Development of Deep Learning Techniques for Image Super Resolution

Alireza Esmaeilzehi

A Thesis
in
The Department
of
Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy (Electrical and Computer Engineering) at
Concordia University
Montréal, Québec, Canada

January 2022

CONCORDIA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Alireza Esmaeilzehi

Entitled: Development of Deep Learning Techniques for Image Super Resolution

and submitted in partial fulfillment of the requirements for the degree of

Doctor Of Philosophy (Electrical and Computer Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

______________________________Chair
Dr. Alex De Visscher

______________________________ External Examiner
Dr. Wasfy Boushra Mikhael

______________________________ External to Program
Dr. Chun-Yi Su

______________________________ Thesis Co-Supervisor
Dr. Omair Ahmad

______________________________ Thesis Co-Supervisor
Dr. M.N.S. Swamy

______________________________ Examiner
Dr. Hassan Rivaz

______________________________ Examiner
Dr. Wei-Ping Zhu

Approved by ______________________________
Dr. Wei-Ping Zhu, Graduate Program Director

2/16/2022 ______________________________
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Development of Deep Learning Techniques for Image Super Resolution

Alireza Esmaeilzehi, Ph.D.

Concordia University, 2022

The images to be used in many of the real-life applications, such as medical imaging, intelligent transportation systems and space explorations, are not of a sufficient quality in view of the degradation processes associated with the image capturing devices. In recent years, deep neural networks have emerged as a sophisticated tool for image restoration. However, many of the existing deep neural networks for image restoration employ a large number of parameters for providing high performance, thus prohibiting their deployment in applications with the constraints on memory and power consumption. Hence, the design of high-performance image restoration convolutional neural networks that employ small number of parameters is of paramount importance. As the performance of deep networks is closely related to the richness of features produced by them, the objective of the thesis is to design deep image restoration neural networks that are capable of producing rich sets of features by using only a small number of parameters. In this thesis, this objective is met by using the suitable prior information associated the degradation processes of the image capturing devices. Three specific degradation models, namely, bicubic downsampling, Gaussian blurring coupled with downsampling and JPEG compression blocking, are considered in designing a number of deep light-weight image restoration networks.

With regard to the first degradation model, i.e., bicubic downsampling operation, several image super resolution networks using the different prior information about this operation are developed. Specifically, four different prior information, namely, multi-scale feature generation, guided feature generation, efficient feature fusion and sparsity prior, are used for developing light-weight image super resolution networks. As to the second degradation model, i.e., Gaussian blurring coupled with downsampling, two deep networks, in which the blurred version of the high-quality images are used in the context of global residual learning as the prior information, are proposed. Finally, with respect to the third degradation model, i.e., JPEG blocking artifacts, two information, namely, robust features generated by the maxout activation units and the high-frequency components generated by the fractal neural networks, are used as prior information to propose couple of image restoration networks.

Extensive experiments are carried out to validate the effectiveness of the various ideas and schemes developed in this thesis for improving the quality of degraded images.

Acknowledgments

I would like to first thank to my PhD supervisors, Prof. M. Omair Ahmad and Prof. M.N.S. Swamy, who infinitely helped me in conducting research, presenting its results in an efficient manner, and supported me in any aspect during my PhD. They were available at any time I needed them and provided all the tools and facilities for conducting my research. In summary, I am extremely happy that have carried out my PhD under their supervisions.

I would like to also thank Concordia University and its Department of Electrical and Computer Engineering, that have provided a desirable environment for me to achieve my goals in PhD.

Finally, I would like to thank my amazing family, my parents (Naghmeh and Nabi), my grandparents (Zivar and Mohammadreza), my awesome uncle (Dr. Abbass) and his family, and my brother (Mohammadreza), for all their supports and consultations. I could not imagine to reach this point of my life without them.

I dedicate this thesis to my family and my supervisors.

Contents

List of Figures

List of Tables

List of Symbols

G_h	Horizontal Gradient
G_v	Vertical Gradient
Shr_α	Soft Shrinkage Operator
$HShr_\alpha$	Hard Shrinkage Operator
σ	Gaussian Blurring Standard Deviation
$\mathbf{H}$	Blurring Operator
$\mathbf{D}$	Downsampling Operator
f_1	Super Resolution Function
f_2	Degradation Function
f_3	Interpolation Function
$\oplus$	Morphological Dilation Operation
$\ominus$	Morphological Erosion Operation
sgn	Sign Function
$\|\|\|_1$	ℓ1 Norm
$\|\|\|_2$	ℓ2 Norm

List of Abbreviations

AP	Average Pooling
CARN	Cascaded Residual Network
$Conv$	Convolutional Layer
DBPN	Deep Back Projection Network
$DiConv$	Dilated Convolutional Layer
DRCN	Deep Recursive Convolutional Network
DRMU	Deep JPEG Image Deblocking using Residual Maxout Units
DRN	Dual Regression Network
DRRN	Deep Recursive Residual Network
DS	Depth-to-Space Operation
EDSR	Enhanced Deep Super Resolution Network
EFFRBNet	Super Resolution Network Edge-assisted Feature Fusing Residual Blocks
FSRCNN	Fast Super Resolution Convolutional Neural Network
MemNet	Memory Persistent Network
MGHCNet	Multi-scale Granular and Uni-scale Holistic Channel Network

MorphoNet	Super Resolution Network using Morphological Residual Blocks
MuRNet	Multi-scale and Resolution-level Recursive Network
PHMNet	Parallel and Hierarchical Multi-scale Network
$PWConv$	Point-wise Convolutional Layer
QF	Quality Factor
RF	Receptive Field
ReLU	Rectified Linear Unit
SD	Space-to-Depth Operation
SE	Squeeze-and-Excitation Operation
SRCNN	Super Resolution Convolutional Neural Network
SRFBN	Super Resolution Feedback Network
SRNMFRB	Super Resolution Network using Multi-receptive Field Residual Blocks
SRNMSM	Super Resolution Network using Multi-scale Spatial and Morphological Residual Blocks
SRNSSI	Super Resolution Network using Spatial and Spectral Information
$SPConv$	Sub-pixel Convolutional Layer
TPCNN	Three-Prior Convolutional Neural Network
UpDCNN	Image Upsampling and Deblurring Convolutional Neural Network
UpDResNN	Image Upsampling and Deblurring Residual Neural Network

Chapter 1

Introduction

1.1 Importance of Image Restoration

Image restoration is an important task in image processing and computer vision and tries to enhance the quality of images degraded by various processes. This task can be categorized into many ill-posed problems such as image super resolution, image deblurring, image denoising, JPEG image deblocking and image demosaicing. Due to the physics of the image acquisition systems, the image restoration modules must be employed at the initial stage in almost all situations, where the images should be processed. To delineate more, the image signal processing of imaging systems results in generating raw images with blur (due to the point spread function of cameras and/or motion), noise (due to the imaging sensors) and compression artifacts and a size that could be smaller than that of the original continuous-space scenes (due to the sampling process carried out in A/D converters). Hence, the quality of these raw images must be improved right after they are acquired.

Image restoration has a wide-range of applications from medical imaging systems such as CT and MR systems to intelligent vehicles and transportation systems. For instance, it is well-known that due to the limited imaging time and dose considered to construct a medical image, its quality is not as good as physicians desire to have. As another example, in the intelligent vehicles, even though other image acquisition devices, such as LiDAR (light detection and ranging) and radar sensors, have recently emerged, camera perhaps is still the most useful imaging modality used by these vehicles. Cameras in Tesla models have been used in their auto-pilot self-driving systems, since they can provide a 360 degree view of the surroundings. However, the quality of the images acquired by cameras are not as satisfactory as the self-driving visual recognition systems require.

From the above paragraphs, it can be concluded that image restoration is a necessary and crucial task in many technologies that use image signals for their proper functioning.

1.2 Image Degradation Models

1.2.1 Image Blurring and Downsampling

The degradation in image super resolution is a decimation process that includes blurring and downsampling. If $x[m,n]$ ($0 \leq m \leq aM$ and $0 \leq n \leq aN$) and $y[m,n]$ ($0 \leq m \leq M$ and $0 \leq n \leq N$) are the high and low resolution images, respectively, the decimation process can be modeled as

$$\begin{aligned} p[m,n] &= x[m,n] * h[m,n] \\ y[m,n] &= p[am,an] \end{aligned} \tag{1.1}$$

where $h[m,n]$ is the blurring kernel that is assumed to be a bicubic kernel, a is scaling factor, $*$ represents the convolution operation and $p[m,n]$ is the blurred signal. The bicubic kernel is a separable two dimensional function that can be formulated as

$$\begin{aligned} h(r,s) &= h(r)h(s) \\ h(r) &= \begin{cases} sinc(r)sinc(\frac{r}{a}) & -a < r < +a \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{1.2}$$

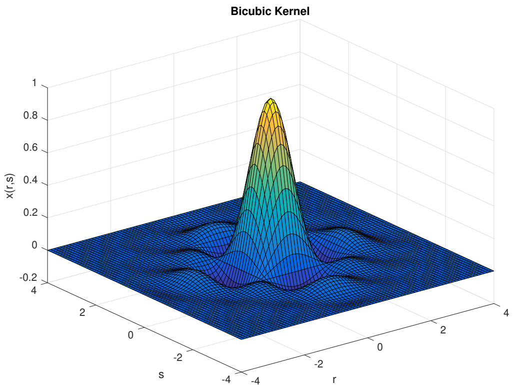


Figure 1.1: Bicubic kernel for a=4 in relation (1.2).

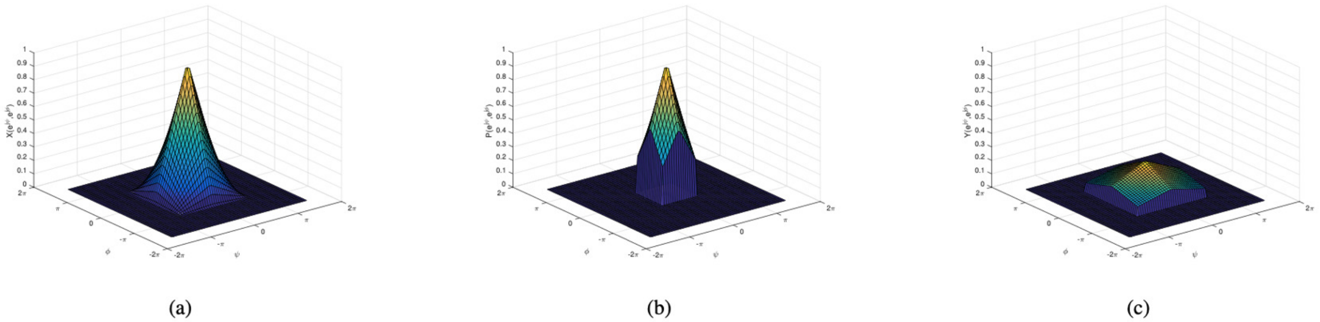


Figure 1.2: The decimation process in frequency domain. (a) original spectrum. (b) smoothing (blurring). (c) downsampling (one period).

This kernel is illustrated in Fig. 1.1 for a=4. The Fourier transform of this bicubic kernel is a rectangular pulse and when this filter is applied to an image, it will simulate blurring. The frequency domain representation of (7.1) is given by

$$\begin{aligned} P(e^{j\varphi}, e^{j\psi}) &= X(e^{j\varphi}, e^{j\psi})H(e^{j\varphi}, e^{j\psi}) \\ Y(e^{j\varphi}, e^{j\psi}) &= \frac{1}{a^2}\sum_{k=0}^{a-1}\sum_{l=0}^{a-1} P(e^{j(\frac{\varphi-2k\pi}{a})}, e^{j(\frac{\psi-2l\pi}{a})}) \end{aligned} \tag{1.3}$$

where φ and ψ are the spatial frequencies. Fig. 1.2 illustrates an example of the decimation process. Fig. 1.2 (a) shows the frequency spectrum of a two-dimensional signal (image),

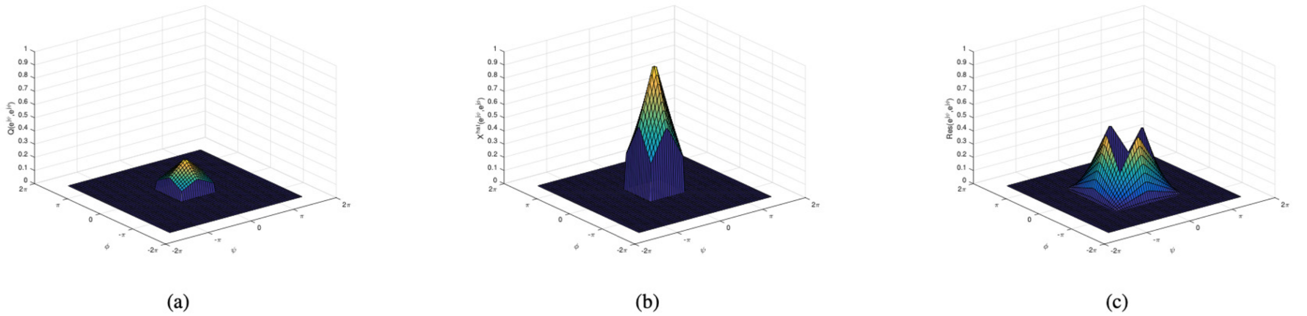


Figure 1.3: The interpolation process in frequency domain. (a) upsampling. (b) smoothing. (c) residual (one period).

$x[m,n]$. Fig. 1.2 (b) depicts the frequency response corresponding to the lowpass filtered image, $P(e^{j\varphi},e^{j\psi})$. It is noted from the spectrum of the blurred image, $p[m,n]$, that some of the high frequency contents of $x[m,n]$ are lost. Fig. 1.2 (c) shows the spectrum $Y(e^{j\varphi},e^{j\psi})$ of the downsampled image $y[m,n]$. We observe from this figure that the dynamic range of the decimated image has been compressed. Further, due to the periodic extension of the spectrum, shown in Fig. 1. 2(c) to the high frequency range, the decimated image $y[m,n]$ would have ringing effects around its edges.

If $\tilde{x}[m,n]$ is the interpolated low resolution image, then the relation between it and the decimated image $y[m,n]$ can be modeled as

$$q[m,n] = \begin{cases} y[\frac{m}{a},\frac{n}{a}] & m,n = 0,\pm a,\pm 2a,... \\ 0 & \text{otherwise} \end{cases} \tag{1.4}$$

$$\tilde{x}[m,n] = q[m,n] * a^2 h[m,n]$$

where $q[m, n]$ is the upsampled image. The relation is given by (7.2) in the frequency domain can be expressed as

$$\begin{aligned} Q(e^{j\varphi}, e^{j\psi}) &= Y(e^{ja\varphi}, e^{ja\psi}) \\ \tilde{X}(e^{j\varphi}, e^{j\psi}) &= Q(e^{j\varphi}, e^{j\psi})a^2H(e^{j\varphi}, e^{j\psi}) \end{aligned} \tag{1.5}$$

Fig. 1.3 presents results of the interpolation process in the frequency domain of the decimated image corresponding to the one shown in Fig. 1.2 (c). The spectrum of the upsampled image is shown in Fig. 1.3 (a). From this figure it is seen that the baseband of the spectrum of the upscaled image $q[m, n]$ has been compressed. Finally, the lowpass filter $h[m, n]$ is applied to recover the interpolated version of the decimated signal. Fig 3 (b) shows the spectrum of the interpolated image corresponding to the original image with spectrum shown in Fig. 1.2 (a). A comparison of Fig. 1.2 (a) and Fig. 1.3 (b) shows that most of the high frequency components are gone; however, due to the aliasing effect, the interpolated image $\tilde{x}[m, n]$ will still have some blurriness as well as ringing effect. Fig. 3 (c) shows the difference between the spectrum of the original image and that of the final interpolated image. For the spectrum of the interpolated image to be a replica of the original image, the difference must ideally have a zero value at all frequencies. The fact that the spectrum of Fig. 1.3 (c) has some significant non-zero components implies that the original and interpolated images do not have a perfect correlation.

1.2.2 JPEG Compression Image Blocking Model

Image compression compacts the useful information in an image. JPEG is one of the classical schemes for image compression that is commonly employed in real-world problems. Since JPEG is a block based compression scheme, the restored images using JPEG compressed images suffer from blocking artifacts.

JPEG compression scheme is based on block transformation. First, the image is divided into blocks of size 8×8 and then each block is transformed into the DCT domain as

$$\begin{aligned} X^c[k,l] = 4\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n]\cos(\frac{k\pi(2m+1)}{2M}) \\ \cos(\frac{k\pi(2n+1)}{2N}) \\ 0 \leq k \leq M-1, 0 \leq l \leq N-1 \end{aligned} \tag{1.6}$$

where $X^c[k,l]$ is the DCT transform of the block (two-dimensional signal) $x[m,n]$. Next, the DCT coefficients of each block are quantized and inputted to an entropy coder such as Huffman coding. For the reconstruction process, the coded coefficients are fed into an entropy decoder and then inputted to a dequantizer. Finally the inverse DCT (IDCT) is applied on the dequantized values and the block is reconstructed. As seen from this procedure, the blocking effect is unavoidable in JPEG image decompression.

1.3 Importance of Deep Learning-based Image Restoration

Traditionally, the problems of image restoration that are ill-posed and non-convex were solved by signal processing approaches. There exist many schemes for image restoration that use sparse representation [9], semi-local interpolation [10], fractional transforms [11] and finite rate of innovations [12]. However, all these hand-crafted image processing schemes suffer from using approximations such as convexity, that prohibit from having an optimal solution. In view of the emergence of deep learning and artificial intelligence techniques, an automatic feature extraction tool based on the original and the degraded data has been created, which directly use data itself as the prior information in order to learn a non-linear mapping in an end-to-end manner.

A neural network by stacking more layers, each followed by a nonlinear activation function, creates a deep network, which increases the learning capability of the network and consequently improves its performance. In view of this and the availability of adequate computational resources, deep learning techniques have become very attractive in computer vision and various other fields. Convolutional neural nets, which are very simple to implement, form a specific category in deep neural networks that have been demonstrated to provide very promising results. In fact, deep convolutional neural nets try to extract the most important features to minimize the loss between the estimated signal and the ground truth, and therefore, provide exceptional performance.

1.4 Brief Literature Review

Most of the deep image restoration techniques use a cascade of convolutional layers and ReLU (Rectified Linear Unit) activations in conjunction with short and long skip connections in order to form a sophisticated signal processing tool for reconstructing a high quality image. For example, in the task of image super resolution, the scheme of [28], referred to as EDSR (Enhanced Deep Super Resolution Network), employs a cascade of so called basic residual blocks, each consisting of two convolutional layers with a ReLU activation in-between and a skip connection, in order to map a degraded low-resolution image to the ground truth image. It should be pointed out that many of these deep learning based schemes for image restoration employ large numbers of parameters and multiply accumulate (MACC) operations in order to provide very high quality images [28], [30], [31], [32]. Considering the limited storage and power consumption of many industrial applications and technologies, the use of these heavy-weight schemes is impractical. On the other hand, there exist many deep learning based schemes that employ small numbers of parameters and MACC operations, and hence, they can be deployed to many real-life applications [14], [16], [68],[4]. For example, the network CARN [14], which uses a cascade of so called enhanced residual blocks, each employing two group-wise convolutions followed by

ReLU activations, one point-wise convolution and a skip connection, is such light-weight network for the task of image super resolution.

1.5 Motivation and Objectives

It is seen from the previous section that the design of light-weight image restoration convolutional neural networks is very crucial in many real-life computer vision applications such as medical imaging, intelligent transportation systems and space explorations. However, the performance of such networks is limited in view of the constraint of using small number of parameters. The quality of the images is very much affected by the various degradation processes associated with the image capturing devices, such as CCD cameras and sensors. Since the performance of any convolutional neural network is very much dependent on the representational capability of features generated by it, the objective of thesis is to develop different light-weight image restoration networks that are capable of producing rich sets of feature maps by various kinds of prior information of the degradation operations of the image capturing devices. In this thesis, we focus on three specific degradation operations, namely, bicubic downsampling, Gaussian blurring coupled with downsampling and JPEG compression blocking , and incorporate various kinds of prior information associated with these three degradation models in the design of the deep light-weight image restoration networks to produce more representable sets of feature maps. Regarding the first degradation model, i.e., bicubic donwsampling, various kinds of deep networks are developed each focusing on a specific prior information, namely, multi-scale feature generation, guided feature generation and efficient feature fusion. Regarding the second degradation model, i.e., Gaussian blurring coupled with downsampling, two different networks using the prior information on this degradation model in the context of global residual learning, are developed. As for the third degradation model, i.e., JPEG compression blocking, two networks, each using a different prior information, are designed.

1.6 Organization of the Thesis

The organization of this thesis is as follows. In Chapter 2, we review the image restoration schemes in the context of super resolution. In Chapters 3, 4, 5 and 6, we develop several image super resolution networks to enhance the quality of images degraded by the bicubic downsampling operation of the image capturing devices, using the various prior information about this operation. Specifically, in Chapter 3, we develop convolutional networks that use multi-scale feature generation as the prior information for super resolving the low-quality images. In Chapter 4, we use the idea of feature generation guided by the information drawn by the application of edge operators, spectral transformation and morphological operators on images, for improving their quality. In Chapter 5, we develop some convolutional neural networks based on the idea of fusion of feature sets produced at different hierarchical points of the deep network. In Chapter 6, we design an ultralight-weight high-performance convolutional neural network for the task of image super resolution by using the bidirectional mapping and sparsity priors. In Chapters 7 and 8, deep image restoration networks are developed based on the Gaussian blurring coupled with downsampling, and JPEG compression blocking degradation models, respectively. Finally, in Chapter 9, some concluding remarks are drawn on the work undertaken in this thesis along with some suggestions for future investigations.

Chapter 2

Background Materials

In this chapter, we review the state-of-the-art deep learning-based schemes for image restoration.

2.1 Light-weight Super Resolution Networks

The super resolution convolutional neural network, SRCNN, [1] is the first attempt for super resolving low resolution images using a convolutional neural network. SRCNN uses a three-layer fully convolutional network in order to map a bicubic interpolated version of the low resolution image to the corresponding ground truth image. Since in this network the spatial resolution of the low resolution image is increased to that of the ground truth before feeding it to SRCNN, the nonlinear mapping is carried out in the high resolution space, thus making the number of operations carried out by the network to be large. In order to address this problem, the network FSRCNN of [13] is designed to carry out a nonlinear mapping between the low and high resolution images by first directly applying the low resolution image to a cascade of small number of convolutional layers and then increasing the spatial resolution of the resulting maps to that of the ground truth image by employing a deconvolutional (convolutional transpose) layer. These two networks, however, are not sufficiently deep for providing the desired performance.

In [14], the authors have developed a super resolution network CARN by using a small number of residual blocks that are densely connected. CARN is a deep network that is able to provide a very good performance using only a small number of parameters. The authors have succeeded in achieving this by paying special attention to the design of the residual block of the network. In their residual block, two group convolution operations followed by a point-wise convolution operation are performed. In the group convolution, the numbers of parameters and operations are kept low by performing the convolution only on groups of channels rather than on the entire set of the channels.

The super light-weight super resolution network s-LWSR of [15] uses a cascade of so called inverse residual blocks in a U-Net architecture. Each inverse residual block consists of two point-wise convolution operations with a depth-wise convolution operation in between them. The use of the inverse residual block in s-LWSR results in reducing the number of parameters of the network at the expense of degrading its performance. However, in order to address this problem, some performance enhancement techniques, such as removal of nonlinear activations from some of the convolutional layers, have been used in this network.

Inspired by the lattice structure of filter banks in signal processing, the authors of [16] have proposed a novel residual block, called the lattice block, for the task of image super resolution. In the lattice block, two sets of convolutional layers are interconnected in a lattice structure. Then, the features generated by these two sets of the convolutional layers are fused in order to form the output of the lattice block. The super resolution network of LatticeNet [16] employs a cascade of 4 lattice blocks for mapping the input low resolution image to the ground truth image. LatticeNet is the best performing state-of-the-art light-weight network that presently exists in the literature of single image super resolution.

2.2 Recursive Super Resolution Networks

In an effort to make the network deep without increasing the number of parameters, recursive neural networks have been developed for the problem of single image super resolution. The network DRCN [19] employs a single convolutional layer and uses it recursively. The network DRRN [20] consists of a single residual block involving two convolutional layers and uses the block recursively. The network MemNet [4] is designed as a cascade of small number of recursive residual blocks that are densely connected. The network SRFBN [25] is another example of a deep recursive network. In this network, in each recursion after the first one, the feature maps of the low resolution image are concatenated with that of the output of the block by using a feedback connection. All these recursive networks are considered to be light-weight in view of their employing a small number of parameters and the depth of the networks is virtually increased through the recursive use of the blocks. However, this recursive use of the parameter sets results in increasing the number of operations significantly.

2.3 Gradient-based Super Resolution Networks

There are a couple of deep super resolution convolutional neural networks [17], [18] that extract the gradients of the feature maps generated by the network and use these gradients for guiding the process of constructing the super resolved images. The network DEGREE [17], in addition to employing the usual loss between the ground truth and estimated high resolution image, also uses the loss between the edges of these two images in order to train the network. The network SPSR in [18] has an architecture having a structure of two parallel branches, a super resolution branch and a gradient branch, and produces rich set of features by generating and fusing the spatial and gradient information about the low resolution image as well as those of the maps produced at various hierarchical levels. In this architecture, the network parameters are optimized in an adversarial training framework.

2.4 Wavelet-based Multi-domain Convolutional Super Resolution Networks

There are several studies in many computer vision tasks in which the idea of multi-domain feature representation using wavelet transform for convolutional neural networks is used. Among these, the works carried out in [106] and [27] use the idea of combining the wavelet transform with the convolutional neural networks for the task of single image super resolution. It has been shown in [106] that by applying the wavelet transform to the low resolution input and the ground truth images, their manifolds become topologically simpler. Hence, the use of the wavelet maps of the low resolution and the ground truth images for constructing the training set of a deep super resolution network facilitates its training process. This network employs a cascade of 5 residual blocks in order to map the wavelet maps of the low resolution image to those of the ground truth image. In [27], a novel U-Net architecture, referred to as the multi-level wavelet convolutional network (MWCN), has been proposed for the task of image super resolution. In this network, pooling and unpooling operations of the conventional U-Net network are replaced, respectively, by the wavelet pooling and wavelet unpooling operations. Unlike the conventional pooling and unpooling operations, the wavelet-based pooling and unpooling operations are inverses of each other, in that the two operations when applied sequentially restore the original feature maps. Consequently, the use of such pooling and unpooling operations enables MWCN to provide a high super resolution performance. However, since the network of MWCN employs convolutional layers with large number of filters, its complexity in terms of the numbers of parameters and operations is very large.

2.5 Deep Heavy-weight Super Resolution Networks

There are a number of networks existing in the literature for single image super resolution that provide very high performance [28], [30], [32]. The network EDSR [28] is the first very deep super resolution network that employs a cascade of 32 residual blocks, each consisting of two convolutional layers with a ReLU activation function in between to provide images of very high quality. The network RCAN [30] is the deepest super resolution network existing in the literature that uses a cascade of 200 residual channel-attention blocks, each consisting of two convolutional layers and one squeeze-and-excitation unit [33]. Its performance is superior to that of EDSR [28] with the number of parameters exceeding 15M. The network SAN [31] employs a cascade of residual blocks, each using a second order channel-attention unit, and provides a performance that is superior to that of RCAN [30] with a slightly lower number of parameters. In [32], the authors have proposed a deep heavy-weight convolutional neural network, referred to as DBPN, that uses a cascade of up-projection and down-projection units based on the idea of back projection introduced in [34]. In each of these units, the projection error maps are first obtained by applying upsampling and downsampling layers to the feature maps input to the unit, and then these maps are used to obtain the residual feature maps of the unit. In [35], a deep heavy-weight super resolution network, referred to as DRN, has been proposed that uses an additional loss function between the degraded version of the estimated high resolution image and the original low resolution image for implementing the idea of back projection [34]. This network uses a cascade of large number of residual channel attention blocks for super resolving the degraded low resolution images. The quality of the images super resolved by these networks is very high in the sense that it is very similar to that of the ground truth images. However, the complexity of these networks in terms of the numbers of layers, parameters and arithmetic operations is extremely high and these networks are considered to be very deep heavy-weight networks. As such, the training of these networks is not easy, and more importantly, the applications of such networks are limited.

2.6 Deep Image Upsampling and Deblurring Networks

The network of [36] is a deblurring network that employs a multi-scale convolutional neural network in order to map a blurred input image to its ground truth image. In this network, the blurred image is first passed through a downsampling Gaussian pyramid. Next, the input image and each of its downsampled versions are fed individually to a cascade of basic residual blocks in order to obtain deblurred images in various scales. In this technique, the reason behind deblurring the images at various scales is to refine the deblurring result at each scale by using the deblurring result from the previous scale. These successive refinements result in eventually obtaining a good deblurred image at the original scale.

There are only a couple of schemes, in which the tasks of image upsampling and deblurring have been carried out jointly by using neural networks [37], [38]. In [37], a network, called gated fusion network (GFN), has been proposed for the task of image upsampling and deblurring by decomposing the feature extraction step into the streams of deblurring and super resolution. The network training is carried out using two loss functions, each corresponding to one of the two tasks. The deblurred and super resolution features obtained from the two streams are fused using a gated mechanism. The features resulting from the gate module are then used for obtaining the deblured high resolution image.

The other work for the joint task of image upsampling and deblurring is the one appearing in [38]. In this scheme, a network referred to as ASDN, has been developed. The network is designed to carry out the deblurring task on the blurred low resolution image followed by a the super resolution task on the resulting deblurred image. The network is trained using a dual supervised learning mechanism exploiting the dependencies between the low resolution and high resolution images.

2.7 Conclusion

It is seen from the literature review carried out in this chapter that although deep heavy-weight image restoration networks are able to provide very high performances, their applications in many real-life situations are limited. On the other hand, the deep light-weight image restoration networks could provide acceptable performances by employing small numbers of parameters and operations. This makes the use of the deep light-weight image restoration networks attractive in many real-life applications.

Chapter 3

Deep Image Super Resolution Networks using Multi-scale Feature Generation

3.1 Introduction

The convolutional neural networks provide a very affective framework for constructing high resolution images, in view of their capability of extracting features at different levels and scales and providing a nonlinear model for the super resolution problem, which is inherently a nonlinear mapping problem. Since different parts of a single image appear in different scales, a deep convolutional network with a superior capability of generating multi-scale features would be more desirable for the super resolution problem. In this chapter, we develop several deep image super resolution networks that use the idea of multi-scale feature generation in order to provide a high performance [89], [93], [95], [97], [98]. Different multi-scale feature generation tools in deep neural networks, such as inverse sub-pixel convolutional layers, granular multi-scale convolutional layers and dilated convolutional layers, are used for developing various deep multi-scale neural networks for image super resolution.

3.2 PHMNet: A Deep Super Resolution Network using Parallel and Hierarchical Multi-scale Residual Blocks

In this section, a novel light-weight deep image super resolution network [95], which generates features at various scales, is presented by proposing a residual block that utilizes two multi-scale feature generation modules, namely, a parallel multi-scale feature fusion module and a hierarchical feature fusion module.

The overall architecture of the proposed super resolution network is shown in Fig. 3.1. In the proposed super resolution scheme, the original low resolution image is first passed though a convolution operation followed by a ReLU activation resulting in the extraction of low resolution feature maps. This convolution operation uses 64 filters each of size of 3×3. The low resolution feature maps are fed as the input to a sequence of 11 units of the proposed residual block in order to generate multi-scale feature maps. The feature maps resulting from the last residual block is upsampled to the resolution of the ground truth image by a sub-pixel convolution operation [8] before reconstructing a high resolution image by a last convolution operation. The sub-pixel convolution operation is performed by employing 64 filters each of size of 3×3, whereas, the reconstruction convolution is carried out by using 3 filters each of support size of 3×3.

The proposed residual bock, depicted in Fig. 3.2, consists of three modules, a parallel multi-scale feature fusion module, a hierarchical feature fusion module and a reconstruction module.

The first module performs two convolution operations in parallel on the feature maps u input to the block. The first convolution in this module carries out normal 3×3 convolution operations using 64 filters and results in feature maps a, whereas the second convolution carries out dilated 3×3 convolution operations with a dilation rate of 2 using 16 filters and produces the feature maps b. The dilated convolution increases the receptive field without

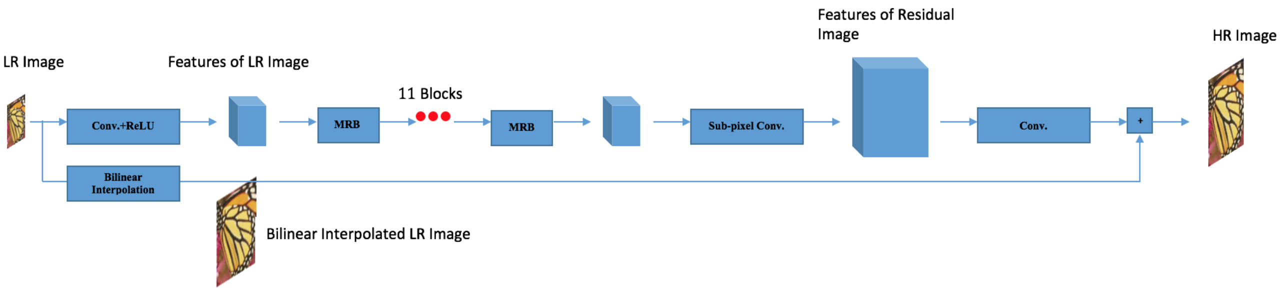


Figure 3.1: Architecture of the proposed super resolution network.

adding to the complexity of the operations, but at the expense of introducing blind spots within the receptive fields. The feature maps a and b are concatenated channel-wise to produce the feature maps d. The basic idea in constructing this module is that the feature maps d produced by it are multi-sale feature maps, in which the disadvantage of creating the blind spots in b through the dilated convolution is compensated by the concatenation of b with a, which is free of such blind spots.

The second module performs a cascade of two sets of convolution operations on the feature maps d. Both these set of convolutions carry out normal 3×3 convolution operations using 64 filters and result in, respectively, feature maps e and f. The feature maps e and f are concatenated channel-wise to produce the feature maps g. The main function of the second module of the residual block is to produce a set of features that is a combination of feature maps extracted at two different hierarchical levels of abstractions.

Finally, in the reconstruction module, the number of channels of the feature tensor g (this number of channels is 128) is reduced to that of the feature tensor u input to the block by performing point-wise convolution operations on g using 64 filters. Lastly, the resulting feature tensor w is added to the feature tensor u in order to construct the residual block's output feature maps v.

It is to be noted that the first module produces a set of feature maps that is extracted explicitly at two different scales. On the other hand, the second module produces a set

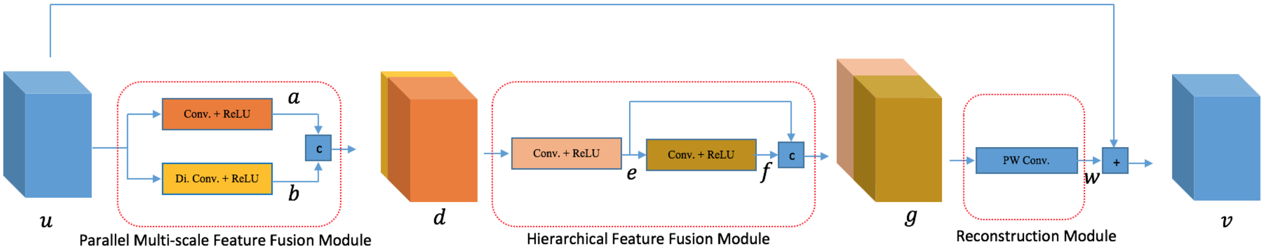


Figure 3.2: Architecture of the proposed residual block. Di. Conv. and PW Conv. represent the dilated convolution operation and point-wise convolution operation, respectively.

of feature maps that is still a combination of features at two different scales, however, extracted indirectly at two different hierarchical levels of abstraction. In the final analysis, the proposed residual block can be regarded to produce multi-scale feature maps, where the feature extraction at different scales are carried out using different strategies.

The proposed super resolution network shown in Fig. 3.1 is referred to as Super Resolution Network with **P**arallel and **H**ierarchical **M**ulti-scale residual blocks (PHMNet) [95].

To train the proposed super resolution network, the sub-images of size 48×48 are extracted from the 800 training images of the DIV2K dataset [42]. The $\ell 1$ norm loss between the ground truth and estimated high resolution images is used to update the weights of the network. The weights of the network are optimized using the stochastic gradient descent optimizer, in which the learning rate is initialized by 0.1 and is decreased by a factor of 10 after each 182500 iterations. The weights of the network are initialized by the method proposed in [7].

The proposed super resolution network is implemented using Keras library [40] and TensorFlow package [41]. PHMNet is trained using a machine with Intel Core i7 CPU @4.2 GHz, 16-GB RAM and Nvidia Titan X GPU.

3.3 MGHCNet: A Deep Multi-scale Granular and Holistic Channel Feature Generation Network for Image Super Resolution

Generation and use of multi-scale features is a significant attribute of a network in enhancing its performance for image super resolution. In this respect, Res2Net [43] has made an important contribution in that it is capable of generating multi-scale features by splitting the input to a Res2Net block at the granular level, which keeps the network complexity low. However, in doing so, the scheme of Res2Net deprives itself from the set of features that could be generated by using directly all the channels of the tensor input to the block.

This section proposes a residual block that aims at overcoming the limitation of Res2Net in that it is capable of generating a richer set of residual features that includes the types of features that are directly extracted from all channels of a tensor input to the block simultaneously, while retaining the characteristics of the granular level multi-scale features generation of Res2Net. The proposed residual block consists of the following three modules:

- *Multi-scale Granular Channel Feature Generation Module:* Following the scheme of Res2Net, this module generates multi-scale feature maps at the granular level of channels.

- *Uni-scale Holistic Channel Feature Generation Module:* This module generates holistic channel features at a single scale.

- *Concatenative Feature Fusion Module:* This module fuses the uni and multi-scale features generated by the first two modules in order to produce the residual features of the block.

In the proposed Multi-scale Granular and Holistic Channel Feature Generation Network (MGHCNet), the original low resolution image is passed through a convolution operation followed by a ReLU activation in order to yield the feature maps of the low resolution image. This convolution operation uses 64 filters with spatial support of 3×3. The low resolution feature maps thus obtained go through a cascade of 16 units of the proposed residual block, whose architecture is described in the following paragraphs, yielding a very rich set of high frequency feature maps. These feature maps then undergo a sub-pixel convolution operation in order to increase their spatial resolution to that of the ground truth image. This sub-pixel convolution operation uses 64 filters with spatial support of 3×3. The feature maps with the increased spatial resolution are passed through a convolution operation to construct the residual signal between the ground truth and the bilinear interpolated version of the low resolution image. This convolution operation uses 3 filters with spatial support of 3×3.

Fig. 3.3 shows the architecture of the proposed residual block for the network presented in the previous paragraph. This residual block consists of three main modules, namely, multi-scale granular channel feature generation module, uni-scale holistic channel feature generation module and feature fusion module. The feature maps **y** input to the block are first passed through a convolution operation followed by a ReLU activation yielding the feature maps **u** given by

$$\mathbf{u} = ReLU\big(W_1(\mathbf{y})\big) \tag{3.1}$$

where the convolution operation W_1 employs 64 filters each with spatial support of 3×3. Then, the feature maps **u** are simultaneously passed through the multi-scale granular channel feature generation module and the uni-scale holistic channel feature generation module. In the multi-scale granular channel feature generation module, the feature maps **u** are split along the channel dimension into four subsets of feature maps, $\mathbf{u}_1$, $\mathbf{u}_2$, $\mathbf{u}_3$ and $\mathbf{u}_4$, each having 16 channels. The feature maps $\mathbf{u}_1$ are kept unaltered. The feature maps $\mathbf{u}_2$ are made to undergo a convolution operation to yield the feature maps $\mathbf{v}_1$ as

$$\mathbf{v}_1 = W_2(\mathbf{u}_2) \tag{3.2}$$

where the convolution operation W_2 uses 16 filters each with kernel size 3×3. The feature maps $\mathbf{v}_1$ are then concatenated with the feature maps $\mathbf{u}_3$ to produce the feature maps $\mathbf{a}$, which in turn, are passed through a convolution operation in order to obtain the feature maps $\mathbf{v}_2$. These operations can be expressed as

$$\begin{aligned} \mathbf{a} &= Conc(\mathbf{v}_1, \mathbf{u}_3) \\ \mathbf{v}_2 &= W_3(\mathbf{a}) \end{aligned} \tag{3.3}$$

where $Conc$ represents the concatenation operation and the convolution operation represented by W_3 uses 16 filters each with spatial support of 3×3. Next, the feature maps $\mathbf{v}_2$ are concatenated with the feature maps $\mathbf{u}_4$ to produce feature maps $\mathbf{b}$, which undergo a convolution operation to yield the feature maps $\mathbf{v}_3$ as

$$\begin{aligned} \mathbf{b} &= Conc(\mathbf{v}_2, \mathbf{u}_4) \\ \mathbf{v}_3 &= W_4(\mathbf{b}) \end{aligned} \tag{3.4}$$

where the convolution operation W_4 uses 16 filters each with spatial support of 3×3. Then, the feature maps $\mathbf{u}_1$, $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$ are concatenated to yield the set of feature maps $\mathbf{c}$. The feature maps $\mathbf{c}$ are passed through a point-wise convolution operation followed by a ReLU activation yielding the output feature maps $\mathbf{p}$ of the multi-scale granular channel feature generation module. These operations can be expressed as

$$\begin{aligned} \mathbf{c} &= Conc(\mathbf{u}_1, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3) \\ \mathbf{p} &= ReLU\big(W_5(\mathbf{c})\big) \end{aligned} \tag{3.5}$$

where the point-wise convolution operation W_5 uses 64 filters each with spatial support of 1×1.

In the uni-scale holistic channel feature generation module, all the channels of the feature maps **u** are passed through a convolution operation followed by a ReLU activation yielding the feature maps **q** as

$$\mathbf{q} = ReLU\big(W_6(\mathbf{u})\big) \tag{3.6}$$

where the convolution operation W_6 uses 64 filters each with spatial support of 3×3.

Note that the granular channel feature subsets of **u**, namely, $\mathbf{u}_1$, $\mathbf{u}_2$, $\mathbf{u}_3$ and $\mathbf{u}_4$, experience varying amount of hierarchical depth in view of each going through a different number of convolution operations. Hence, each of the resulting feature maps $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$ has different receptive fields. As a result, the fusion of $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$ through the operations of concatenation and point-wise convolution produces a rich set of multi-scale granular feature maps **p**. On the other hand, in the second module, a single convolution operation is performed on all the channels **u** in their entirety. Therefore, the resulting features **q** are uni-scale holistic channel feature maps.

In the feature fusion module of the proposed residual block, feature maps **p** and **q** are fused by concatenating them and then performing a point-wise convolution operation on the concatenated set as

$$\begin{aligned} \mathbf{s} &= Conc(\mathbf{p}, \mathbf{q}) \\ \mathbf{r} &= W_7(\mathbf{s}) \end{aligned} \tag{3.7}$$

where the point-wise convolution operation W_7 uses 64 filters each with spatial support of 1×1. Since the set of feature maps **p** is a combination of multi-scale granular channel features and the set of feature maps **q** is a uni-scale holistic channel features, the feature maps **r** generated by (3.7) can be expected to be a very rich residue. Finally the residual feature maps **r** are added to the block's input feature maps **y** to obtain its output feature

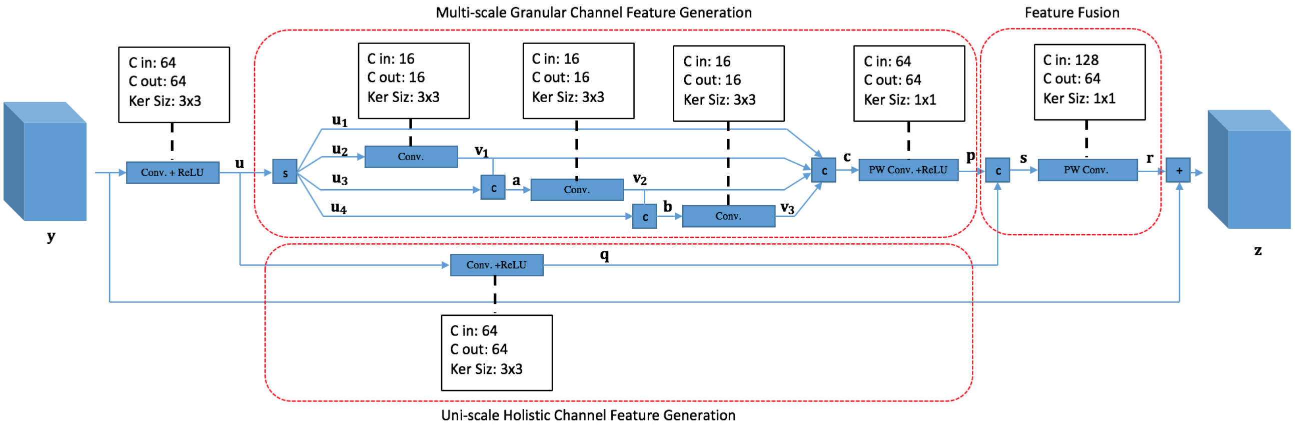


Figure 3.3: Architecture of the proposed residual block. Conv., PW Conv., **s** and **c** represent, respectively, convolution, point-wise convolution, split and concatenation operations.

maps as

$$\mathbf{z} = \mathbf{r} + \mathbf{y} \tag{3.8}$$

The sub-images of size 48×48 are obtained from the 800 training images of the DIV2K dataset [42] in order to form the training set for the proposed super resolution network. The $\ell 1$ norm loss between the ground truth samples and estimated high resolution images is used to update the weights of the network in the backpropagation. The stochastic gradient descent method is used for optimizing the weights of the network. The step size of the stochastic gradient descent is initialized by the value of 0.1 and decreased by a factor of 10 after each 182500 iterations. The mini-batch size is set to 64. The weight decay parameter of the convolution operations is set to 10^{-4}.

3.4 SRNMFRB: A Deep Light-weight Super Resolution Network using Multi-receptive Field Feature Generation Residual Blocks

In this section, we propose a new low complexity residual block to be used in a super resolution network [98]. In the proposed residual block, we use the strategy of generating features in multiple receptive fields. The new residual block comprises three parallel branches as follows:

- In the first and the second branches, one and two convolution operations are, respectively, used with the filters of the same spatial support, namely, 3×3. Thus, the features in the two branches are produced, respectively, in receptive fields of 3 by 3 and 5 by 5.
- In the third branch, through convolution, space-to-depth (inverse pixel shuffle) and depth-to-space (pixel shuffle) operations, the features are produced in a receptive field of 9 by 9.

The features generated in 9 by 9 receptive field through the convolution, space-to-depth and depth-to-space operations in the third branch are different from those generated in 3 by 3 and 5 by 5 receptive fields in the first and second branches of the residual block, where these latter type of features are generated only through the convolution operations. Also, an additional advantage of generating features in the 9 by 9 receptive field by the third branch is to keep the number of operations of the block low in comparison to that using additional two convolution operations to acquire features in a 9 by 9 receptive field. By fusing the features produced by these branches, a rich set of feature maps generated in the different receptive fields for the task of image super resolution.

Fig. 3.4 shows the overall architecture of the proposed super resolution network. It

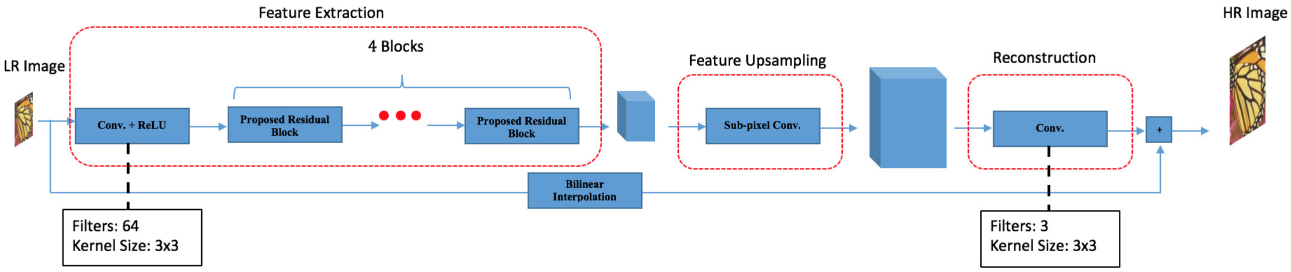


Figure 3.4: Overall architecture of the super resolution network.

is seen from this figure that the proposed super resolution network consists of three parts, namely, feature extraction, feature upsampling and reconstruction. In the feature extraction part, the features of the low resolution image are first extracted using a convolution operation followed by a ReLU (rectified linear unit) activation. This convolution operation uses 64 filters each with kernel size of 3×3. Next, the low resolution feature maps thus obtained are fed to a sequence of 4 units of the proposed residual block. Then, the output feature maps from the last residual block are input to the feature upsampling part of the network, which consists of a sub-pixel convolution operation [8] with 64 filters each with kernel size 3×3. Finally, the upscaled feature maps thus produced are made to undergo the reconstruction part, which consists of a convolution operation with 3 filters each of spatial support of 3×3, in order to reconstruct the residual signal between the target high resolution image and bilinear interpolated version of the low resolution image. Fig. 3.5 shows the architecture of the proposed residual block. First, the feature maps **x** input to the block are passed through a convolution operation followed by a ReLU activation in order to obtain the feature maps **u** as

$$\mathbf{u} = ReLU(W_1(\mathbf{x})) \tag{3.9}$$

where the convolution operation W_1 uses 64 filters with kernel size of 3×3. It is seen that the feature maps **u** are generated in a receptive field of 3 by 3.

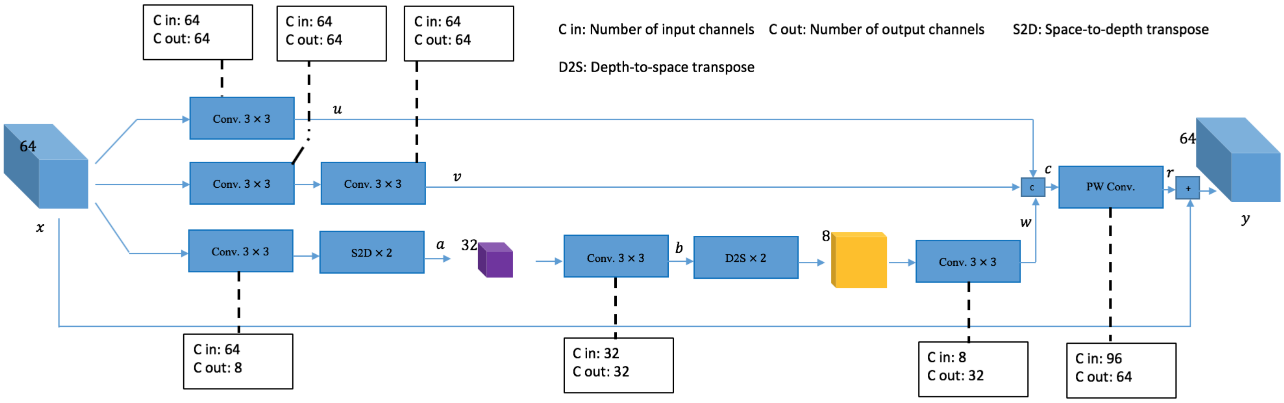


Figure 3.5: Architecture of the proposed residual block. Conv. and PW Conv. denote, respectively, convolution and point-wise convolution operations. All of the convolution operations (except the point-wise convolution operation) are followed by a ReLU activation function.

The feature maps **x** are also made to undergo a cascade of two convolution operations each followed by a ReLU activation yielding the feature maps **v** as

$$\mathbf{v} = ReLU(W_3(ReLU(W_2(\mathbf{x})))) \tag{3.10}$$

where each of the convolution operations W_2 and W_3 uses 64 filters with kernel size of 3×3. It is seen that the feature maps **v** are produced in a receptive field of 5 by 5.

The feature maps **x** are also passed through a convolution operation followed by a ReLU activation and a space-to-depth operation with a factor of 2 in order to generate feature maps **a** as

$$\mathbf{a} = SD(ReLU(W_4(\mathbf{x}))) \tag{3.11}$$

where the convolution operation W_4 uses 8 filters with kernel size of 3×3 and SD represents the space-to-depth operation. Then the feature maps **a** are fed into another convolution operation followed by a ReLU activation in order to yield the feature maps **b** as

$$\mathbf{b} = ReLU(W_5(\mathbf{a})) \tag{3.12}$$

where the convolution W_5 uses 32 filters with kernel size of 3×3. It is seen that the feature maps **b** are produced in a receptive field 7 by 7. Also since these feature maps are generated at a resolution smaller than that of the low resolution image, they are more robust to the spatial variation than the feature maps **u** and **v**.

Fusing feature maps **u**, **v** and **b**, that are obtained in different receptive fields, generates a rich set of features for image super resolution. In this regard, we first increase the resolution of **b** to that of **u** and **v** and then increase its number of channels by performing a depth-to-space operation with a factor of 2 followed by a convolution operation and a ReLU activation, yielding the feature maps **w** as

$$\mathbf{w} = ReLU(W_6(DS(\mathbf{b}))) \tag{3.13}$$

where the convolution operation W_6 uses 32 filters each with kernel size of 3×3 and DS represents the depth-to-space operation. It should be pointed out that the feature maps **w** are obtained in a receptive field 9 by 9. Finally, the feature maps **u**, **v** and **w** are fused using concatenation followed by a point-wise convolution operation given by

$$\begin{aligned} \mathbf{c} &= Concatenate(\mathbf{u}, \mathbf{v}, \mathbf{w}) \\ \mathbf{r} &= W_7(\mathbf{c}) \end{aligned} \tag{3.14}$$

where the point-wise convolution operation W_7 uses 64 filters each with spatial support of 1×1. Finally, the residual feature maps **r** are added to the feature maps **x** input to the block and the output feature maps **y** of the residual block are obtained as

$$\mathbf{y} = \mathbf{r} + \mathbf{x} \tag{3.15}$$

In view of generating residual feature maps in various receptive fields by the proposed residual block, we refer it to as *multi-receptive field feature generation residual block*. Also, we refer to the super resolution network using the proposed residual block to as **S**uper **R**esolution **N**etwork using **M**ulti-receptive **F**ield Feature Generation **R**esidual **B**lock (SRNMFRB) [98].

The 5840000 RGB sub-images of size 48×48 are obtained from the 800 training images of the DIV2K [42] dataset. The weights of the proposed super resolution network are optimized using the $\ell 1$ norm loss between the ground truth and estimated high resolution samples. The stochastic gradient descent optimizer is employed to optimize the loss function. The learning rate of the gradient descent optimizer is initialized with a value of 0.1 and decreased by a factor of 10 after each 182500 iterations. The mini-batch size is set to 64. The weights of the network are initialized by the method proposed in [7].

3.5 MISNet: Multi-resolution Level Feature Interpolating Ultralight-weight Residual Image Super Resolution Network

Many of the super-resolution convolutional neural networks learn the residue between the ground truth and a version of the degraded image interpolated to the same resolution as that of the ground truth image. Since this residual signal is sparse, it is more appropriate to be learned by a convolutional network. In these networks, generally a large number of convolutional layers are used to provide a good super-resolution performance. Also,

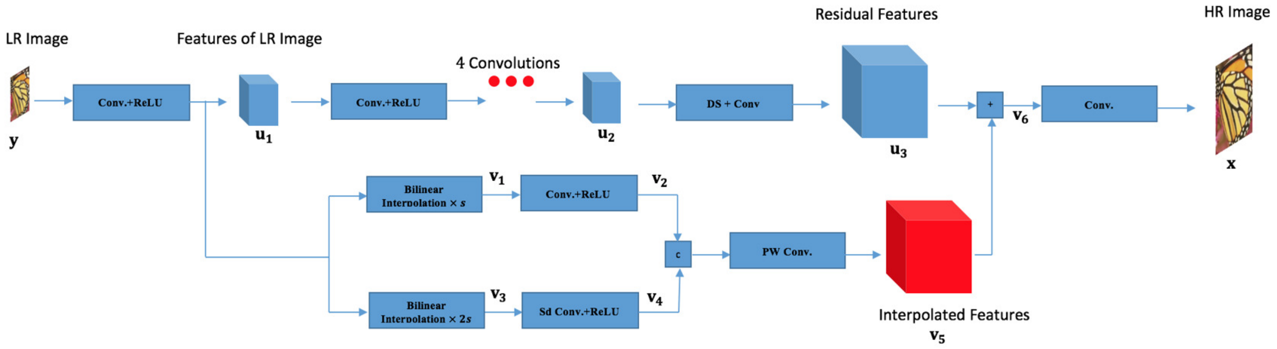


Figure 3.6: The architecture of the proposed ultralight-weight super-resolution network. Conv., PW Conv. and DS denote, respectively, convolution, point-wise convolution and depth-to-space transpose operations.

since the scales of the objects in generic images vary, adding fused interpolated low level features generated at multiple resolution levels to the residual features could improve the super-resolution performance. Therefore, in this section, by incorporating the idea of multi-resolution level interpolation of the low level features into a residual framework, we develop a novel architecture for the task of single image super-resolution [93]. Since in many computer vision applications, such as robotics, the visual quality of the super resolved images is very important, the proposed scheme by focusing on the multi-resolution features is specifically suited for such applications. It is also worth mentioning that since the architecture of the proposed multi-resolution level based feature interpolation is a light-weight architecture and used in a non-recursive manner, the resulting network has an ultralight-weight character.

Fig. 3.6 shows the architecture of the proposed super-resolution network. Let **y** denote the degraded low-resolution image that is input to the network. First, the features of the low-resolution input image **y** are obtained by performing a convolution operation followed by a ReLU activation as

$$\mathbf{u}_1 = F_1(\mathbf{y}) \tag{3.16}$$

where F_1 denotes the convolution operation, which is carried out by employing 32 filters each with kernel size 3×3. The low level feature tensor $\mathbf{u}_1$ is then passed through a cascade of four convolutional layers to obtain the high level feature tensor $\mathbf{u}_2$ as

$$\mathbf{u}_2 = F_2(\mathbf{u}_1) \tag{3.17}$$

where F_2 represents the combined cascade convolution operations of the four layers, in which each layer uses 32 filters each with kernel size 3×3 followed by a ReLU activation. The high level feature tensor $\mathbf{u}_2$ is then subjected to a depth-to-space transpose operation DS followed by a convolution operation F_3 yielding the upsampled feature tensor $\mathbf{u}_3$ given by

$$\mathbf{u}_3 = F_3(DS(\mathbf{u}_2)) \tag{3.18}$$

where the operation DS employs a scaling factor s equal to that of the super-resolution scaling factor, and the convolution operation F_3 is carried out using 32 filters each with kernel size 3×3.

We also obtain features of the low-level feature tensor $\mathbf{u}_1$ at multiple resolution levels. This is accomplished by the second branch of the architecture. Specifically, features are obtained involving two resolution levels, s and $2s$. The feature tensor $\mathbf{v}_1$ at the resolution level s is obtained by passing the low level feature tensor $\mathbf{u}_1$ through a bilinear interpolation operation with the factor s. Specifically, let $u[m, n]$ denote the two-dimensional signal representing one channel of the feature tensor $\mathbf{u}_1$. The two dimensional signal $v[m, n]$ representing the corresponding channel in the feature tensor $\mathbf{v}_1$ is obtained as

$$v[m, n] = z[m, n] * h[m, n] \tag{3.19}$$

where

$$
\begin{aligned}
z[m,n] &= \begin{cases} u[\frac{m}{s},\frac{n}{s}] & m,n = 0, \pm s, \pm 2s, ... \\ 0 & \text{otherwise} \end{cases} \\
h[m,n] &= \begin{cases} (1-\frac{|m|}{s})(1-\frac{|n|}{s}) & |m| \leq s, |n| \leq s \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
\tag{3.20}
$$

The feature tensor $\mathbf{v}_1$ is then passed through the operations of convolution F_4 and rectification to produce the interpolated low level feature tensor $\mathbf{v}_2$ given by

$$\mathbf{v}_2 = F_4(\mathbf{v}_1) \tag{3.21}$$

where the convolution operation F_4 employs 32 filters each with kernel size 3×3. Similarly, the feature tensor $\mathbf{v}_3$ at the resolution level $2s$ is obtained again by passing the low level feature tensor $\mathbf{u}_1$ through a bilinear interpolation operation with the factor $2s$. The feature tensor $\mathbf{v}_3$ thus obtained is made to undergo a strided convolution operation F_5 and rectification to produce another interpolated low level feature tensor $\mathbf{v}_4$ given by

$$\mathbf{v}_4 = F_5(\mathbf{v}_3) \tag{3.22}$$

where the convolution operation F_5 employs 32 filters each with kernel size 3×3 and a stride of 2. The feature tensors $\mathbf{v}_2$ and $\mathbf{v}_4$ are concatenated and the resulting feature tensor $\mathbf{v}_5$ is subjected to a point-wise convolution operation F_6 yielding the new interpolated low level feature tensor $\mathbf{v}_6$ given by

$$\mathbf{v}_6 = F_6(\mathbf{v}_5) \tag{3.23}$$

where the point-wise convolution F_6 uses 32 filters each with size 1×1. Note that the set of features represented by $\mathbf{v}_6$ has a special characteristic in view of the fact that it is produced by involving features generated at two resolution levels, namely, s and $2s$.

