International Journal of Engineering Applied Sciences and Technology, 2020
Vol. 5, Issue 2, ISSN No. 2455-2143, Pages 159-167
Published Online June 2020 in IJEAST (http://www.ijeast.com)

SINGLE IMAGE SUPER-RESOLUTION AND COMPLEXITY-QUALITY TRADE OFF

Divyansh Shekhar Gaur
Department of CSE
IMS Engineering College,
Ghaziabad, Uttar Pradesh, India

Chandan Vishwakarma
Department of CSE, IMS
Engineering College,
Ghaziabad, Uttar Pradesh, India

Abhijeet Kumar
Department of CSE, IMS
Engineering College, Ghaziabad,
Uttar Pradesh, India

***Abstract*— Single Image Super-Resolution is a challenging task that aims to enhance the quality of the image from low resolution to high resolution. Super-resolution techniques can also be used as image lossy compression-decompression technique where low-res image is transmitted and decoded into high-res version at the received end. Various methods have been introduced till date, including algorithmic interpolation methods and deep learning methods. The algorithmic methods such as bilinear interpolation, bicubic interpolation, nearest neighbor and others provide fast processing while deep learning methods provide better quality of super-resolution image. We analyzed the time required to process images using various methods and compared them with the perceptual quality along with the variation in perceptual quality and PSNR metrics.**

Keywords— **Super-Resolution, Image Processing, Deep Learning, Image Super-Resolution, Machine Learning**

I. INTRODUCTION

On the basis of number of pixels in a digital image, a digital image can be a low-resolution (LR) or a high-resolution (HR). This categorization is subjective and there is no specific boundary defining the separation of low-resolution images from high-resolution images, however we will be using this idea as a means to compare two or more versions of same image. Higher-resolution images have more pixels than the low-resolution image, thus they contain more information and finer details about their subject [1]. Capturing low-resolution images is easier and cost effective, but trades off with quality. High-resolution images have better quality but need more resources in terms of camera quality, storage (memory), network bandwidth, *etc.* If we can reduce the number of pixels required to store a digital image, without losing on quality, it would reduce the overall cost of capturing, storing and transferring the images.

Super-resolution (SR) is the method of upscaling and improving the textures, details within an image. A low-resolution image is taken as an input and then it is upscaled to a higher resolution as the output. The details in the high-resolution output are filled in where the details are essentially unknown. It is the highly challenging task of estimating a high resolution (HR) image from its low-resolution (LR) counterpart. Single Image Super-Resolution (SISR) received substantial attention from within the computer vision research community and has a wide range of applications. Various interpolation algorithms have been devised to upscale lower resolution images into HR images. These algorithms are widely popular and have found their way as necessary tools included in most of the image processing libraries. Some of the popular algorithms are bicubic interpolation [3], bilinear interpolation [3], Lanczos [3,5], nearest neighbor interpolation [3], *etc.* We will briefly study these methods in Section 3.

Since introduction of SRCNN [7], deep convolution neural network (CNN) approaches have brought prosperous development. Various network architecture designs and training strategies have continuously improved the SR performance, especially the Peak Signal-to Noise Ratio (PSNR) value. However, these PSNR-oriented approaches tend to result in over-smoothed results without sufficient high-frequency details, since the PSNR metric fundamentally disagrees with the subjective evaluation of human observers [5]. One of the turning points in the way of pursuing visually more accurate results is SRGAN [9] model. The basic model is built with residual blocks and optimized using perceptual loss in a GAN [12] framework. With all these techniques, SRGAN significantly improves the overall visual quality of reconstruction over PSNR-oriented methods. It also uses deep residual blocks with skip-connection and diverges from MSE as the sole optimization target. It defines a novel perceptual loss using high-level feature maps of the VGG [11] network combined with a discriminator that encourages solutions perceptually hard to distinguish from the HR reference images, which is unique from previous works.

The objective is to improve the low-resolution image to be as good (or better) than the target, known as the ground truth, which in this situation is the original image we downscaled into the low-resolution image. However, there remains a considerable difference between SRGAN generated results and the ground-truth (GT) images. However, one limitation that super resolution models suffer with i.e. introduction of distortions called artifacts is still present in SRGAN.

In the ESRGAN [12] model, the key components of SRGAN have been revisited and the model was improved in three aspects. First, the network structure by introducing the Residual-in-Residual Dense Block (RDDB) [11], which is of higher capacity and easier to train, has been improved. Also, Batch Normalization layers have been removed and residual scaling has been used instead along with smaller initialization to facilitate training a very deep network. Second, the discriminator using Relativistic Average GAN [10], which learns to judge the image's reality relative to other image rather than the realness of the image, has been improved. Third, an improved perceptual loss by using the VGG features before activation instead of after activation as in SRGAN, is used.

II. BRIEF OVERVIEW OF INTERPOLATION METHODS

Interpolation is a method of constructing new data points

a) Nearest neighbor [3]

In this method, we expand the image to required size and fill in the gaps by duplicating the value of known pixel which is nearest to the required pixel.

$$I(x,y) = \text{nearestNeighbor}(x, y) \qquad (1)$$

This method is very simple and very fast. However, it introduces jagged edges and pixelated images. It is only good for simple images containing simple object shapes. Detailed images with curves and shades of intensity suffer from loss of quality when processed with nearest neighbor technique.

b) Bilinear Interpolation [3]

In the field of numerical analysis, bilinear interpolation is the extension of linear interpolation in which we interpolate a bivariate function f(x,y) over given set of data points first along one axis and then the other axis. This method is also called bilinear filtering or bilinear texture mapping.

