이학박사 학위논문

Image Quality Enhancement using Deep Neural Networks

(심층 신경망을 활용한 영상 품질 강화 기법)

2021년 8월

서울대학교 대학원

협동과정 계산과학전공

노 형 민

Image Quality Enhancement using Deep Neural Networks

(심층 신경망을 활용한 영상 품질 강화 기법)

지도교수 강 명 주

이 논문을 이학박사 학위논문으로 제출함

2021년 4월

서울대학교 대학원

협동과정 계산과학전공

노 형 민

노 형 민의 이학박사 학위논문을 인준함

2021년 6월

위 원 장 _____

부 위 원 장 _____

위 원 _____

위 원 _____

위 원 _____

Image Quality Enhancement using Deep Neural Networks

A dissertation
submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
to the faculty of the Graduate School of
Seoul National University

by

Hyungmin Roh

Dissertation Director : Professor Myungjoo Kang

Interdisciplinary Program of
Computational Science and Technology
Seoul National University

August 2021

Abstract

In this thesis, we focus on deep learning methods to enhance the quality of a single image. We first categorize the image quality enhancement problem into three tasks: denoising, deblurring, and super-resolution, then introduce deep learning techniques optimized for each problem. To solve these problems, we introduce a novel deep neural network suitable for multi-scale analysis and propose efficient model-agnostic methods that help the network extract information from high-frequency domains to reconstruct clearer images. Experiments on SIDD, Flickr2K, DIV2K, and REDS datasets show that our method achieves state-of-the-art performance on each task. Furthermore, we show that our model can overcome the over-smoothing problem commonly observed in existing PSNR-oriented methods and generate more natural high-resolution images by applying adversarial training.

Key words: Image Deblurring, Image Denoising, Single Image Super-Resolution, Image Enhancement, Deep Learning, Convolutional Neural Network
Student Number: 2015-22685

Contents

List of Figures

LIST OF FIGURES

List of Tables

Chapter 1

Introduction

With the recent development of display technology, high-resolution display devices such as 4K or 8K have become common, and demand for high-resolution images has increased. The most effective way to obtain such high-resolution images would be using filming devices with high-end quality. However, this is usually not a feasible option, mainly due to economic problems. Besides the difficulty of equipping such high-end devices, there remain some challenging problems such as ultra-zooming tasks or restoring historical images where low-resolution images that should be converted to high-resolution ones are already taken. There are so many images or videos in SD, HD, or FHD resolution taken years ago that we need to convert into 4K or 8K to adapt to modern display devices. Therefore, the need for studies on analyzing low-resolution images' characteristics and enhancing their quality is increasing day by day.

Researches on the image quality enhancement problem are often divided into image denoising, image deblurring, and super-resolution problems. For decades, most research has focused on statistical model-based methods [6] such as Maximum A Posteriori Estimation (MAP) and Expectation Maximization algorithms (EM). However, with the recent development of deep learning techniques along with GPU devices, the high performance and potential of learning-based methods have drawn the attention of researchers, and many studies have been proposed accordingly.

Most learning-based methods utilize the high capacity of deep neural networks with remarkable ability to understand the content and style of the image they have shown in visual recognition, including image classification and object detection. Using these high capacities and analytic powers of deep neural networks, learning-based methods have been successfully adapted to the field of image enhancement and have shown better performances compared to traditional model-based methods in laboratory environments.

When applied to real-world problems, however, most learning-based methods have failed to produce such good results while model-based methods are more flexible and applicable to low-resolution images with various kinds of blur and noises. This is because learning-based methods learn how to enhance the quality of images only by analyzing relations between given pairs of low-resolution images and their corresponding high-resolution ones in the training phase. However, in real-world problems, only low-resolution images are given and their high-resolution pairs are unknown. This means that the models have to infer new relations that they have never learned, which often leads to huge performance degradation when they solve real-world problems.

Another problem called the "ill-posed problem" also makes solving real-world problems more challenging; there are countless high-resolution image candidates in solution spaces corresponding to a given low-resolution image, while the number of high-resolution outputs human viewers perceive natural is very small or unique. The ill-posed problem makes it very difficult for deep neural networks to derive natural high-resolution outputs when solving real-world problems. Research on mathematical ways to reduce the solution spaces in unsupervised environments has been recently proposed to deal with the problem.

