

IMAGE SUPER-RESOLUTION USING DEEP FEEDFORWARD NEURAL NETWORKS IN SPECTRAL DOMAIN

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

By
Onur Aydın
March 2018

Image Super-Resolution Using Deep Feedforward Neural Networks In
Spectral Domain

By Onur Aydın

March 2018

We certify that we have read this thesis and that in our opinion it is fully adequate,
in scope and in quality, as a thesis for the degree of Master of Science.

Selim Aksoy(Advisor)

Ramazan Gökberk Cinbiş(Co-Advisor)

Hamdi Dibekliolu

Sinan Kalkan

Approved for the Graduate School of Engineering and Science:

Ezhan Karaşan
Director of the Graduate School

ABSTRACT

IMAGE SUPER-RESOLUTION USING DEEP FEEDFORWARD NEURAL NETWORKS IN SPECTRAL DOMAIN

Onur Aydın

M.S. in Computer Engineering

Advisor: Selim Aksoy

Co-Advisor: Ramazan Gökberk Cinbiş

March 2018

With recent advances in deep learning area, learning machinery and mainstream approaches in computer vision research have changed dramatically from hard-coded features combined with classifiers to end-to-end trained deep convolutional neural networks (CNN) which give the state-of-the-art results in most of the computer vision research areas. Single-image super-resolution is one of these areas which are considerably influenced by deep learning advancements. Most of the current state-of-the-art methods on super-resolution problem learn a non-linear mapping from low-resolution images to high-resolution images in the spatial domain using consecutive convolutional layers in their network architectures. However, these state-of-the-art results are obtained by training a separate neural network architecture for each different scale factor. We propose a novel single-image super-resolution system with the limited number of learning parameters in spectral domain in order to eliminate the necessity to train a separate neural network for each scale factor. As a spectral transform function which converts images from the spatial domain to the frequency domain, discrete cosine transform (DCT) which is a variant of discrete Fourier transform (DFT) is used. In addition, in the post-processing step, an artifact reduction module is added for removing ringing artifacts occurred due to spectral transformations. Even if the peak signal-to-noise ratio (PSNR) measurement of our super-resolution system is lower than current state-of-the-art methods, the spectral domain allows us to develop a single model with a single dataset for any scale factor and relatively obtain better structural similarity index (SSIM) results.

Keywords: super-resolution, deep learning, Fourier transform.

ÖZET

SPEKTRAL ALANDA DERİN İLERİ BESLEMELİ SİNİR AĞLARI KULLANILARAK GÖRÜNTÜ SÜPER ÇÖZÜNÜRLÜĞÜ

Onur Aydın

Bilgisayar Mühendisliği, Yüksek Lisans

Tez Danışmanı: Selim Aksoy

Yardımcı Tez Danışmanı: Ramazan Gökberk Cinbiş

Mart 2018

Derin öğrenme alanındaki son gelişmelerle birlikte, bilgisayarla görüde öğrenme makineleri ve ana yaklaşımlar, sınıflandırıcılarla birlikte kodlanmış sabit özelliklerden uçtan uca eğitilmiş, bilgisayarla görü araştırma alanlarının çoğunda başarılı sonuçlar veren derin evrimsel sinir ağlarına dönüşmektedir. Tek görüntü süper çözünürlüğü, derin öğrenme gelişmelerinden önemli derecede etkilenen alanlardan biridir. Süper çözünürlük problemindeki mevcut en başarılı yöntemlerin çoğu, ağ mimarilerinde ardışık evrimsel katmanları kullanarak uzamsal alanda düşük çözünürlüklü görüntülerden yüksek çözünürlüklü görüntülere doğrusal olmayan bir eşleştirme öğrenirler. Bununla birlikte, bu sonuçlar, her bir farklı ölçek faktörü için ayrı bir sinir ağı mimarisi eğitimi ile elde edilir. Her ölçek faktörü için ayrı bir sinir ağının eğitilmesinin gerekliliğini ortadan kaldırmak için, spektral tanım kümesinde sınırlı sayıda öğrenme parametresine sahip yeni bir tek görüntü süper özünürlük sistemi öneriyoruz. Görüntüleri uzamsal tanım kümesinden frekans tanım kümesine dönüştüren bir spektral dönüşüm fonksiyonu olarak, ayrık Fourier dönüşümünün bir varyantı olan ayrık kosinüs dönüşümü kullanılır. Buna ek olarak, işlem sonrasında, spektral dönüşümlerden dolayı meydana gelen yapay salınımların kaldırılması için bir salınım azaltma modülü eklenmiştir. Süper-çözünürlük sistemimizin en yüksek sinyal-gürültü oranı ölçümü mevcut başarılı yöntemlerden daha düşük olsa bile, spektral tanım kümesi, herhangi bir ölçek faktörü için tek bir veri kümesi ile tek bir model geliştirmemizi ve nispeten daha iyi yapısal benzerlik indeksi sonuçları elde etmemizi sağlar.

Anahtar sözcükler: süper-çözünürlük, derin öğrenme, Fourier dönüşümü.

Acknowledgement

First of all, I would like to thank my supervisors, Dr. Selim Aksoy and Dr. R. Gökberk Cinbiş, for their excellent guidance, encouragement, motivation and support during my studies. In addition to their outstanding research contribution, from them, I have learned an effective 'style' which I could use in every area throughout my life.

I am grateful to my jury members, Dr. Hamdi Dibeklioglu and Dr. Sinan Kalkan for kindly accepting to spend their valuable time to review and evaluate my thesis.

Finally, I must express my very profound gratitude to my family and friends for providing support throughout my years of study and through the process of researching and writing this thesis.

Specifically, I would like to express my gratitude to my mother Nihal Aydın for her endless love and my father Nusret Aydın for his enormous power, assistance and belief. Additionally, I would like to thank to my sister Burcu Aydın for her life guidance and fellowship, Sezer Aydın for her encouragement, Çağla Astarıcı for her unlimited support and incredible patience. Finally, I would like to thank to lovely Fındık family members; Eda, Erkut and Eren.

Contents

1	Introduction	1
1.1	Overview	1
1.2	Scope of the Thesis	3
1.3	Outline of the Thesis	4
2	Related Work	5
2.1	Machine Learning Background	5
2.2	Deep Learning Background	7
2.2.1	Introduction to Neural Networks	7
2.2.2	Feedforward Neural Networks	8
2.2.3	Convolutional Neural Networks	9
2.2.4	Generative Adversarial Networks	11
2.2.5	Transfer Learning	11
2.3	Related Work on Super-Resolution	12

2.3.1	Traditional Methods	12
2.3.2	Deep Learning Methods	13
3	Method	18
3.1	Spectral Domain Transforms	18
3.1.1	Fourier Transform	19
3.1.2	Discrete Cosine Transform	20
3.2	The Developed Model	23
3.2.1	Super Resolution in Spectral Domain	23
3.2.2	Artifact Reduction in Spatial Domain	31
4	Experiments	33
4.1	Dataset Information	33
4.1.1	Training Sets	34
4.1.2	Test Sets	34
4.2	Model Evaluation	35
4.2.1	Peak Signal-to-Noise Ratio (PSNR)	36
4.2.2	Structural Similarity Index (SSIM)	36
4.3	Super-Resolution Experiments	37
4.3.1	Optimization Experiments	37

4.3.2	Evaluation Results	40
4.3.3	Output Images	45
5	Conclusions	52
5.1	Concluding Remarks	52
5.2	Future Work	53
A	Output Images	60

List of Figures

1.1	An Example of Complete Super-Resolution System	2
2.1	AlexNet Convolutional Neural Network Architecture	10
2.2	SRCNN Network Architecture	14
2.3	Architecture of Deep Laplacian Pyramid Networks	16
2.4	Super Resolution Architecture in Frequency Domain	17
3.1	An Example of Two Dimensional DCT	21
3.2	Two Dimensional Discrete Cosine Transform Basis	22
3.3	Spectral Analysis (x2)	25
3.4	Spectral Analysis Log (x2)	25
3.5	Spectral Analysis (x3)	25
3.6	Spectral Analysis Log (x3)	25
3.7	Spectral Analysis (x4)	25
3.8	Spectral Analysis Log (x4)	25

3.9	Bicubic Interpolation	26
3.10	SRCNN	26
3.11	VDSR	26
3.12	LapSRN	26
3.13	SRCNN - First Layer Filters in Spectral Domain	27
3.14	SRCNN - Second Layer Filters in Spectral Domain	28
3.15	SRCNN - Third Layer Filters in Spectral Domain	28
3.16	Most Problematic Frequency Regions	29
3.17	Our Spectral Super Resolution Network	30
3.18	AR-CNN Architecture and Feature Maps	31
3.19	Intermediate Steps of Our Super Resolution System	32
4.1	Set5 Images	34
4.2	Set14 Images	34
4.3	BSDS100 Images	35
4.4	Urban100 Images	35
4.5	Learning Curve for Different Learning Rates	38
4.6	Learning Curve for Different Number of Hidden Units	39
4.7	Learning Curve for PSNR Metric	41
4.8	Bicubic Interpolation	43

4.9 Ours	43
4.10 SRCNN	43
4.11 LapSRN	43
4.12 Output Images of Different SR Models - Set5 1 (x2)	46
4.13 Output Images of Different SR Models - Set5 2 (x2)	46
4.14 Output Images of Different SR Models - Set5 3 (x2)	47
4.15 Output Images of Different SR Models - Set5 5 (x2)	47
4.16 Output Images of Different SR Models - Set14 1 (x2)	48
4.17 Output Images of Different SR Models - Set14 2 (x2)	48
4.18 Output Images of Different SR Models - Set14 5 (x2)	49
4.19 Output Images of Different SR Models - Set14 9 (x2)	49
4.20 Output Images of Different SR Models - Set14 11 (x2)	50
4.21 Output Images of Different SR Models - Set14 13 (x2)	50
4.22 Output Images of Different SR Models - Set14 14 (x2)	51
A.1 Output Images of Different SR Models - Set5 1 (x2)	61
A.2 Output Images of Different SR Models - Set14 1 (x2)	61
A.3 Output Images of Different SR Models - Set14 2 (x2)	62
A.4 Output Images of Different SR Models - Set14 11 (x2)	62
A.5 Output Images of Different SR Models - Set5 1 (x3)	63

A.6	Output Images of Different SR Models - Set14 1 (x3)	63
A.7	Output Images of Different SR Models - Set14 2 (x3)	64
A.8	Output Images of Different SR Models - Set14 11 (x3)	64
A.9	Output Images of Different SR Models - Set5 1 (x4)	65
A.10	Output Images of Different SR Models - Set14 1 (x4)	65
A.11	Output Images of Different SR Models - Set14 2 (x4)	66
A.12	Output Images of Different SR Models - Set14 11 (x4)	66

List of Tables

2.1	The Comparison of Spatial and Frequency Domain SR Models . .	17
4.1	Quantitative Evaluation of SR Models (PSNR - SSIM)	42
4.2	Quantitative Evaluation for Different Setup	44

Chapter 1

Introduction

This chapter provides a brief overview on single-image super resolution problem. First, a simple definition of the problem is provided, which is followed by a discussion of the main challenges. Then, the scope of our model developed for this thesis is given. At the end, a brief outline of following chapters is provided.

1.1 Overview

The main purpose of the single-image super-resolution (SISR) problem is to reconstruct a high-resolution (HR) image from a single low-resolution (LR) image with minimum perceptual resolution loss. From the nature of the problem, super-resolution is an ill-posed inverse problem which searches an inverse function of downsampling and blurring operations. Therefore, in super-resolution problem, there is no unique solution and multiple high-resolution image candidates can be produced from a single low-resolution image. Solutions of super-resolution problem might be applied in different areas such as medical imaging, satellite imaging, enlarging consumer photographs, video surveillance, high definition television broadcasting and so on.

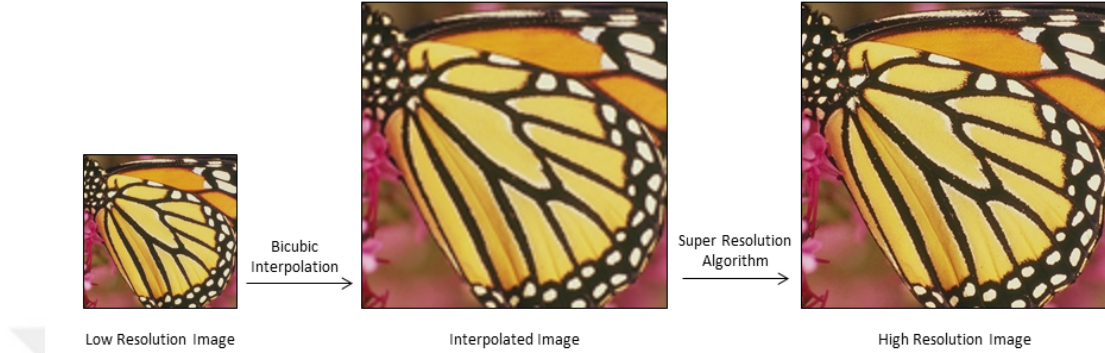


Figure 1.1: An Example of Complete Super-Resolution System

The single image super-resolution problem can be conceptualized from two perspectives. From signal processing perspective, super resolution problem is a kind of interpolation problem which could be considered as filling intermediate pixel values. From machine learning perspective, super-resolution problem is a kind of regression problem which looks for a continuous mapping from low-resolution images to high resolution images. Prior to the development of recent deep learning approaches, traditional machine learning techniques such as local linear regression [1, 2], dictionary learning [3, 4] and random forests [5] were being utilized in the supervised resolution approaches. Recently, deep learning based approaches have lead to significant improvements in the state-of-the-art models by learning an effective nonlinear mapping from the LR images to the HR images [6, 7, 8].

There are predominantly two ways for utilizing deep learning techniques in state-of-the-art super-resolution models. First, in the pre-processing step, most single-image super-resolution systems begin with re-sizing input images to intended dimensions using bicubic interpolation, which is one of the interpolation techniques in the literature. Then, the super-resolution problem reduces to learning a non-linear mapping for transforming the simple interpolation output to a realistic high-resolution image as shown in Figure 1.1. However, due to computational complexities and bias of the interpolation technique, in some super-resolution models, it is not preferable to use bicubic interpolation in pre-processing step. Rather, both dimensions and resolution of the images are increased during the

super-resolution construction. Moreover, while some methods progressively increase the resolution of the image, others explore learning a direct non-linear mapping from low-resolution images to high-resolution images. All of the approaches have some advantages and disadvantages based on some criteria.

In SISR problem, there are a few significant points which an SR model must satisfy. The first one is **precision** which indicates how close an SR model predicts the high-resolution image from its low-resolution counterpart image with minimum loss. Generally, in order to measure the difference between two images in the literature, peak signal-to-noise ratio (PSNR) or the structural similarity index (SSIM) are used. Besides the accurate recovery capability of the model, **efficiency** is another significant concern of SR problem. Especially due to practical reasons, at test time, the model must work fast and have a low computational cost. Another technical concern is **flexibility** of the scale factor. The scale factor is a constant that shows how much an image will grow and training a different model for each scale factor is extremely costly and not efficient. Additionally, as scale factor increases, learning a realistic mapping from low-resolution images to high-resolution images becomes harder. Hence, possible short-cut solutions such as first up-sampling with higher scale factor and then down-sampling to intended dimensions would fail rather than learning a scale factor-specific super-resolution mapping. For these reasons, it is preferable to develop a single model for any scale factor. It is expected from any super-resolution model to satisfy all these features and in the literature, different approaches have been developing for each.

1.2 Scope of the Thesis

In this thesis, we developed a new super-resolution model in the frequency domain with low model complexity. In previous works, convolution neural networks are trained on frequency domain for some specific tasks. For example, Rippel et al. show that convolutional neural networks successfully learn how to classify images completely in Fourier domain [9]. Specifically, discrete cosine transform

is used to compress weights of convolutional neural networks by protecting prediction performance [10]. In another work by Kumar et al, it is indicated that wavelet transforms together with convolutional neural networks generate more successful results than spatial domain methods for single image super resolution problem [11]. In a very recent paper, a super-resolution model is developed in Fourier domain and a novel non-linear activation function is learned to be used in frequency domain [12].

