an adversarial pair near convergence: pg is similar to pdata and D is a partially accurate classifier. (b) In the inner loop of the algorithm D is trained to discriminate samples from data, converging to D ∗ (x) = pdata(x) pdata(x)+pg(x) . (c) After an update to G, gradient of D has guided G(z) to flow to regions that are more likely to be classified as data. (d) After several steps of training, if G and D have enough capacity, they will reach a point at which both cannot improve because pg = pdata. The discriminator is unable to differentiate between the two distributions, i.e. D(x) = 1 2 .

- **Conclusion** This framework admits many straightforward extensions:

1. A conditional generative model p(x | c) can be obtained by adding c as input to both G and D.

2. Learned approximate inference can be performed by training an auxiliary network to predict z given x. This is similar to the inference net trained by the wake-sleep algorithm but with the advantage that the inference net may be trained for a fixed generator net after the generator net has finished training.

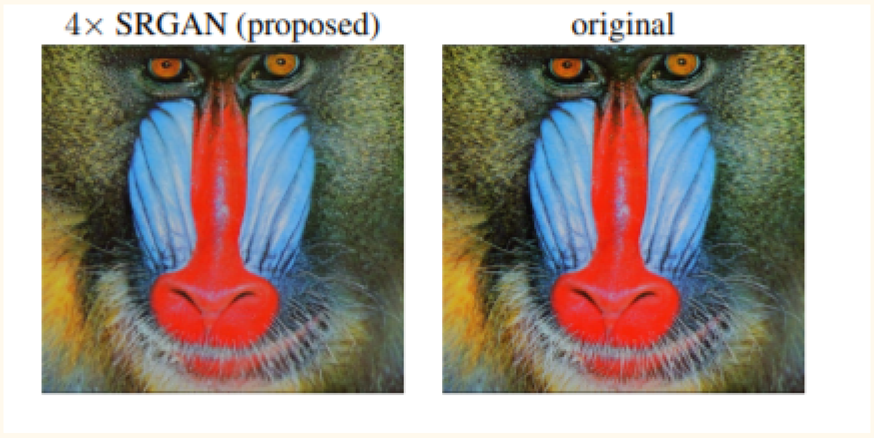
3. One can approximately model all conditionals p(xS | x6S) where S is a subset of the indices of x by training a family of conditional models that share parameters. Essentially, one can use adversarial nets to implement a stochastic extension of the deterministic MP-DBM .

4. Semi-supervised learning: features from the discriminator or inference net could improve performance of classifiers when limited labeled data is available.

5. Efficiency improvements: training could be accelerated greatly by divising better methods for coordinating G and D or determining better distributions to sample z from during training. This paper has demonstrated the viability of the adversarial modeling framework, suggesting that these research directions could prove useful.

2.Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network -Christian Ledig [1]

The highly challenging task of estimating a highresolution (HR) image from its low-resolution (LR) counterpart is referred to as super-resolution (SR). SR received substantial attention from within the computer vision research community and has a wide range of applications



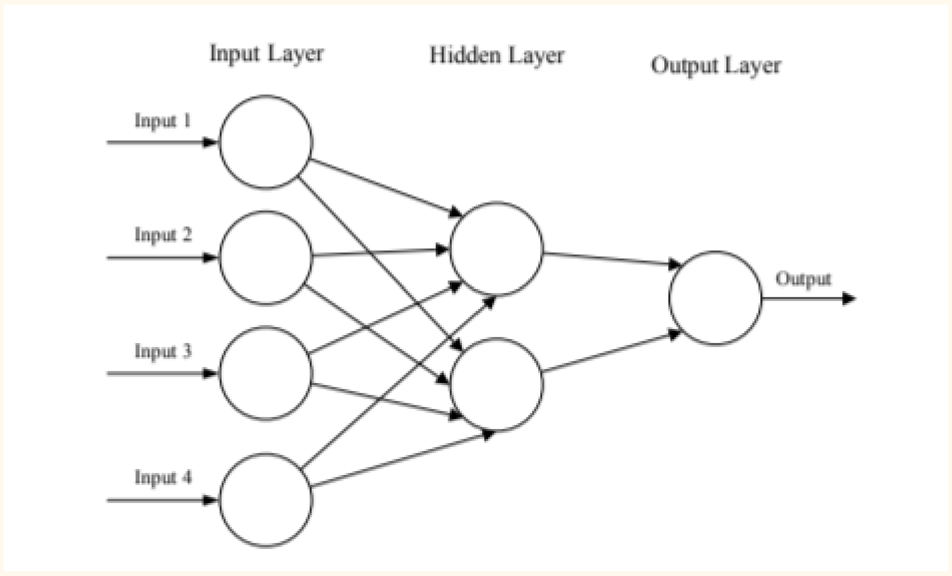
We took the concept of loss function to add to our discriminator network to enhance our work on the previous literature on generative adversarial network to minimise color loss and adversarial loss.

