Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network

Ruchit Vithani 201701070
Purvil Mehta 201701073
Bhargey Mehta 201701074
Kushal Shah 201701111

Abstract

Image super resolution is a very wide spread problem and have several models with great accuracy and speed including deep convolution network. However, the central problem of lack of high frequency component or the fine texture in a super-resolved image remains the same. Recent work on this problem considered the pixel wise mean squared Reconstruction error(MSE). While older models have great accuracy with less MSE and high signal to noise ratio, they often fail to match the fidelity expected in a high resolution image. In this report we look at proposed[10] perceptual loss based Generative Adversarial Network. The perceptual loss allows the high frequency component in the reconstructed image. Our model pushes the generated high resolution image to natural image manifold with the help of a generating network which tends to minimize the error and a discriminator which is trained to differentiate between these two images. We have considered content loss consisting of pre-trained VGG19 model loss to get the finer texture in the reconstructed image.

1. Introduction

The task of taking a low resolution image to produce it's high resolution counterpart is known as super resolution.

A straight-forward solution would be to use a supervised algorithm with a loss function defined as the mean square error between the low resolution image and it's original high resolution counterpart. This also results in a higher PSNR in the resultant image. However, the texture details in the resultant high resolution image are lost since the mean square error loss function blurs the texture in the original image in order to minimise the mean square error.

Blurring is an inherent problem to super resolution task when using a mean square error loss. Hence we can say that a lower mean square error between the generated high resolution image and the original image doesn't necessarily imply a good super resolution.

So in this report we will be looking at the super resolution task using a GAN. In the architecture named SRGAN, we will be using a deep residual network and also look at a loss to capture perceptual similarity which is actually the MSE produced in a different vector space. This vector space is actually produced by layers of the VGG network.

Some examples results of the said architecture are shown below.

Figure 1: (a) Original Image (b) SRGAN Image

2. Related Work

Initially, the mathematical model for mapping low resolution pixels to its corresponding high resolution pixel was used. Traditionally, the sparse coding based method [7] was to used to generate this dictionaries. Sparse coding approach was used to transform a low resolution image to its corresponding high resolution counterpart. In these approaches, the low resolution image is first encoded into a latent sparse representation vector. This vector is then used to reproduce the corresponding high resolution image. SISR problem is inherently ill-posed since a multiplicity of so-

lutions exist for any given low-resolution pixel. In other words, it is an under-determined inverse problem, of which solution is not unique. Such a problem is typically mitigated by constraining the solution space by strong prior information. Thus, the sparse-coding-based method is one of the representative external example-based SR methods which allow us to build the dictionary having prior information. Sparse coding was the first method developed to solve this problem.

In [6], the authors have shown that there is an intimate connection between sparse coding and neural network. They have shown that a neural network can efficiently approximate the sparse coding. Based on this results, a deep learning based approach which integrates sparse coding has been studied in [12]. In their approach, a feed forward neural network is trained to encode the LR image into a sparse vector. Then this sparse vector is used to extract HR image, again using a feed forward neural network.

Researchers have incorporated deep learning into sparse coding methods in order to achieve good performance.In 2015, [3] authors used the deep convolution neural network (SRCNN) for the first time to solve the problem. They used a convolutional neural network that directly learns an end-to-end mapping between low- and high-resolution images which gives the faster computational complexity on different channels simultaneously.

In [4], researchers proposed an improved version of the architecture explained above. The architecture mentioned above is very computationally heavy so it cannot work in real time when applied to a video stream. The achieved framerate is just 1.32 fps as opposed the required framerate of 24 fps.

The proposed architecture first removes the bicubic interpolation layer. Hence the time complexity thus depends only on the original LR image and not the size of the output image. It has been found that upscaling the image and then applying feature extraction results in a better performance although at the cost of computational cost. To resolve this issue, the architecture has shrinking layers present between the input and the sparse mapping layer. This allows the model to extract the necessary features before providing them to the mapping layer in between. At the end we have expanding layers before the final deconvolution layer. These expanding layers rebuild the image to the size of the target HR image and the final deconvolutional layer produces the HR image.