In this thesis, we explore the use of frequency-domain deep neural networks for the single-image super-resolution problem. In our approach, firstly, a non-linear mapping from bicubic interpolated images to high resolution images using deep feed-forward networks is learned completely in frequency domain. Then, in spatial domain, a pre-trained artifact reduction network is used to remove resultant frequency artifacts due to frequency domain transforms. We comprehensively evaluate our approach on benchmark datasets and discuss its advantages and disadvantages. Our model provides us a fast and efficient model to remove the necessity to train a different model for each scale factor where a single model would be able to do super resolution on any intended scale factor.

1.3 Outline of the Thesis

In Chapter 2, the necessary technical background on machine learning and the related work on single-image super-resolution problem is provided. In Chapter 3, the details of the proposed model on single image super-resolution problem is given. Additionally, the usage of ringing artifact reduction model in post-processing step is mentioned. In Chapter 4, experiments and results of the proposed model are compared with other state-of-the-art single image super-resolution models. In Chapter 5, conclusions of this study are proposed and possible future work are discussed.

Chapter 2

Related Work

This chapter provides the necessary technical background on machine learning and the related work on single-image super-resolution problem. In the first part, introductory machine learning definitions and techniques are mentioned. In the second part, fundamental neural network and deep learning techniques from feed-forward neural networks to convolutional neural networks are introduced in detail. In the third part, both traditional and deep learning state of the art approaches on single-image super-resolution problem are provided.

2.1 Machine Learning Background

Machine learning gives computers the ability to learn without being explicitly programmed. A computer program is considered to be learning if it improves its performance on future tasks after making observations about the world. There are different reasons why we want our program to learn. First of all, it is not possible to get rid of all situations that program may encounter during the runtime and it is hard to detect possible changes over time using traditional programming techniques. Additionally, sometimes, it is impossible to find a solution to solve a specific problem. For example, the human brain is great at recognizing objects,

but it is considered highly difficult to construct a computer program to do the equivalent, without using learning algorithms. For these reasons, it is a must to make computers to learn a good generalization from data or environment to solve many real-world problems. Machine learning can be categorized into three broad classes, depending on the characteristics of the data and the problem.

Supervised Learning: For given input-output example pairs, the definition of the supervised learning is learning a mapping function from input space to output space [13]. Generally, input space might be a feature vector or unstructured raw data such as image or text and output space might be a set of discrete class labels in classification problem or a set of continuous real numbers in regression problem. Single-image super-resolution problem is a kind of supervised learning, since the main aim is learning a mapping from continuous low-resolution images to continuous high-resolution images.

Unsupervised Learning: Unlike supervised learning, unsupervised learning algorithms use only input data without any output label and try to find a structure in the data on its own. There are different unsupervised learning algorithms which only use unlabeled data, such as clustering algorithms, autoencoders, and generative adversarial networks [13].

Reinforcement Learning: In the context of reinforcement learning, an autonomous agent must learn to perform a task by trial and error, without any guidance. The agent learns from a series of reinforcements, rewards or punishments [13]. Currently, reinforcement learning system based applications are capable to reach human-level performance on many tasks [14] and profoundly increased the performance in the robotics area [15].

According to problem and data type, more problem types are available in machine learning literature such as semi-supervised learning, weakly supervised learning, active learning, one-shot learning, zero-shot learning and so on [13].

2.2 Deep Learning Background

Deep learning allows computational models to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains. Deep learning discovers complex structures in large datasets by using the back propagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer [16].

Modern deep learning provides a powerful framework for supervised learning. By adding more layers and more units within a layer, a deep network can represent functions of increasing complexity. Most tasks that consist of mapping an input vector to an output vector, and that are easy for a person to do rapidly, can be accomplished via deep learning, given sufficiently large models and sufficiently large data sets of labeled training examples [17].

2.2.1 Introduction to Neural Networks

Artificial neural networks have been inspired by discoveries in neuroscience, particularly from the theory that mental activity principally consists of electrochemical signals in a network, called neurons. The neurons in the biological network are fired when a linear combination of its inputs surpasses a specific hard or soft threshold. The neurons are associated with neurotransmitters and the neural connections are the ones transporting the electrochemical signals. In the artificial neural network, the neurotransmitters can be considered as weights and the neurons as activation functions. The neural network consist of a collection of nodes that are connected together into a network and the properties of the network are determined by its topology and the properties of each node in the network [17].

Mathematically, a neuron is modeled as Equation 2.1. x_i is the input value, w_{ij} is the corresponding weight for the connection from i^{th} neuron to j^{th} neuron, b_j is the corresponding bias unit and f is the nonlinear activation function. In the literature, in order to train deep networks effectively, there are various activation functions such as sigmoid, hyperbolic tangent, rectified linear unit (RELU) [18], exponential linear unit (ELU) [19] and so on.

$$y_j = f\left(\sum_{i \in I} x_i w_{i,j} + b_j\right) \quad (2.1)$$

The main target of neural network learning systems is the finding the optimal weights and bias units to minimize problem-specific loss function. In order to minimize loss function, the training samples are passed through the network and the network output is compared with the actual output. The resulting error is used to update the weights of the neurons such that the loss function decreases gradually. This is done using the backpropagation algorithm [20] [21]. Iteratively passing batches of data through the network and updating the weights, so that the loss function is decreased, is an optimization problem. Stochastic gradient descent (SGD), AdaGrad, RMSProp, Adam are some of the main optimization algorithms used to train deep neural networks [22].

2.2.2 Feedforward Neural Networks

Feedforward neural networks form the basis of many significant neural networks being used in the recent times, such as convolutional neural networks (CNN) and recurrent neural networks (RNN). Feedforward neural networks are a kind of directed acyclic graph which means that there are no feedback connections or loops in the network. Generally, these networks are known by multiple names such as vanilla neural network, multilayer perceptron, or fully connected networks.

Single layer neural network without any activation function is capable to learn a linear function ($y = xW$) and this network corresponds to linear regression problem. If an activation function is added to output nodes of this network, a logistic regression model, which is also a linear classification method ($y = f(xW)$),

is learned from this network. Since two networks learn a linear function, the problem of finding the best weight parameters turns to a concave optimization problem and only one local minimum point occurs in function space. For this reason, optimizing linear networks are relatively easier than deeper networks [17]. On the other hand, if at least one hidden layer is added to the network, the optimization problem turns to a convex problem and function space has multiple local minimum points. Therefore, due to convergence to the local minimum problem, optimizing the network turns to a harder problem. Although the difficulty of the training, multilayer networks are capable to learn nonlinear functions $y = f(f(xW_1)W_2)$ that increase the performance of the applications [17].

2.2.3 Convolutional Neural Networks

Convolutional neural networks are very similar to feedforward neural networks. They are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function. And they still have a loss function and all the training and optimization techniques for learning feedforward neural networks still apply.

One of the drawbacks of the feedforward neural networks is that the number of learnable weights dramatically increases as input dimension increases. Clearly, this full connectivity is wasteful and these parameters would quickly lead to overfitting. By convolution operation, convolutional neural networks look local connectivities and vastly reduce the number of parameters and make the forward function more efficiently compared to feedforward neural networks.

In the literature, there are dozens of convolutional neural network architectures for different tasks in various domains such as computer vision, natural language processing, speech recognition and so on. LeNet [21] is the first successful application of convolutional nets developed in 90s and AlexNet [23] is the network popularized convolutional networks in computer vision. AlexNet is the standard

convolutional network in the literature and the other popular architectures are derived from this network such as GoogLeNet [24], VGGNet [25] and ResNet [26]. In AlexNet architecture which is shown in Figure 2.1, four different layers are stacked consecutively. The first type of layer is **convolution layer** which performs the convolution operation on input image/activation. This layer computes a dot product between local regions in the input and learned weights on the corresponding kernel and determines the values of local regions in the output. The second type of layer is activation or **RELU layer**. This layer applies an element-wise activation function overall activation map. In order to ensure non-linearity of network, a non-linear activation function must be used. Otherwise, even if multiple layers are used, the network is only able to learn a linear function. In the standard architecture, rectified linear units (RELU) activation function is used. The third type of layer is **pooling layer**. This layer down-samples the activation map along spatial dimension. In the standard architecture, max pooling operation is used. Simply, in max pooling operation, highest activation map passed through this layer. Finally, the fourth type of layer is **fully connected layer**. After passing multiple convolution, RELU and pooling layers, an N-dimensional feature vector occurs at the end. Finally, this feature is classified using a dense feedforward neural network. Due to every node is connected to each other, this layer called fully connected layer in deep learning area. Additionally, in order to decrease the overfitting on fully connected layers, dropout regularization technique is used. In overall architecture, convolution, RELU, and pooling layers are responsible for feature extraction part and fully connected layer is responsible for classifying extracted features from previous layers [27].

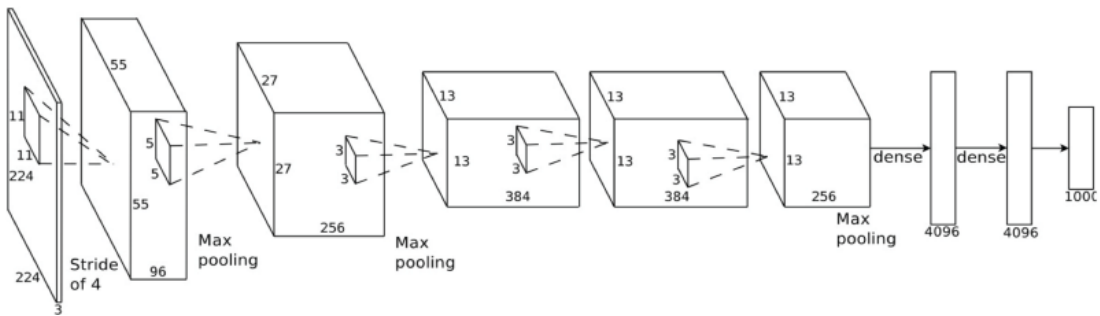


Figure 2.1: AlexNet Convolutional Neural Network Architecture [23]

2.2.4 Generative Adversarial Networks

Generative adversarial networks (GAN) are neural networks which learn to create artificial data similar to a set of input data. In generative adversarial networks, there two different neural architectures trained simultaneously. The first network (generator network) is trained for learning a mapping to generate artificial examples from latent noise. The second network (discriminator network) is trained for discriminating generated artificial data from real data. In other words, the discriminator network is just a binary classifier which says whether the input is fake or not [28, 29].

$$\min_G \max_D V(D, G) = \mathbf{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbf{E}_{z \sim p_z(z)}[\log(1 - D(G(x)))] \quad (2.2)$$

Since both networks are trained simultaneously, two different optimizers and two different loss functions (generative loss and discriminative loss) are used. Both networks are trained competitively such that while generator network intends to generate realistic indistinguishable samples, discriminator network differentiates real samples from artificial ones. This method of training a generative adversarial network is taken from game theory called the minimax game [28].

2.2.5 Transfer Learning

Traditional machine learning algorithms work under the assumption that there is sufficient number of labeled data for the related problem domain and models fail down when there is not enough data. In other words, for each task and domain, a different model has to be trained with available data. However, in deep learning era, transfer learning provides us to develop a system with small number of samples by transferring knowledge from one domain to another. For example, it is possible to do an MRI image classification system by transferring pre-trained weights of a convolutional neural network classifier which is trained with millions of real world photos. By transfer learning, for every separate task, the necessity to train deep networks from scratch is removed [30, 31].

2.3 Related Work on Super-Resolution

2.3.1 Traditional Methods

In this part, traditional methods on single-image super-resolution problem are introduced. These methods use traditional signal processing and machine learning techniques to increase the resolution of images.

Nearest Neighbor Interpolation

The nearest neighbor interpolation assigns the unknown pixel to the value of nearest pixel point and does not consider other pixels. This is the simplest and piecewise-constant interpolation technique in the literature.

Bilinear Interpolation

Bilinear interpolation assigns the unknown pixel value by fitting a linear curve to nearest four neighbors (2x2) and predict the value on the calculated curve. Bilinear interpolation is smoother than nearest neighbor interpolation.

Bicubic Interpolation

Bicubic interpolation is one of the widely used traditional interpolation methods because of its simplicity and accuracy. Additionally, it is widely used in most super-resolution models in pre-processing step and comparing models. Bicubic interpolation finds the unknown pixel by fitting a polynomial curve to nearest sixteen neighbors (4x4) and predict the value on the calculated curve. Bicubic interpolation is the smoothest interpolation technique among bilinear and nearest neighbor interpolation.

Adjusted Anchored Neighborhood Regression (A+)

Adjusted anchored neighborhood regression for fast super-resolution method is dictionary based SR method which learns a regression model from low resolution to high resolution over dictionaries. In dictionary-based regression model, l_2 norm regularization is used and this turns the problem into Ridge Regression [2].

Random Forest Super-Resolution (RFL)

RFL super-resolution uses random forests to learn a direct non-linear mapping from low-resolution to high-resolution [5]. Random forest regression gives very close results to locally linear regression and a regularization technique is developed for both input and output space.

Self-Similarity Super-Resolution (SelfExSR)

Self-similarity super-resolution method increases the size of internal dictionary size by using geometric transformations and no external dictionary is used. In order to handle perspective distortion and local shape deformation, geometric patch transformation is decomposed [32].

2.3.2 Deep Learning Methods

In this part, deep learning approaches on single-image super-resolution problem are introduced. In recent years, deep learning methods exceed the performance of state-of-the-art methods in many computer vision problems. Super-resolution problem has also been affected positively by recent deep learning advancements.

Super-Resolution Convolutional Neural Network (SRCNN)

SRCNN is the first and simplest deep learning architecture which uses a 3-layered convolutional neural network, for single-image super-resolution problem. Currently, it is the benchmark model for our development and each SR model which uses deep learning approach. Even if the performance of SRCNN model is exceeded currently, due to its simplicity and speed, SRCNN is still a valuable architecture in SR problem for comparing models.

In SRCNN (Figure 2.2), after applying bicubic interpolation in preprocessing step, the image is passed through three consecutive convolution layers. As stated in the paper, the operation in the first convolution layer is the patch extraction and representation layer, the second layer performs a nonlinear mapping operation and the third convolutional layer is the final layer which reconstructs the

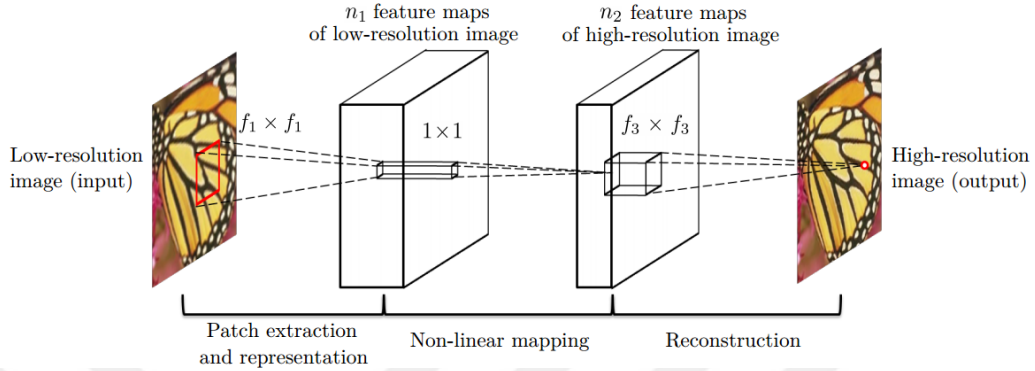


Figure 2.2: SRCNN Network Architecture [6]

high resolution image from activations of the second convolutional layer. At the end, high resolution image is recovered by using three consecutive convolution layers.

During the optimization of the network, l_2 loss function is minimized using stochastic gradient descent optimizer. Additionally, output high resolution image is reconstructed directly. One drawback of the SRCNN model is that for each scaling factor, a separate model is trained. On the other hand, SRCNN provides the flexibility that with larger datasets and deeper architectures, the SR performance might be outperformed [6].

Faster Super-Resolution Convolutional Neural Network (FSRCNN)

Differing from SRCNN that uses bicubic interpolation in preprocessing step, FSR-CNN introduces a deconvolution layer at the end of the network and the mapping is learned directly from the original low-resolution image. This architecture provides us to learn deeper convolutional layers with lower computational cost.

In the architecture of FSRCNN, seven convolutional layers and one deconvolutional layer is used and architecture is divided into five different regions according to their task such that feature extraction, shrinking, mapping, expanding and deconvolution. Like SRCNN, during the optimization, l_2 loss function is minimized and as an optimizer stochastic gradient descent is used. Additionally, direct high resolution image reconstruction is performed [33].