We formulate the perceptual loss as the weighted sum of a content loss (l SR X) and an adversarial loss component as

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

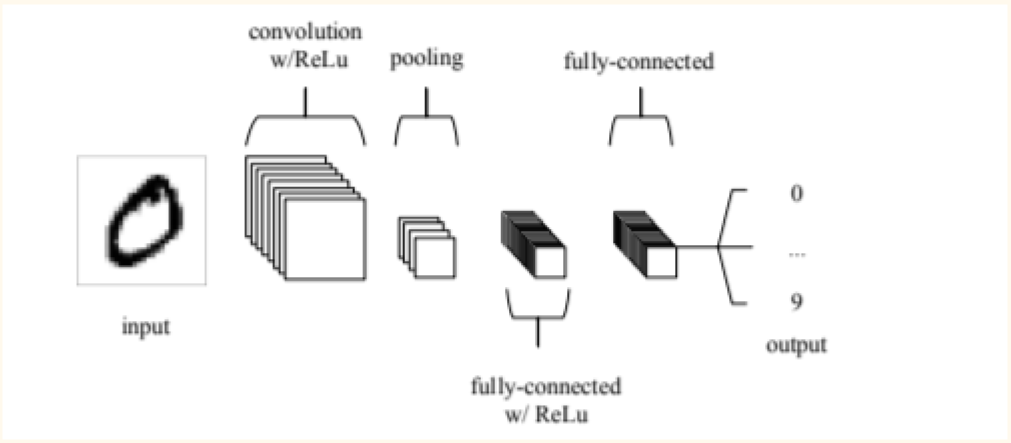
3.An Introduction to Convolutional Neural Networks Keiron- O'Shea and Ryan Nas [5]

Artificial Neural Networks (ANNs) are computational processing systems of which are heavily inspired by way biological nervous systems (such as the human brain) operate. ANNs are mainly comprised of a high number of interconnected computational nodes (referred to as neurons), of which work entwine in a distributed fashion to collectively learn from the input in order to optimise its final output. The basic concept of an ANN can be explained below in the picture



Convolutional Neural Networks (CNNs) are analogous to traditional ANNs in that they are comprised of neurons that self-optimise through learning. Each neuron will still receive an input and perform a operation (such as a scalar product followed by a non-linear function) - the basis of countless ANNs. From the input raw image vectors to the final output of the class score, the entire of the network will still express a single perceptive score function (the weight). The last layer will contain loss functions associated with the classes, and all of the regular tips and tricks developed for traditional ANNs still apply.

1. As found in other forms of ANN, the input layer will hold the pixel values of the image.
2. The convolutional layer will determine the output of neurons of which are connected to local regions of the input through the calculation of the scalar product between their weights and the region connected to the input volume. The rectified linear unit (commonly shortened to ReLu) aims to apply Introduction to Convolutional Neural Networks 5 an ’elementwise’ activation function such as sigmoid to the output of the activation produced by the previous layer.
3. The pooling layer will then simply perform downsampling along the spatial dimensionality of the given input, further reducing the number of parameters within that activation.
4. The fully-connected layers will then perform the same duties found in standard ANNs and attempt to produce class scores from the activations, to be used for classification. It is also suggested that ReLu may be used between these layers, as to improve performance.

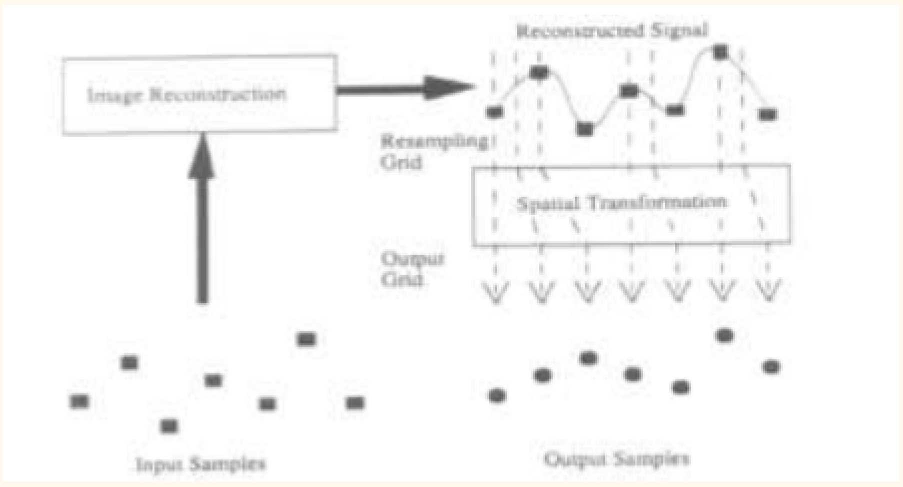


4.Interpolation Techniques in Image Resampling - Manjunatha. S Malini M Patil [6]