It is assumed that we know the value of f at the four points $Q_{11} = (x_1, y_1)$, $Q_{11} = (x_1, y_2), Q_{21} = (x_2, y_1), and Q_{22} = (x_2, y_2)$. We can write the solution to the interpolation problem as

$$\mathrm{f}(x, y) = a_0 + a_1\mathrm{x} + a_2\mathrm{y} + a_3\mathrm{xy} \qquad (2)$$

Where the solution of the following linear system gives the value of the coefficients

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1y_1 \\ 1 & x_1 & y_2 & x_1y_2 \\ 1 & x_2 & y_1 & x_2y_1 \\ 1 & x_2 & y_2 & x_2y_2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \mathrm{f(Q_{11})} \\ \mathrm{f(Q_{12})} \\ \mathrm{f(Q_{21})} \\ \mathrm{f(Q_{22})} \end{bmatrix} \qquad (3)$$

c) Bicubic Interpolation [3]

Bicubic interpolation is an extension of cubic interpolation. It is slower than bilinear interpolation and nearest neighbor interpolation and is used when speed is not a great issue. However, it is still faster than deep learning methods. Bicubic interpolation considers 16 pixels (4x4 grid) and results in lesser artifacts and smoother image.

Suppose the function values f and the derivatives f_x, f_y and f_{xy} are known at the four corners (0,0), (0,1), (1,0) and (1,1) of the unit square, the interpolated surface can be written as

$$p(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j \qquad (4)$$

The interpolation problem consists of determining the 16 coefficients (a_{ij}). The equations for those can be found in detail at ref. 3.

III. BRIEF OVERVIEW OF DEEP LEARNING METHODS

With the introduction of SRCNN [4] (deep convolutional neural network for super-resolution) in 2014, the field of super-resolution achieved a breakthrough. Several deep learning methods including residual networks and GANs have been explored since then as the solution to the ill-posed problem of super-resolution. The power of neural networks to automatically figure out the transformation, given the dataset has proved to be incredibly useful to achieve higher quality SR images having greater amount of detail, sharpness, and lesser number of artifacts. Let's go through some of the remarkable approaches below-

a) SRCNN [4]

SRCNN was a breakthrough in the field of super-resolution, unlocking the potential of deep convolutional neural networks for super-resolution applications and other abstract computer vision problems. SRCNN architecture was simple in terms of computational power and resources required, yet effective in quality. It had two convolution layers, the first one extracted the features from the low-resolution image, while the second layer maps these feature maps to high resolution. The architecture of SRCNN is shown in **Fig. 1.**

It used the PSNR as loss function. PSNR (Peak Signal to Noise Ratio) is based on mean square error (MSE) between the ground truth (HR image) and image obtained through super-resolution (SR image). **Reference 5** showed that distortion measures such as PSNR and SSIM vary greatly from the perceptual quality (human perception of image quality) of images.

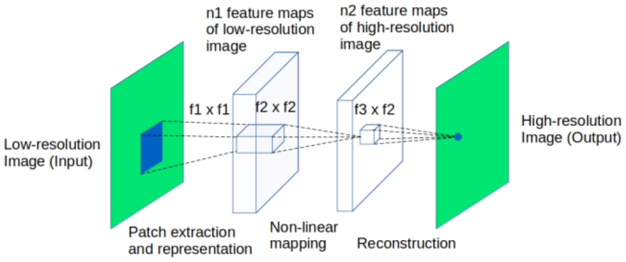
n1 feature maps of low-resolution image
n2 feature maps of high-resolution image
f1 x f1
f2 x f2
f3 x f2
Low-resolution Image (Input)
High-resolution Image (Output)
Patch extraction and representation
Non-linear mapping
Reconstruction

Figure 1 Two-layer architecture of SRCNN. Adapted from Ref. 7

b) *Residual Dense Network (RDN) for Super-Resolution*

Yulun *et. al.* proposed RDN for SISR [14] which used residual blocks to allow more layers in the network. In residual blocks, residuals from the previous layers are added to output of later layers so as to preserve the outputs from the previous layers. It was also based on PSNR maximization by minimizing MSE between HR (ground truth) and SR image (RDN output). Fig. 2 (a) shows structure of a residual block, Fig. 2(b) shows a dense block and Fig. 2(c) shows how residual block and dense blocks have been merged together to form Residual Dense Block (RDB) which is the basic building block of RDN. The complete architecture is shown in Fig. 2(d).