In order to make the parameters of the architecture to be learnable in a residual framework, the low level multi-resolution level feature tensor $\mathbf{v}_6$ is added to the high-level feature tensor $\mathbf{u}_3$ to yield the estimated high-quality feature maps denoted by $\mathbf{w}$. Finally, the estimated high-resolution image $\mathbf{x}$ is constructed by subjecting the feature tensor $\mathbf{w}$ to a convolution operation F_7 given by image $\mathbf{x}$ as

$$\mathbf{x} = F_7(\mathbf{w}) \tag{3.24}$$

where the convolution operation F_7 uses 3 filters each with kernel size 3×3. We refer to our proposed network as **M**ulti-resolution Level Feature **I**nterpolating Ultralight-weight Residual Image **S**uper Resolution Network (MISNet) [93].
The sub-images of size 48×48 are extracted from the 800 images of the DIV2K dataset [42] in order to form samples for the training of the proposed super-resolution network. The $\ell 1$ norm of the loss between the ground truth samples in a batch and corresponding estimated high-resolution samples is minimized in order to obtain the optimal values for the network parameters. The batch size is set as 64. The stochastic gradient descent optimizer with the initial learning rate of 0.1 is employed for minimizing the $\ell 1$ norm loss. The weight decay parameter of the convolutions is set as 10^{-4}.

3.6 MuRNet: A Deep Recursive Network for Super Resolution of Bicubically Interpolated Images

In many real-world applications, such as printing systems and cameras, the low resolution images are inherently interpolated to the desired resolution level using the bicubic interpolation operation. Although, the bicubic interpolation operation provides acceptable visual qualities for the smooth regions of an image, it produces artifacts in the high frequency regions of the image such as those containing edges. Many deep learning image super resolution schemes [28], [44], [30], [31], carry out a mapping between the original low resolution image and ground truth. However, this prevents their applicability to some of the real-world applications, in which the interpolated version of the low resolution images cannot be avoided, since such images result from image capturing devices such as printing machines.

The focus of the existing deep learning schemes for the image super resolution problem is to provide high performance. However, this is generally achieved at the expense of an increased number of parameters. One way of controlling the increase in the number of parameters is the use of a feature generating block in a recursive framework of the network, in which an adequate number of recursions involving such a block are carried out so that the network's representational capability becomes sufficiently high. Fusing different types of features could produce richer and more representable feature maps that could enhance the performance of a super resolution network. The main idea of the proposed scheme [89] is to design a block that could impart to the network a good representational capability when used in a recursive framework by generating a rich set of features and fusing them. In the design of the proposed recursive block, the following three strategies are employed to produce an enriched combination of features.

- *Multi-scale Convolution:* Features using different spatial ranges are generated by employing convolutions with kernels of different sizes, that is, the generated features are characterized by both the short and long range spatial information.

- *Sub-pixel Convolution:* Convolution operations followed by a depth-to-space transpose operation (also known as pixel shuffle operation) are carried out in order to generate features with different resolution levels.

- *Feature Fusion:* The features produced from different spatial ranges and those from different resolution levels are fused with the features that are used to produce these

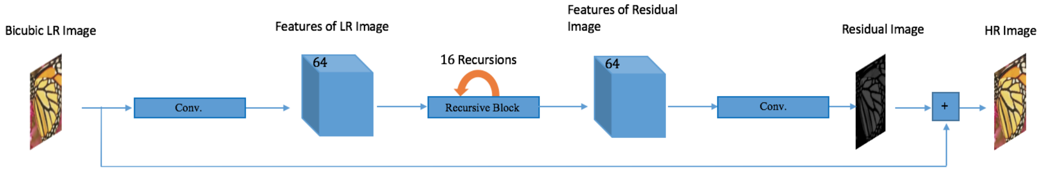


Figure 3.7: Overall architecture of the proposed super resolution scheme.

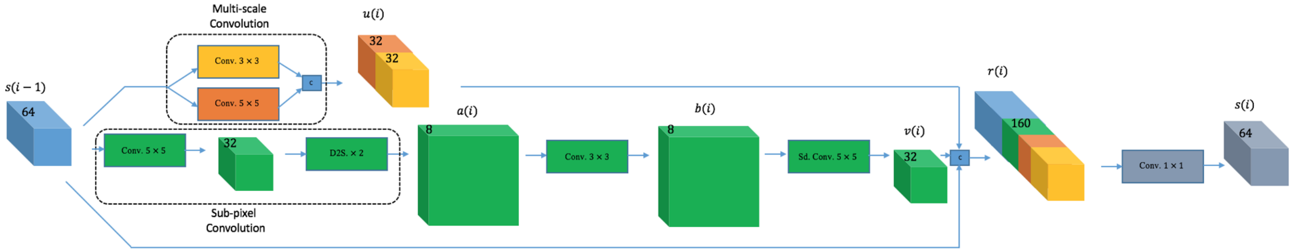


Figure 3.8: Architecture of the proposed recursive block.

two types of features in order for the recursive block to provide a very rich and representational set of feature maps.

The overall architecture of the proposed network for image super resolution is shown Fig. 3.7. The bicubic interpolated low resolution image, whose spatial resolution is the same as that of the ground truth, is fed to the proposed network. The features of the bicubic interpolated image are extracted using a convolutional layer, which employs 64 convolutional filters each of size 7×7. These feature maps are then processed by a recursive block to be developed and explained in the following paragraphs. Use of a recursive network would increase the nonlinear mapping capability of the image super resolution scheme, and therefore, would result in a better estimation of the high resolution image, if a sufficiently large number of recursions is used, while keeping the number of parameters of the network unchanged. However, since the effective depth of the network increases as the number of recursions is increased, the gradient vanishing problem of the network would appear thus hindering its learning process. To address this problem, a global residual skip connection is used in the proposed scheme through which the residue between the ground truth and bicubic interpolated image is learnt by the deep network. The output feature maps of the recursive block after completing all the recursions are fed to a convolutional layer, that employs a single convolutional filter of size 7×7, to obtain the residual image. Finally, the residual image is added to the bicubic interpolated image and the estimated high resolution image is yielded.

Our objective in this work is to increase the representational capability of the image super resolution network by extracting and fusing a variety of different types of feature maps, while at the same time keeping the number of parameters and the number of operations as low as possible. To lower the number of parameters of the network, the nonlinear mapping for image super resolution is carried out using a recursive block, which uses the same set of parameters from one recursion to the next. Since the number of operations in recursive network is directly proportional to the number of recursions carried out using the recursive block, special effort is made to have the number of operations carried out by the proposed recursive block in each recursion to a minimum by keeping its number of parameters as low as possible. To obtain a rich set of features, two different types of feature maps are generated in the recursive block. Specifically, features with different spatial ranges and features at different resolution levels are generated and concatenated with the features input to the recursive block.

Fig. 3.8 shows the architecture of the proposed recursive block, in which $s(i-1)$ and $s(i)$ represent the input and output features of the block in the ith recursion. We now explain how the two types of features are generated using convolution operations and how the generated feature maps are fused with the feature maps input to the recursive block.

(i) *Feature Generation with Different Spatial Ranges*: The convolution operation generates a new feature value by processing a local information. The smaller and larger size of convolutional kernels can result in capturing short and long-range local information to generate new feature values. The feature maps with these characteristics are obtained by employing a multi-scale convolution layer in the first branch of the proposed recursive block. The feature maps $u(i)$ are obtained through applying a multi-scale convolution operation on the input feature maps $s(i-1)$. The multi-scale convolution is carried out by performing convolution operations with kernels of different sizes and concatenating the results. While carrying out the multi-scale convolution, the first 32 channels are obtained by applying 32 kernels each of size 3×3 and another 32 channels are obtained by applying 32 kernels each of size 5×5, and the two sets of feature maps thus obtained are concatenated to obtain the feature maps $u(i)$.

(ii) *Feature Generation at Different Resolution Levels*: To generate the features obtained at different resolution levels for image super resolution, one can use a feature map upscaling method. The deconvolution operation (also known as transposed convolution operation) and the sub-pixel convolution operation are two of the neural network based methods for upscaling the feature maps. However, since the former uses zero padding to increase the spatial resolution of a feature map, the resulting feature maps suffer from check-board artifacts. Therefore, a sub-pixel convolutional layer is adopted in the second branch of the proposed recursive block to upscale the feature maps. In order to upscale the input feature maps $s(i-1)$, they are made to undergo a sub-pixel convolution operation, which consists of a convolutional layer with 32 kernels each of size 5×5, and a depth-to-space transpose operation with a depth-to-space factor of 2. Therefore, the resulting feature tensor $a(i)$ has 8 channels. It should be noted that by using this sub-pixel convolutional layer, the spatial resolution of each feature map in $a(i)$ is increased by a factor of 2. Feature maps $a(i)$ are further convolved with a convolutional layer using 8 kernels each of size 3×3 to obtain the feature maps $b(i)$. By using this convolution operation, the features in the new resolution level are processed. Thus, the resulting feature tensor $b(i)$ has 8 channels.

(iii) *Feature Fusion*: As mentioned earlier, our objective in designing the recursive block is to generate a rich set of feature maps with good representational capability. To this end, we concatenate the two generated feature maps, $u(i)$ and $b(i)$, with the feature maps that are input to the block, $s(i-1)$. The spatial dimension of feature maps $b(i)$ has a mismatch with that of the feature maps $s(i-1)$ and $u(i)$. Therefore, the spatial resolution of the channels of $b(i)$ is reduced to that of $s(i-1)$ and $u(i)$ through a strided convolutional layer with 32 kernels each of size 5×5 before the concatenation operation. The resulting feature tensor $v(i)$ has 32 channels. The stride in this convolution must have the same value as the depth-to-space factor of the sub-pixel convolution. The output $v(i)$ of the strided convolution operation is now concatenated with the feature maps $u(i)$ and $s(i-1)$ to obtain the feature tensor $r(i)$, which has 160 channels. Finally, the number of channels of the feature tensor $r(i)$ is reduced to that of $s(i-1)$ by employing a 1×1 convolutional layer in order to make its number of channels to be the same as that of the input to the block.

In view of the feature generating capabilities of the recursive block, as discussed above, and its use by the proposed scheme, we call our network a **Mu**ltiple spatial **R**ange and **R**esolution level feature generating deep recursive **Net**work (MuRNet) [89].

Since human eyes are more sensitive to the illumination information, the original RGB image is first transformed into a YCbCr image and then only its Y channel is input to the network. At the output, the restored Y channel is recombined to the Cb and Cr channels and the restored YCBCr image is transformed back into an RGB image.

As for DRRN [35] and MemNet [4], images from the dataset BSD200 [23] and those provided by Yang *et al.* [6] are also used to train MuRNet. It should be pointed out that the *Woman* image is removed from the BSD200 training set, since the same image is also used for evaluation. All the images are divided into 30363 sub-images each of size 48×48 to produce the training samples. In order to generate the degraded low resolution images, the bicubic downsampling operation is applied to the original high resolution images. Data augmentation including flipping and rotating by 90, 180 and 270 degrees is employed to increase the number of training samples to 242904. As in [3], MuRNet is multi-scale trained, and therefore, one set of parameters is adequate for all of the scaling factors. The weights of all the layers are initialized by the He et al. method [7], which takes into consideration the activation function as well as the number of filters and kernel sizes.

For updating the weights of the network and optimizing the loss function, the stochastic gradient descent (SGD) method along with the Nestrov acceleration algorithm and the momentum parameter with a value of 0.9 is employed. Initially a value of 0.1 is used for the learning rate and then it is decreased by a factor of 10 after each 10 epochs. The weight decay parameter is set to 10^{-4}.

It has been reported in [28] that using ℓ_1 norm representing the loss between the high resolution estimation and the ground truth leads to an improved performance in comparison to that obtained by employing the ℓ_2 norm loss function. However, since MemNet and DRRN, as two important light-weight recursive networks for super resolution, employ ℓ_2 loss function between the high resolution estimation and the ground truth, the same loss function is used for MuRNet.

3.7 Experimental Results

3.7.1 Experimental Results of PHMNet

The proposed residual block of PHMNet consists of two main modules to generate multi-scale features, one directly producing features at two different scales and the other one indirectly producing features at two different scales through their generation from two hierarchical levels of abstraction. To investigate the impact of each module individually on the network performance, we form two variants of the proposed residual block, namely, *Varaint 1* and *Variant 2*, containing either only the parallel multi-scale feature fusion module or only the hierarchical feature fusion module. The performance results of the super resolution networks using the proposed residual block and its two variants are given in Table 3.1. As seen from this table, removing any of the two multi-scale feature generation modules from the proposed residual block leads to a performance degradation.

The performance results in terms of PSNR and SSIM metrics of the proposed super resolution network and the state-of-the-art light-weight super resolution schemes, namely, super resolution using a convolutional neural network (SRCNN) [1], very deep network for super resolution (VDSR) [3], deeply recursive convolutional network (DRCN) [19], laplacian super resolution network (LapSRN) [29], deep residual recurrent network (DRRN) [20], very deep persistent memory network (MemNet) [4], information distillation network (IDN) [45], cascading residual networks (CARN) [14] and super resolution using a feedback network (SRFBN) [25], are given in Table 3.2. It is seen from this table that the proposed super resolution network provides 18 best values of PSNR and SSIM metrics out of a total of 24 values. Also, CARN [14] stands out as the second best performance method with 6 best values of PSNR and SSIM metrics.

The complexity of the various state-of-the-art light-weight super resolution schemes are given in Table 3.3. It is seen from this table that the proposed super resolution network PHMNet employs 104K less number of parameters than the second best super resolution network in terms of performance, namely, CARN does. In this regard, one can conclude that the proposed network provides the best results, when the performance and the complexity are both taken into consideration.

Table 3.1: Results on the ablation study of the proposed residual block of PHMNet.

Network with	*Set5*	*Set14*	*BSD100*
Variant 1	31.88 (0.8906)	28.56 (0.7811)	27.49 (0.7350)
Variant 2	32.04 (0.8934)	28.63 (0.7830)	27.56 (0.7371)
Proposed	32.12 (0.8948)	28.65 (0.7841)	27.58 (0.7386)

Table 3.2: PSNR (SSIM) values resulting from applying PHMNet and various state-of-the-art methods to images of four benchmark datasets.

Dataset	Scaling	Bicubic	SRCNN [1]	VDSR [3]	DRCN [19]	LapSRN [29]	DRRN [20]	MemNet [4]	IDN [45]	SRFBN [25]	CARN [14]	PHMNet (Proposed) [95]
Set5	×2	33.66 (0.9299)	36.66 (0.9542)	37.53 (0.9587)	37.63 (0.9588)	37.52 (0.959)	37.74(0.9591)	37.78 (0.9597)	37.83 (0.9600)	37.78 (0.9597)	37.76 (0.9590)	37.84 (0.9607)
	×3	30.39 (0.8682)	32.75 (0.9090)	33.66 (0.9213)	33.82 (0.9226)	N/A	34.03 (0.9244)	34.09 (0.9248)	34.11 (0.9253)	34.20 (0.9255)	34.29 (0.9255)	34.33 (0.9277)
	×4	28.42 (0.8104)	30.48 (0.8628)	31.35 (0.8838)	31.53 (0.8854)	31.54 (0.885)	31.68 (0.8888)	31.74 (0.8893)	31.82 (0.8903)	31.98 (0.8923)	32.13 (0.8937)	32.12 (0.8948)
Set14	×2	30.24 (0.8688)	32.42 (0.9063)	33.03 (0.9124)	33.04 (0.9118)	33.08 (0.913)	33.23 (0.9136)	33.28 (0.9142)	33.30 (0.9148)	33.35 (0.9156)	33.52 (0.9166)	33.54 (0.9177)
	×3	27.21(0.7385)	29.28 (0.8209)	29.77 (0.8314)	29.76 (0.8311)	N/A	29.96 (0.8349)	30.00 (0.8350)	29.99 (0.8354)	30.10 (0.8372)	30.29 (0.8407)	30.48 (0.8455)
	×4	26.00 (0.7027)	27.49 (0.7503)	28.01 (0.7674)	28.02 (0.7670)	28.19 (0.772)	28.21 (0.7721)	28.26 (0.7723)	28.25 (0.7730)	28.45 (0.7779)	28.60 (0.7806)	28.65 (0.7841)
BSD100	×2	29.56 (0.8431)	31.36 (0.8879)	31.90 (0.8960)	31.85 (0.8942)	31.80 (0.895)	32.05 (0.8973)	32.08 (0.8978)	32.08 (0.8985)	32.00 (0.8970)	32.09 (0.8978)	32.16 (0.9001)
	×3	27.21 (0.7385)	28.41 (0.7863)	28.82 (0.7976)	28.80 (0.7963)	N/A	28.95 (0.8004)	28.96(0.8001)	28.95 (0.8013)	28.96 (0.8010)	29.06 (0.8034)	29.16 (0.8081)
	×4	25.96 (0.6675)	26.90 (0.7101)	27.29 (0.7251)	27.23 (0.7233)	27.32 (0.728)	27.38 (0.7284)	27.40 (0.7281)	27.41 (0.7297)	27.44 (0.7313)	27.58 (0.7349)	27.58 (0.7386)
Urban100	×2	26.88 (0.8403)	29.50 (0.8946)	30.76 (0.9140)	30.75 (0.9133)	30.41 (0.910)	31.23 (0.9188)	31.31 (0.9195)	31.27 (0.9196)	31.41 (0.9207)	31.92 (0.9256)	31.56 (0.9239)
	×3	24.46 (0.7349)	26.24 (0.7989)	27.14 (0.8279)	27.15 (0.8276)	N/A	27.53 (0.8378)	27.56 (0.8376)	27.42 (0.8359)	27.66 (0.8415)	28.06 (0.8493)	28.02 (0.8515)
	×4	23.14 (0.6577)	24.52 (0.7221)	25.18 (0.7524)	25.14 (0.7510)	25.21 (0.756)	25.44 (0.7638)	25.50 (0.7630)	25.41 (0.7632)	25.71 (0.7719)	26.07 (0.7837)	25.91 (0.7826)

The values in the red font indicate the best performance and those in the blue font represent the second best performance.

Table 3.3: Complexity of various super resolution schemes.

Method	Number of Parameters
SRCNN [1]	57K
VDSR [3]	665K
DRCN [19]	1770K
DRRN [20]	297K
MemNet [4]	677K
IDN [45]	553K
SRFBN-S [25]	483K
CARN [14]	1592K
PHMNet (Proposed)	1488K

Fig. 3.9 shows the *img008* super resolved images using the proposed network and CARN selected from *Urban100* dataset. It is seen from this figure that the image super resolved by CARN contains some textures, which do not exist in the original ground truth image. On the other hand, the image super resolved by PHMNet contains the textures more similar to those of the ground truth image.

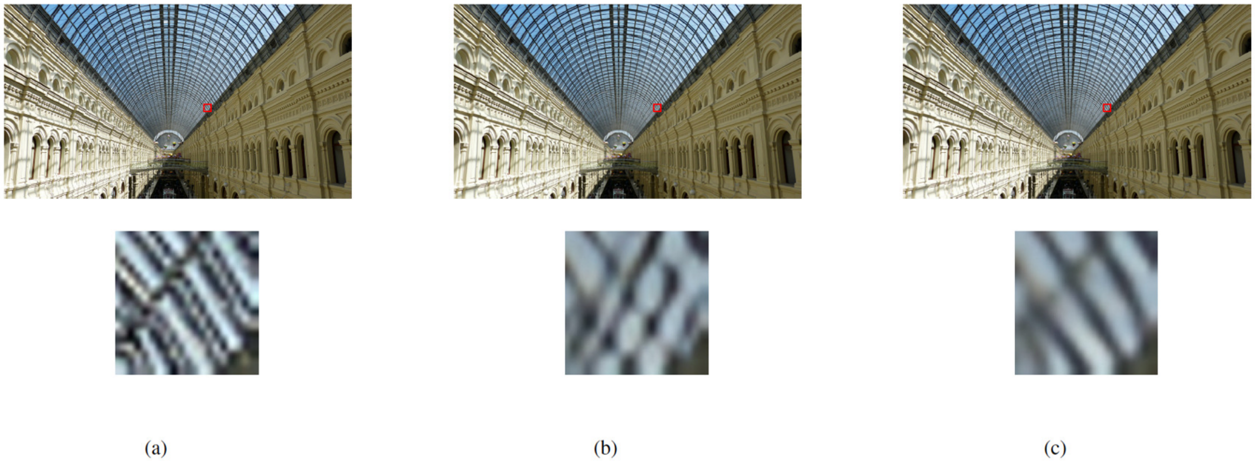


Figure 3.9: Visual quality of *img008* images super resolved by PHMNet and CARN with upscaling factor 3. (a) Ground truth image. Images super resolved by (b) CARN and (c) PHMNet.

Table 3.4: PSNR (SSIM) Results on the Ablation Study of the Proposed Residual Block of MGHCNet.

Network with	*Set5*	*Set14*	*BSD100*
Variant 1	34.25 (0.9268)	30.37 (0.8436)	29.07 (0.8068)
Variant 2	34.25 (0.9269)	30.38 (0.8437)	29.09 (0.8067)
Proposed	34.35 (0.9275)	30.44 (0.8446)	29.16 (0.8075)

3.7.2 Experimental Results of MGHCNet

In this section, first we carry out an ablation study on the proposed residual block to show the effectiveness of the various ideas used in its design. Then, the performance and complexity of the proposed super resolution network is compared with that of the state-of-the-art light-weight super resolution networks using four benchmark datasets, [21], [22], [23],[24].

The two feature generation modules in the proposed residual block of MGHCNet are multi-scale granular channel feature generation module and uni-scale holistic channel feature generation module. To investigate the contribution of each module on the network performance, we form two variants of the proposed residual block by removing only one of

Table 3.5: PSNR (SSIM) of Super Resolution Network employing Two Residual Blocks with Scaling Factor 3.

Network with	*Set5*	*Set14*	*BSD100*
RCAN Block	34.34 (0.9273)	30.39 (0.8434)	29.11 (0.8069)
Proposed	34.35 (0.9275)	30.44 (0.8446)	29.16 (0.8075)

Table 3.6: PSNR (SSIM) Values Resulting from Applying MGHCNet and Various State-of-the-art Methods to Images of Four Benchmark Datasets.

Dataset	Scaling	Bicubic	SRCNN [1]	VDSR [3]	DRCN [19]	LapSRN [29]	DRRN [20]	MemNet [4]	IDN [45]	SRFBN [25]	CARN [14]	MGHCNet (Proposed) [97]
Set5	×2	33.66 (0.9299)	36.66 (0.9542)	37.53 (0.9587)	37.63 (0.9588)	37.52 (0.959)	37.74(0.9591)	37.78 (0.9597)	37.83 (0.9600)	37.78 (0.9597)	37.76 (0.9590)	37.95 (0.9609)
	×3	30.39 (0.8682)	32.75 (0.9090)	33.66 (0.9213)	33.82 (0.9226)	N/A	34.03 (0.9244)	34.09 (0.9248)	34.11 (0.9253)	34.20 (0.9255)	34.29 (0.9255)	34.35 (0.9275)
	×4	28.42 (0.8104)	30.48 (0.8628)	31.35 (0.8838)	31.53 (0.8854)	31.54 (0.885)	31.68 (0.8888)	31.74 (0.8893)	31.82 (0.8903)	31.98 (0.8923)	32.13 (0.8937)	32.17 (0.8947)
Set14	×2	30.24 (0.8688)	32.42 (0.9063)	33.03 (0.9124)	33.04 (0.9118)	33.08 (0.913)	33.23 (0.9136)	33.28 (0.9142)	33.30 (0.9148)	33.35 (0.9156)	33.52 (0.9166)	33.65 (0.9182)
	×3	27.21(0.7385)	29.28 (0.8209)	29.77 (0.8314)	29.76 (0.8311)	N/A	29.96 (0.8349)	30.00 (0.8350)	29.99 (0.8354)	30.10 (0.8372)	30.29 (0.8407)	30.44 (0.8456)
	×4	26.00 (0.7027)	27.49 (0.7503)	28.01 (0.7674)	28.02 (0.7670)	28.19 (0.772)	28.21 (0.7721)	28.26 (0.7723)	28.25 (0.7730)	28.45 (0.7779)	28.60 (0.7806)	28.69 (0.7839)
BSD100	×2	29.56 (0.8431)	31.36 (0.8879)	31.90 (0.8960)	31.85 (0.8942)	31.80 (0.895)	32.05 (0.8973)	32.08 (0.8978)	32.08 (0.8985)	32.00 (0.8970)	32.09 (0.8978)	32.22 (0.9008)
	×3	27.21 (0.7385)	28.41 (0.7863)	28.82 (0.7976)	28.80 (0.7963)	N/A	28.95 (0.8004)	28.96(0.8001)	28.95 (0.8013)	28.96 (0.8010)	29.06 (0.8034)	29.16 (0.8075)
	×4	25.96 (0.6675)	26.90 (0.7101)	27.29 (0.7251)	27.23 (0.7233)	27.32 (0.728)	27.38 (0.7284)	27.40 (0.7281)	27.41 (0.7297)	27.44 (0.7313)	27.58 (0.7349)	27.59 (0.7384)
Urban100	×2	26.88 (0.8403)	29.50 (0.8946)	30.76 (0.9140)	30.75 (0.9133)	30.41 (0.910)	31.23 (0.9188)	31.31 (0.9195)	31.27 (0.9196)	31.41 (0.9207)	31.92 (0.9256)	31.75 (0.9254)
	×3	24.46 (0.7349)	26.24 (0.7989)	27.14 (0.8279)	27.15 (0.8276)	N/A	27.53 (0.8378)	27.56 (0.8376)	27.42 (0.8359)	27.66 (0.8415)	28.06 (0.8493)	27.99 (0.8503)
	×4	23.14 (0.6577)	24.52 (0.7221)	25.18 (0.7524)	25.14 (0.7510)	25.21 (0.756)	25.44 (0.7638)	25.50 (0.7630)	25.41 (0.7632)	25.71 (0.7719)	26.07 (0.7837)	25.88 (0.7813)

The values in the red font indicate the best performance and those in the blue font represent the second best performance.

Table 3.7: Complexity of Various Super Resolution Schemes.

Method	Number of Parameters
SRCNN [1]	57K
VDSR [3]	665K
DRCN [19]	1770K
DRRN [20]	297K
MemNet [4]	677K
IDN [45]	553K
SRFBN-S [25]	483K
CARN [14]	1592K
MGHCNet (Proposed)	1548K

the modules from the block. In *Variant 1*, the uni-scale holistic channel feature generation module is removed from the proposed residual block, whereas in *Variant 2*, the multi-scale granular channel feature generation module is removed from the proposed residual block. Table 3.4 shows the performance of the proposed residual block of MGHCNet and its two variants on the three benchmark datasets with the scaling factor of 3. It is seen from this

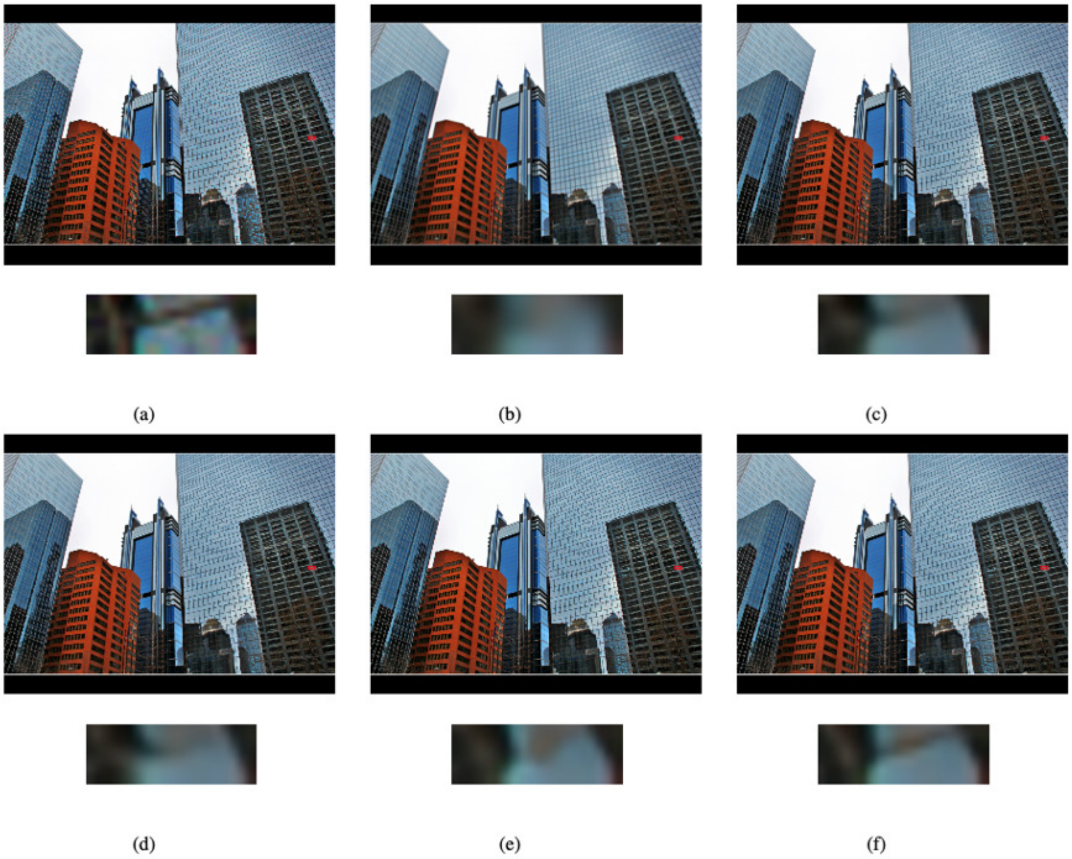


Figure 3.10: Visual comparison of *img099* images super resolved by MGHCNet and the state-of-the-art methods with upscaling factor 3. (a) Ground truth image. (b) Bicubic. (c) LapSRN. (d) DRRN. (e) CARN. (f) MGHCNet.

table that removing any of the two feature generation modules from the proposed residual block results in the performance degradation of the network. It is worth noting that the *Variant 1* and *Variant 2* have a comparable performance on three benchmark datasets and each has a PSNR value, which is about 0.1 dB lower than that provided by the proposed residual block. This means that each of the two modules has also equal contribution to improving the network performance.

The proposed residual block of MGHCNet fuses the uni-scale and multi-scale feature maps in order to improve the network representability. Recently, in [30], a residual channel attention network (RCAN) has been proposed by using a residual block consisting of two convolution layers and a squeeze-and-excitation module. The network of RCAN has been shown to provide a very good performance using the squeeze-and-excitation module. We now replace the proposed residual block by the residual block of RCAN in order to compare the network performance using this and our proposed residual block. We use 20 units of RCAN's block compared to 16 of ours in order to have a comparable number of parameters. The performance results are shown in Table 3.5. It is seen from these results that the proposed residual block results in a performance superior to that provided when RCAN residual block is used.

The performance of the proposed super resolution network (MGHCNet) and that of the nine state-of-the-art light-weight schemes are given in Table 3.6 on the four benchmark datasets. It is seen from this table that the proposed network outperforms the other state-of-the-art light-weight super resolution networks in 19 out of 24 cases of the PSNR and SSIM metrics. In the remaining cases of these two metrics, the proposed network is the second best performing scheme.

Fig. 3.10 shows the visual quality of the super resolved images obtained by applying the proposed and some of the best performing networks on *img 099* from the *Urban 100* dataset with the scaling factor 3. It is seen from the zoomed segments of the images in this figure that the segments of the super resolved image obtained by applying the proposed network are sharper.

Table 3.7 gives the number of parameters of the state-of-the-art light-weight networks used in comparison. It is seen from this table that the proposed network employs 44K less number of parameters than CARN does, which is the second best performing method. Based on the results given in Tables 3.6 and 3.7, one can conclude that the proposed network provides a high performance by employing a modest number of parameters.

Table 3.8: Impact of removing a branch on the network performance of SRNMFRB.

Network with	*Set5*	*Set14*	*BSD100*	*Urban100*	Parameters
Variant 1	37.84	33.44	32.10	31.31	395K
Variant 2	37.71	33.42	32.04	31.12	246K
Variant 3	37.81	33.49	32.14	31.43	486K
Proposed	37.88	33.56	32.16	31.50	560K

3.7.3 Experimental Results of SRNMFRB

The proposed residual block of SRNMFRB is composed of three parallel branches. In order to investigate the contribution of each of the three branches on the network performance, we form three variants of the proposed residual block, namely, *Variant 1*, *Variant 2* and *Variant 3*, by removing, respectively, first, second and third branch from the proposed block. Table 3.8 gives the performance of the network using the proposed residual block and its three variants on four benchmark datasets with the scaling factor of 2. It is seen from this table that removing any of the three branches from the residual block results in a degraded performance. It is also noted from Table 3.8 that despite the fact that the third branch, which generates features in a resolution lower than that of the original low resolution image, accounts for a very small number of parameters in the proposed network, its removal from the residual block degrades the network performance considerably. This shows that the third branch of the proposed residual block enhances the representational capability of the network significantly by generating features that are obtained in a receptive field higher than that of the first two branches.

The performance of the proposed super resolution network and nine state-of-the-art light-weight super resolution networks, namely, super resolution using a convolutional neural network (SRCNN) [1], very deep network for super resolution (VDSR) [3], sparse coding network (SCN) [2], Laplacian super resolution network (LapSRN) [29], deep recursive residual network (DRRN) [20], very deep persistent memory network (MemNet) [4], information distillation network (IDN) [45], cascaded residual network [14] and super

Table 3.9: PSNR (SSIM) values resulting from applying SRNMFRB and various state-of-the-art methods to images of four benchmark datasets.

Dataset	Scaling	Bicubic	SRCNN [1]	VDSR [3]	SCN [2]	LapSRN [29]	DRRN [20]	MemNet [4]	IDN [45]	CARN_M [14]	SRFBN [25]	SRNMFRB [98]
Set5	×2	33.66 (0.9299)	36.66 (0.9542)	37.53 (0.9587)	36.93 (0.9552)	37.52 (0.959)	37.74(0.9591)	37.78 (0.9597)	37.83 (0.9600)	37.53 (0.9583)	37.78 (0.9597)	37.88 (0.9607)
	×4	28.42 (0.8104)	30.48 (0.8628)	31.35 (0.8838)	30.86 (0.8710)	31.54 (0.885)	31.68 (0.8888)	31.74 (0.8893)	31.82 (0.8903)	31.92 (0.8903)	31.98 (0.8923)	31.83 (0.8903)
Set14	×2	30.24 (0.8688)	32.42 (0.9063)	33.03 (0.9124)	32.56 (0.9069)	33.08 (0.913)	33.23 (0.9136)	33.28 (0.9142)	33.30 (0.9148)	33.26 (0.9141)	33.35 (0.9156)	33.56 (0.9175)
	×4	26.00 (0.7027)	27.49 (0.7503)	28.01 (0.7674)	27.64 (0.7578)	28.19 (0.772)	28.21 (0.7721)	28.26 (0.7723)	28.25 (0.7730)	28.42 (0.7762)	28.45 (0.7779)	28.49 (0.7803)
BSD100	×2	29.56 (0.8431)	31.36 (0.8879)	31.90 (0.8960)	31.40 (0.8884)	31.80 (0.895)	32.05 (0.8973)	32.08 (0.8978)	32.08 (0.8985)	31.92 (0.8960)	32.00 (0.8970)	32.16 (0.9002)
	×4	25.96 (0.6675)	26.90 (0.7101)	27.29 (0.7251)	27.03 (0.7161)	27.32 (0.728)	27.38 (0.7284)	27.40 (0.7281)	27.41 (0.7297)	27.44 (0.7304)	27.44 (0.7313)	27.47 (0.7346)
Urban100	×2	26.88 (0.8403)	29.50 (0.8946)	30.76 (0.9140)	29.52 (0.8970)	30.41 (0.910)	31.23 (0.9188)	31.31 (0.9195)	31.27 (0.9196)	31.23 (0.9193)	31.41 (0.9207)	31.50 (0.9231)
	×4	23.14 (0.6577)	24.52 (0.7221)	25.18 (0.7524)	24.52 (0.7260)	25.21 (0.756)	25.44 (0.7638)	25.50 (0.7630)	25.41 (0.7632)	25.62 (0.7694)	25.71 (0.7719)	25.55 (0.7706)

The values in the red font indicate the best performance and those in the blue font represent the second best performance.

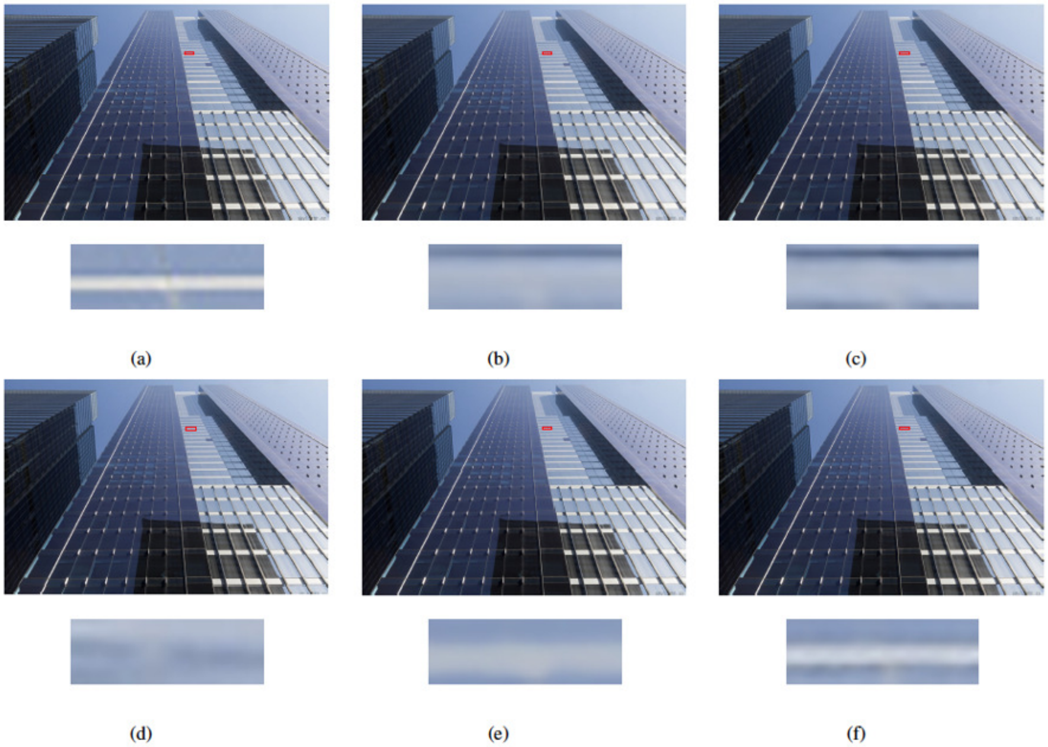


Figure 3.11: Visual comparison of *img033* images super resolved by SRNMFRB and the state-of-the-art methods with upscaling factor 4. (a) Ground truth image. (b) VDSR. (c) DRRN. (d) IDN. (e) CARN_M. (f) SRNMFRB.

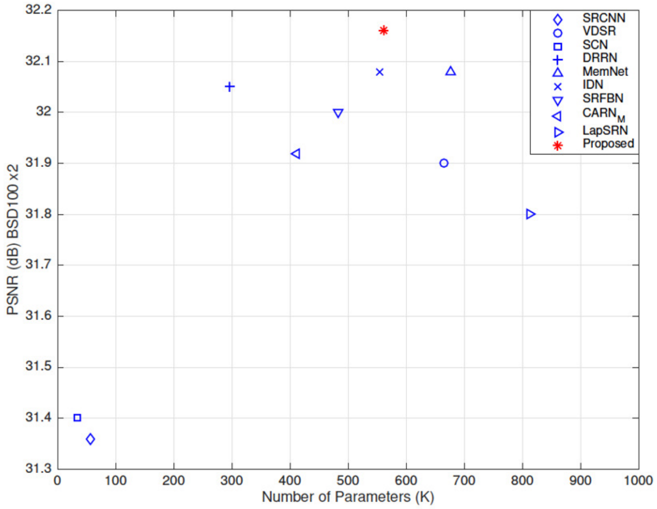


Figure 3.12: A plot of the PSNR versus number of parameters for different light-weight deep networks when applied to *BSD100* images with a scaling factor of 2 (Proposed refers to SRNMFRB).

resolution using a feedback network (SRFBN) [25], are given in Table 3.9. It should be pointed out that all the networks presented in Table 3.9 employ less than one million parameters. It is seen from this table that SRFBN [25] and the proposed SRNMFRB are the two best performing networks among all the state-of-the-art light-weight super resolution networks in terms of PSNR and SSIM metrics with the latter outperforming the former in 12 out of 16 cases of the two metrics.

Fig. 3.11 shows the visual quality of the images obtained by super resolving *img033* from the *Urban100* dataset using the proposed and some of the state-of-the-art light-weight super resolution networks with the scaling factor $\times 4$. It is seen from the zoomed parts of the images that the image that most resembles the ground truth results from the use of the proposed super resolution network.

Fig. 3.12 depicts a plot of PSNR versus number of parameters of the super resolution networks used for the comparison. It is seen from this figure that when both the performance and complexity of the state-of-the-art light-weight super resolution networks are simultaneously taken into consideration, the proposed network provides the best results. Finally, it should be pointed out that the number of arithmetic operations for the proposed network is simply proportional to its number of parameters. However, the same is not true for the second best performing network, namely, SRFBN [25], in view of its being a recursive feedback network.

3.7.4 Experimental Results of MISNet

In order to show the effectiveness of the multi-resolution level feature interpolation used by MISNet on the network performance, we form two variants of the proposed network, namely, *Variant 1* and *Variant 2*. In *Variant 1*, the low level feature maps are bilinear interpolated only at a single resolution, i.e., at the desired level s of the image super-resolution. In other words, the network does not employ the idea of multi-resolution levels of feature interpolation.*Variant 2* is formed by removing the second branch all together from the architecture of Fig. 3.6. Therefore, this variant does not perform feature interpolation even at a single level and degenerates the network into a residual-free architecture. Table 3.10 gives the performance, in terms of PSNR and SSIM metrics, of the proposed MISNet and its two variants, when the scaling factor 4 is used. By comparing the corresponding results of *Variant 1* and the proposed network of this table, it is seen that the generation of the features at multi-resolution level of feature interpolation indeed results in a superior performance. Also, by comparing the corresponding results of *Variant 2* and the proposed network, it is seen that the performance of *Variant 2* is very much inferior to that of the proposed network, thus, showing that the use of multi-resolution level feature interpolation in the framework of residual learning is very effective in providing a very good super-resolution performance.

The performance in terms of PSNR and SSIM of the proposed MISNet and that of the other state-of-the-art ultralight-weight super-resolution networks in the literature, namely,

Table 3.10: PSNR (SSIM) Results of the Ablation Study Performed on the Proposed MISNet.

Network with	*Set5*	*Set14*	*Urban100*
Variant 1	30.96 (0.8771)	28.02 (0.7692)	24.79 (0.7408)
Variant 2	30.58 (0.8718)	27.76 (0.7629)	24.54 (0.7298)
Proposed	30.97 (0.8773)	28.04 (0.7692)	24.82 (0.7419)

Table 3.11: PSNR (SSIM) Values Resulting from Applying the Proposed and Various State-of-the-art Ultralight-weight Super Resolution Networks to Images from Four Benchmark Datasets.

Dataset	Scaling	Bicubic	SRCNN [1]	SCN [2]	FSRCNN [13]	PISR [46]	MISNet(Proposed)
Set5	×2	33.66 (0.9299)	36.66 (0.9542)	36.93 (0.9252)	37.00 (0.9558)	37.33 (0.9576)	37.31 (0.9585)
	×4	28.42 (0.8104)	30.48 (0.8628)	30.86 (0.8710)	30.71 (0.8657)	30.95 (0.8759)	30.97 (0.8773)
Set14	×2	30.24 (0.8688)	32.42 (0.9063)	32.56 (0.9069)	32.63 (0.9088)	32.79 (0.9105)	33.03 (0.9130)
	×4	26.00 (0.7027)	27.49 (0.7503)	27.64 (0.7578)	27.59 (0.7535)	27.77 (0.7615)	28.04 (0.7692)
BSD100	×2	29.56 (0.8431)	31.36 (0.8879)	31.40 (0.8884)	31.53 (0.8920)	31.65 (0.8926)	31.70 (0.8945)
	×4	25.96 (0.6675)	26.90 (0.7101)	27.03 (0.7161)	26.98 (0.7150)	27.08 (0.7188)	27.11 (0.7230)
Urban100	×2	26.88 (0.8403)	29.50 (0.8946)	29.52 (0.8970)	29.88 (0.9020)	30.24 (0.9071)	30.22 (0.9080)
	×4	23.14 (0.6577)	24.52 (0.7221)	24.52 (0.7260)	24.62 (0.7280)	24.82 (0.7393)	24.82 (0.7419)

The values in the red font indicate the best performance.

the super-resolution convolutional neural network (SRCNN) [1], fast super-resolution convolutional neural network (FSRCNN) [13], sparse coding network (SCN) [2] and privileged information super-resolution network (PISR) [46], on the four benchmark datasets with the scaling factors 2 and 4 are given in Table 3.11. It is seen from the results of this table that the proposed super-resolution network outperforms these other state-of-the-art ultralight-weight super-resolution schemes on the all the four benchmark datasets. Since the idea of multi-resolution level feature interpolation, as used in the proposed scheme, is better suited in reconstructing the objects with different scales in an image, it can be expected to provide superior visual quality super resolved images. This advantage of the use of the multi-resolution level feature interpolation in the proposed scheme is indeed seen from the significantly higher SSIM values in Table 3.11 as provided by our scheme in comparison to that provided by PISR [46], the best performing ultralight-weight scheme existing in the literature. In view of the fact that the proposed super-resolution scheme employs only 60K

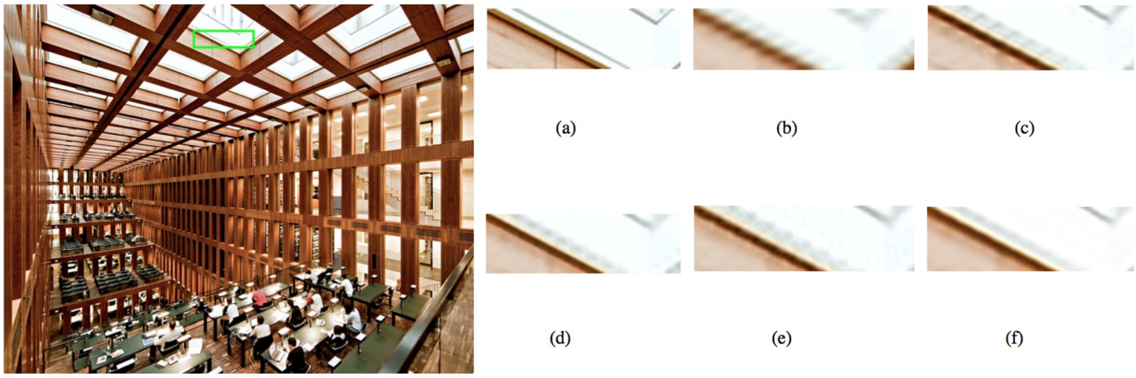


Figure 3.13: The visual quality of *img049* images super resolved by various schemes with the scaling factor of 4. (a) Ground truth. (b) Bicubic. (c) SRCNN. (d) SCN. (e) FSRCNN. (f) MISNet (Proposed).

parameters for super resolving the low-resolution images, indeed can be considered to be an ultralight-weight network.

Fig. 3.13 shows the *Urban100 img049* super resolved images obtained by using the proposed and other state-of-the-art ultralight-weight networks, when the scaling factor 4 is used. It is to be noted that we have not been able to include the super resolved image obtained from using the PISR scheme [46] in our comparison of the visual quality of the super resolved images, since the trained parameters for the scaling 4 has not been made available by the authors of this scheme. It is seen from the zoomed segments of the super resolved images obtained from the various schemes that the proposed network is able to recover edges of this part of the ceiling of the building that is most similar to the corresponding part of the ground truth image.

3.7.5 Experimental Results of MuRNet

First, the effect of the various kernel sizes for the multi-scale convolution on the performance of MuRNet is investigated. For this purpose, three different combinations of the kernel sizes, namely, 3×3 *and* 7×7, 5×5 *and* 7×7, and 3×3 *and* 5×5 are used for

Table 3.12: PSNR (SSIM) values of MuRNet with various combinations of kernel sizes for the multi-scale convolution, when applied to *Set14* images.

Upscaling	3×3 *and* 5×5	3×3 *and* 7×7	5×5 *and* 7×7
×2	33.43 (0.9160)	33.35 (0.9154)	33.40 (0.9158)
×3	30.16 (0.8384)	30.13 (0.8376)	30.12 (0.8381)
×4	28.46 (0.7772)	28.41 (0.7759)	28.44 (0.7765)

Table 3.13: PSNR (SSIM) values of MuRNet with various number of recursions, when applied to *Set14* images.

Upscaling	$16 Recursions$	$11 Recursions$	$6 Recursions$
×2	33.43 (0.9160)	33.32 (0.9149)	33.21 (0.9142)
×3	30.16 (0.8384)	30.09 (0.8371)	30.04 (0.8361)
×4	28.46 (0.7772)	28.37 (0.7750)	28.26 (0.7726)

Table 3.14: Impact of using single and multiple local spatial ranges on the performance of MuRNet when applied to *Set14* images.

Upscaling	MuRNet	Variant (3×3)	Variant (5×5)
×2	33.43 (0.9160)	33.34 (0.9153)	33.32 (0.9151)
×3	30.16 (0.8384)	30.09 (0.8378)	30.08 (0.8378)
×4	28.46 (0.7772)	28.38 (0.7758)	28.38 (0.7757)

the multi-scale convolutions. The performance of the super resolution network for each of these three different combinations of the kernel spatial supports is given by Table 3.12. As seen from this table, the super resolution network using the multi-scale convolution with the combination of 3×3 *and* 5×5 for spatial supports provides the best performance with an additional advantage of consuming the least number of parameters. Since the proposed recursive super resolution network, MuRNet, is sufficiently deep and its overall receptive field is large enough, the use of the larger kernel sizes for the multi-scale convolution is not helpful in improving the performance.

We now investigate the effect of the number of recursions using the proposed recursive block on the performance of the network. Since our objective is to design a super resolution

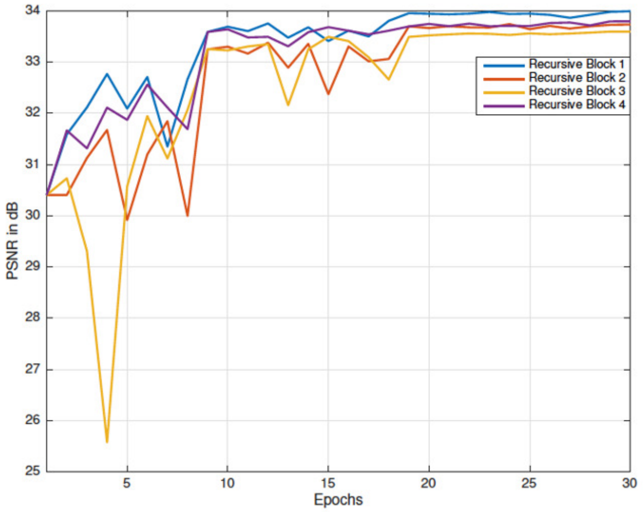


Figure 3.14: Training curves of the network with the architectural details of the recursive blocks specified in Table 3.16, obtained from *Set5* images downscaled by a factor of 3.

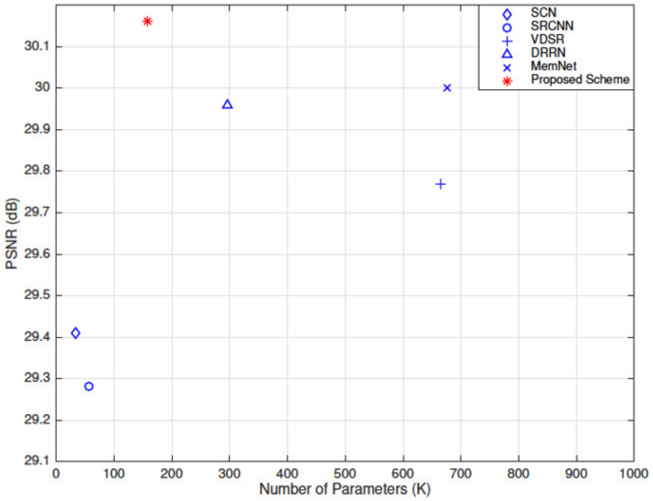


Figure 3.15: A plot of the performance versus number of parameters for MuRNet and different light-weight deep networks when applied to *Set14* images with a scaling factor of 3.

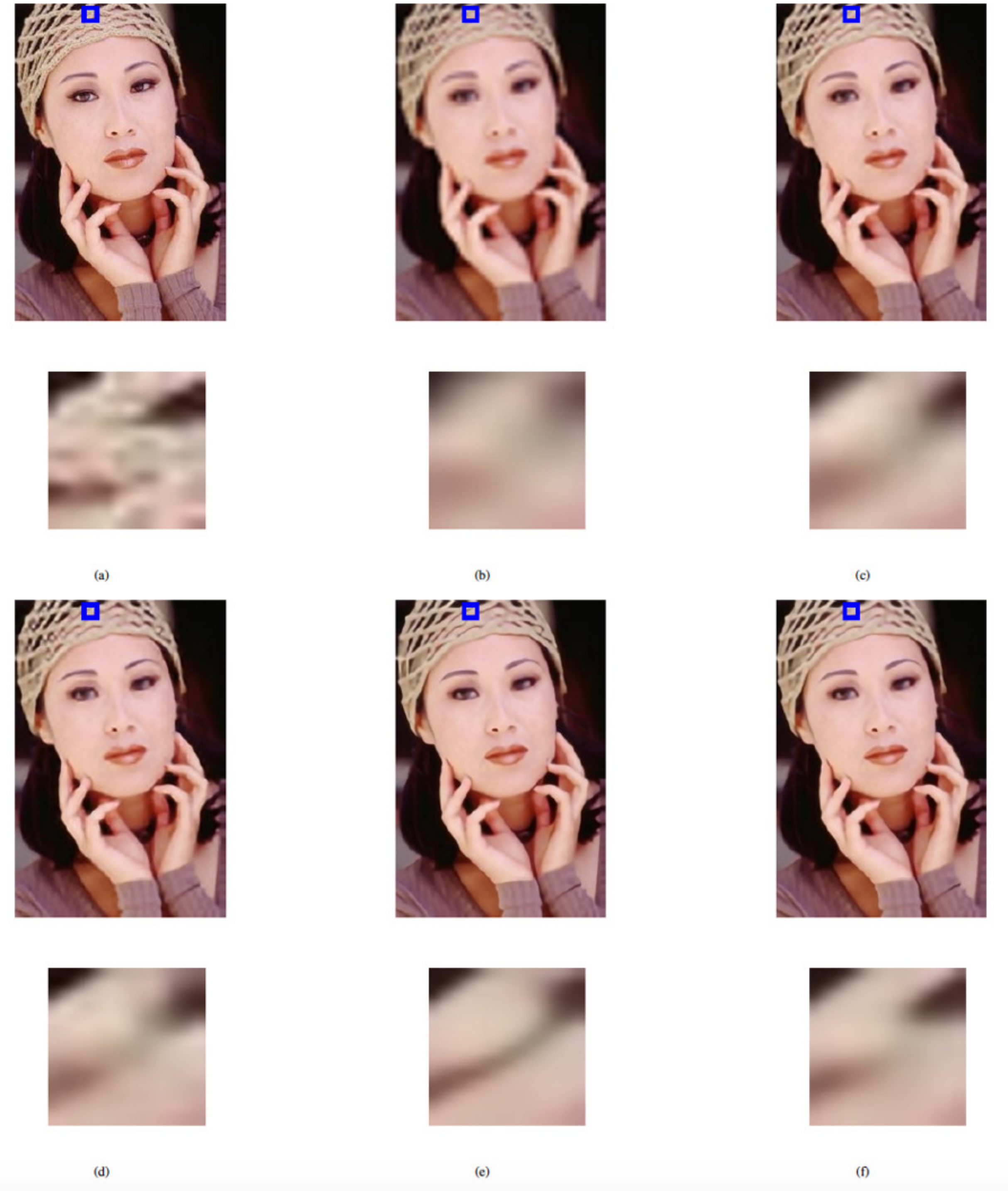


Figure 3.16: Visual qualities of the *Woman* images super resolved with a scaling factor of 4 obtained by applying MuRNet and the state-of-the-art schemes. (a) Ground truth. (b) Bicubic. (c) A+.(d) SRCNN. (e) DRRN. (f) MuRNet.

Table 3.15: Impact of using single and multiple resolution levels on the performance of MuRNet when applied to *Set14* images.

Upscaling	MuRNet	Variant (Single Resolution)
×2	33.43 (0.9160)	33.32 (0.9150)
×3	30.16 (0.8384)	30.11 (0.8378)
×4	28.46 (0.7772)	28.41 (0.7760)

Table 3.16: Architectural details of the recursive block of MuRNet after removing its individual branches.

Case	Recursive Block
1	With skip connection, multi-scale convolution and sub-pixel convolution
2	With multi-scale convolution and sub-pixel convolution
3	With skip connection and sub-pixel convolution
4	With skip connection and multi-scale convolution

Table 3.17: PSNR (SSIM) values* resulting from applying MuRNet and various light-weight methods to images of three benchmark datasets.

Dataset	Scaling	Bicubic	A+ [47]	SRCNN [1]	SCN [2]	VDSR [3]	MSCN [49]	DRRN [20]	MemNet [4]	MuRNet (Proposed) [89]
Set5	×2	33.66 (0.9299)	36.54 (0.9544)	36.66 (0.9542)	36.93 (0.9552)	37.53 (0.9587)	37.16 (0.9565)	37.74 (0.9591)	37.78 (0.9597)	37.67 (0.9598)
	×3	30.39 (0.8682)	32.58 (0.9088)	32.75 (0.9090)	33.10 (0.9144)	33.66 (0.9213)	33.33 (0.9155)	34.03 (0.9244)	34.09 (0.9248)	33.99 (0.9236)
	×4	28.42 (0.8104)	30.28 (0.8603)	30.48 (0.8628)	30.86 (0.8732)	31.35 (0.8838)	31.08 (0.8740)	31.68 (0.8888)	31.74 (0.8893)	31.67 (0.8871)
Set14	×2	30.24 (0.8688)	32.28 (0.9056)	32.42 (0.9063)	32.56 (0.9074)	33.03 (0.9124)	32.85 (0.9084)	33.23 (0.9136)	33.28 (0.9142)	33.43 (0.9160)
	×3	27.21(0.7385)	29.13 (0.8188)	29.28 (0.8209)	29.41 (0.8238)	29.77 (0.8314)	29.65 (0.8272)	29.96 (0.8349)	30.00 (0.8350)	30.16 (0.8384)
	×4	26.00 (0.7027)	27.32 (0.7491)	27.49 (0.7503)	27.64 (0.7578)	28.01 (0.7674)	27.87 (0.7624)	28.21 (0.7721)	28.26 (0.7723)	28.46 (0.7772)
BSD100	×2	29.56 (0.8431)	31.21 (0.8863)	31.36 (0.8879)	31.40 (0.8884)	31.90 (0.8960)	31.65 (0.8928)	32.05 (0.8973)	32.08 (0.8978)	31.96 (0.8977)
	×3	27.21 (0.7385)	28.29 (0.7835)	28.41 (0.7863)	28.50 (0.7885)	28.82 (0.7976)	28.66 (0.7941)	28.95 (0.8004)	28.96 (0.8001)	28.91 (0.8012)
	×4	25.96 (0.6675)	26.82 (0.7087)	26.90 (0.7101)	27.03 (0.7161)	27.29 (0.7251)	27.19 (0.7229)	27.38 (0.7284)	27.40 (0.7281)	27.37 (0.7294)

∗ The values in the red font indicate the best performance and those in the blue font represent the second best performance.

network, which employs a smaller number of parameters and has the number of multiply-accumulate operations as low as possible, we implement the proposed scheme with 6, 11 and 16 recursions. The performance of MuRNet with these three numbers of recursions is listed in Table 3.13. It is seen from these results that MuRNet with 16 recursions outperforms the shallower versions of MuRNet in terms of both PSNR and SSIM.