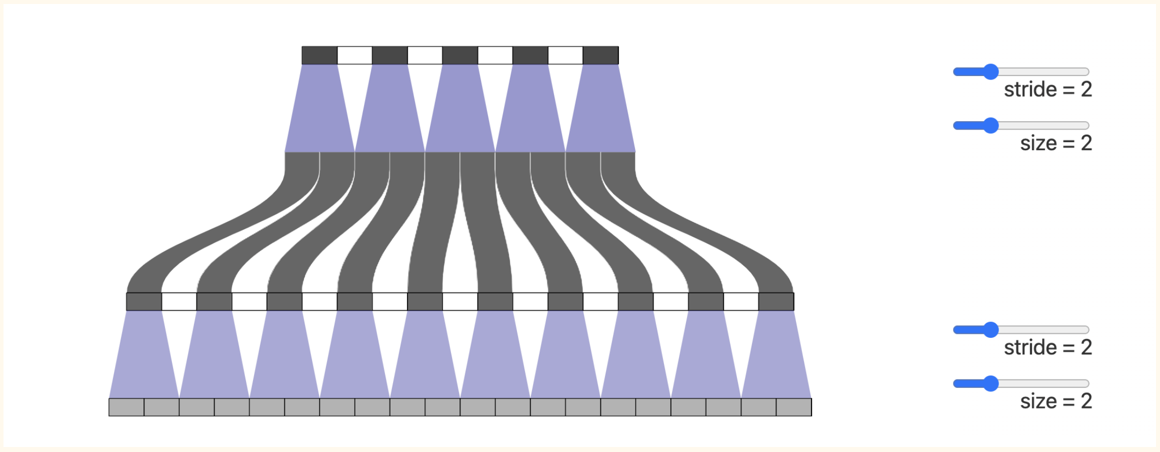
The authors come up with the distinguished procedure to find resampling in images. The main idea is that resampling establishes a regular interrelationship between pixels by interpolation. To find these interrelationships, a linear model is used under which every pixel is pretended that it belongs to a non-resampled group and a resampled group, each with equal possibility. The restricted possibility for a specific pixel closeness to the resampled class is suspected to be Gaussian, while the conditional or restricted possibility for a specific pixel associated to the other group is assumed to be identical. An Expectation Maximization (EM) algorithm can be used to carry out assessment of a pixel's possibility in linear consortium with its adjacent pixels and the unidentified weights of the grouping. In the Expectation step, the possibility of a pixel belonging to the resampled group is intended. In the step, Maximization the exact 568 International Journal of Engineering & Technology method of the correlations between samples is assessed. The stopping condition is prescribed when there is a mismatch in the weights among two succeeding reiterate is quite small. During this stage, the matrix of possibility values got in the expectations step for all image pixels is called the ―Probability map (p-map)‖. For that image this p-map is periodic and crest in the 2D Fourier spectrum of the p-map signifies the resampling. In p-map, a probability value very near to 1 then it shows there is resampling of pixel.

Resampling Resampling is the method of varying sampling rate of the targeted discrete image. During manipulation, geometrical transformation of digital image are usually involved, which is definite at one set of coordinate points. Image resampling is a complicated process, so a frame work is required which can be used to classify and analyze resampling methods. Many times the formation of conclusive image forgery demands geometrical transformations. Totally, such geometric transformation propose a resampling of the source image. These operations imposed in the pixel domain, it will affect the locations of samples. Therefore the image must be resampled to a new sampling lattice. Resampling establishes specific interrelationship in the samples of image, which can be used as an proof of tampering. During resampling procedure , when sample is down, meaning that it reduces the pixels in image, facts are destroyed from the image. When sample is up, or escalation of the pixels in the image. Hence it affects the display size of image. Resampling is decomposed in to 3 sub process

1. Reconstruction of a continuous intensity surface from a discrete image.
2. Transformation of that continuous surface.
3. Sampling of the transformed surface to produce a discrete image.