Figure 1.1 shows the achievements of recent learning-based studies on Single Image Super-Resolution. As Figure 1.1 illustrates, many studies on the architectural design of deep neural networks have been proposed over the years and have shown great performances. However, they have recently reached the limit; little progress has been made except for marginal im-

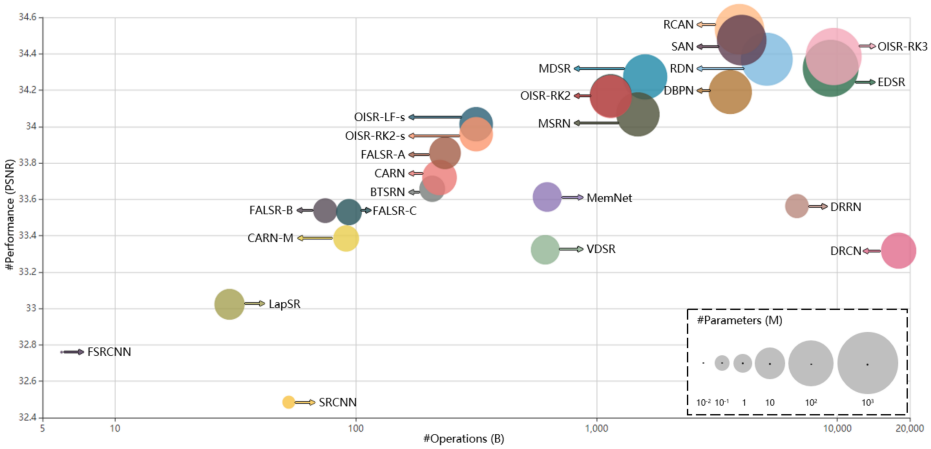


Figure 1.1: Performances of recent super-resolution methods [36]. The x-axis and the y-axis denote the number of operations and PSNR performances, respectively, while the size of circle represents the number of parameters of each networks.

provements on performances. This is because deep neural networks are originally optimized for understanding the content of images based on the high capacity of deeply stacked layers, so they are less capable of interpreting and restoring detailed information of corrupted images. Accordingly, recent studies are more focused on conveying mathematical properties of images to the existing models rather than designing deeper networks.

In keeping with this trend, we not only propose novel architectures of deep neural network for image enhancement problems but also introduce some state-of-the-art model-agnostic methods to make networks capable of producing sharper and more realistic images by providing abstract characteristics and high-frequency components of images with a little modification in the structure of existing models.

Chapter 2

Preliminaries

Deep learning techniques have shown remarkable flexibility that they can be applied to various vision tasks such as classifying or localizing objects by analyzing feature maps extracted from convolutional layers, as well as the ability to transferring new styles to images or synthesizing objects. Of course, such image analysis capability and the ability to generate and synthesize natural images are also beneficial for image quality enhancements such as denoising, deblurring, and super-resolutions. In recent years, learning-based methods have demonstrated better performances and proved to be more effective than traditional model-based methods.

2.1 Image Denoising

2.1.1 Problem Formulation: AWGN

In the image enhancement problem, the relation between the low-resolution image and the corresponding high-resolution image is often expressed as follows:

$$I_{LR} = (k * I_{HR}) \downarrow_s + n \tag{2.1.1}$$

where k is the blur kernel, s is the scale factor, n is the additive noise, and I_{LR} and I_{HR} are low-resolution image and its corresponding high-resolution image, respectively.

The objective of the image denoising problem is to eliminate noise n from Equation (2.1.1). To focus on removing n, most studies assume that k and s are identity mapping. This makes denoising easier than deblurring or super-resolution as the scale factor $s = 1$ makes input and output images having the same size, and the damage to the pixels of each location is relatively not severe as k is the identity kernel.

To successfully detach noise n from images, we need to know what kind of noise has been added to the image. Most denoising studies assume that additive white Gaussian noise, which is signal-independent, is given. However, in a low-light condition, the image could have different kinds of signal-dependent noises, such as Poisson-Gaussian noise.

Additive White Gaussian Noise. Most studies assume the additive white gaussian noise (AWGN) which can be expressed as follows:

$$I_{LR}(x, y) = I_{HR}(x, y) + n(x, y) \tag{2.1.2}$$

where noise n is independent and identically distributed and follows zero-mean Gaussian distribution.