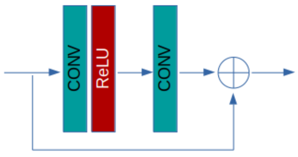

Figure 2 (a) Residual Block [14]

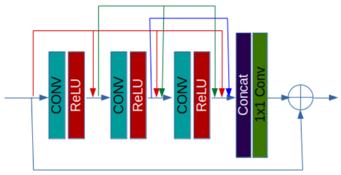

Figure 2(b) Dense Block [14]

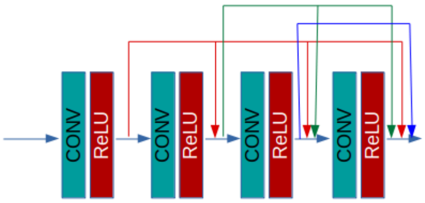

Figure 2(c) Residual Dense Block [14]

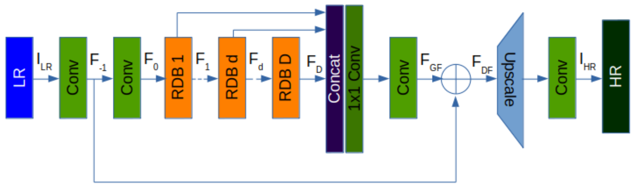

Figure 2(d) Residual Dense Network Architecture [14]

c) *SRGAN [6]*

SRGAN was the first architecture to explore the use of generative adversarial networks for SISR applications. It also used perceptual index approach instead of distortion (PSNR) based approach. The improvement in quality of SR images achieved using this architecture created a new benchmark for all future works. It used 5 residual blocks and a skip connection from first activation layer in the generator to the first batch normalization (BN) [5] layer after the residual blocks (fig. 3). This BN layer is followed by two up-sampling blocks to achieve 4x up-sampling. The output of the generator is then fed to a discriminator which has a VGG-style architecture containing stack of convolution activation and pooling layers. The discriminator uses VGG feature loss as proposed in refs. 10 as perceptual loss and predicts if the output of generator appears to be a real image or a fake image.

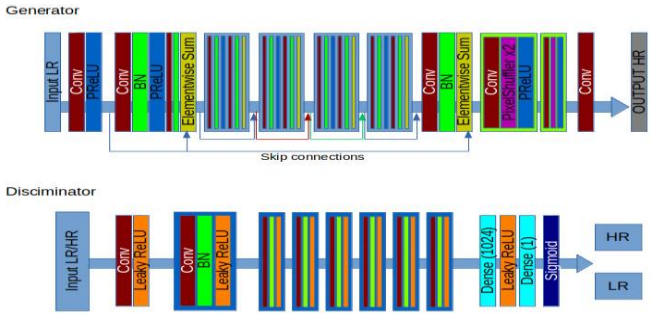

Figure 2 Architecture of Generator and Discriminator networks in SRGAN. Adapted from Ref. 9.

d) ESRGAN [12]

Enhanced Super-Resolution Generative Adversarial Network is an improvement over SRGAN. It uses Residual-in-Residual Dense Blocks (RRDB) [12] and has a greater number of layers that SRGAN. It removes the batch normalization layer [5] and replaces discriminator with relativistic discriminator [13]. Hence, it is based on Relativistic Average GANs (RaGAN) [6]. It focuses on a balance between perceptual index and distortion metrics such as PSNR, SSIM, *etc.* through network interpolation [12]. The authors claim that ESRGAN produces visually more perceptive super resolution image than its predecessor SRGAN. The RaGAN is also harder to train than simple discriminator used by SRGAN.

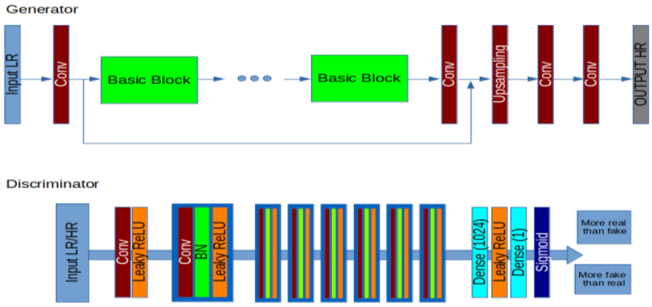

Figure 3 Architecture of generator (top) and discriminator (bottom) of ESRGAN. Adapted from Ref 12.

IV. METHODOLOGY AND RESULT

We used pre-trained Residual Dense Network (RDN) and ESRGAN trained on Div2K dataset. The RDN network we used, is designed for 2x up-sampling, so we used two RDN networks back-to-back to get 4x upscaled images. The interpolation methods and ESRGAN are tuned to produce 4x up-sampled images. We fed random 50 images from Set5, Set14, Urban 100, BSD 100, Sun-Hays 80 datasets.