The proposed scheme for the design of the recursive block is based on the strategies of feature generations using multiple local spatial ranges and different resolution levels. We now carry out an ablation study on the impact of these two strategies on the network performance, as well as on the impact of the individual contributions of the three types of

Table 3.18: Complexity of the various light-weight super resolution schemes.

Method	Number of Parameters	Number of MACC Operations
SRCNN	57K	52.7G
VDSR	665K	612.6G
DRRN	297K	6796.9G
MemNet	677K	2662.4G
MuRNet (Proposed)	157K	2066.8G

Table 3.19: Performance and complexity of the best performing networks on images from validation set of *DIV2K* dataset degraded by realistic image processing artifacts.

Upscaling	MemNet	DRRN	MuRNet	
Number of Parameters	677K	297K	157K	309K
Performance	27.62 (0.8120)	27.86 (0.8200)	27.71 (0.8148)	28.32 (0.8244)

Table 3.20: Performance and complexity of MuRNet and recursive networks using the state-of-the-art feature generating blocks. RN denotes recursive network.

Upscaling	RN using DBPN Blocks	RN using the block of [48]	RN using RDN block	MuRNet
Parameters	414K	286K	333K	157K
×2	33.23 (0.9137)	33.40 (0.9154)	33.41 (0.9153)	33.43 (0.9160)
×3	30.04 (0.8357)	30.15 (0.8381)	30.15 (0.8381)	30.16 (0.8384)
×4	28.30 (0.7727)	28.37 (0.7750)	28.46 (0.7768)	28.46 (0.7772)

Table 3.21: Performance of ultralight-weight MuRNet and MuRNet when applied to *Set5* images.

Upscaling	Ultra-light-weight MuRNet	MuRNet
×2	37.38 (0.9583)	37.67 (0.9598)
×3	33.61 (0.9200)	33.99 (0.9236)
×4	31.32 (0.8806)	31.67 (0.8871)

features fused by the recursive block on the overall performance of the network.

To investigate the effect of feature generation using different spatial ranges, as opposed to that using a single spatial range, through the first branch of the recursive block on the performance of image super resolution, the multi-scale convolution layer of the recursive block with 32 filters of spatial support of 3×3 and 32 filters of spatial support of 5×5 is replaced by a convolution layer with 64 filters each of spatial support of only 3×3 (Variant (3×3)) or 5×5 (Variant (5×5)). In other words, the recursive block no longer uses multiple local spatial ranges. Table 3.14 shows the performance of MuRNet with multi-scale and uni-scale convolutions on the *Set14* [22] images. As seen from this table, MuRNet

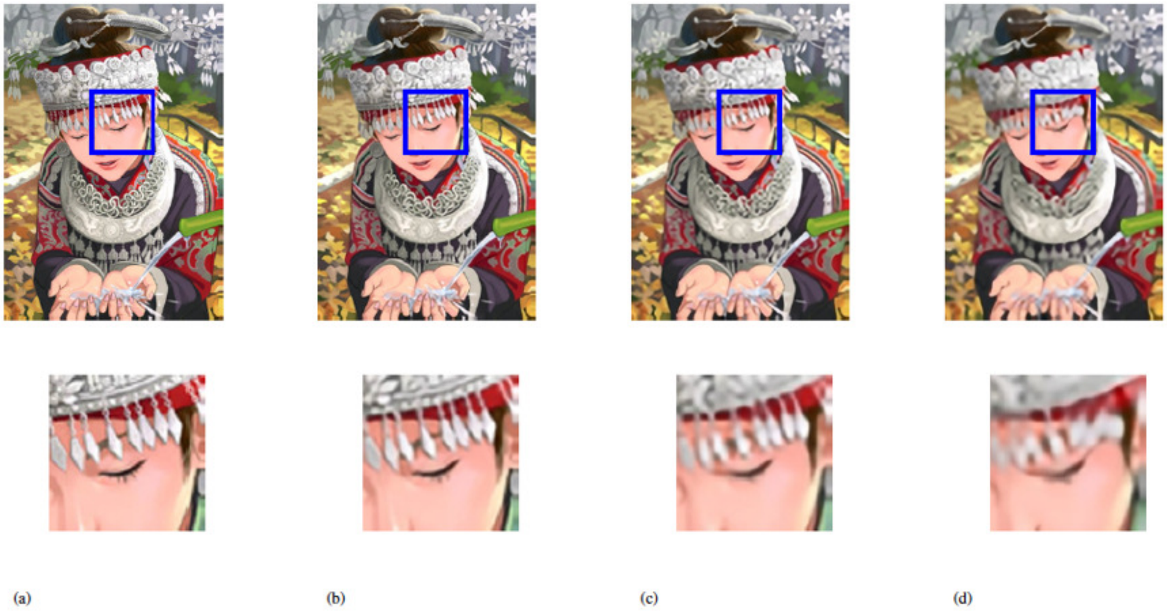


Figure 3.17: Images obtained by applying MuRNet on the *Comic* image with different upscaling factors. (a) Ground truth. (b) Upscaling factor 2. (c) Upscaling factor 3. (d) Upscaling factor 4.

with multi-scale convolutions outperforms MuRNet with uni-scale convolutions in terms of both the objective and subjective metrics. This shows that using the multi-scale convolution operation to extract features with the use of a combination of local short and long spatial ranges improves the representational capability of the super resolution network.

To investigate the effect of feature generation at different resolution levels, as opposed to that using a single resolution level, through the second branch of the recursive block on the performance of super resolution network, the sub-pixel convolution layer, which consists of a convolution operation with 32 filters of spatial support of 5×5 and a depth-to-space transpose operation with a scaling factor 2, is replaced by a convolution layer with 32 filters of 5×5 spatial support. In other words, the depth-to-space transpose part of the sub-pixel convolution is removed from the second branch of the recursive block. Also, the strided convolution layer with 32 filters of spatial support of 5×5 and stride 2, is replaced

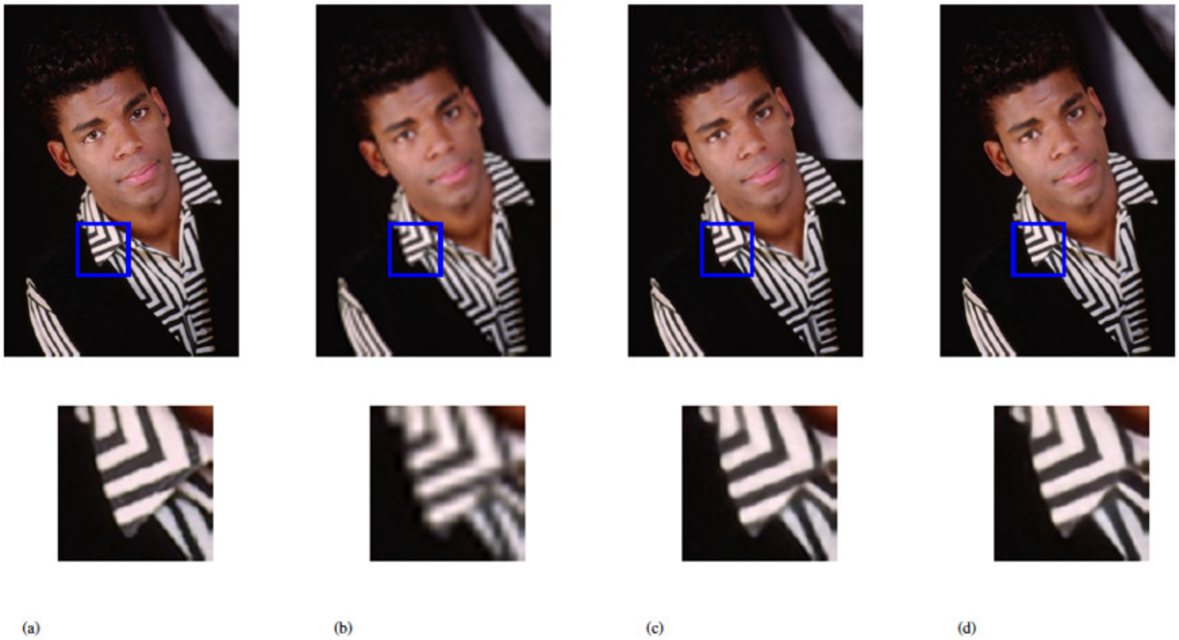


Figure 3.18: Visual qualities of the *302008* images with a scaling factor of 4 obtained by applying MuRNet and its ultralight-weight version. (a) Ground truth. (b) Bicubic Interpolation. (c) MuRNet. (d) ultra-light weight MuRNet with $33K$ parameters.

by a convolution layer with 32 filters of 5×5 spatial support. Thus, with these modifications, the recursive block generates features only at a single resolution level (Variant (Single Resolution)). The performance of MuRNet using single and multiple resolution levels are shown in Table 3.15. It is seen from this table that the super resolution network using multiple resolution levels outperforms the network using only a single resolution level. This happens despite the fact that the latter version of the network uses a slightly larger number of parameters. Thus, we conclude that the feature generation at different resolution levels improves the representational capability, and hence, the performance of the super resolution network.

Finally, to study the contribution of each of the individual types of features for fusion in the recursive block, we obtain the performance of the network by removing one of the branches at a time. The architectural details of the recursive block for each of these three cases are given in Table 3.16 and Fig. 3.14 shows the learning curves corresponding to each of these three cases on the *Set5* [21] images with a scaling factor of 3. It is seen from this figure that removing any of the branches from the proposed recursive block results in degrading the image super resolution performance significantly. Since the default number of recursions in the proposed recursive block is 16, the proposed super resolution network becomes effectively deep with these many recursions. In this case, the gradient vanishing problem could degrade the performance of image super resolution, if it is not properly handled. The skip connections as carried out by the third branch of the recursive block handles this problem very effectively. As seen from the learning curve in Fig. 3.14, removing the third branch from the recursive block, i.e., case 2, degrades the performance of the network quite significantly. It is also seen that removing the multi-scale convolution branch from the proposed recursive block, i.e., case 3, has the largest degrading impact on the performance of the network.

We now provide the performance of proposed MuRNet using the benchmark datasets, *Set5* [21], *Set14* [22] and *BSD100* [23]. The performance is compared with that of six state-of-the-art light-weight schemes for the super resolution of the interpolated images, namely, super resolution via convolutional neural network (SRCNN) [1], sparse coding network (SCN) [2], mixture sparse coding network (MSCN) [49], very deep network for super resolution (VDSR) [3], deep residual recursive network (DRRN) [20] and MemNet [4].

Table 3.17 gives the performance results in terms of PSNR and SSIM for the various schemes. It is seen from this table that the best performance is provided by DRRN, MemNet and MuRNet. Out of the 18 performance values (both PSNR and SSIM combined together), DRRN stands out as the second best in 10 of these values. MemNet has 9 best and 7 second best values and MuRNet scores 9 best and 1 second best values. Also, in a number of instances where MuRNet provides the third best result, they are very close to the second best ones. Based on these results, the proposed MuRNet can be regarded to generally outperform DRRN and to be closely comparable to MemNet.

Table 3.18 shows the complexity in terms of numbers of parameters and multiply-accumulate operations employed by the networks. It is seen that the number of parameters employed by MuRNet is approximately one-half and one-fifth of that of DRRN and MemNet, respectively. Also, the number of multiply-accumulate operations of MuRNet for an image of size 1280×720 is 30% and 77% of that of DRRN and MemNet, respectively. Thus, proposed MuRNet has the lowest complexity among light-weight category of the networks for super resolution of interpolated images.

In order to investigate the effectiveness of the proposed network in super resolving the images that are degraded by realistic image processing artifacts, we train the proposed MuRNet, as well as DRRN and MemNet, the two best performing super resolution networks in the literature, using 30363 sub-images each of size 48×48 that are randomly selected from images of the Flickr dataset [51] and their corresponding low resolution versions degraded by realistic image processing artifacts [52]. Table 3.19 gives the number of parameters used and the performances of these three networks on the benchmark validation set from the *DIV2K* dataset. It is to be noted that the images contained in this dataset are degraded by some unknown but realistic image processing artifacts [52]. It is seen from this table that MuRNet with the default number of parameters of 157K outperforms MemNet. On the other hand, the performance of MuRNet is inferior to that of DRRN, which employs 297K parameters. Thus, in order to provide a fair comparison with DRRN, we increase the number of parameters of MuRNet to about the same level as that of DRRN, i.e., 309K, by increasing the number of filters in each of the layers of MuRNet from 32 to 64. As seen from Table 3.19, MuRNet with the increased number of parameters significantly outperforms DRRN as well. It is also interesting to note that in this case, the performance of MuRNet then becomes very significantly higher than that of MemNet with the number of parameters of the former being more than 50% lower than that of the latter.

Fig. 3.15 depicts the plot of the PSNR values versus the number of parameters for the super resolution networks used for comparison, when they are applied to *Set14* images with the upscaling factor of 3. It is seen from this figure that MuRNet provides the best performance with a very small number of parameters among all the super resolution networks considered.

Fig. 3.16 shows the *Woman* image super resolved by the various image super resolution schemes with the upscaling factor of 4. It is seen from the zoomed part of the *Woman* image in this figure that all the methods except MuRNet fail to reconstruct the texture of this part in the woman's hat. However, the reconstructed texture by MuRNet is very similar to that of the ground truth.

Fig. 3.17 shows the super resolved *Comic* images obtained from MuRNet using the upscaling factors of 2, 3 and 4. As expected, the reconstructed images using a smaller scaling factor contains more details and high frequency contents.

We now implement three recursive networks, the first one using the upsampling and downsampling blocks proposed in DBPN [32], the second one employing the block proposed in [48] and the third one using the residual block proposed in RDN [44], and compare their performance with that of the proposed MuRNet. Table 3.20 gives the number of parameters employed by these recursive networks along with their performance on the images of the Set14 dataset. It is seen from this table that the proposed MuRNet outperforms the recursive network using the blocks of DBPN [32] by employing much smaller number of parameters. Also, it is seen from Table 3.20 that the performance of the proposed MuRNet is marginally superior to those of the networks using the blocks of [48] and [44], while the numbers of parameters of the two latter networks are significantly larger than that of the proposed MuRNet.

One of the main objectives of the design of MuRNet has been to design a light-weight network with a good performance for the super resolution problem. In order to study the performance of MuRNet with the number of parameters reduced from that of the default number, we obtain its performance by reducing number of filters employed by it. For this purpose, we use 32 filters for each of the convolution operations used for extracting the features from the bicubic interpolated image and point-wise convolution used in the recursive block, 8 filters for each convolution operation performed in the multi-scale convolution, 16 filters for the convolution operation of the sub-pixel convolution and also 16 filters for the strided convolution, and 4 filters for the second convolution of the second branch of the recursive block, resulting in only 33K parameters employed by the entire network.

Table 3.21 gives the performance of MuRNet and its ultralight-weight version with reduced number of parameters, when these networks are applied to the *Set5* images. It is seen from this table that by reducing the number of parameters of MuRNet by one-fifth, its performance degrades by about 0.3dB on the images of *Set5* dataset. However, even with this degradation, the performance of the ultralight-weight version of MuRNet is much superior to that of SRCNN.

In Fig. 3.18, the visual quality of images obtained by applying MuRNet and its ultralight-weight version are compared. Figs. 3.18 (a) and (b) show, respectively, the original image *302008* from the *BSD100* dataset and its degraded version obtained from applying bicubic downsampling, whereas Figs. 3.18 (c) and (d) are the restored super resolved images obtained from these two networks. It is seen from this figure that even though the ultralight-weight MuRNet employs very small number of parameters, it is still able to reconstruct the edges and textures of the original image.

Table 3.22: PSNR (SSIM) values* resulting from applying various light-weight multi-scale feature generating methods to images of three benchmark datasets.

Dataset	Scaling	MISNet [93]	MuRNet [89]	SRNMFRB [98]	PHMNet [95]	MGHCNet [97]
Set5	×2	37.31 (0.9585)	37.67 (0.9598)	37.88 (0.9607)	37.84 (0.9607)	37.95 (0.9609)
	×3	N/A	33.99 (0.9236)	N/A	34.33 (0.9277)	34.35 (0.9275)
	×4	30.97 (0.8773)	31.67 (0.8871)	31.83 (0.8903)	32.12 (0.8948)	32.17 (0.8947)
Set14	×2	33.03 (0.9130)	33.43 (0.9160)	33.56 (0.9175)	33.54 (0.9177)	33.65 (0.9182)
	×3	N/A	30.16 (0.8384)	N/A	30.48 (0.8455)	30.44 (0.8456)
	×4	28.04 (0.7692)	28.46 (0.7772)	28.49 (0.7803)	28.65 (0.7841)	28.69 (0.7839)
BSD100	×2	31.70 (0.8945)	31.96 (0.8977)	32.16 (0.9002)	32.16 (0.9001)	32.22 (0.9008)
	×3	N/A	28.91 (0.8012)	N/A	29.16 (0.8081)	29.16 (0.8075)
	×4	27.11 (0.7230)	27.37 (0.7294)	27.47 (0.7346)	27.58 (0.7386)	27.59 (0.7384)
Urban100	×2	30.22 (0.9080)	N/A	31.50 (0.9231)	31.56 (0.9239)	31.75 (0.9254)
	×3	N/A	N/A	N/A	28.02 (0.8515)	27.99 (0.8503)
	×4	24.82 (0.7419)	N/A	25.55 (0.7706)	25.91 (0.7826)	25.88 (0.7813)

∗ The values in the red font indicate the best performance.

3.8 Comparison between Various Proposed Deep Image Super Resolution Networks using Multi-scale Feature Generation

Performance results of various various deep multi-scale image super resolution networks proposed in this chapter are given in Table 3.22. From the results of this table, the following conclusions can be drawn. First, it is seen that the super resolution networks of PHMNet, MGHCNet, SRNMFRB and MISNet are designed to be applied on the original degraded low resolution image. On the other hand, the purpose of designing MuRNet is to enhance the quality of the bicubically interpolated versions of the low resolution images. Hence, the number of arithmetic operations employed by MuRNet is significantly larger than those employed by PHMNet, MGHCNet, SRNMFRB and MISNet. However, it has several real-life applications such as printers and scanners. Second, the super resolution network MISNet is the lightest network proposed in this chapter by employing 60K parameters. Hence, it is extremely useful in situations that require very high speed super resolution. Third, the super resolution SRNMFRB is the best network that provides a trade-off between network performance and complexity. This network provides an acceptable performance by employing less than 1M parameters. Fourth, the super resolution networks PHMNet and MGHCNet are able to provide high super resolution performance by employing around 1.5M parameters. Hence, these two networks are examples of deep light-weigh high-performance convolutional networks for the task of image super resolution.

3.9 Conclusion

In this chapter, several image super resolution networks based on the idea of multi-scale feature generation have been proposed. Various multi-scale feature generation techniques, such as the inverse sub-pixel convolution operations, multi-scale convolutions and dilated convolutions, have been employed for designing deep multi-scale feature generation networks. It can be concluded from the experimental results obtained in this chapter that the idea of generating features at multiple scales is indeed helpful in improving the super resolution performance.

Chapter 4

Deep Image Super Resolution Networks with Guided Feature Generation

4.1 Introduction

Design of a residual block that provides a rich set of features while requiring only small numbers of parameters and operations is crucial for the task of single image super resolution. This is especially important in applications with limited power and storage capacity. In this chapter, we propose various residual blocks, that use the idea of guided feature generation, for producing rich sets of information for image super resolution [84], [87], [92], [96], [103], [105]. Specifically, we use three guided feature generation strategies, namely, edge extraction, spectral feature generation and morphological feature generation, for enhancing the performance of the light-weight image super resolution networks.

4.2 EFFRBNet: A Deep Super Resolution Network using Edge-assisted Feature Fusion Residual Blocks

Most of the light-weight super resolution schemes utilize the basic residual block of ResNet or its variants. The convolution operations in the residual block make it to learn the residual high frequency signal and the skip connection of the residual block facilitates the passage of the low frequency components of the input image. Using a specific module in the residual block that facilitates learning the high frequency residual signal, could further improve the network performance. In this section, a learnable nonlinear edge extraction module is developed to extract the edges of the input feature maps to the residual block, and therefore, makes the learning the high frequency components of the ground truth image easier.

In the overall architecture of the proposed super resolution network, first, the original low resolution image is passed through a convolution operation followed by a ReLU activation in order to extract the low resolution features. This convolution operation uses 64 filters each of spatial support of 3×3. Next, the low resolution feature maps thus obtained are fed to a sequence of 12 units of the proposed residual block, whose architecture is described in the following paragraphs. Then, the output of the last residual block is upscaled using a sub-pixel convolution operation [8] in order to restore its spatial resolution to that of the ground truth. The upscaled feature maps are passed through a convolution operation to construct the residual signal between the ground truth and the bilinear interpolated version of the low resolution image. This convolution operation uses 3 filters of spatial support of 3×3 each corresponding to one color channel. Finally, the residual signal thus obtained is added to the bilinear interpolated version of the low resolution image yielding an estimated high resolution image.

A basic residual block consists of two convolution operations interleaved by a ReLU activation function, and a skip connection between the input and output feature maps of the

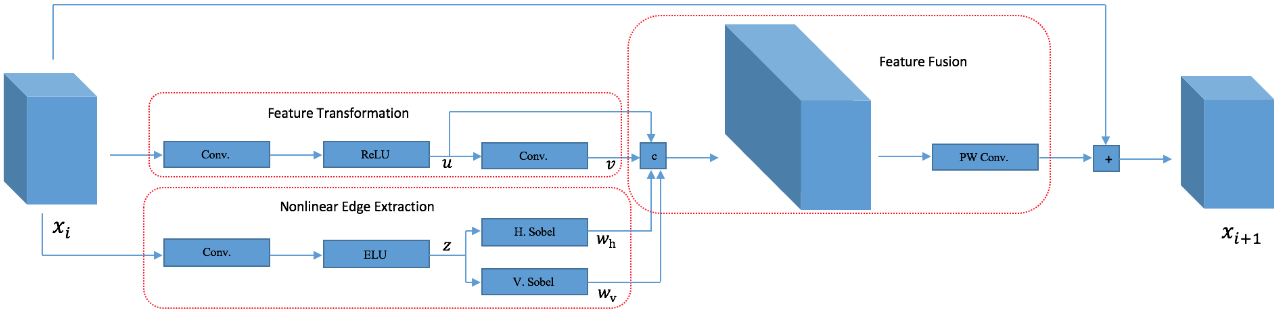


Figure 4.1: The architecture of the proposed residual block. Conv., H. Sobel and V. Sobel, respectively, represent the convolution operation, the horizontal Sobel operation and the vertical Sobel operation. The symbol **c** represents the concatenation operation, and PW Conv. denotes the point-wise convolution operation.

block. The residual signal between the input and output feature maps of the residual block consists mainly of high frequency components. To this high-frequency signal, the feature maps of the input to the block are added through the skip connection, resulting in the output of the block to have a more enhanced high frequency components. One can propose to introduce a nonlinear edge extraction in the residual block. The nonlinear edge extraction would extract the edges of the feature maps input to the block, and therefore, this results in generating the residual feature maps that have more enhanced high frequency components.

Fig. 4.1 shows the proposed residual block for the image super resolution problem. As seen from this figure, the proposed residual block consists of three modules, namely, feature transformation, nonlinear edge extraction and feature fusion modules. The input feature maps to the residual block first undergo a feature transformation in the first module, which consists of two convolution operations and a ReLU activation between them, yielding the first set of residual feature maps. In this step, the outputs of the two convolution operations, u and v, are concatenated to improve the representational capability of the block. Each of the two convolution operations uses 64 filters with the spatial support of 3×3. To facilitate generating high frequency residual feature maps, in the second module of the block, we carry out a nonlinear edge extraction, which is done by a convolution operation, an ELU activation and a pair of Sobel operations. This nonlinear edge extraction performed in this module, provides horizontal and vertical edge maps w_{h} and w_{v} having additional high frequency components for constructing the residual signal. The convolution operation used in this module employs 64 filters each of the spatial support of 3×3. Let $x[m,n]$ denote the two-dimensional signal representing a single feature map of z marked in Fig. 4.1. The horizontal and vertical edges obtained from the application of the corresponding two Sobel operators to $x[m,n]$, respectively, are given by

$$\begin{aligned} y_1[m,n] &= x[m-1,n+1] + 2x[m-1,n] + x[m-1,n-1] \\ &\quad -x[m+1,n+1] - 2x[m+1,n] - x[m+1,n-1] \\ y_2[m,n] &= x[m+1,n-1] + 2x[m,n-1] + x[m-1,n-1] \\ &\quad -x[m+1,n+1] - 2x[m,n+1] - x[m-1,n+1] \end{aligned} \tag{4.1}$$

The collection of all the $y_1[m,n]$ maps, each corresponding to a single feature map of z, produces the edge maps w_{h}. Similarly, the edge maps w_{v} are produced from the collection of the $y_2[m,n]$ maps. The edge maps w_{h} and w_{v} are concatenated to the feature maps u *and* v along the channel dimension, resulting in a very rich set of residual feature maps r with enhanced high frequency components. Next, the concatenated residual features r are fused using a point-wise convolution operation yielding the final set of residual feature maps of the block. The point-wise convolution operation used in the feature fusion module employs 64 filters each of the spatial support of 1×1. Finally, the residual feature maps thus obtained are added to the feature maps input to the block to construct the output feature maps. Use of the rich set of residual features resulting from each of residual blocks in the super resolution network improves its representational capability, and thus, enhances its performance.

The proposed residual block shown in Fig. 4.1 that employs the three modules is referred to as *edge-assisted feature fusion residual block (EFFRB)* and the proposed super resolution network using EFFRB as Super Resolution Network with **E**dge-assisted **F**eature **F**usion **R**esidual **B**lock (EFFRBNet) [96].

The proposed super resolution network (EFFRBNet) is trained using the 800 training images of DIV2K dataset [42]. The ground truth samples are constructed by extracting the sub-images of size 48×48 from the training images. The $\ell 1$ norm loss is used between the ground truth and estimated high resolution images. This loss function is optimized using the stochastic gradient descent (SGD) optimizer. The learning rate is initialized by a value of 0.1, which is decreased by a factor of 10 after each 182500 iterations. The parameters of the network are initialized by the method proposed in [7]. The mini-batch size is set to 64.

4.3 SRNSSI: A Deep Light-Weight Network for Single Image Super Resolution using Spatial and Spectral Information

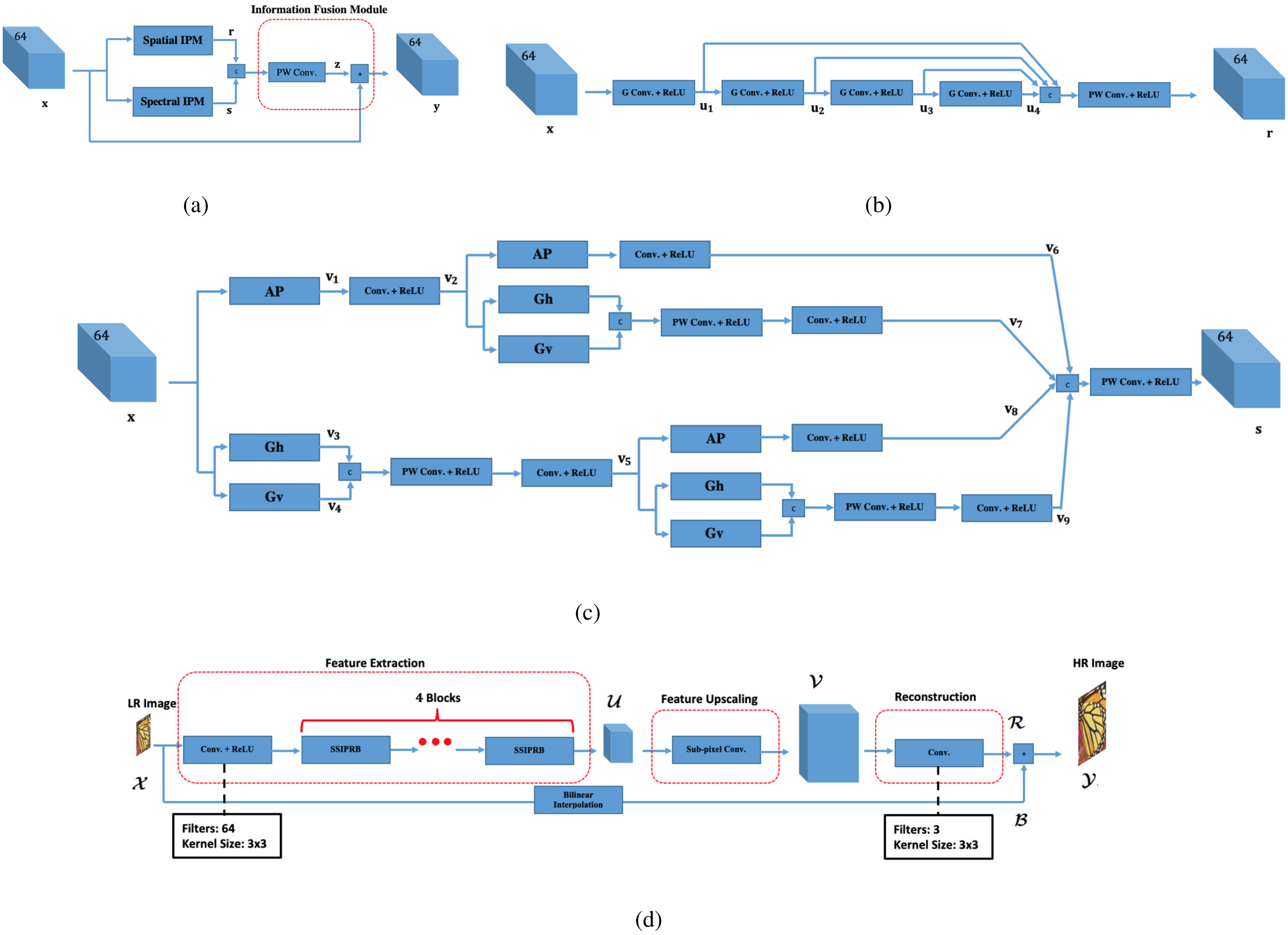
Most of the existing designs of the residual blocks for their use in image super resolution networks make use of only the spatial information contained in the input to the block. However, experimental studies in psychophysics have shown that visual information processing in human and mammalian visual systems is strongly dependent on the spatial frequency (spectral) content of the visual stimulus (input) to these systems [53]. There are couple of networks [106],[27] that generate and use spectral features for the task of image super resolution, but they ignore spatial features. Therefore, the residual blocks that use both the spatial and spectral contents of the input can be expected to provide a superior performance. The proposed residual block is designed to make use of both the spectral and the spatial information contained in the input to it. The proposed residual block consists of three modules. The first two modules produce feature maps, respectively, corresponding to the spatial and spectral information contents of the signal input to the block. The third module simply fuses the two types of feature maps produced by the first two modules. The residual block designed using this philosophy can be expected to provide a very rich set of features. Since the focus of the design of the residual blocks is on producing an enriched set of features, the network should be able to use only a small number of blocks, and hence, a small number of parameters in the network to achieve a desired accuracy.

We now develop a new residual block that produces a very rich set of feature maps by using both the spatial and spectral information that is present in the input to the block. Fig. 4.2 (a) shows a high-level architecture of the proposed residual block. It is seen from this figure that the input feature tensor **x** is processed simultaneously by two modules each to be designed to operate in a different domain. The first module is a spatial information processing module, whereas the second one is a spectral information processing module. The maps **r** and **s** resulting from these two modules are concatenatively fused using an information fusion module. The block is adapted to operate in a residual mode by using a skip connection between its input **x** and the output **y**.

Fig. 4.2 (b) shows the architecture of the spatial information processing module. In this module, the input feature tensor **x** consisting of 64 channels undergoes four group convolution operations producing, respectively, four feature tensors $\mathbf{u}_1$, $\mathbf{u}_2$, $\mathbf{u}_3$ and $\mathbf{u}_4$ as

$$\mathbf{u}_i = ReLU\big(W_i(\mathbf{u}_{i-1})\big) \quad i = 1, ..., 4 \quad and \quad \mathbf{u}_0 = \mathbf{x} \tag{4.2}$$

where each of the group convolution operations W_i uses two groups of filters, each employing 32 filters of kernel size $3 \times 3 \times 32$. Use of the convolution operations on groups of channels rather than on the entire set of input channels reduces the numbers of parameters and operations in the block considerably. Each of the feature tensors $\mathbf{u}_1$, $\mathbf{u}_2$, $\mathbf{u}_3$ and $\mathbf{u}_4$ is produced at a different hierarchical level of abstraction. Then, these feature tensors



(a)

(b)

(c)

(d)

Figure 4.2: Architecture of the proposed image super resolution network. (a) Proposed spatial and spectral information processing residual block. (b) Spatial information processing module. (c) Spectral information processing module. (d) Network overall architecture. IPM, Conv., G Conv. and PW Conv., respectively, denote information processing module, convolution operation, group convolution operation and point-wise convolution operation.

are concatenatively fused using a point-wise convolution operation in order to generate the module's output feature tensor **r** as

$$\mathbf{r} = ReLU\big(W_5(CONC(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4))\big) \tag{4.3}$$

where W_5 is a point-wise convolution operation using 64 filters each of kernel size $1 \times 1 \times (4 \times 64)$.

The strategy used in the design of the second module is to produce feature maps at different spectral decomposition levels of abstraction. Fig. 4.2 (c) shows the architecture of the spectral information processing module using two levels of spectral decomposition. In the first level of spectral decomposition, the feature tensor **x** input to the module undergoes an average pooling operation as

$$\mathbf{v}_1 = AP(\mathbf{x}) \tag{4.4}$$

where AP denotes the average pooling operation using a kernel size 2×2 and a stride of 1. Since the stride of the average pooling operation is unity, the spatial resolution of the feature tensor $\mathbf{v}_1$ is the same as that of the feature tensor **x**. Let $x_l[m,n]$ represent the two-dimensional signal of the $l-th$ channel of the feature tensor **x**. By applying the average pooling operation to the window of x_l located at $[m,n]$, a signal $k_l[m,n]$ is obtained as

$$k_l[m,n] = \sum_{i,j \in \mathcal{N}_{2\times 2}^{(m,n)}} x_l[i,j] \tag{4.5}$$

where $\mathcal{N}_{2\times 2}^{(m,n)}$ represents the set of indices of the pooling window located at (m,n), i.e., $\mathcal{N}_{2\times 2}^{(m,n)} = \{(m,n),(m+1,n),(m,n+1),(m+1,n+1)\}$. The spectral component $\mathbf{v}_1$ is further processed through the convolution operation given by

$$\mathbf{v}_2 = ReLU\big(W_6(\mathbf{v}_1)\big) \tag{4.6}$$

where W_6 is a convolution operation using 32 filters each of kernel size $3 \times 3 \times 64$. In order to produce the second spectral component of the first level of decomposition of the feature tensor **x**, the horizontal and vertical gradients of this tensor are obtained using the horizontal and vertical gradient operators G_h and G_v as

$$\begin{aligned} \mathbf{v}_3 &= G_h(\mathbf{x}) \\ \mathbf{v}_4 &= G_v(\mathbf{x}) \end{aligned} \tag{4.7}$$

The horizontal and vertical gradient operators G_h and G_v in (4.7) are applied to the signal $x_l[m, n]$ to produce its two gradient components as

$$\begin{aligned} p_l[m, n] &= x_l[m + 1, n] - x_l[m, n] \\ q_l[m, n] &= x_l[m, n + 1] - x_l[m, n] \end{aligned} \tag{4.8}$$

A concatenation of the above two gradients is made to undergo a point-wise convolution operation followed a regular convolution operation producing the second spectral component $\mathbf{v}_5$ of **x** as

$$\mathbf{v}_5 = ReLU\big(W_8(ReLU(W_7(CONC(\mathbf{v}_3, \mathbf{v}_4))))\big) \tag{4.9}$$

where W_7 represents a point-wise convolution operation employing 32 filters each of kernel size $1 \times 1 \times 128$ and W_8 represents a convolution operation using 32 filters each of kernel size $3 \times 3 \times 32$. It should be noted that the feature tensors $\mathbf{v}_2$ and $\mathbf{v}_5$ represent, respectively, the low and high frequency components of the input feature tensor **x** after its first level of spectral decomposition. A second level of spectral decomposition can be carried out by repeating the process of the first level on the feature tensors $\mathbf{v}_2$ and $\mathbf{v}_5$ individually. This process can be further continued to higher levels of decomposition. Fig. 4.2 (c) depicts the case of having only two levels of spectral decomposition. In this case after the second level of spectral decomposition, we have the feature tensors $\mathbf{v}_6$, $\mathbf{v}_7$, $\mathbf{v}_8$ and

$\mathbf{v}_9$ representing, respectively, the low-low, low-high, high-low and high-high frequency components of the feature tensor **x**. Finally, in the second (spectral) module, these four components are concatenatively fused to produce the module's output **s** as

$$\mathbf{s} = ReLU\big(W_9(CONC(\mathbf{v}_6, \mathbf{v}_7, \mathbf{v}_8, \mathbf{v}_9))\big) \tag{4.10}$$

where W_9 represents a point-wise convolution operation using 64 filters each of kernel size $1 \times 1 \times (4 \times 32)$.

As seen from Fig. 4.2 (a), the feature tensors **r** and **s** obtained from the spatial and spectral information processing modules are concatenatively fused in the information fusion module using a point-wise convolution operation in order to generate the residual feature tensor **z** as

$$\mathbf{z} = W_{10}(CONC(\mathbf{r}, \mathbf{s})) \tag{4.11}$$

where W_{10} represents a point-wise convolution operation using 64 filters each of kernel size $1 \times 1 \times 128$. Finally, the residual feature tensor **z** is added to the input feature tensor **x** to produce the block's output **y** as

$$\mathbf{y} = \mathbf{z} + \mathbf{x} \tag{4.12}$$

We refer to the proposed residual block of Fig. 4.2 (a) as *spatial and spectral information processing residual block* (SSIPRB).

The overall architecture of the super resolution network using the proposed residual block is shown in Fig. 4.2 (d). It is seen from this figure that this super resolution network consists of three stages, namely, *Feature Extraction*, *Upscaling* and *Reconstruction* stages. Let $\boldsymbol{\mathcal{X}}$ denote the original low resolution image input to the network. First, the input image $\boldsymbol{\mathcal{X}}$ is passed through the feature extraction stage consisting of a convolutional layer followed by a cascade of four units of the proposed residual block (SSIPRB), in order to produce the output feature tensor $\boldsymbol{\mathcal{U}}$ of the feature extraction stage as

$$\boldsymbol{\mathcal{U}} = Res_4\Big(Res_3\big(Res_2\big(Res_1\big(ReLU(W_{11}(\boldsymbol{\mathcal{X}}))\big)\big)\big)\Big) \tag{4.13}$$

where W_{11} is a convolution operation using 64 filters each of kernel size $3 \times 3 \times 3$ and Res_i ($i = 1, 2, 3, 4$) represents the operation of the $i - th$ residual block. The high-level feature tensor $\boldsymbol{\mathcal{U}}$ as produced by the feature extraction stage is made to undergo an upsampling operation using a sub-pixel convolution [8] in order to generate the feature tensor $\boldsymbol{\mathcal{V}}$ with a spatial resolution equal to that of the ground truth image. The upscaled feature tensor $\boldsymbol{\mathcal{V}}$ is then passed through the reconstruction stage consisting of a convolutional layer in order to produce the residual signal $\boldsymbol{\mathcal{R}}$ as

$$\boldsymbol{\mathcal{R}} = W_{12}(\boldsymbol{\mathcal{V}}) \tag{4.14}$$

where W_{12} is a convolution operation employing 3 filters, corresponding to the R, G and B components of the color image, each with spatial support of 3×3. Finally, the residual signal $\boldsymbol{\mathcal{R}}$ is added to the bilinear interpolated version $\boldsymbol{\mathcal{B}}$ of the low resolution input image in order to produce the network's output $\boldsymbol{\mathcal{Y}}$, which is the estimated high resolution image.

The proposed super resolution network shown in Fig. 4.2 (d) is referred to as *super resolution network using spatial and spectral information* (SRNSSI) [87].

We now provide an enhanced version of SRNSSI (ESRNSSI). This version uses 6 units of the proposed residual block instead of 4 units used by the original version. In addition, the features maps produced by each of the six residual blocks are concatenated and passed through a point-wise convolution operation $\big($employing 64 filters each with kernel size $1 \times 1 \times (6 \times 64)\big)$ and ReLU activation before performing feature upscaling. These modifications are aimed at making the feature extraction stage to provide features that are deeper and richer than that provided by SRNSSI. Despite the fact that the enhanced network is obtained at the expense of a slightly larger number of parameters, it is still in the category of light-weight deep networks.

For the training of the proposed super resolution networks, the DIV2K [42] image dataset is utilized. Sub-images of size 48×48 are extracted from the 800 training images of the DIV2K dataset to construct the training samples. The weights of the proposed networks are updated by optimizing the $\ell 1$ norm of the loss between the ground truth and the estimated high resolution images. The process of optimization of the $\ell 1$ norm loss is carried out by using the stochastic gradient descent (SGD) optimizer. The initial learning is carried out using a step size of 0.1. The learning rate is decreased by a factor of 10 after each 182500 iterations. The weights of all the convolution operations are initialized by the method of He et al. [7]. The mini-batch size is set to 64 in our experiments.

4.4 MorphoNet: a Deep Image Super Resolution Network using Hierarchical and Morphological Feature Generating Residual Blocks

The quality of an image is very much dependent on the texture and image representation of the image. Hence, the success of an image super resolution process can be judged from its capability in enhancing the textures and structures in the image super resolved by it. Morphological operations are the nonlinear mathematical operations that while performing signal processing aim at the textures and structures of the signal.

In this section, we present, for the first time, a deep image super resolution architecture in a residual framework by proposing a novel residual block that is capable of producing features of the image based on its morphology, as well as the conventional convolutional features [92]. The morphological features are learned by using the erosion and dilation operations and fused with the other hierarchical features to produce very rich set of features.

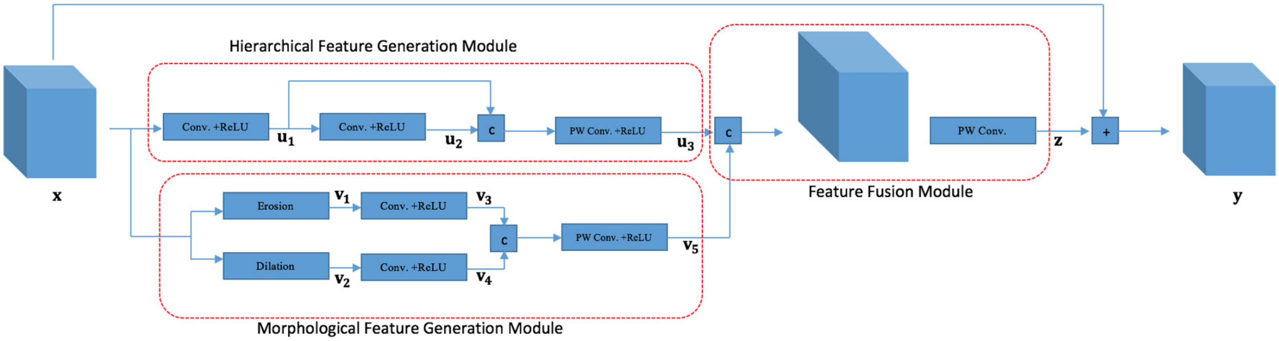


Figure 4.3: Architecture of the proposed residual block. Conv. and PW Conv., respectively, denote the convolution and point-wise convolution operations.

The proposed network consists of four parts, namely, a feature extraction part that generates features of the low resolution input image, a nonlinear mapping part that maps the low level features to hierarchically higher level features through a cascade of the new residual blocks and upsampling and image reconstruction parts that provide the estimated high resolution image. In the feature extraction part, the features of the input image are extracted using a convolution operation employing 64 filters each with kernel size 3×3 followed by a ReLU activation operation. The nonlinear mapping part is composed of a cascade of 11 residual blocks. The architecture of the proposed residual block is developed in the next paragraph The upsampling part uses a depth-to-space transpose operation [8] with a scaling factor equal to that of the super resolution scaling factor. Finally, the high resolution image is constructed using 3 convolutional filters each with kernel size 3×3 by the image reconstruction part. We refer the proposed super resolution network to as MorphoNet [92].

The proposed residual block is shown in Fig. 4.3. This block consists of three modules, a hierarchical feature generation module, a morphological feature generation module and a feature fusion module. In the hierarchical feature generation module, the feature tensor $\mathbf{x}$ input to the residual block is made to undergo a cascade of two convolution operations each followed by a ReLU activation operation yielding, respectively, two feature tensors $\mathbf{u}_1$

and $\mathbf{u}_2$ given by

$$\begin{aligned}\mathbf{u}_1 &= ReLU(W_1(\mathbf{x}))\\ \mathbf{u}_2 &= ReLU(W_2(\mathbf{u}_1))\end{aligned} \tag{4.15}$$

where each of the convolution operations W_1 and W_2 employs 64 filters with kernel size 3×3. The two feature tensors are then concatenatively fused as

$$\mathbf{u}_3 = ReLU(W_3(CONC(\mathbf{u}_1, \mathbf{u}_2))) \tag{4.16}$$

where $W3$ represents a point-wise convolution operations using 64 filters. In the hierarchical feature generation module, the features are learned solely though the convolution operation. In contrast, in the morphological feature generation module, features are also naturally learned but guided by morphological operations. In this module, the input feature tensor **x** first undergoes in parallel through the streams of the morphological erosion and dilation operations and then the resulting tensors $\mathbf{v}_1$ and $\mathbf{v}_2$ are convolved to produce morphologically guided features $\mathbf{v}_3$ and $\mathbf{v}_4$, respectively. Let $\mathbf{x}^k$ represent the kth channel of the feature tensor **x**. Then, the kth channel of the feature tensor $\mathbf{v}_1$ resulting from the erosion operation is given by

$$\mathbf{v}_1^k[m,n] = (\mathbf{x}^k \ominus \mathbf{b})[m,n] = \min_{(i,j)\in B} \mathbf{x}^k[m+i, n+j] \tag{4.17}$$

where **b** is the structuring element defined over a neighborhood B. Similarly, the kth channel of the feature tensor $\mathbf{v}_2$ resulting from, using the same structuring element **b** and defined over the same neighborhood B as for the erosion operation, is given by

$$\mathbf{v}_2^k[m,n] = (\mathbf{x}^k \oplus \mathbf{b})[m,n] = \max_{(i,j)\in B} \mathbf{x}^k[m+i, n+j] \tag{4.18}$$

The feature tenors $\mathbf{v}_3$ and $\mathbf{v}_4$ are then obtained, respectively, by applying convolution operations to the feature tensors $\mathbf{v}_1$ and $\mathbf{v}_2$ as

$$\begin{aligned} \mathbf{v}_3 &= ReLU(W_4(\mathbf{v}_1)) \\ \mathbf{v}_4 &= ReLU(W_5(\mathbf{v}_2)) \end{aligned} \tag{4.19}$$

where each of the convolution operations W_4 and W_5 uses 64 filters with kernel size 3×3. The two morphological feature tensors $\mathbf{v}_3$ and $\mathbf{v}_4$ are concatenatively fused to yield the feature tensor:

$$\mathbf{v}_5 = ReLU(W_6(CONC(\mathbf{v}_3, \mathbf{v}_4))) \tag{4.20}$$

where the point-wise convolution operation W_6 uses 64 filters. Next, the feature tensors $\mathbf{u}_3$ and $\mathbf{v}_5$ obtained, respectively, from the hierarchical and morphological feature generation modules, are fused to obtain the block's residual feature tensor given by

$$\mathbf{z} = W_7(CONC(\mathbf{u}_3, \mathbf{v}_5)) \tag{4.21}$$

where W_7 is a point-wise convolution operation using 64 filters. Finally, the feature tenor **x** input to the residual block is added to the residual feature tensor **z** to yield the output feature tensor **y** of the residual block.

For training the proposed super resolution network, sub-images of size 48×48 are extracted from the 800 training images of the *DIV2K* [42] dataset. The parameters of the proposed network are updated using the $\ell 1$ norm of the loss between the ground truth and estimated high resolution training samples. The $\ell 1$ norm loss is minimized using the stochastic gradient descent optimizer with the initial learning rate of 0.1. The batch size and weight decay parameter of the convolution operations are set to 64 and 10^{-4}, respectively.

4.5 SRNMSM: A Deep Light-weight Image Super Resolution Network using Multi-scale Spatial and Morphological Feature Generating Residual Blocks

A deep network with a capability of producing features corresponding to the textures and structures of a high quality high resolution image could be very beneficial for the task of image super resolution. As mentioned in the previous section, morphological operations are nonlinear operations that process signals aiming at their textures and structures. Hence, incorporating these mathematical operations in the design of a super resolution convolutional neural network could make such a design to provide a high quality super resolved image. In this section, a novel residual block with a capability of producing features corresponding to the textures and structures of high quality images by introducing in it the morphological operations of erosion, dilation, opening and closing is proposed and used in a residual convolutional network for the task of image super resolution. It is shown that in view of the idea of the morphological operations introduced in the design of the residual block, the super resolution performance of the network is significantly improved.

Fig. 4.4 shows the architecture of the proposed residual block. As seen from this figure, the proposed block consists of three modules, a multi-scale spatial feature generation module, a morphological feature generation module, and a feature fusion module. The input feature tensor **x** is simultaneously fed to the two feature generation modules. In the multi-scale spatial feature generation module, the feature tensor **x** undergoes the operations of convolution and dilated convolution in parallel producing, respectively, the feature tensors $\mathbf{u}_1$ and $\mathbf{u}_2$, as given by

$$\begin{aligned}\mathbf{u}_1 &= W_1(\mathbf{x})\\ \mathbf{u}_2 &= W_2(\mathbf{x})\end{aligned} \tag{4.22}$$

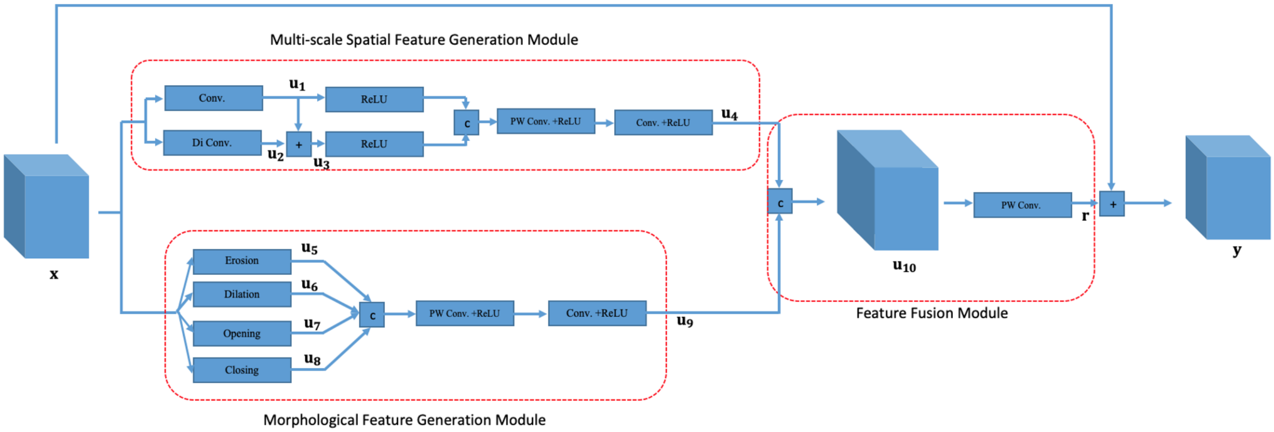


Figure 4.4: Architecture of the proposed residual block. Conv., Di Conv. and PW Conv., respectively, denote the convolution, dilated convolution and point-wise convolution operations.

where both the convolution operation W_1 and the dilated convolution operation W_2, employ 32 filters each with kernel size 3×3 and the dilation rate in W_2 is 2. Thus, the feature tensors $\mathbf{u}_1$ and $\mathbf{u}_2$ are obtained at two different scales. However, the use of the dilated convolution operation in the multi-scale spatial feature generation module produces the gridding artifacts [107]. In order to diminish the effect of these artifacts, the feature tensor $\mathbf{u}_1$, which is free of gridding artifacts, is added to the feature tensor $\mathbf{u}_2$ producing the feature tensor $\mathbf{u}_3$. The feature tensors $\mathbf{u}_1$ and $\mathbf{u}_3$ after passing them through ReLU activations are concatenatively fused and the resulting feature tensor is made to undergo a cascade of point-wise convolution operation and a convolution operation, each followed by a ReLU activation, to yield the feature tensor $\mathbf{u}_4$ given by

$$\mathbf{u}_4 = W_3(CONCAT(ReLU(\mathbf{u}_1), ReLU(\mathbf{u}_3))) \tag{4.23}$$

where W_3 represents a cascade of two convolution operations each using 64 filters of kernel sizes 1×1 and 3×3, respectively.

In the morphological feature generation module, the feature tensor **x** undergoes a parallel of four morphological operations, namely, erosion, dilation, opening and closing, producing, respectively, the feature tensors $\mathbf{u}_5$, $\mathbf{u}_6$, $\mathbf{u}_7$ and $\mathbf{u}_8$. The kth channel of the feature tensor $\mathbf{u}_5$ resulting from the erosion operation is given by

$$\mathbf{u}_5^k[m,n] = (\mathbf{x}^k \ominus \mathbf{b})[m,n] = \min_{(i,j)\in B} \mathbf{x}^k[m+i, n+j] \tag{4.24}$$

where $\mathbf{x}^k[m,n]$ denotes the two-dimensional signal representing the kth channel of the feature tensor **x** at the pixel position (m,n) and **b** is the structuring element defined over a neighborhood B of size 2×2 around (m,n). Similarly, the kth channel of the feature tensor $\mathbf{u}_6$ resulting from the dilation operation, using the same structuring element **b** and defined over the same neighborhood B, is given by

$$\mathbf{u}_6^k[m,n] = (\mathbf{x}^k \oplus \mathbf{b})[m,n] = \max_{(i,j)\in B} \mathbf{x}^k[m+i, n+j] \tag{4.25}$$

The kth channels of the feature tensors $\mathbf{u}_7$ and $\mathbf{u}_8$ resulting from the opening and closing operations using the same structuring element **b** are, respectively, given by

$$\begin{aligned} \mathbf{u}_7^k[m,n] &= \big((\mathbf{x}^k \ominus \mathbf{b}) \oplus \mathbf{b}\big)[m,n] \\ \mathbf{u}_8^k[m,n] &= \big((\mathbf{x}^k \oplus \mathbf{b}) \ominus \mathbf{b}\big)[m,n] \end{aligned} \tag{4.26}$$

The four morphological feature tensors $\mathbf{u}_5$, $\mathbf{u}_6$, $\mathbf{u}_7$ and $\mathbf{u}_8$ are then concatenatively fused and the resulting feature tensor is made to undergo a cascade of point-wise convolution operation and a convolution operation, each followed by a ReLU activation, to yield the feature tensor $\mathbf{u}_9$ given by

$$\mathbf{u}_9 = W_4(CONCAT(\mathbf{u}_5, \mathbf{u}_6, \mathbf{u}_7, \mathbf{u}_8)) \tag{4.27}$$

where W_4 represents a cascade of the two convolution operations carried out using 64 filters of kernel sizes 1×1 and 3×3, respectively.

In the feature fusion module, the feature tensors $\mathbf{u}_4$ and $\mathbf{u}_9$ obtained from the two feature generation modules are concatenated and the resulting feature tensor $\mathbf{u}_{10}$ is made to undergo a point-wise convolution operation producing a rich set of residual feature maps of the block as given by

$$\mathbf{r} = W_5(CONCAT(\mathbf{u}_4, \mathbf{u}_9)) \tag{4.28}$$

where the operation W_5 represents a point-wise convolution operation employing 64 filters. Finally, the block's residual feature tensor **r** is added to its input feature tensor **x** in order to produce the block's output **y** as

$$\mathbf{y} = \mathbf{x} + \mathbf{r} \tag{4.29}$$

In the overall architecture of the proposed super resolution network, first the low resolution input image is converted into its feature maps by a feature extraction module consisting of a convolutional layer using 64 filters each with kernel size 3×3 and a ReLU activation. The resulting feature maps are then passed through a nonlinear mapping module formed by a cascade of 5 units of the residual blocks in Fig. 4.4 to yield the high level feature maps. Next, the resolutions of the high level feature maps are restored to that of the ground truth image by employing an upscaling module that consists of a sub-pixel convolutional layer. Finally, the feature maps that are output from the upscaling module are passed through a reconstruction module to obtain the image **R**, which is the residue between the ground truth image and the bilinearly interpolated version **B** of the low resolution image. The reconstruction module consists of a convolutional layer using 3 filters with kernel size 3×3.

We refer to the proposed image super resolution network as **S**uper **R**esolution **N**etwork using **M**ulti-scale **S**patial and **M**orphological feature generating residual blocks (SRN-MSM) [84].

In order to train our convolutional neural network, we use the images of the *DIV2K* [42] dataset. The samples of the training set are formed by extracting sub-images of size 48×48 from the 800 images of this dataset. The mean absolute error ($\ell 1$ norm loss) is used as the loss function between the ground truth samples and the estimated high resolution samples, in order to update the parameters of the network. The optimization process is carried using the stochastic gradient descent (SGD) technique. The learning process is started with the step size of 0.1 and decreased by a factor of 10 after each $182,500$ iterations. A value of 10^{-4} is assigned to the weight decay parameters for carrying out the convolution operations. The method of [7] is used for initializing the parameters of convolution operations. A value of 64 is chosen as the batch size.

The proposed SRNMSM is implemented using Keras library [40] and TensorFlow package [41]. The proposed SRNMSM is trained using a machine with Intel Core i9 CPU @3.3 GHz, 32-GB RAM and Nvidia GeForce GTX 1080 GPU.

4.6 Experimental Results

4.6.1 Experimental Results of EFFRBNet

We now study the impact of each of the two individual modules of the proposed residual block, namely, the feature transformation module and the nonlinear edge extraction module, on the network performance. Two variants of the proposed residual block, *Variant 1* and *Variant 2*, are formed by using, respectively, the feature transformation module or the nonlinear edge extraction module.

The network performance results using the proposed residual block and its two variants are given in Table 4.1. It is seen from this table that the network performance degrades, when only one of the modules is used. However, the degradation in performance over that using the proposed module is much severe in the absence of the feature transformation

Table 4.1: Results on the ablation study of the proposed residual block of EFFRBNet.

Network with	*Set5*	*Set14*	*BSD100*
Variant 1	34.19 (0.9260)	30.29 (0.8419)	29.07 (0.8058)
Variant 2	32.41 (0.9001)	29.00 (0.8163)	28.32 (0.7877)
EFFRB	34.31 (0.9270)	30.39 (0.8435)	29.10 (0.8066)

Table 4.2: PSNR (SSIM) values resulting from applying EFFRBNet and various state-of-the-art schemes to images of four benchmark datasets.