Multiplicative White Gaussian Noise. The most common signal-dependent noise is multiplicative white Gaussian noise or speckle noise that can be expressed as follows:

$$I_{LR}(x, y) = I_{HR}(x, y) + I_{HR}(x, y) \odot n(x, y) \tag{2.1.3}$$

where $\odot$ is element-wise multiplication. Here, n follows a normal distribution, as in the case of AWGN, but the magnitude of the noise added to the image is proportional to the pixel intensity.

Poisson-Gaussian Noise. The Poisson-Gaussian noise is given by the sum of the signal-dependent Poisson noise and the signal-independent

Gaussian noise, which can be expressed as follows:

$$I_{LR}(x, y) = I_{HR}(x, y) + n_p\left(I_{HR}(x, y)\right) + n_g(x, y) \tag{2.1.4}$$

where n_p and n_g denote the signal-dependent Poisson noise that is proportional to the image and the signal-independent Gaussian noise, respectively.

This thesis focuses on removing Additive White Gaussian Noise from images as with most existing studies.

2.1.2 Existing Methods

The learning-based methods learn the characteristics of noisy images in the spatial domain to find hidden deep image prior and use them to reconstruct noise-free images. In 2017, Zhang et al. [40] proposed DnCNN with 17 or 20 convolutional layers with Batch Normalizations followed by ReLU activation functions. The output of DnCNN is added to the original noisy image to get the denoised image. That is, the model is trained to predict noise maps from input images and subtract them to get noise-free images. In 2018, Zhang et al. [41] proposed FFDNet, which takes a tunable noise level map as input and deals with spatially variant noises. Before forwarding through their model, they reshaped the input image to four downsampled sub-images and added tunable noise level maps. They put those noisy downsampled images into FFDNet together and outputs four denoised sub-images, which are finally used to reconstruct the output image. Park et al. [25] proposed DHDN, one of the state-of-the-art denoising methods in the NTIRE 2019 Challenge. Inspired by U-Net [27], which downsamples feature maps in the model instead of manually downsampling images, DHDN replaced convolutional layers with Densely Connected Residual Blocks (DCR Blocks) to introduce dense connectivity [11] and residual learning [10] to their model. These studies utilized neural network's ability to find deep image prior from the spatial domain. However, some studies claimed that spectral analysis is much more reliable and effective than spatial analysis.

In 2019, Zhao et al. [45] proposed WDnCNN, a discrete wavelet DnCNN which restores images from different parts of the frequency spectrum, arguing that removing noise in the frequency spectra is more efficient because noise mainly exhibits as high-frequency components. They also proposed Batch Normalization Module (BNM) to normalize highly imbalanced coefficients from different frequency spectra.

2.2 Image Deblurring

2.2.1 Problem Formulation: Blind Deblur

As mentioned in section 2.1.1, we assume that the relation of low-resolution images I_{LR} and corresponding high-resolution images I_{HR} can be expressed as following:

$$I_{LR} = (k * I_{HR}) \downarrow_s + n \qquad \text{(2.1.1 revisited)}$$

The research on the deblurring problem is mainly divided into two categories: the Non-blind method and the Blind method. The Non-blind method is to restore I_{HR} where blur kernel k is given. On the other hand, studies on the blind method assume that k is unknown, making the problem far more difficult to solve.

Most traditional model-based methods focused on restoring accurate I_{HR} by analyzing the mathematical properties of k in non-blind situations. However, the learning-based methods, which analyze images by observing various kinds of big data, focus more on studying end-to-end models that can reconstruct I_{HR} from different kinds of k with a single model in blind situations rather than solving constrained problems where k is given.

2.2.2 Existing Methods

This section introduces some remarkable learning-based deblurring studies, divided into three categories: Non-blind method, Blind-Method, and Blur Kernel Estimation.

Non-Blind Methods. Many studies have attempted to solve the Non-blind problem by combining recent deep learning techniques with traditional mathematical methods such as deconvolving images using Wiener filters [37]. In 2017, Kruse et al. [15] proposed a method to restore I_{HR} by combining the improved Wiener filter for given blur kernel k with feature maps extracted from the CNN model. To combine feature maps with Wiener filter, they proposed an FFT-based deconvolution which requires the circular blur assumption. In 2018, Wang et al. [35] introduced a general non-blind deconvolution method that can handle different types of k and different levels of n. They first deconvolve I_{LR} with regularized Wiener filter and then input them to the neural network to predict residual maps, which is a similar approach to USRNet that Zhang et al. [39] proposed in 2020. Zhang et al. iteratively input I_{LR} to deconvolution and USRNet to make I_{HR} estimation sharper. Though non-blind methods show great performances in their studies, they are difficult to apply to real-world problems because they cannot produce such good results where the blur kernel k is unknown.