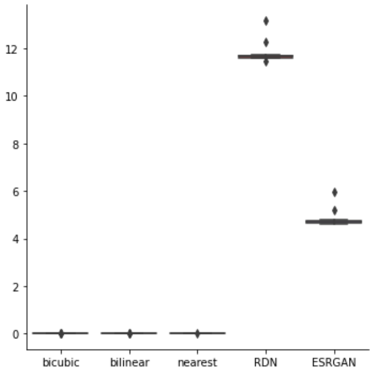

Figure 5(a) Time taken (in sec) to process each image

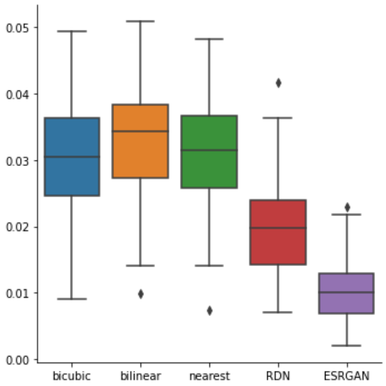

Figure 5(b) Perceptual quality in each method

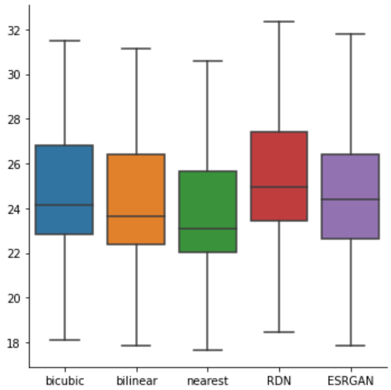

Figure 5(c) PSNR score of each method

We recorded the time taken to generate up-sampled images by each method for each image and then performed box plot analysis to compare how much the time taken, perceptual quality and PSNR obtained vary from model to model. We also verified the box plot analysis by visually verifying the generated up-sampled images.

The hardware we used had Intel Xeon 2.30 GHz Dual Core with 11 Giga bytes of RAM. The time taken to compute the results will vary greatly from processor to processor as it is a function of processor throughput, but the gain in perceptual quality will not be much affected.

We recorded the time in seconds as a measure of time-complexity of super-resolution methods, and perceptual loss from VGG-19 (trained on ImageNet) as inverse of perceptual quality (lower loss means higher perceptual quality).

From the box plots for time taken (in processing) in Fig 6(a) by different methods to process 50 images, we can see that the time taken by interpolation methods is negligible in comparison to deep learning methods. We also observe that 4x up-sampling using dual RDN model back-to-back is a lot slower and is twice as large than 4x up-sampling ESRGAN model.

From Fig 5(b) and 5(c), we can see that nearest neighbor method performs bad in terms of both perceptual quality and PSNR. ESRGAN achieves highest perceptual quality (lowest perceptual loss) and lower PSNR than RDN network. Hence, we can deduce that ESRGAN provides the best in terms of perceptual quality and is better than RDN in terms of time.

From Fig. 5(a), (b) and (c), we can infer that interpolation methods are better in terms of time complexity but the deep learning methods can achieve better results in terms of quality.

In Fig. 6, we provide some of the up-sampled images along with low resolution (LR) and ground truth (HR) images for visual comparison of perceptual quality. The findings from box plot analysis of perceptual quality can be verified from visual comparison of images. In these images, we can see that deep learning methods achieve more detailed up-sampled images than interpolation techniques, and produce smoother results with lesser artifacts.

The original data points used to create box-plots is provided in appendices A, B and C. The values can deviate slightly with each execution.

V. CONCLUSION

Here, we reviewed existing super-resolution techniques and showed that deep learning methods are quite costly in terms of time and hardware resources required, but also give better results. We can use the traditional interpolation methods when time-complexity is of utmost importance and very high quality of up-sampled image is not required. When more emphasis is on the quality of resultant images, then we can use state-of-the-art deep learning methods but suffer in terms of time and resources required.

VI. FUTURE WORK

We performed our analysis on limited dataset and limited number of super-resolution techniques. In future, we hope that the analysis will be expanded to other techniques as well and the performance of these techniques be evaluated on different hardware and operating systems to find a more substantial evidence of the complexity-quality tradeoff.