Dataset	Scaling	Bicubic	SRCNN [1]	VDSR [3]	DRCN [19]	LapSRN [29]	DRRN [20]	MemNet [4]	IDN [45]	SRFBN [25]	CARN [14]	EFFRBNet (Proposed) [96]
Set5	×2	33.66 (0.9299)	36.66 (0.9542)	37.53 (0.9587)	37.63 (0.9588)	37.52 (0.959)	37.74(0.9591)	37.78 (0.9597)	37.83 (0.9600)	37.78 (0.9597)	37.76 (0.9590)	38.00 (0.9612)
	×3	30.39 (0.8682)	32.75 (0.9090)	33.66 (0.9213)	33.82 (0.9226)	N/A	34.03 (0.9244)	34.09 (0.9248)	34.11 (0.9253)	34.20 (0.9255)	34.29 (0.9255)	34.31 (0.9270)
	×4	28.42 (0.8104)	30.48 (0.8628)	31.35 (0.8838)	31.53 (0.8854)	31.54 (0.885)	31.68 (0.8888)	31.74 (0.8893)	31.82 (0.8903)	31.98 (0.8923)	32.13 (0.8937)	32.02 (0.8928)
Set14	×2	30.24 (0.8688)	32.42 (0.9063)	33.03 (0.9124)	33.04 (0.9118)	33.08 (0.913)	33.23 (0.9136)	33.28 (0.9142)	33.30 (0.9148)	33.35 (0.9156)	33.52 (0.9166)	33.67 (0.9187)
	×3	27.21(0.7385)	29.28 (0.8209)	29.77 (0.8314)	29.76 (0.8311)	N/A	29.96 (0.8349)	30.00 (0.8350)	29.99 (0.8354)	30.10 (0.8372)	30.29 (0.8407)	30.39 (0.8435)
	×4	26.00 (0.7027)	27.49 (0.7503)	28.01 (0.7674)	28.02 (0.7670)	28.19 (0.772)	28.21 (0.7721)	28.26 (0.7723)	28.25 (0.7730)	28.45 (0.7779)	28.60 (0.7806)	28.61 (0.7824)
BSD100	×2	29.56 (0.8431)	31.36 (0.8879)	31.90 (0.8960)	31.85 (0.8942)	31.80 (0.895)	32.05 (0.8973)	32.08 (0.8978)	32.08 (0.8985)	32.00 (0.8970)	32.09 (0.8978)	32.23 (0.9012)
	×3	27.21 (0.7385)	28.41 (0.7863)	28.82 (0.7976)	28.80 (0.7963)	N/A	28.95 (0.8004)	28.96(0.8001)	28.95 (0.8013)	28.96 (0.8010)	29.06 (0.8034)	29.10 (0.8066)
	×4	25.96 (0.6675)	26.90 (0.7101)	27.29 (0.7251)	27.23 (0.7233)	27.32 (0.728)	27.38 (0.7284)	27.40 (0.7281)	27.41 (0.7297)	27.44 (0.7313)	27.58 (0.7349)	27.54 (0.7364)
Urban100	×2	26.88 (0.8403)	29.50 (0.8946)	30.76 (0.9140)	30.75 (0.9133)	30.41 (0.910)	31.23 (0.9188)	31.31 (0.9195)	31.27 (0.9196)	31.41 (0.9207)	31.92 (0.9256)	31.79 (0.9261)
	×3	24.46 (0.7349)	26.24 (0.7989)	27.14 (0.8279)	27.15 (0.8276)	N/A	27.53 (0.8378)	27.56 (0.8376)	27.42 (0.8359)	27.66 (0.8415)	28.06 (0.8493)	27.83 (0.8473)
	×4	23.14 (0.6577)	24.52 (0.7221)	25.18 (0.7524)	25.14 (0.7510)	25.21 (0.756)	25.44 (0.7638)	25.50 (0.7630)	25.41 (0.7632)	25.71 (0.7719)	26.07 (0.7837)	25.73 (0.7761)

The values in the red font indicate the best performance and those in the blue font represent the second best performance.

Table 4.3: Complexity of EFFRBNet and various super resolution schemes.

Method	Number of Parameters
SRCNN [1]	57K
VDSR [3]	665K
DRCN [19]	1770K
DRRN [20]	297K
MemNet [4]	677K
IDN [45]	553K
SRFBN-S [25]	483K
CARN [14]	1592K
EFFRBNet (Proposed)	1499K

module. This is not surprising, since in the absence of the feature transformation module, the network simply degenerates into an edge extraction network, that is, it is no longer a super resolution network. The performance of the proposed and nine of the state-of-the-art light-weight super resolution schemes on four benchmark datasets, namely, *Set 5* [21], *Set 14* [22], *BSD 100* [23] and *Urban 100* [24], with various scaling factors is given in Table 4.2. It is seen that the proposed super resolution scheme and CARN provide the best

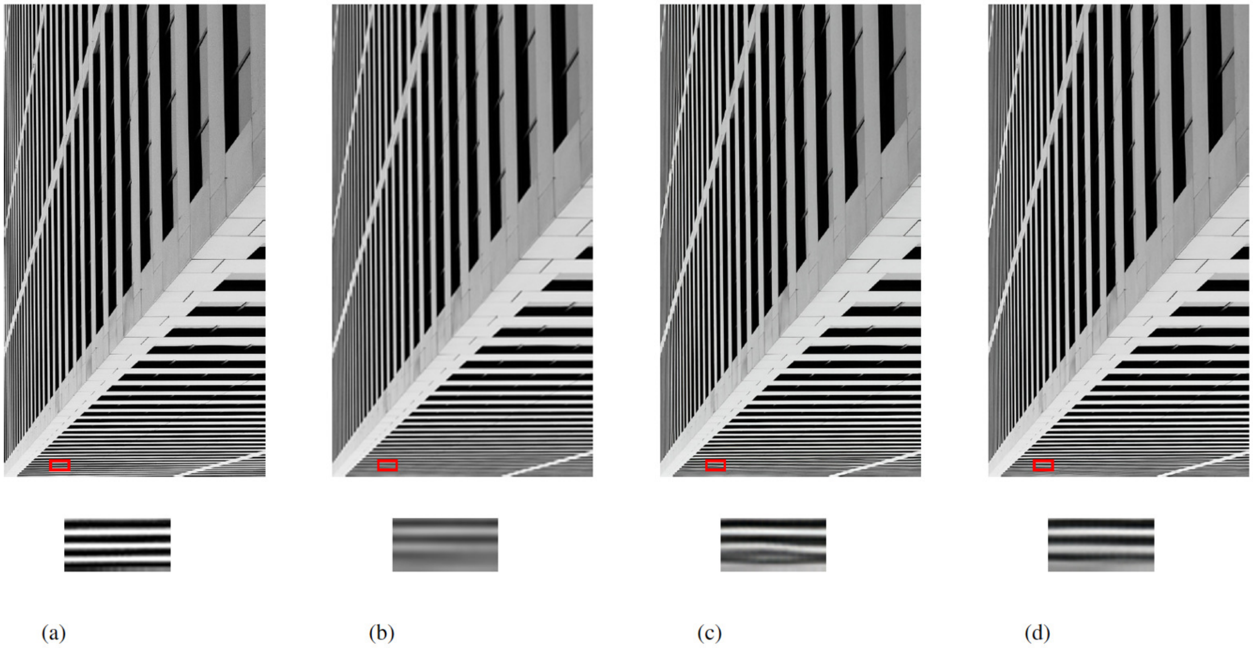


Figure 4.5: Visual quality of *img011* images super resolved by EFFRBNet and CARN with upscaling factor 3. (a) Ground truth image, Images super resolved by (b) Bicubic Interpolation (c) CARN. (d) EFFRBNet.

values for both the PSNR and SSIM metrics in 16 and 8 cases, respectively. In this respect, the performance of the proposed super resolution network is superior to that of CARN.

The number of parameters of each of the light-weight super resolution schemes used for comparison is given in Table 4.3. It is seen from this table that the proposed network employs around 100K less number of parameters than CARN does, which has the second best performance. Thus, based on the results shown in Table 4.2 and Table 4.3, the proposed EFFRBNet outperforms all the state-of-the-art light-weight super resolution schemes, when both the performance and complexity of the networks are taken into consideration.

Fig. 4.5 shows the visual quality of the *img011* images from the *Urban100* dataset super resolved by CARN and the proposed EFFRBNet with the scaling factor of 3. As seen from this figure, some of the edges resulting from the application of CARN are distorted and are much different from those of the original image. On the other hand, the edges resulting from the application of EFFRBNet have a better similarity to those of the ground truth.

4.6.2 Experimental Results of SRNSSI

In this section, we first perform a number of experiments on SRNSSI in order to show the effectiveness of the proposed residual block for the problem of single image super resolution. In this regard, we carry out experiments related to the design of the spectral information processing module and usefulness of the various ideas used in the design of the spatial information processing module. Next, the performance and complexity of the proposed super resolution networks, SRNSSI and ESRNSSI, are presented and compared with that of the light-weight state-of-the-art schemes for image super resolution existing in the literature. Finally, we evaluate the performance of an ultralight-weight version of the proposed super resolution network by using only one residual block.

As mentioned in Section 4.3, the objective of the spectral information processing module of the proposed residual block is to generate spectral features at different levels of decomposition. We first illustrate the ability of this module in generating the spectral features corresponding to the approximate and detail subbands of the feature tensor that is input to the module. Fig. 4.6 shows the spectral features after the first level of decomposition corresponding to the approximate and detail subbands in four selected feature maps obtained when the *Baby* image from the *Set5* dataset [21] is input to the block. The maps shown in the first row of this figure are the spectral features corresponding to the approximate subbands, whereas those in the second row correspond to the detail subbands. Fig. 4.7 shows one selected spectral feature map corresponding to each of the low-low, low-high, high-low and high-high frequency subbands when the *Baby* image is input to the block after the feature maps resulting from the first level of decomposition undergo the second level of decomposition. An examination of the feature maps illustrated in Figs. 4.6 and 4.7 shows

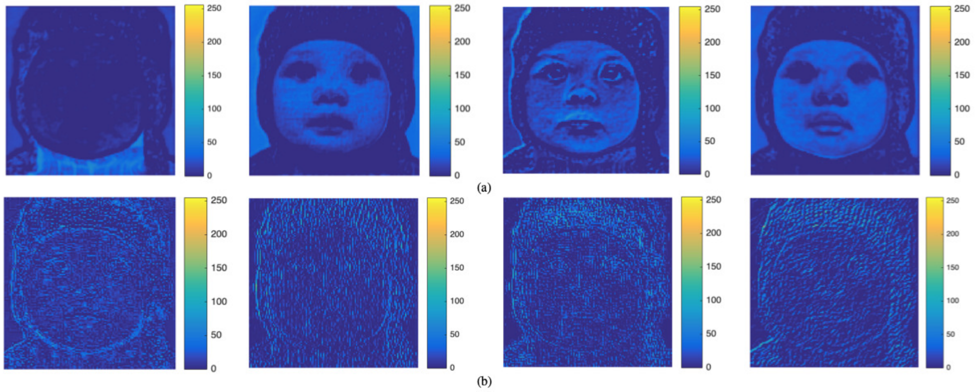


Figure 4.6: Spectral features corresponding to approximate and detail subbands of the *Baby* image obtained from the spectral information processing module using one level of decomposition.(a) Spectral features corresponding to the approximate subbands. (b) Spectral features corresponding to detail subbands.

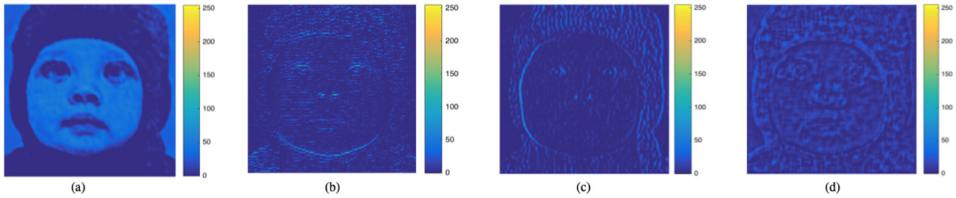


Figure 4.7: Spectral features corresponding to different subbands of the *Baby* image obtained from the spectral information processing module using two levels of decomposition. Spectral feature corresponding to the (a) low-low subband, (b) low-high subband, (c) high-low subband, (d) high-high subband.

that the spectral information processing module successfully decomposes its input into different subbands of a decomposition level and extracts spectral features and combines them in order to provide a very rich set of feature maps.

In order to investigate the impact of the spectral information processing module on the network performance and complexity, we remove this module from the residual blocks of the SRNSSI network. For this study, images from *Set5* [21], *Set14* [22], *BSD100* [23] and *Urban100* [24] datasets with the scaling factor 4 are input to the resulting network and the results in terms of PSNR are given in Table 4.4 and compared with that of the original SRNSSI network using the residual blocks with only one spectral level of decomposition. It is seen from this table that without using spectral features, the network performance deteriorates. In order to examine whether the performance deterioration of the network results from the removal of the proposed spectral information processing module or from the reduction of the number of parameters resulting from this removal, we increase the number of convolutional filters in the spatial information processing module from 32 to 40. With this change in the residual block, the number of parameters in the network remains comparable to that of the network using the original residual block. The performance of the resulting network is given in the second row of Table 4.4. By comparing the results given in the second and fourth rows of this table, it is quite clear that it is the proposed spectral information processing module that is responsible for improving the network performance.

We now form another variant of the proposed residual block, in which we maintain the structure of the residual block but replace each of the average pooling (AP), horizontal gradient (G_h) and vertical gradient (G_v) operators by a learnable depth-wise convolutional layer employing 64 filters with kernel size 3×3. We refer to this variant of the residual block as *SSIPRB with Learnable Conv*. It should be noted that the AP, G_h and G_v operators do not employ any trainable parameter, whereas their replacements by depth-wise convolutional layers do. As a result, the number of parameters employed by the network using the *SSIPRB with Learnable Conv* is somewhat larger than that using original SSIPRB. The performance of the proposed SRNSSI and the one using *SSIPRB with Learnable Conv* on the four benchmark datasets with scaling factor 4 are given in Table 4.4. It is seen from the results of this table that the performance of the network using *SSIPRB with Learnable Conv* is inferior to that of the proposed SRNSSI, indicating that directional decompositions have a direct impact in generating a richer set of features.

Next, we perform an experiment to study the impact of the number of decomposi-

Table 4.4: Impact of the Spectral Information Processing Module on the Network Performance and Complexity.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*	Params
w/o Module	32.02	28.64	27.56	25.75	373K
w/o Module (More Filters)	32.03	28.66	27.57	25.77	579K
with Module (Learnable Conv.)	32.04	28.66	27.57	25.78	561K
with Module	32.06	28.68	27.58	25.80	553K

Table 4.5: Impact of the Number of Spectral Decomposition Levels on the Network Performance and Complexity.

Number of Levels	*Set5*	*Set14*	*BSD100*	*Urban100*	Parameters
1	32.06	28.68	27.58	25.80	553K
2	32.09	28.70	27.59	25.85	737K

Table 4.6: Impact of Haar Spectral Information Processing Module on the Network Performance and Complexity.

Network with	*Set5*	*Set14*	*BSD100*	*Urban100*	Parameters
SSIPRB-Haar (2 level)	32.05	28.67	27.58	25.83	742K
SSIPRB (2 level)	32.09	28.70	27.59	25.85	737K

Table 4.7: Impact of Group Convolutions on the Network Performance and Complexity.

Number of Groups	*Set5*	*Set14*	*BSD100*	*Urban100*	Parameters
8	31.98	28.62	27.54	25.70	332K
4	32.04	28.64	27.56	25.77	406K
2	32.06	28.68	27.58	25.80	553K
1	32.13	28.70	27.60	25.86	848K

Table 4.8: Impact of Fusing Hierarchical Features on the Network Performance and Complexity.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*	Parameters
W/O Fusing	32.01	28.66	27.57	25.77	486K
With Fusing	32.06	28.68	27.58	25.80	553K

tion levels used in the spectral information processing module of the residual blocks on the network performance and complexity. Table 4.5 presents the PSNR values and the number of parameters of the network using one and two levels of spectral decomposition, when the images from the four benchmark datasets with the scaling factor 4 are input to the network. It is seen from the results of this table that increasing the levels of spectral decomposition from one to two helps in improving the performance of the network. However, this improvement results in increasing the number of parameters by 33%. Since our goal is to have a light-weight high-performance super resolution network, we set the default value of the number of spectral decomposition levels as 1 in all of our experiments. It is seen from the table that even when the network blocks use only one level of spectral decomposition, it still provides a very good performance by employing just 553K parameters.

In the spectral information processing module of the proposed residual block, the high-frequency components are extracted by employing the horizontal and vertical gradient operators in parallel, followed by concatenation, point-wise convolution and another convolution with kernel size 3×3. Therefore, the directional high-frequency components of the input feature tensor are first fused to generate a rich set of high-frequency feature maps and then processed. One could instead use two-dimensional Haar wavelet filters to generate the spectral features. In this case, the four 2×2 Haar wavelet filters, namely, low-low, low-high, high-low and high-high Haar filters, each followed by a convolution operation employing 64 filters with kernel size 3×3, can be applied to the input feature tensor. Then, the resulting four feature maps can be concatenated and passed through a point-wise convolution operation employing 64 filters to produce the spectral feature maps. The performance and complexity of the proposed SRNSSI that employs SSIPRB with two levels of spectral decomposition and that of the network employing the SSIPRB with the Haar spectral information processing module on the images with scaling factor 4 are given in Table 4.6. It is seen from the results of this table that the new approach of generating the spectral features using Haar wavelet filters deteriorates the network performance even if the resulting network uses slightly larger number of parameters. The main difference between the proposed spectral information processing module and the spectral information processing module using Haar filters is that in the former the spectral features with high-frequency components are fused and processed multiple times as compared to only one

Table 4.9: Performance (in terms of PSNR and SSIM) and Complexity of the Proposed SRNSSI and Various State-of-the-art Light-weight Image Super Resolution Networks.

Dataset	Scaling	SRCNN [1]	SCN [2]	DRRN [20]	DRCN [19]	LapSRN [29]	CARN-M [14]	SRFBN-S [25]	s-LWSR_32 [15]	LatticeNet [16]	SRNSSI	ESRNSSI
Set5	×2	36.66 (0.9542)	36.93 (0.9252)	37.74(0.9591)	37.63 (0.9588)	37.52 (0.959)	37.53 (0.9583)	37.78 (0.9597)	N/A	38.15 (0.9610)	37.95 (0.9610)	38.17 (0.9618)
	×3	32.75 (0.9090)	33.10 (0.9136)	34.03 (0.9244)	33.82 (0.9226)	N/A	33.99 (0.9236)	34.20 (0.9255)	N/A	34.53 (0.9281)	34.33 (0.9272)	34.56 (0.9290)
	×4	30.48 (0.8628)	30.86 (0.8710)	31.68 (0.8888)	31.53 (0.8854)	31.54 (0.885)	31.92 (0.8903)	31.98 (0.8923)	32.04 (0.893)	32.30 (0.8962)	32.06 (0.8938)	32.34 (0.8969)
Set14	×2	32.42 (0.9063)	32.56 (0.9069)	33.23 (0.9136)	33.04 (0.9118)	33,08 (0.913)	33.26 (0.9141)	33.35 (0.9156)	N/A	33.78 (0.9193)	33.58 (0.9181)	33.92 (0.9205)
	×3	29.28 (0.8209)	29.41 (0.8235)	29.96 (0.8349)	29.76 (0.8311)	N/A	30.08 (0.8367)	30.10 (0.8372)	N/A	30.39 (0.8424)	30.41 (0.8439)	30.57 (0.8467)
	×4	27.49 (0.7503)	27.64 (0.7578)	28.21 (0.7721)	28.02 (0.7670)	28.19 (0.772)	28.42 (0.7762)	28.45 (0.7779)	28.15 (0.776)	28.68 (0.7830)	28.68 (0.7845)	28.83 (0.7883)
BSD100	×2	31.36 (0.8879)	31.40 (0.8884)	32.05 (0.8973)	31.85 (0.8942)	31.80 (0.895)	31.92 (0.8960)	32.00 (0.8970)	N/A	32.25 (0.9005)	32.20 (0.9008)	32.38 (0.9028)
	×3	28.41 (0.7863)	28.50 (0.7885)	28.95 (0.8004)	28.80 (0.7963)	N/A	28.91 (0.8000)	28.96 (0.8010)	N/A	29.15 (0.8059)	29.12 (0.8071)	29.23 (0.8097)
	×4	26.90 (0.7101)	27.03 (0.7161)	27.38 (0.7284)	27.23 (0.7233)	27.32 (0.728)	27.44 (0.7304)	27.44 (0.7313)	27.52 (0.734)	27.62 (0.7367)	27.58 (0.7382)	27.68 (0.7410)
Urban100	×2	29.50 (0.8946)	29.52 (0.8970)	31.23 (0.9188)	30.75 (0.9133)	30.41 (0.910)	31.23 (0.9193)	31.41 (0.9207)	N/A	32.43 (0.9302)	31.67 (0.9244)	32.24 (0.9303)
	×3	26.24 (0.7989)	26.21 (0.8010)	27.53 (0.8378)	27.15 (0.8276)	N/A	27.55 (0.8385)	27.66 (0.8415)	N/A	28.33 (0.8538)	27.85 (0.8467)	28.24 (0.8557)
	×4	24.52 (0.7221)	24.52 (0.7260)	25.44 (0.7638)	25.14 (0.7510)	25.21 (0.756)	25.62 (0.7694)	25.71 (0.7719)	25.87 (0.779)	26.25 (0.7873)	25.80 (0.7777)	26.13 (0.7884)
Number of Parameters		57K	33K	297K	1774K	813K	412K	483K	571K	777K	553K	856K
Number of MACC Operations		52.7G	37.8G	6796.9G	17974.3G	149.4G	32.5G	1045.3G	32.9G	43.6G	31.1G	47.9G

The values in the red, blue and cyan fonts, respectively, indicate the best, second best and third best performance.

Table 4.10: Performance (in terms of PSNR and SSIM) and Complexity of the Proposed SRNSSI and Wavelet-based Super Resolution Convolutional Neural Networks.

Dataset	Scaling	MWCN [27]	Network of [106]	ESRNSSI
Set5	×2	37.91 (0.9600)	38.06 (0.9602)	38.17 (0.9618)
	×3	34.17 (0.9271)	34.45 (0.9272)	34.56 (0.9290)
	×4	32.12 (0.8941)	32.23 (0.8952)	32.34 (0.8969)
Set14	×2	33.70 (0.9182)	34.04 (0.9205)	33.92 (0.9205)
	×3	30.16 (0.8414)	30.56 (0.8450)	30.57 (0.8467)
	×4	28.41 (0.7816)	28.80 (0.7856)	28.83 (0.7883)
BSD100	×2	32.23 (0.8999)	32.26 (0.9006)	32.38 (0.9028)
	×3	29.12 (0.8060)	29.18 (0.8071)	29.23 (0.8097)
	×4	25.96 (0.6675)	27.62 (0.7355)	27.68 (0.7410)
Urban100	×2	32.30 (0.9296)	32.63 (0.9330)	32.24 (0.9303)
	×3	28.13 (0.8514)	28.50 (0.8587)	28.24 (0.8557)
	×4	26.27 (0.7890)	26.42 (0.7940)	26.13 (0.7884)
Number of Parameters		16.1M	16.6M	0.8M

time in the latter. Hence, the proposed spectral information processing module by utilizing the directional interdependencies between the high-frequency components of the input feature tensor is able to generate richer set of spectral features.

The spatial information processing module uses group convolution operations in order to keep the numbers of parameters and operations of the block low. In order to investigate

Table 4.11: Performance of the Ultralight-weight SRNSSI.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*	Parameters
SRCNN	30.48	27.49	26.90	24.52	57K
FSRCNN	30.71	27.59	26.98	24.62	13K
Ultralight-weight SRNSSI	31.26	28.21	27.23	25.04	39K

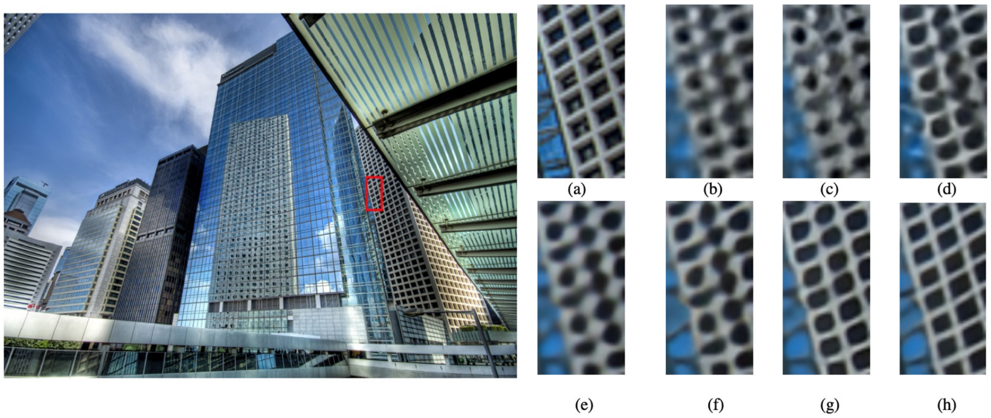


Figure 4.8: Visual quality of the *img061* images super resolved by light-weight networks. (a) Ground truth. (b) SRCNN [1]. (c) SCN [2]. (d) LapSRN [29]. (e) CARN-M [14]. (f) SRFBN-S [25]. (g) SRNSSI. (h) ESRNSSI.

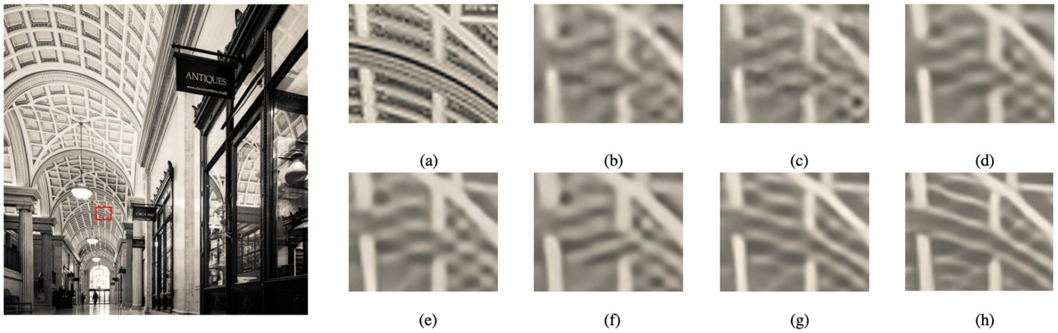


Figure 4.9: Visual quality of the *img083* images super resolved by light-weight networks. (a) Ground truth. (b) SRCNN [1]. (c) SCN [2]. (d) LapSRN [29]. (e) CARN-M [14]. (f) SRFBN-S [25]. (g) SRNSSI. (h) ESRNSSI.

the impact of the number of groups of convolutions used in each layer of the module on the network performance and complexity, we perform experiments by carrying out the operations of the layer by dividing the input channels into 1, 2, 4 and 8 groups. Table 4.7 gives

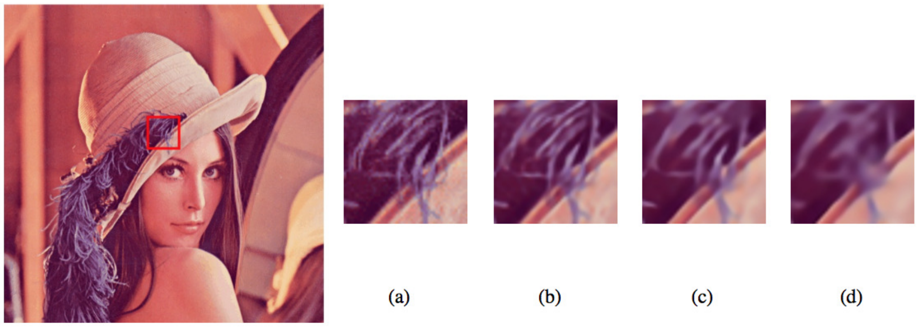


Figure 4.10: Visual quality of the *Lena* images super resolved by SRNSSI. (a) Ground truth. (b) Super resolved with scaling factor 2. (c) Super resolved with scaling factor 3. (d) Super resolved with scaling factor 4.

the PSNR values of the images of the four benchmark datasets super resolved by the network with the scaling factor 4, and the number of parameters required by it for each of the four cases of the group convolutions. It is seen from this table that the performance of the network is enhanced with increasing complexity as the number of group of convolutions is decreased. However, a close examination of the results in this table suggests the choice of a default value of 2 for the number of group of convolutions in the proposed block. This default value provides a good balance between the performance and the complexity of the network using this block.

In order to investigate the impact of fusing features from the different hierarchical levels in the spatial information processing module on the network performance and complexity, we remove all the skip connections from the module. Thus, the resulting module simply consists of the four convolutional layers. Table 4.8 gives the PSNR values and number of parameters of the network using the resulting module on the images of the four benchmark datasets with the scaling factor 4. It is seen from the results of this table that by not fusing the spatial features from different hierarchical levels, the performance of the network deteriorates by as much as 0.05dB while providing savings of only 70K in the number of network parameters.

We now evaluate the performance of the proposed network, SRNSSI, and its enhanced version, ESRNSSI. Both networks use only one level of spectral decomposition in the spectral information processing module and four layers of group convolutions in the spatial information processing module. The performance of the proposed super resolution networks is compared with those of nine light-weight state-of-the-art super resolution neural networks, namely, super resolution convolutional neural network (SRCNN) [1], sparse coding network (SCN) [2], deep recursive residual network (DRRN) [20], deep recursive convolutional network (DRCN) [19], Laplacian pyramid super resolution network (LapSRN) [29], cascaded residual network (CARN) [14], super resolution feedback network (SRFBN) [25], super light-weight super resolution network (s-LWSR) [15] and super resolution network using lattice blocks (LatticeNet) [16], on the images of the four benchmark datasets with scaling factors 2, 3 and 4. The results in terms of PSNR and SSIM for different scaling factors are given in Table 4.9. It is seen from this table that the proposed SRNSSI provides 8 second best and 14 third best values of PSNR and SSIM among all the state-of-the-art light-weight super resolution networks by employing 553K parameters and 31.1G MACC operations. These results of SRNSSI compares with those of LatticeNet that provides 3 best and 15 second best values of PSNR and SSIM by employing 777K parameters and 43.6G MACC operations. On the other hand, the proposed ESRNSSI provides 21 best and 3 second best values of PSNR and SSIM at the expense of consuming 303K more parameters and 16.8G more MACC operations in comparison to those consumed by SRNSSI, and 79K more parameters and 4.3G more MACC operations in comparison to those consumed by LatticeNet. This analysis of the results given in Table 4.9 shows that LatticeNet, SRNSSI and ESRNSSI networks are clearly the best performing light-weight networks, if the super resolution performance, the number of parameters and number of MACC operations are simultaneously taken into consideration.

As the super resolution network of [106] and MWCN of [27] use the idea of spectral feature generation for the task of image super resolution, we compare the performance and complexity of the proposed ESRNSSI network with that of these two networks in Table 4.10. It is seen from the results of this table that the proposed ESRNSSI network outperforms the other two wavelet-based super resolution convolutional neural networks in 17 out of 24 values of the PSNR and SSIM metrics on the four benchmark datasets. It should be pointed out that the number of parameters employed by ESRNSSI is only a very small fraction (4%) of that employed by the network of [106] or by MWCN [27].

Figs. 4.8 and 4.9 show the visual quality of the images *img061* and *img083* from the *Urban100* dataset with the scaling factor 4, when super resolved by some of the light-weight networks. It is seen from these figures that the quality of the image super resolved by each of the two proposed networks is much superior to those obtained by using the other networks. It is seen from the zoomed segments of the images in Fig. 4.8 that the shapes of the rectangular windows are more precisely recovered by using the two proposed networks. Similarly, it is seen from the zoomed segments of the images in Fig. 4.9 that the arcs in the hallway ceiling of the building are more accurately preserved in the images super resolved by the two proposed networks. Finally, it can be noted from the two figures that, as expected, the quality of the images super resolved by ESRNSSI is superior to those super resolved by SRNSSI.

Fig. 4.10 shows the *Lena* images from the *Set14* dataset with the scaling factors 4, 3 or 2 when they are super resolved by the proposed SRNSSI network. It is seen from this figure that all the images super resolved by the proposed SRNSSI network have good visual qualities. However, as expected, when the scaling factor is decreased from 4 to 2, the edges and textures in the reconstructed image become sharper.

We now form an ultralight-weight version of the proposed SRNSSI network and compare its performance with that of SRCNN [1] and its faster version, namely, FSRCNN [13].

In order to form the ultralight-weight version of the proposed network, we modify the architecture of the proposed SRNSSI in a number of ways. First, the number of filters in the first convolutional layer of the feature extraction stage is reduced to 32 from 64. Each of the group convolution operations in the spatial information processing module still uses 2 groups of convolutions, but each employs 16 instead of 32 filters with kernel size 3×3. Each of the convolution operations in the spectral information processing module uses 16 instead of 32 filters with kernel size 3×3. All the point-wise convolution operations in the residual block are performed using 32 filters. The feature extraction stage uses only one instead of four units of the residual block. Compared to the 553K parameters employed by the original SRNSSI, its ultralight-weight version employs only 39K parameters.

The performance and number of parameters of our ultralight-weight network are compared with those of the two other ultralight-weight networks existing in the literature, SRCNN [1] and its faster version, FSRCNN [13], that, respectively, employ 57K and 13K parameters. The average PSNR values on all the images of the four datasets with the scaling factor 4 super resolved by these three ultralight-weight networks are given in Table 4.11. It is seen from this table that the ultralight-weight version of our proposed network outperforms both SRCNN and FSRCNN.

4.6.3 Experimental Results of MorphoNet

In this section, we first perform an ablation study on the proposed residual block of MorphoNet in order to show the effectiveness of the morphological feature generation module in the residual block on the network performance. We then present and compare the performance of the proposed network on four benchmark datasets, namely, *Set5* [21], *Set14* [22], *BSD100* [23] and *Urban100* [24] and its complexity with that of the state-of-the-art light-weight image super resolution networks existing in the literature.

In order to investigate the effectiveness of the morphological feature generation

Table 4.12: Results on the ablation study of the proposed residual block of MorphoNet.

Network with	*Set5*	*Set14*	*BSD100*	*Urban100*
Variant	34.38 (0.9277)	30.44 (0.8446)	29.15 (0.8080)	27.98 (0.8502)
Proposed	34.52 (0.9284)	30.53 (0.8455)	29.20 (0.8085)	28.05 (0.8510)

Table 4.13: PSNR (SSIM) values resulting from applying MorphoNet and various state-of-the-art methods to images of four benchmark datasets.

Dataset	Scaling	Bicubic	SRCNN [1]	VDSR [3]	DRCN [19]	LapSRN [29]	MemNet [4]	IDN [45]	SRFBN [25]	CARN [14]	IMDN [54]	OISR [55]	MorphoNet (Proposed) [92]
Set5	×2	33.66 (0.9299)	36.66 (0.9542)	37.53 (0.9587)	37.63 (0.9588)	37.52 (0.959)	37.78 (0.9597)	37.83 (0.9600)	37.78 (0.9597)	37.76 (0.9590)	38.00 (0.9605)	38.02 (0.9605)	38.04 (0.9614)
	×3	30.39 (0.8682)	32.75 (0.9090)	33.66 (0.9213)	33.82 (0.9226)	N/A	34.09 (0.9248)	34.11 (0.9253)	34.20 (0.9255)	34.29 (0.9255)	34.36 (0.9270)	34.39 (0.9272)	34.52 (0.9284)
	×4	28.42 (0.8104)	30.48 (0.8628)	31.35 (0.8838)	31.53 (0.8854)	31.54 (0.885)	31.74 (0.8893)	31.82 (0.8903)	31.98 (0.8923)	32.13 (0.8937)	32.21 (0.8948)	32.14 (0.8947)	32.23 (0.0.8951)
Set14	×2	30.24 (0.8688)	32.42 (0.9063)	33.03 (0.9124)	33.04 (0.9118)	33.08 (0.913)	33.28 (0.9142)	33.30 (0.9148)	33.35 (0.9156)	33.52(0.9166)	33.63 (0.9177)	33.62 (0.9178)	33.77 (0.9196)
	×3	27.21(0.7385)	29.28 (0.8209)	29.77 (0.8314)	29.76 (0.8311)	N/A	30.00 (0.8350)	29.99 (0.8354)	30.10 (0.8372)	30.29 (0.8407)	30.32 (0.8417)	30.35 (0.8426)	30.53 (0.8455)
	×4	26.00 (0.7027)	27.49 (0.7503)	28.01 (0.7674)	28.02 (0.7670)	28.19 (0.772)	28.26 (0.7723)	28.25 (0.7730)	28.45 (0.7779)	28.60 (0.7806)	28.58 (0.7811)	28.63 (0.7819)	28.77 (0.7855)
BSD100	×2	29.56 (0.8431)	31.36 (0.8879)	31.90 (0.8960)	31.85 (0.8942)	31.80 (0.895)	32.08 (0.8978)	32.08 (0.8985)	32.00 (0.8970)	32.09 (0.8978)	32.19 (0.8996)	32.20 (0.9000)	32.32 (0.9025)
	×3	27.21 (0.7385)	28.41 (0.7863)	28.82 (0.7976)	28.80 (0.7963)	N/A	28.95 (0.8004)	28.96(0.8001)	28.96 (0.8010)	29.06 (0.8034)	29.09 (0.8046)	29.11 (0.8058)	29.20 (0.8085)
	×4	25.96 (0.6675)	26.90 (0.7101)	27.29 (0.7251)	27.23 (0.7233)	27.32 (0.728)	27.40 (0.7281)	27.41 (0.7297)	27.44 (0.7313)	27.58 (0.7349)	27.56 (0.7353)	27.60 (0.7369)	27.65 (0.7391)
Urban100	×2	26.88 (0.8403)	29.50 (0.8946)	30.76 (0.9140)	30.75 (0.9133)	30.41 (0.910)	31.31 (0.9195)	31.27 (0.9196)	31.41 (0.9207)	31.92 (0.9256)	32.17 (0.9283)	32.21 (0.9290)	32.06 (0.9288)
	×3	24.46 (0.7349)	26.24 (0.7989)	27.14 (0.8279)	27.15 (0.8276)	N/A	27.56 (0.8376)	27.42 (0.8359)	27.66 (0.8415)	28.06 (0.8493)	28.17 (0.8519)	28.24 (0.8544)	28.05 (0.8510)
	×4	23.14 (0.6577)	24.52 (0.7221)	25.18 (0.7524)	25.14 (0.7510)	25.21 (0.756)	25.50 (0.7630)	25.41 (0.7632)	25.71 (0.7719)	26.07 (0.7837)	26.04 (0.7838)	26.17 (0.7888)	26.01 (0.7830)

The values in the red font indicate the best performance and those in the blue font represent the second best performance.

Table 4.14: Complexity of MorphoNet and various super resolution schemes.

Method	Number of Parameters
SRCNN [1]	57K
VDSR [3]	665K
DRCN [19]	1770K
MemNet [4]	677K
IDN [45]	553K
SRFBN-S [25]	483K
CARN [14]	1592K
IMDN [54]	715K
OISR [55]	1550K
MorphoNet (Proposed)	1414K

module on the network performance, we form a variant of the proposed residual block by removing this module from the residual block. Table 4.12 gives the performance of the super resolution network employing the proposed residual block and its variant on the four benchmark datasets with the scaling factor 3. Our objective in the design of the proposed residual block is to generate morphological features in addition to the conventional hierarchical features that are generated solely through the convolutional operations in order to provide a very rich set of features. It is seen by comparing the results of this table

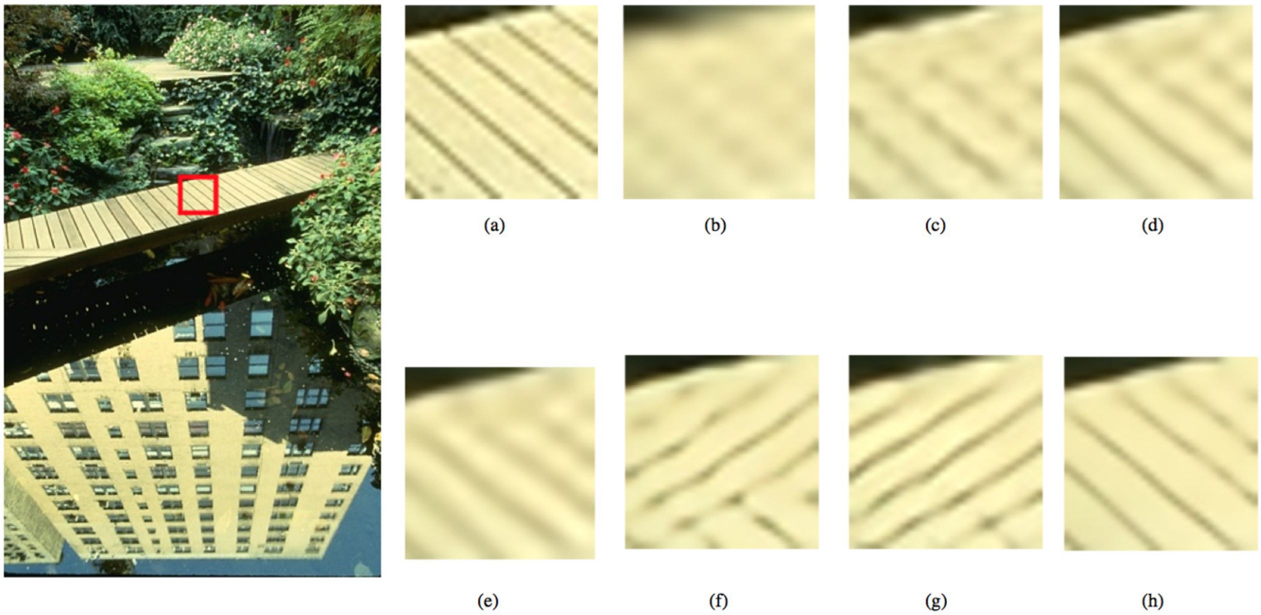


Figure 4.11: Visual quality of images *img021* super resolved by various schemes. (a) Ground truth. (b) Bicubic. (c) SRCNN. (d) VDSR. (e) DRCN. (f) CARN. (g) IMDN. (h) Morphonet.

corresponding to using the proposed residual block and its variant, that by removing the morphological feature generation module, the performance of the network gets reduced significantly.

Table 4.13 gives the performance in terms of the PSNR and SSIM metrics of the proposed super resolution network and those of ten other super resolution neural networks, namely, super resolution convolutional neural networks (SRCNN) [1], very deep super resolution network (VDSR) [3], deep recursive convolutional network (DRCN) [19], Laplacian pyramid super resolution network (LapSRN), memory persistent network (MemNet) [4], information distillation network (IDN) [45], super resolution feedback network (SRFBN) [25], cascaded residual network (CARN) [14], information multi-distillation network (IMDN) [54] and ODE-inspired super resolution network (OISR) [55]. It is seen from this table that the proposed network generally outperforms all the networks used in our comparison. Specifically, it is seen that the proposed network outperforms OISR, which is the best performing state-of-the-art super resolution network in the light-weight category employing the number of parameters in the neighborhood of $1.5M$ parameters or less, in 18 out of 24 cases of the PSNR and SSIM metric values.

Table 4.14 gives the complexity in terms of the number of parameters of the proposed and state-of-the-art light-weight super resolution networks. It is seen from this table that the proposed network employs $1.4M$ parameters, which is lower than the $1.55M$ parameters employed by the OISR network. Thus, considering the complexity and performance together, the proposed network can be regarded to be the best network among all the light-weight super resolution networks.

Fig. 4.11 shows the zoomed segments of the image *img021* from the *BSD100* dataset super resolved by various light-weight super resolution networks. It is seen from this figure that the image super resolved by the proposed network has the best visual quality. Specifically, the ridge textures of the pathway in the wooden bridge are recovered more accurately by the proposed network in comparison to that recovered by all the other networks. In particular, the ridge orientation in the recovered images by CARN [14] and IMDN [54] are completely altered from that of the ground truth image.

4.6.4 Experimental Results of SRNMSM

In this section, we first investigate the impact of each of the two feature generation modules on the super resolution performance of the proposed network (SRNMSM). We then, carry out an experiment to investigate the impact of the size of the structuring element used for the morphological operators on the network performance. Next, we investigate the impact of the use of de-gridding strategy employed in the multi-scale spatial feature generation module on the network performance. We also investigate the impact of using different feature fusion strategies in the proposed residual block on the performance of image super resolution. Then, we investigate the impact of replacing the morphological operations by

Table 4.15: Impact of Each Feature Generation Module on the Performance of the SRNMSM in terms of PSNR (dB).

Network	*Set5*	*Set14*	*BSD100*	*Urban100*
Variant 1	33.94	30.17	28.94	27.45
Variant 2	34.25	30.35	29.07	27.72
Proposed SR Network	34.36	30.44	29.13	27.93

Table 4.16: Impact of Combining the Four Morphological Operations on the Performance of SRNMSM in terms of PSNR (dB).

Network	*Set5*	*Set14*	*BSD100*	*Urban100*
No Morphological Operator	34.25	30.35	29.07	27.72
with Only Erosion	34.33	30.43	29.12	28.87
with Only Dilation	34.31	30.43	29.12	27.84
with Only Opening	34.34	30.43	29.12	27.82
with Only Closing	34.35	30.41	29.11	27.83
Proposed SR Network	34.36	30.44	29.13	27.93

Table 4.17: Impact of the Size of the Morphological Operators on the Performance of SRNMS in terms of PSNR (dB).

Morphological Operator Size	*Set5*	*Set14*	*BSD100*	*Urban100*
2×2	34.36	30.44	29.13	27.93
3×3	34.29	30.45	29.12	27.86
4×4	34.30	30.41	29.12	27.86

Table 4.18: Impact of Reducing Gridding Artifacts in the Multi-scale Spatial Feature Generation Module on the Performance of SRNMSM in terms of PSNR (dB).

Network	*Set5*	*Set14*	*BSD100*	*Urban100*
w/o De-gridding Strategy	34.30	30.44	29.12	27.83
Proposed SR Network	34.36	30.44	29.13	27.93

the gradient operations on the performance of image super resolution. Finally, we compare the performance, the number of parameters and the number of MACC operations of the proposed super resolution network with those of the state-of-the-art low-complexity super resolution convolutional neural networks available in the literature. Our proposed residual block for the task of image super resolution consists of two feature generation modules, multi-scale spatial feature generation module and morphological feature generation module. In order to investigate the impact of each of these two modules on the network

Table 4.19: Impact of using Different Feature Fusion Strategies on the Performance of SRNMSM in terms of PSNR (dB).

Network	*Set5*	*Set14*	*BSD100*	*Urban100*
Summation Fusion	34.29	30.42	29.11	27.85
Proposed SR Network	34.36	30.44	29.13	27.93

Table 4.20: Impact of Using the Morphological and Gradient Operations on the Performance of SRNMSM in terms of PSNR (dB).

Network	*Set5*	*Set14*	*BSD100*	*Urban100*
with Gradient Operations	34.35	30.40	29.12	27.90
Proposed SR Network	34.36	30.44	29.13	27.93

Table 4.21: Comparison between the Performance (in terms of PSNR (dB) and SSIM) of SRNMSM and the Light-weight Convolutional Neural Networks for Image Super Resolution.

Dataset	Scaling	SRCNN [1]	SCN [2]	DRRN [20]	DRCN [19]	LapSRN [29]	CARN-M [14]	SRFBN-S [25]	RMUN [56]	LMAN-S [57]	IMDN [54]	SRNMSM
Set5	×2	36.66 (0.9542)	36.93 (0.9252)	37.74(0.9591)	37.63 (0.9588)	37.52 (0.959)	37.53 (0.9583)	37.78 (0.9597)	37.77 (0.9600)	37.94 (0.9603)	38.00 (0.9605)	38.04 (0.9614)
	×3	32.75 (0.9090)	33.10 (0.9136)	34.03 (0.9244)	33.82 (0.9226)	N/A	33.99 (0.9236)	34.20 (0.9255)	34.12 (0.9251)	34.31 (0.9265)	34.36 (0.9270)	34.36 (0.9279)
	×4	30.48 (0.8628)	30.86 (0.8710)	31.68 (0.8888)	31.53 (0.8854)	31.54 (0.885)	31.92 (0.8903)	31.98 (0.8923)	31.84 (0.8901)	32.12 (0.8939)	32.21 (0.8948)	32.17 (0.8948)
Set14	×2	32.42 (0.9063)	32.56 (0.9069)	33.23 (0.9136)	33.04 (0.9118)	33.08 (0.913)	33.26 (0.9141)	33.35 (0.9156)	33.21 (0.9143)	33.49 (0.9167)	33.63 (0.9177)	33.75 (0.9191)
	×3	29.28 (0.8209)	29.41 (0.8235)	29.96 (0.8349)	29.76 (0.8311)	N/A	30.08 (0.8367)	30.10 (0.8372)	33.00 (0.8360)	30.24 (0.8397)	30.32 (0.8417)	30.44 (0.8446)
	×4	27.49 (0.7503)	27.64 (0.7578)	28.21 (0.7721)	28.02 (0.7670)	28.19 (0.772)	28.42 (0.7762)	28.45 (0.7779)	28.32 (0.7750)	28.53 (0.7798)	28.58 (0.7811)	28.71 (0.7843)
BSD100	×2	31.36 (0.8879)	31.40 (0.8884)	32.05 (0.8973)	31.85 (0.8942)	31.80 (0.895)	31.92 (0.8960)	32.00 (0.8970)	32.02 (0.8979)	32.08 (0.8984)	32.19 (0.8996)	32.28 (0.9020)
	×3	28.41 (0.7863)	28.50 (0.7885)	28.95 (0.8004)	28.80 (0.7963)	N/A	28.91 (0.8000)	28.96 (0.8010)	28.94 (0.8016)	29.02 (0.8030)	29.09 (0.8046)	29.13 (0.8076)
	×4	26.90 (0.7101)	27.03 (0.7161)	27.38 (0.7284)	27.23 (0.7233)	27.32 (0.728)	27.44 (0.7304)	27.44 (0.7313)	27.44 (0.7314)	27.51 (0.8340)	27.56 (0.7353)	27.59 (0.7382)
Urban100	×2	29.50 (0.8946)	29.52 (0.8970)	31.23 (0.9188)	30.75 (0.9133)	30.41 (0.910)	31.23 (0.9193)	31.41 (0.9207)	31.10 (0.9181)	31.85 (0.9251)	32.17 (0.9283)	31.91 (0.9271)
	×3	26.24 (0.7989)	26.21 (0.8010)	27.53 (0.8378)	27.15 (0.8276)	N/A	27.55 (0.8385)	27.66 (0.8415)	28.11 (0.8359)	28.02 (0.8487)	28.17 (0.8519)	27.93 (0.8488)
	×4	24.52 (0.7221)	24.52 (0.7260)	25.44 (0.7638)	25.14 (0.7510)	25.21 (0.756)	25.62 (0.7694)	25.71 (0.7719)	25.50 (0.7663)	25.96 (0.7813)	26.04 (0.7838)	25.85 (0.7792)
Number of Parameters		57K	33K	297K	1774K	813K	412K	483K	662K	709K	715K	695K
Number of MACC Operations		52.7G	37.8G	6796.9G	17974.3G	149.4G	32.5G	1045.3G	67.7G	22.9G	40.9G	40.3G

Red font indicates the best and blue font indicates the second best performance.

Table 4.22: Comparison between the Performance (PSNR in dB) of the Network using Residual Block of MoephoNet and the Proposed Network.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*
with Residual Block of [92]	34.32	30.43	29.11	27.85
Proposed Network	34.36	30.44	29.13	27.93

Table 4.23: Comparison between the Performance (PSNR in dB and SSIM) and Complexity of the Light-weight Version of SeaNet [58] and the Proposed SRNMSM.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*	Params
Light-weight SeaNet [58]	34.24 (0.9264)	30.35 (0.8427)	29.07 (0.8058)	27.77 (0.8449)	1128K
Proposed SR Network	34.36 (0.9279)	30.44 (0.8446)	29.13 (0.8076)	27.93 (0.8488)	695K

Table 4.24: Comparison between the Performance (PSNR in dB and SSIM) of Some Traditional Image Super Resolution Schemes and the Proposed SRNMSM.

Method	*Twelve Testing Images* [59]
NARM [61]	24.32 (0.7579)
NCSR [60]	28.77 (0.8808)
REPS-SR [59]	29.08 (0.8739)
Proposed SR Network	31.27 (0.9121)

Table 4.25: CPU Inference Time of IMDN [54] and the Proposed SRNMSM.

Network	*CPU Inference Time (second)*
IMDN [54]	2.4222
Proposed SR Network	2.1633

performance, we form two variants of the proposed residual block of SRNMSM, namely, *Variant 1* and *Variant 2*. *Variants 1* and *2* are obtained by removing, respectively, the multi-scale spatial feature generation module and the morphological feature generation module from the proposed residual block. Table 4.15 gives the performance of the proposed super resolution network, when it employs the proposed residual block and its two variants and the proposed and two resulting networks are applied to the images of four benchmark datasets, namely, *Set5* [21], *Set14* [22], *BSD100* [23] and *Urban100* [24], with the scaling factor 3. It is seen from the results of this table that removing the morphological feature generation module from the proposed residual block results in degrading the super resolution performance significantly. It is also seen from this table that the use of the multi-scale spatial features is necessary for providing a high super resolution performance. It should be pointed out that our idea in generating and using morphological features is to supplement the conventional spatial features with these additional features in order to further enhance the super resolution performance and compensate any performance degradation resulted from the use of only small number of residual blocks in a low-complexity super resolution network.

We now consider the impact of each of the four morphological operators individually on the super resolution performance of the network, that is, the features produced by the multi-scale spatial feature generation module are fused with the features produced by the morphological feature generation module that uses only one morphological operator at a time. For this purpose, we form four variants of the morphological feature generation module, each using only a single morphological operator, and therefore, four different corresponding networks. Table 4.16 gives the performance of the network using no morphological operators, i.e., the network using only the multi-scale spatial feature generation module, the four networks each using only one morphological operator, as well as the network using all the four operators (i.e., the proposed network). It is seen from the results of this table that the use of any one of the morphological operators improves the performance of the network over that not using any of them. Also, it is seen that combining all the four morphological operations provides the best performance.

In order to visualize the richness of the feature maps generated by the morphological feature generation module, we show in Fig. 4.12, four morphological maps obtained by applying the dilation, erosion, opening and closing morphological operators to a feature map of the *Barbara* image. It is seen from this figure that the four morphological feature maps possess various textures and structures. The fusion of these four morphological maps results in producing a rich set of features for image super resolution.

Fig. 4.13 shows the visual quality of the image *img085* from *Urban100* dataset, which is super resolved by the proposed SRNMSM and its variant that does not employ morphological feature generation module. A comparison of the zoomed versions of the images shown in Figs. 4.13 (a), (b), (c) and (d) demonstrates that the idea of using morphological operators in the proposed super resolution network has a significant impact in the quality of the super resolved image.

All the four morphological operators employed for generating morphological features of SRNMSM use the kernel size 2×2. In order to investigate the impact of different kernel

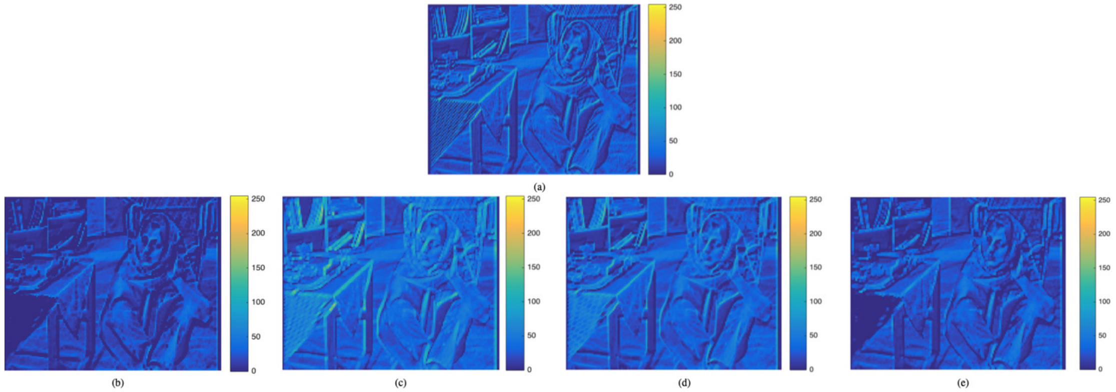


Figure 4.12: Feature maps generated by the morphological feature generation module, when the *Barbara* image is input to the network. (a) Feature map input to the residual block. (b) Feature map obtained after applying the dilation operation. (c) Feature map obtained after applying the erosion operation. (d) Feature map obtained after applying opening operation. (e) Feature map obtained after applying closing operation.

sizes of morphological operators on the network performance, we also use two other kernel sizes, namely, 3×3 and 4×4. For this study, the images of the four benchmark datasets with scaling factor 3 are used. The performance of the network using each of these three kernel sizes for the morphological operators is given in Table 4.17. It is seen from this table that the super resolution network using the kernel size 2×2 provides the best performance. However, by comparing the results of this table with those of Table 4.15, it is seen that the use of morphological operations regardless the size of its operators improves the network performance.

It was mentioned that the use of dilated convolution operation produces gridding artifacts in the feature maps. In order to suppress these artifacts, we added the features generated by the regular convolution operation to those generated by the dilated convolution operation. In order to see the impact of adding these two sets of feature maps in reducing these artifacts, in Table 4.18, we provide the performance results of the network both with and without the addition of features obtained using regular and dilated convolution operations, on the images of the four benchmark datasets with the scaling factor 3. It is seen from

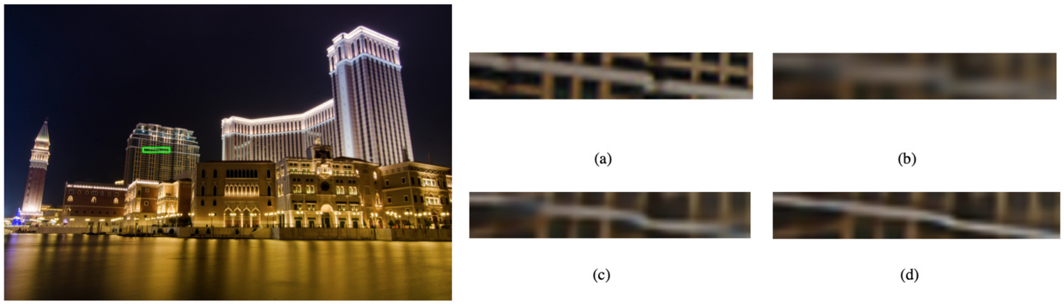


Figure 4.13: Visual quality of the image *img085* super resolved by the proposed network and its *Variant 2* that does not employ any morphological operator. (a) Ground truth. (b) Bicubic. (c) *Variant 2*. (d) Proposed SRNMSM.

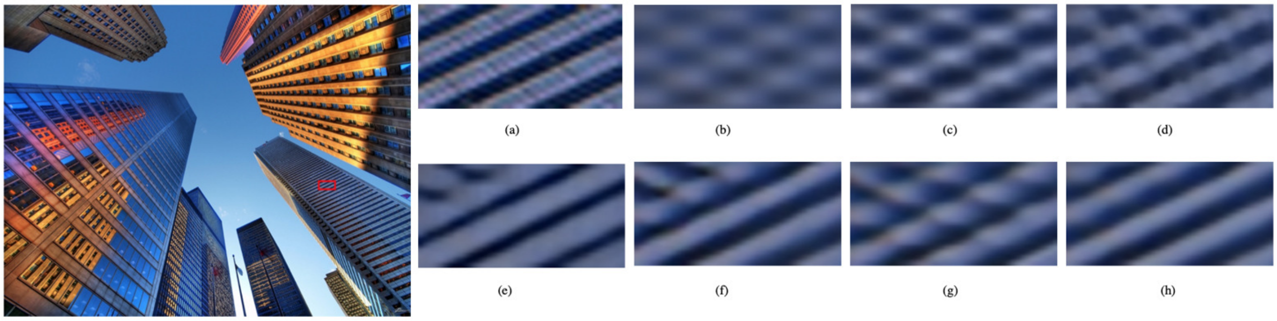


Figure 4.14: Visual quality of the *img012* images super resolved by the light-weight networks. (a) Ground truth. (b) Bicubic. (c) SRCNN [1]. (d) SCN [2]. (e) LapSRN [29]. (f) CARN-M [14]. (g) IMDN [54]. (h) SRNMSM.