Blind Methods. Recent works are more focused on the blind methods as they are more flexible and applicable to real-world problems. In 2017, Nah et al. [22] proposed DeepDeblur, which takes the Gaussian pyramid of downsampled blurry images as the input and outputs estimated latent image pyramid. The model could estimate sharp latent features from multi-scaled receptive fields by taking different scales of blurry images. Similarly, in 2018, Tao et al. [33] also adapted the coarse-to-fine scheme to their model. Instead of the Gaussian pyramid, they first input downsampled blurry images to their model and then upsample the output and iteratively input them to the network until the output images and target images have the same size. On the other hand, in 2018, Kupyn et al. [16] proposed DeblurGAN, which uses a conditional GAN framework and content loss. In 2019, Kupyn et al. [17] developed their model by applying FPN to the generator and global/local discriminators.

Blur Kernel Estimation. Although some studies have attempted to solve the deblurring problem without using any information about the blur kernel, many researchers attempted to improve the performance of blind methods by predicting the unknown blur kernel k from given I_{LR}. In 2019, Cornillère et al. [7] trained a kernel discriminator to analyze the output image to determine whether the appropriate kernel was used. By minimizing the error of the kernel discriminator, they could predict the suitable blur kernels that fit best to deblur given images. Bel-Klingler et al. [5] used patch GAN to generate natural downscaled fake patches from images. Once the generator is sufficiently trained, the network gives us a suitable blur kernel of the image.

2.3 Single Image Super-Resolution

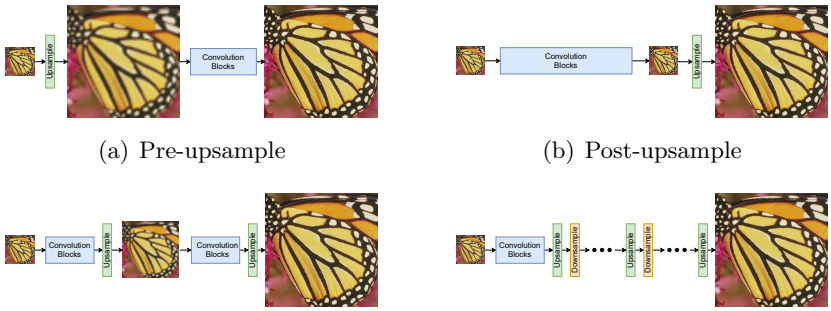
2.3.1 Problem Formulation: SISR

Again, we express that the relation of low-resolution images I_{LR} and corresponding high-resolution images I_{HR} as following:

$$I_{LR} = (k * I_{HR}) \downarrow_s + n \qquad \text{(2.1.1 revisited)}$$

In the Single Image Super-Resolution, or SISR studies, scale factor s are usually set to 2, 4, or 8. However, where $s = 2$, existing state-of-the-art methods already perform so well and they produce very similar results, which are often difficult to determine the differences for human viewers. On the other hand, where $s = 8$, the damage to I_{LR} is so severe that it is not suitable for accurate performance comparisons between models. Therefore our work mainly targets solving SISR problems with $s = 4$.

While researchers have proposed many different learning-based super-resolution methods, when and how to upsample images have been one of the main issues. Firstly, the answer to when to apply the upsampling module is often divided into four choices [36]: pre-upsampling, post-upsampling, progressive upsampling, and iterative upsampling.



(a) Pre-upsample (b) Post-upsample

(c) Progressive upsample (d) Iterative upsample

Figure 2.1: Upsample methods

Pre-Upsampling. The pre-upsampling method uses traditional methods such as bilinear or bicubic to make images larger and then insert them into neural networks to eliminate artifacts including blur and noises. This method has been popular because the structure is intuitive as the neural networks only need to finetune the coarse images. However, as research advances, the required scaling factor and the size of images to be processed for super-resolution have increased, leading to a significant increase in time and space complexity. For this reason, pre-upsampling frameworks have not been used as much in recent years.