Low-RES	BICUBIC	BILINEAR	NEAREST	RDN	ESRGAN	High-res

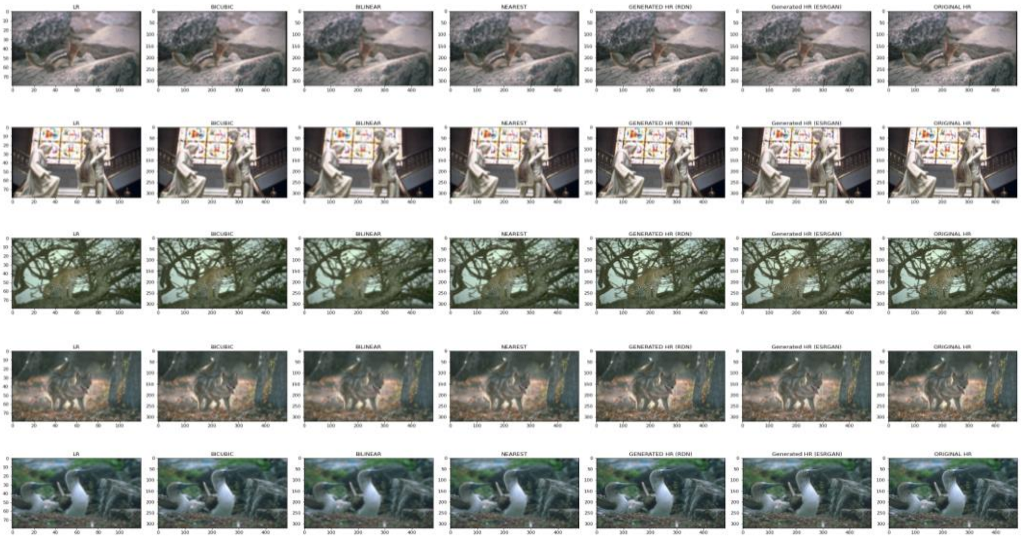

Figure 4 Visual Comparison of Various Super-Resolution Techniques

Appendix A. Original Time Data (in seconds) Obtained from Processing 50 Images

S. No.	Bicubic	Bilinear	Nearest Neighbor	RDN	ESRGAN
0	0.007579383999999578	0.0027717590000015946	0.001980271999997285	12.267165829999996	5.965925309000006
1	0.004460104000003184	0.0025869160000127067	0.0016992549999912399	11.670979426000002	4.72672398200001
2	0.003562477000002673	0.0026453400000150395	0.0016583780000019033	11.676178948	4.7049235309999915
3	0.003661230000005844	0.0026667659999759508	0.0017566300000169122	11.681173274000002	4.735065338999988
4	0.0027280200000063815	0.0026004220000004352	0.001635368999984621	11.664747739999996	4.725968684000009
5	0.003219570999988264	0.002894799999978659	0.0016337600000042585	11.717679228999998	4.717264918000012
6	0.003128445000015745	0.002607590000025084	0.0016603229999532232	11.679592281999987	4.727130561000024
7	0.0026326340000082382	0.0025923050000074	0.0017452220000109264	11.667579013000022	4.728191941000034
8	0.003304046000039307	0.0026257490000034522	0.0017620670000724203	11.683989728000029	4.732709289000013
9	0.003042794999942089	0.002584578999972109	0.0016913480000084746	11.689055863000021	4.797432963999995
10	0.002660847000015565	0.0026366540000708483	0.0016706630000271616	11.670868238000025	4.795673203999968
11	0.003133197000011023	0.0027088250000133485	0.001756923000016286	13.17593035099992	4.766447483000093
12	0.003313435999984904	0.0025921090000338154	0.0017668750000439104	11.688464582000051	4.749271402999966
13	0.0025910509999675924	0.0027307149999842295	0.0017063450000023295	11.707560759999978	4.762258969000072
14	0.003260248999936266	0.002649404999942817	0.0017153289999214394	11.684746931999939	4.746063554999978
15	0.0033310280000478087	0.002634581000052094	0.0017029399999728412	11.694885675000023	4.724702725000043
16	0.003197731999989628	0.002638548000049923	0.0016698550000455725	11.66526243599992	4.724622724000028
17	0.002781430000140972	0.0025401999998848623	0.0016890570000214211	11.667619637999906	4.7277002509999875
18	0.0034838589999708347	0.002781148999929428	0.0016514700000698213	11.678317593999964	4.7298642369999016
19	0.0032999410000229545	0.002920450999909008	0.0016869580001639406	11.639928214000065	4.70318687200006
20	0.0030046609999772045	0.002763513999980205	0.0017444420000174432	11.675189443999898	4.7162827570000445
21	0.003192346000105317	0.0026496420000512444	0.001711267000018779	11.679639852000037	4.731588795999869
22	0.003348056000049837	0.0027415100000780512	0.0018660480000107782	11.696564522000017	4.725245137999991
23	0.003036078999912206	0.002805999999964115	0.0017590509999081405	11.712209972999972	4.74596738199989
24	0.003343012999948769	0.002578309000000445	0.0017521830000077898	11.701053499999944	4.700680191999936
25	0.003285665000021254	0.002684975999954986	0.0017016580000017711	11.70467726700008	4.69624333999991
26	0.0033988610000506014	0.0027995049999844923	0.0017587620000085735	11.670957043000044	4.704917918000092
27	0.003295038000032946	0.002643639000098119	0.0018034849999821745	11.71531685900004	4.677353202999939
28	0.0024797940000098606	0.002642747000209056	0.001637291999941226	11.610760374999927	4.659262024999862
29	0.0033323499999369233	0.002725056999906883	0.0016693960001248342	11.639175912000155	4.641894183999966
30	0.0032689719998870714	0.0026199270000688557	0.0016578940001181763	11.615398824000067	4.646078959000079
31	0.00258060100009061	0.0025653779998719983	0.0016430789999049011	11.626699164000001	4.651012693999974
32	0.004257052999946609	0.0026445950002198515	0.001749729000039224	11.641495536000093	4.646556278999924
33	0.003503100999751041	0.002673276000223268	0.0016615280001133215	11.651674913000079	4.65980850599999
34	0.0030085409998719115	0.002634812999986025	0.0016506739998476405	11.630830005000007	4.640224993000174
35	0.0032317609998244734	0.0026270640000802814	0.0016732630001570215	11.614361313000245	4.624586708000152
36	0.0030872900001668313	0.002583370999673207	0.001690482000412885	11.638729535000039	4.629252958000052
37	0.0029483840003194928	0.0026098460002685897	0.0017133100000137347	11.634634223000376	4.655748230999961
38	0.0032195009998758906	0.002581489000021975	0.0017313710000053106	11.603257128000223	4.609997676999683
39	0.0025549239999236306	0.002630225000302744	0.00171004300000277	11.603884995000044	4.6108390299996245
40	0.0028480830001171853	0.0029965240000819904	0.0017172170000776532	11.601651047999894	4.613169876000029
41	0.0031398520000038843	0.0025745569996615814	0.0016443779995825025	11.616923145999863	4.614808810999875
42	0.003061182999772427	0.002574275999904785	0.0016482539999742585	11.621157259000029	4.668053205999968
43	0.0032159930001398607	0.0025530800003252807	0.0016387010000471491	11.611122032999901	4.670242114999837
44	0.0030894380001882382	0.0026229919999423146	0.0016373600001315936	11.61877133400003	4.625280145000033
45	0.002948249999917607	0.0025394929998583393	0.001625917000183108	11.606779457999892	4.613760662000004
46	0.0030272170001808263	0.0025625139996918733	0.0016209059999710007	11.615923672999998	5.214159522000045
47	0.003382482000233722	0.0027244490001976374	0.00167377400020996	11.472350108999763	4.620119126000191
48	0.0030329430001074797	0.0025950740000553196	0.0016201589996853727	11.61761556600004	4.608540755999911
49	0.0031056759999046335	0.002595984999970824	0.0018702729998949508	11.644844248000027	4.653949328000181