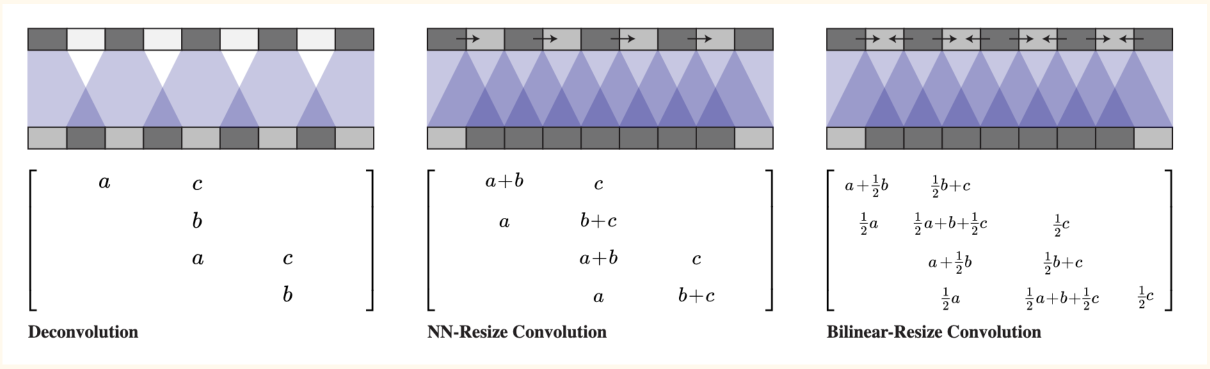
5.Deconvolution & Checkerboard Artifacts - Odena et al. - 2016 - [7]



Unfortunately, deconvolution can easily have “uneven overlap,” putting more of the metaphorical paint in some places than others. In particular, deconvolution has uneven overlap when the kernel size (the output window size) is not divisible by the stride (the spacing between points on the top).

Another approach is to separate out upsampling to a higher resolution from convolution to compute features. For example, you might resize the image (using nearest-neighbor interpolation or bilinear interpolation) and then do a convolutional layer.

Resize-convolution layers can be easily implemented in TensorFlow using tf.image.resize_images(). For best results, use tf.pad() before doing convolution with tf.nn.conv2d() to avoid boundary artifacts.



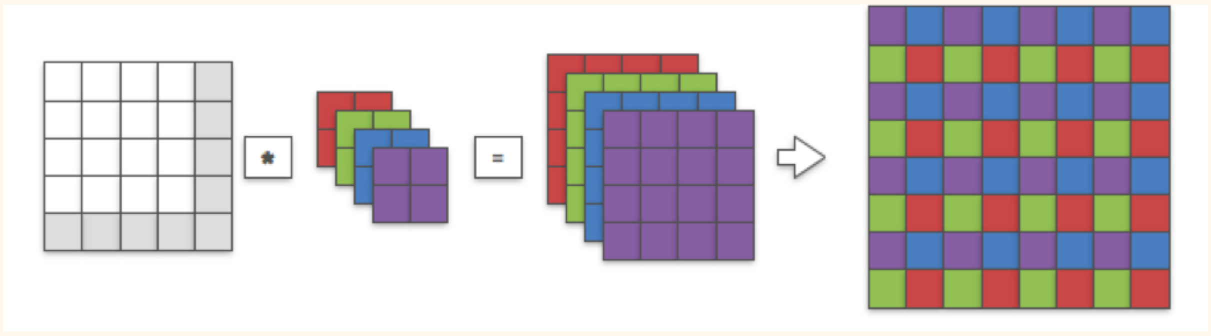
6.Checkerboard artifact free sub-pixel convolution - Andrew Aitken et al - [8]

Sub- pixel convolution is a specific implementation of a devolution layer that can be interpreted as a standard convolution in low- resolution space followed by a periodic shuffling operation.

Sub-pixel convolution has the advantage over standard resize convolutions that, at the same computational complexity, it has more parameters and thus better modelling power. Sub-pixel convolution is constrained to not allow deconvolution overlap , however it suffers from checkerboard artifacts following random initialization.

An initialization method for sub-pixel convolution known as convolution NN resize, compared to the sub-pixel convolution initialized with the schemes designed for standard convolution kernels, it is free from checkerboard artifacts immediately after initialization. Compared to resize convolution, at the same computational complexity, it has more modelling power and converges to solutions with smaller test errors.

The first two sources of artifacts deconvolution overlap and random initialization, can be eliminated by the resize convolution. Resize convolution first upscales the LR feature maps using nearest neighbour interpolation and then employs a standard convolutional layer with both input and output in HR space. Resize convolutions become a popular choice for generative modelling to alleviate checkerboard artifacts.



Method and Implementation

Our aim is to have a generating network represented by G that can minimise the low resolution artefacts and blurriness, or, to generate an HR image for an LR image that we input in. We aim to do this by training the network on a loss that is more akin to how humans perceive images and feature within thereof.