Very Deep Super Resolution (VDSR)

Unlike SRCNN architecture which uses only 3 convolutional layers, VDSR stacks 20 convolutional layers. In other words, VDSR is the deeper version of SRCNN architecture. Furthermore, VDSR added a residual connection from the input layer to output layer in order to solve the vanishing gradients problem while training deep networks. Differing from SRCNN, VDSR uses ten times higher learning rate and due to scale augmentation technique, only one network is trained for all scaling factors rather than training a separate network [7].

Super-Resolution Generative Adversarial Networks (SRGAN)

In SRGAN model, super resolution network is trained in adversarial setting. The generator learns to generate an HR image from input LR image. On the other hand, the discriminator network learns to discriminate generated super-resolved image and original high resolution image. Both networks are trained end-to-end in adversarial setting. The SRGAN method optimizes the network using the perceptual loss and the adversarial loss for photo-realistic SR [34].

Laplacian Super-Resolution Networks (LapSRN)

Unlike SRCNN, in LapSRN, no interpolation step is used during preprocessing step. Instead, the dimensions are progressively increased using Laplacian pyramid framework using 27 convolutional layers as shown in Figure 2.3. Additionally, residual connections are used in the architecture. During optimization, instead of l_2 loss function, Charbonnier loss function is used which is a robust loss function to handle outliers and more proper to the human visual perception.

The model presented in that paper consists of two parts such that feature extraction and image reconstruction. Transpose convolutional layer connects to two different layers. First is reconstruction layer and the other one is the convolutional layer that represents better resolutions. Contrary to other networks that are mentioned, the network designed in that paper has a less complex feature extraction. Because feature extraction is shared between lower and higher levels and complex mappings can be learned by this network better. During image reconstruction, input images are upsampled with the order of two and transposed

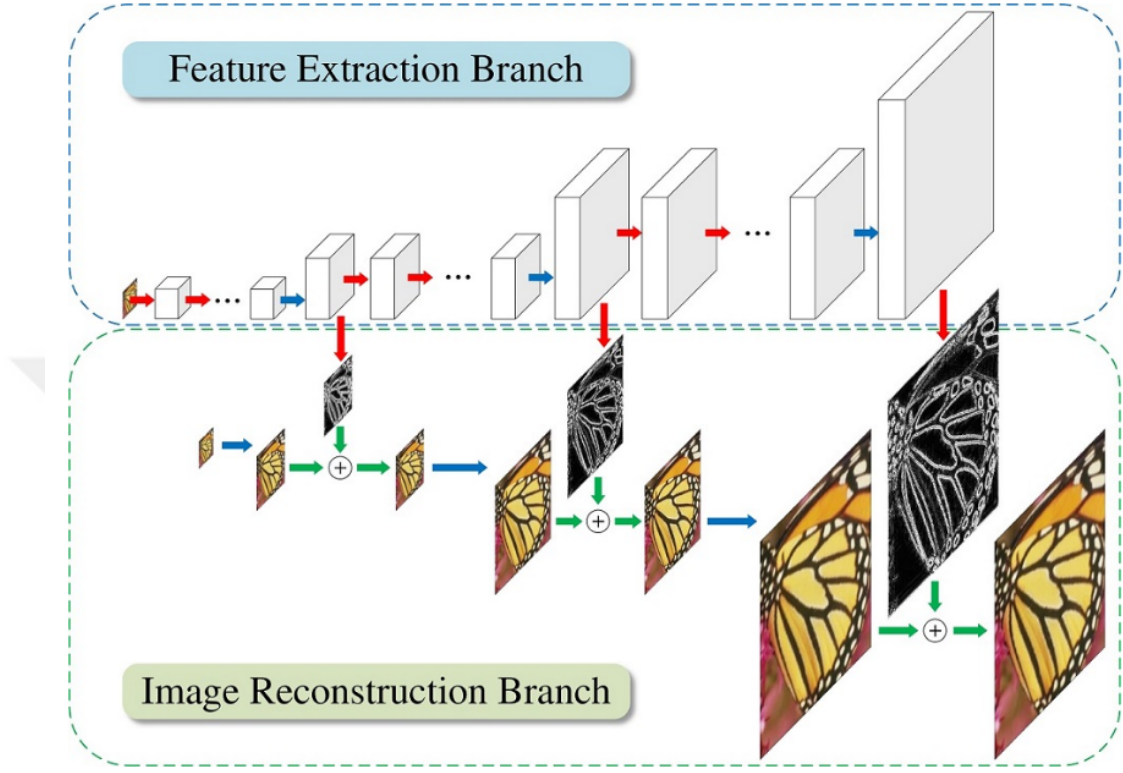


Figure 2.3: Architecture of Deep Laplacian Pyramid Networks [8]

convolutional layer. Then the pixel-wise sum of residual images from feature extraction and the upscaled image is used to produce a HR image as the output [8].

Wavelet Domain Super-Resolution (CNNWSR)

CNNWSR is a super-resolution network which trained for predicting discrete wavelet transform coefficients of target HR image. Then, using the reconstruction power of wavelet transform, the high resolution image is reconstructed from predicted wavelet coefficients using two-dimensional inverse wavelet transform. In the prediction model of CNNWSR, three convolutional layers are used like SRCNN architecture, except SRCNN produces one output image while CNNWSR predicts three output images which contain all wavelet coefficients. This is the first architecture combines the deep learning and spectral approaches in super-resolution problem. However, the main problem of the CNNWSR architecture is the scale limitation which stems from the nature of wavelet transform [11].

Frequency Domain Super Resolution (FourierSR)

In a very recent unpublished work, a Fourier domain approach to super resolution problem is made. In this paper, as shown in Figure 2.4, a deep network is constructed and learned totally in Fourier domain. Additionally, using properties of Fourier transform, such as convolution in the spatial domain is equivalent to multiplication in the frequency domain and vice versa, convolutional layers are treated as multiplications and non-linear activation functions are substituted by convolutions in frequency domain [12].

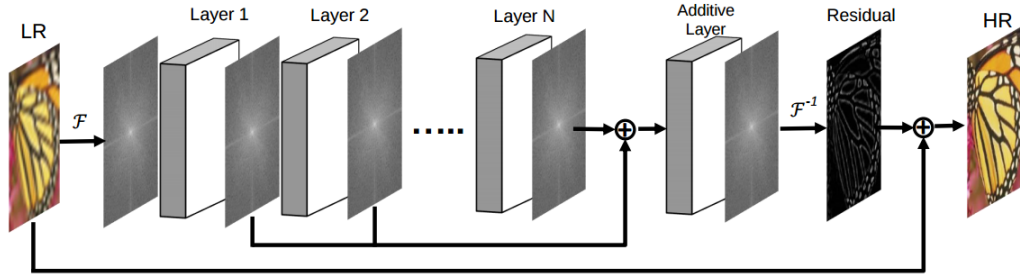


Figure 2.4: Super Resolution Architecture in Frequency Domain [12]

Although Fourier transform brings the computational efficiency, the frequency domain is not a proper domain for highly accurate super-resolution. From Table 2.1, it is observed that PSNR and SSIM values (PSNR/SSIM) of super resolution in Fourier domain are relatively lower than spatial domain methods.

Table 2.1: The Comparison of Spatial and Frequency Domain SR Models

Dataset	Scale	Bicubic	FourierSR	SRCNN
Set5	2	33.66/0.930	35.20/0.943	36.66/0.954
Set14	2	30.24/0.868	31.40/0.895	32.42/0.906
BSDS100	2	29.56/0.843	30.58/0.877	31.36/0.887
Set5	3	30.39/0.860	31.42/0.883	32.75/0.909
Set14	3	27.55/0.774	28.32/0.802	29.28/0.820
BSDS100	3	27.21/0.738	27.79/0.772	28.41/0.786
Set5	4	28.42/0.810	29.35/0.827	30.48/0.862
Set14	4	26.00/0.702	26.62/0.727	27.49/0.750
BSDS100	4	25.96/0.667	26.42/0.696	26.90/0.710

Chapter 3

Method

In this chapter, the details of our model for single-image super-resolution problem are given. First of all, spectral domain techniques to transform signals from time domain (or spatial domain) to frequency domain are investigated. Then, the usage and contributions of spectral domain techniques in super-resolution problem for our model are given in detail. In the last section, our complete super-resolution system is described in detail. In the initial part of the system, the neural network architecture and learning procedure to learn a mapping from low resolution images to high resolution images are described. At the end, artifact reduction network in post-processing step in order to eliminate spectral ringing artifacts is described.

3.1 Spectral Domain Transforms

In this section, spectral domain transform functions are investigated. Basically, spectral domain transforms the representation of a time (spatial) domain signal to the frequency domain. Two different spectral transforms are investigated in this section such that Fourier transform and its variant, discrete cosine transform which are a quite popular transforms in signal processing domain.

3.1.1 Fourier Transform

First of all, Fourier transform and its different variants are provided. The basic definition of Fourier Transform is decomposing any differentiable real or complex signal into different complex exponential functions which oscillate at different frequencies. In other words, Fourier Transform converts the representation of a time domain signal to the frequency domain. Fourier Transform provides us to analyze signals in the frequency domain which might be easier than time domain analysis in some situations. However, due to basis functions, Fourier Transform produces complex outputs even if the input is a real signal. For a continuous signal f and its frequency domain representation F , one-dimensional Fourier Transform is defined as follows [35]:

$$F(w) = \frac{1}{2\pi} \int_{-\infty}^{\infty} f(x) e^{-jwx} dx \quad (3.1)$$

Additionally, the signal transformed to the frequency domain is reconstructed back via one-dimensional Inverse Fourier Transform which is defined as follows:

$$f(x) = \int_{-\infty}^{\infty} F(w) e^{jwx} d\tau \quad (3.2)$$

3.1.1.1 1-D Discrete Fourier Transform

Fourier Transform is a complex-valued operation for continuous signals. However, in digital systems, every signal is discrete and things that are valid for continuous signals begin to lose their validity for discrete signals. For this reason, Discrete Fourier Transform (DFT) is specifically designed to be used in digital systems. For a discrete signal f and its frequency domain representation F , one-dimensional Discrete Fourier Transform is defined as follows [35]:

$$F[u] = \frac{1}{M} \sum_{x=0}^{M-1} f[x] e^{-2\pi j(\frac{ux}{M})} \quad (3.3)$$

The signal transformed to the frequency domain is reconstructed back via one-dimensional Inverse Discrete Fourier Transform which is defined as follows:

$$f[x] = \sum_{u=0}^{M-1} F[u] e^{2\pi j(+\frac{ux}{M})} \quad (3.4)$$

3.1.1.2 2-D Discrete Fourier Transform

Until now, Fourier Transforms for signals in one-dimensional space are mentioned. In this section, Fourier Transforms for two-dimensional discrete signals such as image, video, etc., are introduced. For a two dimensional discrete signal f and its frequency domain representation F , two-dimensional Discrete Fourier Transform is defined as follows [35]:

$$F[u, v] = \frac{1}{NM} \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f[x, y] e^{-2\pi j(\frac{xu}{N} + \frac{yv}{M})} \quad (3.5)$$

The signal transformed to frequency domain is reconstructed back via two-dimensional Inverse Discrete Fourier Transform which is defined as follows:

$$f[x, y] = \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} F[u, v] e^{2\pi j(\frac{xu}{N} + \frac{yv}{M})} \quad (3.6)$$

3.1.2 Discrete Cosine Transform

In previous section, Fourier Transform and its variants are given in detail. However, even if the input signal is real, Fourier Transforms are complex valued operations which output a complex signal. On the other hand, discrete cosine transform which decomposes a signal into cosine functions which oscillates at different frequencies is a real valued transform and gives a real signal in output.

3.1.2.1 1-D Discrete Cosine Transform

For a one dimensional discrete signal f and its frequency domain representation F , one-dimensional Discrete Cosine Transform is defined as follows [35]:

$$F[u] = a(u) \sum_{x=0}^{N-1} f[x] \cos\left(\frac{\pi(2x+1)u}{2N}\right) \quad (3.7)$$

The signal transformed to frequency domain is reconstructed back via two-dimensional Inverse Discrete Cosine Transform which is defined as follows:

$$f[x] = \sum_{u=0}^{N-1} a(u) F[u] \cos\left(\frac{\pi(2x+1)u}{2N}\right) \quad (3.8)$$

3.1.2.2 2-D Discrete Cosine Transform

For a two dimensional discrete signal f and its frequency domain representation F , two-dimensional Discrete Cosine Transform is defined as follows [35]:

$$F[u, v] = a(u)a(v) \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f[x, y] \cos\left(\frac{\pi(2x+1)u}{2N}\right) \cos\left(\frac{\pi(2y+1)v}{2M}\right) \quad (3.9)$$

The signal transformed to frequency domain is reconstructed back via two-dimensional Inverse Discrete Cosine Transform which is defined as follows:

$$f[x, y] = \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} a(u)a(v) F[u, v] \cos\left(\frac{\pi(2x+1)u}{2N}\right) \cos\left(\frac{\pi(2y+1)v}{2M}\right) \quad (3.10)$$

Where $a(x)$ is defined as:

$$a(x) = \begin{cases} \sqrt{\frac{1}{N}}, & x = 0 \\ \sqrt{\frac{2}{N}}, & x \neq 0 \end{cases} \quad (3.11)$$

In the Figure 3.1, an example of 2-D DCT of given image is provided. As it is seen, visually, it is impossible to understand what the image is and there is no local correlation among pixel values in spectral domain. Top left corner values represent low, bottom right corner values represent high frequencies.



Figure 3.1: An Example of Two Dimensional DCT

Additionally, in the Figure 3.2, the basis functions for two-dimensional discrete cosine transform is provided. Each basis function is a two-dimensional representation of the combination of two cosine functions which are oscillated in different frequencies. As going from top to bottom and left to right, frequencies of cosine functions increase. For this reason, basis functions in the top left corner have lower frequency values and basis functions in the bottom right corner have higher frequency values. The first basis function which is located in top left corner is constant DC term and this basis function consists of all ones. In other words, DC term is equivalent to calculate the mean value of transformed input patch:

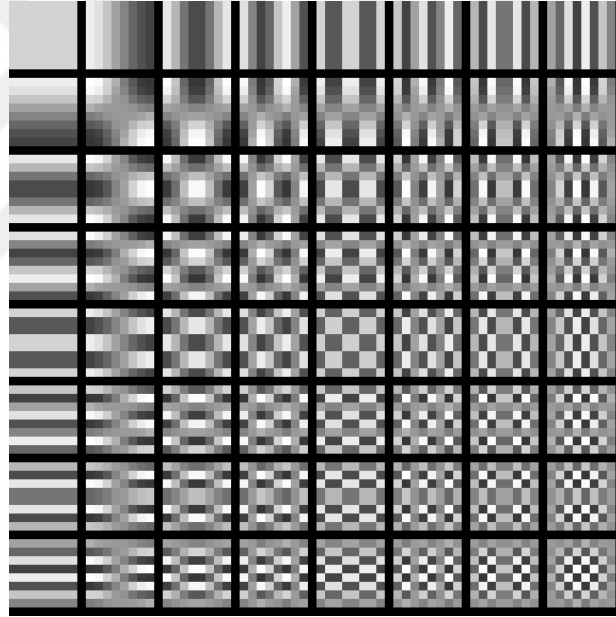


Figure 3.2: Two Dimensional Discrete Cosine Transform Basis

A significant feature of discrete cosine transform is energy compaction property. By this property, DCT preserves considerably more information in low frequencies. Discrete cosine transform is the best approximation of Karhunen-Loève transform which is theoretically ideal linear dimension reduction technique. The principal component analysis is a specific version of Karhunen-Loève transform and has data dependent basis functions. Constant cosine basis functions of discrete cosine transform ideally approximate Karhunen-Loève transform.

3.2 The Developed Model

In this part, the details of the developed model, network architectures and their implementations are given. First of all, initial super-resolution model is developed using neural networks in the spectral domain. After super-resolution phase, another network is used in spatial domain in order to decrease ringing artifact effect which is caused by spectral transforms.

3.2.1 Super Resolution in Spectral Domain

In our super-resolution model, firstly, a significant analysis and observation between low-resolution and high-resolution images in spectral domain are made. Additionally, extra observations over different super-resolution models are made in order to compare bicubic interpolation. Then, from these observations, a network architecture and learning system is constructed to learn a mapping from low-resolution to high-resolution. In this section, the details of the developed super-resolution model and the architecture are given.