Appendix B. Original Perceptual Loss Data Obtained by Processing 50 Images

S. No.	Bicubic	Bilinear	Nearest neighbor	RDN	ESRGAN
0	0.038261037319898605	0.04167238622903824	0.039311982691287994	0.02915184386074543	0.012914184480905533
1	0.0313359759747982	0.0350642055273056	0.0364922471344471	0.022674240171909332	0.011459116823971272
2	0.03198288381099701	0.0341142937541008	0.026952123269438744	0.023668060079216957	0.010385812260210514
3	0.026054661720991135	0.030563456937670708	0.03302658349275589	0.015361322090029716	0.009224506095051765
4	0.025406518951058388	0.030376529321074486	0.032348550856113434	0.017951471731066704	0.011726003140211105
5	0.012209104374051094	0.014033729210495949	0.016213618218898773	0.0069831921719014645	0.0025020786561071873
6	0.031686462461948395	0.036059457808732986	0.04023713245987892	0.020647548139095306	0.016098476946353912
7	0.03249253332614899	0.036137904971838	0.03104800544679165	0.019729439169168472	0.012392694130539894
8	0.025914998725056648	0.03095337189733982	0.03129902482032776	0.014193966053426266	0.008387911133468151
9	0.02294543758034706	0.0272988248616457	0.027850614860653877	0.01417218241840601	0.006671884097158909
10	0.04186218976974487	0.044680800288915634	0.03396793454885483	0.027070026844739914	0.011876747012138367
11	0.03212607279419899	0.035372957587242126	0.0317436158657074	0.018660778179764748	0.00857567973434925
12	0.018229283392429352	0.021413123235106468	0.02268354408442974	0.012598770670592785	0.006495893932878971
13	0.031110674142837524	0.03461355343461037	0.026381921023130417	0.020459862425923347	0.006556888576596975
14	0.022727860137820244	0.02653936669230461	0.02283789962530136	0.014895826578140259	0.0076104020699858665
15	0.03202320262789726	0.034589733928442	0.03664625063538551	0.02102801389992237	0.012862556613981724
16	0.018954798579216003	0.02176116406917572	0.02021619863808155	0.010187895968556404	0.0037669269368052483
17	0.017117753624916077	0.01965927518904209	0.014688192866742611	0.010241934098303318	0.00534470472484827
18	0.030465468764305115	0.033204153180122375	0.028635399416089058	0.01895425096154213	0.008250141516327858
19	0.01714985817670822	0.019025184214115143	0.01549491286277771	0.010813349857926369	0.004198748152703047
20	0.043113693594932556	0.047667596489191055	0.04286706820130348	0.029056066647171974	0.013659099116921425
21	0.04168746992945671	0.04259819537401199	0.02992876060307026	0.03531350940465927	0.01698603294789791
22	0.025131074711680412	0.027401212602853775	0.024830052629113197	0.019044043496251106	0.010169940069317818
23	0.0389886237680912	0.0444825105369091	0.04126613959670067	0.025563230738043785	0.012953992933034897
24	0.037697646766901016	0.04211295768618584	0.048266369849443436	0.02267105132341385	0.013124856166541576
25	0.02933507040143013	0.03207510709762573	0.03456082567572594	0.019985681399703026	0.012575184926390648
26	0.033262431621551514	0.03634350374341011	0.03368643298745155	0.02207004278898239	0.01073510479182005
27	0.02562703751027584	0.02898176573216915	0.029385192319750786	0.014594183303415775	0.007981020957231522
28	0.01657787710428238	0.017434537410736084	0.01402782741934061	0.009827432222664356	0.002699281321838498
29	0.03333497419953346	0.03450945392251015	0.03196341544389725	0.030242513865232468	0.01478023175150156
30	0.02453155443072319	0.026979103684425354	0.025521056726574898	0.017853451892733574	0.011746951378881931
31	0.04669450595974922	0.049870382994413376	0.04710381478071213	0.03635929897427559	0.02292053773999214
32	0.039124999195337296	0.04350616782903671	0.03824332356452942	0.024110758677124977	0.015316716395318508
33	0.027588622644543648	0.030586518347263336	0.025542253628373146	0.01797265000641346	0.00925983302295208
34	0.019815336912870407	0.02337249554693699	0.029903221875429153	0.009879645891487598	0.003953264094889164
35	0.02922949381172657	0.03204585239291191	0.03262379392981529	0.01992223411798477	0.008982755243778229
36	0.009035594761371613	0.0099562993273139	0.0074426522478461266	0.007089640479534864	0.0020106337033212185
37	0.029102833941578865	0.032132744789123535	0.03009197860956192	0.018170613795518875	0.008483447134494781
38	0.043524231761693954	0.04801403731107712	0.04636494442820549	0.024845421314239502	0.021845366805791855
39	0.030667267739772797	0.037185702472925186	0.044275883585214615	0.012498250231146812	0.0020299789030104876
40	0.031998708844184875	0.03575841337442398	0.028723526746034622	0.022606493905186653	0.009166468866169453
41	0.037391405552625656	0.03937724605202675	0.03484807163476944	0.02820504456758499	0.014385579153895378
42	0.03959216549992561	0.042115453630685806	0.026450658217072487	0.027284972369670868	0.009806019254028797
43	0.027831509709358215	0.030846239998936653	0.04135167598724365	0.01724926009774208	0.009947090409696102
44	0.03686871379613876	0.0385451577603817	0.036088377237319946	0.028912890702486038	0.021512124687433243
45	0.021242687478661537	0.024565843865275383	0.020117683336138725	0.012137715704739094	0.005050988402217627
46	0.04940325394272804	0.050904177129268646	0.04476860910654068	0.04174376651644707	0.017624253407120705
47	0.03022598847746849	0.03502681106328964	0.033902429044246674	0.019718900322914124	0.01200183853507042
48	0.03495444729924202	0.03791012987494469	0.03715262562036514	0.023465655744075775	0.011546168476343155
49	0.01836214028298855	0.02260730043053627	0.02392416074872017	0.011051738634705544	0.004673904739320278