The authors in [9]. propose a Deeply Recursive Convolutional Network (DRCN) for the image super-resolution task. The peculiarity of the network lies in its recursive structure and in the skip-connections which connects the layers skipping the in-between layers and establishes the identity mappings. The recursive structure means that the output of a layer is passed as the input to the same layer multiple times. Compared to the previous maximum claim of 3, the particular DRCN applies the same convolutional layer 16 times. Each recursive layer's out-put makes a weighted contribution to the final super-resolved image the weights are learned during the training procedure. However, contradictory to the suggestions made in this report, the proposed DRCN uses the traditional MSE loss.

3. Intro to GANs

The architecture that we will be using falls under the category of GANs. We will first see what GANs are and then we will introduce how they will be used in our problem settings.

GAN stands for Generative Adversarial Networks- the idea is to use 2 networks

- G Generative: The task of this network is to generate data samples i.e. $G(z^{(i)})$ from a given input distribution i.e. $z^{(i)}$.
- A Adversarial: The task of this network is to discriminate between a given data sample and identify it as belonging or not belonging to the original distribution i.e. $x^{(i)}$.

Thus the role of G network can be thought of as a counterfeiter that tries making copies (i.e. $G(z^{(i)})$) of some original currency (i.e. $x^{(i)}$) and the role of D network can be thought of as to distinguish the "real" currency $x^{(i)}$ from the "fake" generated ones i.e. $G(z^{(i)})$.

3.1. Discriminator D

The D network in essence performs a classification task of classifying given input data as either real or fake. Its output $D(x)$ is a variable between 0 and 1 which denotes the probability of x being real.

The task of the discriminator D thus is to correctly classify actual data $x^{(i)}$ as real and generated data from G i.e. $G(z^{(i)})$ as fake. Thus $D(x^{(i)})$ denotes the probability that x_i is real and hence should be close to 1. Similarly $D(G(z^{(i)}))$ is fake and hence should be close to 0 or equivalently $1 - D(G(z^{(i)}))$ should be close to 1.

The binary cross entropy loss is given by

$$L = \frac{1}{m}\Sigma_i^m \left(\log\left(D(x^{(i)})\right) + \log\left(1 - D(G(z^{(i)}))\right)\right) \quad (1)$$

. We can see that minimising this loss function captures the behaviour that we intended.

3.2. Generator G

Let's denote p_{data} to be the original data distribution which we don't know analytically but want to emulate using GANs. As discussed earlier G takes samples from a probability distribution different from the distribution (note that

the distribution is different since we don't have the original distribution in the first place) we are interested in. These samples are labelled $z^{(i)}$.

The generator network then produces $G(z^{(i)})$ which ideally belong to the original distribution p_{data}. The task of the generator is to produce $G(z^{(i)})$ which "fools" the discriminator and hence translates to maximising $D(G(z^{(i)}))$ or minimising $1 - D(G(z^{(i)}))$ in the mathematical setting.

The loss function adopted for the generator G is thus given by

$$L = -\frac{1}{m}\Sigma_i^m \log\left(1 - D(G(z^{(i)}))\right) \tag{2}$$

which captures the intended behavior from the generator.

3.3. Training GANs

The training of GANs is quite similar to training a deep neural network. There are k updates of the discriminator network D, followed by a single update of the generator network G. Note that k is a hyper parameter of the model.

3.4. Intro to SRGAN

Now we will look at how GANs will be used for the task of super resolution. Note that we have access to the low resolution images which are then used to generate the high resolution images. So the low resolution I_{LR} are actually $z^{(i)}$ and the generated high resolution images I_{HR} are nothing but $G(z^{(i)})$.

Our interest lies in obtaining the original high resolution images, hence the ground truth high resolution images form the $x^{(i)}$ whose distribution is to be emulated by the generator network G.