We aim to achieve this by using a feed forward deconvolution network as our generator Gsimilar to Ledig et al [1]. If we assume N total images to train on, $I_{HR,\,i}$ represent the ith HR image in the dataset, and $I_{LR,\,i}$ its LR counterpart. Assuming a loss function that closely represents our human perception of an image as L This means our goals remains to solve the equation:

$$\theta_{SR} \;=\; min(\frac{1}{N}\Sigma_{i=1}^{i=n} L\ (G(I_{LR,n}),\ I_{HR,n}))$$

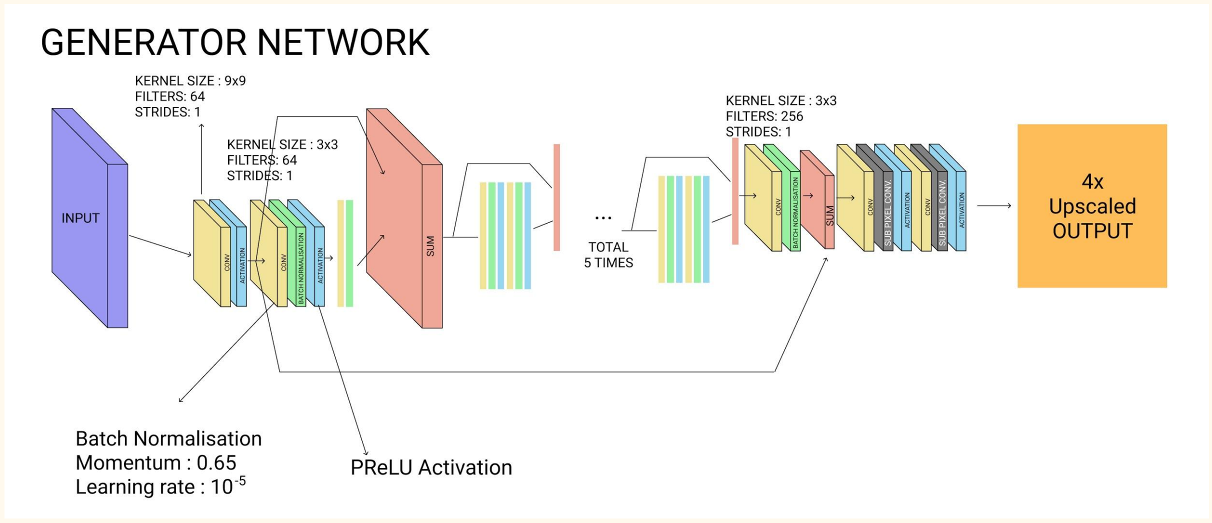
This arbitrary loss function has been proposed by Ledig et al and called as perceptual loss. Here, after seeing problems in the said loss, we propose a new loss function. We name this colour bounded perceptual loss. We will discuss this further later.

Network Design

Building upon the general adversarial network building blocks of Goodfellow et al [2], we use the same ideology to work according to the classic minmax problem for adversarial networks.

$$\min_{\theta_G} \max_{\theta_D} \mathbb{E}_{I^{HR} \sim p_{\text{train}}(I^{HR})}[\log D_{\theta_D}(I^{HR})] + \mathbb{E}_{I^{LR} \sim p_G(I^{LR})}[\log(1 - D_{\theta_D}(G_{\theta_G}(I^{LR}))]$$

We use the same ideas to build the generator network as in Ledig et al [1]. We use an initial deconv layer with 64 9x9 kernels, followed by a PReLU activation. Then we use this as the base, and pass it through a residual block with deconv 64 layers of 3x3 kernels, followed by batch normalisation and PReLU activation, another deconv layer and activation, before it is added back into the main base network and the new base network is created hence. This is the same idea of deep residual learning as in Zhang et al. We do this 5 times before finally passing it to a deconv layer of kernel size 3x3 and 256 filters. This result is then added back to the original base network. The output is then passed through two layers of sub pixel convolution layers to upscale it by a factor of 4 proposed by Shi et al[3].



This is how our generator network finally looks like. Pretty similar to Ledig et al with no substantial changes.