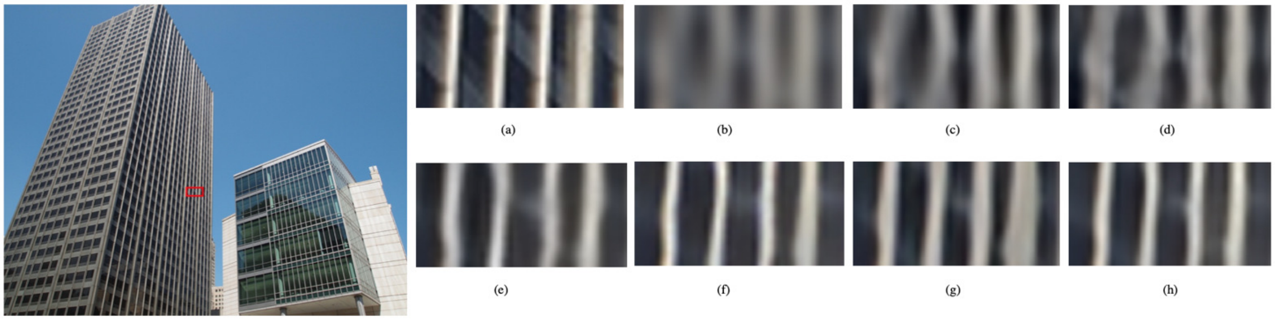


Figure 4.15: Visual quality of the *img096* images super resolved by the light-weight networks. (a) Ground truth. (b) Bicubic. (c) SRCNN [1]. (d) SCN [2]. (e) LapSRN [29]. (f) CARN-M [14]. (g) IMDN [54]. (h) SRNMSM.

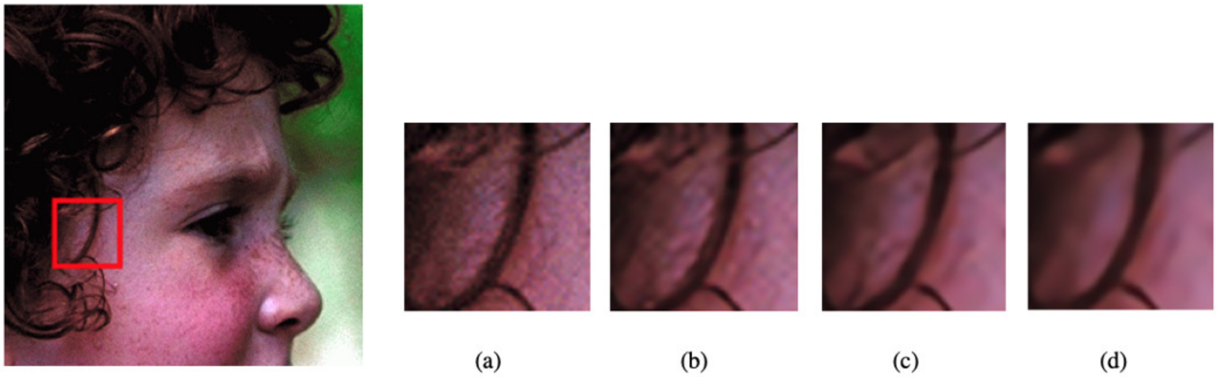


Figure 4.16: Visual quality of the *Girl* image super resolved by the proposed network. (a) Ground truth. (b) Super resolved with scaling factor 2. (c) Super resolved with scaling factor 3. (d) Super resolved with scaling factor 4.

the results of this table that the de-gridding strategy used in the multi-scale spatial feature generation module is indeed effective in reducing the gridding artifacts, and consequently, enhancing network performance.

In deep convolutional neural networks, there exist two common operations for fusing features, concatenation and element-wise summation. Feature fusion module of the proposed residual block uses the former operation for fusing multi-scale spatial features with the morphological features, since it allows a weighted fusion of the features from the corresponding pixel positions of the various channels. In order to show the superiority of this type of fusion over that of using the element-wise summation, we also provide in Table 4.19 the performance results of the network on the images of the four benchmark datasets with the scaling factor 3 using element-wise summation of the multi-scale spatial and morphological features. It is seen from this table that using the concatenative fusion is a better way of fusing the two types of features than the element-wise summation is.

The morphological operators are concerned with manipulating the edges and boundaries of the maps. There are four basic morphological operators, namely, erosion, dilation, opening and closing, aiming at extracting the structural and textural information of the feature maps. Specifically, the dilation and erosion operators affect the feature maps by, respectively, thickening and thinning the edges and boundaries, and the opening operator and closing operator, respectively, removes the small objects and fills up the small holes in the feature maps. In the proposed morphological feature generation module, the cascade of the point-wise and regular convolutional operations, instead of being applied directly on the maps input to the module, are applied on the four variants of the input maps, each obtained by manipulating the edges and boundaries of the maps through the four morphological operators. On the other hand, the edge or gradient extraction operators by acting directly on the input maps produce edges and boundaries contained in such maps. Therefore, if the four morphological operators in our proposed module are replaced by the horizontal and vertical gradient operators, the cascade of the point-wise and regular convolutional operations extract the features of the edge maps produced by the gradient operators. Hence, it can be expected that the features produced by our proposed morphological feature generation module are richer than those produced by using a module, in which the morphological operators are replaced by the gradient operators.

Table 4.20 provides the performance of the proposed network that uses the morphological operators and that of the network in which the morphological operators are replaced by the horizontal and vertical gradient operators, on the images of the four benchmark datasets with the scaling factor 3. It is seen from the results of this table that the network with the morphological operators outperforms that using the gradient operators.

The performance in terms of PSNR and SSIM and the number of parameters and MACC operations of the proposed image super resolution network are presented and compared with those of the state-of-the-art low-complexity super resolution networks in Table 4.21. The comparison is made by applying the networks, SRCNN [1], SCN [2], DRRN [20], DRCN [19], LapSRN [29], CARN-M [14], SRFBN-S [25], RMUN [56], LMAN-S [57] and IMDN [54], on the images of the four benchmark datasets with the scaling factors 2, 3 and 4. It is seen from this table that SRNMSM's performance is superior to that of all the networks used in our comparison in 17 out of 24 PSNR and SSIM values. This performance of the proposed network compares with that of IMDN [54], which provides the best performance in 8 cases of the 24 metric values. It is also seen from this table that the proposed SRNMSM provides the best performance by employing, respectively, 20 K and 0.6 G smaller parameters and operations than the second best performing network IMDN [54] does.

Both the proposed residual block and the residual block of MorphoNet employ morphological feature guidance strategy for generating rich sets of feature maps. In order to compare the impact of these two residual blocks on the network performance, we use 4 residual blocks of MorphoNet in the proposed super resolution network and compare its performance with that of the proposed super resolution network in Table 4.22 on the images of the four benchmark datasets with the scaling factor 3. It should be pointed out that these two networks employ comparable number of parameters. It is seen from the results of Table 4.22 that the proposed super resolution network outperforms the super resolution network using the residual block of [92].

We now compare the performance of the proposed network with that of SeaNet, which uses the gradient information for image super resolution. For a fair comparison, we bring down the number of parameters of SeaNet to the level of SRNMSM. For this purpose, we implement a light-weight version of SeaNet by using two convolutional layers for rough image reconstruction module, two multi-scale residual blocks for soft-edge reconstruction module and two residual blocks for image refinement module. Table 4.23 gives the performance and number of parameters of the proposed SRNMSM and those of the light-weight version of SeaNet on the images of the four benchmark datasets with the scaling factor 3. It is seen from the results of this table that the proposed SRNMSM outperforms SeaNet in that the former provides PSNR values that are higher in the range 0.06 dB to 0.16 dB than those provided by the latter.

In Table 4.24, we compare the performance of the proposed network with that of some traditional image super resolution schemes, namely, REPS-SR [59], NCSR [60] and NARM [61], on the twelve testing images used in [59] with the scaling factor 3. It is seen from the results of this table that the proposed network significantly outperforms these traditional schemes in view of its nonlinear end-to-end mapping capability.

In order to evaluate the execution time of the proposed image super resolution network, we obtain the inference times on a single CPU core clocked at 2.9 GHz, when SRNMSM and IMDN, the two best performing super resolution networks in Table 4.25, are implemented to super resolve the *Butterfly* RGB image of size 256×256 with the scaling factor 3. Table XI gives the CPU inference time (in seconds) for these super resolution networks. It is seen that the proposed network takes an inference time of $2.1633\ s$, which is 10% smaller than that by IMDN.

Figs. 4.14 and 4.15 show the images *img012* and *img096* from the *Urban100* dataset and their super resolved versions obtained from the proposed and various state-of-the-art low-complexity super resolution networks with the scaling factor 4. It is seen from the zoomed segments of the super resolved images that the similarity in the orientations of the building window frames recovered by using the proposed scheme is more in line to those of the ground truth images.

Fig. 4.16 shows the super resolved images when the proposed SRNMSM network is applied to the *Girl* image of *Set5* dataset downsampled with the scaling factors 2, 3 and 4. A comparison of the zoomed segments shown in Figs. 4.16 (b), (c) and (d) with that of Fig. 4.16 (a) demonstrates that the hair strand in the super resolved images is very similar to that in the ground truth image for all the scaling factors.

Table 4.26: PSNR (SSIM) values* resulting from applying various light-weight feature guiding methods to images of three benchmark datasets.

Dataset	Scaling	EFFRBNet [96]	SRNMSM [84]	MorphoNet [92]	ESRNSSI [87]
Set5	×2	38.00 (0.9612)	38.04 (0.9614)	38.04 (0.9614)	38.17 (0.9618)
	×3	34.31 (0.9270)	34.36 (0.9279)	34.52 (0.9284)	34.56 (0.9290)
	×4	32.02 (0.8928)	32.17 (0.8948)	32.23 (0.8951)	32.34 (0.8969)
Set14	×2	33.67 (0.9187)	33.75 (0.9191)	33.77 (0.9196)	33.92 (0.9205)
	×3	30.39 (0.8435)	30.44 (0.8446)	30.53 (0.8455)	30.57 (0.8467)
	×4	28.61 (0.7824)	28.71 (0.7843)	28.77 (0.7855)	28.83 (0.7883)
BSD100	×2	32.23 (0.9012)	32.28 (0.9020)	32.32 (0.9025)	32.38 (0.9028)
	×3	29.10 (0.8066)	29.13 (0.8076)	29.20 (0.8085)	29.23 (0.8097)
	×4	27.54 (0.7364)	27.59 (0.7382)	27.65 (0.7391)	27.68 (0.7410)
Urban100	×2	31.79 (0.9261)	31.91 (0.9271)	32.06 (0.9288)	32.24 (0.9303)
	×3	27.83 (0.8473)	27.93 (0.8488)	28.05 (0.8510)	28.24 (0.8557)
	×4	25.73 (0.7761)	25.85 (0.7792)	26.01 (0.7830)	26.13 (0.7884)

∗ The values in the red font indicate the best performance.

4.7 Comparison between Various Proposed Deep Image Super Resolution Networks using Guided Feature Generation

The performance results of the various deep light-weight image super resolution networks using guided feature generation proposed in this chapter are given in Table 4.26. From the results of this table, the following points can be highlighted. First, the super resolution networks SRNSSI and SRNMSM are two high performance networks that employ less than 1M parameters and still provide very high performance for image super resolution. The super resolution network EFFRBNet provides a high super resolution performance, but at the expense of employing more than 1M parameters. Of all the super resolution networks proposed in this chapter, ESRNSSI is the best one by employing less than 1M parameters and providing the highest image super resolution performance.

4.8 Conclusion

In this chapter, we have proposed several deep light-weight image super resolution networks by employing the idea of guided feature generation. Specifically, we have used three guided feature generation processes, namely, edge extraction, spectral feature generation and morphological feature generation, for improving the representational capability of a deep super resolution network and enhancing its performance. The results of the experiments carried out in this chapter have shown the effectiveness of the guided feature generation process for deep image super resolution convolutional networks.

Chapter 5

Deep Image Super Resolution Networks using Efficient Feature Fusion Techniques

5.1 Introduction

Deep image super resolution networks produce feature maps at various hierarchical levels, and fusing these features can further boost the performance of the networks. By fusing the features produced by a network, new rich and representable features can be obtained. In this chapter, we propose different feature fusion strategies for efficiently combining features of different residual blocks and also features within a residual block [85], [86], [90].

5.2 CompNet: A New Feature Fusion Technique for Deep Single Image Super Resolution Convolutional Neural Networks

Learning the residual signal is the goal of the networks that are based on residual learning. The residual signal is the difference between the original high resolution image and the interpolated version of the low resolution image. The proposed deep network, referred to as CompNet, consists of several convolutional layers each followed by a ReLU activation function. The architecture of the network [90] is shown in Fig. 5.1. The interpolated low resolution input signal at node A is fed to the first convolutional layer of this network, which produces at node B the first set of feature maps of the interpolated low resolution image. Between the nodes B and C, a total of d_1 convolutional layers are placed. This is followed by placing another d_2 convolutional layers between the nodes C and D and a single convolutional layer between the nodes C and E. The feature maps produced at the nodes B, E and D are then concatenated through a block represented by Con. Thus, node F represents a set of concatenated feature maps, which is fed to the last convolutional layer placed between the nodes F and G. Finally, the interpolated image from node A and the result from the node G are added to produce the final estimated high resolution image. The signal at node G, therefore, represents the estimated residual image. All of the layers of the network produce different feature maps of the interpolated image.

Let **u**, **v** and **w** be the feature vectors at nodes B, C and D, respectively, each produced by the convolutional layers and activation functions, say, ReLU. As one progresses deeper into the network, the features produced become more sparse and they are the results of the network undergoing increasingly more nonlinearity. Thus, in the feature vectors, **u** are the least sparse and they have been produced by the network at a node where it has undergone the least nonlinearity. On the other hand, the converse is true for the feature vectors **w**.

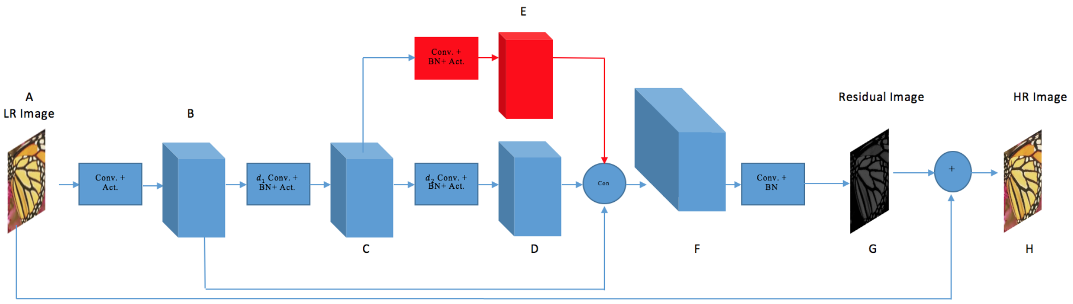


Figure 5.1: CompNet architecture. Conv., BN and Act. imply convolutional layer, batch normalization and activation function (in our work is specified as ReLU), respectively. Also, d_1 and d_2 refer to as the number of blocks, each including convolutional layer, batch normalization and activation function. The block with red color is considered for dimensionality reduction purpose.

However, depending on the value of d_1, the feature vectors represented by $\mathbf{v}$ have these two characteristics in between that of $\mathbf{u}$ and $\mathbf{w}$. Thus, a feature vector that is composed by using these three feature vectors can be expected to be a better representative of the spectrum of an estimated residual image. In view of this expectation, in this investigation, the three types of vectors are concatenated as

$$\mathbf{c} = \begin{bmatrix} \mathbf{w} \\ \mathbf{v} \\ \mathbf{u} \end{bmatrix} \tag{5.1}$$

Let m_1, m'_2 and m_3 be the numbers of slices to produce the tensors at nodes B, C and D, respectively, i.e. $\mathbf{u} \in \mathbb{R}^{m_1}$, $\mathbf{v} \in \mathbb{R}^{m'_2}$ and $\mathbf{w} \in \mathbb{R}^{m_3}$. Then, $\mathbf{c} \in \mathbb{R}^{m_1+m'_2+m_3}$. Since the sparsity of $\mathbf{v}$ is in between that of $\mathbf{u}$ and $\mathbf{w}$, and amount of the nonlinearity used to produce the feature vectors $\mathbf{v}$ is also in between that applied to produce $\mathbf{u}$ and $\mathbf{w}$, we propose a dimensionality reduction of $\mathbf{v}$ from m'_2 to m_2 by placing another convolutional layer between nodes C and E before carrying out the concatenation operation. The function of the last convolutional layer is selection of the features from **u**, **v** and **w** in constructing the estimated residual image.

The final feature vector, **c** in CompNet is expected to be less sparse in comparison to that provided by VDSR [3]. However, the feature vectors **c** experiences about the same amount of nonlinearity as that experienced by VDSR.

Each of the layers of the segment of the network shown in blue color between the nodes B and D uses 64 filters, each of support size 3×3. Since the purpose of the layer in the red segment of the network is dimensionality reduction, this layer employs only 32 filters each of spatial support size of 1×1. Also, since our objective is to include in the concatenation process the feature vectors whose characteristics of sparsity and nonlinearity are in between of those at nodes B and D, we chose $d_1 = d_2 = d$ and we set $d = 9$, so that the network is sufficiently deep. The final convolutional layer employs a single filter of size $3 \times 3 \times 160$, for the reconstruction of the residual image. Each of the convolutional layers except the last one are followed by a ReLU activation function.

As in other super resolution schemes that are based on deep learning, in our scheme also sub-images are used for the training of CompNet. Sub-images of size 48×48 with no overlap are used for the training. However, since CompNet is a fully convolutional network, it can be trained and tested on images of any size. In addition, multi-scale training is utilized for CompNet, and therefore, the training dataset consists of samples upscaled by various factors. This process not only removes the need of individual networks for each upscaling factor, but as has been shown in [3], it also improves the robustness of the network leading to a better performance.

The effective receptive field in CompNet, when the number of layers has a default value of 20 and the filter spatial support of 3×3, is 41, which is a little less than the size of the input sub-image. We have noticed that increasing the depth of CompNet over its default value of 20 does not improve its performance even though the corresponding effective receptive field remains within the size of the sub-images. Therefore, for the proposed CompNet, we keep the depth at 20 layers. For initializing the weights of our network, the method due to He et al. [7], which is based on the layer hyper parameters and the use of the ReLU activation function, is used. In this method, the kernel with a spatial support of $s \times s$ is randomly initialized with a Gaussian distribution having a zero mean and a variance of $\frac{2}{s^2 n}$, where n is the layer width.

In our experiments, we use *BSD200* [23] and *91 images* [6] that have, respectively, 200 and 91 images, as a training dataset. After data augmentation, a total number of 151815 sub-images are generated from this dataset and utilized as the the training samples. The training dataset is divided into batches each of size 64 (with the exception of the last batch), Thus each epoch has 2373 iterations (backpropagations). The number of epochs is set as 80 for the sake of consistency, when comparing the proposed scheme with other schemes. Also, the weight decay parameter is set as 10^{-4}. Since the main objective metric for evaluating the single image super resolution is the peak signal-to-noise ratio ($PSNR = 10\log_{10}(\frac{255^2}{MSE})$, where MSE represents the mean squared error), the loss function used by CompNet is the mean squared error.

All of the deep learning tasks are implemented using Keras [40] that is backended by TensorFlow package [41]. The training procedure for CompNet is conducted on a machine with Intel Core i7 CPU @4.2 GHz, 16 GB installed memory and GPU Nvidia Titan X (Pascal).

It should be pointed out that the feature fusion technique of CompNet is used for developing other image super resolution networks such as SRSubBandNet [101].

5.3 FPNet: A Deep Light-weight Interpretable Neural Network using Forward Prediction Filtering for Efficient Single Image Super Resolution

The use of residual blocks in a deep network facilitates the flow of the information in the forward and backpropagation. The super resolution schemes of [62], [63] and [64] are deep networks that focus on the way the feature maps within each residual block and those resulting from a combination of residual blocks are combined. However, in all these networks, the feature maps are simply fused through the concatenation and the point-wise convolution operations. In this section, we propose a deep super resolution network [85] in which the feature maps of the residual blocks are combined in the same way as it is done in a forward prediction error (FPE) filtering of adaptive signal processing that allows the extraction of a weighted combination of the feature maps generated by a set of residual blocks.

Here, first the FPE filter as used in adaptive signal processing is briefly reviewed and then employed to develop the proposed super block for its use in a deep super resolution network. The training details of the proposed network are also described.

Let X_n denote a stationary discrete-time random process. The value of X_n at time l, i.e., $X(l)$, can be predicted from a linear combination of its values at M previous samples, i.e., $X(l-1), ..., X(l-M)$, as

$$\hat{X}(l|\mathcal{X}_{l-1}) = \sum_{k=1}^{M} w_k^* X(l-k) \tag{5.2}$$

where $\mathcal{X}_{l-1}$ represents set of values of the random process X_n at samples $l-1, ..., l-M$, and w_k denotes the kth weight of the adaptive filter. In the theory of adaptive signal processing, the weight vector $\mathbf{w} = [w_1, w_2, ..., w_M]^T$ is obtained in such a way that the following error is minimized

$$f_M(l) = X(l) - \hat{X}(l|\mathcal{X}_{l-1}) \tag{5.3}$$

By minimizing $f_M(l)$, the resulting weight vector **w** is given by

$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{r} \tag{5.4}$$

where **R** and **r**, respectively, denote the auto-correlation matrix of the random process X_n at samples $l-1, ..., l-M$ and the cross-correlation vector between the the value of random process X_n at sample l and its values at the previous M samples. The error given by (5.3) is referred to as the forward prediction error (FPE).

Fig. 5.2 shows the proposed architecture of SB. The feature tensor **u** input to SB is passed through a cascade of $M+1$ dense residual blocks. Let $\mathbf{R}_i$ $(i = 1, ..., M+1)$ denote the operation of the ith residual block of SB. The ith residual block by operating on its input feature tensor $\mathbf{v}_{i-1}$ $(\mathbf{v}_0 = \mathbf{u})$ produces the output feature tensor $\mathbf{v}_i$ given by

$$\mathbf{v}_i = \mathbf{R}_i(\mathbf{v}_{i-1}) \quad i = 1, ..., M+1 \tag{5.5}$$

The output feature tensor $\mathbf{v}_i$ produced by the ith residual block in SB correspond to the value of the $\{l - M + (i-1)\}th$ sample of the random process X_n. Next, each of the feature tensors $\mathbf{v}_i$'s $(i = 1, ..., M)$ is recalibrated using, respectively, the feature recalibration modules $\mathbf{P}_i$'s $(i = 1, ..., M)$. For the feature recalibration, we use a squeeze-and-excitation (SE) unit [33], which consists of a cascade of global average pooling operation and two point-wise convolutions. The first point-wise convolution operation employs 4 filters followed by a ReLU activation, whereas the second one uses 64 filters followed by a sigmoid activation. The module $\mathbf{P}_i$ weights each channel of the feature tensor $\mathbf{v}_i$ $(i = 1, ..., M)$

individually. The recalibrated feature tensor $\mathbf{r}_i$ is obtained as

$$\mathbf{r}_i = \mathbf{P}_i(\mathbf{v}_i) \quad i = 1, ..., M \tag{5.6}$$

In accordance with (5.2), the recalibrated feature tensors $\mathbf{r}_i$'s are added to provide the feature tensor $\hat{\mathbf{v}}_{M+1}$ as

$$\hat{\mathbf{v}}_{M+1} = \sum_{i=1}^{M} \mathbf{r}_i \tag{5.7}$$

The feature tensor $\hat{\mathbf{v}}_{M+1}$ given by the above equation corresponds to $\hat{X}(l|\mathcal{X}_{l-1})$. Now, FPE given by (5.3) can be implemented to provide the feature tensor $\mathbf{e}$ given by

$$\mathbf{e} = \mathbf{v}_{M+1} - \hat{\mathbf{v}}_{M+1} \tag{5.8}$$

Note that the feature tensor $\mathbf{e}$ is a residual feature tensor in the architecture of SB and corresponds to the forward prediction error $f_M(l)$ given by (5.3). In order to provide an enhanced learning to SB, the feature tensor $\mathbf{e}$ is first further processed through a convolution operation F to yield feature tensor $F(\mathbf{e})$. Then, $F(\mathbf{e})$ is added to $\mathbf{v}_{M+1}$ to provide the output feature maps denoted by $\mathbf{z}$ of the super block as

$$\mathbf{z} = \mathbf{v}_{M+1} + F(\mathbf{e}) \tag{5.9}$$

where the convolution operation F employs 64 filters, each with kernel size 3×3.

As seen from Fig. 5.2, the proposed SB uses a cascade of several residual blocks $\mathbf{R}_i$'s. Each residual block consists of three convolutional layers, each composed of 64 filters with kernel size 3×3, and followed by ReLU activation function. The outputs of the three convolutional layers are concatenated and the resulting feature tensor undergoes a pointwise convolution operation using 64 filters in order to form the residual feature maps of the residual block. Finally, the features that are input to this dense residual block are added to

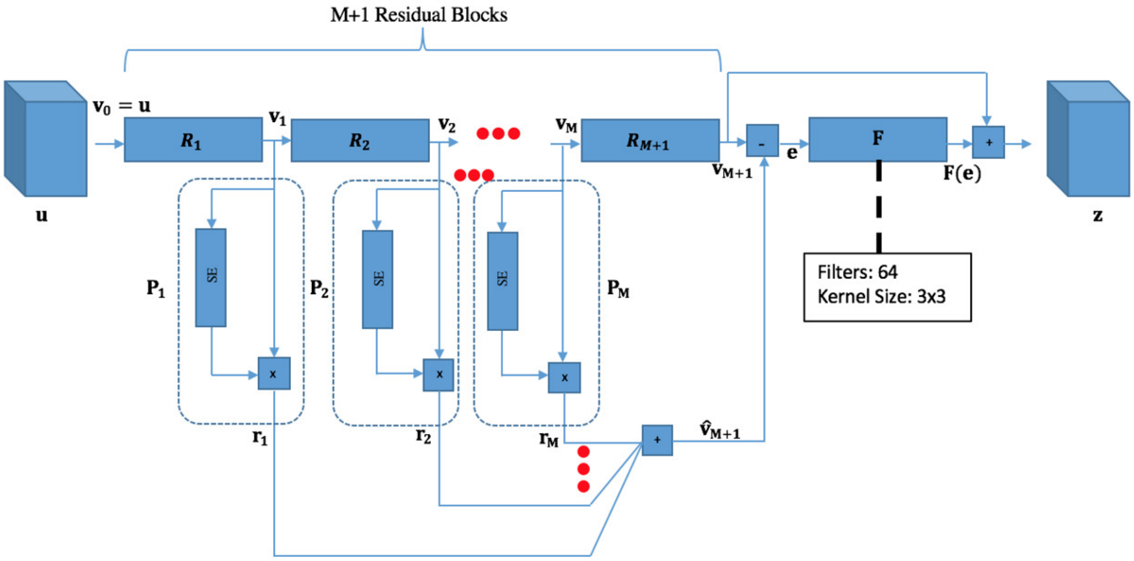


Figure 5.2: Architecture of the proposed super block. SE denotes the squeeze-and-excitation unit.

the residual features to yield the output features of the block.

In the overall architecture of the proposed super resolution network, first the low resolution input image is made to undergo a convolution operation using 64 filters each with kernel size 3×3 and a ReLU activation. The output of this layer is subjected to a cascade of 3 units of SBs and the spatial resolution of the resulting high level feature maps are restored to that of the ground truth image by passing them through a sub-pixel convolution operation. Next, these feature maps with increased spatial resolution are fed to a convolutional layer in order to construct the residual image between the bilinearly interpolated version the low resolution image and the ground truth image. This convolutional layer employs 3 filters each with kernel size 3×3.

We refer to the proposed image super resolution network as **F**orward **P**rediction **Net**work for image super resolution (FPNet) [85].

The theory of forward prediction error (FPE) filtering of adaptive signal processing allows the estimation of a sample of a random signal from a linear combination of the samples in a set of its preceding samples. This is in fact possible in view of the assumption that in practical situations, the current sample is correlated with the samples that immediately precede it. The values of the adaptive weights in the linear combination are, therefore, determined by minimizing the error between the current sample and the linear combination of its preceding samples. In the proposed super block for the task of image super resolution shown in Fig. 5.2, the feature tensors $\mathbf{v}_1, \dots, \mathbf{v}_{M+1}$ are computed by applying convolution operations on their preceding feature tensors staring from feature tensor **u** input to the super block. Hence, all these feature tensors are correlated. Therefore, the feature tensor $\mathbf{v}_{M+1}$ can be estimated as a linear combination of the feature tensors $\mathbf{v}_1, \dots, \mathbf{v}_M$. As in FPE, we obtain the adaptive weights determined by the modules $\mathbf{P}_i$'s by minimizing the error tensor **e**, which is the difference between the feature tensor $\mathbf{v}_{M+1}$ and the linear combination of $\mathbf{v}_1, \dots, \mathbf{v}_M$, i.e., $\hat{\mathbf{v}}_{M+1}$. The minimization of the feature tensor **e** is achieved through the residual end-to-end mapping of the deep network that is composed of a cascade of the proposed super blocks. Note that the error tensor **e**, that is the residual tensor between $\mathbf{v}_{M+1}$ and $\hat{\mathbf{v}}_{M+1}$, necessarily has the high frequency information, and therefore, its addition to the feature tensor $\mathbf{v}_{M+1}$ results in a richer set of features.

Finally, it should be noted that the proposed super block is a non-linear system. Hence, unlike the FPE filtering, which is a linear predictive filtering scheme, a closed form expression such as the one given by (5.4) does not exist for the weight modules $\mathbf{P}_i$'s in the linear combination of feature tensors $\mathbf{v}_1, \dots, \mathbf{v}_M$. The values of these weights, along with those of all the other parameters, are obtained through a supervised learning of the network employing the proposed super blocks.

For the training of the proposed network, the dataset *DIV2K* [42] that consists of 800 training images is considered. The training samples of size 48×48 are extracted from the images of this dataset. The $\ell 1$ norm of the loss between the ground truth samples and the estimated high resolution samples obtained by applying the proposed network to the degraded low resolution sub-images is used for updating the network parameters. The $\ell 1$ norm loss is minimized by using the method of stochastic gradient descent. A value of 64 is chosen for the training batch size.

5.4 SRNHARB: A Deep Light-weight Image Super Resolution Network using Hybrid Activation Residual Blocks

The task of image super resolution is essentially a non-linear mapping between the low resolution input image and the ground truth image. Therefore, the use of ReLU activation between the two convolution operations in a residual block imparts to the network the necessary non-linearity. It also disentangles the dense set of information contained in the feature maps into a sparse robust set of information, and therefore, simplifies the learning process of the model. However, the feature rectification carried out by ReLU results in losing information associated with the negative-valued features that might otherwise be useful for the task of image super resolution. In order to address this problem, one could employ other non-linear activations that allow the passage of the negative-valued features, such as parametric ReLU (PReLU) or ELU. Even though the use PReLU or ELU also brings sparsity, the feature maps produced by these activations are not as sparse as those produced by the use of ReLU. In this section, we propose a new residual block, in which both the positive and negative-valued features of the input to the block, each with sufficient sparsity, are processed separately. Specifically, the input maps to the block are decomposed into positive-valued features and negative-valued features by using ReLU and inverted and negated ReLU activations, respectively, and processed separately in two parallel streams by group convolutions. Thus, in this mechanism of processing the feature maps, the individual streams are able to preserve the sparsity that is provided by the use of a single activation unit, and the use of group convolutions allows to control the increase in the computational complexity resulting from processing both the positive and negative-valued features rather than processing only the positive-valued ones.

Here, we first develop the architecture of the proposed residual block [86]. We then present the overall architecture of the image super resolution network using this residual block. Finally, we explain the training details of the proposed super resolution network.

Fig. 5.3 shows the architecture of the proposed residual block. As seen from this figure, the feature tensor **x** input to the residual block is first made to undergo a convolution operation W_1, which uses 64 filters each with kernel size 3×3, yielding the feature tensor $\mathbf{u}_1$ as given by

$$\mathbf{u}_1 = W_1(\mathbf{x}) \tag{5.10}$$

The feature tensor $\mathbf{u}_1$ is then simultaneously passed through the ReLU and inverted and negated ReLU activations yielding, respectively, feature tensors $\mathbf{u}_2$ and $\mathbf{u}_3$, which are given by

$$\begin{aligned} \mathbf{u}_2 &= ReLU(\mathbf{u}_1) \\ \mathbf{u}_3 &= -ReLU(-\mathbf{u}_1) \end{aligned} \tag{5.11}$$

It is seen from (5.11) that the feature tensors $\mathbf{u}_2$ and $\mathbf{u}_3$ contain, respectively, the positive and negative-valued features of $\mathbf{u}_1$. Hence, the information in $\mathbf{u}_1$ is completely preserved. The feature tensors $\mathbf{u}_2$ and $\mathbf{u}_3$ can be processed individually by a convolutional layer. However, this increases the complexity of the residual block. Since one of our main objectives in developing the proposed residual block is to design it in a light-weight manner, we apply group convolutions W_2 and W_3, respectively, to the feature tensors $\mathbf{u}_2$ and $\mathbf{u}_3$, producing feature tensors $\mathbf{u}_4$ and $\mathbf{u}_5$ as given by

$$\begin{aligned} \mathbf{u}_4 &= W_2(\mathbf{u}_2) \\ \mathbf{u}_5 &= W_3(\mathbf{u}_3) \end{aligned} \tag{5.12}$$

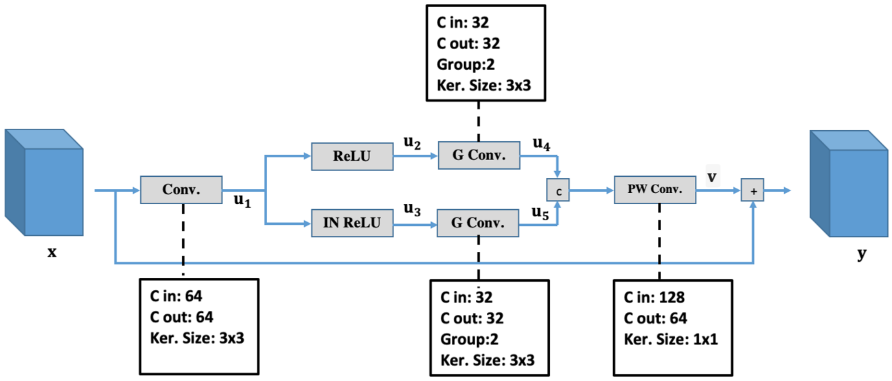


Figure 5.3: Architecture of the proposed residual block. Conv., G Conv., PW Conv. and IN ReLU, respectively, denote convolution, group convolution, point-wise convolution and inverted and negated ReLU activation.

where each of the group convolution operations W_2 and W_3 employs two groups of 32 filters each with kernel size 3×3. Since in our group convolutions, the convolution operations are carried out only on one-half of the input channels rather than all the channels, the complexity of employing the two group convolution operations W_2 and W_3 by the proposed residual block is the same as that of employing a single regular convolution operation using 64 filters. Next, the feature tensors $\mathbf{u}_4$ and $\mathbf{u}_5$ are concatenated and the resulting feature tensor is made to undergo a point-wise convolution operation W_4 yielding the residual feature tensor:

$$\mathbf{v} = W_4\big(CONCAT(\mathbf{u}_4, \mathbf{u}_5)\big) \tag{5.13}$$

where the point-wise convolution operation W_4 employs 64 filters. Finally, the residual feature tensor $\mathbf{v}$ is added to the block's input feature tensor $\mathbf{x}$ in order to obtain its output $\mathbf{y}$.

Fig. 5.4 shows the overall architecture of the proposed image super resolution network. It is seen from this figure that in this network, the low resolution input image $\mathbf{X}$ first undergoes a convolution operation employing 64 filters each with kernel size 3×3 yielding the feature maps $\mathbf{U}$. Next, the feature tensor $\mathbf{U}$ is made to undergo the operation of a cascade of 9 units of proposed residual block. Each residual block generates a set of feature maps

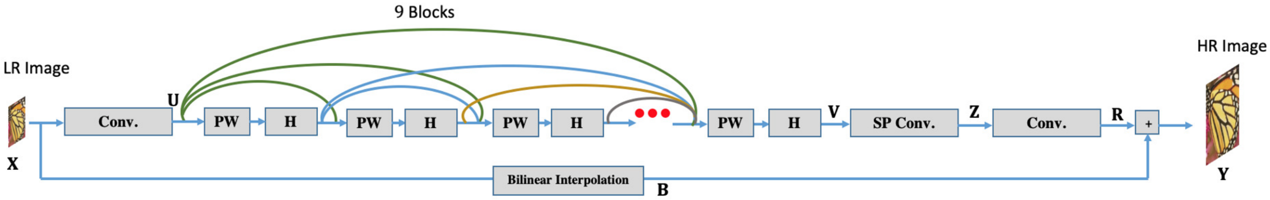


Figure 5.4: Network overall architecture. SP Conv., PW and H, respectively, denote the sub-pixel convolution operation, point-wise convolution operation and HARB block.

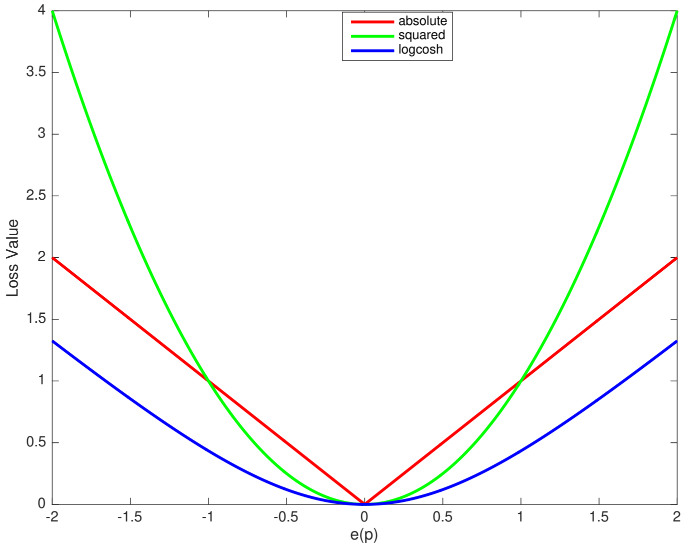


Figure 5.5: Plot of different objectives functions representing the loss between the ground truth and high resolution training samples.

at a distinct hierarchical level of abstraction. Hence, fusing the outputs of various residual blocks using dense connections results in generating rich sets of features by the residual blocks. However, the use of dense connections between residual blocks increases the network complexity. In order to generate rich sets of feature maps by using dense connections and at the same time keep the network complexity low, we place a point-wise convolution operation using 64 filters before each residual block to keep the number of feature channels to be processed by the block low. The output feature tensor $\mathbf{V}$ obtained from the $9th$ residual block is fed to a sub-pixel convolutional layer [8] and the spatial resolution of its feature maps is increased to that of the ground truth image. Finally, the upscaled feature tensor **Z** obtained at the output of the sub-pixel convolutional layer is made to undergo a convolution operation using 3 filters each with kernel size 3×3 in order to obtain the residual signal **R** between the ground truth image and the bilinear interpolated version **B** of the low resolution input image.

We refer to the proposed residual block of Fig. 5.3 as hybrid activation residual block (HARB), in view of its using both the ReLU and inverted and negated ReLU activations, and to the proposed super resolution network of Fig. 5.4 as the **S**uper **R**esolution **N**etwork using **H**ybrid **A**ctivation **R**esidual **B**locks (SRNHARB) [86].

In order to train our convolutional neural network, we use the images of the *DIV2K* [42] dataset. The samples of the training set are formed by extracting sub-images of size 48×48 from the 800 images of this dataset.

For the training of the proposed network, we consider three objective functions, $\ell 2$ norm, $\ell 1$ norm and logcosh objective functions. Fig. 5.5 illustrates the plots of the absolute value, squared value and logcosh value of the error *e(p)* representing the difference between the estimated high resolution value and the ground truth value of pixel *p*. It is clear from this figure that since the slope of the absolute function is larger than that of either of the other two functions when the value of *e(p)* is small, it can be expected that the training of a network using the $\ell 1$ norm objective function would be the fastest. However, since the gradient of the the absolute function is not a continuous function at *e(p)* $= 0$, wheres as that of the other two functions are, it can be expected that a better convergence for an optimal solution can be achieved by either $\ell 2$ norm based or logcosh based objective functions. In view of these characteristics of the three objective functions, we adopt the strategy for training SRNHARB in two parts. Initially, the weights of SRNHARB are updated for a certain number of iterations using the $\ell 1$ norm based objective function, and then, the network is fine-tuned by using the objective functions which is either $\ell 2$ norm based or logcosh based.

The optimization process is carried using stochastic gradient descent (SGD) technique. The learning process is started with the step size of 0.1 and decreased by a factor of 10 after each $182,500$ iterations. A value of 10^{-4} is assigned to the weight decay parameters for carrying out the convolution operations. The method of [7] is used for initializing the parameters of convolution operations. A value of 64 is chosen as the batch size. The number of iterations after which the objective function is switched from the $\ell 1$ norm based to the $\ell 2$ norm based or logcosh based is $547,500$, which is a number that is empirically determined.

5.5 Experimental Results

5.5.1 Experimental Results of CompNet

In this section, the results of various experiments that are conducted using CompNet are presented and analyzed. The results of CompNet and five state-of-the-art schemes namely, A+ [47], RFL [65], SRCNN, cascaded SCN (CSCN) and VDSR, are given in Table 5.1. It is seen from the results of this table that in almost all the cases, CompNet outperforms all of the state-of-the-art schemes. In some cases, improvement in the performance provided by CompNet is quite significant. For instance, in the case of *Set14* test set (upscaled by 3), CompNet yields a PSNR value that is 0.29 dB higher than that given by VDSR along with the similarity measure that $\%0.54$ higher.

With the default settings, CompNet is composed of 19 layers each of width 64 followed by a layer of width one. Now, we consider two separate variations in the default settings of the hyper parameters of CompNet. In the first one, we reduce the depth of CompNet from 20 to 16. The resulting network is referred to as reduced-depth CompNet (*RD CompNet*), whereas, in the second one, the width of CompNet is reduced from 64 to 32 and refer the resulting network as reduced-width CompNet (*RW CompNet*). Table 5.2 gives the total number of parameters for the three settings of CompNet as well as that for SRCNN and

Table 5.1: PSNR (SSIM) Values Resulting from Applying CompNet and Various State-of-the-art Methods to Images of Three Datasets.

Dataset	Scaling	Bicubic	A+ [47]	RFL [65]	SRCNN [1]	CSCN [2]	VDSR [3]	DEGREE [17]	CompNet [90]
Set5	×2	33.66 (0.9299)	36.54 (0.9544)	36.54(0.9537)	36.66 (0.9542)	37.00 (0.9557)	37.53 (0.9587)	37.54(0.9584)	**37.58** (**0.9596**)
	×3	30.39(0.8682)	32.58(0.9088)	32.43(0.9057)	32.75(0.9090)	33.18 (0.9153)	33.66(0.9213)	**33.72** (0.9204)	33.67(**0.9219**)
	×4	28.42(0.8104)	30.28(0.8603)	30.14(0.8548)	30.48(0.8628)	30.94(0.8755)	31.35(**0.8838**)	**31.43** (0.8818)	31.35 (0.8833)
Set14	×2	30.24(0.8688)	32.28(0.9056)	32.26(0.9040)	32.42(0.9063)	32.65 (0.9081)	33.03(0.9124)	33.01 (0.9118)	**33.29** (**0.9149**)
	×3	27.21(0.7385)	29.13(0.8188)	29.05(0.8164)	29.28(0.8209)	29.41 (0.8234)	29.77(0.8314)	29.87(0.8317)	**30.06** (**0.8368**)
	×4	26.00(0.7027)	27.32(0.7491)	27.24(0.7451)	27.49(0.7503)	27.71 (0.7592)	28.01(0.7674)	28.02 (0.7646)	**28.26** (**0.7732**)
BSD100	×2	29.56(0.8431)	31.21(0.8863)	31.16(0.8840)	31.36(0.8879)	31.46 (0.8891)	31.90(0.8960)	31.76 (0.8939)	**31.91** (**0.8972**)
	×3	27.21(0.7385)	28.29(0.7835)	28.22(0.7806)	28.41(0.7863)	28.52 (0.7883)	28.82(0.7976)	28.69 (0.7937)	**28.84** (**0.7995**)
	×4	25.96(0.6675)	26.82(0.7087)	26.75(0.7054)	26.90(0.7101)	27.06 (0.7167)	**27.29** (0.7251)	27.14 (0.7200)	27.28 (**0.7272**)

Bold font indicates the best.

Table 5.2: Complexity of CompNet and Various Super Resolution Schemes.

Method	Number of Parameters
SRCNN (Reproduced)	57281
VDSR (Reproduced)	665921
ComNet	673605
RD CompNet	524869
RW CompNet	170405

VDSR. It is seen from this table that the number of parameters for SRCNN is the lowest. However, it is not a deep network. On the other hand, CompNet with the two new settings has a complexity lower than that of VDSR with RW CompNet having a considerably lower complexity. Fig. 5.6 gives the PSNR value as a function of the number of epochs. It is seen from this figure that CompNet provides a performance that is superior to that of RD CompNet or RW CompNet. However, the performance of the two latter networks are not substantially different, thus indicating the robustness of CompNet with respect to its width and depth. It is worth noting that the number of parameters of RW CompNet is substantially lower than that of CompNet with only a modest decrease in the performance.

We now run another experiment using the proposed network in which the number of parameters is reduced from 673K to 636K by removing one of the nonlinear mapping layers

Table 5.3: PSNR (SSIM) results of CompNet with 19 layers and VDSR. The bolded values are the best in the comparison.

Dataset	scaling	VDSR	CompNet
Set5	×2	37.53 (0.9587)	**37.54** (**0.9594**)
	×3	**33.66**(0.9213)	33.64(**0.9215**)
	×4	**31.35**(**0.8838**)	31.29 (0.8829)
Set14	×2	33.03(0.9124)	**33.25** (**0.9149**)
	×3	29.77(0.8314)	**30.05** (**0.8367**)
	×4	28.01(0.7674)	**28.23** (**0.7730**)
BSD100	×2	31.90(0.8960)	31.90 (**0.8971**)
	×3	28.82(0.7976)	**28.83** (**0.7994**)
	×4	**27.29** (0.7251)	27.27 (**0.7271**)

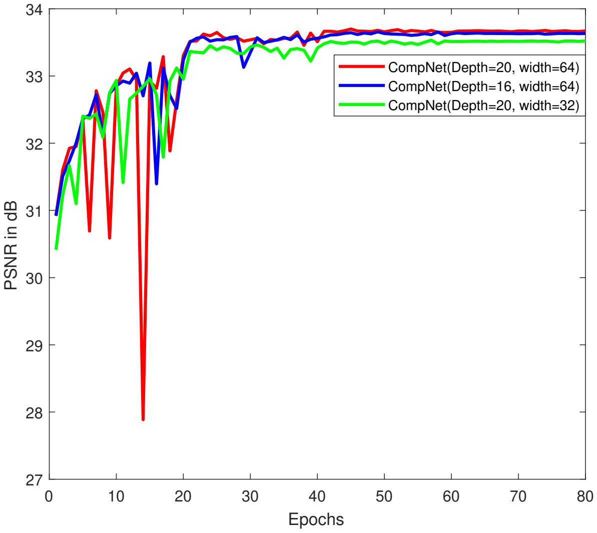


Figure 5.6: CompNet convergence for different settings of hyperparameters on *Set 5* with upscaling factor 3.

(i.e., layer 19). We apply the network with reduced number of parameters on the three benchmark datasets. The results obtained from our network and that obtained by applying VDSR, which uses 665K parameters, are shown in Table 5.3. It is seen from this table that the proposed network still yields a performance superior to that given by VDSR. It is clear from the results of this experiment that the performance gain of the proposed network over VDSR cannot be simply attributed to the use of a larger number of parameters, but rather to the use of an appropriate composition of the features in reconstructing the residue.

5.5.2 Experimental Results of FPNet

This section first carries out an ablation study by finding the performance of the super resolution network that employs the proposed super block. Then, the performance of the proposed scheme is compared with that of the low-complexity super resolution networks that are available in the literature on benchmark datasets, *Set5* [21], *Set14* [22], *BSD100* [23] and *Urban100* [24]. The complexity of the proposed network is also compared.

In the proposed super resolution network of FPNet, we use a cascade of 3 units of SB, each of which employs 4 ($M = 3$) dense residual blocks. The effectiveness of the proposed super block is demonstrated by forming its three variants, each focusing on a separate idea used in its design, and using them in the super resolution network. The super block of *Variant 1* consists of only the dense residual blocks without using $\mathbf{P}_i$ units, and the network has simply 12 units of the dense residual blocks. *Variant 2* is formed by replacing the dense residual blocks of SBs by simple residual blocks, each consisting of a cascade of 3 convolutional layers employing 64 filters with kernel size 3×3. *Variant 3* is obtained by removing the $\mathbf{P}_i$ units from *Variant 2*, i.e., this variant consists of just 12 units of the simple residual blocks. The PSNR values of the images obtained by the network using individually the three variants when the network is applied to the images downsampled by the scaling factor 4 from *Set5* and *Urban100* datasets are given in Table 5.4. The results

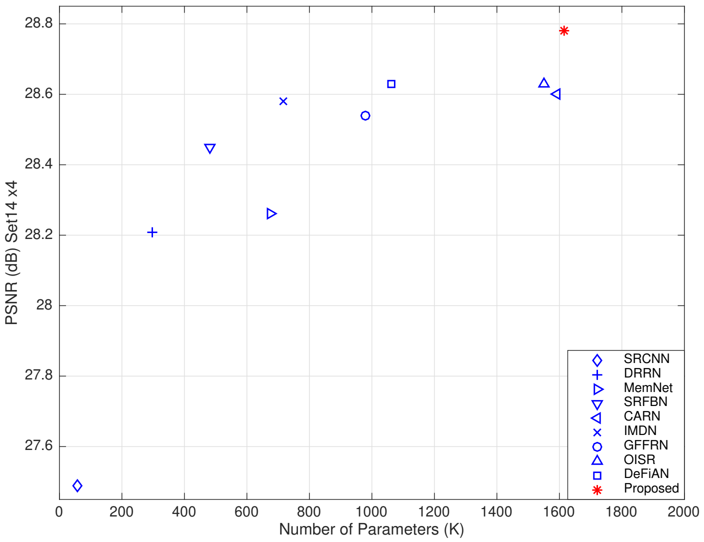


Figure 5.7: Plot of performance versus number of parameters of the various light-weight image super resolution networks.

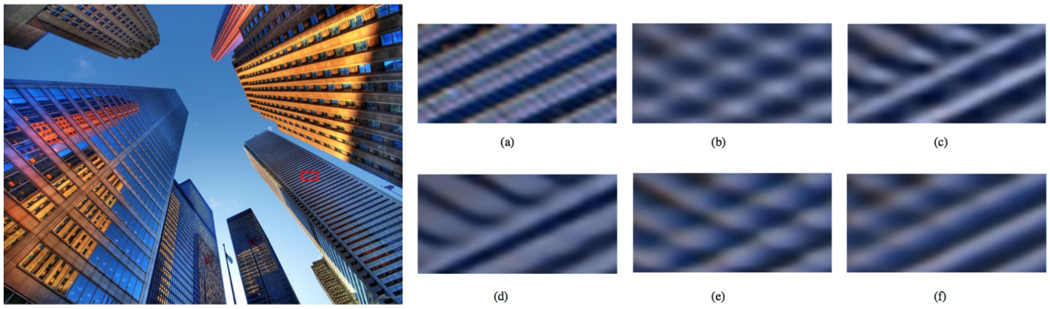


Figure 5.8: Visual quality of images *img012* super resolved by various schemes. (a) Ground truth. (b) SRCNN. (c) VDSR. (d) DRRN. (e) IMDN. (f) FPNet.

of the table show that the proposed scheme outperforms significantly all its three variants. By comparing the performance of *Variant 1* with that of the proposed network, and the performance *Variant 3* with that of *Variant 2*, it is clear that $\mathbf{P}_i$ units, which implement the idea of forward prediction, are very important in the design of the proposed super block. It is also to be noted that despite the significance of $\mathbf{P}_i$ units in improving the performance of the proposed network, they do not add much to its complexity. We now compare the performance of our scheme with that of the ten low-complexity schemes. These light-weight

Table 5.4: Results of Ablation Studies on the Proposed FPNet.

Benchmark	*Variant 1*	*Variant 2*	*Variant 3*	*Proposed*
PSNR on Set5	32.25	32.30	32.27	32.32
PSNR on Urban100	26.04	26.03	25.79	26.09

Table 5.5: PSNR (SSIM) values resulting from applying FPNet and various state-of-the-art methods to images of four benchmark datasets.

Dataset	Scaling	Bicubic	SRCNN [1]	DRRN [20]	MemNet [4]	SRFBN [25]	CARN [14]	DeCoNAS [62]	GFFRN-L [63]	IMDN [54]	DeFiAN [66]	OISR [55]	FPNet (Proposed) [85]
Set5	×2	33.66 (0.9299)	36.66 (0.9542)	37.74(0.9591)	37.78 (0.9597)	37.78 (0.9597)	37.76 (0.9590)	37.96 (0.9594)	37.96 (0.9603)	38.00 (0.9605)	38.03 (0.9605)	37.84 (0.9504)	38.13 (0.9616)
	×3	30.39 (0.8682)	32.75 (0.9090)	34.03 (0.9244)	34.09 (0.9248)	34.20 (0.9255)	34.29 (0.9255)	N/A	34.27 (0.9263)	34.36 (0.9270)	34.42 (0.9273)	34.39 (0.9272)	34.48 (0.9285)
	×4	28.42 (0.8104)	30.48 (0.8628)	31.68 (0.8888)	31.74 (0.8893)	31.98 (0.8923)	32.13 (0.8937)	N/A	32.03 (0.8934)	32.21 (0.8948)	32.16 (0.8942)	32.14 (0.8947)	32.32 (0.8962)
Set14	×2	30.24 (0.8688)	32.42 (0.9063)	33.23 (0.9136)	33.28 (0.9142)	33.35 (0.9156)	33.52(0.9166)	33.63 (0.9175)	33.51 (0.9169)	33.63 (0.9177)	33.63 (0.9181)	33.62 (0.9178)	33.83 (0.9198)
	×3	27.21(0.7385)	29.28 (0.8209)	29.96 (0.8349)	30.00 (0.8350)	30.10 (0.8372)	30.29 (0.8407)	N/A	30.29 (0.8409)	30.32 (0.8417)	30.34 (0.8410)	30.35 (0.8426)	30.53 (0.8454)
	×4	26.00 (0.7027)	27.49 (0.7503)	28.21 (0.7721)	28.26 (0.7723)	28.45 (0.7779)	28.60 (0.7806)	N/A	28.54 (0.7803)	28.58 (0.7811)	28.63 (0.7810)	28.63 (0.7819)	28.78 (0.7856)
BSD100	×2	29.56 (0.8431)	31.36 (0.8879)	32.05 (0.8973)	32.08 (0.8978)	32.00 (0.8970)	32.09 (0.8978)	32.15 (0.8986)	32.13 (0.8992)	32.19 (0.8996)	32.20 (0.8999)	32.16 (0.8993)	32.29 (0.9018)
	×3	27.21 (0.7385)	28.41 (0.7863)	28.95 (0.8004)	28.96(0.8001)	28.96 (0.8010)	29.06 (0.8034)	N/A	29.07 (0.8039)	29.09 (0.8046)	29.12 (0.8053)	29.11 (0.8058)	29.20 (0.8086)
	×4	25.96 (0.6675)	26.90 (0.7101)	27.38 (0.7284)	27.40 (0.7281)	27.44 (0.7313)	27.58 (0.7349)	N/A	27.54 (0.7347)	27.56 (0.7353)	27.58 (0.7363)	27.60 (0.7369)	27.66 (0.7394)
Urban100	×2	26.88 (0.8403)	29.50 (0.8946)	31.23 (0.9188)	31.31 (0.9195)	31.41 (0.9207)	31.92 (0.9256)	32.03 (0.9265)	31.91 (0.9263)	32.17 (0.9283)	32.20 (0.9286)	32.21 (0.9290)	32.04 (0.9278)
	×3	24.46 (0.7349)	26.24 (0.7989)	27.53 (0.8378)	27.56 (0.8376)	27.66 (0.8415)	28.06 (0.8493)	N/A	28.03 (0.8493)	28.17 (0.8519)	28.20 (0.8528)	28.24 (0.8544)	28.19 (0.8534)
	×4	23.14 (0.6577)	24.52 (0.7221)	25.44 (0.7638)	25.50 (0.7630)	25.71 (0.7719)	26.07 (0.7837)	N/A	25.94 (0.7815)	26.04 (0.7838)	26.10 (0.7862)	26.17 (0.7888)	26.09 (0.7850)

The values in the red font indicate the best performance and those in the blue font represent the second best performance.

Table 5.6: Performance of the Network with Re-balancing Feature Fusion and the Proposed FPNet.

Benchmark	*R-balancing Feature Fusion*	*Proposed*
PSNR on Set5	32.28	32.32
PSNR on Urban100	26.05	26.09

Table 5.7: Performance, Number of Parameters and Microprocessor Inference Time of the Proposed FPNet and its Lighter Version.

Evaluation	*FPNet-Light*	*Proposed*
Number of Params	292K	1615K
PSNR on Set5	31.86	32.32
Microprocessor Inference Time	2.1386	11.3072

Table 5.8: Impact of Weight Quantization on the FPNet Performance.

Network	*PSNR on Set5*	*Model Size (MB)*
QFPNet 1	32.23	116.17
QFPNet 2	32.17	116.17
QFPNet 3	32.28	116.17
Proposed	32.32	125.61

networks are SRCNN [1], DRRN [20], MemNet [4], CARN [14], SRFBN [25], DeCoNAS [63], GFFRN [62] IMDN [54], DeFiAN [66] and OISR [55], each employing less than 2M parameters. Table 5.5 shows the results of comparison in terms of PSNR and SSIM metrics. These results show that the other super resolution networks are outperformed by the proposed network in 18 out of a total of 24 cases of the values of the two metrics for the images of the benchmark datasets used.

In Fig. 5.7, the performance of the proposed network along with those of the other networks is plotted as a function of the number of parameters employed by the network. It can be seen from this plot that the performance of the proposed network is higher than that of the scheme of OISR [55], which by employing the same number of parameters as by ours stands as the second best.

The feature re-balancing fusion technique [64] is an effective way of fusing features obtained at various hierarchical levels for the task of image super resolution. In order to compare the impact of the proposed feature fusion technique that is based on the idea of FPE and that of the feature re-balancing fusion [64] on the performance of a deep image super resolution network, we form a deep neural network employing 12 units of the dense residual block (used in our network) with the feature re-balancing fusion technique using four convolutional operations with dilation rates 1, 2, 3 and 3, and compared its performance with that of the proposed network in Table 5.6. As seen from the results of this table, the proposed network outperforms the network with the feature re-balancing fusion technique [64].

We now implement the proposed super resolution network, which has 3 super blocks each consisting of 4 dense residual blocks, on a Raspberry Pi 4 microprocessor with 4 GB RAM and 32 GB memory, and obtain its average microprocessor inference time for super resolving the images of *Set5* dataset with the scaling factor 4. We also implement a lighter version of the proposed network, *FPNet-Light*, having only 1 super block and 2 dense residual blocks. Note that *FPNet-Light* employs only 292K parameters compared to 1615K parameters of FPNet. Table 5.7 gives the numbers of parameters, average PSNR values (in dB) and the microprocessor inference times (in second) of the proposed FPNet and its lighter version, *FPNet-Light*. It is seen from this table that the inference time of the lighter version of FPNet is almost one-fifth of that of the original version with the PSNR value reduced by about 0.4 dB. However, it should be noted that the PSNR value provided by this lighter version is higher than those provided by some of the networks used in Table 5.5.