3.2.1.1 Model Development

Before constructing network architecture, an initial data analysis is made in the spectral domain. Most super-resolution problems use bicubic interpolation technique to resize low-resolution images to intended dimensions in pre-processing step. For this reason, this analysis is made between bicubic interpolated images and their high resolution versions. Therefore, it is tried to investigate the exact problem behind bicubic interpolation.

First of all, a low resolution image is resized to intended dimensions using bicubic interpolation method (e.g. scale factors of x2, x3, x4). Then, each image is divided into patches with a stride of patch size. Independent of the scale factor, each patch corresponds to 16x16 patch in low resolution version. Hence, patch

dimensions are 32x32, 48x48 and 64x64 for scale factors 2, 3 and 4 respectively. After generating patches, two-dimensional discrete cosine transform of each patch is computed for both interpolated and high resolution images. In order to observe the weaknesses of bicubic interpolation, the mean square error between bicubic interpolated patches and high resolution patches is calculated and averaged across patches. Finally, results of mean square error analysis in spectral domain is attained.

For three different scale factors, the average mean square error calculations are given in Figures 3.3, 3.5 and 3.7 and their logarithmic versions are given in Figures 3.4, 3.6 and 3.8. In these figures, each pixel in images corresponds a frequency value over a 2-dimensional array. Simply, top left corner pixels represent low frequencies, bottom right corner pixels represent high frequencies and remaining regions represent mid-frequencies. From mean square error analysis, it would be said that bicubic interpolation works well (or less problematic) for low and high frequencies, but, the actual problem is in mid-frequencies. However, in most papers, due to problems on edges of super-resolution images, it is blindly stated that SR algorithms fail to predict high frequency terms.

After these observations, we concentrate on fixing this problematic band-pass region in our neural network model. Moreover, for different scale factors, it is observed that a common mid-band region is problematic and this common pattern allows us to develop a single model for all scale factors. It is important to state that even current and previous state-of-the-art super resolution methods do not have this property and it is still needed to develop a separate network for each scale factor which brings the requirement of more training duration and more data with different scales. Additionally, during test time, it is required to hold whole network models in memory for every scale factor. For these reasons, it is better to design a super resolution model which fits all scale factors.

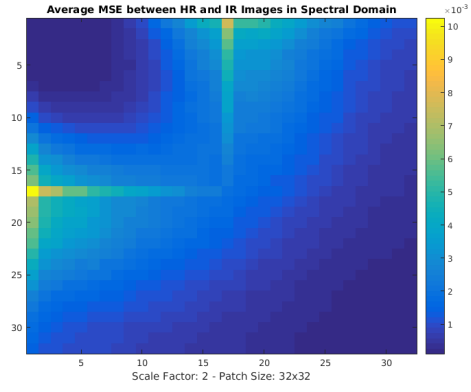


Figure 3.3: Spectral Analysis (x2)

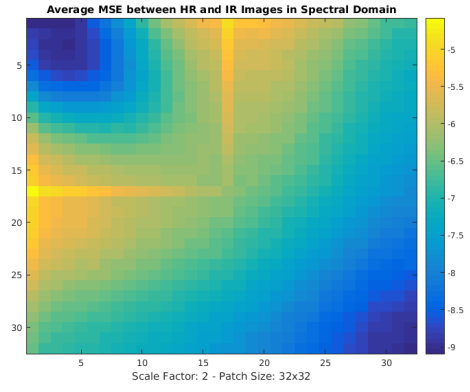


Figure 3.4: Spectral Analysis Log (x2)

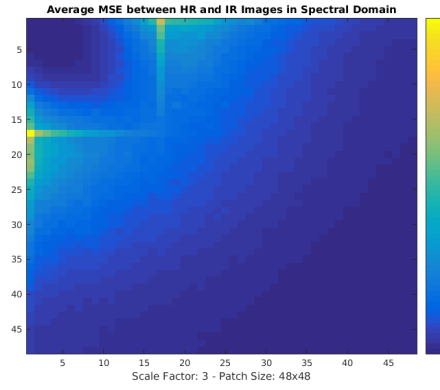


Figure 3.5: Spectral Analysis (x3)

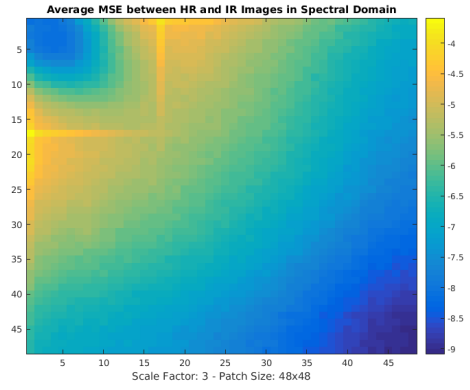


Figure 3.6: Spectral Analysis Log (x3)

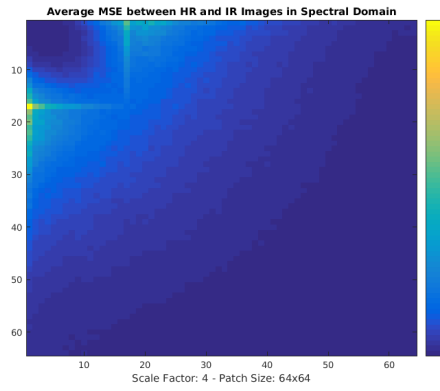


Figure 3.7: Spectral Analysis (x4)

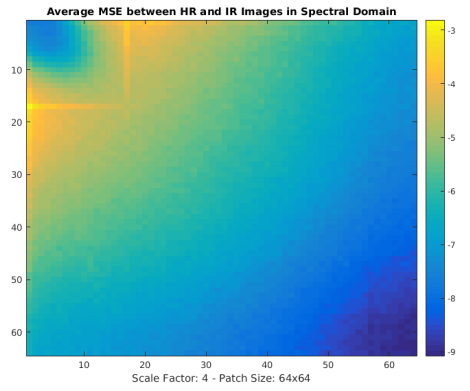


Figure 3.8: Spectral Analysis Log (x4)

After observing the mean square error between bicubic interpolated and high resolution images in the spectral domain, the same analysis is made for different super resolution models such as SRCNN, VDSR, and LapSRN and it is observed how each SR model fixes low resolution images in the spectral domain. From Figures 3.9, 3.10, 3.11 and 3.12, very similar outputs are obtained with bicubic interpolation except lower frequency values are fixed better. However, there are still some problems in mid-frequency bands. The performance of SR algorithms in sorted ascending order are bicubic interpolation, SRCNN, VDSR and LapSRN and this pattern is also observed in spectral domain quite clearly.

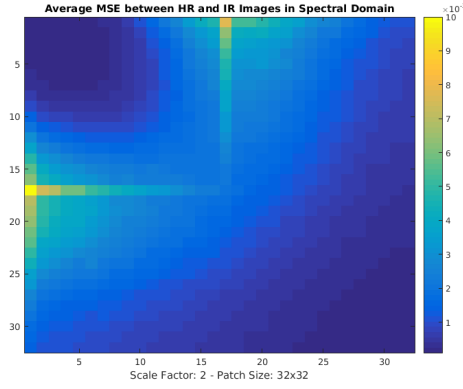


Figure 3.9: Bicubic Interpolation

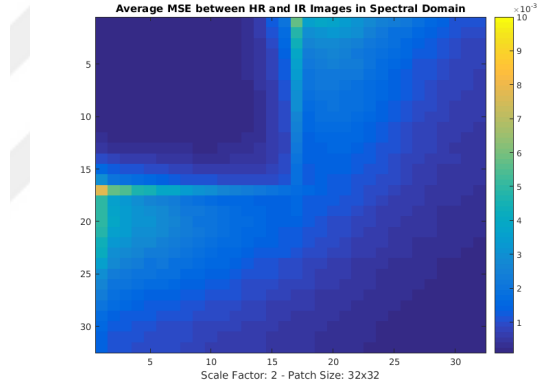


Figure 3.10: SRCNN

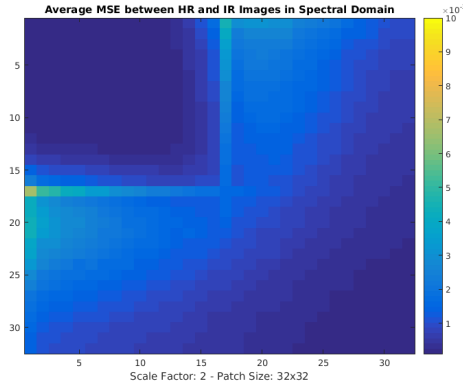


Figure 3.11: VDSR

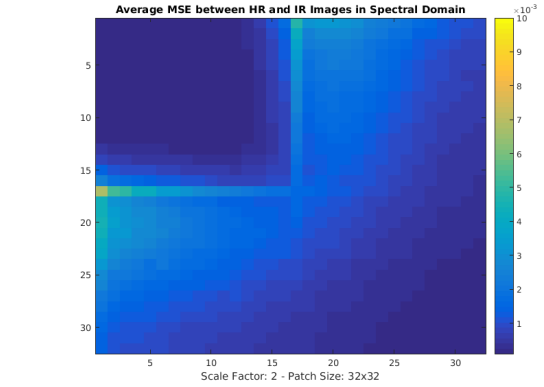


Figure 3.12: LapSRN

After analyzing different super resolution networks in the spectral domain, as stated in previous section, it is understood that these algorithms are able to recover lower frequencies but mid-frequencies are still problematic. After these data input-output analysis, spatial filters learned in SRCNN architecture are analyzed

in spectral domain in order to understand what SRCNN learns in the spectral domain. SRCNN is three-layered convolutional neural network architecture and in the first layer 64 9x9 filters, in the second layer 32 5x5 filters and in the third layer 32 5x5 filters are available. Additionally, there are different filter size setups are available for SRCNN architecture, but this setup is arbitrarily selected among others.

In the analysis, two-dimensional discrete cosine transform of each spatial filter is taken. Moreover, for better visual quality, 64-point FFT's are used in two-dimensional DCT calculation. The learned filters in spectral domain are shown in Figures 3.13, 3.14 and 3.15. From filters below, learned filters are generally low pass filters which give more spikes in low-frequency regions. Additionally, even if for some filters, there are a few spikes occur in high and middle frequencies, at the end, these filters are not able to fix these regions.

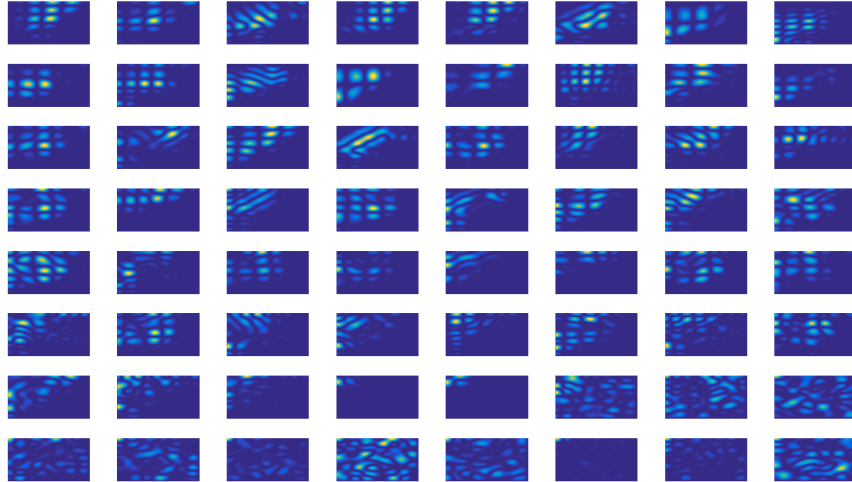


Figure 3.13: SRCNN - First Layer Filters in Spectral Domain

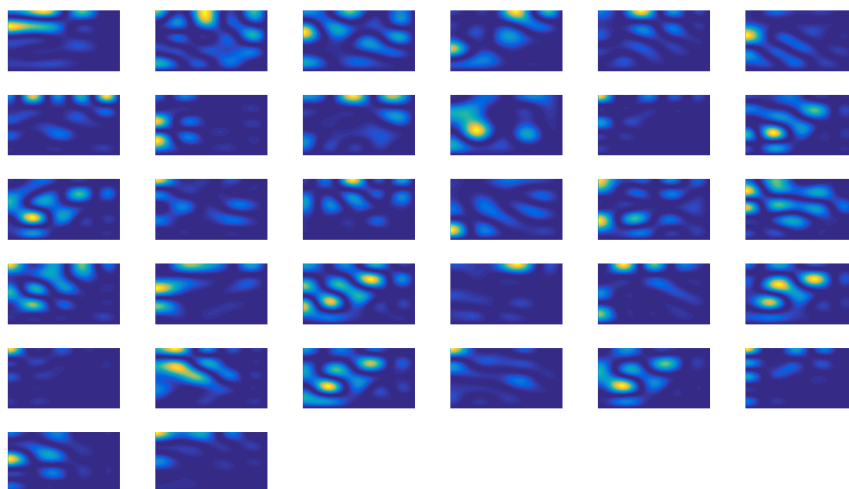


Figure 3.14: SRCNN - Second Layer Filters in Spectral Domain

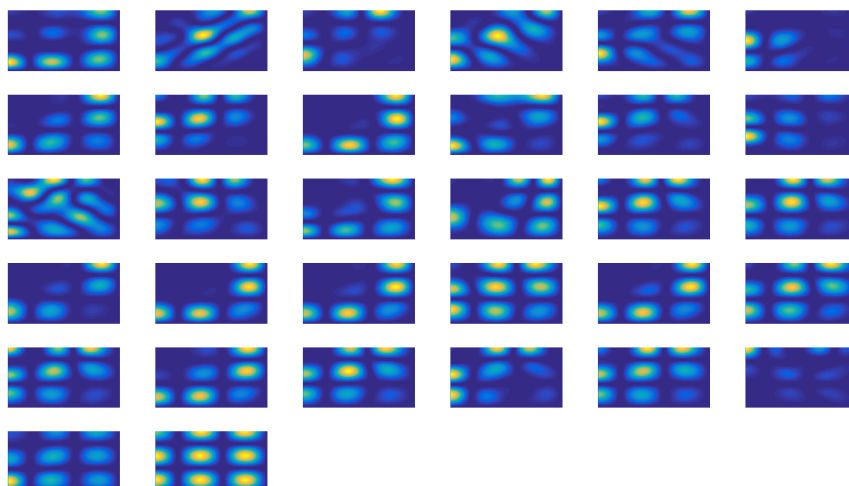


Figure 3.15: SRCNN - Third Layer Filters in Spectral Domain

3.2.1.2 The Network Architecture

The main function of our neural network architecture is learning a mapping from low resolution images to high resolution images so that the problematic region which is discussed in the previous section is fixed. Additionally, the developed network must be usable for any scale factor. Therefore, input and output dimensions must be compatible for each scale factor. For these reasons, most 512 erroneous frequency values in mean square error analysis, which is shown in Figure 3.16, are determined and for input and output, these 512 mid-frequency values are used. For scale factor of two, images are divided into 32x32 patches and there are a total of 1024 two dimensional frequency values in the spectral domain. Therefore, half of the patches are included in the learning process.

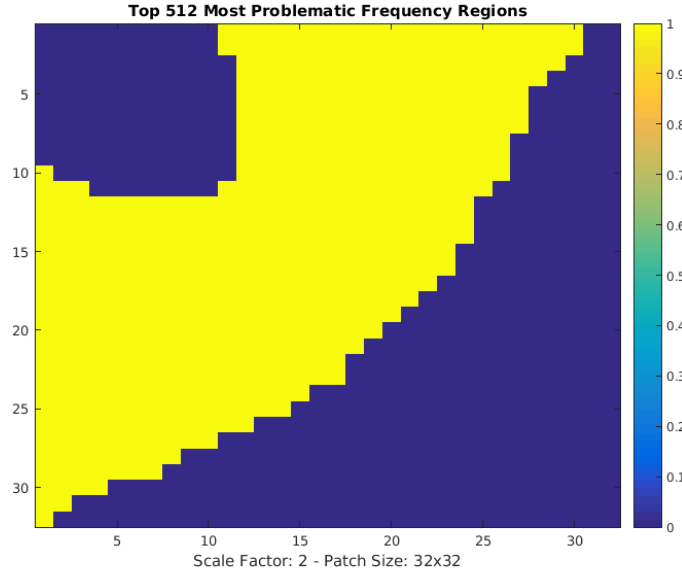


Figure 3.16: Most Problematic Frequency Regions

For the input to the neural network, the problematic region of bicubic interpolated patches and for the output to the neural network, the corresponding problematic region of high resolution patches are given. Then, a neural network architecture is trained for minimizing the mean square error between predicted and high resolution versions of the problematic frequencies. In order to learn this mapping, a feedforward fully connected neural network is constructed. Unlike

the spatial domain, there is no homogeneous distribution between pixel values in the spectral domain. For this reason, fully connected networks are preferred over convolutional neural networks. In the network architecture, a four-layered fully connected neural network is used and in each layer, 512 hidden units are used. Therefore, total of 1050624 parameters are learned in the network architecture. All weights are initialized using Xavier normal initializer [36] and the network is trained using Adam optimizer [22] with the batch size of 128. An overview of our super-resolution system is shown in Figure 3.17.