Post-Upsampling. The post-upsampling is a method to increase computational efficiency by forwarding small-sized feature maps in neural networks and then upscale those features just before yielding output images. In contrast to pre-upsampling, which computes feature maps with the size of output images, post-upsampling analyzes the input-image-sized feature maps and produces feature maps containing high-frequency information. Using these feature maps, the model finally generates output images by applying upsampling modules such as pixel shuffle and transposed convolution at the end of the neural networks. Thanks to its high computational efficiency, post-upsampling has become one of the most widely used mainstream frameworks.

Progressive Upsampling. Progressive upsampling is similar to post-upsampling in that it upsamples the feature maps already processed in neural networks. However, unlike post-upsampling, which applies multiple upsampling steps at once, progressive upsampling applies upsampling modules step by step, stabilizing the optimization scheme for large scaling factors. However, as most datasets do not contain upsampled ground-truth for each step, it is not easy to optimize intermediate auxiliary branches. So it is highly likely that artifacts might be generated due to the intermediate upsampling modules, which can adversely affect the later part of neural networks, damaging the quality of output images.

Iterative Upsampling. Iterative Upsampling is a framework that iterates upsample and downsample, which expects neural networks to perform a deeper analysis of the mutual relationships of low-resolution and high-resolution images. However, the structure of neural networks often becomes complex, and optimization becomes unstabilized under this framework, so it is not yet widely used. Nevertheless, as research on stabilizing neural network structures and learning schemes develops, this framework also has the potential to produce great performances in future research.

Secondly, upsampling techniques used by most studies are often divided into two cases; transposed convolution [21] and pixel shuffle [29].

Transposed Convolution. A transposed convolution, also known as a fractionally strided convolution [21], reverses the forward and backward passes by transposing the matrix operation of convolutions to implement the upsampling process.

$$f_{i+1} = k * f_i = C f_i \qquad \textit{(Convolution)} \tag{2.3.1}$$

$$f_{j+1} = C^T f_j \qquad \textit{(Transposed Convolution)} \tag{2.3.2}$$

where convolution with kernel k is expressed as multiplying a matrix C with the feature map f_i. Normally in deep learning, convolutions compute the output in the manner of many-to-one. That is, multiplying by the

matrix C reduces the dimension of feature maps. On the other hand, we can expand the dimension by multiplying by the transposed matrix C^T. Although the transposed convolution has shown sharper results compared to traditional upscaling methods such as bicubic interpolation, it is not used much recently as it often yields checkerboard artifacts in the output.

Pixel Shuffle. Pixel shuffle, also known as the sub-pixel convolution layer [29], upsamples images by rearranging feature map of shape $(C \times s^2, H, W)$ to $(C, H \times s, W \times s)$. To upscale features with pixel shuffle, the previous layer should have s^2-times more channel than the number of channels we want, which increases the computational cost. However, as the upscaling process of pixel shuffle does not use any explicit interpolation filter, the layer can implicitly learn the necessary features for upscaling. Pixel shuffle has shown relatively stable performance in many novel studies and has been widely used recently.

2.3.2 Existing Methods

In recent years, many studies have been proposed to solve the SISR problem using different deep-learning techniques. In 2015, Dong et al. [8] introduced deep learning methods into the SISR problem, proposing SRCNN that is a fully convolutional neural network that enables end-to-end mapping between input and output images. In 2016, Kim et al. [14] proposed VDSR that utilizes contextual information spread over large patches of images using large receptive fields to convolutional layers. In 2017, Tai et al. [31] proposed a very deep network structure consisting of 52 convolutional layers called DRRN by designing a recursive block with a multi-path structure while Ledig et al. [18] proposed SRResNet with 16 blocks of deep ResNet and also introduced GAN-based SRGAN which is optimized for perceptual loss calculated on feature maps of the VGG [30] network.

In 2017, Lim et al. [20] proposed a novel model named EDSR. They removed every batch normalization from their network and stacked 16 residual blocks, which extracts high-frequency information from low-resolution images. In the same year, Tong et al. [34] proposed SRDenseNet, which

consists of 8 dense blocks [11] and skip connections that combine feature maps from different levels. In 2018, Zhang et al. [44] introduced a residual dense block that allows direct connections from preceding blocks, leading to a continuous memory mechanism. Zhang et al. [43] also proposed a novel model called RCAN, which added channel attention to EDSR and introduced a Residual in Residual module to construct a 10 times deeper network. They used skip connections with various lengths to help their model separately extract abundant low-frequency features and scarce but important high-frequency information from low-resolution images.