We now analyze the network performance as a function of the model size by performing an experiment of applying weight pruning, weight quantization and bit-precision optimization on different convolutional layers of the proposed network. First of all, we would like to point out that our network does not converge to an acceptable solution when the weight quantization and bit-precision optimization are performed on the convolutional layers of more than one dense residual block, whether the blocks chosen for such an optimization are from a single or multiple super blocks. Hence, we perform an 8-bit quantization and bit-precision optimization on the weights of the all convolutional layers of only one dense residual block, namely, the last dense residual block of only the first, second or the third super block at a time. The three corresponding networks are referred to as *QFPNet 1*, *QFPNet 2* and *QFPNet 3*, respectively. Table 5.8 gives the average PSNR performance (in dB) and the model size (in million bytes (MB)) of the original and the three quantized versions of the proposed network, when these versions are applied to the images of *Set5* dataset with the scaling factor 4. It is seen from this table that this weight quantization and bit-precision optimization result in a model size reduction of 7.5% and also that the performance of the network is least affected when this weight optimization is performed on the last dense residual block of the last super block.

In Fig. 5.8, the versions of *img12* image selected from *Urban100* downsampled with the scaling factor 4 and super resolved by the different light-weight networks are shown. The zoomed parts of these super resolved images show that the similarity in the orientations of the building windows recovered by using the proposed scheme are more in line to those of the ground truth image.

5.5.3 Experimental Results of SRNHARB

An ablation study is first carried out in this section in order to show the effectiveness of the proposed residual block for image super resolution. Also, the performance and complexity of the proposed network are compared with those of the light-weight super resolution networks that exist in the literature when the networks are applied on the four benchmark datasets [21], [22], [23], [24].

In order to investigate the impact of the hybrid feature rectification mechanism employed by the proposed hybrid activation residual block on the network performance, we consider several variants of HARB based on the way the features are rectified. *Variant 1* is obtained by removing the bottom branch of HARB, which processes the negative-valued features, and replacing the group convolutional layer of the top branch with the regular convolutional layer (in order to keep the complexity of *Variant 1* to be about the same as that of HARB). Effectively, this block consists of a cascade of a convolutional layer, a ReLU activation and another convolutional layer. Both the convolutional layers in this block employ 64 filters each with kernel size 3×3. It should be noted that *Variant 1* is essentially the basic residual block used in EDSR [28]. *Variants 2* and *3* have the same architecture as that of the *Variant 1*, except that *Variant 2* uses a PReLU activation and *Variant 3* employs an ELU activation between the two convolutional layers. Thus, both *Variants 2* and *3* process the negative-valued features as well as positive-valued features. However, the negative-valued features are modified according to the activation functions PReLU and ELU. Nine units of HARB and ten units of these three variants are used in the super resolution network in

Table 5.9: Impact of the Proposed Feature Rectification Mechanism Employed by HARB on the Network Performance.

Super Resolution Network employing	Block Architecture	*Set5*	*Set14*	*BSD100*	*Urban100*	Parameters
Variant 1	Conv+ReLU+Conv	32.30	28.79	27.64	25.99	973K
Variant 2	Conv+PReLU+Conv	32.29	28.77	27.65	26.03	973K
Variant 3	Conv+ELU+Conv	32.25	28.75	27.63	25.95	973K
Variant 4	Conv. + Positive Feature Rectification	32.29	28.78	27.66	26.05	937K
Hybrid Activation Residual Block	Conv. + Proposed Feature Rectification	32.35	28.80	27.66	26.09	937K

order to keep the complexity of the resulting networks about the same. Table 5.9 gives the performance of the proposed super resolution network and those using the three variants, when they are applied to the images of the four benchmark datasets (*Set5* [21], *Set14* [22], *BSD100* [23] and *Urban100* [24]), with the scaling factor 4. The four networks are trained using the $\ell 1$ norm objective function. It is seen from the results of Table 5.9 that the super resolution network using the proposed HARB outperforms the network using either of these three variants, by employing comparable number of parameters.

Now we form yet another variant of HARB, *Variant 4*, in which the inverted and negated ReLU activation is replaced by the ReLU activation. The basic idea behind using this variant is the same as that of *Variant 1*, that is, to process only the positive-valued features, but to keep the basic structure of HARB preserved. The super resolution results of using this variant in the architecture of the super resolution network are also given in Table 5.9. It is seen that the performance of the network using this variant is also not as good as that of the network using HARB. It can be concluded that processing both positive and negative-valued features improves the super resolution performance and that the idea of hybrid rectification used in HARB is an effective way of achieving this.

We now illustrate in Fig. 5.9, by taking an example of *Zebra* image from *Set14* dataset, the process of decomposing of the features of this image by HARB block into positive and negative-valued features, $\mathbf{u}_2$ and $\mathbf{u}_3$, and then processing them individually to produce the feature tensors $\mathbf{u}_4$ and $\mathbf{u}_5$, respectively. Note that in the schemes that use only ReLU, $\mathbf{u}_3$ and $\mathbf{u}_5$ are not available. Figs. 5.9 (a)-(d) show the spatial contents of the four typical maps

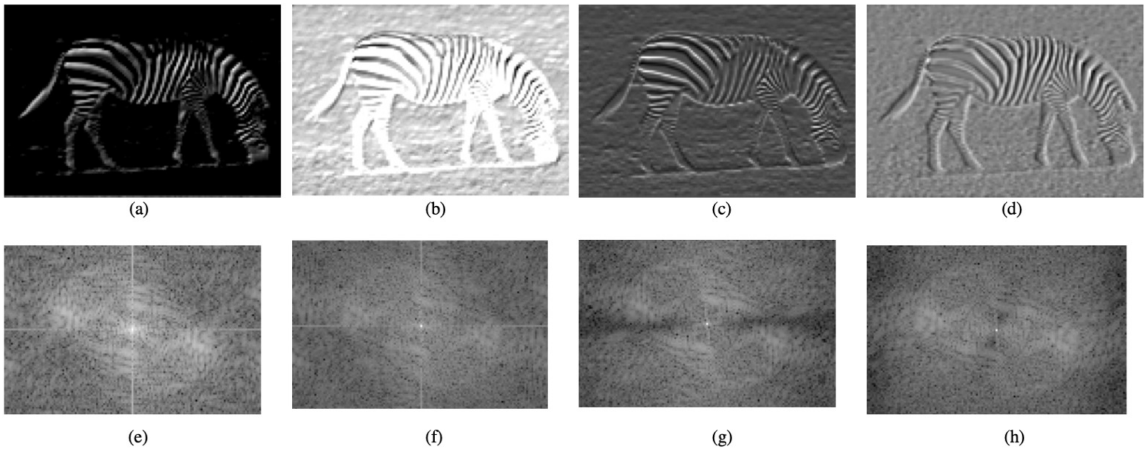


Figure 5.9: Selected feature maps obtained by the proposed residual block. (a) Feature map obtained after ReLU activation. (b) Feature map obtained after inverted and negated ReLU activation. (c) Feature map obtained by the group convolution after ReLU. (d) Feature map obtained by the group convolution after inverted and negated ReLU. (e) 2DFFT of feature in (a). (f) 2DFFT of feature in (b). (g) 2DFFT of feature in (c). (h) 2DFFT of feature in (d).

each selected from the tensors $\mathbf{u}_2$, $\mathbf{u}_3$, $\mathbf{u}_4$ and $\mathbf{u}_5$, respectively. Similarly, Figs. 5.9 (e)-(h), show the frequency domain contents obtained by applying the two-dimensional fast Fourier transform (2DFFT) to the spatial domain feature maps of Figs. 5.9 (a)-(d). In view of the fact that, since $\mathbf{u}_4$ and $\mathbf{u}_5$ have different spatial and frequency contents, the feature tensor $\mathbf{v}$ obtained by their fusion provides a richer set of features.

The proposed residual block, HARB, uses group convolution operations in order to keep the complexity of the super resolution network low. The group convolution operation in each of the two branches of HARB uses two groups of 32 filters with kernel size 3×3. In order to investigate the impact of using group convolution operations by HARB on the network performance and complexity, we form another variant of the residual block, referred to as *Variant 5*, in which the group convolution is replaced by the regular convolution using 32 filters with kernel size 3×3. Table 5.10 gives the performance on the four benchmark datasets with the scaling factor 4 and the number of parameters of the super resolution

Table 5.10: Impact of using Group Convolutions in HARB on the Performance and Complexity of the Network.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*	Parameters
Variant 5	32.33	28.79	27.66	26.07	1018K
HARB	32.35	28.80	27.66	26.09	937K

Table 5.11: Impact of using Dense Connections between HARB Units on the Performance and Complexity of the Network.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*	Params	MACC
No Dense Connection	32.19	28.73	27.62	25.95	921K	47.38G
Proposed Network	32.35	28.80	27.66	26.09	937K	53.04G

Table 5.12: Impact of using Different Training Strategies on the Performance of SRNHARB.

Network Trained with	*Set5*	*Set14*	*BSD100*	*Urban100*
$\ell 2$ *norm*	31.99	28.70	27.59	25.86
logcosh	31.94	28.65	27.54	25.72
$\ell 1$ *norm*	32.35	28.80	27.66	26.09
$\ell 1$ *norm followed by* $\ell 2$ *norm*	32.34	28.86	27.67	26.15
$\ell 1$ *norm followed by logcosh*	32.35	28.86	27.68	26.15

network when it uses 9 units of HARB and also when it uses 10 units of *Variant 5*. The network in these two cases is trained using the $\ell 1$ norm objective function. It is seen from the results of the table that the network employing HARB outperforms that using *Variant 5*. It is noted that despite the fact that there are inter-channel feature communications in *Variant 5*, the network performance using this variant is lower. This is due to the fact that HARB generates larger number of feature maps than *Variant 5* does. This shows that generating larger number of feature maps is more important for the image super resolution of our network than the inter-channel feature communication is.

In order to investigate the impact of using dense connections between various units of HARB on the network performance as well as on the number of MACC operations, we remove from the proposed network the dense connections. We also add one more unit of HARB to the resulting network, so that the number of parameters in the networks with and without dense connections remains comparable. The two networks are trained with

Table 5.13: Comparison between the Performance and Complexity of the Light-weight Convolutional Neural Networks for Image Super Resolution.

Dataset	Scaling	SRCNN [1]	SCN [2]	DRRN [20]	DRCN [19]	LapSRN [29]	CARN [14]	SRFBN-S [25]	PAN [68]	IMDN [54]	LatticeNet [16]	A2F [69]	SRNHARB [86]
Set5	×2	36.66 (0.9542)	36.93 (0.9252)	37.74(0.9591)	37.63 (0.9588)	37.52 (0.959)	37.76 (0.9590)	37.78 (0.9597)	38.00 (0.9605)	38.00 (0.9605)	38.15 (0.9610)	38.04 (0.9607)	38.04 (0.9612)
	×3	32.75 (0.9090)	33.10 (0.9136)	34.03 (0.9244)	33.82 (0.9226)	N/A	34.29 (0.9255)	34.20 (0.9255)	34.40 (0.9271)	34.36 (0.9270)	34.53 (0.9281)	34.50 (0.9278)	34.55 (0.9289)
	×4	30.48 (0.8628)	30.86 (0.8710)	31.68 (0.8888)	31.53 (0.8854)	31.54 (0.885)	32.13 (0.8937)	31.98 (0.8923)	32.13 (0.8948)	32.21 (0.8948)	32.30 (0.8962)	32.28 (0.8955)	32.35 (0.8969)
Set14	×2	32.42 (0.9063)	32.56 (0.9069)	33.23 (0.9136)	33.04 (0.9118)	33.08 (0.913)	33.52 (0.9166)	33.35 (0.9156)	33.59 (0.9181)	33.63 (0.9177)	33.78 (0.9193)	33.67 (0.9184)	33.71 (0.9185)
	×3	29.28 (0.8209)	29.41 (0.8235)	29.96 (0.8349)	29.76 (0.8311)	N/A	30.29 (0.8407)	30.10 (0.8372)	30.36 (0.8423)	30.32 (0.8417)	30.39 (0.8424)	30.39 (0.8427)	30.63 (0.8474)
	×4	27.49 (0.7503)	27.64 (0.7578)	28.21 (0.7721)	28.02 (0.7670)	28.19 (0.772)	28.60 (0.7806)	28.45 (0.7779)	28.61 (0.7822)	28.58 (0.7811)	28.68 (0.7830)	28.62 (0.7828)	28.86 (0.7877)
BSD100	×2	31.36 (0.8879)	31.40 (0.8884)	32.05 (0.8973)	31.85 (0.8942)	31.80 (0.895)	32.09 (0.8978)	32.00 (0.8970)	32.18 (0.8997)	32.19 (0.8996)	32.25 (0.9005)	32.18 (0.8996)	32.25 (0.9013)
	×3	28.41 (0.7863)	28.50 (0.7885)	28.95 (0.8004)	28.80 (0.7963)	N/A	29.06 (0.8034)	28.96 (0.8010)	29.11 (0.8050)	29.09 (0.8046)	29.15 (0.8059)	29.11 (0.8054)	29.22 (0.8090)
	×4	26.90 (0.7101)	27.03 (0.7161)	27.38 (0.7284)	27.23 (0.7233)	27.32 (0.728)	27.58 (0.7349)	27.44 (0.7313)	27.59 (0.7363)	27.56 (0.7353)	27.62 (0.7367)	27.58 (0.7364)	27.68 (0.7405)
Urban100	×2	29.50 (0.8946)	29.52 (0.8970)	31.23 (0.9188)	30.75 (0.9133)	30.41 (0.910)	31.92 (0.9256)	31.41 (0.9207)	32.01 (0.9273)	32.17 (0.9283)	32.43 (0.9302)	32.27 (0.9294)	31.84 (0.9254)
	×3	26.24 (0.7989)	26.21 (0.8010)	27.53 (0.8378)	27.15 (0.8276)	N/A	28.06 (0.8493)	27.66 (0.8415)	28.11 (0.8511)	28.17 (0.8519)	28.33 (0.8538)	28.28 (0.8546)	28.33 (0.8561)
	×4	24.52 (0.7221)	24.52 (0.7260)	25.44 (0.7638)	25.14 (0.7510)	25.21 (0.756)	26.07 (0.7837)	25.71 (0.7719)	26.11 (0.7854)	26.04 (0.7838)	26.25 (0.7873)	26.17 (0.7892)	26.15 (0.7888)
Number of Parameters		57K	33K	297K	1774K	813K	1592K	483K	272K	715K	777K	1010K	937K
Number of MACC Operations		52.7G	37.8G	6796.9G	17974.3G	149.4G	90.9G	1045.3G	28.2G	40.9G	43.6G	56.7G	53.04G

Red font indicates the best and blue font indicates the second best performance.

Table 5.14: Average Inference Time and Memory Consumption of the Best Performing Networks on the *Set5* images.

Network	Performance (dB)	Inference Time (s)	Memory (MB)
A2F-M	32.28	2.8119	105.69
LatticeNet	32.30	2.6319	73.71
SRNHARB	32.35	2.6559	124.56

the $\ell 1$ norm objective function. Table 5.11 gives the performance as well as the complexity in terms of numbers of parameters and MACC operations for the two networks. It is seen from this table that the use of dense connections between HARB units significantly enhances the performance of the proposed SRNHARB. This performance improvement is obtained in view of the fact that the information fusion resulting from dense connections leads to the extraction of a richer set of hierarchical features. However, as expected, the performance improvement resulting from the dense connections is achieved at the expense of a slightly larger number of MACC operations. In order to investigate the impact of using different objective functions for training the network on its performance, we train the network using the $\ell 1$ norm, $\ell 2$ norm and logcosh based objective functions alone, and also using $\ell 1$ norm objective function followed by the $\ell 2$ norm objective function and $\ell 1$ norm

Table 5.15: Performance of SRNHARB and LatticeNet in Restoring Original Images from the Versions with Realistic Degradation.

Network	Performance: PSNR (SSIM)
LatticeNet [16]	27.91 (0.8180)
Proposed SRNHARB	28.58 (0.8321)

objective function followed by the logcosh objective function. Table 5.12 provides the performance results of the network models each obtained by training the network with one of these five training strategies on images of the datasets *Set5*, *Set14*, *BSD100* and *Urban100*, when the scaling factor 4 is used. The results of this table show that training the network with either $\ell 2$ norm based or logcosh objective functions provides a performance that is inferior to that obtained by training the network with the $\ell 1$ norm based objective function. It is also seen from the results of this table that fine-tuning the network that is initially trained with the $\ell 1$ norm based objective function, by using either the $\ell 2$ norm objective function or logcosh objective functions, results in enhancing the network performance significantly further without increasing number of parameters. However, since fine-tuning the network with the logcosh objective function results in a performance that is slightly superior to that with the $\ell 2$ norm, we train our network with the $\ell 1$ norm objective function and fine-tune with the logcosh objective function.

Now, the performance and complexity results of the proposed network on the images of the four benchmark datasets are presented and compared with those of eleven state-of-the-art lightweight networks. Eleven networks that are used in this comparison are SRCNN [1], SCN [2], DRCN [19], DRRN [20], LapSRN [29], CARN [14], SRFBN-S [25], PAN [68], IMDN [54], LatticeNet [16], and A2F-M [69]. The performance results, number of parameters and number of MACC operations of the various networks are given in Table 5.13. In this table, the number of MACC operations is given for an image of size 1280×720 and scaling factor of 4. It is seen from this table that among the eleven networks used for comparison, SRNHARB provides the best PSNR and SSIM values in 17 out of 24 cases

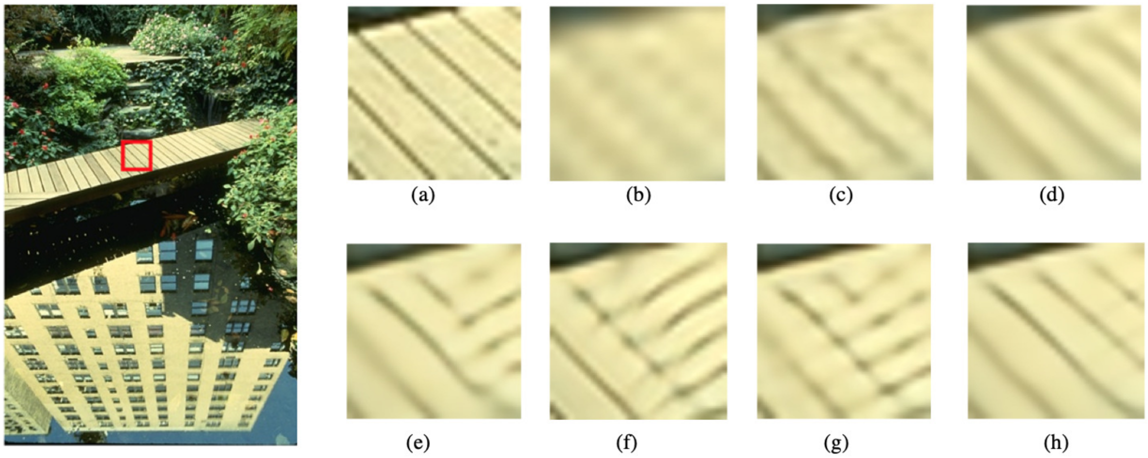


Figure 5.10: Visual quality of images *img021* from *BSD100* super resolved by various schemes with scaling factor 4. (a) Ground truth. (b) Bicubic. (c) SRCNN. (d) DRCN. (e) CARN. (f) IMDN. (g) A2F-M. (h) SRNHARB.

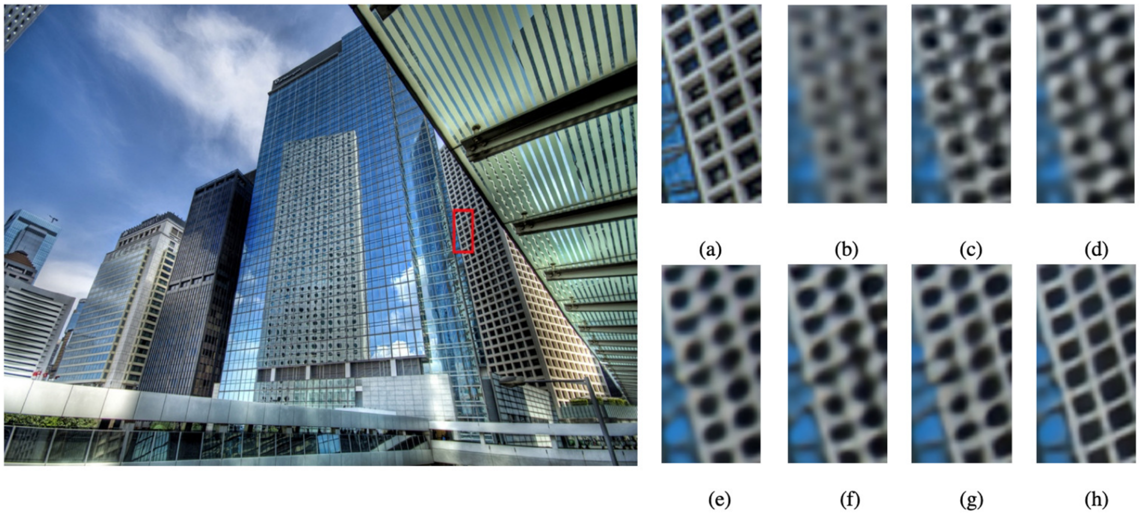


Figure 5.11: Visual quality of images *img061* from *Urban100* super resolved by various schemes with scaling factor 4. (a) Ground truth. (b) Bicubic. (c) SRCNN. (d) DRCN. (e) CARN. (f) IMDN. (g) A2F-M. (h) SRNHARB.

and it gives the second best performance in 3 of the remaining cases. This performance of the proposed network compares with those of LatticeNet and A2F-M, which provide the best and second best values in 8 and 12 cases, and 1 and 6 cases, respectively. Thus, in

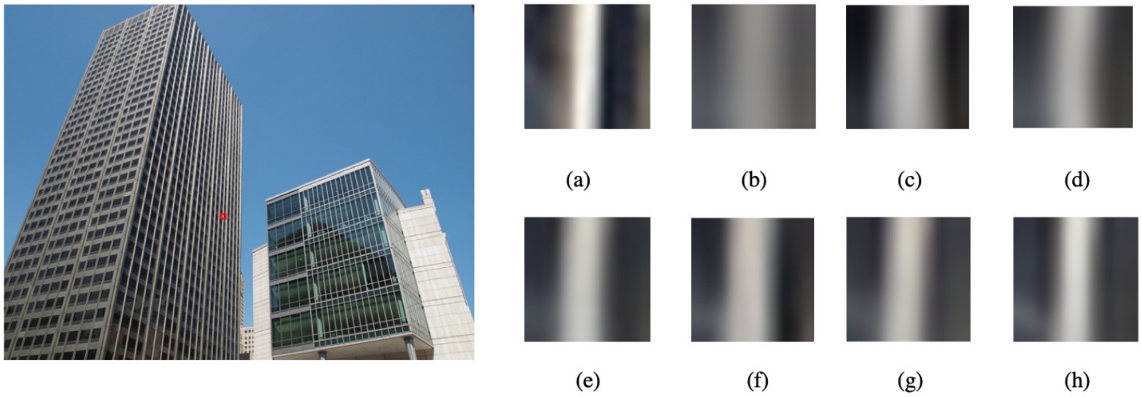


Figure 5.12: Visual quality of images *img096* from *Urban100* super resolved by various schemes with scaling factor 4. (a) Ground truth. (b) Bicubic. (c) SRCNN. (d) DRCN. (e) CARN. (f) IMDN. (g) A2F-M. (h) SRNHARB.

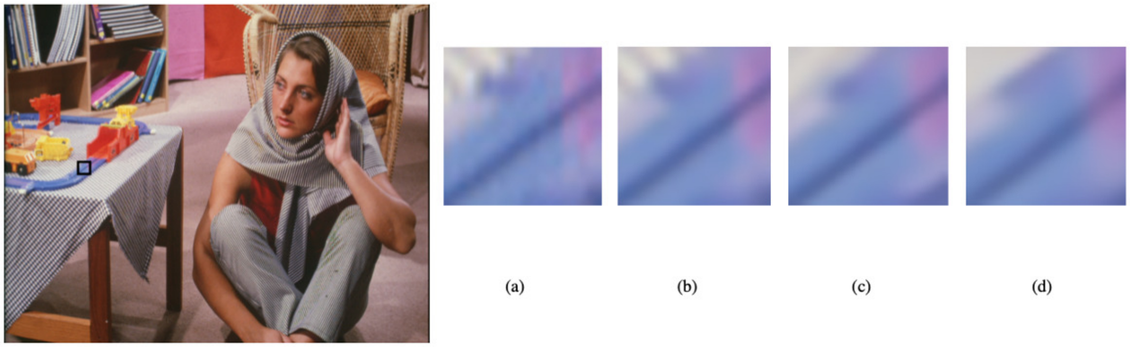


Figure 5.13: Visual quality of *Barbara* image super resolved by the proposed SRNHARB. (a) Ground truth. (b) Upscaling factor 2. (c) Upscaling factor 3. (d) Upscaling factor 4.

terms of PSNR and SSIM values, the proposed SRNHARB, LatticeNet and A2F-M networks can be considered to be the best, second best and third best networks, respectively. It is also seen from this table that among these three networks, in terms of the numbers of parameters and MACC operations, LatticeNet, SRNHARB and A2F-M are, respectively, the best, second best and third best networks. Therefore, if the PSNR and SSIM values and the numbers of parameters and MACC operations are considered simultaneously for evaluating the performance of the networks, then the proposed SRNHARB and LatticeNet can be considered to be comparable and the best performing networks, whereas A2F-M can be considered to be the second best performing network.

Table 5.14 gives the average CPU inference time (in second) and amount of memory consumption (in million bytes), along with the average PSNR values (in dB), for super resolving the images of the *Set5* dataset with the scaling factor 4 by the three best performing light-weight networks. It is seen that the proposed SRNHARB has an inference time that is close to that of LatticeNet, but somewhat smaller than that of A2F-M. However, SRNHARB provides a PSNR performance that is 0.05 dB and 0.07 dB higher than those provided by LatticeNet and A2F-M, respectively. Finally, it should be pointed out that the downside of the proposed network is that it consumes about 18%, 69% more memory than A2F-M and LatticeNet, respectively, does.

In Fig. 5.10, we show the visual quality of the images obtained by super resolving the image *img021* of the *BSD100* dataset, using the best performing light-weight neural networks. The zoom segments of the super resolved images show that the networks CARN [14], IMDN [54] and A2F-M [69] fail to restore the orientations of the ridges on the bridge correctly. On the other hand, the proposed SRNHARB is able to recover these ridges with the same orientations as those of the ground truth image.

In Figs. 5.11 and 5.12, we show the visual quality of the images obtained by super resolving two of the images from the *Urban100* dataset, namely, *img061* and *img096*, using the best performing light-weight super resolution neural networks. The zoom segments of the super resolved images corresponding to *img061* show that only the proposed SRNHARB is able to recover the rectangular windows similar in quality to that of the ground truth image. Similarly, it is seen from the zoom segments of the super resolved images corresponding to *img096* that the proposed SRNHARB results in restoring an image with the best visual quality.

Fig. 5.13 shows the zoomed segment of a part of the toy on the table in the *Barbara* image that is taken from the super resolved images obtained by the proposed SRNHARB with the scaling factors 2, 3 and 4, along with the corresponding segment of that of the ground truth. It is seen from this figure that the proposed SRNHARB is able to recover the edge on the toy properly in each of the three cases of the scaling factors.

So far in our experiments that we have carried out, we have studied the effectiveness of the various light-weight super resolution networks when they are applied on the images that are degraded through the bicubic downsampling operation and shown that when both the performance and complexity are considered together, the proposed SRNHARB and LatticeNet are the two best networks. It would be interesting to evaluate the performance of these two networks when they are applied to the images that are degraded more realistically. For this purpose, we now use the Flicker dataset [51] for training and the validation set from the *DIV2K* dataset [42] for testing of these two networks. The images in these two datasets are degraded using operations similar to the image signal processing methods used by low-end devices for the formation of the images [52]. For the training, sub-images of size 48×48 are randomly selected from the Flicker dataset. Table 5.15 gives the performance of SRNHARB and LatticeNet on the images of the validation set of the *DIV2K* dataset. It is seen from this table that the proposed network has a significant advantage over LatticeNet in terms of both PSNR and SSIM metrics.

5.6 Comparison between Different Deep Image Super Resolution Networks using Feature Fusion Techniques

The performance results of CompNet, FPNet and SRNHARB are compared in Table 5.16. From the results of this table, the following points can be made. First, in view of the fact that CompNet has the simplest network architecture among the three networks, it provides

Table 5.16: PSNR (SSIM) values* resulting from applying various light-weight feature fusing methods to images of three benchmark datasets.

Dataset	Scaling	CompNet [90]	FPNet [85]	SRNHARB [86]
Set5	×2	37.58 (0.9596)	38.13 (0.9616)	38.04 (0.9612)
	×3	33.67 (0.9219)	34.48 (0.9285)	34.55 (0.9289)
	×4	31.35 (0.8833)	32.32 (0.8962)	32.35 (0.8969)
Set14	×2	33.29 (0.9149)	33.83 (0.9198)	33.71 (0.9185)
	×3	30.06 (0.8368)	30.53 (0.8454)	30.63 (0.8474)
	×4	28.26 (0.7732)	28.78 (0.7856)	28.86 (0.7877)
BSD100	×2	31.91 (0.8972)	32.29 (0.9018)	32.25 (0.9013)
	×3	28.84 (0.7995)	29.20 (0.8086)	29.22 (0.8090)
	×4	27.28 (0.7272)	27.66 (0.7394)	27.68 (0.7405)
Urban100	×2	N/A	32.04 (0.9278)	31.84 (0.9254)
	×3	N/A	28.19 (0.8534)	28.33 (0.8561)
	×4	N/A	26.09 (0.7850)	26.15 (0.7888)

∗ The values in the red font indicate the best performance.

the lowest super resolution performance. However, it employs smaller numbers of parameters and operations than the other two networks proposed in this chapter. Second, both the super resolution networks of FPNet and SRNHARB are able to provide very high performance despite the fact that both are light-weigh networks. This can be mainly attributed to their efficient design strategy. Third, the numbers of parameters and operations employed by SRNHARB are slightly lower than those employed by FPNet. Hence, the use of SRNHARB is preferred over that of FPNet in the applications that require high speed super resolution.

5.7 Conclusion

In this chapter, three deep light super resolution networks that efficiently fuse the features obtained by various convolutional layers and residual blocks of the network have been proposed. Based on the results obtained in this chapter, one could argue that all the proposed feature fusion techniques, namely, fusing sparse and representable features, fusing features based on the idea of FPE of adaptive signal processing, and fusing the positive and negative-valued features, are indeed helpful in providing high super resolution performance.

Chapter 6

TPCNN: An Ultralight-weight Three-Prior Convolutional Neural Network for Single Image Super Resolution

6.1 Introduction

The task of image super resolution is crucial in many applications, such as computer vision and medical imaging. Conventionally the task of image super resolution was carried out by formulating it as a constrained optimization problem and then solving it using suitable numerical techniques. However, after the emergence of deep neural networks, the focus of the researchers in this area has been almost entirely on designing deep convolutional neural network architectures that indeed have provided remarkable performance for the task of image super resolution. Even though unified methods of combining the two approaches has a greater potential of providing a superior performance for the task of image super resolution, with the exception of very few works, not much attention has been paid in developing such a unified method for this task. In this chapter, we propose a three-prior formulation of the optimization problem for image super resolution and develop an ultralight-weight convolutional neural network for its solution [102].

6.2 Development of the Proposed TPCNN

In this section, first, two algorithms, namely, the iterative shrinkage and thresholding algorithm (ISTA) [70] and the learned iterative shrinkage and thresholding algorithm (LISTA) [72] that provide iterative schemes for obtaining a closed form expression for an optimization problem with sparsity constraints, are briefly reviewed. These algorithms are concerned with solving an underdetermined system, $\mathbf{u} = \mathbf{A}\mathbf{v}$, where $\mathbf{u} \in \mathrm{I\!R}^M$ and $\mathbf{A} \in \mathrm{I\!R}^{M\times N}$ ($M < N$) are given and $\mathbf{v} \in \mathrm{I\!R}^N$ is a solution vector of the system. There are infinite number of solutions to this problem. One can obtain a transform domain sparse solution for $\mathbf{v}$ by formulating the optimization problem given by

$$\hat{\mathbf{v}} = \underset{\mathbf{v}}{\text{argmin}}(\|\mathbf{u} - \mathbf{A}\mathbf{v}\|_2^2 + \alpha\|T(\mathbf{v})\|_1) \tag{6.1}$$

where $\|.\|_1$ and $\|.\|_2$ denote the $\ell 1$ and $\ell 2$ norms, respectively, α is the regularization parameter, and $T(.)$ represents a sparse transform operator. In [70], an iterative algorithm using gradient descent has been proposed, which is given by

$$\begin{aligned} \mathbf{s}(i) =& \mathbf{v}(i-1) - \mathbf{A}^T\big(\mathbf{A}\mathbf{v}(i-1) - \mathbf{u}\big) \\ \mathbf{v}(i) =& T^{-1}\Bigg(Shr_\alpha\Big(T\big(\mathbf{s}(i)\big)\Big)\Bigg) \end{aligned} \tag{6.2}$$

Table 6.1: Description of Symbols used for Developing the Proposed Image Super Resolution Scheme.

Symbol	Description	Symbol	Description
x	Ground Truth High resolution Image	f_3	Interpolation Function
y	Degraded Low resolution Image	**P**	Low Resolution Feature Maps
H	Blurring Operator	**Z**	High Resolution Feature Maps
D	Downsampling Operator	**Q**	Feature Maps Obtained by Function f_1
f_1	Super Resolution Function	**T**	Degraded Version of the High Resolution Feature Maps Obtained by Function f_2
f_2	Degradation Function	**W**	Interpolated Version of the Low Resolution Feature Maps Obtained by Function f_3

where $Shr(.)$ is the shrinkage operator defined as

$$Shr_\alpha(p) = \begin{cases} sgn(p)(|p| - \alpha) & |p| \geq \alpha \\ 0 & |p| < \alpha \end{cases} \tag{6.3}$$

with $sgn(.)$ denoting the sign function. It should be pointed out that the shrinkage operator used in (6.3) could be replaced by other nonlinear or by piece-wise linear operators.

The learned iterative shrinkage and thresholding algorithm (LISTA) of [72] is a scheme that provides an optimal solution using (6.2) by utilizing a recurrent neural network. In this algorithm, an end-to-end mapping is performed to obtain optimum values for the parameters used in (6.2), which yields a performance that is superior to that given by ISTA. The network of LISTA is categorized as a light-weight network in view of its using RNN. One of the applications of LISTA is the sparse coding network (SCN) [2] used for single image super resolution.

We now develop a three-prior formulation for the optimization problem to carry out the task of single image super resolution. Table 6.1 gives the symbols used in developing the proposed image super resolution scheme along with their corresponding descriptions.

Let $\mathbf{y} \in \mathbb{R}^m$ denote the vector representing the original low resolution image and $\mathbf{x} \in \mathbb{R}^n$ be the corresponding vector representing the desired high resolution image. We wish to obtain a mapping function $f_1 : \mathbb{R}^m \rightarrow \mathbb{R}^n$, which maps the degraded low resolution image **y** to the high resolution image **x** such that we are also able to achieve two

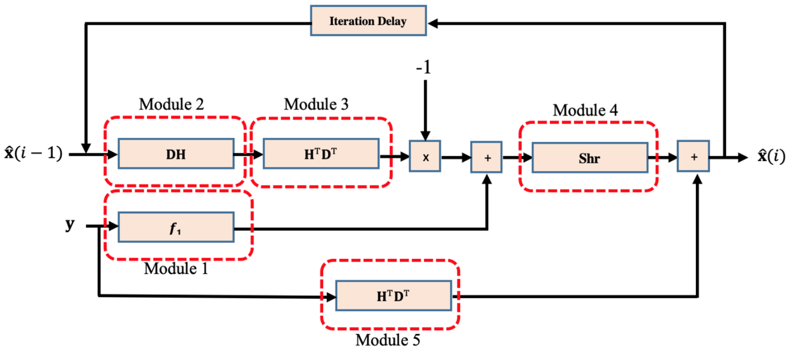


Figure 6.1: A block-level implementation of the proposed iterative shrinkage and thresholding algorithm.

additional goals by using the functions $f_2 : \mathbb{R}^n \rightarrow \mathbb{R}^m$ and $f_3 : \mathbb{R}^m \rightarrow \mathbb{R}^n$ capable of doing the following tasks: the mapping function f_2, is required to provide the low resolution image $\mathbf{y}$, when it is applied to the high resolution image $\mathbf{x}$, whereas the mapping function f_3 is required to provide an interpolated image, whose residue with the desired high resolution image $\mathbf{x}$ is sparse. Thus, the overall formulation of the optimization problem can be expressed as

$$\begin{aligned} \hat{\mathbf{x}} &= \underset{\mathbf{x}}{\operatorname{argmin}}\left(\frac{1}{2}\|\mathbf{x} - f_1(\mathbf{y})\|_2^2\right) \\ &\text{subject to} \begin{cases} \|f_2(\mathbf{x}) - \mathbf{y}\|_2^2 < \epsilon_1 \\ \|\mathbf{x} - f_3(\mathbf{y})\|_0 < \epsilon_2 \end{cases} \end{aligned} \tag{6.4}$$

where ϵ_1 and ϵ_2 are constants with small values. The optimization problem given by (4) is NP-hard and can be relaxed by replacing the $\ell 0$ norm with the $\ell 1$ norm as

$$\begin{aligned} \hat{\mathbf{x}} &= \underset{\mathbf{x}}{\operatorname{argmin}}\left(\frac{1}{2}\|\mathbf{x} - f_1(\mathbf{y})\|_2^2\right) \\ &\text{subject to} \begin{cases} \|f_2(\mathbf{x}) - \mathbf{y}\|_2^2 < \epsilon_1 \\ \|\mathbf{x} - f_3(\mathbf{y})\|_1 < \epsilon_3 \end{cases} \end{aligned} \tag{6.5}$$

where ϵ_3 is a constant with a small value. The Lagrange multipliers allow the above optimization problem to be equivalently formulated as an unconstrained optimization problem given by

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}(\frac{1}{2}\|\mathbf{x} - f_1(\mathbf{y})\|_2^2 + \frac{1}{2}\|f_2(\mathbf{x}) - \mathbf{y}\|_2^2 + \gamma\|\mathbf{x} - f_3(\mathbf{y})\|_1) \tag{6.6}$$

where γ is a regularization parameter. Therefore, the optimization problem as formulated above for the task of image super resolution becomes a three-prior formulation. In the first prior, the super resolution operation between the low and high resolution images is carried out by the function f_1. The second prior models the process of degrading the high resolution image **x** into the low resolution image **y** using the function f_2. Through the third prior, it is ensured that the interpolation operation carried out by f_3 produces an image whose residue with respect to the desired high resolution image **x** is highly sparse. Since the function f_2 is supposed to transform the high resolution image **x** into a low resolution image, we model this degradation operation by using two matrix operators, **H** and **D**. The matrix operator **H** when applied to **y** yields a blurred image, whereas the matrix operator **D** when applied to this blurred image yields the degraded low resolution image **y**. Thus, **DH** can be regarded as a degrading operator encompassing both blurring and downsampling, which when applied to the high resolution **x** results in a degraded low resolution image **y**. This degradation process is in line with the degradation model used in most of the state-of-the-art super resolution schemes, in which the degradation operation is carried out by the bicubic downsampling consisting of two parts: blurring with bicubic kernel and downsampling. The function f_3 is essentially an upgrading function, that is, it must restore the degraded low resolution image to a high resolution image. We realize this operation by applying the transposition of the degrading matrix operator **DH**, i.e., $\mathbf{H}^T\mathbf{D}^T$ to the low resolution image **y**. With the functions f_2 and f_3 so modeled, we can re-write the formulation of the optimization problem given by (6.6) as

$$\begin{aligned}\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}(\frac{1}{2}\|\mathbf{x} - f_1(\mathbf{y})\|_2^2 + \frac{1}{2}\|\mathbf{D}\mathbf{H}\mathbf{x} - \mathbf{y}\|_2^2 + \\ \gamma\|\mathbf{x} - \mathbf{H}^T\mathbf{D}^T\mathbf{y}\|_1)\end{aligned} \tag{6.7}$$

An iterative shrinkage and thresholding algorithm [70] for the solution of the optimization problem in (6.7) can be devised as given below

$$\begin{aligned}\hat{\mathbf{x}}(i) =& Shr_\gamma\Big(\hat{\mathbf{x}}(i-1) - \big(\hat{\mathbf{x}}(i-1) - f_1(\mathbf{y})\big) \\ &- \mathbf{H}^T\mathbf{D}^T\big(\mathbf{D}\mathbf{H}\hat{\mathbf{x}}(i-1) - \mathbf{y}\big) - \mathbf{H}^T\mathbf{D}^T\mathbf{y}\Big) + \mathbf{H}^T\mathbf{D}^T\mathbf{y}\end{aligned} \tag{6.8}$$

where $Shr_\gamma(.)$ is a shrinkage operator. Equation (6.8) can be simplified to give the final iterative formula for solving our optimization problem, as

$$\hat{\mathbf{x}}(i) = Shr_\gamma\big(f_1(\mathbf{y}) - \mathbf{H}^T\mathbf{D}^T\mathbf{D}\mathbf{H}\hat{\mathbf{x}}(i-1)\big) + \mathbf{H}^T\mathbf{D}^T\mathbf{y} \tag{6.9}$$

Equation (6.9) is an iterative solution of the optimization problem of (6.7).

We now develop an ultralight-weight network that implements the algorithm given by (6.9) to provide the super resolved image. We also describe the training details of the proposed ultralight-weight super resolution network. As seen from (6.9), the iterative algorithm has three main parts. The first part in this algorithm concerns obtaining the high resolution image $f_1(\mathbf{y})$ from the degraded image $\mathbf{y}$. The second part concerns applying the degrading operation, as carried out by the operator $\mathbf{DH}$, on the super resolved image $\hat{\mathbf{x}}(i-1)$ obtained in the previous iteration and then carrying out the upgrading operation, as carried out by the operator $\mathbf{H}^T\mathbf{D}^T$, on the resulting image. The third part concerns applying the upgrading operation, as carried out by the operator $\mathbf{H}^T\mathbf{D}^T$, on the degraded image $\mathbf{y}$. By considering these three parts of the algorithm and the interaction between them, we can have its high-level block representation, as shown in Fig. 6.1, where Modules 1, 2, 3, 4

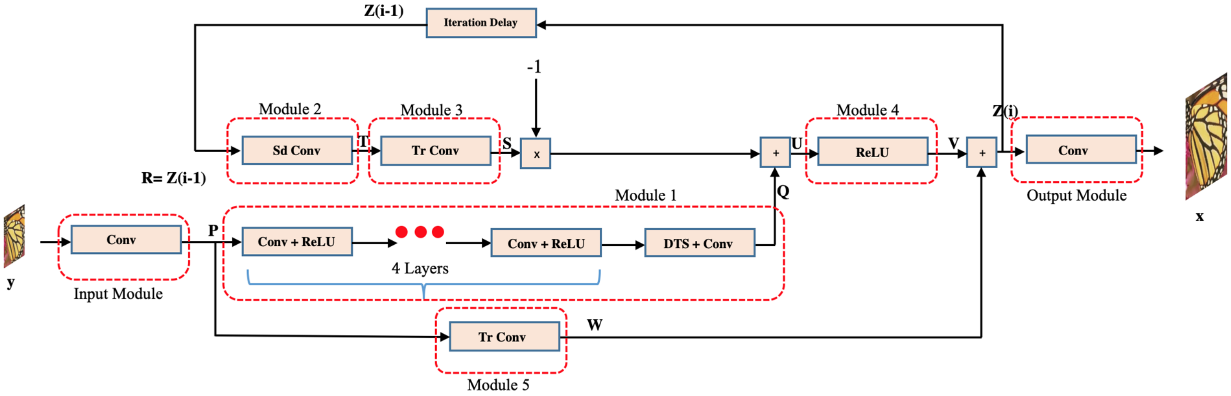


Figure 6.2: A neural network architecture of the proposed three-prior convolutional neural network (TPCNN). Sd Conv, Tr Conv and DTS, respectively, denote strided convolution, transposed convolution and depth-to-space transpose operations.

and 5, correspond, respectively, to the operations f_1, $\mathbf{DH}$, $\mathbf{H}^T\mathbf{D}^T$, Shr_γ and $\mathbf{H}^T\mathbf{D}^T$ in (6.9). We now develop a low-complexity high-performance convolutional neural network architecture, shown in Fig. 6.2, for the implementation of this high-level block representation given in Fig. 6.1. The degraded input image $\mathbf{y}$ is first transformed into the feature maps $\mathbf{P}$ using Input Module. Input Module is implemented by employing a convolution operation using 32 filters each of kernel size 3×3. Next, we focus on implementing the function f_1 of Module 1 by performing on $\mathbf{P}$ the following operations in cascade to produce the feature maps $\mathbf{Q}$: four convolutions, each followed by a ReLU activation, a depth-to-space (DTS) transpose operation [8], and finally another convolution operation. Each of the five convolutional layers, used for the implementation of this module, employs 32 filters with kernel size 3×3. The scaling factor of the depth-to-space transpose operation is equal to the up-scaling factor used for the super resolution. Next, we consider implementing the Modules 2 and 3. Module 2 performs a degrading operation $\mathbf{DH}$, whereas Module 3 performs an up-grading operation $\mathbf{H}^T\mathbf{D}^T$. Here, we relax the relationship between the operations $\mathbf{DH}$ and $\mathbf{H}^T\mathbf{D}^T$ by treating them independently in order to provide an increased degree of freedom to the learning process of the network. Modules 2 and 3 are implemented using, respectively, a strided convolution (St Conv.) and a transposed convolution (Tr Conv.), both employing

32 filters each of kernel size 3×3 and a stride equal to the upscaling factor used for the super resolution. These Modules operate in cascade on the feature maps $\mathbf{R}$ to produce the maps represented by $\mathbf{S}$. Module 4 by performing a shrinkage operation on the feature maps $\mathbf{U}$ produces the maps denoted by $\mathbf{V}$. This module is implemented using the ReLU function. In Experimental Results section, we provide an empirical justification of replacing of the shrinkage operator of (6.3) by ReLU function. Module 5, as Module 3, also performs the upgrading operation $\mathbf{H}^T\mathbf{D}^T$, but on the feature maps $\mathbf{P}$ to produce the feature maps represented by $\mathbf{W}$. For the implementation of this module, we use the same parameters as used for the implementation of Module 3. The feature maps $\mathbf{Z}$, produced after a suitable number of iterations, are made to undergo a convolution operation using Output Module in order to produce the final high resolution image $\mathbf{x}$. The Output Module is implemented using a convolutional layer with 3 filters each of kernel size 3×3.

The network as implemented by the architecture given in Fig. 6.2 is referred to as **T**hree-**P**rior **C**onvolutional **N**eural **N**etwork (TPCNN) [102], since it provides an efficient solution to a three-prior formulated optimization problem for the task of image super resolution.

For the training of the proposed network, the dataset DIV2K [42] that consists of 800 training images is considered. The training samples of size 48×48 are extracted from the images of this dataset. The extracted samples are augmented in order to form the complete training set through their rotations by 90°, 180° and 270° and flipping horizontally. The $\ell 1$ norm of the loss between the ground truth samples and the estimated high resolution samples obtained by applying the proposed network to the degraded low resolution sub-images is used for updating the network parameters. The $\ell 1$ norm loss is minimized by using the method of stochastic gradient descent. The mini-batch size is set to 64. The Keras library [40] and TensorFlow package [41] are employed for implementing the proposed ultralightweight network. The training of the proposed network is carried out on a machine with

Intel Core i7 CPU @4.2 GHz, 16-GB RAM and Nvidia Titan X GPU.

6.3 Experimental Results of TPCNN

In this section, we first investigate the impact of the number of recursions carried out by the architecture of Fig. 6.2 on the super resolution performance. Then, we carry out ablation studies on the proposed super resolution scheme. Next, we investigate the impact of employing shrinkage operators other than ReLU in Module 4 on the network performance. Then, the performance and complexity of the proposed network are compared with those of the state-of-the-art ultralight-weight super resolution networks on the four benchmark datasets, *Set5* [21], *Set14* [22], *BSD100* [23] and *Urban100* [24].

Since the proposed iterative shrinkage and thresholding algorithm given by (6.9) is a recursive algorithm, its neural network implementation shown in Fig. 6.2 has a recursive part. In order to study the impact of the number of recursions on the performance of the network, we obtain the average PSNR values of the super resolved images yielded by the network after 1, 2 and 3 recursions for each of the four evaluation datasets. Table 6.2 gives these results along with the number of multiply-accumulate (MACC) operations. It is seen from this table that there is only a marginal improvement in the PSNR values when the number of recursions is increased from 1 to 2 and almost no improvement when the number of recursions is further increased to 3. However, there is a significant increase in the network complexity associated with these increases in the number of recursions. Specifically, the number of MACC operations increases by one-third and two-thirds when 2 and 3 recursions are, respectively, used over that of using only a single recursion. It should be pointed out that in [2], a similar marginal performance improvement by increasing the number of recursions is observed. In view of this analysis of the results in Table 6.2, we employ only 1 recursion in all our experiments. The proposed formulation of the optimization problem for the task of image super resolution consists of three priors. In order to investigate the

Table 6.2: Impact of Number of Recursions on the Network Performance and MACC Operations.

Network with	*Set5*	*Set14*	*BSD100*	*Urban100*	MACC
1 *Recursion*	31.15	28.12	27.18	24.93	8.74G
2 *Recursions*	31.15	28.15	27.19	24.95	11.63G
3 *Recursions*	31.17	28.15	27.19	24.95	14.52G

impact of each of the three priors on the image super resolution performance, we consider the following three variants of the proposed network.

Variant 1 is the network that results from the solution of the optimization problem using only the first prior. In this case, the solution is given by

$$\hat{\mathbf{x}} = f_1(\mathbf{y}) \tag{6.10}$$

and the architecture of the variant is as depicted in Fig. 6. 3 (a). It is seen from this figure that this architecture is simply that of a shallow convolutional neural network. *Variant 2* is the network that results from solving the optimization problem involving first two priors, that is, in this variant the prior corresponding to the sparsity constraint is removed. In this case, the solution of the optimization problem is given by

$$\hat{\mathbf{x}}(i) = f_1(\mathbf{y}) - \mathbf{H}^T\mathbf{D}^T\mathbf{D}\mathbf{H}\hat{\mathbf{x}}(i-1) + \mathbf{H}^T\mathbf{D}^T\mathbf{y} \tag{6.11}$$

and network that results from this solution is as shown in Fig. 6.3 (b). *Variant 3* of the network is obtained from solving the optimization problem consisting of the first and third priors, i.e., the degradation constraint is not included. The solution of this modified optimization problem is given by

$$\hat{\mathbf{x}} = Shr_\gamma\big(f_1(\mathbf{y}) - \mathbf{H}^T\mathbf{D}^T\mathbf{y}\big) + \mathbf{H}^T\mathbf{D}^T\mathbf{y} \tag{6.12}$$

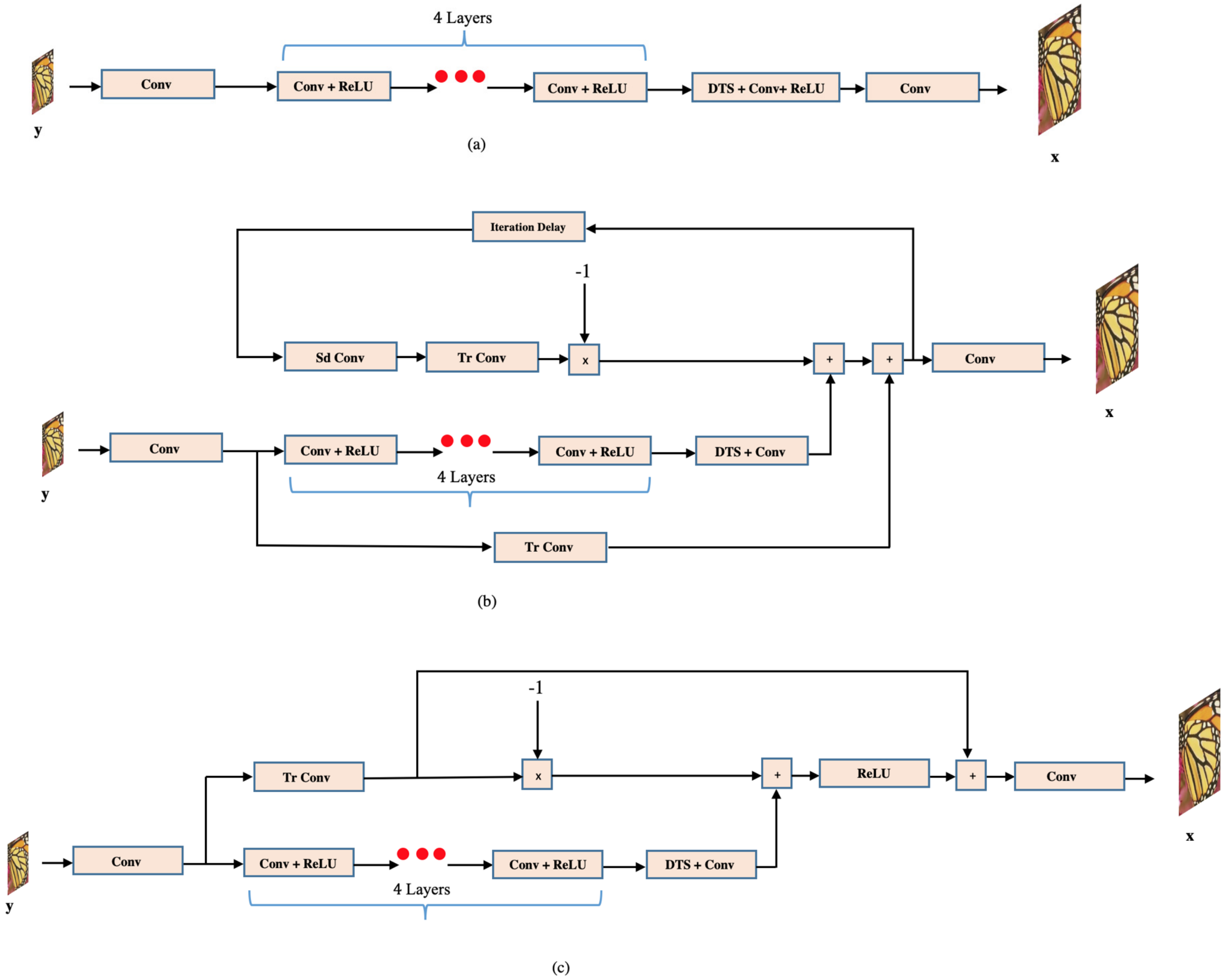


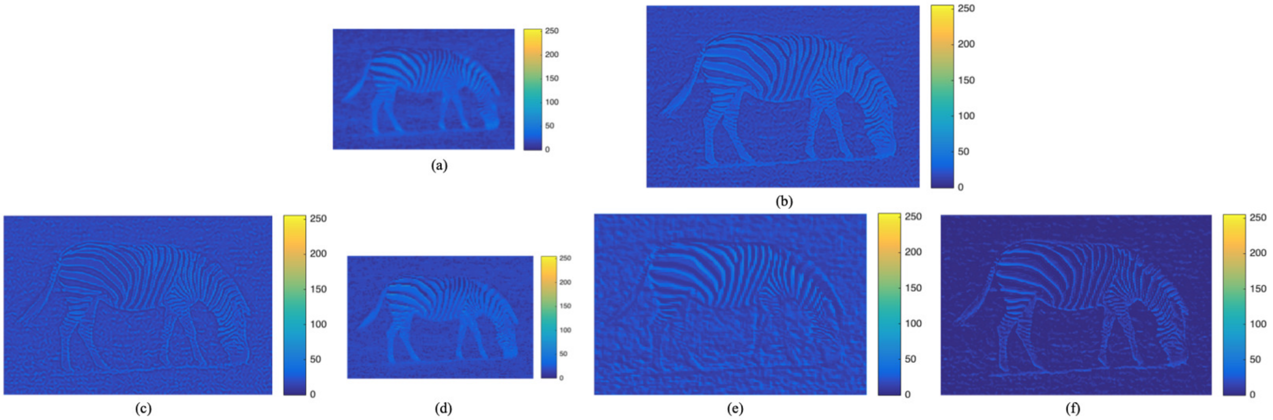
Figure 6.3: Variants of TPCNN using (a) only the first prior (Variant 1). (b) the first and second priors (Variant 2), and (c) the first and third priors (Variant 3).

and the network that results from this solution is shown in Fig. 6.3 (c).

Table 6.3 gives the performance results of the proposed super resolution network and its three variants on the images of the four evaluation benchmark datasets when the scaling factor 4 is used. It is seen from the results of this table that by removing the second or third prior individually or both together from the optimization problem, the performances of the resulting networks significantly degrade. It is also seen from this table that the performance

Table 6.3: PSNR Values of Images Super Resolved by TPCNN and Its Variants.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*
TPCNN Variant 1	30.75	27.81	26.95	24.52
TPCNN Variant 2	31.04	28.05	27.14	24.90
TPCNN Variant 3	30.95	28.02	27.11	24.85
TPCNN	31.15	28.12	27.18	24.93



(a) (b) (c) (d) (e) (f)

Figure 6.4: A Typical Feature Map of (a) Feature Tensor **P**, (b) Feature Tensor **Z**, (c) Feature Tensor **Q**, (d) Feature Tensor **T**, (e) Feature Tensor **W**, and (f) Residue between **Z** and **W**.

of *Variant 1* that uses only the first prior can still further be improved by incorporating in it either of the two other priors.

The function f_1 is supposed to perform an operation on the degraded input low resolution feature maps **P** to produce the feature maps **Q**, which are closer to the high resolution feature maps **Z**, i.e., they should also necessarily have the fine details of the high resolution image. The function f_2 is supposed to carry out an operation on the high resolution feature maps **Z** to generate the feature maps **T**, which are close to the input low resolution feature maps **P**. The function f_3 is supposed to perform an operation on the low resolution feature maps **P** to generate the feature maps **W**, so that the residue between **Z** and **W** is sparse, i.e., the feature maps **W** should necessarily have the course information of the high resolution image. Fig. 6.4 depicts a typical feature map for each of the outputs **P**, **Q**, **T**, **W** and **Z**. The

Table 6.4: Impact of using Different Shrinkage Operators on the Network Performance.

Network with	*Set5*	*Set14*	*BSD100*	*Urban100*
Soft Shrinkage Operator	31.06	28.09	27.15	24.90
Hard Shrinkage Operator	31.03	28.05	27.12	24.88
ReLU	31.15	28.12	27.18	24.93

Table 6.5: PSNR (SSIM) Values Resulting from Applying TPCNN and Different Ultralight-weight State-of-the-art Super Resolution Convolutional Neural Networks to Images of Four Benchmark Datasets.