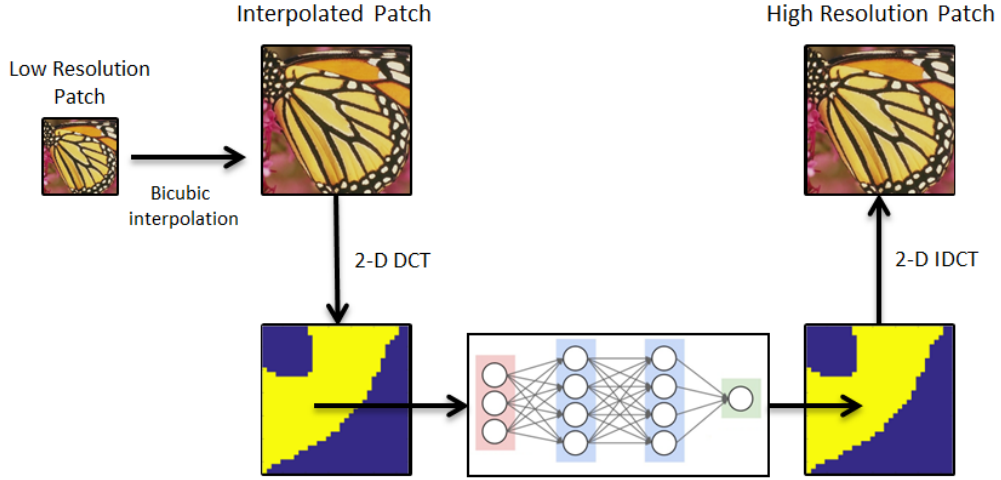


Figure 3.17: Our Spectral Super Resolution Network

In order to prevent overfitting, after each fully connected layers, dropout layers are placed [37]. Moreover, because discrete cosine transform mostly produces numbers between -1 and 1 (except DC term), the hyperbolic tangent function is used rather than rectified linear unit which is a better choice for vanishing gradients problem [18]. Especially in the output layer, using hyperbolic tangent function allow us to constrain output to be between -1 and 1. Beside all these setups, more optimization techniques, activation function, and training procedures are tried such as RMSProp optimizer, SRELU activation function or batch normalization [38], the best performance is obtained using given setup.

3.2.2 Artifact Reduction in Spatial Domain

After training our spectral super resolution network, some ringing artifacts are observed due to used spectral transforms. Since Fourier transform of a rectangular function is sinc function those oscillations approach to infinity, these artifacts are observed when some changes are made in the spectral domain. Because Fourier transform and discrete cosine transform are very similar functions, a similar situation is also valid for DCT. This is a very common problem for many image processing applications. For instance, JPEG compression algorithm uses discrete cosine transform and compression process in spectral domain generates similar artifacts in the spatial domain.

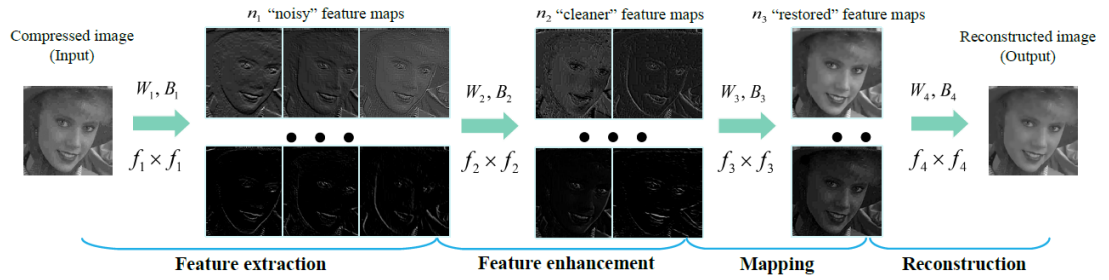


Figure 3.18: AR-CNN Architecture and Feature Maps [39]

Like super resolution problem, deep neural networks are also used to solve compression artifact reduction problem. For instance, AR-CNN is the first deep learning architecture to reduce compression artifacts from JPEG images. As shown in Figure 3.18, AR-CNN architecture which uses consecutive convolutional layers is very close to SRCNN super resolution architecture and it is used as a baseline method for artifact reduction problem in the literature [39]. After AR-CNN model, a deeper version of AR-CNN architecture is developed. In this architecture, 8 consecutive convolution layers are used and between these convolutional layers, skip connections are added to prevent gradient vanishing problem [40].

After our spectral super resolution model, in order to remove ringing artifacts, a pre-trained model of AR-CNN model is used. Even if the model is not re-trained end-to-end manner, a considerable amount of increase in our super-resolution

model is obtained. Even if better results would be obtained from our complete super resolution system, we don't want to end-to-end train a separate network to solve artifact reduction problem. Because, at this time, the problem would turn to learn an SRCNN model in the spatial domain. Additionally, AR-CNN model is trained on a completely different dataset. Therefore, it is more valuable to transfer the artifact reduction knowledge to super-resolution domain. In conclusion, when complete super resolution system is thought, using spectral transforms in the initial step, super-resolution problem is converted to artifact reduction problem and it is solved using pre-trained artifact reduction neural networks in the spatial domain. In the Figure 3.19, intermediate steps of our complete single-image super-resolution system and increasing of PSNR value in each step is shown.

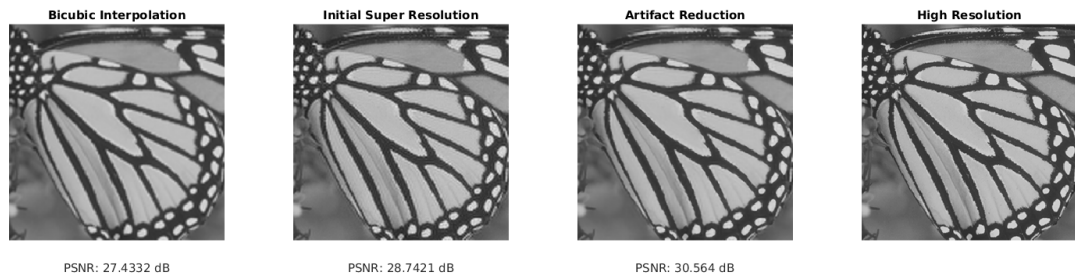


Figure 3.19: Intermediate Steps of Our Super Resolution System

Chapter 4

Experiments

In this chapter, the details of our experimental setup and completed experiments are given. In the first part, datasets used during experiments both in training and testing phase are mentioned. Then, evaluation metrics for super-resolution problem which are widely used in the literature are described mathematically. Finally, evaluation metrics and output images of experiments done for the developed model is shared.

4.1 Dataset Information

An advantageous of super-resolution problem is that any kind of image data can be used directly in both training and testing phases and unlike some problems like classification, detection or segmentation, no human effort is needed to label images manually. Any image is being both data and label at the same time in super-resolution problem. The original image is taken as high-resolution image or in other words output label. Then, low-resolution image or in other words input data is obtained by down-sampling original high-resolution image. At the end, input data and output label are prepared without any cost using any image dataset. In the following section, details of training and test sets are given.

4.1.1 Training Sets

In training phase, three different image datasets are used. These are BSDS200, General100, and T91. There are 200, 100 and 91 images in datasets respectively and there are 391 images in total. These are the widely used datasets in the literature for single-image super-resolution problem.

4.1.2 Test Sets

In test phase, four different image datasets, Set5, Set14, BSDS100, and Urban100, are used. These are shown in Figure 4.1, 4.2, 4.3 and 4.4 respectively. There are 5, 14, 100 and 100 images in datasets respectively and there are 219 images in total. Additionally, Set5 dataset is used as development set and it is used for model selection and parameter tuning. Especially Set5 and Set14 datasets are very commonly used in the literature. In the following figures, samples from image datasets are shared for gaining better insight about images.



Figure 4.1: Set5 Images

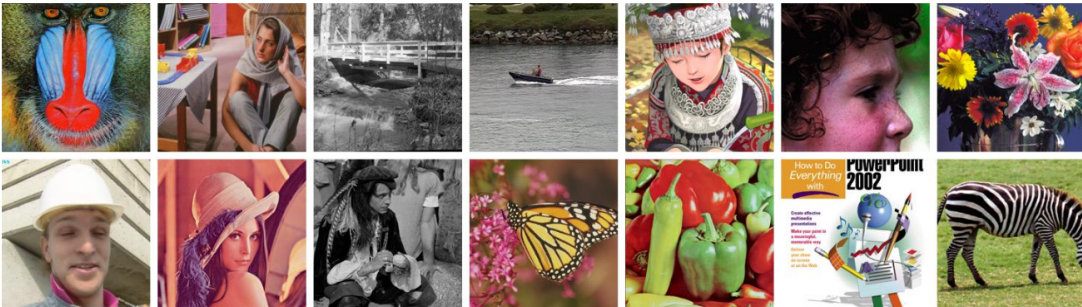


Figure 4.2: Set14 Images

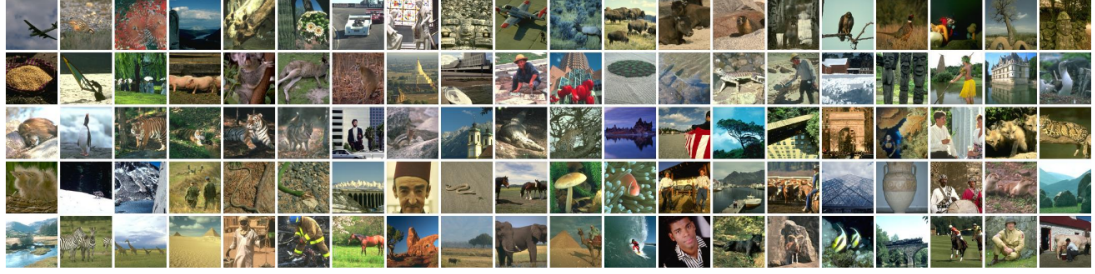


Figure 4.3: BSDS100 Images

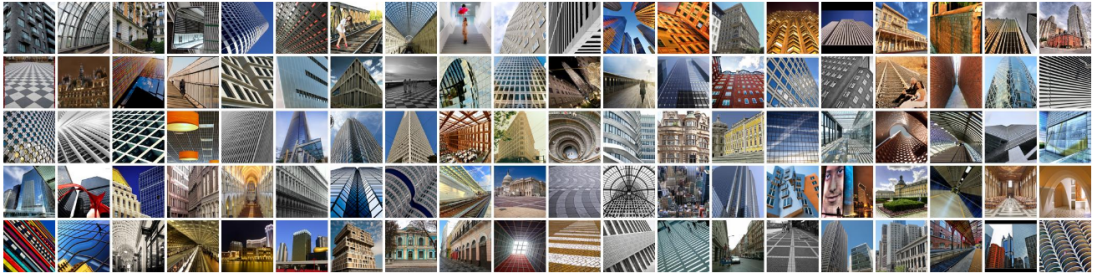


Figure 4.4: Urban100 Images

4.2 Model Evaluation

In this section, evaluation metrics used in super-resolution problem are mathematically described. There are two different evaluation metrics used in this thesis. These are Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). For super-resolution problem, it is preferable to obtain high PSNR and SSIM values from models. However, high PSNR or SSIM value does not always mean high perceptual quality. For instance, sometimes, it is possible to obtain a high-resolution image from low-resolution with lower PSNR, but higher perceptual quality. For this reason, current evaluation metrics are a bit problematic and finding a better evaluation metric for super-resolution problem is still an open research problem. In the following section, mathematical details of peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) are given.

4.2.1 Peak Signal-to-Noise Ratio (PSNR)

Peak signal-to-noise ratio is a metric which measures the quality difference between two input images. Theoretically, higher PSNR means higher image quality. For $M \times N$ dimensional images I_1 and I_2 , PSNR is described as follows:

$$PSNR = 10 \log_{10} \left(\frac{R^2}{MSE} \right) \quad (4.1)$$

$$MSE = \frac{\sum_{M,N} [I_1(m,n) - I_2(m,n)]^2}{MN} \quad (4.2)$$

In the equation, MSE is the mean square error between two images and R is the maximum range of the input image. For instance, for double precision floating data type, R is equal to 1 and 8-bit unsigned integer data type R is equal to 255. For color images, RGB images are converted to YCbCr color space and PSNR is calculated on luminance channel only due to sensitivity of human eye to luminance channel.

4.2.2 Structural Similarity Index (SSIM)

The Structural Similarity Index (SSIM) quality assessment index is based on the computation of three terms, namely the luminance term, the contrast term and the structural term. The overall index is calculated by multiplying all three terms. For $M \times N$ dimensional images I_1 and I_2 , SSIM is described as follows:

$$SSIM(x, y) = [l(x, y)]^\alpha [c(x, y)]^\beta [s(x, y)]^\gamma \quad (4.3)$$

where $l(x, y)$, $c(x, y)$ and $s(x, y)$ are defined as follows:

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \quad (4.4)$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \quad (4.5)$$

$$s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \quad (4.6)$$

where $\mu_x, \mu_y, \sigma_x, \sigma_y$ and σ_{xy} are the local means, standard deviations, and cross-covariance for images x, y . If $\alpha = \beta = \gamma = 1$ (default values) and $C_3 = C_2/2$ (default selection of C_3) the index simplifies to:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (4.7)$$

4.3 Super-Resolution Experiments

In this section, experimental details of our super-resolution model are given. For measuring the performance of SR algorithm, peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) are used as mentioned in the previous section. In this section, our model is compared with other super-resolution models in PSNR and SSIM measurements which are mentioned in previous section.

4.3.1 Optimization Experiments

In this part, experiments on hyperparameters of network optimization and other sub-experiments which fortify the purpose of our model are provided. Initially, it is discussed how different hyperparameter selections affect the performance of the network architecture. In order to optimize the network, using grid search technique, extremely many network and hyperparameter setups are constructed and trained/tested. At the end, the best network architecture and hyperparameter selection are used as final network structure. In this part, results and learning

curves of a few experiments on selection of important hyperparameters are provided. Additionally, other sub-experiments are provided in order to support the importance of our model.

Learning Rate Experiments

After making a set of experiments, we determined a set of hyperparameters and network architecture design which give the best super-resolution performance. Then, by changing learning rate parameter, it is intended to observe the effect of the change of learning rate on the designed network. For three different learning rate parameters, experiments are completed and learning curves are plotted for training data which is shown in the Figure 4.5. Additionally, for validation data, we obtained very similar learning curves as train data. From learning curves, we observed that with highest learning rate, network converged very rapidly and as learning rate decreases, convergence time of the network increases. Interestingly, all networks with different learning curves are converged to same point and same loss function is obtained with each network (Even if it is not shown in the figure, we observed that with more epochs, yellow line also converged to same point).