Hence if our generator network G produces high resolution image $G(z^{(i)})$ from given input low resolution image $z^{(i)}$ which successfully fools the discriminator network D, then the obtained image is a good estimation of the original high resolution image $x^{(i)}$.

4. Architecture

The previous section threw light on how a generic GAN is supposed to work. This section suggests how the GAN is applied to the specific problem of super resolution in the paper. The section would consist of description of architecture of the generative network and the discriminator network.

4.1. Generator Model

Since deeper network architectures do possess the ability to model the mappings with very high complexity, it is reasonable to believe that these networks can increase the accuracy substantially. However, they are difficult to train. To over-come the difficulty, we use two solutions namely, batch-normalization and skip-connection. While the former deals with co-variate shift the latter is used to take care of identity mappings and vanishing gradients. Identity mappings might seem trivial but representing them via a network of convolutional layers is not a simple task and hence, we just skip the layers using skip-connections. This brings us to residual block which can be diagrammatically and mathematically represented as follows:

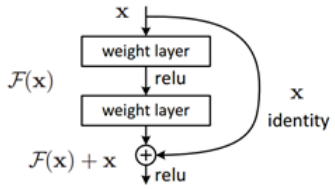

Figure 3: Skip Model

A single residual block of generative network consists of convolution layer, batch-normalization, parametric-relu activation unit, convolution layer, batch normalization and finally an element wise sum with the output of the previous layer (skip connection) in order. There are 16 such blocks followed by two learning up-scaling layers. The layer can be represented in the figure 2.

Mathematically, our goal is to train the network G_θ parameterized by θ such that the input low resolution image I^{LR} is closely mapped to the original high-resolution image I^{HR} . Or equivalently, $G_\theta(I^{LR}) \approx I^{HR}$. For this purpose, we use the perceptual loss function explained in the section that follows.

4.2. Adversarial Network (D)

Adversarial network essentially discriminates between original high-resolution images and the fake images generated by Generative network. Mathematically, the output of the network is the probability of image being an original high-resolution image. So, ideally, the generated images $G(I^{LR})$ should give the output $D(G(I^{LR})) = 0$. However, by the virtue of alternating training of Generator and Discriminator, the generator eventually learns how to fool the discriminator and this is exactly how we push our solution to the space of natural images.

Each block of the Discriminator network possesses a convolutional layer followed by batch-normalization and finally, a leaky-relu (α = 0.2) activation unit. The kernel size in each block is kept 3. We keep on increasing the number of filters in the convolutional layer from 64 to 512. We keep the stride at 2 which essentially means reducing the resolution of input images. The blocks are followed by a dense layer and finally a sigmoid function that outputs the probability. Following is the diagrammatic representation of the discriminator network.