Dataset	Scaling	Bicubic	SRCNN [1]	SCN [2]	FSRCNN [13]	PISR [46]	DRN UW [35]	DBPN UW [32]	PAN UW [68]	TPCNN (Proposed)
Set5	×2	33.66 (0.9299)	36.66 (0.9542)	36.93 (0.9252)	37.00 (0.9558)	37.33 (0.9576)	37.12 (0.9586)	37.48 (0.9589)	37.43 (0.9591)	37.18 (0.9589)
	×3	30.39 (0.8682)	32.75 (0.9090)	33.10 (0.9136)	33.16 (0.9140)	33.31 (0.9179)	33.36 (0.9184)	33.39 (0.9186)	33.32 (0.9194)	33.41 (0.9189)
	×4	28.42 (0.8104)	30.48 (0.8628)	30.86 (0.8710)	30.71 (0.8657)	30.95 (0.8759)	31.00 (0.8775)	31.10 (0.8787)	31.05 (0.8786)	31.15 (0.8803)
Set14	×2	30.24 (0.8688)	32.42 (0.9063)	32.56 (0.9069)	32.63 (0.9088)	32.79 (0.9105)	32.84 (0.9132)	33.19 (0.9133)	32.81 (0.9131)	32.98 (0.9135)
	×3	27.21(0.7385)	29.28 (0.8209)	29.41 (0.8235)	29.43 (0.8242)	29.57 (0.8276)	29.83 (0.8322)	29.55 (0.8323)	29.84 (0.8342)	29.88 (0.8336)
	×4	26.00 (0.7027)	27.49 (0.7503)	27.64 (0.7578)	27.59 (0.7535)	27.77 (0.7615)	28.01 (0.7681)	28.10 (0.7685)	28.07 (0.7695)	28.12 (0.7707)
BSD100	×2	29.56 (0.8431)	31.36 (0.8879)	31.40 (0.8884)	31.53 (0.8920)	31.65 (0.8926)	31.54 (0.8945)	31.81 (0.8947)	31.59 (0.8942)	31.67 (0.8950)
	×3	27.21 (0.7385)	28.41 (0.7863)	28.50 (0.7885)	28.53 (0.7910)	28.61 (0.7919)	28.62 (0.7941)	28.47 (0.7955)	28.64 (0.7954)	28.68 (0.7956)
	×4	25.96 (0.6675)	26.90 (0.7101)	27.03 (0.7161)	26.98 (0.7150)	27.08 (0.7188)	27.10 (0.7218)	27.15 (0.7227)	27.12 (0.7231)	27.18 (0.7247)
Urban100	×2	26.88 (0.8403)	29.50 (0.8946)	29.52 (0.8970)	29.88 (0.9020)	30.24 (0.9071)	30.22 (0.9096)	30.48 (0.9100)	30.30 (0.9103)	30.30 (0.9094)
	×3	24.46 (0.7349)	26.24 (0.7989)	26.21 (0.8010)	26.43 (0.8080)	26.67 (0.8153)	26.72 (0.8176)	26.67 (0.8191)	26.76 (0.8187)	26.78 (0.8194)
	×4	23.14 (0.6577)	24.52 (0.7221)	24.52 (0.7260)	24.62 (0.7280)	24.82 (0.7393)	24.88 (0.7435)	24.90 (0.7437)	24.87 (0.7444)	24.93 (0.7459)
Number of Parameters		-	57K	33K	13K	13K	70K	58K	70K	52K
Number of MACC Operations		-	52.7G	37.8G	4.6G	4.6G	5.83G	9.33G	5.15G	8.7G

The values in the red font show the best performance and those in the blue font indicate the second best performance.

first row in this figure depicts a typical map of the feature tensor **P**, and a typical map of the feature tensor **Z**. The second row shows a typical map from each of the feature tensors **Q**, **T** and **W** resulting, respectively, from the operations of the functions f_1, f_2 and f_3, as well as a typical map of the residue between **Z** and **W**. The following observations can be made from the feature maps presented in this figure. (i) The feature map of **Q** depicted in Fig. 6.4 (c) contains fine high resolution information similar to that in the high resolution feature map of **Z** shown in Fig. 6.4 (b). This confirms that the function f_1 indeed performs the intended task of mapping a low resolution feature map to the high resolution feature map. (ii) The feature map **T** depicted in Fig. 6.4 (d) is obtained by applying the function f_2

to the feature map **Z**. It is seen from the feature map of **T** that it is very similar to the feature map of **P** (Fig. 6.4 (a)). However, the feature map of **T** of Fig. 6.4 (d) is somewhat sharper than the feature map of **P** of Fig. 6.4 (a), since it is obtained by applying the function f_2 on a sharp high resolution feature map of **Z**. This observation confirms that the function f_2 models the intended operation of degradation. (iii) The feature map of **W** depicted in Fig. 6.4 (e) is obtained by applying the function f_3 to the feature map of **P** (Fig. 4 (a)). Fig. 6.4 (f) shows a residue map obtained by subtracting this feature map of **W** from the corresponding feature map of **Z**. It is seen that this residue map has a dark-tone colormap indicating that it is dominated by small feature values. This observation indicates that the function f_3 used in the sparsity prior successfully implements its intended task.

The iterative shrinkage and thresholding algorithm given by (6.9) for the solution of optimization problem formulated for the proposed super resolution scheme involves a shrinkage operator. This shrinkage operator is performed by Module 4 in our proposed neural network implementation shown in Fig. 6.2. We have used the ReLU function to carry out this operation. We now examine the impact on the network performance by using two other shrinkage operators, namely, the soft shrinkage operator given by (6.3) and the hard shrinkage operator defined as

$$HShr_\alpha(p) = \begin{cases} p & |p| \geq \alpha \\ 0 & |p| < \alpha \end{cases} \tag{6.13}$$

The performance results using these two shrinkage operators along with that by using ReLU in the proposed network are given in Table 6.4. It is seen from this table that the network using ReLU as the shrinkage operator provides the best performance on all benchmark datasets with scaling factor 4.

We now compare the performance and complexity of the proposed network with those of the other ultralight-weight super resolution networks, namely, SRCNN [1], FSRCNN

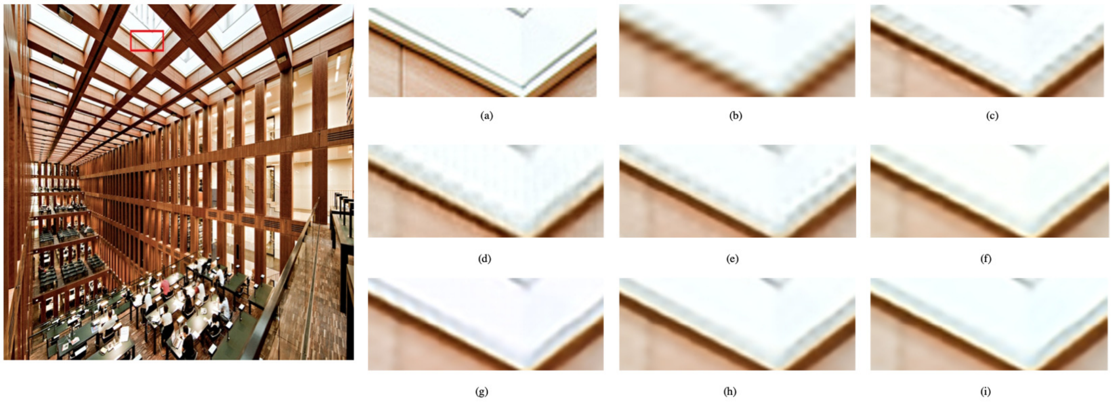


Figure 6.5: Visual quality of the images super resolved by the various schemes when the scaling factor 4 is used on the image *img049* of the *Urban100* dataset. (a) Ground truth. (b) Bicubic. (c) SRCNN. (d) FSRCNN. (e) SCN. (f) DRN UW. (g) PAN (UW). (h) DBPN (UW). (i)TPCNN.

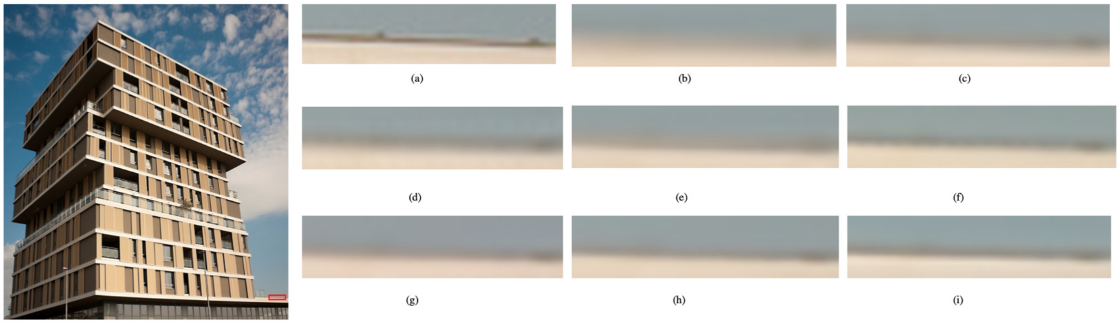


Figure 6.6: Visual quality of the images super resolved by the various schemes when the scaling factor 4 is used on the image *img087* of the *Urban100* dataset. (a) Ground truth. (b) Bicubic. (c) SRCNN. (d) FSRCNN. (e) SCN. (f) DRN UW. (g) PAN (UW). (h) DBPN (UW). (i)TPCNN.

[13], SCN [2], and PISR [46], as well as with those of the ultralight-weight versions of DRN [35], DBPN [32] and PAN [68], on the images with the three different scaling factors for their degradation from the four benchmark datasets. For this performance comparison, we use the ultralight-weight versions of DRN, DBPN and PAN by bringing down their numbers of parameters to the levels that are comparable to that our proposed TPCNN. An ultralight-weight version of DRN [35] is obtained by employing three residual channel-attention blocks, each using convolutional layers with 32 filters of kernel size 3×3. We

obtain an ultralight-weight version of DBPN [32] by using only one up-projection unit and one down-projection unit, each employing convolutional layers with 32 filters of kernel size 3×3. An ultralight-weight version of PAN [68] is obtained by employing only two residual blocks, each using convolutional layers with 32 filters. Table 6.5 gives the performance in terms of PSNR and SSIM, and numbers of parameters and MACC operations of all the ultralight-weight networks. It is seen from this table that the proposed network provides a performance superior to that of all the other networks used for the comparison in 16 out of the 24 cases of the PSNR and SSIM values.

It is to be noted that in [35], the performance of its unsupervised model has been already compared with that of the network of [67], and the performance of this model of the former has been shown to be superior to that of the latter. Therefore, we do not directly compare the performance of our proposed network with that of the network of [67]. Instead, we first obtain an unsupervised model of our proposed TPCNN and that of the ultralight-weight version of the network of [35] through the unsupervised learning process of [35] and compare the performance of these two models. For the training of these two networks, we obtain a set of paired training samples of the original ground truth images and their corresponding bicubically downsampled degraded versions. We also obtain a set of unpaired training samples by applying the different image degradation processes to the ground truth images, including the Gaussian blurring followed by the downsampling. Table VI gives the performance of these two unsupervised models on the images of the four benchmark datasets that are degraded by the Gaussian blurring with the kernel of size 7×7 and the standard deviation of $\sigma = 1.6$ followed by a downsampling operation with the scaling factor 4. It is seen from the results of this table that the unsupervised model of the proposed TPCNN provides a performance that is superior to that provided by the unsupervised model of the ultralight-weight version of the network of [35].

We now compare the visual quality of the images with the scaling factor 4 obtained

Table 6.6: Performance of TPCNN and DRN on the Task of Unsupervised Image Super Resolution.

Network	*Set5*	*Set14*	*BSD100*	*Urban100*
Unsupervised DRN	27.96	25.96	25.76	23.52
Unsupervised TPCNN	28.39	26.33	26.01	23.72

by applying the proposed TPCNN and the other ultralight-weight networks on the images *img049* and *img087* from the *Urban100* dataset. Figs. 6.5 and 6.6 show the visual qualities of the images super resolved by the various networks. It is seen that the images super resolved by the proposed TPCNN have sharper structural details and that the edges of the images super resolved by the other networks have some ringing artifacts.

6.4 Conclusion

In this chapter, our objective has been to design an ultralight-weight super resolution scheme using a shallow convolutional neural network for the task of image super resolution. To achieve this objective, we have first proposed a formulation of the optimization problem involving three priors for the task of image super resolution. The first prior focuses on learning a function that transforms a low resolution image to the high resolution one. The second prior is concerned with learning a degradation model of the high resolution image and the third imposes a sparsity constraint on the residue between the high resolution image and the interpolated low resolution image. The optimization problem thus formulated has been solved by providing an iterative shrinkage and thresholding algorithm. The resulting algorithm has been implemented by developing a neural network architecture that employs small number of parameters and also requires small number of operations. The proposed scheme for image super resolution has been evaluated by performing extensive experiments on four benchmark datasets. It has been shown that each of the three priors used in our formulation of the optimization problem has a significant impact on the performance of the proposed scheme. The proposed scheme has been compared with other ultralight-weight super resolution networks and has been shown to outperform them. Finally, it should be pointed out that the super resolution scheme proposed in this chapter is an ultralight-weight network that employs the least number of parameters among all the super resolution networks proposed in this thesis. Hence, one could choose to use TPCNN in numerous applications with very limited numbers of parameters and operations. On other hand, it is obvious that in view of employing very small number of parameters by TPCNN, its performance is slightly inferior to that of the other proposed networks.

Chapter 7

Deep Joint Image Upsampling and Deblurring Networks

7.1 Introduction

In most of the deep learning-based image restoration schemes, [1],[4], [3], the ground truth image is decimated, but not blurred. However, CCD cameras impart blurring to the captured images, which can be modeled as a convolution operation with a Gaussian point spread function. This blurring phenomena is not fully represented if the ground truth image is degraded only through the bicubic decimation operation. In this chapter, new schemes based on a deep convolutional neural networks for solving the problem of image restoration through upsampling and deblurring are proposed [88], [99], [104]. The networks consists of two stages carrying out the tasks of upsampling and deblurring, respectively.

7.2 UpDCNN: A New Scheme for Image Upsampling and Deblurring using a Deep Convolutional Neural Network

In this section, the architecture of the first proposed scheme for image upsampling and deblurring is described [99]. The deep convolutional network representing the proposed architecture will be referred to as the upsampling-deblurring convolutional neural network (UpDCNN). To the best of our knowledge, UpDCNN is the first scheme based on deep convolutional neural networks that takes both the Gaussian blurring and downsampling operations of image acquisition individually into consideration for restoring the original image.

The overall neural network architecture of the proposed scheme is shown in Fig. 7.1. Denoting the ground truth image by $x[m, n]$ ($0 \leq m \leq aM$ and $0 \leq n \leq aN$) and the degraded image by $z[m, n]$ ($0 \leq m \leq M$ and $0 \leq n \leq N$), the degradation process process can be modeled as

$$\begin{aligned} p[m, n] &= x[m, n] * h[m, n] \\ z[m, n] &= p[am, an] \end{aligned} \tag{7.1}$$

where $h[m, n]$ (assumed to be a Gaussian kernel with the standard deviation σ) is the impulse response representing the blurring phenomenon of the camera lens, $p[m, n]$ is the resulting blurred image and a is the scaling factor. In order to restore the ground truth image $x[m, n]$ from the degraded image $z[m, n]$, the inverse operations, consisting of upsampling and deblurring, should be performed in that order on the blurred downsampled image. Due to the nature of the subsampling operation used in the model given by (7.1), a lossless reconstruction of the ground truth from its blurred downsampled version is not achievable. However, one can expect to obtain a good estimate of the ground truth image from the degraded image $z[m, n]$ using the following approach.

1. The blurred downsampled image $z[m,n]$ is brought to a higher resolution space via the bilinear interpolation as

$$q[m,n] = \begin{cases} z[\frac{m}{a}, \frac{n}{a}] & m,n = 0, \pm a, \pm 2a, ... \\ 0 & \text{otherwise} \end{cases} \tag{7.2}$$
$$u[m,n] = q[m,n] * f[m,n]$$

where $q[m,n]$ is the zero-padded upsampled version of $z[m,n]$, $u[m,n]$ is the upsampled smoothed image and $f[m,n]$ is the smoothing bilinear kernel. Since the subsampling operation is not invertible, the upsampled image would suffer from the ringing effect especially around the edges. If one could succeed in removing the ringing effect completely in the upsampled image, the resulting image would be a blurred version of the ground truth. In practice, this is not possible. However, we can reduce the ringing effect by solving the following optimization problem

$$\begin{aligned} \hat{g} &= \underset{g}{\text{argmin}}(\| \, g(u[m,n]) - p[m,n]\|^2) \\ \hat{p}[m,n] &= \hat{g}(u[m,n]) \end{aligned} \tag{7.3}$$

where $\|.\|$ denotes the $\ell 2$ norm and $g(.)$ represents convolution operations through a sequence of convolutional layers each followed b a ReLU activation function. Thus, the output of stage 1 is $\hat{p}[m,n]$, which is the estimation of the blurred version $p[m,n]$ of the ground truth.

2. The residual signal $r[m,n]$ between the ground truth and the estimation $\hat{p}[m,n]$ of the blurred image is given by

$$r[m,n] = x[m,n] - \hat{p}[m,n] \tag{7.4}$$

We can express this residual as

$$r[m,n] \approx (\hat{p}[m,n] * h^{+}[m,n]) - \hat{p}[m,n] \tag{7.5}$$

where $h^{+}[m,n]$ is the pseudo-inverse of the blurring kernel $h[m,n]$ given by

$$h[m,n] * h^{+}[m,n] \approx \delta[m,n] \tag{7.6}$$

$\delta[m,n]$ being the two-dimensional Kronecker signal. Use of Wiener deconvolution is one of the approaches for obtaining the pseudo-inverse of the blurring kernel. According to this approach, the pseudo-inverse of the blurring kernel $h[m,n]$ in the frequency domain is obtained as

$$H^{+}(e^{j\varphi,j\psi}) = \frac{H^{c}(e^{j\varphi,j\psi})}{H(e^{j\varphi,j\psi})H^{c}(e^{j\varphi,j\psi}) + \tau} \tag{7.7}$$

where c denotes the conjugation operation and $\frac{1}{\tau}$ is SNR of the blurred image $p[m,n]$.

Equation (7.5) can be re-written as

$$\begin{aligned} r[m,n] &\approx (\hat{p}[m,n] * h^{+}[m,n]) - (\hat{p}[m,n] * \delta[m,n]) \\ &\approx \hat{p}[m,n] * (h^{+}[m,n] - \delta[m,n]) \end{aligned} \tag{7.8}$$

Using $d[m,n]$ to denote $h^{+}[m,n] - \delta[m,n]$, an estimation of the ground truth image can be obtained in terms of the blurred image as

$$\begin{aligned} x[m,n] &= r[m,n] + \hat{p}[m,n] \\ &\approx (\hat{p}[m,n] * d[m,n]) + \hat{p}[m,n] \end{aligned} \tag{7.9}$$

The output feature maps of stage 1, which is the estimated blurred image $\hat{p}[m,n]$ image, is fed to the stage 2. This input to stage 2 is passed through a sequence of convolutional layers and the residual signal between the ground truth and blurred image is obtained.

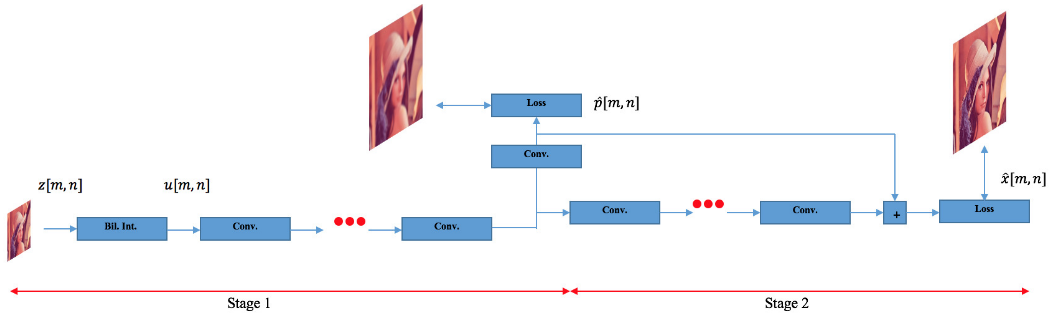


Figure 7.1: Architecture of UpDCNN. In the figure, **Conv.** and **Bil. Int.** refer to convolutional layer and the bilinear interpolation, respectively.

Stage 1 has four convolutional layers. The first convolutional layer in stage 1 is utilized for extracting the features of the bilinear interpolated image. The next two convolutional layers in stage 1 are employed for nonlinear mapping between the interpolated image and the blurred version of ground truth$p[m, n]$. These two layers strive to suppress the ringing effect that is produced by the bilinear interpolation and employ 64 filters with the kernel size of 3×3. The last convolutional layer in stage 1 is used for reconstructing the estimated blurred image and uses 1 filter with the kernel size of 3×3. The mean squared error between the blurred version of the ground truth $p[m, n]$ and the estimated blurred image is considered to the loss function of stage 1 for updating its weights.

Stage 2 consists of 21 convolutional layers with the kernel size 3×3. The first convolutional layer in this stage is employed for feature extraction from the blurred image. The next 19 convolutional layers in stage 2 are dedicated for nonlinear mapping between the blurred image $\hat{p}[m, n]$ and the residual signal $r[m, n]$ between ground truth and the blurred image and the widths of each of these layers are set to 64. The last convolutional layer of stage 2 is devoted to reconstruct the residual image from its features. Each of the convolutional layers in UpDCNN is followed by a batch normalization and a ReLU activation function with the exception of the two reconstruction layers. The mean squared error between upsampled deblurred image $\hat{x}[m, n]$ (i.e. the estimate of the ground truth) and the ground truth $x[m,n]$ is used for updating its weights. The two stages of UpDCNN are jointly trained using the two loss functions.

The sub-images of size 48×48 are extracted from 200 images of *BSD200* dataset [23] and 91 images of the Yang et al. dataset [6] to train the proposed network. Data augmentation involving flipping and rotation is used to increase the number of training samples. Stochastic gradient descent with the momentum parameter of 0.9 and initial learning rate of 0.1 is used to update the weights of UpDCNN in each iteration. The learning rate is decreased by a factor of 10 after every 10 epochs. The total number of 40 epochs is used to update the weights of UpDCNN. The weigh decay parameter is also set to 10^{-4}.

Keras deep learning library [40] and TensorFlow package [41] are employed for implementing the proposed UpDCNN. The training of UpDCNN is carried out by a machine with Intel Core i7 CPU @4.2 GHz, 16 GB installed memory and GPU Nvidia Titan X (Pascal).

7.3 UPDResNN: A Deep Light-Weight Image Upsampling and Deblurring Residual Neural Network

The desire for achieving high-accuracy performance for computer vision tasks has diverted the current design trend of the convolutional neural networks towards very deep architectures at the expense of employing large numbers of parameters and operations. However, this design trend has precluded the deployment of the networks so designed from their applications to many real-world applications that involve mobile devices and portable cameras with their low-power and light-weight requirements. In this section, for solving the joint problem of image upsampling and deblurring, we develop a three-stage light-weight convolutional neural network architecture [88], which tackles this joint problem in an integrated manner through a residual learning guided by simultaneous minimization of two

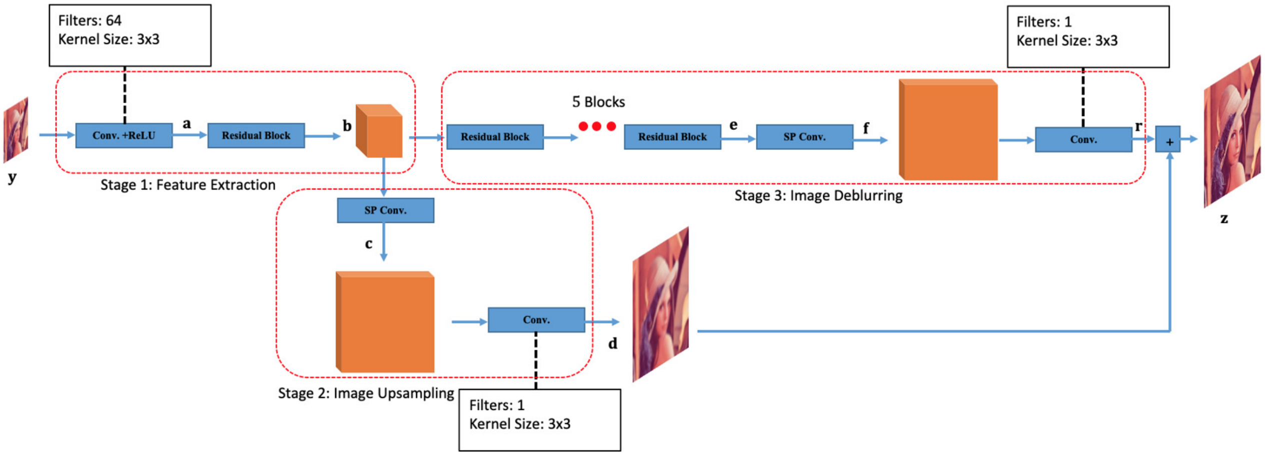


Figure 7.2: The overall architecture of the proposed UpDResNN. Conv. and SP conv., respectively, denote convolution and sub-pixel convolution operations.

loss functions, one representing the difference between the upsampled version of the low quality input image and the blurred version of the ground truth image, and the other one representing the difference between the upsampled deblurred version of the low quality input image and the ground truth image.

Image formation process in an image acquisition system, which produces a low quality

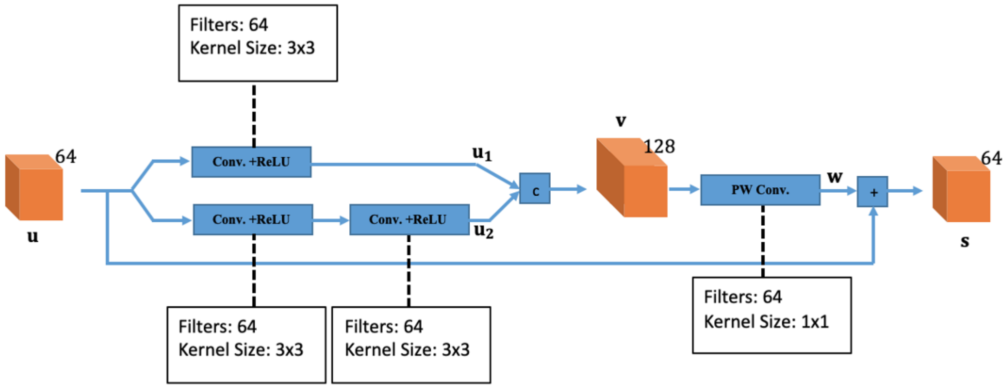


Figure 7.3: Architecture of the proposed residual block of UPDResNN. PW Conv. represents the point-wise convolution operation.

image **y**, can be modeled by blurring followed by a downsampling operation. In order to estimate (restore) the ground truth image **x** from the low quality image **y**, we propose a network consisting of three stages as depicted in Fig. 7.2. *Stage 1* starting from the low quality image **y** generates its feature maps **b**. The other two stages carry out operations of upsampling and deblurring that are converse to that of the image formation process. *Stage 2* produces an estimate of a blurred version of the ground truth (upsampled version **d** of the low quality image) from the feature maps **b**. *Stage 3*, again starting from the feature maps **b**, produces a residue **r** between the ground truth image and the upsampled image **d**. Finally, in this stage, **r** and **d** are combined to produce a high quality estimate of the ground truth image. We now describe in detail the three stages of proposed network.

Stage 1: Feature Extraction. The low quality image **y** is transformed to the YCbCr color space and its luminance content (Y channel) is passed through a convolution operation, a ReLU activation and an operation carried out by one unit of the proposed residual block, to be described in the next paragraphs. The output of this stage are the feature maps **b** given by

$$\mathbf{b} = Res(\underbrace{ReLU(W_1(\mathbf{y}))}_{\mathbf{a}}) \tag{7.10}$$

where W_1 denotes a convolution operation using 64 filters with the kernel size of 3×3 and Res represents the operation carried out by the proposed residual block. The maps **b** are the features of the blurred downsampled low quality image **y**, that must be upsampled and deblurred.

Stage 2: Image Upsampling. The downsampling operation, as performed by the CCD sensors during the image formation process, does not have a corresponding inverse operation. However, through a suitable upsampling operation one can obtain an image that is very close to the blurred version of the ground truth. In this regard, in this stage, the feature maps **b** of the low quality image are first made to undergo a sub-pixel convolution operation yielding feature maps **c** whose spatial resolutions are the same as that of the ground truth image. Finally, the feature maps **c** thus obtained are made to go through a convolution operation producing the upsampled image **d** as

$$\mathbf{d} = W_2(\mathbf{c}) \tag{7.11}$$

where the convolution operation W_2 employs one filter with kernel size of 3×3. In order to force the upsampled version **d** of the low quality input image to be close to the blurred version of the ground truth, we minimize the $\ell 1$ norm loss between these two images.

The upsampled image **d** produced by this stage is thus a blurred version of the ground truth image and it has the approximation content of the ground truth, i.e., its low frequency content. Therefore, it possesses an important information about the ground truth image. On the other hand, it does not have the edges and details of the ground truth. The design of *Stage 3* is, therefore, aimed at restoring this missing information.

Stage 3: Image Deblurring. The desired missing information as mentioned above can be regarded as the residue between the ground truth image **x** and the upsampled image **d**. Therefore, for this stage, we develop a sub-network to realize (learn) this residual signal. In *Stage 3*, the feature maps **b** are fed to a cascade of 5 units of the proposed residual block in order to produce the feature maps **e** as

$$\mathbf{e} = \underbrace{Res(...Res(\mathbf{b})))))}_{\text{5 Units of the residual block}} \tag{7.12}$$

The feature maps **e** are then undergone through a sub-pixel convolution operation yielding the feature maps **f**, which have the same spatial resolutions as that of the ground truth image. Next, the feature maps **f** are made to undergo a convolution operation in order to obtain the residual signal **r** as

$$\mathbf{r} = W_3(\mathbf{f}) \tag{7.13}$$

where the convolution operation W_3 uses one filter with kernel size of 3×3. Finally, the residual signal **r** is added to the upsampled image **d** and the estimated high quality image **z** is obtained as

$$\mathbf{z} = \mathbf{r} + \mathbf{d} \tag{7.14}$$

We now aim at designing a residual block to be used by the proposed light-weight upsampling and deblurring network that is characterized by two attributes. First, the proposed residual block must employ a local skip connection between its input and output in order to facilitate the flow of information in the backpropagation, and therefore, help in curtailing the gradient vanishing problem. Second, the proposed residual block should generate features at multiple receptive fields and fuse them in order to improve the representational capability of the network.

Fig. 7.3 shows the architecture of the proposed residual block. The feature maps **u** input to the block are first passed through a convolution operation followed by a ReLU activation yielding the feature maps $\mathbf{u}_1$ as

$$\mathbf{u}_1 = ReLU(G_1(\mathbf{u})) \tag{7.15}$$

where the convolution operation G_1 employs 64 filters each with kernel size 3×3. The feature maps **u** at the same time are also made to undergo a cascade of two convolution operations each followed by a ReLU activation in order to generate the feature maps $\mathbf{u}_2$ as

$$\mathbf{u}_2 = ReLU(G_3(ReLU(G_2(\mathbf{u})))) \tag{7.16}$$

where the convolution operations G_2 and G_3 use 64 filters each with kernel size 3×3. The net effect of carrying out two 3×3 convolution operations as performed by G_2 and G_3 in cascade is to generate a receptive field of size 5×5 [39]. This same size of receptive field could also be achieved by using a single 5×5 convolution operation. However, the use of a single 5×5 convolution operation would result in larger numbers of parameters and operations than that required by the two 3×3 convolution operations in cascade. The feature maps $\mathbf{u}_1$ and $\mathbf{u}_2$ are generated in 3×3 and 5×5 receptive fields, respectively. Therefore, a fusion of $\mathbf{u}_1$ and $\mathbf{u}_2$ using their concatenation followed by a point-wise convolution would result in producing feature maps with an enhanced representational capability. The operation of fusing $\mathbf{u}_1$ and $\mathbf{u}_2$ that yields the feature maps $\mathbf{w}$ can be described as

$$\begin{aligned} \mathbf{v} &= Concat([\mathbf{u}_1, \mathbf{u}_2]) \\ \mathbf{w} &= G_4(\mathbf{v}) \end{aligned} \tag{7.17}$$

where the point-wise convolution operation G_4 uses 64 filters each with kernel size 1×1. Finally, the feature maps $\mathbf{w}$ are added to the feature maps $\mathbf{u}$ input to the block in order to obtain the block's output feature maps $\mathbf{s} = Res(\mathbf{u})$ as

$$\mathbf{s} = \mathbf{u} + \mathbf{w} \tag{7.18}$$

We refer to the proposed joint image upsampling and deblurring network as deep lightweight image **U**psampling and **D**eblurring **Res**idual **N**eural **N**etwork (UPDResNN) [88].

The proposed joint image upsampling and deblurring network is trained using the images from *BSD 200* dataset [23] and *Yang et al.* dataset [6]. These images are then divided into 111320 sub-images each of size 48×48. Data augmentation, including flipping and rotations by 90, 180 and 270 degrees, is employed to increase the number of training samples to 445280. For evaluating the proposed network, we use four testing benchmark datasets, namely, *Set 10* [73], *Set 5* [21], *Set 14* [22] and *BSD 100* [23] datasets. It should be mentioned that the *Woman* image is contained in both the *BSD 200* training dataset and the *Set 5* test set. Also, the *Parthenon* image is contained in both the *BSD 200* training dataset and the *Set 10* test set. Therefore, in order to avoid having overlap between the training and testing datasets, we have removed the *Woman* and *Parthenon* images from *BSD 200* training dataset.

In order to train the proposed joint image upsampling and deblurring network, we employ two loss functions, one for image upsampling and the other one for image deblurring. The loss function used for the output of the image upsampling stage, *Stage 2*, is defined as

$$L_1(\boldsymbol{\Theta_1}, \boldsymbol{\Theta_2}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{d}_i(\boldsymbol{\Theta_1}, \boldsymbol{\Theta_2}) - \mathbf{h} * \mathbf{x}_i\|_1 \tag{7.19}$$

where $\boldsymbol{\Theta_1}$ and $\boldsymbol{\Theta_2}$ are the sets of parameters used by *Stage 1* and *Stage 2*, respectively, and, $\mathbf{x}_i$ and $\mathbf{d}_i$ are, respectively, the ith samples of the ground truth and upsampled estimation of the input image, $\mathbf{h}$ is a Gaussian blurring kernel used to obtain a blurred version of the ground truth image and N denotes the batch size used in each iteration for updating the network weights. The loss function used for the output of the image deblurring stage, *Stage 3*, is defined as

$$L_2(\boldsymbol{\Theta_1}, \boldsymbol{\Theta_2}, \boldsymbol{\Theta_3}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{z}_i(\boldsymbol{\Theta_1}, \boldsymbol{\Theta_2}, \boldsymbol{\Theta_3}) - \mathbf{x}_i\|_1 \tag{7.20}$$

where $\boldsymbol{\Theta_3}$ is the set of parameters used for *Stage 3* and $\mathbf{z}_i$ is the ith sample of the estimated image.

The loss functions given by (7.19) and (7.20) are minimized using the stochastic gradient descent optimizer. The initial learning rate is set to 0.1 and then is decreased by a factor of 10 after each 10 epochs. A total of 40 epochs are used to train the network. The weights of the network are initialized with the method proposed by He et al. [7]. The weight decay parameter of the convolution operations is set to 10^{-4}.

Keras deep learning library [40] and TensorFlow package [41] are employed for implementing the proposed image upsampling and deblurring network. The training of the proposed network is carried out by a machine with Intel Core i7 CPU @4.2 GHz, 16 GB installed memory and GPU Nvidia Titan X (Pascal).

7.4 Experimental Results

7.4.1 Experimental Results of UpDCNN

In this section, the performance of UpDCNN is obtained and compared with that of the state-of-the-art schemes for image upsampling and deblurring. Also, the effect of changing the hyper-parameters on the performance of UpDCNN is investigated.

For evaluating the performance of UpDCNN, the benchmark of *Ten Images* [73] is used. The Gaussian kernels with standard deviations of $\sigma = 1$ and $\sigma = 1.6$ are utilized for the purpose of blurring and the bicubic decimation with an scaling factor of 3 is used for subsampling. UpDCNN is compared with the state-of-the-art schemes for image upsampling and deblurring, including the sparse coding network (SCN) [2], the centralized sparse representation for image restoration (CSR) [73] and the image super resolution with non-local means (NLM) [74].

The comparison between the various schemes, in the case of upscaling factor 3 and Gaussian blurring kernels $\sigma = 1$ and $\sigma = 1.6$ on *Ten Images* dataset [2], is shown in Table 7.1. As seen from this table, UpDCNN outperforms the other schemes in terms of PSNR. It is worth noting that the difference between the performances of UpDCNN and SCN in the case of the Gaussian blurring kernel with $\sigma = 1$ is 0.49 dB.

To verify the functionality of the two stages of the UpDCNN, their outputs are shown in Fig. 7.6 (e) and (d), respectively, when the network is fed with the blurred downsampled *Butterfly* image. As seen from Fig. 7.6 (e), the image resulting from stage 1 has less ringing artifact with respect to using only bilinear or bicubic interpolated images shown in Fig. 7.6 (c) and (d), respectively. Also, as seen from Fig. 7.6 (f), the final estimated image of UpDCNN has a very pleasant quality with no ringing or blurring artifacts.

Table 7.1: The performance of UpDCNN and the state-of-the-arts on the *Ten Images* that are blurred with Gaussian kernel and downscaled by factor of 3.

Blurring Kernel	CSR	NLM	SCN	UpDCNN
$\sigma = 1$	29.32	29.26	29.65	30.14
$\sigma = 1.6$	29.63	28.25	29.90	29.97

Table 7.2: The performance of stage 1 of UpDCNN, when the standard deviation of the Gaussian blurring kernel is $\sigma = 1$.

Method of Upsampling	Bilinear Upsampling	Bicubic Upsampling	UpDCNN Upsampling
PSNR	22.33	23.13	24.99

Table 7.3: The performance of UpDCNN, when a shallower network is used for stage 1.

Blurring Kernel	UpDCNN ($S_1 = 2$)	UpDCNN ($S_1 = 4$)
$\sigma = 1$	30.14	30.14
$\sigma = 1.6$	29.94	29.97

Table 7.4: The performance of UpDCNN, when a shallower network is used for stage 2.

Blurring Kernel	UpDCNN ($S_2 = 11$)	UpDCNN ($S_2 = 21$)
$\sigma = 1$	29.84	30.14

In order to investigate the functionality of stage 1 of UpDCNN, the PSNRs of the bilinear and the bicubic interpolated images and the image obtained from stage 1 of UpDCNN are given in Table 7.2. As seen from this table, stage 1 of UpDCNN provides a PSNR, which is significantly higher than those provided by the bilinear and the bicubic interpolations. It is worth noting that the difference between the performances of stage 1 of UpDCNN and the bilinear interpolation alone is 2.66 dB, which is significant. This signifies that stage 2 of UpDCNN starts with a better initial point. These results demonstrate the importance of the stage 1 in image upsampling and deblurring via convolutional neural networks.

To see the impact of changing the hyperparameters on the performance of UpDCNN, the number of convolutional layers (S_1) in the stage 1 is decreased from 4 to 2. The performances of UpDCNN with the default and new settings are given in Table 7.3. As seen from this table, even though the deeper network for stage 1 leads to a better performance

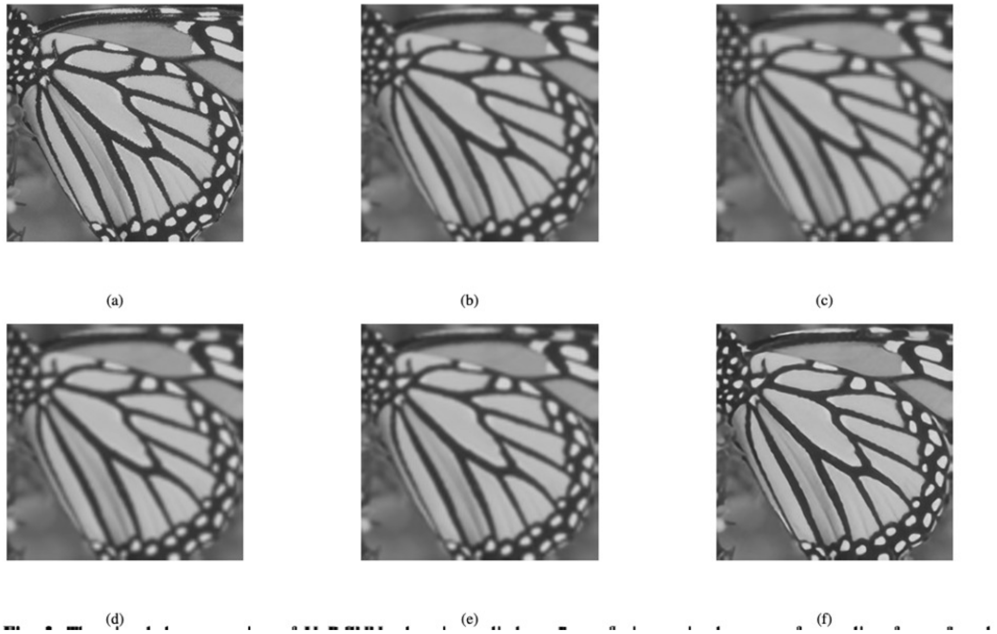


Figure 7.4: The visual demonstration of UpDCNN when is applied on *Butterfly* image in the case of upscaling factor 3 and Gaussian blurring kernel with $\sigma = 1.6$. (a) Ground truth. (b) Blurred image. (c) Bilinear upsampling of the blurred downsampled image. (d) Bicubic upsampling of the blurred downsampled image. (e) The image obtained from stage 1 of UpDCNN. (e) The image obtained from stage 2 of UpDCNN.

of UpDCNN, performance gain is not significant.

Finally, the performance of UpDCNN in the case of using a shallower network for stage is given in Table 7.4. The standard deviation of the Gaussian blurring kernel is set to $\sigma = 1$ and the number of convolutional layers (S_2) in the stage 2 is reduced from 21 to 11. As seen from this table, when the number of layers is decreased in stage 2, the performance of the scheme drops considerably. This shows the importance of utilizing a deep model for stage 2.

7.5 Experimental Results of UpDResNN

In this section, the results of the experiments carried out on the proposed light-weight image upsampling and deblurring network are presented. First ablation studies on the network are conducted. Next, in order to evaluate the effectiveness of *Stage 1* and *Stage 2* in providing an upsampled version of the low quality image, the quality of the output image **d** produced by *Stage 2* is compared with those produced by the classical methods of image upsampling. Finally, the performance and complexity of the proposed network employing the residual block of Fig. 7.3 are presented.

The purpose of *Stage 2*, in conjunction with *Stage 1* in the proposed network is to learn the residue between the ground truth and its blurred version **d**. To investigate the impact of *Stage 2* on the network performance, we form a variant of the proposed network, namely, *Variant 1*. The architecture of this variant is shown in Fig. 7.5. It is seen from this figure that the architecture of *Variant 1* is obtained by replacing the operation of *Stage 2* of the proposed network by the bilinear interpolation operation. Therefore, the residual learning in *Variant 1* becomes the conventional global residual learning instead of the global residual learning performed in the proposed joint image upsampling and deblurring network using *Stage 2*. It should be pointed out that since the network of *Variant 1* employs 6 residual blocks, it has the same number of parameters as that of the proposed image upsampling and deblurring network. Table 7.5 gives the performance of the proposed network and *Variant 1* on the *Set 10* [73], *Set 5* [21], *Set 14* [22] and *BSD 100* [23] images with the scaling factor 3 and blurring kernel standard deviation $\sigma = 1$. It is seen from this table that removing *Stage 2* from the proposed network results in degrading the network performance significantly. The reason for the residual learning performed by the proposed network to be superior to that of *Variant 1* (conventional global residual learning) can be provided as follows.

As seen from *Variant 1*, in the conventional global residual learning, the residual

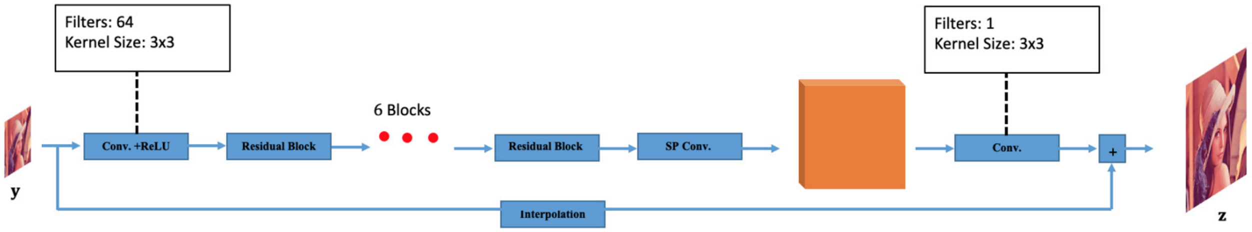


Figure 7.5: Architecture of *Variant 1* of the proposed UPDResNN.

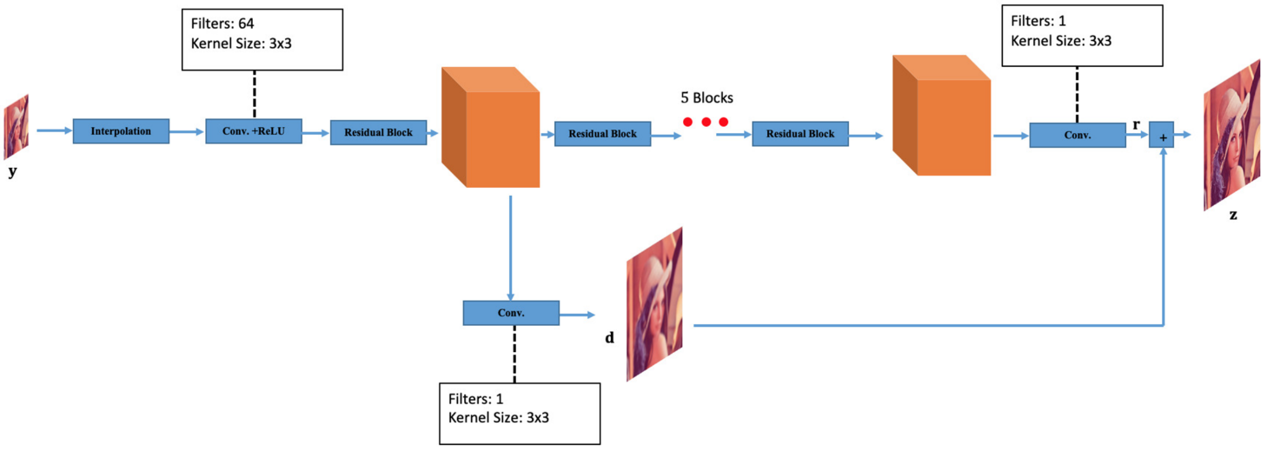


Figure 7.6: Architecture of *Variant 2* of the proposed UPDResNN.

Table 7.5: Impact of employing *Stage 2* on the network performance of UpDResNN in terms of PSNR in dB, when the scaling factor is 3 and the blur kernel standard deviation is $\sigma = 1$.

Network with	*Set 10*	*Set 5*	*Set 14*	*BSD100*
Variant 1	28.51	31.72	28.78	27.92
Proposed	30.27	33.57	29.95	28.77

signal consists of the difference between the ground truth image and the upsampled interpolated version of the low quality degraded input image. Therefore, this residual signal contains the interpolation artifacts. Since the downsampling part of the image degradation

Table 7.6: Impact of using sub-pixel convolutions on the network performance of UpDResNN in terms of PSNR in dB, when the scaling factor is 3 and the blur kernel standard deviation is $\sigma = 1$.

Network with	*Set 10*	*Set 5*	*Set 14*	*BSD100*
Variant 2	30.07	33.32	29.88	28.71
Proposed	30.27	33.57	29.95	28.77

Table 7.7: Impact of using $\ell 1$ norm loss on the network performance of UpDResNN in terms of PSNR in dB, when the scaling factor is 3 and the blur kernel standard deviation is $\sigma = 1$.

Network with	*Set 10*	*Set 5*	*Set 14*	*BSD100*
Variant 3	30.05	33.34	29.89	28.72
Proposed	30.27	33.57	29.95	28.77

Table 7.8: Performance in terms of PSNR in dB of the upsampling stage of the proposed UpDResNN.

Method	Bilinear Upsampling	Bicubic Upsampling	Upsampling Sub-network
Performance	26.02	26.57	28.03

cannot be modeled by the converse operation of image interpolation, in the case of conventional global residual learning, these artifacts cannot be learnt from the ground truth image. In the proposed network, on the other hand, the upsampling of the low quality degraded input image is carried out through *Stage 1* and *Stage 2* leading to the high resolution blurred version of the ground truth image. Therefore, in this case, the residual signal, which is the difference between the ground truth image and its blurred version, contains the high frequency components of the ground truth image without any interpolation artifact. Hence, the downsampling part of the image degradation is better modeled by the converse operation carried out by the *Stage 1* and *Stage 2* of the proposed network.

It is to be noted that in the proposed network, the deblurring part of the image restoration is carried out in low resolution and the resolution of the resulting residue is then raised to that of the ground truth image only at the end of this stage. In order to study the advantage of this approach used by *Stage 3*, we form a variant of the proposed network, namely,

Table 7.9: PSNR (SSIM) values resulting from applying UpDResNN and various state-of-the-art methods to images of four benchmark datasets.

Dataset	Scaling Factor and Blur Kernel SD	DB [36]+ SRCNN [1]	DB [36]+ EDSR [28] (Light-weight)	DB [36]+ MSRN [75] (Light-weight)	GFN [37] (Light-weight)	ADSN [38] (Light-weight)	Proposed
Set10	$\times 3$ and $\sigma = 1$	29.51 (0.8555)	29.96 (0.8649)	29.92 (0.8644)	30.05 (0.8665)	30.08 (0.8666)	30.27 (0.8689)
	$\times 3$ and $\sigma = 1.6$	29.41 (0.8522)	29.92 (0.8636)	29.97 (0.8631)	30.01 (0.8644)	29.98 (0.8636)	30.19 (0.8672)
	$\times 4$ and $\sigma = 1$	27.44 (0.7910)	27.78 (0.8003)	27.77 (0.8001)	27.76 (0.8002)	27.77 (0.8001)	27.78 (0.8004)
	$\times 4$ and $\sigma = 1.6$	27.33 (0.7909)	27.76 (0.8015)	27.73 (0.8020)	27.72 (0.8026)	27.70 (0.8018)	27.97 (0.8050)
Set5	$\times 3$ and $\sigma = 1$	32.78 (0.9106)	33.25 (0.9169)	33.21 (0.9168)	33.31 (0.9177)	33.46 (0.9176)	33.57 (0.9188)
	$\times 3$ and $\sigma = 1.6$	32.67 (0.9089)	33.21 (0.9160)	33.32 (0.9160)	33.34 (0.9172)	33.35 (0.9164)	33.57 (0.9186)
	$\times 4$ and $\sigma = 1$	30.53 (0.8688)	30.87 (0.8741)	30.85 (0.8741)	30.82 (0.8740)	30.90 (0.8743)	30.99 (0.8753)
	$\times 4$ and $\sigma = 1.6$	30.37 (0.8682)	30.91 (0.8762)	30.95 (0.8769)	30.83 (0.8765)	30.96 (0.8761)	31.27 (0.8783)
Set14	$\times 3$ and $\sigma = 1$	29.55 (0.8259)	29.79 (0.8316)	29.75 (0.8312)	29.79 (0.8326)	29.80 (0.8318)	29.95 (0.8338)
	$\times 3$ and $\sigma = 1.6$	29.50 (0.8235)	29.79 (0.8311)	29.85 (0.8307)	29.83 (0.8314)	29.77 (0.8305)	29.91 (0.8329)
	$\times 4$ and $\sigma = 1$	27.77 (0.7600)	27.90 (0.7657)	27.87 (0.7653)	27.84 (0.7650)	27.89 (0.7651)	28.00 (0.7658)
	$\times 4$ and $\sigma = 1.6$	27.68 (0.7606)	28.01 (0.7676)	27.97 (0.7679)	27.91 (0.7680)	27.94 (0.7672)	28.14 (0.7684)
BSD100	$\times 3$ and $\sigma = 1$	28.48 (0.7893)	28.65 (0.7946)	28.61 (0.7939)	28.65 (0.7955)	28.68 (0.7962)	28.77 (0.7969)
	$\times 3$ and $\sigma = 1.6$	28.46 (0.7866)	28.66 (0.7943)	28.70 (0.7937)	28.70 (0.7945)	28.68 (0.7947)	28.78 (0.7968)
	$\times 4$ and $\sigma = 1$	26.98 (0.7162)	27.10 (0.7210)	27.08 (0.7209)	27.06 (0.7209)	27.08 (0.7210)	27.14 (0.7215)
	$\times 4$ and $\sigma = 1.6$	26.94 (0.7186)	27.15 (0.7241)	27.11 (0.7246)	27.10 (0.7248)	27.12 (0.7242)	27.23 (0.7238)

The values in the red font indicate the best performance and those in the blue font represent the second best performance.

Table 7.10: Complexity of UpDResNN and various schemes used for comparison.

Method	Number of Parameters	Number of MACC
DB [36]+SRCNN [1]	446K	56.9G
DB [36]+EDSR [28] (Light-weight)	1168K	55.2G
DB [36]+MSRN [75] (Light-weight)	906K	53.4G
GFN [37] (Light-weight)	1487K	85.0G
ADSN [38] (Light-weight)	942K	42.5G
Proposed	708K	41.1G

Variant 2. The architecture of *Variant 2* is shown in Fig. 7.6. It is seen from this figure that the idea behind *Variant 2* is to make all the processing of the network to be done in high resolution instead of the low resolution as done in the proposed network. In order to realize this idea, the *Variant 2* is formed by removing the sub-pixel convolution operations from

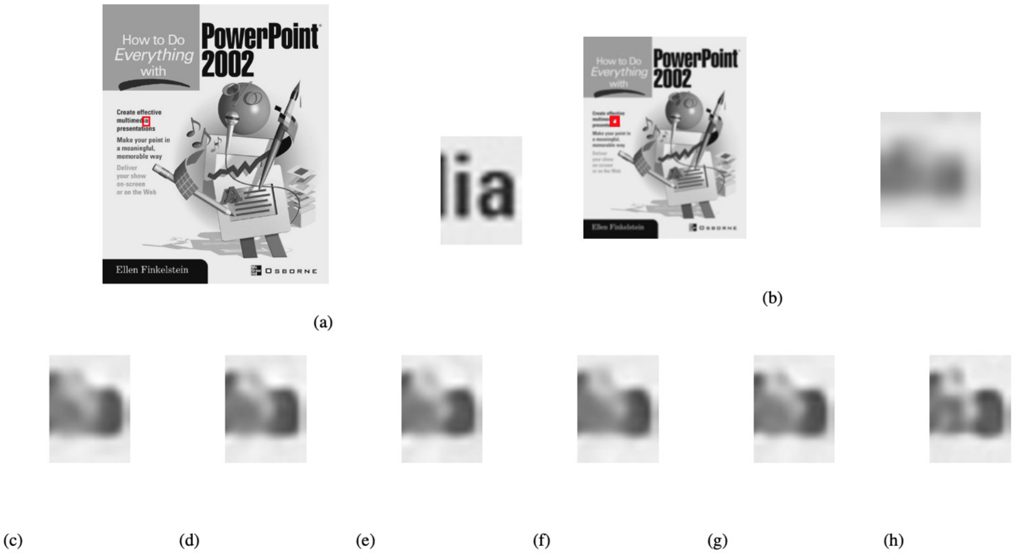


Figure 7.7: Visual comparison of *Powerpoint* image from *Set 14* dataset upsampled and deblurred by the proposed network and light-weight versions of the state-of-the-art schemes with the scaling factor 3 and blurring kernel standard deviation $\sigma = 1.6$. (a) Ground truth. (b) Degraded image. (c) DB [36]+SRCNN [1]. (d) DB [36]+EDSR [28] (Light-weight). (e) DB [36]+MSRN (Light-weight). (f) GFN (Light-weight). (g) ADSN (Light-weight). (h) UPDResNN.

Stage 2 and *Stage 3* of the proposed network and add a bilinear interpolation operation at the beginning of the architecture. Table 7.6 gives the performance, in terms of PSNR in dB, of the proposed network and *Variant 2* on the four benchmark datasets. The results of this table confirm that replacing the sub-pixel convolution by the bilinear interpolation results in a significant performance degradation.

The proposed image upsampling and deblurring network is trained using the $\ell1$ norm loss function. We now study the impact of replacing the $\ell1$ norm loss function by the $\ell2$ norm loss function. We call the network using $\ell2$ norm loss function as *Variant 3*. The architecture of *Variant 3* is the same as that of the proposed network, except the former is trained using the $\ell2$ norm loss function, whereas the latter is trained using the $\ell1$ norm loss. Table 7.7 gives the performance, in terms of PSNR in dB, of the proposed network and *Variant 3* on the four benchmark datasets. It is seen from this table that using the $\ell2$

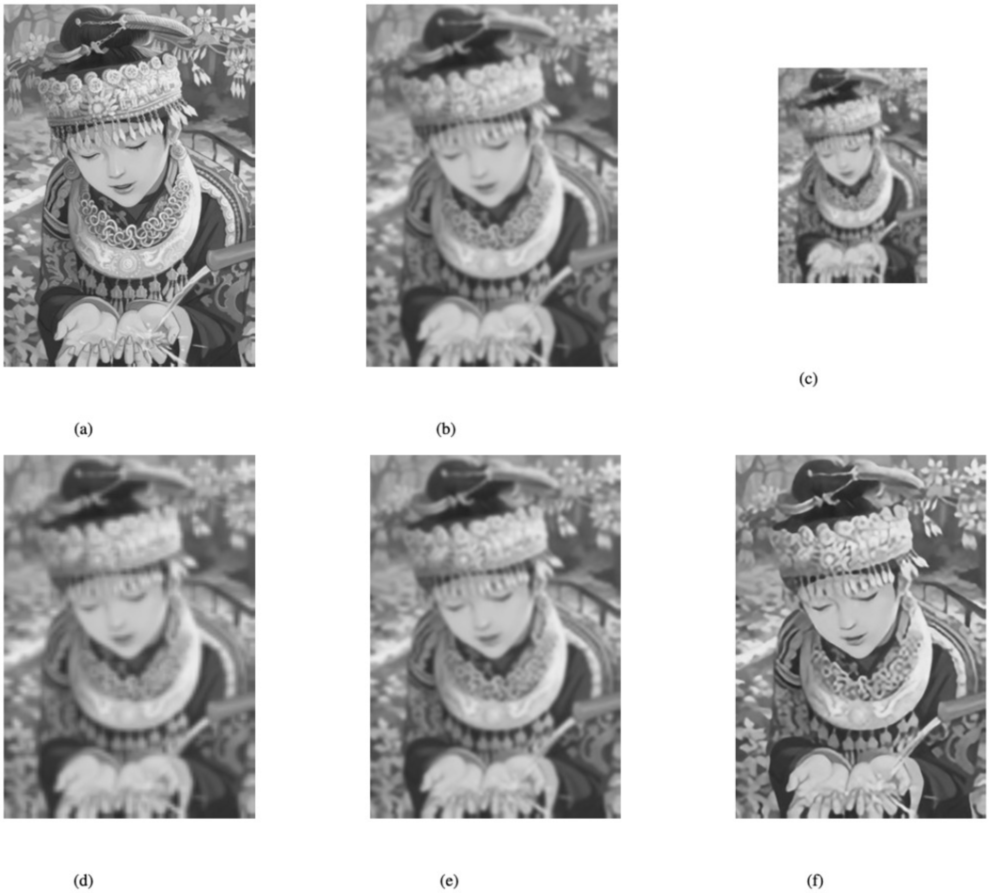


Figure 7.8: *Comic* images resulting from UPDResNN. (a) Ground truth. (b) Blurred Image. (c) Blurred and downsampled image. (d) Bilinear Upsampled image. (e) Upsampled image by the proposed network. (f) Upsampled and deblurred image by the proposed network. Please zoom in to see the details.

norm loss function for training the proposed network results in degrading its performance significantly.

Stage 2 of the proposed network produces a high resolution image **d** corresponding to the low quality image **y**. We now compare in Table 7.8, the quality of the image **d** with those obtained by upsampling **y** using bilinear and bicubic interpolations. It is seen from this table that the of quality (in terms of PSNR in dB) of the image provided by the upsampling stage of the proposed scheme is significantly superior to that obtained by using the classical image upsampling methods.