Appendix C. Original PSNR Data Obtained by Processing 50 Images

S. No.	Bicubic	Bilinear	Nearest Neighbor	RDN	ESRGAN
0	21.668415443932613	21.397282471087685	21.013702052369002	22.214222091383718	21.53143480365145
1	25.40425095447327	24.981881542544237	24.39691501600842	26.34221976182182	25.961627588894807
2	23.884533528787717	23.572576972988976	23.12016528987284	24.524289973171772	24.377121191625285
3	27.98017481185701	27.44347434424091	26.647819630156565	28.814401796652795	27.961252000170354
4	24.10093200103356	23.731050139404672	23.254676739083713	24.605878107199043	23.67621370980739
5	29.997683387311476	29.51478274205156	28.481189411055354	31.246601139771784	30.77647223634713
6	23.631251402317204	23.15673495807204	22.739033930890017	24.51097642605017	23.30891684158514
7	21.23464025760147	20.97736613892596	20.819947776083644	21.59751018407578	20.816296445064246
8	24.833240414939276	24.2805575808606	23.66586082767104	25.714242552268246	24.99839178178105
9	26.971939420326635	26.505822477014437	25.981035861092497	27.574161524723554	26.786828368819407
10	20.832331725170654	20.46032285080306	19.846619809797048	22.13192081508661	22.235951000216126
11	24.733514487747055	24.1345569305974	23.658940728946018	25.59661137405648	24.83283702325201
12	27.552725632112804	26.943520318913894	26.308979411975702	28.29421680282444	26.924194232585865
13	25.961078708214835	25.527504782358346	25.059867437391283	26.96595646593082	26.352127542316634
14	22.993930019146873	22.665180354612566	22.297100821447007	23.509203099283308	22.622543112791817
15	21.112951036510644	20.523572625380474	20.02503328282284	22.39713030446216	21.674520252991304
16	30.713315588286676	30.263522198177043	29.444431154737913	31.759044556957466	30.592230223950082
17	29.403854936401757	28.991640410916624	28.25012138100636	30.066155128700082	29.009893252652688
18	22.892095829766145	22.527808440649373	21.986960144800957	23.962082603676368	23.6590222152119
19	23.25829734785657	22.820819026741702	22.429634455885722	24.08142653732365	23.631395894869527
20	19.423166330627886	19.062259164359567	18.725962835185232	20.10115171920695	19.74552159446003
21	21.25212026667242	20.889365591099825	20.57140348384278	22.005589236897176	21.550420018529863
22	23.16071409992478	22.927827026937084	22.67458527398416	23.450584712257815	22.710415791662456
23	23.69158050019942	23.24476711151291	22.607714307611744	24.482766184809634	23.945104992306362
24	24.180893079996693	23.539555455594673	22.672970701521304	25.44449712865554	24.36272712093018
25	24.53652524778152	24.031908606151525	23.59003526404954	25.396818910435073	24.31266354679262
26	25.11097559552165	24.64788365452392	23.974697158229574	25.865532088540554	24.758561503566924
27	27.04159898780602	26.56657253446381	26.011790927988084	27.726056690683887	26.427950268186592
28	27.027392308358674	26.53233052459362	25.769456217628253	29.290982069491818	29.53949324505374
29	18.10077024083107	17.867989990746263	17.63547502674127	18.432357036869483	17.867309843520445
30	27.152723565886056	26.675070356880024	26.11912141516095	27.88128060347742	26.49249577901298
31	19.247869786030346	19.05698071203789	18.981856630086885	19.540971023727597	18.875501088624755
32	22.78960654186967	22.289704211631577	21.85843867715907	23.48594073635535	22.557076984926468
33	21.96408705553975	21.671420574929037	21.33783632625951	22.659254907447256	22.008693667446654
34	29.58413472533795	28.88372976605813	27.578344779911845	31.584504744419565	31.0712055600682
35	24.173499427706098	23.671891653337394	23.068827207221606	25.019453633022245	24.376684518976596
36	26.25745399603676	26.120744427073745	26.00089709044525	26.497577457000784	25.543052818859977
37	25.56595804240272	25.222316010902798	24.691385016845885	26.283589873433737	25.616791132152528
38	25.88956743311689	25.500823675704027	25.04526363333522	26.67744966353051	25.10297676494809
39	27.64762385096967	26.705410179401028	25.384261588690777	30.410404545375176	31.551098759554343
40	23.631772866883033	23.180705084241495	22.728719123977584	24.70779118350073	24.567978971176963
41	23.203233667369766	22.837973468518197	22.358621257557367	23.974364818802233	23.440690486748824
42	24.08539931870684	23.762282428188644	23.369309747654164	24.86056324236479	24.54853352623992
43	24.164395827271935	23.470702272216965	22.638105495226615	25.39008526374918	25.47656628930989
44	22.59032902028713	22.32015390252291	22.120665625589744	22.952726720578596	22.489422054608916
45	31.489697779634728	31.134146242128388	30.609863594891454	32.34943715919747	31.81016627724077
46	20.302488068687893	19.946847128115287	19.682712100900787	20.93727771495656	20.511983981771998
47	22.918406012962528	22.568179892871246	22.287695720932025	23.352038382774346	22.646772869972054
48	21.953505770978502	21.28474645333114	20.616147779240634	23.97438667959934	24.130222404335065
49	29.51587826754741	29.10213767690026	28.46593949958628	30.183808297756713	28.736823800221458