Until 2019, studies have mainly focused on modifying networks' architectural design by introducing or combining various kinds of neural blocks. However, as the neural networks became sufficiently deep and wide, structural modifications alone could expect nothing but only small marginal improvements. To overcome such issues, researchers have recently focused on the intrinsic limitations of the SISR problem or attempted to combine their neural networks with traditional model-based methods.

In 2020, Guo et al. [9] introduced cycle consistency to their network to solve the intrinsic ill-posed problem; there are infinite high-resolution images that can be downsampled to the given low-resolution input images. They reconstructed the RCAB proposed by RCAN [43] into a UNet [27] structure. In this process, they also produced images with 1/2 and 1/4 size of the target resolution from the low-resolution inputs, and then compared them with downscaled output images. Through this process, which is named dual regression, they could maintain cycle consistency and enable their networks trained with unlabeled data at the same time. Pan et al. [23] constrained their network with input information by utilizing a pixel substitution scheme from low-resolution images. They added degraded image blurred by known blur kernel to the input image and forwarded them iteratively into the deblurring network. From this process, they tried to convert a given difficult blind kernel problem to an easy non-blind problem so that their model can restore sharp images more easily.

Instead of solving the ill-posed problem by giving cycle consistency to the network with constraint from input information, several attempts

have been proposed to create a human interpretable network structure by applying meaningful kernel to the convolutional layers of the network. Huang et al. [12] introduced a Multi-Scale Hessian Filtering (MSHF) consisting of kernels that extract edges from multi-scale, leading their model to approach the high-frequency information of images from different angles and scales. On the other hand, Shang et al. [28] uses rectangular-shaped receptive fields such as 1×3 or 3×1 in parallel rather than randomly initializing 3×3 convolutional kernels. In this way, their model, named RFB-ESRGAN, becomes human interpretable and could adaptively analyze both horizontal and vertical information of images.

A study has also been proposed to apply knowledge distillation to the SISR problem to enable models to use the rich information in high-resolution images during the training phase. Lee et al. [19] forward the encoded feature of HR images to the teacher network, which shares the same structure as the student network, allowing the teacher network to use privileged information to obtain better outcomes. The student network then used variational information distillation [3] technique that allows the teacher network to distill their encoded features to the student network so it can learn how to extract privileged information, allowing the model to extract more meaningful features from a given low-resolution input.

As EDSR [20] and RCAN [43] separately extract shallow and deep features from the image on RGB color space, a study that tried to take a step further from color domain to frequency domain and decompose high frequency and low-frequency information has been proposed. Pang et al. [24] split input images into high, medium, and low frequencies and passed them to the network individually, and then aggregated each convoluted feature map adaptively to generate high-resolution images. However, instead of using mathematical methods such as FFT or DWT, they simply divided the frequency domain using three convolutional layers, which is easy to fail to extract valid and meaningful frequency information.

Chapter 3

Image Denoising

In this chapter, we propose novel modules that enhance the performance of the denoising model. We introduce our proposed methods in Section 3.1, and provide experimental results on DIV2K and SIDD dataset in Section 3.2

3.1 Proposed Methods

This section introduces some novel techniques and architecture of a deep neural network for state-of-the-art image denoising.

3.1.1 Multi-scale Edge Filtering

For successful denoising, it is important to understand the structure of images. In particular, we need to separate high-frequency and low-frequency regions and make adaptively appropriate analyses for each region to successfully detach the noise map from the original image. This is because the distribution of pixel value appears different in each region; pixels in high-frequency regions often have large variance while smaller variances are more observed in low-frequency areas.

We propose a module that extracts edges from given images to obtain information about high-frequency areas. The obtained information is

transferred to the network and used to increase restoring performance by focusing more on high-frequency regions that are difficult to reconstruct. The module consists of convolutional layers initialized with pre-defined filters, making the back-propagation scheme possible and enabling end-to-end optimization when the network is training the data.

To illustrate the layers in our module, let us first look at how convolution works in deep neural networks. Mathematically, convolution is expressed as Equation (3.1.1).

$$G(x,y) = \omega * F(x,y) = \sum_{dx=-w}^{w} \sum_{dy=-h}^{h} \omega(dx,dy)F(x+dx,y+dy) \quad (3.1.1)$$

Here, a kernel ω is given as a small matrix, usually in 3×3. As shown in Equation (3.1.1), discrete operations are applied to ω with each receptive field $\{F(x+dx,y+dy)|dx \in [-w,w] \text{ and } dy \in [-h,h]\}$ to process the feature map $F(x,y)$.