Our main goal in the design of the proposed UpDResNN is to provide a good performance for the task of joint image upsampling and deblurring by employing a small number of parameters. We now form three image upsampling and deblurring networks by cascading a light-weight version of the deblurring network of [36] with SRCNN [1] and with light-weight versions of EDSR [28] and MSRN [75]. Additionally, we form the light-weight versions of the two joint image upsampling and deblurring networks, namely, GFN [37] and ADSN [38]. The purpose of forming light-weight versions of the networks for the task of joint image upsampling and deblurring is to make their levels of complexity to be approximately the same as that of the proposed network in order to make a fair comparison. In the deblurring network of [36], we employ only one unit of its residual block. The network SRCNN [1] consists of three convolutional layers with spatial sizes of 9×9, 5×5 and 5×5, respectively. The light-weight version of EDSR [28] is formed by stacking 3 units of its residual blocks, each consisting of two convolution operations and a ReLU activation in-between. It should be noted that the original residual bock of EDSR [28] uses convolutional layers with 256 filters. However, we use residual blocks employing convolutional layers with 128 filters for this network in order to make it light-weight. The light-weight version of MSRN [75] is formed by staking 2 units of its residual blocks, each generating features in multiple scales by employing convolution operations with spatial sizes of 3×3 and 5×5. In the light-weight version of GFN [37], we use one unit of its residual block in the deblurring module as well as one unit in the super resolution module, and a cascade of 3 units in the reconstruction module. In the light-weight version of ADSN [38], we employ one unit of residual channel-attention block in the deblurring module and a cascade of 4 units of back-projection residual blocks in the super resolution module. All these networks are trained for the task of image upsampling and deblurring using the same training dataset. Table 7.9 gives the performance, in terms of PSNR in dB and SSIM, of these light-weight networks along with that of the proposed one on the four benchmark datasets. It should be pointed out that we use a comparable number of parameters for all the networks used in comparison. It is seen from Table 7.9 that the proposed network outperforms the light-weight versions of other state-of-the-art networks in the cases of various scaling factors and different blurring kernel standard deviations.

Fig. 7.7 shows the visual qualities of the images restored by using the six networks on the *Powerpoint* image from the *Set 14* dataset, which is degraded by the Gaussian blurring kernel with $\sigma = 1.6$ and the scaling factor 3. It is seen from this figure that the proposed network recovers the textures and details of the image more precisely than the other networks do.

Fig. 7.8 shows the ground truth, blurred ($\sigma = 1.6$) and the low quality blurred and downsampled (scaling factor 3) versions of the *Comic* image from the *Set 14* dataset along with images produced at the output of *Stage 2* and *Stage 3* of the proposed network. This figure also shows the bilinear interpolated version of the low quality image. It is seen from the images of this figure that the upsampled image produced at the output of *Stage 2* of the proposed network is similar to the blurred version of the ground truth and is much superior to the bilinear interpolated image. It is also seen from this figure that final upsampled and deblurred image produced at the output of *Stage 3* of the proposed network has a good quality and is very similar to the ground truth image.

Table 7.10 gives the numbers of parameters and MACC operations of the proposed and other light-weight versions of the state-of-the-art schemes used for comparison. It is seen from this table that the proposed network outperforms the light-weight versions of EDSR [28] and MSRN [75], augmented by the deblurring network of [36], as well as the light-weight versions of GFN [37] and ADSN [38], by employing smaller numbers of parameters and operations.

Table 7.11: PSNR (in dB) values resulting from applying the UpDResNN and UPDCNN on the four benchmark datasets.

Dataset	Degradation	UPDCNN	UpDResNN
Set10	×3 and $\sigma = 1$	30.14	30.27
	×3 and $\sigma = 1.6$	29.97	30.19
Set5	×3 and $\sigma = 1$	33.41	33.57
	×3 and $\sigma = 1.6$	33.14	33.57
Set14	×3 and $\sigma = 1$	29.92	29.95
	×3 and $\sigma = 1.6$	29.78	29.91
BSD100	×3 and $\sigma = 1$	28.76	28.77
	×3 and $\sigma = 1.6$	28.69	28.78

7.5.1 Comparison between the Proposed Deep Joint Image Upsampling and Deblurring Networks

We now compare the performance, in terms of PSNR in dB, of the two deep joint image upsampling and deblurring networks proposed in this chapter, UpDCNN and UpDResNN. The comparison results are given in Table 7.11. It is seen from the results of this table that UpDResNN outperforms UPDCNN considerably. It should be pointed out that the former network employs 708K parameters compared to 745K parameters employed by the latter. Therefore, UpDResNN provides its improved performance despite using smaller number of parameters.

7.6 Conclusion

In this chapter, we have proposed two deep light-weight joint image upsampling and deblurring convolutional neural networks. For developing these two networks, we have proposed a novel global residual learning approach, in which the blurred version of the ground truth image is used. The results of the extensive experiments have shown the superiority of the two proposed networks, UPDCNN and UPDResNN, over those that use other conventional global residual learning approaches for the task of joint image upsampling and deblurring. However, between these two proposed networks, UPDResNN provides a performance superior to that of UPDCNN despite using a smaller number of parameters.

Chapter 8

Deep JPEG Image Deblocking Networks

8.1 Introduction

Image compression compacts the useful information in an image in order to reduce its size for various purposes such as transmission and storage. JPEG is one of the classical schemes for image compression that is commonly employed in real-world situations including image software and printers. Since JPEG is a block based compression scheme, the restored images using JPEG compressed images suffer from blocking artifacts.

JPEG compression scheme is based on block transformation. First, the image is divided into blocks of size 8×8 and then each block is transformed into the DCT domain as

$$\begin{aligned} X^c[k,l] = 4 \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x[m,n] \cos(\frac{k\pi(2m+1)}{2M}) \\ \cos(\frac{k\pi(2n+1)}{2N}) \\ 0 \le k \le M-1, 0 \le l \le N-1 \end{aligned} \tag{8.1}$$

where $X^c[k,l]$ is the DCT transform of the block (two-dimensional signal) $x[m,n]$. Next, the DCT coefficients of each block are quantized and inputted to an entropy coder such as Huffman coding. For the reconstruction process, the coded coefficients are fed into an entropy decoder and then inputted to a dequantizer. Finally the inverse DCT (IDCT) is applied on the dequantized values and the block is reconstructed. As seen from this procedure, the blocking effect is unavoidable in JPEG image decompression.

Deep neural networks have proved to provide very promising performance in various fields of image processing and computer vision. Deep convolutional neural nets have the capability of nonlinear end-to-end mapping that makes them suitable for image JPEG deblocking. Several works based on convolutional neural networks, such as [76], have been proposed aiming to reduce the artifacts in decompressed images. In this chapter, we develop two deep light-weight convolutional neural networks for the task of JPEG image deblocking that are able to provide high performances [94], [100].

8.2 Deep JPEG Image Deblocking using Residual Maxout Units

In this section, the concept of the proposed deep JPEG image deblocking using residual maxout units (DRMU) is explained [100]. In addition, the training details of the network along with its default hyper-parameters are described.

Residual blocks could facilitate the learning process of a deep network. The skip connections that are utilized in the residual blocks could facilitate the information flow more effectively in backpropagation and diminish the gradient vanishing problem. Maxout activation function proposed in [77] selects the most useful features for reconstructing the ground truth. Also, learning various nonlinearities ranging from a simple to a highly complex nonlinearity is possible by using the maxout activation function. By applying the maxout activation function on the feature vectors of the decompressed image and setting the objective of the network to obtain the ground truth, the most robust and useful features for image deblocking are preserved and the other less informative features discarded.

Therefore, the maxout activation function could find the optimal features for the task of JPEG image deblocking.

In our work, the maxout activation function outputs the maximum value of each of the m consecutive feature values along the channel dimension (Fig. 8.1). For instance, if the dimension of the feature vector input to the maxout activation function is 64 and $m = 8$, then the output feature vector has the dimension of 8. The 8 features so obtained are the most robust and relevant features for mapping between the input and output.

In this section, a combination of the residual blocks alleviating the gradient vanishing problem in a deep network and the maxout activation function used to learn the suitable and robust features is utilized for JPEG image deblocking.

Our deep JPEG image deblocking network, DRMU, includes three stages as shown in Fig. 8.2. First, the luminance channel (Y) of a JPEG decompressed image is inputted to the first stage, which is a convolutional layer used for feature extraction. Next, the resulting feature vectors of the decompressed image are fed to a sequence of N cascaded RMUs, which carries out the construction of the features of the residual between the ground truth and the decompressed image. In our experiments, the default number of RMUs is set to $N = 20$. Finally, the output of the sequence of RMUs is fed to the third stage, which is a convolutional layer used for the construction of an estimate of the ground truth.

We will now discuss the proposed architecture of RMU. The architecture is shown in Fig. 8.3. If $\mathbf{x}_i \in \mathbb{R}^n$ is the input to the first convolution block in the ith residual maxout unit (RMU), then the convolution operation of this block yields, $\mathbf{u}_i = f(\mathbf{x}_i) \in \mathbb{R}^n$. The resulting feature vectors, $\mathbf{u}_i)$, are fed to the maxout block of RMU, which obtains the maximum of each of the m consecutive features along the channel dimension, resulting in new feature vectors, $\mathbf{v}_i \in \mathbb{R}^{\frac{n}{m}}$. The feature vectors, $\mathbf{v}_i$, are fed to the 1×1 convolution operation, $g(.)$, accompanied by a ReLU activation function to yield the feature vectors, $\mathbf{w}_i = ReLU(g(\mathbf{z}_i)) \in \mathbb{R}^n$. These two operations are carried out by the third block in the

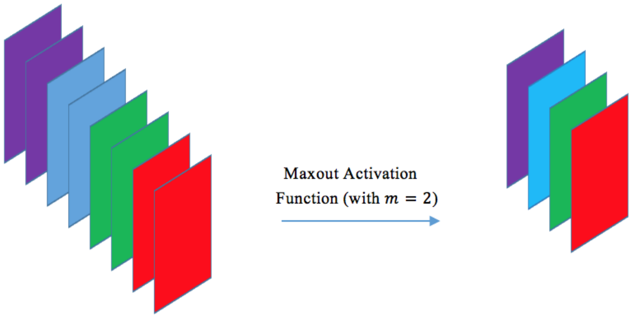


Figure 8.1: Maxout activation function gets the maximum of each of $m = 2$ consecutive slices.

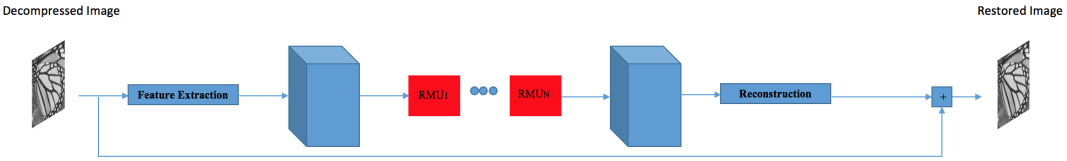


Figure 8.2: The architecture of the proposed DRMU. **RMU** denotes residual maxout unit.

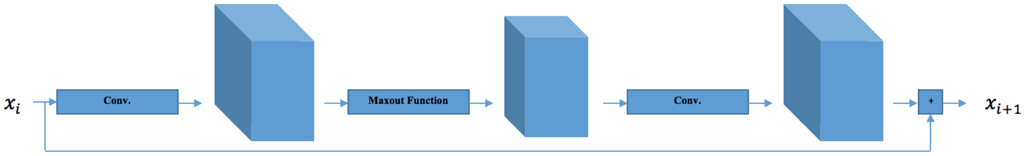


Figure 8.3: Residual maxout unit (RMU). **Conv.** denotes convolution operation. Note that the second convolution operation is followed by ReLU.

RMU shown in Fig. 8.2. Finally, the feature vectors $\mathbf{w}_i$ obtained from this block are added to the feature vectors inputted to the ith RMU. The relation between the input and output of the ith RMU is expressed as

$$\mathbf{x}_{i+1} = \mathbf{w}_i + \mathbf{x}_i \tag{8.2}$$

The convolutional layer that is used for feature extraction of the decompressed image and first convolutional layer at each RMU employ 64 filters of size 3×3. The maxout activation functions in the RMUs are applied on each 8 consecutive slices of their input tensors. Finally, the last convolutional layers in RMUs, that carry out the *dimensionality expansion*, utilize 64 filters of size 1×1.

To train DRMU, first the subimages of size 40×40 are extracted from the training and validation images of *BSD300* dataset [23] and are then degraded via JPEG compressor of MATLAB. Also, flipping and rotation are carried out to augment the training samples. To update the weights of the network in each iteration, stochastic gradient descent with the momentum parameter of 0.9 is utilized. The loss function in DRMU is taken as the mean squared error between ground truth and estimated deblocked images. The network is trained for the JPEG compression quality factors of 10 and 20 separately.

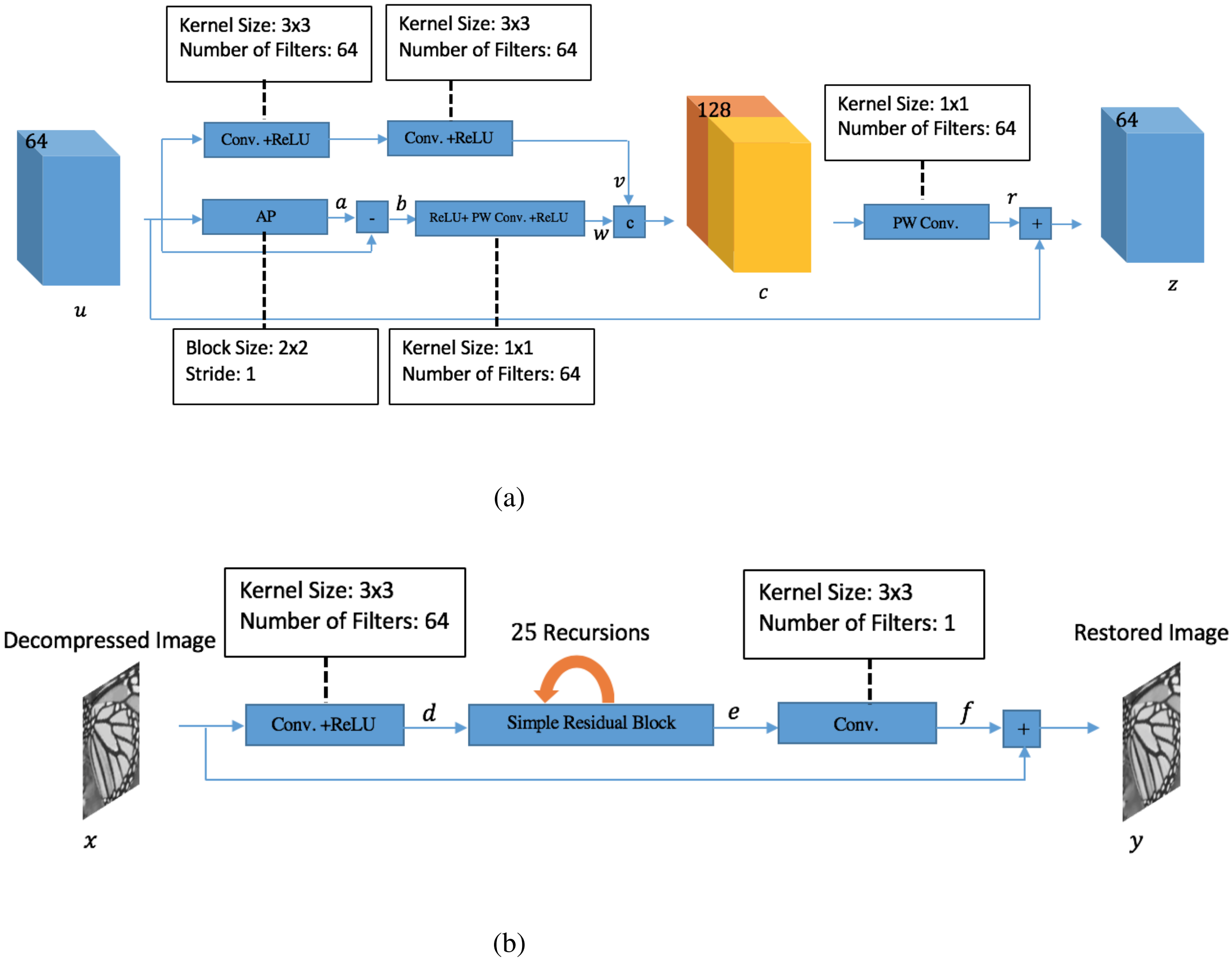
Every convolution operation is followed by a batch normalization process for normalizing the distribution of the data. The weights in all layers in our fully convolutional scheme are randomly initialized using the method of He, et al.[7].

8.3 Development of New Fractal and Non-fractal Deep Residual Networks for Deblocking of JPEG Decompressed Images

Deep learning based JPEG image deblocking schemes can be categorized into recursive and non-recursive classes of neural networks. The recursive neural networks for JPEG image deblocking, such as in [82], use a recursive block that employs the same set of parameters in all recursions. Therefore, these networks are ultra light-weight, i.e., they employ a very small number of parameters. The class of non-recursive neural networks for JPEG image deblocking [80], [100], [81] employ a cascade of learnable blocks all having the same architecture, but each having its own set of parameters. Even though the number of parameters used in a non-recursive neural network is larger than that of a recursive neural network, the number of operations in the non-recursive neural network is proportional only to the number of parameters employed by it. In this section, we develop two residual blocks [94], referred to as simple residual block (SRB) and compound residual block (CRB), to be used, respectively, in recursive (non-fractal) and non-recursive (fractal) frameworks of deep convolutional networks for deblocking of JPEG decompressed images. The main idea in designing SRB is that residual features are generated in two streams. In the first stream, the residual features are obtained directly from the input feature maps of the block by employing two convolutional layers, whereas in the second stream, the residual features are obtained only from the high frequency component of the input feature maps of the block. On the other hand, the main idea in designing CRB is that an enhanced representational capability is imparted to the network using this block by replacing each of the two convolutional layers of SRB by itself, thus providing a fractal character to the resulting compound residual block.

Here, we first develop the architecture of SRB and that of the recursive (non-fractal) image deblocking network using this block. This is followed by the development of the architecture of CRB and that of the non-recursive (fractal) deblocking network employing this block.

Use of the residual blocks in a deep network facilitates the flow of the information in backpropagation and helps in addressing the gradient vanishing problem. Basic residual blocks for image restoration [28] generally consist of two convolutional layers with a ReLU activation in between. Such a residual block learns the residue between its input and output feature maps, which consists mainly of high frequency content. Thus, adding a light-weight module to this residual block that generates features that are generated specifically from the high frequency component of the block's input and adding these features to the features directly obtained from the input of the block should further enhance the representational capability of the block. We now propose a residual block that generates such a rich set of high frequency residual features. The architecture of the proposed block is shown in Fig. 8.4 (a). It is seen from this figure that feature maps **u** input to the block are first made to undergo two convolution operations each followed by a ReLU activation in order to yield the feature maps **v**. Each of the two convolution operations uses 64 filters with kernel size 3×3. At the same time, in order to obtain the high frequency component of **u**, the feature maps **u** are first passed through an average pooling operation and the resulting maps **a** consist of the low frequency component of **u**. The average pooling operation uses stride 1 and kernel size 2×2. It is to be noted that since the stride of the average pooling operation is 1, the spatial resolution of the feature maps **a** is the same as that of the feature maps **u**. Next, the feature maps **a** are subtracted from the feature maps **u** to obtain the feature maps **b**, which now contains the high frequency component of the feature maps **u**. The feature maps **b** are then passed through a point-wise convolution operation followed by a ReLU activation to yield the high frequency feature maps **w**. This point-wise convolution operation uses 64 filters each with kernel size 1×1. The high frequency feature maps **w**



(a)

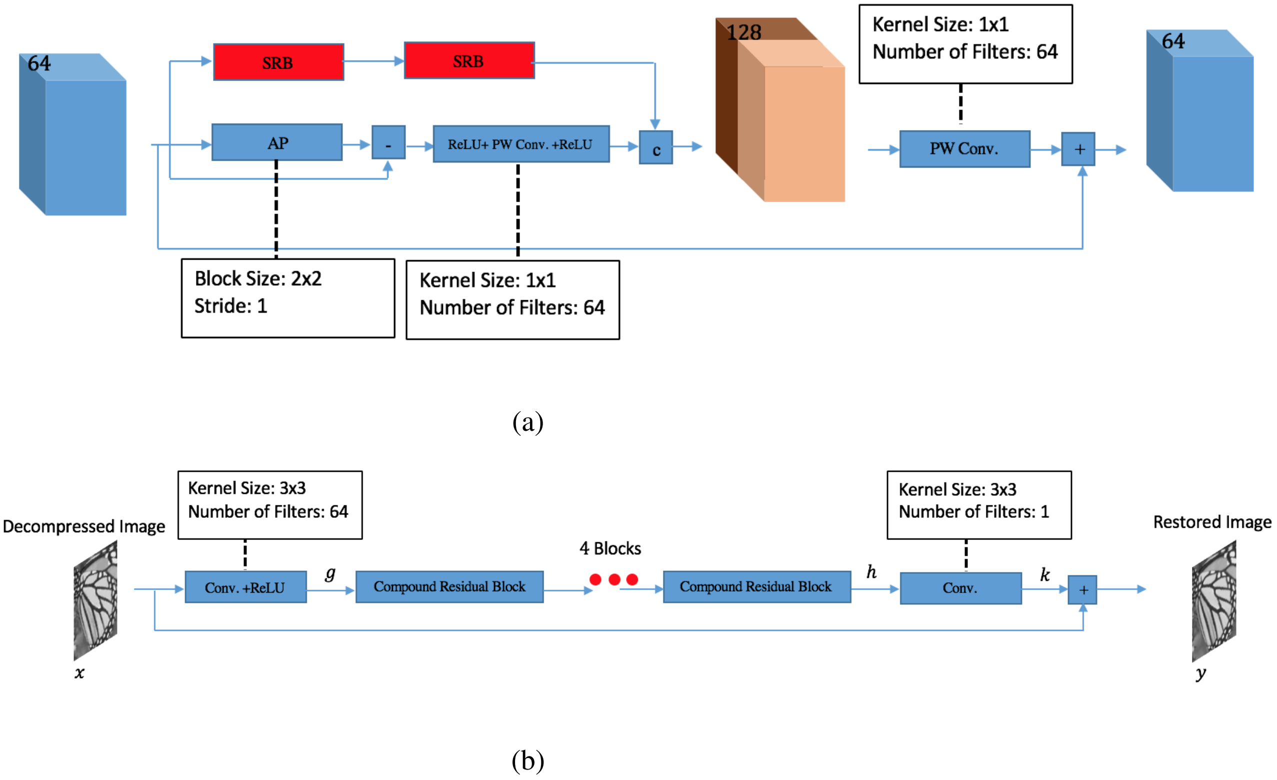
(b)

Figure 8.4: Architecture of the proposed recursive network. (a) Architecture of SRB, where *AP* and *PW Conv.* denote, respectively, the average pooling and point-wise convolution. (b) Overall architecture of the recursive network.

are then concatenated with the feature maps **v** and the resulting feature maps **c** undergo a point-wise convolution operation in order to yield the feature maps **r** corresponding to the block's output residual signal. The point-wise convolution operation uses 64 filters each with kernel size 1×1. The residual feature maps **r** thus obtained are added to the maps **u**, the block's input features, to obtain the output feature maps **z** for the block.

Now, in order to design an ultra light-weight JPEG image deblocking network, we employ SRB in a recursive network. The architecture of the recursive JPEG deblocking network is shown in Fig. 8.4 (b). It is seen from this figure that the JPEG decompressed image **x** first undergoes a convolution operation in order to obtain its low quality feature maps **d**. This convolution operation uses 64 filters each with kernel size 3×3. The low quality feature maps thus obtained are then passed through SRB, which is used recursively for 25 times, to obtain the feature maps **e** of the residual signal between the ground truth and the JPEG decompressed image. The feature maps obtained from the final recursion of SRB are then passed through a convolution operation to obtain the residue **f**. This convolution operation uses one filter with kernel size 3×3. Finally, the residual signal **f** is added to the JPEG decompressed input image **x** to obtain the estimated high quality image **y**.

In order to design a non-recursive (fractal) high performance light-weight JPEG deblocking network, we first convert SRB to CRB by replacing each of the two convolutional



(a)

(b)

Figure 8.5: Architecture of the proposed non-recursive network. (a) Architecture of CRB. (b) Overall architecture of the non-recursive network.

layers in SRB by itself. The architecture of the resulting CRB is shown in Fig. 8.5 (a). It is seen from this figure that CRB fuses the features from different hierarchies and levels of abstractions and thus imparts an enhanced representational capability to the network using this block. The architecture of the non-recursive network using this compound residual block, CRB, is shown in Fig. 8.5 (b). It is seen from this figure that the JPEG decompressed image **x** is first passed through a convolution operation that extracts its feature maps **g**. This convolution operation employs 64 filters each with kernel size 3×3. The low quality feature maps thus obtained are then passed through 4 units of CRB, to obtain the feature maps **h** of the residue **k** between the ground truth and the JPEG decompressed image. The feature maps **h** resulting from the last CRB are passed through a convolution operation using one filter with kernel size 3×3 yielding the residual signal **k**. This residual signal is finally added to the JPEG decompressed input image **x** to obtain the estimated deblocked image **y**. It needs to be pointed out that the use of the compound residual blocks provides the proposed non-recursive network with three levels of residual connections. The first two are short and middle range connections as furnished by the fractal structure of CRBs and the third one establishes a long range skip connection in the network. These three types of residual connections provide the network with *Residual-in-Residual* character that facilitates it in providing a highly rich set of high frequency features.

The two proposed networks are trained using the the sub-images of size 40×40 from *BSD300* dataset [23]. The $\ell 1$ norm loss between the ground truth and estimated deblocked samples is optimized using the stochastic gradient descent (SGD) optimizer to update the weights of the network in each iteration. The learning rate of SGD is initialized by a value of 0.1 and decreased by a factor of 10 after each 10 epochs. The network is trained for a total of 40 epochs.

Table 8.1: PSNR (SSIM) values resulting from applying DRMU and various state-of-the-art methods to images of two benchmark datasets.

Dataset	Quality Factor	JPEG	SA-DCT [83]	ARCNN [76]	TNRD [79]	DnCNN [80]	DRMU (ours) [100]
Classic5	10	27.82 (0.7595)	28.88 (0.8071)	29.03 (0.7929)	29.28 (0.7992)	29.40 (0.8026)	29.43 (0.8041)
	20	30.12 (0.8344)	30.92 (0.8663)	31.15 (0.8517)	31.47 (0.8576)	31.63 (0.8610)	31.63 (0.8613)
Live1 [78]	10	27.77 (0.7730)	28.65 (0.8093)	28.96 (0.8076)	29.15 (0.8111)	29.19 (0.8123)	29.31 (0.8178)
	20	30.07 (0.8512)	30.81 (0.8781)	31.29 (0.8733)	31.46 (0.8769)	31.59 (0.8802)	31.67 (0.8832)

8.4 Experiments

8.4.1 Experimental Results of DRMU

In this section, experiments are carried out to obtain the objective and subjective results of DRMU in terms of PSNR and SSIM metrics. The results obtained from the proposed DRMU are compared with that of the state-of-the-art methods for JPEG image deblocking via deep neural networks.

The results of the proposed DRMU and those from using SA-DCT [83], ARCNN [76], TNRD [79] and DnCNN [80] are given in Table 8.1. As seen from this table, DRMU outperforms all the schemes, both in terms of the objective and subjective metrics. These results demonstrate the effectiveness of using the maxout activation function for JPEG image deblocking. By employing the maxout activation function, the competition between the various features is carried out and the most useful features are selected for image deblocking. Moreover, the skip connections in the residual blocks allow the DRMU network to become deeper without suffering from the gradient vanishing effect.

Fig. 8.6 shows the quality of the deblocked images obtained by applying the ARCNN and the proposed DRMU schemes on the JPEG decompressed image of *Barbara* with the quality factor of 10. It is obvious from Fig. 8.6 (c) and (d) that the visual quality of the proposed scheme is superior to that from ARCNN. Also, the blocking artifact is considerably reduced in Fig. 8.6 (d).

Table 8.2: The performance of deeper DRMU, when the quality factor is 20.

Dataset	16 Residual Maxout Units	20 Residual Maxout Units
Classic5	31.60 (0.8607)	31.63 (0.8613)
Live1	31.64 (0.8827)	31.67 (0.8832)

Table 8.3: The performance of DRMU with larger receptive field, when the quality factor is 20.

Dataset	RF 3×3	RF 5×5
Classic5	31.63 (0.8613)	31.70 (0.8628)
Live1	31.67 (0.8832)	31.73 (0.8844)

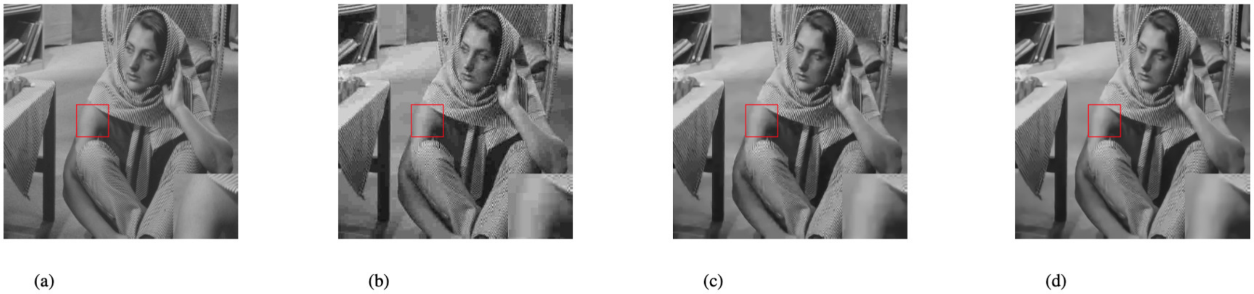


Figure 8.6: The visual comparison between the proposed DRMU and ARCNN applied on JPEG decompressed *Barbara* image with the quality factor of 10. (a) Ground truth. (b) JPEG. (c) ARCNN. (d) DRMU.

Fig. 8.7 shows the restored *Lenna* image using DRMU in the cases of the quality factors 10 and 20. It is seen from the images obtained by using the proposed scheme that it is quite effective in removing the blocking artifacts in the case of the JPEG image decompressed by either quality factor.

To study the effect of the network depth for the proposed scheme, the number of residual maxout units is changed from 20 to 16. The performance results of the network with these two settings are given in Table 8.2. It is seen from this table that making the network deeper improves the results further. However, this performance gain achieved by adding more residual maxout units is only marginal.

Finally, the receptive fields (RF) of all convolutional layers (except the layer that is

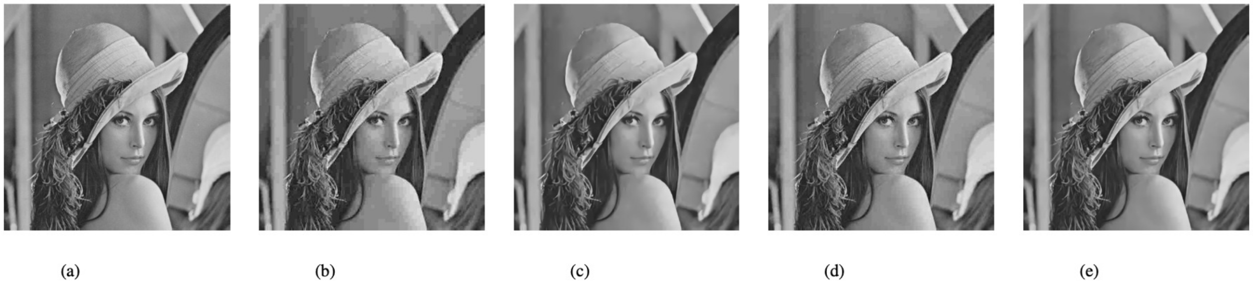


Figure 8.7: The visual demonstration of DRMU when is applied on the *Lenna* JPEG decompressed image in the case of various quality factors. (a) Ground truth. (b) Decompressed using JPEG with quality factor 10. (c) Deblocked using DRMU, when the JPEG quality factor is 10. (d) Decompressed using JPEG with quality factor 20. (e) Deblocked using DRMU, when the JPEG quality factor is 20. Please zoom in to see the details.

employed for dimensionality expansion) are increased to 5×5. The performance of DRMU with a larger receptive field is given in Table 8.3. As seen from this table, in the proposed DRMU, the receptive field of 5×5 provides a higher performance than that provided by the receptive field of 3×3. The larger receptive fields extract more contextual information from the the decompressed image and lead to an improved performance.

8.4.2 Experimental Results of the Deep Fractal and Non-Fractal Deblocking Networks

In this section, we first perform ablation studies on SRB and CRB to show their effectiveness for the task of JPEG image deblocking in the proposed recursive and non-recursive networks. We then compare the performance and complexity of the two proposed networks with those of the state-of-the-art deep JPEG deblocking networks.

In order to generate a rich set of high frequency residual features, the residual block SRB has been used to extract the high frequency components of its input feature maps. The branch containing the pooling layer and the point-wise convolution layer is the main idea used in designing SRB. To investigate the effectiveness of SRB on the performance of

Table 8.4: The performance of the recursive network (RN) and non-recursive network (NN) using SRB and CRB and their variants.

Dataset	RN with *SRB Variant*	RN	NN with *SRB*	NN
Classic5	29.21 (0.7993)	29.28 (0.8010)	29.46 (0.8053)	29.56 (0.8088)
Live1	29.20 (0.8149)	29.23 (0.8157)	29.33 (0.8183)	29.42 (0.8208)

Table 8.5: The impact of the number of recursions on the performance of the recursive network.

Dataset	5 Recursions	15 Recursions	25 Recursions
Classic5	28.89 (0.7880)	29.01 (0.7920)	29.28 (0.8010)
Live1	28.93 (0.8056)	29.04 (0.8091)	29.23 (0.8157)

the recursive network, we remove the branch containing the pooling layer and the point-wise convolution layer from this block. We refer to this variant of SRB as *SRB Variant*. The left side of Table 8.4 gives the performance results of the recursive network employing SRB and its variant, when the network is applied to the JPEG decompressed images with the QF value of 10. It is seen from the results of this table that removing the branch containing the average pooling layer and the point-wise convolution operation from SRB results in degrading the network performance. In order to investigate the impact of the fractal character of CRB on the performance of the non-recursive network, we remove this fractal character from CRBs that converts CRBs into SRBs. The right side of Table 8.4 gives the performance results of the proposed non-recursive network with CRBs and SRBs. It is seen from these results that removing the fractal character of CRB results in a significant performance degradation.

To investigate the impact of the number of recursions on the performance of the recursive deblocking network, we first reduce the number of recursions from 25 to 15 and 5, and then compare the performance of the recursive network with these three values of the recursions in Table 8.5. It is seen from this table that as the number of recursions reduces, the network performance deteriorates.

Table 8.6: PSNR and SSIM results obtained by using the proposed recursive and non-recursive networks and the state-of-the-art networks for deblocking of JPEG decompressed images. The **red** values are the bests in comparison.

Dataset	Quality Factor	JPEG	ARCNN [76]	TNRD [79]	DCSC [82]	Proposed [94]	DnCNN [80]	LPIO [81]	DRMU [100]	Proposed [94]
Classic5	10	27.82 (0.7595)	29.03 (0.7929)	29.28 (0.7992)	29.25 (0.8030)	29.28 (0.8010)	29.40 (0.8030)	29.35 (0.8010)	29.43 (0.8041)	29.56 (0.8088)
	20	30.12 (0.8344)	31.15 (0.8517)	31.47 (0.8576)	31.43 (0.8600)	31.41(0.8578)	31.63 (0.8610)	31.58 (0.8560)	31.63 (0.8613)	31.78(0.8642)
Live1 [78]	10	27.77 (0.7730)	28.96 (0.8076)	29.15 (0.8111)	29.17 (0.8150)	29.23 (0.8157)	29.19 (0.8120)	29.17 (0.8110)	29.31 (0.8178)	29.42 (0.8208)
	20	30.07 (0.8512)	31.29 (0.8733)	31.46 (0.8769)	31.48 (0.8800)	31.51 (0.8803)	31.59 (0.8800)	31.52 (0.8760)	31.67 (0.8832)	31.80(0.8857)
Number of Parameters		-	106K	21K	94K	91K	737K	1394K	761K	728K

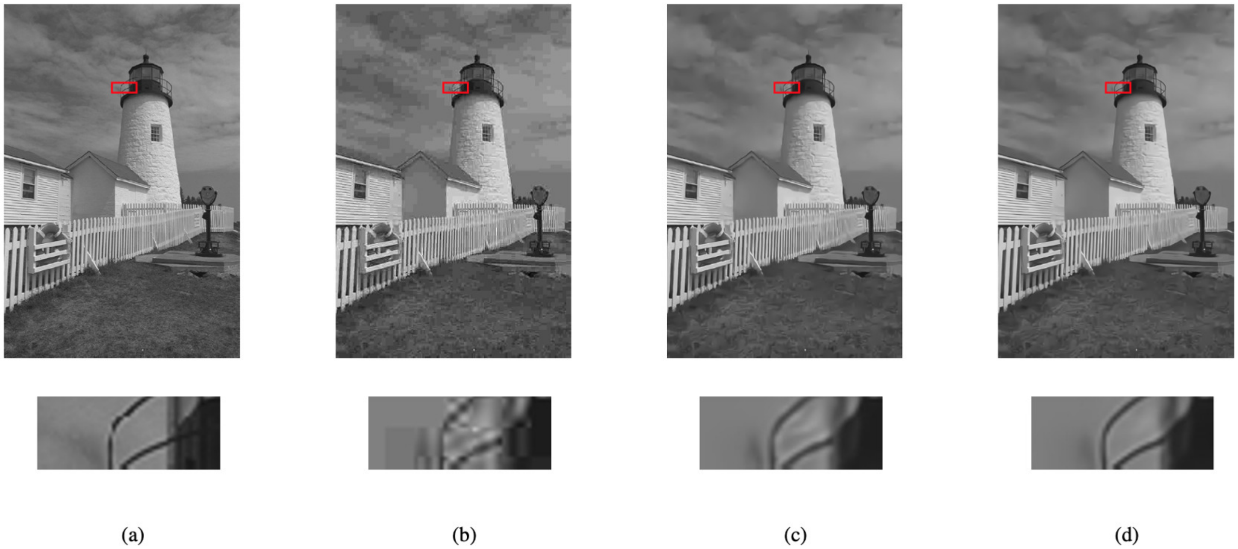


Figure 8.8: Visual quality of the *Lighthouse* images from *LIVE 1* dataset deblocked by applying the proposed recursive and non-recursive networks. (a) Ground truth. (b) JPEG image with QF 10. (c) Image obtained from the recursive network. (d) Image obtained from the non-recursive network.

The proposed recursive deblocking network is designed to yield a good performance with a very small number of parameters. Hence, its performance is compared with the ultra light-weight state-of-the-art networks, in which the number of parameters employed is less than 110K. On the other hand, the proposed non-recursive deblocking network is designed to provide a superior performance with a modest number of parameters. Hence, its performance is compered with the state-of-the-art networks with the number of parameters less than 1.5M. The performance results and the number of parameters of these two types of networks are given, respectively, in the left and right sides of Table 8.6. It is seen from these results that both of the proposed deblocking networks provide the best performance results compared to the respective state-of-the-art networks.

Fig. 8.8 shows the visual quality of the deblocked images obtained by applying the proposed recursive and non-recursive networks on the *Lighthouse* image from *LIVE1* dataset, when QF is 10. It is seen from this figure that both the recursive and non-recursive proposed networks successfully reduce the JPEG compression artifacts. However, as expected, the deblocked image obtained from the proposed non-recursive network has more similarity to the ground truth in comparison to the image deblocked by the proposed recursive network.

8.5 Conclusion

In this chapter, we have developed two deep light-weight convolutional neural networks for the task of JPEG image deblocking. These two networks use, respectively, maxout action units and fractal neural networks, in their networks architectures. Based on the experiments carried out in this chapter, it has been shown that both the ideas of employing maxout activation units and fractal neural networks are indeed helpful in enhancing the JPEG image deblocking performance.

Chapter 9

Conclusion

9.1 Concluding Remarks

In many real-life applications, such as medical imaging, intelligent transportation systems and classifiers [91], the acquired images lack high quality due to various types of degradations associated with image capturing devices and require improvement in their quality. In recent years, the design of deep neural networks has emerged to be a very promising tool for image restoration. However, the use of deep neural networks using a large number of parameters to provide the desired accuracy for these applications is not practical in view of their excessive memory and power consumptions. Hence, the design of image restoration convolutional neural networks that employ small number of parameters and yet provide high accuracy is very crucial in many real-life applications especially mobile applications.

As the richness of the feature maps produced by a deep network has a direct impact on its performance, the objective of the thesis has been to develop different light-weight image restoration networks that are capable of generating rich sets of feature maps by using various kinds of prior information about the quality of degradation operations associated with the image capturing devices. In this thesis, three specific degradation models, associated with bicubic downsampling, Gaussian blurring coupled with downsampling and JPEG compression blocking artifacts, have been considered and suitable prior information associated with these three degradation models have been used for developing deep light-weight image restoration networks.

In Chapters 3, 4, 5 and 6 of this thesis, we have developed several image super resolution networks to enhance the quality of images degraded by the bicubic downsampling operation of the image capturing devices, using the various prior information about this operation. Specifically, three different prior information, namely, multi-scale feature generation, guided feature generation and efficient feature fusion have been used for developing light-weight image super resolution networks.

In Chapter 7, deep networks for improving the quality of images degraded by the Gaussian blurring followed by downsampling, have been proposed. Specifically, the blurred version of the high-quality images has been used in a global residual learning as the prior information.

In Chapter 8, deep networks for suppressing the JPEG blocking artifacts of the decompressed images have been designed. Specifically, two prior information, namely, robust feature generation and use of the high-frequency components have been used for designing these deep image restoration networks.

The results of the extensive experiments have shown the effectiveness of all the deep light-weight image restoration networks proposed in this thesis.

9.2 Suggestions for Future Investigations

In this thesis, different networks have been developed for restoring images degraded by the processes inherited in the capturing devices. The design of all these networks is based on the supervised learning strategies. However, the design of deep networks using the supervised learning strategies has the drawback of requiring large number of training samples and the network design cannot handle unknown image degradation processes. In future, the design strategies developed in this thesis can be further investigated so that they can be used in an unsupervised training environment.

References

[1] C. Dong, C.C. Loy, K. He and X. Tang, “Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295-307, February 2016.

[2] D. Liu, Z. Wang, B. Wen, J. Yang, W. Han and T. Huang, “Robust single image super-resolution via deep networks with sparse prior," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3194-3207, July 2016.

[3] J. Kim, J.K. Lee and K.M. Lee, “Accurate image super-resolution using very deep convolutional network,” in *Proc. CVPR*, 2016.

[4] Y. Tai, J. Yang, X. Liu, and C. Xu, “MemNet: A persistent memory network for image restoration," in *Proc. ICCV*, 2017.

[5] D. Martin, C. Fowlkes, D. Tal and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. ICCV*, 2001.

[6] J. Yang, J. Wright, T. Huang and Y. Ma, “Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861 - 2873, May 2010.

[7] K. He, X. Zhang, S. Ren and J. Sun, “Delving deep into rectifiers: Surpassing human level performance on imagenet classification," in *Proc. ICCV*, 2015.

[8] W. Shi, J. Caballero, F. Huszr, J. Totz, A.P. Aitken, R. Bishop, D. Rueckert and Z. Wang, “Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. CVPR*, 2016.

[9] J. Yang, J. Wright, T.S. Huang and Y. Ma, “ Image super-resolution via sparse representation,” *IEEE Transactions on Image Processing*, vol. 19, no. 11, 2010, pp. 2861-2873.

[10] K. Guo, X. Yang, H. Zha, W. Lin and S. Yu, “Multiscale semilocal interpolation with antialiasing,” *IEEE Transactions on Image Processing*, vol. 21, no. 2, 2011, pp. 615-625.

[11] D. We, “Image super-resolution reconstruction using the high-order derivative interpolation associated with fractional filter functions,” *IET Signal Processing*, vol. 10, no. 9, 2016, pp. 1052-1061.

[12] L. Baboulaz and P.L. Dragotti, “Exact feature extraction using finite rate of innovation principles with an application to image super-resolution,” *IEEE Transactions on Image Processing*, vol. 18, no. 2, 2009, pp. 281-298.

[13] C. Dong, C.C. Loy and X. Tang,“Accelerating the Super-Resolution Convolutional Neural Network," in *Proc. ECCV*, 2016.

[14] N. Ahn, B. Kang and K.A. Sohn,“Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proc. ECCV*, 2018.

[15] B. Li, B. Wang, J. Liu, Z. Qi and Y. Shi, “s-LWSR: Super Lightweight Super-Resolution Network," *IEEE Transactions on Image Processing*, vol. 29, 2020, pp. 8368-8380.

[16] X. Luo, Y. Xie, Y. Zhang, Y. Qu, C. Li and Y. Fu, “LatticeNet: Towards Lightweight Image Super-resolution with Lattice Block," in *Proc. ECCV*, 2020.

[17] W. Yang, J. Feng, J. Yang, F. Zhao, J. Liu, Z. Guo and S. Yan, “Deep Edge Guided Recurrent Residual Learning for Image Super-Resolution," *IEEE Transactions on Image Processing*, vol. 26, no. 12, 2017, pp. 5895-5907.

[18] C. Ma, Y. Rao, Y. Cheng, C. Chen, J. Lu and J. Zhou, “Structure-Preserving Super Resolution with Gradient Guidance," in *Proc. CVPR*, 2020.

[19] J. Kim, J. K. Lee, and K. M. Lee, “Deeply-recursive convolutional network for image super-resolution," in *Proc. CVPR*, 2016.

[20] Y. Tai, J. Yang, X. Liu, “Image super-resolution via deep recursive residual network," in *Proc. CVPR*, 2017.

[21] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-L. Alberi-Morel, “Low- complexity single-image super-resolution based on nonnegative neighbor embedding", in *Proc. BMVC*, 2012.

[22] R. Zeyde, M. Elad, and M. Protter, “On single image scale-up using sparse-representations," in *Curves and Surfaces*, 2012.

[23] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. ICCV*, 2001.

[24] J.B. Huang, A. Singh and N. Ahuja, “Single image super-resolution from transformed self-exemplars," in *Proc. CVPR*, 2015.

[25] Z. Li, J. Yang, and Z. Liu, and X. Yang and G. Jeon and W. Wu, “Feedback Network for Image Super-Resolution," in *Proc. CVPR*, 2019.

[26] W. Bae, J. Yoo and J.C. Ye, “Beyond Deep Residual Learning for Image Restoration: Persistent Homology-Guided Manifold Simplification," in *Proc. CVPRW*, 2017.

[27] P. Liu, H. Zhang, K. Zhang, L. Lin and W. Zuo, “Multi-level Wavelet-CNN for Image Restoration," in *Proc. CVPRW*, 2018.

[28] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee. “Enhanced deep residual networks for single image super-resolution," in *Proc. CVPR*, 2017.

[29] W-S. Lai, J-B. Huang, N. Ahuja, and M-H. Yang, “Deep laplacian pyramid networks for fast and accurate super-resolution," in *Proc. CVPR*, 2017.

[30] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong and Y. Fu,“Image Super-Resolution Using Very Deep Residual Channel Attention Networks," in *Proc. ECCV*, 2018.

[31] T. Dai, J. Cai, Y. Zhang, S-T. Xia and L. Zhang, “Second-order Attention Network for Single Image Super-Resolution," in *Proc. CVPR*, 2019.

[32] M. Haris, G. Shakhnarovich and N. Ukita, “Deep backprojection networks for super-resolution," in *Proc. CVPR*, 2018.

[33] J. Hu, L. Shen, S. Albanie, G. Sun and E. Wu, “Squeeze-and-excitation networks", in *Proc. CVPR*, 2018.

[34] M. Irani and S. Peleg, "Improving resolution by image registration,“ *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 3, 1991, pp. 231-239.

[35] Y. Guo, J. Chen, J. Wang, Q. Chen, J. Cao, Z. Deng, Y. Xuy and M. Tany, “Closed-loop Matters: Dual Regression Networks for Single Image Super-Resolution," in *Proc. CVPR*, 2020.

[36] S. Nah, T.H. Kim and K.M. Lee, “Deep multi-scale convolutional neural network for dynamic scene deblurring," in *Proc. CVPR*, 2017.

[37] X. Zhang, H. Dong, Z. Hu, W-S. Lai, F. Wang and M-H. Yang, “Gated fusion network for joint image deblurring and super-resolution," in *Proc. BMVC*, 2018.

[38] D. Zhang, Z. Liang, and J. Shao, “Joint image deblurring and super-resolution with attention dual supervised network," *Neurocomputing*, vol. 412, pp.187-196, October 2020.

[39] H. Lin, Z. Shi, and Z. Zou, “Maritime semantic labeling of optical remote sensing images with multi-scale fully convolutional network," *Remote Sensing*, vol. 9, no. 5, p. 480, May 2017.

[40] F. Chollet. (2015). Keras. Accessed: January 6, 2021. [Online]. Available: https://github.com/keras-team/keras

[41] M. Abadi et al. (2015). TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Accessed: January 6, 2021. [Online]. Available: https://www.tensorflow.org

[42] E. Agustsson, R. Timofte, "Ntire 2017 Challenge on Single Image Super-Resolution: Dataset and Study," in *Proc. CVPR*, 2017.

[43] S-H. Gao, M-M Cheng, K. Zhao, X-Y. Zhang, M-H. Yang, P. Torr,"Res2Net: A New Multi-scale Backbone Architecture", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no.2, 2019, pp. 652-662.

[44] Y. Zhang, Y. Tian, Y. Kong, B. Zhong and Y. Fu, "Residual Dense Network for Image Super-Resolution," in *Proc. CVPR*, 2018.

[45] Z. Hui, X. Wang and X. Gao, "Fast and accurate single image super-resolution via information distillation network," in *Proc. CVPR*, 2018.

[46] W. Lee, J. Lee, D. Kim and B. Ham, "Learning with Privileged Information for Efficient Image Super-Resolution," in *Proc. ECCV*, 2020.

[47] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution", in *Proc. ACCV*, 2014.

[48] K. Jiang and Z. Wang and P. Yi and G. Wang and T. Lu and J. Jiang, "Edge-enhanced GAN for remote sensing image superresolution," *IEEE Transactions on Geoscience and Remote Sensing*, 2019.

[49] D. Liu and Z. Wang and N. Nasrabadi and T. Huang,"Learning a Mixture of Deep Networks for Single Image Super-Resolution," in *Proc. ACCV*, 2016.

[50] C. Ledig, L. Theis, F. Husz?ar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., " Photo-realistic single image superresolution using a generative adversarial network," in *Proc. CVPR*, 2017.

[51] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, C.C. Loy, Y. Qiao and X. Tang, "ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks," in *Proc. ECCV*, 2018.

[52] A. Lugmayr, et al., "NTIRE 2020 challenge on real-world image super-resolution: methods and results," in *Proc. CVPR*, 2020.

[53] A. Bovik, "The Essential Guide to Image Processing", Academic Press.

[54] Z. Hui, X. Gao, Y. Yang and X. Wang, " Lightweight Image Super-Resolution with Information Multi-distillation Network," in *Proc. ACMMM*, 2019.

[55] X. He, Z. Mo, P. Wang, Y. Liu, M. Yang and J. Cheng, “ODE-inspired Network Design for Single Image Super-Resolution," in *Proc. CVPR*, 2019.

[56] Z. Jiang, H. Zhu, Y. Lu, G. Ju and A. Men,“Lightweight Super-Resolution Using Deep Neural Learning," *IEEE Transactions on Broadcasting*, vol. 66, no. 4, pp. 814-823, December 2020.

[57] J. Wan , H. Yin , Z. Liu, A. Chong and Y. Liu, “Lightweight Image Super-Resolution by Multi-Scale Aggregation," *IEEE Transactions on Broadcasting*, October 2020.

[58] F. Fang, J. Li and T. Zeng, “Soft-Edge Assisted Network for Single Image Super-Resolution," *IEEE Transactions on Image Processing*, vol. 29, 2020, pp. 4656-4668.

[59] S. Huang, J. Sun, Y. Yang, Y. Fang, P. Lin and Y. Que, “Robust Single-Image Super-Resolution Based on Adaptive Edge-Preserving Smoothing Regularization," *IEEE Transactions on Image Processing*, vol. 27, no. 6, 2018, pp. 2650-2663.

[60] W. Dong, L. Zhang, G. Shi, and X. Li, “Nonlocally centralized sparse representation for image restoration,” *IEEE Transactions on Image Processing*, vol. 22, no. 4, 2013, pp. 1620-1630.

[61] W. Dong, L. Zhang, R. Lukac, and G. Shi, “Sparse representation based image interpolation with nonlocal autoregressive modeling,” *IEEE Transactions on Image Processing*, vol. 22, no. 4, 2013, pp. 1382-1394.

[62] P. Lei and C. Liu, “An Efficient Group Feature Fusion Residual Network for Image Super-Resolution," in *Proc. ACCV* , 2020.

[63] J.Y. Ahn and N.I. Cho,“Neural Architecture Search for Image Super-Resolution Using Densely Constructed Search Space: DeCoNAS," in *arXiv:2104.09048*, 2021.

[64] Y. Huang, J. Li, X. Gao, W. Lu and Y. Hu, “Improving Image Super-Resolution via Feature Re-Balancing Fusion," in *Proc. ICME*, 2019.

[65] S. Schulter, C. Leistner, and H. Bischof, "Fast and accurate image upscaling with super-resolution forests," in *Proc. CVPR*, 2015.

[66] Y. Huang, J. Li, X. Gao, Y. Hu and W. Lu, “Interpretable Detail-Fidelity Attention Network for Single Image Super-Resolution," *IEEE Transactions on Image Processing*, vol. 30, 2021, pp. 2325-2339.

[67] Y. Yuan, S. Liu, J. Zhang, Y. Zhang, C. Dong and L. Lin, “Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks," in *Proc. CVPR*, 2018.

[68] H. Zhao, X. Kong, J. He, Y. Qiao and C. Dong, “Efficient Image Super-Resolution Using Pixel Attention," in *Proc. ECCVW*, 2020.

[69] , X. Wang and Q. Wang and Y. Zhao and J. Yan and L. Fan and L. Chen, “A computationally efficient superresolution image reconstruction algorithm,” in *Proc. ACCV*, 2020.

[70] I. Daubechies, M. Defrise, and C. De Mol, “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Commun. Pure Appl. Math.*, 2004.

[71] C.J. Rozell, D.H. Johnson, R.G. Baraniuk, and B.A. Olshausen, “Sparse coding via thresholding and local competition in neural circuits", *Neural Comput.*, 2008.

[72] K. Gregor and Y. LeCun, “Learning fast approximations of sparse coding," in *Proc. ICML*, 2010.

[73] W. Dong, L. Zhang and G. Shi, “Centralized sparse representation for image restoration," in *Proc. ICCV*, 2011.

[74] K. Zhang, X. Gao, D. Tao, and X. Li, "Single image super-resolution with non-local means and steering kernel regression," *IEEE Transactions on Image Processing*, vol. 21, no. 11, pp. 4544-4556, 2012.

[75] J. Li, F. Fang, K. Mei and G. Zhang, “Multi-scale residual network for image super-resolution," in *Proc. ECCV*, 2018.

[76] C. Dong, Y. Deng, C.C. Loy and X. Tang, "Compression artifacts reduction by a deep convolutional network,” in *Proc. ICCV*, 2015.

[77] I.J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville and Y. Bengio, "Maxout networks," in *Proc. ICML*, 2013.

[78] H. R. Sheikh, Z. Wang, L. Cormack, and A. C. Bovik, "Live image quality assessment database release 2", 2005.

[79] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, 2016, pp. 1256 - 1272.

[80] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: residual learning of deep CNN for image denoising," *IEEE Transactions on Image Processing*, 2017.

[81] Q. Fan, D. Chen, L. Yuan, G. Hua, N. Yu, and B. Chen, "Decouple learning for parameterized image operators," in *Proc. ECCV*, 2018.

[82] X. Fu, Z-J. Zha, F. Wu, X. Ding and J. Paisley, "JPEG artifact reduction via deep convolutional sparse coding," in *Proc. ICCV*, 2019.

[83] A. Foi, V. Katkovnik, and K. Egiazarian. "Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images," *IEEE Transactions on Image Processing*, vol. 16, no. 5, 2007, pp. 1395-1411.

[84] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “SRNMSM: A deep light-weight image super resolution network using multi-scale spatial and morphological feature generating residual blocks," *IEEE Transactions on Broadcasting*, Early Access, 2021.

[85] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “FPNet: A deep light-weight interpretable neural network using forward prediction filtering for efficient single image super resolution, *IEEE Transactions on Circuits and Systems II*, Early Access, 2021.

[86] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “SRNHARB: A deep light-weight image super resolution network using hybrid activation residual blocks," *Signal Processing: Image Communication*, vol. 99, 2021, p. 116509.

[87] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “SRNSSI: A deep light-weight network for single image super resolution using spatial and spectral information," *IEEE Transactions on Computational Imaging*, vol. 7, 2021, pp. 409-421.

[88] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “UPDResNN: A deep light-weight image upsampling and deblurring residual neural network," *IEEE Transactions on Broadcasting*, vol. 67, no. 2, 2021, pp. 538-548.

[89] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “MuRNet: A deep recursive network for super resolution of bicubically interpolated images," *Signal Processing: Image Communication*, vol. 94, 2021, p. 116228.

[90] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “CompNet: A new scheme for single image super resolution based on deep convolutional neural network," *IEEE Access*, vol. 6, 2018, pp. 59963 - 59974.

[91] A. Esmaeilzehi and H. Abrishami Moghaddam, “Nonparametric kernel sparse representation-based classifier," *Pattern Recognition Letters*, vol. 89, no. 4, 2017, pp. 46-52.

[92] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “MorphoNet: a deep image super resolution network using hierarchical and morphological feature generating residual blocks," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-5, 2021.

[93] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “MISNet: Multi-resolution level feature interpolating ultralight-weight residual image super resolution network," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-5, 2021.

[94] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “Development of new fractal and non-fractal deep residual networks for deblocking of JPEG decompressed images, in *Proc. IEEE International Conference on Image Processing (ICIP)*, pp. 1271-1275, 2020.

[95] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “PHMNet: A deep super resolution network using parallel and hierarchical multi-scale residual blocks," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-5, 2020.

[96] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “EFFRBNet: A deep super resolution network using edge-assisted feature fusion residual blocks," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-5, 2020.

[97] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “MGHCNET: A deep multi-scale granular and holistic channel feature generation network for image super resolution, in *Proc. IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1-6, 2020.

[98] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “SRNMFRB: A deep lightweight super resolution network using multi-receptive field feature generation residual blocks," in *Proc. IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1-6, 2020.

[99] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “UpDCNN: A new scheme for image upsampling and deblurring using a deep convolutional neural network," in *Proc. IEEE International Conference on Image Processing (ICIP)*, pp. 2154-2158, 2019.

[100] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “Deep JPEG image deblocking using residual maxout units," in *Proc. IEEE International Conference on Image Processing (ICIP)*, pp. 2681-2685,2019.

[101] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “SRSubBandNet: A new deep learning scheme for single image super resolution based on subband reconstruction," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-5, 2019.

[102] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “TPCNN: An ultralight-weight three-prior convolutional neural network for single image super resolution," In peer review.

[103] A. Esmaeilzehi, M.O. Ahmad and M.N.S. Swamy, “DSegAN: A Deep Light-weight Segmentation-based Attention Network for Image Restoration," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022.

[104] A. Esmaeilzehi, L. Ma, M.N.S. Swamy and M.O. Ahmad, “Analysis of the Robustness of Deep Light-weight Image Super Resolution Networks against Realistic Distribution Shifts in Test Images," To be submitted for publication.

[105] A. Esmaeilzehi, L. Ma, M.N.S. Swamy and M.O. Ahmad, “HighBoostNet: A Deep Light-weight Image Super Resolution Network using High-boost Residual Blocks," To be submitted for publication.

[106] W. Bae, J. Yoo and J.C. Ye, “Beyond Deep Residual Learning for Image Restoration: Persistent Homology-Guided Manifold Simplification," in *Proc. CVPRW*, 2017.

[107] F. Yu, V. Koltun and T. Funkhouser, “Dilated Residual Networks," in *Proc. CVPR*, 2017.