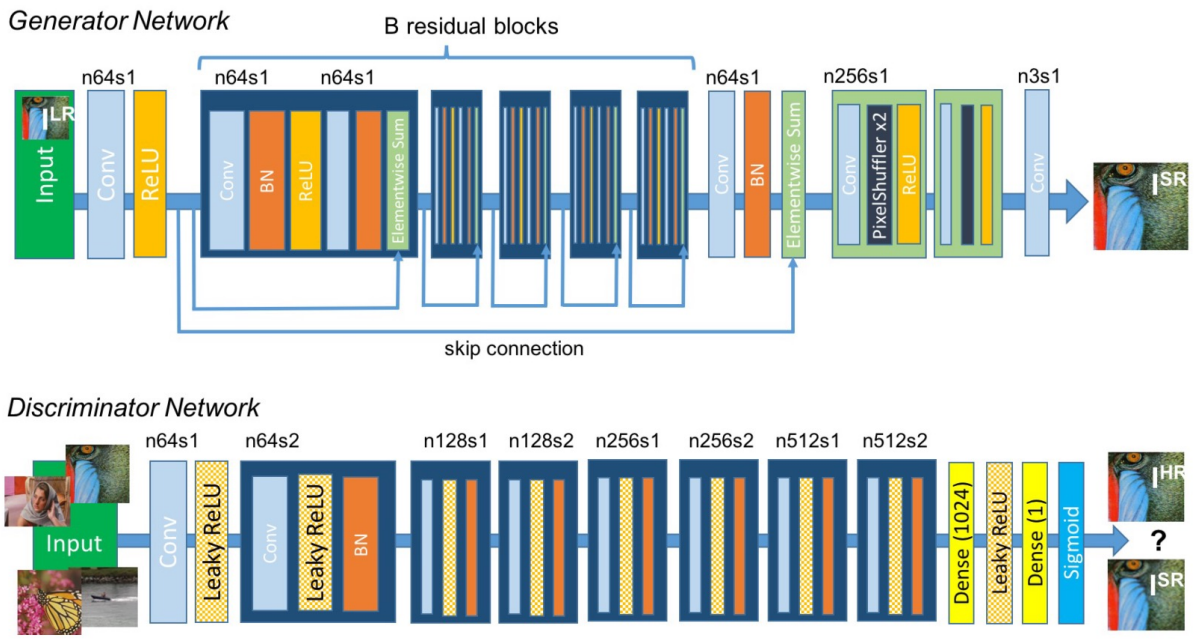

Figure 2: Architecture of Generator and Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) indicated for each convolutional layer.

4.3. Pre-trained Model - VGG19

The VGG network architecture was introduced by Simonyan and Zisserman in their 2014 paper, Very Deep Convolutional Networks for Large Scale Image Recognition [11]. VGG-19 is a convolutional neural network that is 19 layers deep. VGG19 is pre trained over million images available in ImageNet dataset. VGG19 is the multi-class classification, which has ability to classify the image over 100+ classes.

We used this model to identify the features of the high resolution image. In order to calculate the Perceptual loss, we used j^{th} convolution layer featured map (before the max polling and after activation function) which we denoted by $\phi_{i,j}$ in equation 5. We took the MSE loss between the features of original high resolution image I^{HR} and generated high resolution image $G(I^{LR})$ to get the perceptual loss discussed in 5.1.

5. Loss Functions

Pixel-wise loss functions such as MSE always face challenges to reconstruct the high frequency or finer texture details in the recovered image. MSE is the pixel wise average of the square of difference between the two images and typically overly-smooth and have very poor perceptual quality. We illustrate the fact that the better MSE or high PSNR don't need to be better perceptually in figure 4. Instead of using MSE as a primary loss function, we used the MSE of features of the trained HR images and ground truth HR images. We used the perceptual loss function instead. Perceptual loss functions are used when comparing two different images that look similar, like the same photo but shifted by one pixel. The function is used to compare high level differences, like content and style discrepancies, between images.

5.1. Perceptual Loss

The perceptual loss l^{SR} is first proposed by et al. Johnson [8] in his paper and then by the et al. Bruna [2]. We further improved upon that which gives more importance to the feature characteristics and defined our perceptual loss as the weighted sum of the content loss l^{SR}_{X} and adversarial loss $l^{\text{SR}}_{\text{Gen}}$ as follows:

$$l^{\text{SR}} = \underbrace{l^{\text{SR}}_{\text{X}}}_{\text{ContentLoss}} + \underbrace{10^{-3} l^{\text{SR}}_{\text{Gen}}}_{\text{AdversarialLoss}} \tag{3}$$

5.2. Content Loss

The pixel wise loss function is defined as

$$l^{\text{SR}}_{\text{MSE}} = \frac{1}{r^2 W H} \sum_{x=1}^{rW} \sum_{y=1}^{rH} (I^{HR}_{x,y} - G_{\theta_G}(I^{LR})_{x,y})^2 \tag{4}$$

As we disussed earlier, high PSNR and less MSE often struggles to recover the high frequency information from the recovered image, we used featured loss with the help