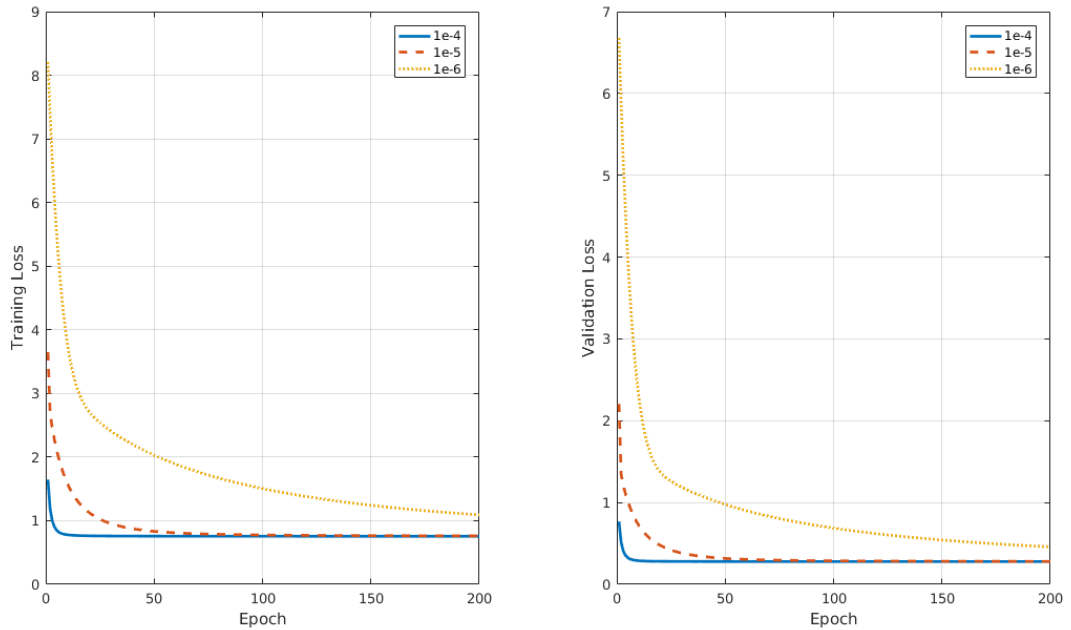


Figure 4.5: Learning Curve for Different Learning Rates

The Number of Hidden Units Experiments

Like learning curve experiments, we selected best hyperparameters and network architecture design which give the best super-resolution performance. Then, by changing the number of hidden units per layers, it is intended to observe the effect of the change of learning rate on the designed network. For three different number of hidden units parameters, experiments are completed and learning curves are plotted for training data which is shown in the Figure 4.6. From learning curves, we observed that even if the convergence time change as increasing number of hidden units, the network converged to same point like learning rate experiments. Additionally, with different number of hidden units combinations in intermediate layers, we obtained very similar results.

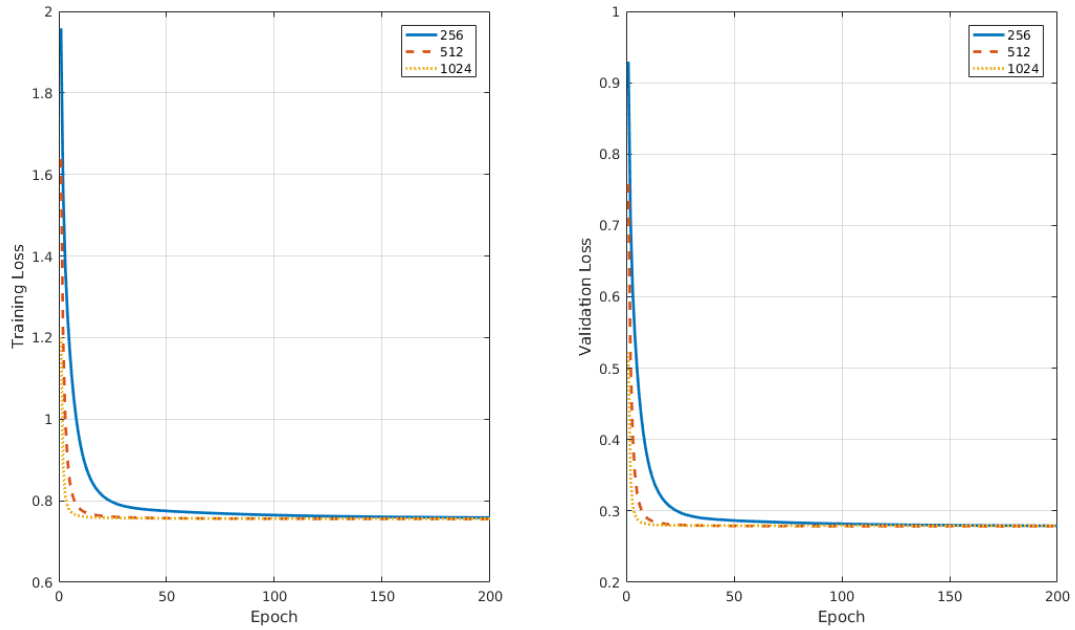


Figure 4.6: Learning Curve for Different Number of Hidden Units

Artifact Reduction Experiment

In order to measure the importance of artifact reduction module after super-resolution network, a set of experiments are made by using different combinations of these modules. In the first set of experiments, three different combinations which are only bicubic interpolation, bicubic interpolation followed by artifact reduction and bicubic interpolation followed by super-resolution and artifact reduction, are used. Respectively, 33.69 dB, 33.82 dB and 35.53 dB PSNR values are obtained using Set5 test dataset. In other words, even if artifact reduction network is a deep image enhancement technique, just using AR network increases the super-resolution performance only 0.13 dB. In the second set of experiments, similar experiment is made with SRCNN network as a super resolution module. Additionally, an ensemble of our network and SRCNN is used. In all these experiments, we obtained lower PSNR values than bicubic interpolation. In conclusion, all these experiments are an indication of consistency of our model.

Artifact Reduction JPEG Quality Experiment

In artifact reduction model, there are four different pre-trained models. Each network is trained with different image sets according to their JPEG compression quality ($Q = 10, 20, 30$ and 40). At the end of all four experiments, the network which gives highest PSNR value is used as a final model. Best PSNR value is obtained with $Q=40$ which is the highest JPEG quality.

4.3.2 Evaluation Results

In Table 4.1, PSNR and SSIM values of ten different super-resolution models are provided. Additionally, the Figure 4.7 indicates how average PSNR value which is favored by loss function is increased during the training. In experiments, four different test datasets and three scale factors ($x2$, $x3$, $x4$) are used. First of all, bicubic interpolation is the simplest super-resolution technique in the literature and lowest PSNR and SSIM results are obtained from its output images as expected. The measurements of bicubic interpolation is the lower bound for super-resolution problem. From Table 4.1, it would not be wrong to generalize

that spatial domain super-resolution models are far better models than frequency domain methods and deeper models are more successful than traditional and shallower ones. Additionally, increasing the dimension of images progressively gives relatively better results than direct reconstruction methods.

In comparing our model with other super-resolution architectures, it is observed that our super-resolution model is able to give closer results to other state-of-the-art models on SSIM metric rather than PSNR metric. Since in state-of-the-art models, the deep networks are trained for minimizing mean square error loss in spatial domain, this favors to obtain high PSNR results. However, since our model is trained in spectral domain, intended high PSNR values are not obtained. Rather, better SSIM results are obtained. In other words, our model produces images closer to ground truth in statistical manner, not pixel-wise manner.

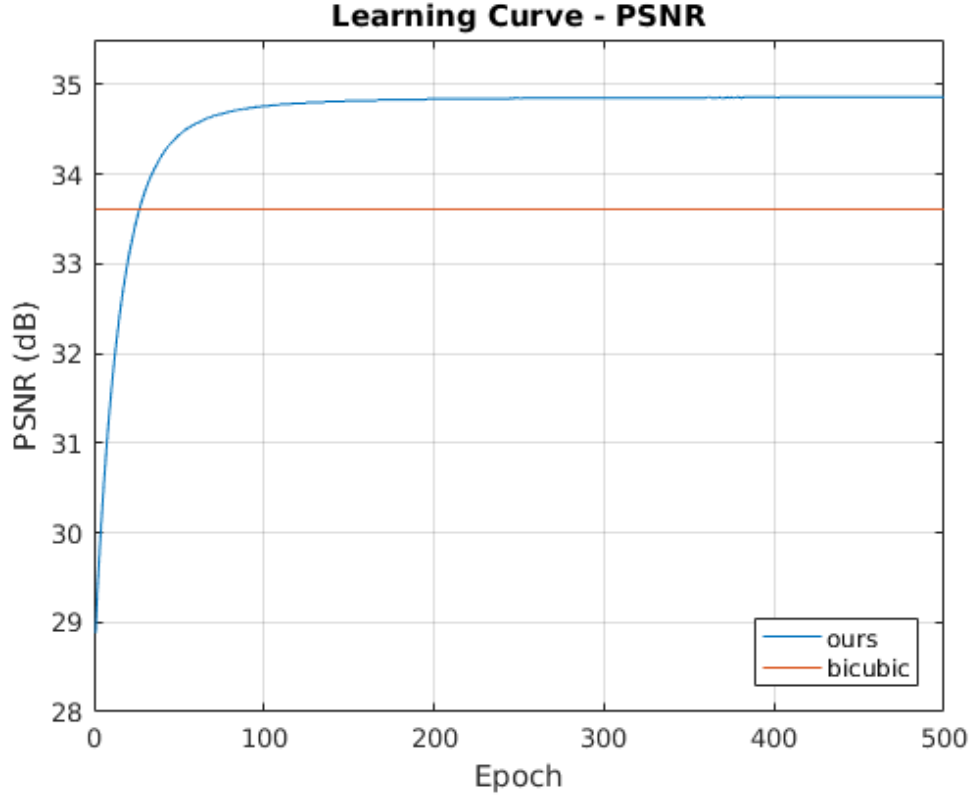


Figure 4.7: Learning Curve for PSNR Metric

Table 4.1: Quantitative Evaluation of SR Models (PSNR - SSIM)

Algorithm	Scale	Set5	Set14	BSDS100	Urban100
Bicubic	2	33.69 - 0.931	30.25 - 0.870	29.57 - 0.844	26.89 - 0.841
FourierSR	2	35.20 - 0.943	31.40 - 0.895	30.58 - 0.877	-
<i>Ours</i>	2	35.53 - 0.953	31.64 - 0.904	30.64 - 0.884	28.15 - 0.882
A+	2	36.60 - 0.955	32.32 - 0.906	31.24 - 0.887	29.25 - 0.895
RFL	2	36.59 - 0.954	32.29 - 0.905	31.18 - 0.885	29.14 - 0.891
SelfExSR	2	36.60 - 0.955	32.24 - 0.904	31.20 - 0.887	29.55 - 0.898
SRCNN	2	36.72 - 0.955	32.51 - 0.908	31.38 - 0.889	29.53 - 0.896
FSRCNN	2	37.05 - 0.956	32.66 - 0.909	31.53 - 0.892	29.88 - 0.902
VDSR	2	37.53 - 0.959	33.05 - 0.913	31.90 - 0.896	30.77 - 0.914
LapSRN	2	37.52 - 0.959	33.08 - 0.913	31.80 - 0.895	30.41 - 0.910
Bicubic	3	30.41 - 0.869	27.55 - 0.775	27.22 - 0.741	24.47 - 0.737
FourierSR	3	31.42 - 0.883	28.32 - 0.802	27.79 - 0.772	-
<i>Ours</i>	3	31.44 - 0.906	28.41 - 0.828	27.78 - 0.788	24.78 - 0.781
A+	3	32.62 - 0.909	29.15 - 0.820	28.31 - 0.785	26.05 - 0.799
RFL	3	32.47 - 0.906	29.07 - 0.818	28.23 - 0.782	25.88 - 0.792
SelfExSR	3	32.66 - 0.910	29.18 - 0.821	28.30 - 0.786	26.45 - 0.810
SRCNN	3	32.78 - 0.909	29.32 - 0.823	28.42 - 0.788	26.25 - 0.801
FSRCNN	3	33.18 - 0.914	29.37 - 0.824	28.53 - 0.791	26.43 - 0.808
VDSR	3	33.67 - 0.921	29.78 - 0.832	28.83 - 0.799	27.14 - 0.829
LapSRN	3	33.82 - 0.922	29.87 - 0.832	28.82 - 0.798	27.07 - 0.828
Bicubic	4	28.43 - 0.811	26.01 - 0.704	25.97 - 0.670	23.15 - 0.660
FourierSR	4	29.35 - 0.827	26.62 - 0.727	26.42 - 0.696	-
<i>Ours</i>	4	29.21 - 0.852	26.55 - 0.755	26.33 - 0.721	23.42 - 0.701
A+	4	30.32 - 0.860	27.34 - 0.751	26.83 - 0.711	24.34 - 0.721
RFL	4	30.17 - 0.855	27.24 - 0.747	26.76 - 0.708	24.20 - 0.712
SelfExSR	4	30.34 - 0.862	27.41 - 0.753	26.84 - 0.713	24.83 - 0.740
SRCNN	4	30.50 - 0.863	27.52 - 0.753	26.91 - 0.712	24.53 - 0.725
FSRCNN	4	30.72 - 0.866	27.61 - 0.755	26.98 - 0.715	24.62 - 0.728
VDSR	4	31.35 - 0.883	28.02 - 0.768	27.29 - 0.726	25.18 - 0.754
LapSRN	4	31.54 - 0.885	28.19 - 0.772	27.32 - 0.727	25.21 - 0.756

In this section, for our super-resolution system, we made same mean square analysis in spectral domain which we made in chapter 3. In the Figure 4.8, 4.9, 4.10 and 4.11, analysis results for bicubic interpolation, our model, SRCNN and LapSRN are given for comparison. First of all, as we discussed in the previous chapter, bicubic interpolation is problematic in mid-frequency regions. SRCNN and LapSRN models are able to recover the region between low and middle frequencies. However, there is still some problems on mid-frequency regions. On the other hand, our model is able to recover our selected problematic region considerably. However, there are some problems in lower frequencies of this problematic region. Moreover, for all methods, middle frequencies are still problematic and it is hard to recover this frequency bands.

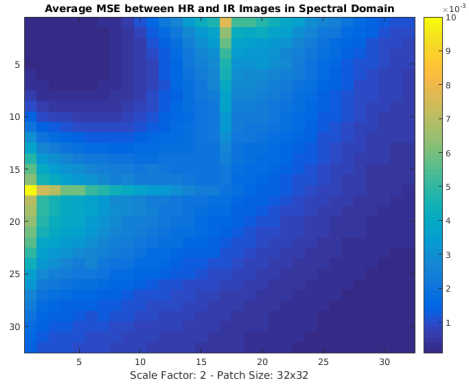


Figure 4.8: Bicubic Interpolation

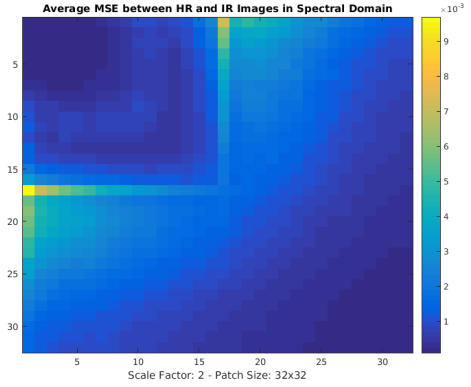


Figure 4.9: Ours

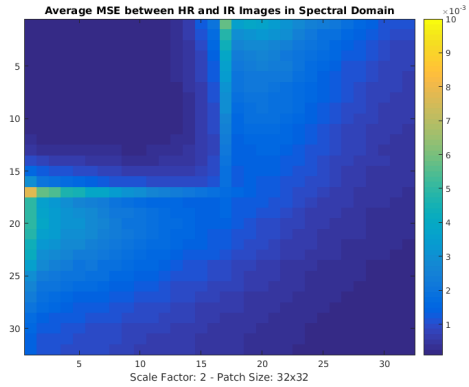


Figure 4.10: SRCNN

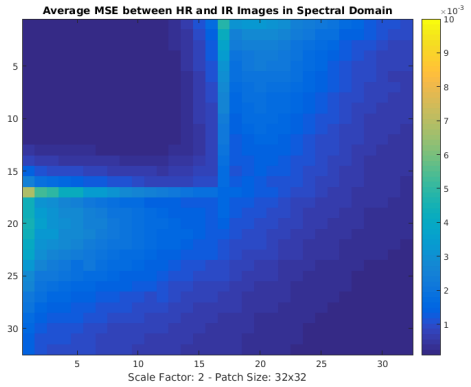


Figure 4.11: LapSRN

Table 4.2: Quantitative Evaluation for Different Setup

Setup	Scale	Set5		Set14		BSDS100	
SR+AR	2	35.53	0.953	31.64	0.904	30.64	0.884
SR	2	34.72	0.941	31.15	0.903	30.37	0.889
SR-Full Patch	2	27.68	0.893	25.98	0.865	24.98	0.862
Bicubic	2	33.69	0.931	30.25	0.870	29.57	0.844
SR+AR	3	31.44	0.906	28.41	0.828	27.78	0.788
SR	3	31.08	0.901	28.17	0.826	27.66	0.801
SR-Full Patch	3	24.37	0.889	21.46	0.755	21.65	0.735
Bicubic	3	30.41	0.869	27.55	0.775	27.22	0.741
SR+AR	4	29.21	0.852	26.55	0.755	26.33	0.721
SR	4	28.93	0.846	26.39	0.753	26.19	0.719
SR-Full Patch	4	22.13	0.835	21.32	0.708	20.88	0.658
Bicubic	4	28.43	0.811	26.01	0.704	25.97	0.670

In addition to quantitative evaluation of SR models in Table 4.1, the results in extra experiments for intermediate model setups are shared in Table 4.2. In this table, results for bicubic interpolation, our complete model (SR+AR), our super-resolution model (SR) which learns the super-resolution mapping only for 512-dimensional problematic frequency region and our super-resolution model (SR - Full patch) which learns the super-resolution mapping for all 1024-dimensional frequency regions. The aim of this experiments for intermediate steps is explaining the reason why we do not use all frequencies in super-resolution mapping and use only problematic frequencies in spectral domain. Additionally, in order to measure the impact of artifact reduction module after our super-resolution mapping, these intermediate experiments are done. From Table 4.2, using all frequencies in super-resolution mapping completely fails down bicubic interpolation images. When, low frequencies in spectral mapping are changed in the super-resolution network and since, there is not any component to balance the offset change, using all frequency values in SR mapping effect the mean value of complete image. For this reason, there is a huge gap between bicubic interpolation and full patch super-resolution results. Therefore, using only the problematic region and omitting low-frequencies provides us to learn without changing mean value of the image. Additionally, high-frequencies are omitted due to decrease the number of parameters used in the network. For artifact reduction module, we

observed that using artifact reduction module provide us a considerable amount of increase in our complete SR system. While artifact reduction module decreases the ringing artifacts, it increases the resolution of images. To conclude, Table 4.2 explain the reason of using only problematic frequencies and the impact of the artifact reduction module.