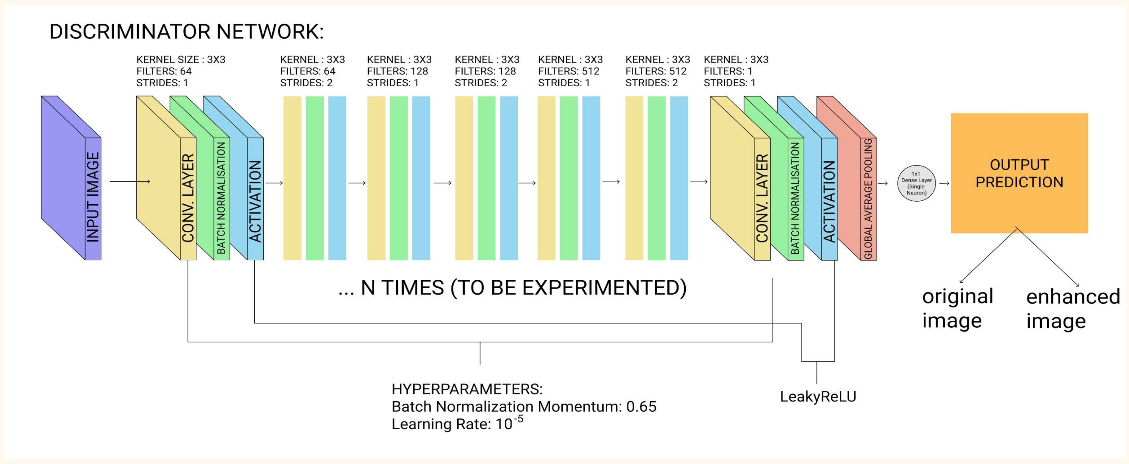
However, what has changed is the discriminator network. We took an approach which would let us use images of any size and resolution for the training of the discriminator network.

The Discriminator Architecture

To achieve this dimensional invariance, i.e, non dependence of the network on the size of input images, we use a pure convolutional network. The problem arises due to the fact that a dense layer requires a fixed number of inputs and outputs in order to function. Although the network described in Ledig et al[1] is convolution only, with the exception of a dense layer at the end. This dense layer forces you to have a fixed size image, which leads to an inferior learning.

Thus we propose a new discriminator model, working on top of the discriminator in Ledig et al[1], that does away with the dense networks. Instead what we use is a deconvolution layer with a single filter, so it produces a 2D matrix with all our predictions, followed by batch normalisation and activation. We pass this further into a Global Average Pooling layer, taking the idea from conah et al[4]. The global average pooling layer converts the matrix into a single value which we finally pass onto a single neuron with the idea that this distributes the weights of the average pooling layer for further biasing.

Along with this the network uses convolution, batch normalisation, and activation blocks with alternating 2x2 and 1x1 strides, and increasing kernel size as in the figure below.



Our discriminator network is now fully convolutional and works on inputs of any arbitrary size. We have hence achieved dimensional invariance with respect to training in both the generator and the discriminator of our network.

The next most important thing that remains to be defined is the loss function. Here, we use a modified version of the perceptual loss function as defined in Ledig et al[1] and call it a colour bounded perceptual loss function.

Colour Bounded Perceptual Loss

The main idea is to build a loss function different from traditional loss functions that produce a PSNR or SSIM score and update according to that. The idea is to create a loss function that can better represent how we as humans see and perceive photos. This was called the perceptual loss function in Ledig et al[1]. Ledig et al describes two kinds of losses : a VGG content loss and another, Adversarial loss. We observed gaps as these methods capture greatly how a human sees a photo, i.e, the VGG loss (discussed further below) does a great job of identifying features in a manner close to how we identify features such as lines and curves in our world, and the adversarial loss shows how the image deviates from reality. However, a gap was found as this does not include a loss for the colours used in the image. As was seen in training too, this lead to images that are not in the same colour shades as the original, however still seem like real images. An example of such a transformation would be changing human skin colour tones. A different toned skin is still realistic, and would fool both the VGG and Adversarial losses, but still would be very different from the original image.Hence we propose a new colour loss function that estimates a loss between the original image and our generated one. Our final loss function looks like:

$$L_{SR} = L_{content} + 0.3\, L^{SR}_{BCE} + 10^{-3} L^{SR}_{Gen}$$

Colour Loss

The goal of this function is to create a loss value when the generated image deviates in the colours used by the CNN generator network. To achieve this, we devise this new colour loss function represented by L^{SR}_{BCE}. Here, we simply take a binary cross entropy on the generated and the HR images and use that as the loss, in combination with other losses. Thus, having N total images, and I_G as our generated image, our colour loss function would be:

$$L^{SR}_{BCE} = -\frac{1}{N}\left(\sum_{i=0}^{n} I_{HR,\,i} log(I_{G,\,i})\right)$$

This represents with reasonable effectiveness, the deviations in colour of our generated images.

Content loss

Our content loss remains the same as in Ledig et al. It is the mean squared error between the outputs of the VGG 19 network by the group at oxford, for our generated image and the original HR image. This is given by:

$$l^{SR}_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

Adversarial loss

Adversarial loss is the simple binary cross entropy loss between the prediction and reality. For generated images, it is the loss between the image and the value 0. For the original HR image, it is the loss between the image and the value 1.

$$l^{SR}_{Gen} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

Residual Layers

SRGAN residual block is used in the SRGAN generator for image super resolution and uses a PReLU activation function to help training. We have changed the residual block layers from 5 to 16 in our model, and changed the activation from PReLU to LeakyReLU. This has shown to give better results.