Figure 4: Comparison of less MSE and bad perceptual Quality

of the pre trained model VGG discussed in 4.3. Instead of taking average of the pixel wise errors, we developed the VGG19 loss using the featured map of the last convolutional layer after the activation function is applied. Since VGG19 is useful to classify the the images into different categories, it accounts for the features of the images. We passed the original HR image and fake HR image which is recovered with the help of generator network from the VGG19 model. We then took the MSE difference such that the model tries to learn the sharp transition in features from the original HR image. We then define the VGG loss as follows,

$$l^{\mathrm{SR}}_{\mathrm{VGG}} = \frac{1}{WH}\sum_{x=1}^{W}\sum_{y=1}^{H}(\phi(I^{HR})_{x,y} - \phi(G_{\theta_G}(I^{LR}))_{x,y})^2 \tag{5}$$

where $G_{\theta_G}(I^{LR})$ is the reconstructed image and I^{HR} is the original image.

5.3. Adversarial Loss

In addition to the content loss, we also added the adversarial loss in the perceptual loss stated in equation 3. As discussed in the section 3.2, generator network tries to fool the discriminator network. This loss helps in achieving this. We defined the generator loss of the network as follows,

$$l^{SR}_{Gen} = \sum_{n=1}^{N} -log(D_{\theta_D}(G_{\theta_G}(I^{LR}))) \tag{6}$$

where, $D_{\theta_G}(I^{LR})$ is the probability that the reconstructed image $D_{\theta_G}(I^{LR})$ is classified as the real image by the discriminator network. The job of the Generator network is to maximize the probability of **the fake image getting classified as the real image by the discriminator network**. In order to maximize $D_{\theta_D}(G(I^{LR}))$ probability, we minimize the $-D_{\theta_D}(G(I^{LR}))$ [5].

5.4. Discriminator Loss

As mentioned in the section 3.1, we used the same Discriminator loss defined in the equation 1

6. Experiments and results

6.1. Dataset used

We perform all our experiments on DIV2K dataset [1]. This dataset consists of three splits : training, validation and test. The training part consists of 800 high definition high resolution image. The validation and test sets both contain 100 high definition high resolution images. Further, this dataset also provides down-sampled images of training set (the down-sampling factors are $2, 3$ and 4). In our experiments, we use only high resolution images and not use ready to available downsampled images. We downsample all the training images by the factor of 4.

6.2. Training details

We implemented the SRGAN in python programming language using pytorch[1] framework. Similar to original paper, we extract a random crop of size 96×96. In addition, we also use the same parameters as those used in original paper, namely : We use LeakyRelu activation function with $\alpha = 0.2$. We use pretrained VGG19 network to define VGG loss described in section 5. VGG loss has been scaled by the factor of 0.006 same as done in the original work. We use Adam optimizer with ($\beta = 0.9$). All our images have been scaled in the range of $[0, 1]$. We do not scale the HR images in the range of $[-1, 1]$ as done in the original paper. Our experiments show that this does not affect the performance of the network significantly.

Based on the investigation of content loss in the original paper, we see that the content loss $l^{SR}_{VGG/5,4}$ defined on

[1]https://pytorch.org/

higher level features of VGG network is producing better SSIM scores. Intutively, this suggests that the MSE loss defined on higher level features of VGG network is more sensitive towards optimization of SRGAN network. So, we define the content loss based on the output feature representations of VGG19 network.

6.3. Results and Analysis

We have trained the model with 150 epochs. We observed that PSNR increases as the epochs increases as model trains to generate. We plot different curves to visualise the training process. Some of the results are shown below. Since we are working on reducing MSE and increasing features between fake HR and original HR, we observed that MSE is reducing as epochs increase. Refer figure 9 for results.