VII. ACKNOWLEDGEMENT

We sincerely thank our guide Dr. Pankaj Aggarwal (Professor at IMS Engineering College, Ghaziabad) for his patient guidance and constructive suggestions for the research in the area of Image Super-Resolution. This research is made possible due to his willingness to devout his time generously.

VIII. REFERENCES

1. Sung Cheol Park, Min Kyu Park and Moon Gi Kang, 2003, "Super-resolution image reconstruction: a technical overview," 10.1109/MSP.2003.1203207
2. Pascal Getreuer, 2011, " Linear Methods for Image Interpolation, Image Processing On Line", 10.5201/ipol.
3. Upendra bhatt, Dr.Annapurna singh, 2016," A Review on Image Resolution Enhancement Methods in Spatial and Frequency Domain", (Pg455-499)
4. C. Dong, C. C. Loy, K. He and X. Tang, 2015, "Image Super-Resolution Using Deep Convolutional Networks", 10.1109/TPMAI.2015.2439281
5. Y. Blau and T. Michaeli, 2019, "The Perception-Distortion Tradeoff",10.1109/CVPR.2019.00831
6. Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi, 2017,"Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network", 10.1109/CVPR.2017.19
7. Chia-Hung Yeh, Chih-Hsiang Huang, Li-Wei Kang, 2019,"Multi scale deep residual Learning- Based Single image Haze Removal via Image Decomposition", 10..1109/TIP.2019.2957929
8. Oleksii Sidorov, Jon Yngye Hardeberg, 2019, "Deep Hyperspectral Prior- Single Image Denoising, Inpainting, Super - Resolution", 10.1109/ICCVW.2019.00477
9. Alireza Esmaeilzehi, M.Omair Ahmad, M.N. S. Swamy,2019, "SRSubBandNet: A New Deep Learning Scheme for Single Image Super Resolution Based on Subband Reconstruction", 10.1109/ISCAS.2019.8702351
10. Donghyeon Lee, Sangheon Lee, Hoseong Lee, Kyujoong Lee, Hyuk-Jae Lee, 2019, " Resolution-Preserving Generative Adversarial Networks for Image Enhancement", 10.1109/ACCESS.2019.2934320
11. Y. Zhang, Y. Tian, Y. Kong, B. Zhong and Y. Fu, 2018, "Residual Dense Network for Image Super-Resolution," 10.1109/CVPR.2018.00344
12. Xein Deng, 2019, "Enhancing Image Quality via Style Transfer for Single Image Super-Resolution", 10.1109/LSP.2018.2805809
13. Harsh Nilesh Pathak, Xinxin Li, Shervin Minaee, Brooke Cowan, 2018," Efficient Super Resolution for Large-Scale Images Using Attentional GAN", 10.1109/BigData.2018.8622477
14. Thomas Richter, Annelie Habermann, André Kaup, 2015," Super-resolution for mixed-resolution multiview images using a relative frequency response estimation method", 10.1109/VCIP.2015.7457827