4.3.3 Output Images

In this section, sample output images of our super-resolution model are provided. Additionally, in order to compare our model, output images of other state-of-the-art models and high-resolution ground truth images are given. In Appendix A, we also provide additional resulting images for scale factors of x2, x3 and x4.

In overall, output images indicate that our model gives much better results than bicubic interpolation both visually and mathematically. The remaining super-resolution models give relatively closer results to each other. Additionally, even if PSNR value of a specific model is higher than ours, in some images, our models give visually more satisfying results.

In Figure 4.12, even if different PSNR values are obtained from every model, output images are visually very close to each other. Additionally, perceptually, the output of our model gives better result. Similar to Figure 4.12, same comments are also valid for from Figure 4.14 to Figure 4.22. In the ground truth image in Figure 4.13, it is easy to observe patch boundaries in the images. In addition to this original patches, our model also divides the image into different patches and apply discrete cosine transform. Since, it is hard to deal with patch boundaries in frequency domain, our model gives relatively lower PSNR values and visually low quality images.



Figure 4.12: Output Images of Different SR Models - Set5 1 (x2)



Figure 4.13: Output Images of Different SR Models - Set5 2 (x2)



Figure 4.14: Output Images of Different SR Models - Set5 3 (x2)

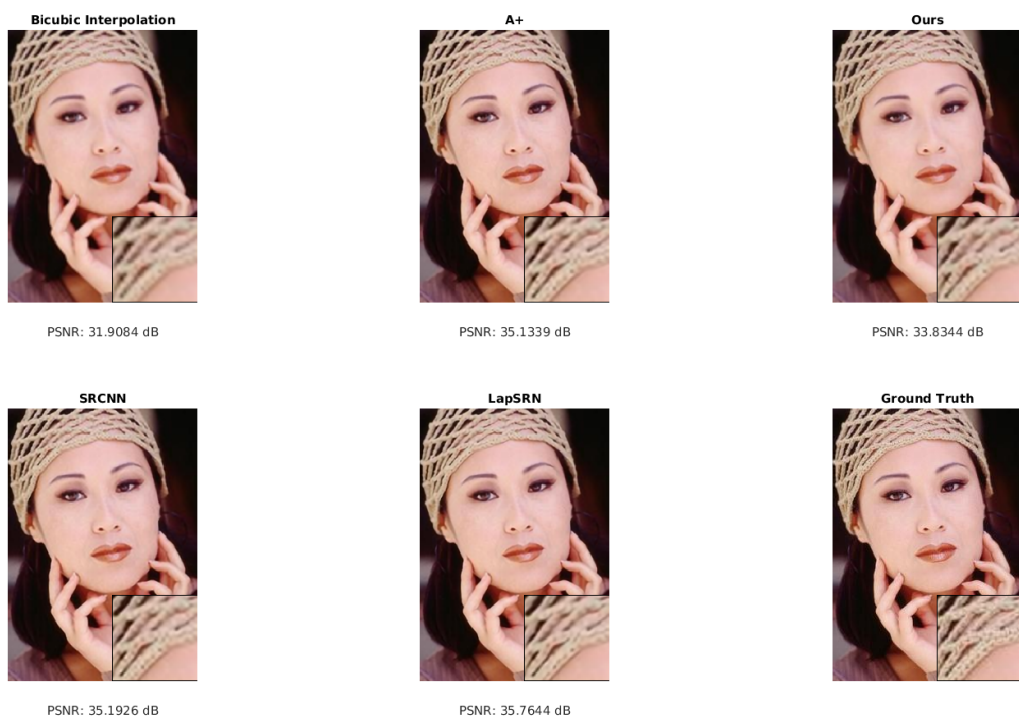


Figure 4.15: Output Images of Different SR Models - Set5 5 (x2)



Figure 4.16: Output Images of Different SR Models - Set14 1 (x2)



Figure 4.17: Output Images of Different SR Models - Set14 2 (x2)

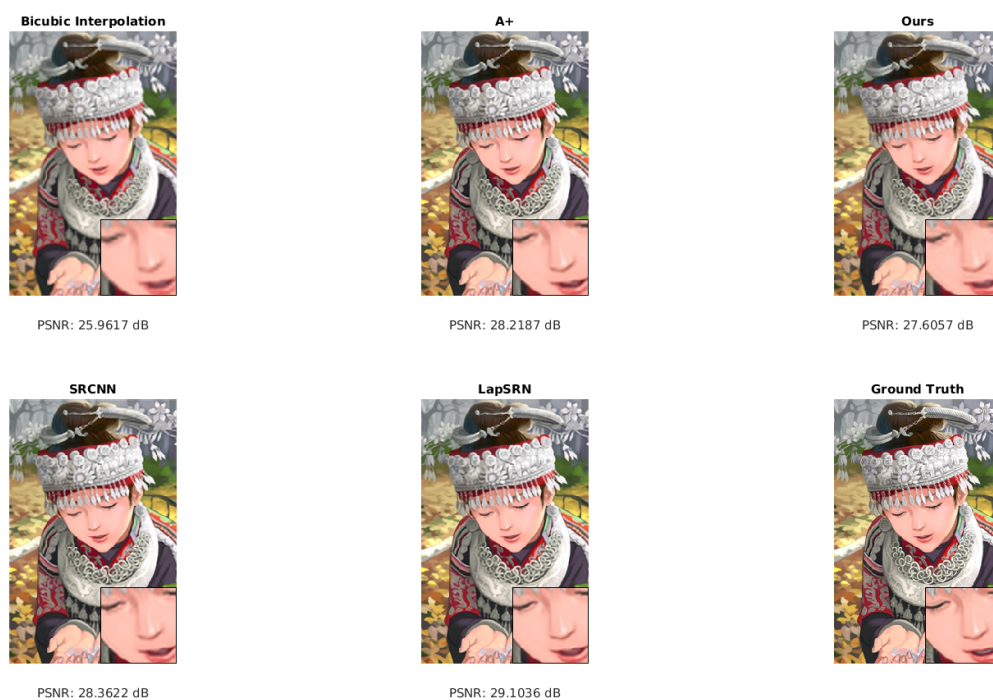


Figure 4.18: Output Images of Different SR Models - Set14 5 (x2)

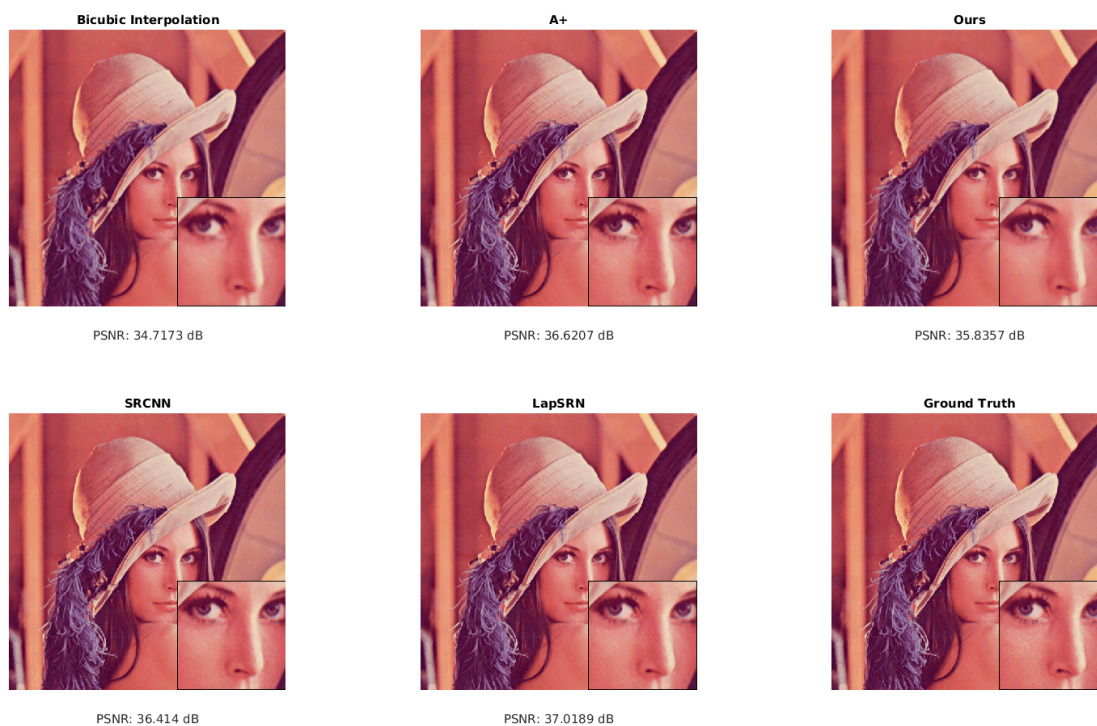


Figure 4.19: Output Images of Different SR Models - Set14 9 (x2)

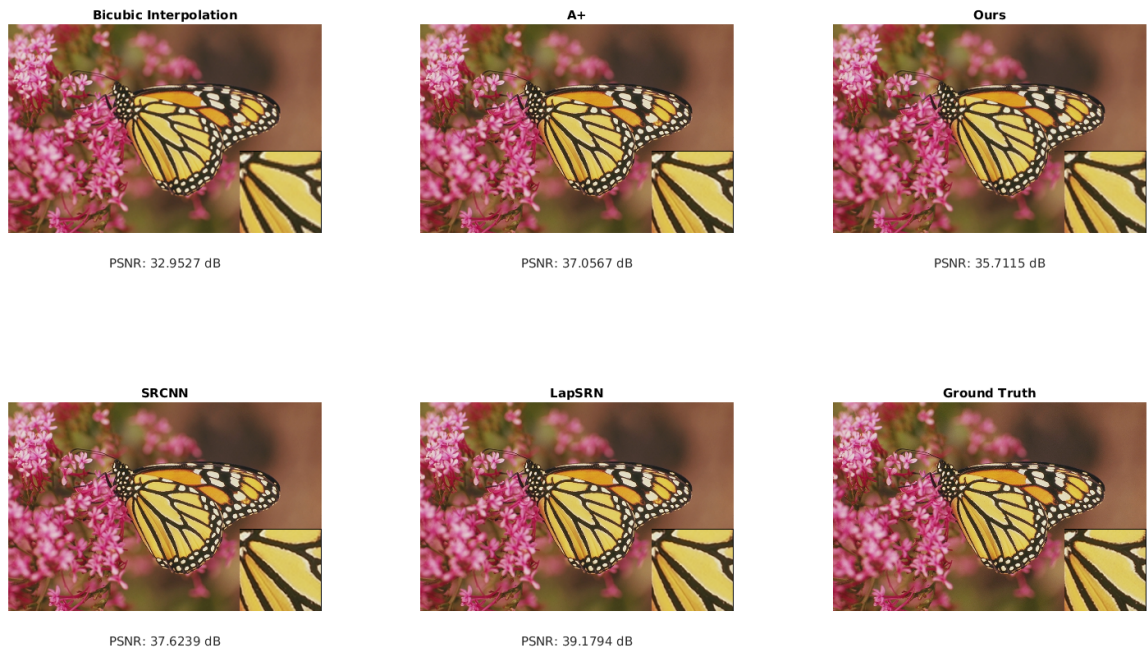


Figure 4.20: Output Images of Different SR Models - Set14 11 (x2)

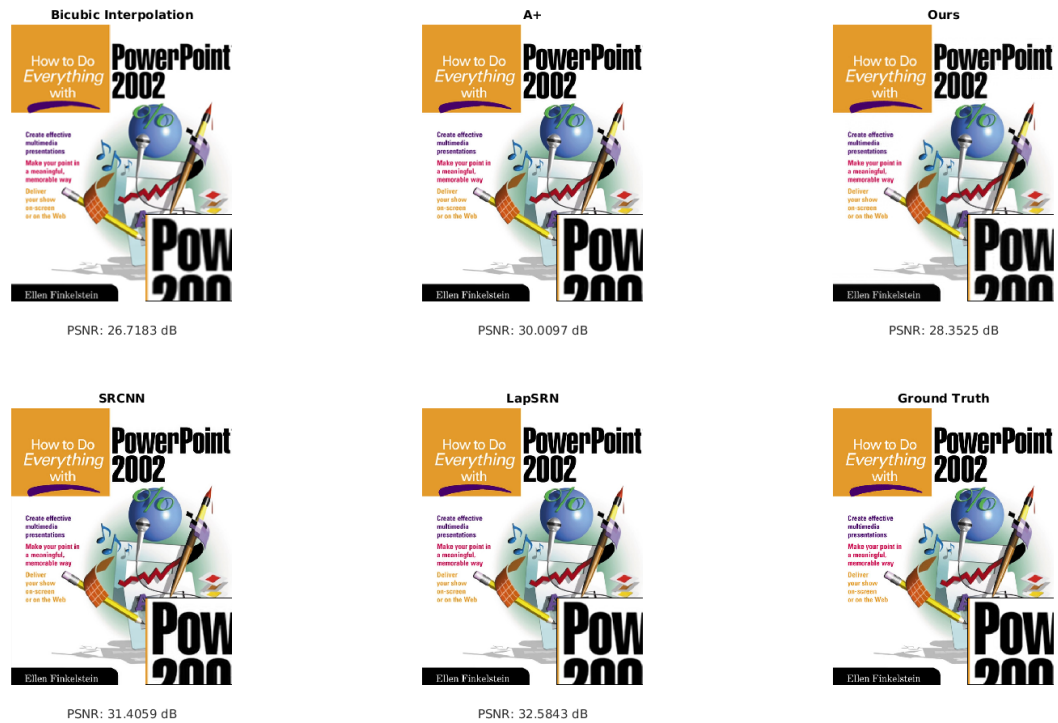


Figure 4.21: Output Images of Different SR Models - Set14 13 (x2)

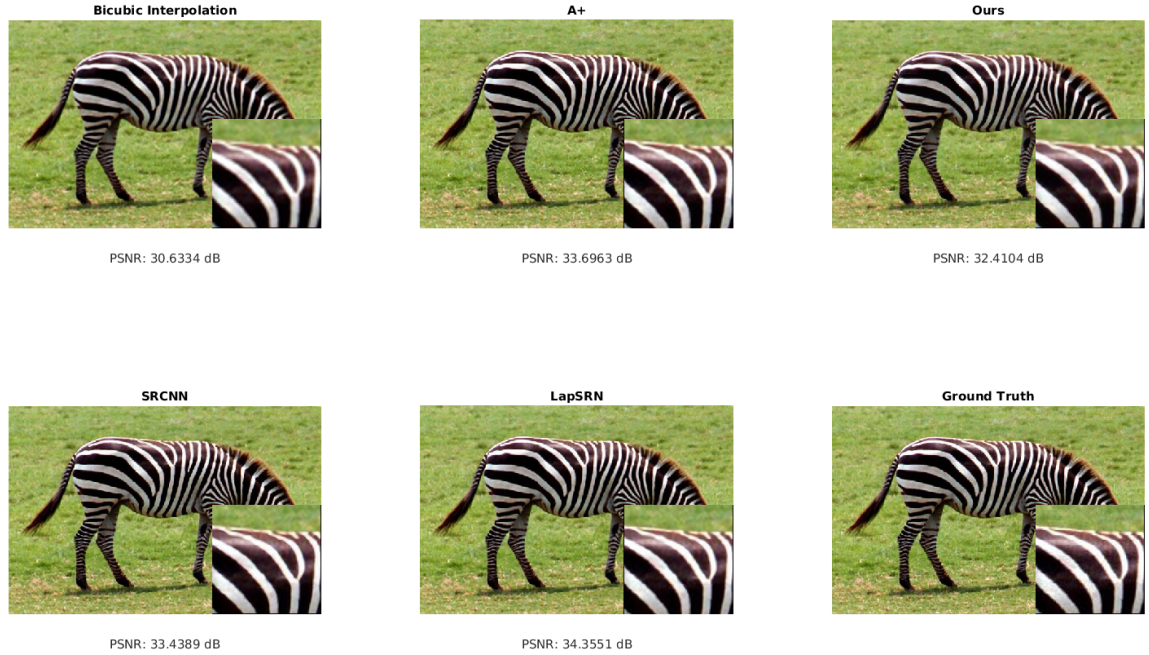


Figure 4.22: Output Images of Different SR Models - Set14 14 (x2)

To sum up, we observed that we obtained results comparable to the state-of-the-art methods on benchmark datasets, especially using SSIM metric. Qualitatively, we observe little difference between our results and other state-of-the-art methods. These results show the applicability of spectral domain deep learning approaches to super-resolution problem.