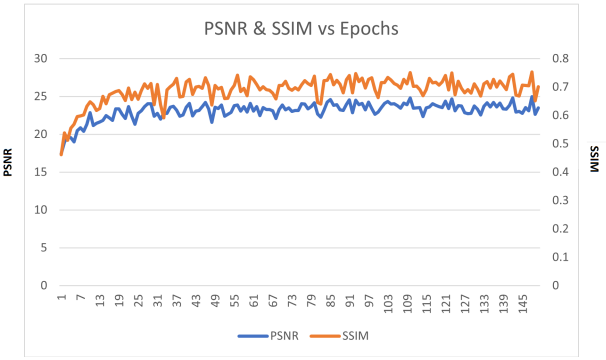

Figure 5: Picture to Noise Ratio vs Epochs in training process

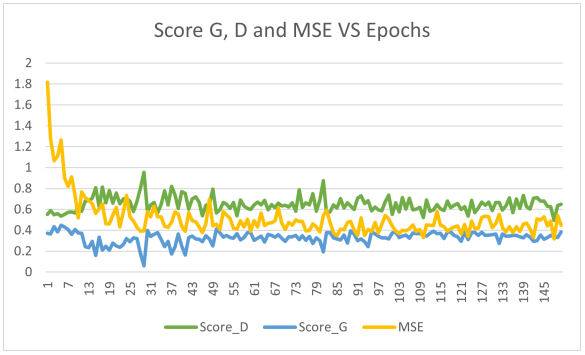

Figure 6: MSE vs epochs in training process

7. Limitations

The model tries to hallucinate the details. In other words, the model tries to ”manufacture” the details that has no relationship with the original HR image. A plausible reason behind this could be the perceptual loss. Perceptual loss, in simple terms is taking the mean squared error between real and generated images after passing them through several layers of VGG network. The later layers of VGG network are trained to identify abstract patterns in an image. So, when we try to minimize the MSE over VGG feature maps (the perceptual loss), we actually force certain patterns in an image. Hence, for an example, a very smooth pattern in an actual high resolution image could be mapped to artificially added high-frequency textured pattern. There is a two-fold impact of this. The first is, this could result in lower perceptual quality of an image. The second is, the SRGAN model is not suitable for application that require the exact details like surveillance and medical applications. Image 7 and image 8 are the results that indicates the problem:

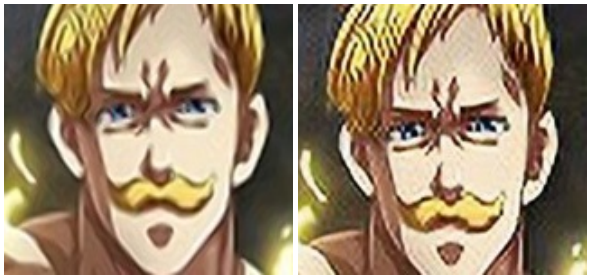

Figure 7: (a) Bicubic Interpolation (b) SRGAN Image

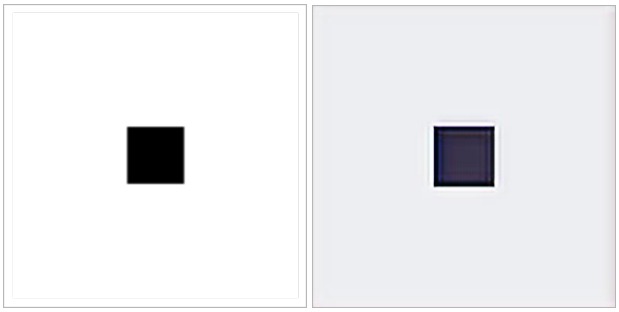

Figure 8: (a) Original HR Image (b) SRGAN Image

8. Conclusion

The paper focuses on perceptual quality of image rather than the computational efficiency or the exact details of the image. The paper proposes two models namely, SRResNet and SRGAN for the super-resolution task. While the former is optimized using conventional MSE loss and sets a new state-of-art for PSNR, the latter highlights the limitations of PSNR/MSE based optimization for the super-resolution. A major highlight of the paper is its novel perceptual loss function mentioned in the above sections. By experiments