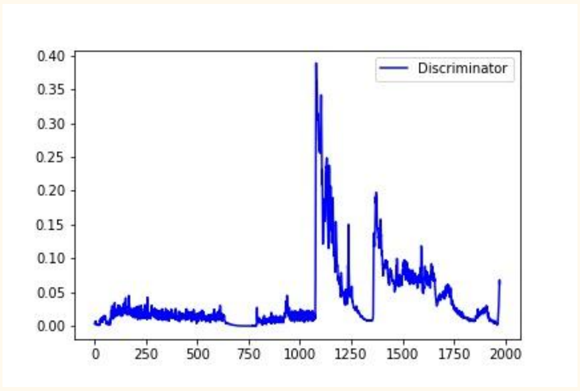
EXPERIMENTATION AND RESULTS

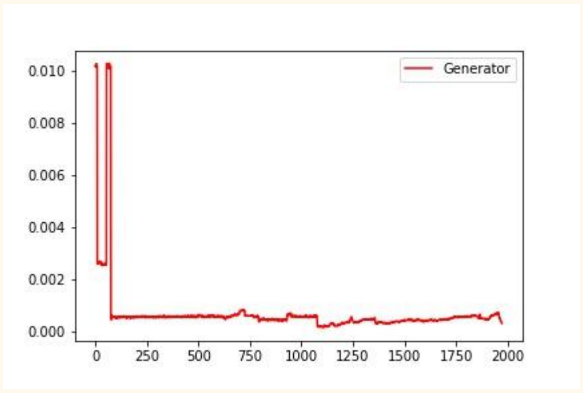
DATABASE

For the project, the DIV2K database was chosen. It contains 800 training images and 100 validation images. All images are of 2K HD resolution. These images were shuffled randomly, and sometimes cropped from random places, to create a variance in the dataset used. All of these 900 images have a low resolution counterpart in the dataset which is downscaled by a factor of 4 using bicubic interpolation. The dataset is the benchmark used in most Super resolution deep learning based approaches. It contains images of varying genres, including man made as well as natural objects, sceneries, water bodies, mountains, faces, houses, animals, food items, among others.

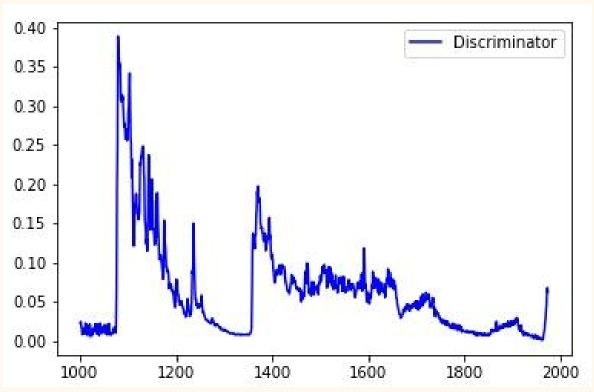
TRAINING DETAILS

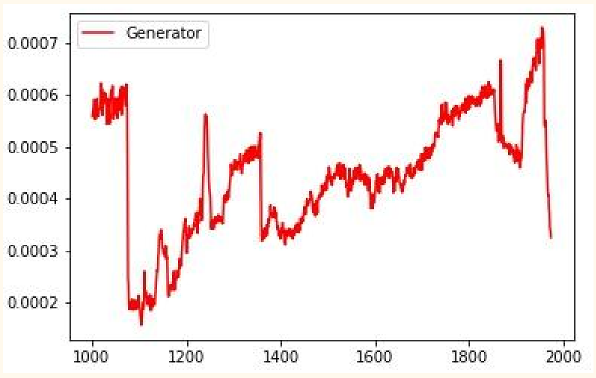
The network was trained on a google collaborator notebook. Mostly, this means it was trained on an NVIDIA T4 GPU with around 16GB VRAM and another 14 GB on computer RAM. The training was performed for a total of 2000 epochs. Two sets of models were trained in this. The first model, referred by network (1) used our colour bounded perceptual loss function for all the 2000 iterations. The other model, referred by network (2) used the original perceptual loss as defined by Ledig et al for the first ~1000 iterations and our colour bounded perceptual loss for the next 1000 iterations. It was settled on that the batch normalisation layers should use a momentum of 0.5 in both the discriminator and generator. The results however prove our network to be more on the unstable side and sometimes endures mutations that render it to a very high loss. However, after about 1000 iterations, a somewhat stability was observed. The learning rate chosen for the training was 10^{-5} in both the training models. The losses produced can be depicted here:





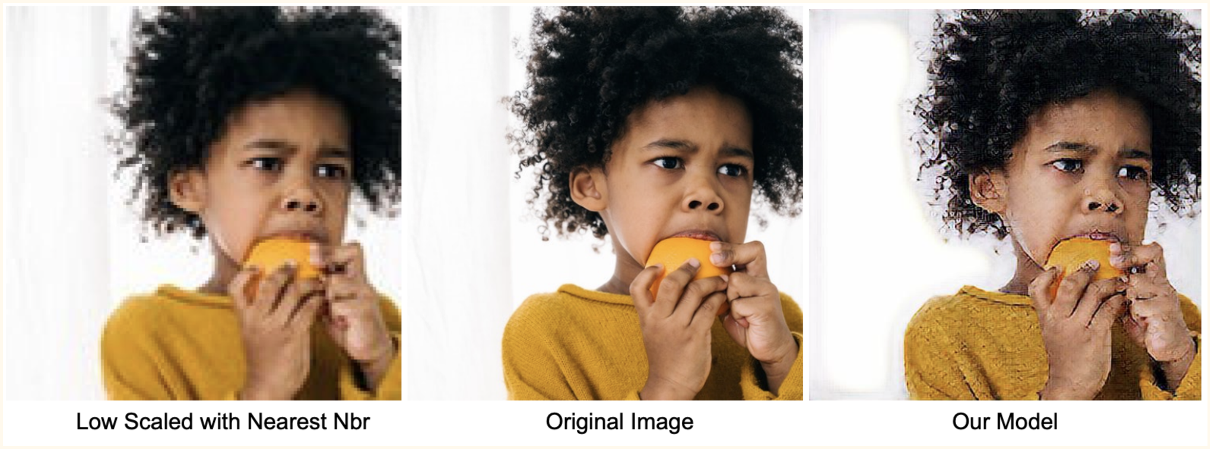
The losses after 1000 iterations, zoomed in, look like:





Opinion Testing

The LR images were upscaled using nearest neighbour and bicubic methods to a 4x upsampling. This, along with our generated images were shown to people to judge which looks better and clearer to a person, and in which, one can identify features and lines better. It was seen that almost all the times, the generated images were better in people's opinion. The images comparison looks like:



Comparison on traditional comparison benchmarks show these methods to not be the greatest improvement, but as is clear visually, the generated images are perceivably superior in almost all cases. Hence, the idea proposed in Ledig et al [1] is extended further that PSNR and SSIM are not the best benchmarks when it comes to realistic photo realistic super resolution.

Method	PSNR	SSIM
Nearest Neighbour	23.9647	0.6928
Proposed DCSRGAN	**24.6983**	**0.7073**

CONCLUSION

We have proposed improvements over the work of Ledig et al by 2 main new proposals. An introduction of a dimensional invariant discriminator network, consisting of pure convolutional layers, and a new colour loss function, to assist the general perceptual loss as in Ledig et al.

We also extend the idea tossed by Ledig et al that PSNR and SISM have limitations and are not the best when it comes to predicting comparative quality of an image. This idea was elaborated by an opinion testing, however, an extensive mean opinion testing could not be performed due to time and fund shortage.

Our project has achieved a 4x upscaling using two different training methods by a factor of 4x, and with popular opinion, creates more realistic and colour accurate images as compared to the original SRGAN. Not only this, but though the SRGAN was able to achieve this in about 2×10^5epochs, our network was able to achieve commendable results in only about 2000 epochs, which is a great improvement.

CONTRIBUTIONS

- Rajat Goel 18103108 - Python Implementation with Tensorflow, finding colour loss, checkpointing
- Shreya Grover 18103113 - Understanding the project, devisign the methodology, finding colour loss and dimensional invariance
- Sarthak Sharma 18103112 - Literature review, suggesting Global Average Pooling, dimensional invariance

All members collaboratively worked in google colab notebooks to implement and improve the network and training. Inferences were also drawn collaboratively

REFERENCES

1. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network - Ledig et al

2. Generative Adversarial Nets - Ian J. Goodfellow et al

3. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network - Shit el al

4. Anysize GAN: A solution to the image-warping problem

5. An Introduction to Convolutional Neural Networks Keiron- O'Shea and Ryan Nas

6. Interpolation Techniques in Image Resampling - Manjunatha. S Malini M Patil

7. Odena, et al., "Deconvolution and Checkerboard Artifacts", Distill, 2016.

8. Checkerboard artifact free sub-pixel convolution - Andrew Aitken et al