Chapter 5

Conclusions

In this chapter, final concluding remarks and possible future directions are discussed. In conclusion part, the advantages and disadvantages of training neural networks in the spectral domain are discussed and some possible improvements are given. Since, in the literature, there are only a few works that combine deep learning and spectral domain methods, it might be helpful to use these future directions.

5.1 Concluding Remarks

In this thesis, we have developed a complete single-image super-resolution learning system. Our SR system consists of two phases such that the first model learns a super-resolution mapping from LR images to HR images completely in the spectral domain and the second model tries to remove ringing artifact effects which are caused from spectral transformation used in the first phase. In super-resolution model, even if we could not obtain state-of-the-art results, training a single model for all scale factors which is provided by spectral transforms is quite a novel idea. For artifact reduction problem, we transfer knowledge from one

domain to another without any training. Additionally, training an artifact reduction model specific to our problem from scratch might increase the performance in a considerable amount of order. In overall, from our super-resolution system, visually very closer results to other state-of-the-art models are obtained. Additionally, rather than higher PSNR value which favors to MSE in spatial domain, more competitive results are obtained according to SSIM metric.

5.2 Future Work

In this section, we share our learning outcomes which we gained after writing this thesis, in order to give insights into possible future work for super-resolution problem and overall deep learning literature.

First of all, for super-resolution problem, there are various advantages and disadvantages of both spatial and spectral domain. For this reason, combining spatial domain and spectral domain and using advantages of both domains would increase the overall performance.

Secondly, with a better prediction model in the spectral domain, using one single network for all scale factors might be very effective for super-resolution problem. For instance, in the pre-processing step, doing bicubic interpolation might make the prediction process harder. Therefore, developing a completely different model without bicubic interpolation might be better to obtain more accurate results.

Thirdly, we observed that it is a bit harder to train a neural network in spectral domain. Most probably, this is because when you change a value in the spectral domain, it affects whole spatial domain and it becomes harder to find the global minimum point for necessary recovery performance. However, if it is understood by developing new paradigms such as an optimizer, a loss function or a better learning architecture in spectral domain, it might be revolutionary for whole deep learning literature.

Fourthly, In the literature, there are many solutions for image completion problem such as PixelRNN [41] and PixelCNN [42]. Using similar architectures, super-resolution problem would be solved in spectral domain as well. In other words, the spectral domain would allow us to convert super-resolution problem to image completion problem.

Finally, new approaches might be developed by training deep learning models in spectral domain for other computer vision problems.

Bibliography

- [1] C.-Y. Yang and M.-H. Yang, “Fast direct super-resolution by simple functions,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 561–568, 2013.
- [2] R. Timofte, V. De Smet, and L. Van Gool, “A+: Adjusted anchored neighborhood regression for fast super-resolution,” in *Asian Conference on Computer Vision*, pp. 111–126, Springer, 2014.
- [3] J. Yang, J. Wright, T. Huang, and Y. Ma, “Image super-resolution as sparse representation of raw image patches,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, IEEE, 2008.
- [4] 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, pp. 2861–2873, 2010.
- [5] S. Schuler, C. Leistner, and H. Bischof, “Fast and accurate image upscaling with super-resolution forests,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3791–3799, 2015.
- [6] 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, 2016.
- [7] J. Kim, J. Kwon Lee, and K. Mu Lee, “Accurate image super-resolution using very deep convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1646–1654, 2016.

- [8] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, “Deep laplacian pyramid networks for fast and accurate super-resolution,” *arXiv preprint arXiv:1704.03915*, 2017.
- [9] O. Rippel, J. Snoek, and R. P. Adams, “Spectral representations for convolutional neural networks,” in *Advances in Neural Information Processing Systems*, pp. 2449–2457, 2015.
- [10] Y. Wang, C. Xu, S. You, D. Tao, and C. Xu, “Cnnpack: Packing convolutional neural networks in the frequency domain,” in *Advances in neural information processing systems*, pp. 253–261, 2016.
- [11] N. Kumar, R. Verma, and A. Sethi, “Convolutional neural networks for wavelet domain super resolution,” *Pattern Recognition Letters*, vol. 90, pp. 65–71, 2017.
- [12] J. Li, S. You, and A. Robles-Kelly, “A frequency domain neural network for fast image super-resolution,” *arXiv preprint arXiv:1712.03037*, 2017.
- [13] S. Russell, P. Norvig, and A. Intelligence, “A modern approach,” *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, vol. 25, p. 27, 1995.
- [14] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing atari with deep reinforcement learning,” *arXiv preprint arXiv:1312.5602*, 2013.
- [15] E. Tzeng, C. Devin, J. Hoffman, C. Finn, X. Peng, S. Levine, K. Saenko, and T. Darrell, “Towards adapting deep visuomotor representations from simulated to real environments,” *arXiv preprint arXiv:1511.07111*, 2015.
- [16] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
- [18] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

- [19] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (elus),” *arXiv preprint arXiv:1511.07289*, 2015.
- [20] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, “Handwritten digit recognition with a back-propagation network,” in *Advances in neural information processing systems*, pp. 396–404, 1990.
- [21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [22] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, pp. 1097–1105, 2012.
- [24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.
- [25] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [26] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [27] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *European conference on computer vision*, pp. 818–833, Springer, 2014.

- [28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, pp. 2672–2680, 2014.
- [29] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv preprint arXiv:1511.06434*, 2015.
- [30] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” in *Advances in neural information processing systems*, pp. 3320–3328, 2014.
- [31] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1717–1724, 2014.
- [32] J.-B. Huang, A. Singh, and N. Ahuja, “Single image super-resolution from transformed self-exemplars,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5197–5206, 2015.
- [33] C. Dong, C. C. Loy, and X. Tang, “Accelerating the super-resolution convolutional neural network,” in *European Conference on Computer Vision*, pp. 391–407, Springer, 2016.
- [34] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” *arXiv preprint arXiv:1609.04802*, 2016.
- [35] A. V. Oppenheim, *Discrete-time signal processing*. Pearson Education India, 1999.
- [36] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

- [37] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting.,” *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [38] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International Conference on Machine Learning*, pp. 448–456, 2015.
- [39] C. Dong, Y. Deng, C. Change Loy, and X. Tang, “Compression artifacts reduction by a deep convolutional network,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 576–584, 2015.
- [40] P. Svoboda, M. Hradis, D. Barina, and P. Zemcik, “Compression artifacts removal using convolutional neural networks,” *arXiv preprint arXiv:1605.00366*, 2016.
- [41] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, “Pixel recurrent neural networks,” *arXiv preprint arXiv:1601.06759*, 2016.
- [42] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, “Conditional image generation with pixelcnn decoders,” *arXiv preprint arXiv:1606.05328*, 2016.

Appendix A

Output Images

In this appendix, we provide additional example outputs on benchmark datasets using our method and other state-of-the-art methods.

Figure A.1, A.5, A.9 shows results on baby image of Set5 dataset for scale factors x2, x3 and x4 respectively.

Figure A.2, A.6, A.10 shows results on baboon image of Set14 dataset for scale factors x2, x3 and x4 respectively.

Figure A.3, A.7, A.11 shows results on barbara image of Set14 dataset for scale factors x2, x3 and x4 respectively.

Figure A.4, A.8, A.12 shows results on monarch image of Set14 dataset for scale factors x2, x3 and x4 respectively.



Figure A.1: Output Images of Different SR Models - Set5 1 (x2)



Figure A.2: Output Images of Different SR Models - Set14 1 (x2)



Figure A.3: Output Images of Different SR Models - Set14 2 (x2)



Figure A.4: Output Images of Different SR Models - Set14 11 (x2)



Figure A.5: Output Images of Different SR Models - Set5 1 (x3)



Figure A.6: Output Images of Different SR Models - Set14 1 (x3)

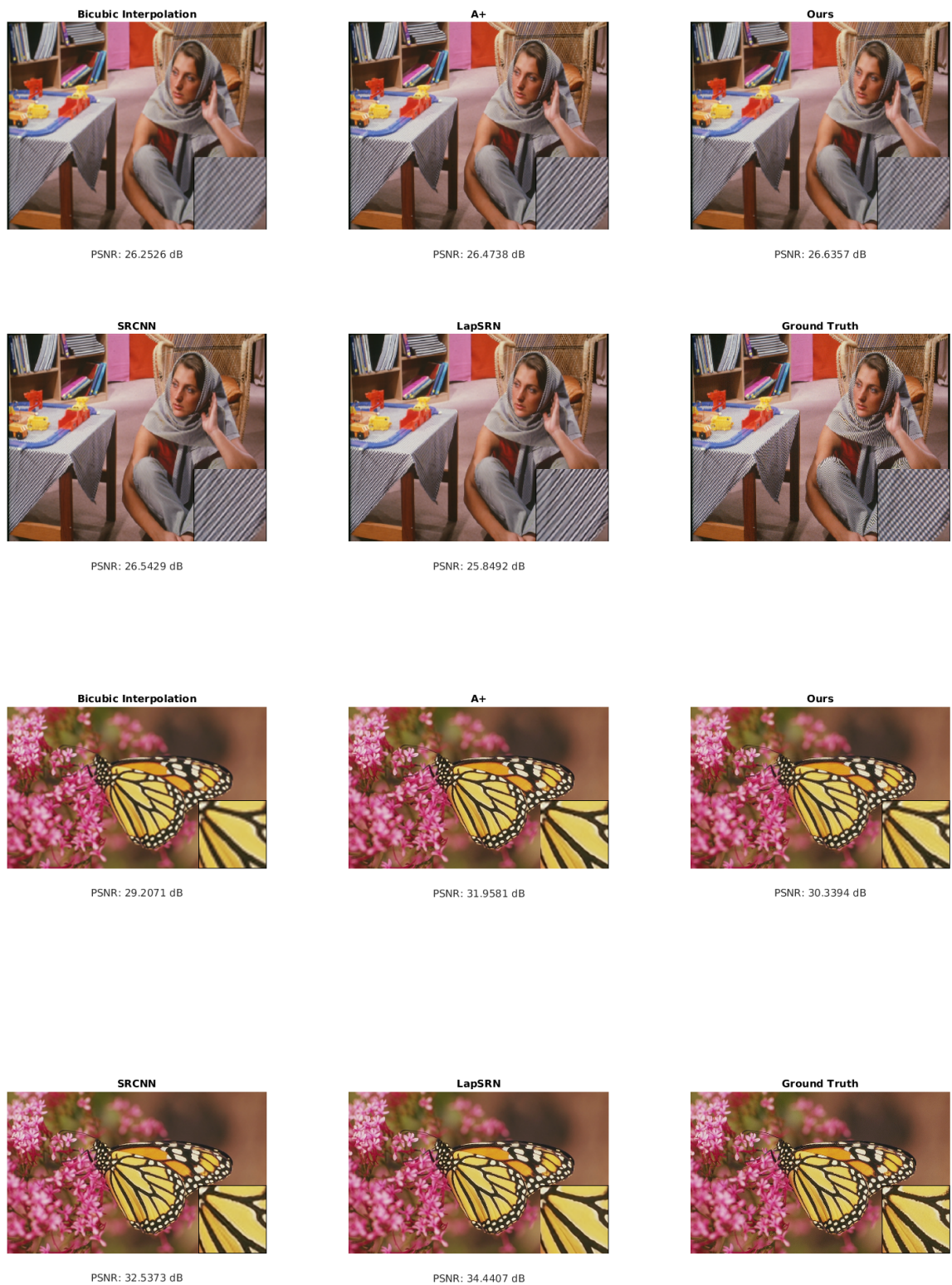


Figure A.8: Output Images of Different SR Models - Set14 11 (x3)



Figure A.9: Output Images of Different SR Models - Set5 1 (x4)



Figure A.10: Output Images of Different SR Models - Set14 1 (x4)



Figure A.11: Output Images of Different SR Models - Set14 2 (x4)

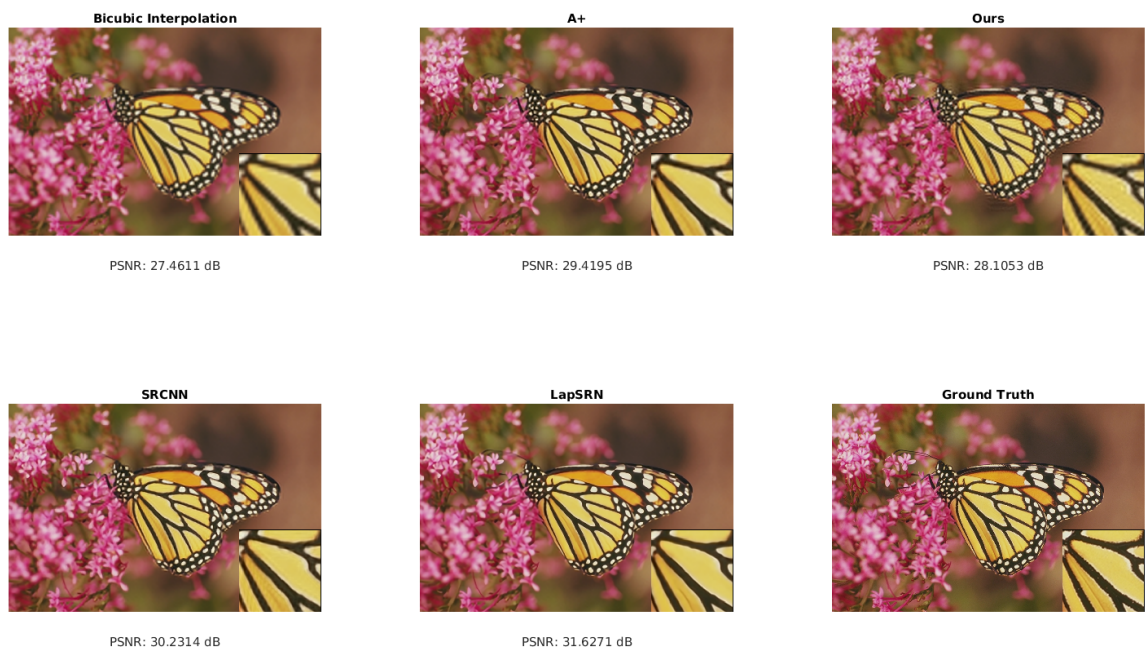


Figure A.12: Output Images of Different SR Models - Set14 11 (x4)