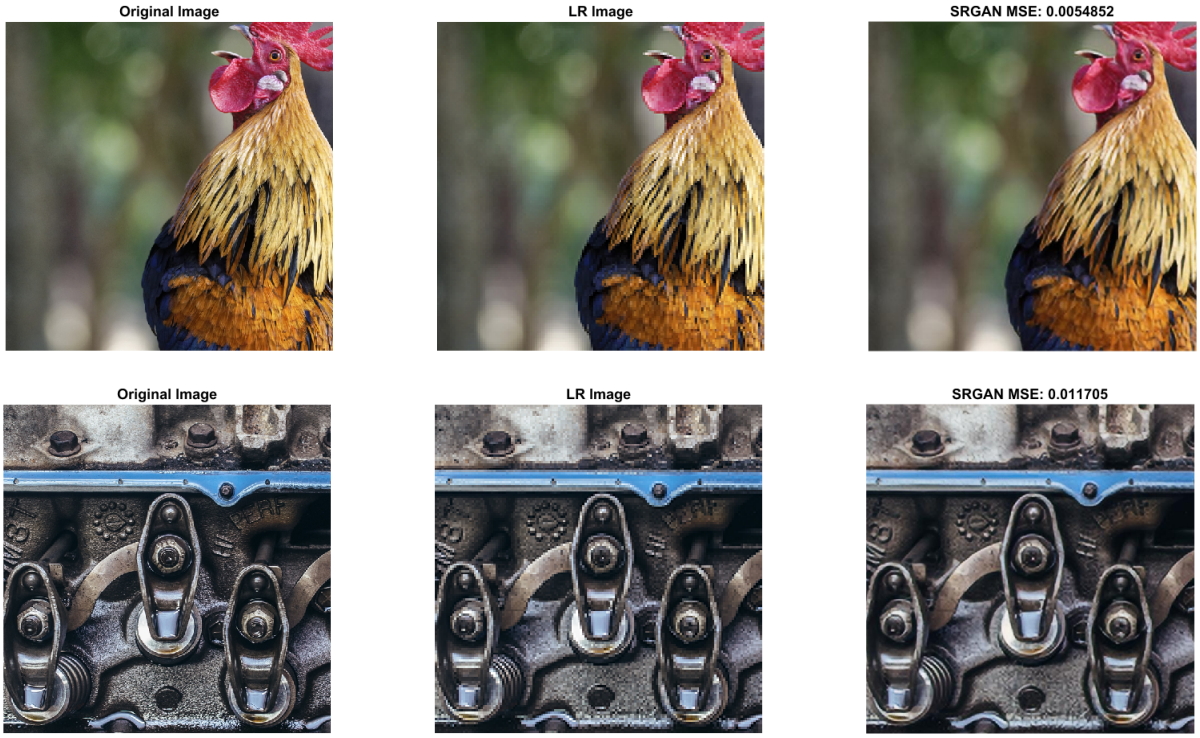

Figure 9: (a) Original HR Image (b) Corresponding LR Image (c) SRGAN Generated Image

and extensive mean opinion score analysis, the writers confirm that the images super-resolved by SRGAN are visually more similar to the original high-resolution images. It shall be noted that the SRGAN does not necessarily fill the "actual" details in the up-scaled image. It rather tries to hallucinate the details in other words, tries to fill the details that make the image more visually attractive. Such fill-up of the details might not solve the purpose where actual fine details are needed for an example surveillance or medical applications.

References

[1] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1122–1131, 2017.

[2] Joan Bruna, Pablo Sprechmann, and Yann LeCun. Super-resolution with deep convolutional sufficient statistics, 2016.

[3] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *CoRR*, abs/1501.00092, 2015.

[4] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. volume 9906, pages 391–407, 10 2016.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, 2014.

[6] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, Madison, WI, USA, 2010. Omnipress.

[7] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang. Convolutional sparse coding for image super-resolution. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015.

[8] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution, 2016.

[9] Jiwon Kim, Jung Lee, and Kyoung Lee. Deeply-recursive convolutional network for image super-resolution. pages 1637–1645, 06 2016.

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

[11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[12] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang. Deep networks for image super-resolution with sparse prior. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 370–378, 2015.