Generally, elements of the kernel are randomly initialized in deep learning methods. However, by fixing those elements by Equation (3.1.2), (3.1.3) or (3.1.4), it can be used to extract edge information from images. Also, by increasing the kernel's size to 5×5 or 7×7 rather than 3×3, we can easily extract the edge information in larger scales.

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad \text{and} \quad G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad (3.1.2)$$

Equation 3.2: Prewitt Filter

We extract multi-scale edge information from images as shown in Figure 3.1 using 9 convolutional layers that consist of 3×3, 5×5, and 7×7-sized kernels with fixed values initialized by second-order derivation filters shown in Equation (3.1.4). Multi-scale edge information is combined with deep feature maps extracted from the network and then is used to reconstruct sharp images.

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad \text{and} \quad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{3.1.3}$$

Equation 3.3: Sobel Filter

$$G_x = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad G_y = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad G_{xy} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \tag{3.1.4}$$

Equation 3.4: Second Order Derivation Filter

3.1.2 Feature Attention Module

As mentioned in Section 2.3.2, RCAN [43] achieved better results by adding channel attention to residual blocks from EDSR [20]. Figure 3.2 (a) illustrates the concept of channel attention. The channel attention takes a vector pooled from feature maps as input and feed-forward it through a series of convolutional layers. Here, the layers give us weights for each channel by operating dot products for local channel-wise regions from the average pooled vector. This process allows the network to determine the importance between channels in the feature map and focus on channels with more information.

EDSR and RCAN restore images using feature maps obtained by summing the shallow features and deep features. However, as they simply added two features, they have failed to consider the relative importance of shallow and deep features. Since shallow and deep features contain different kinds of information, such as low and high-frequency, their importance cannot be the same. Also, the characteristic of given image changes which

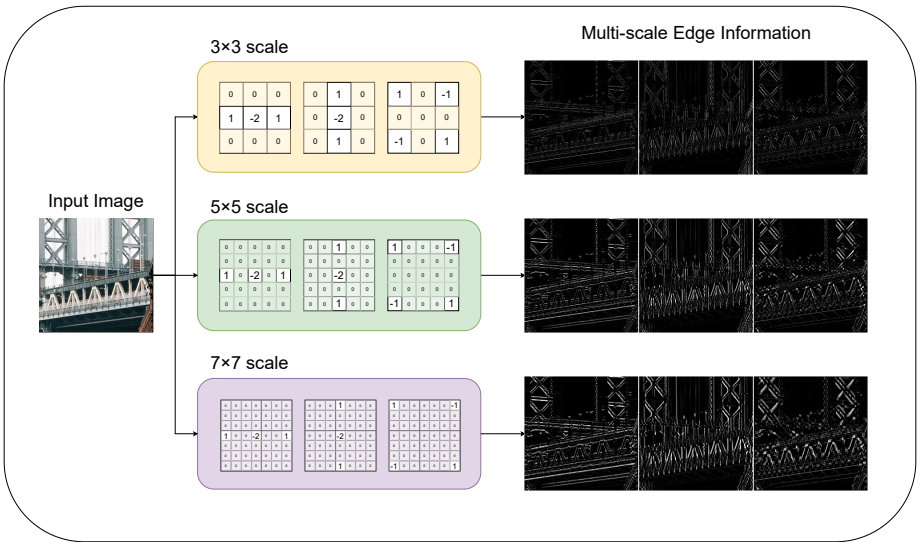
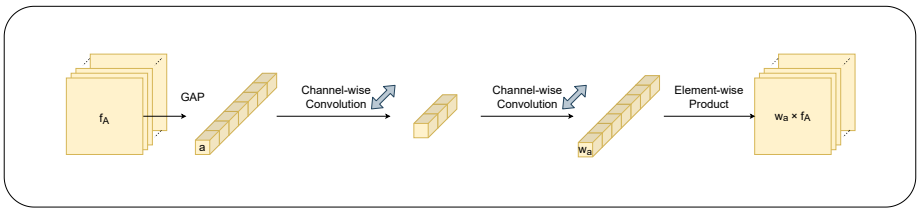


Figure 3.1: Multi-Scale Edge Detection Module

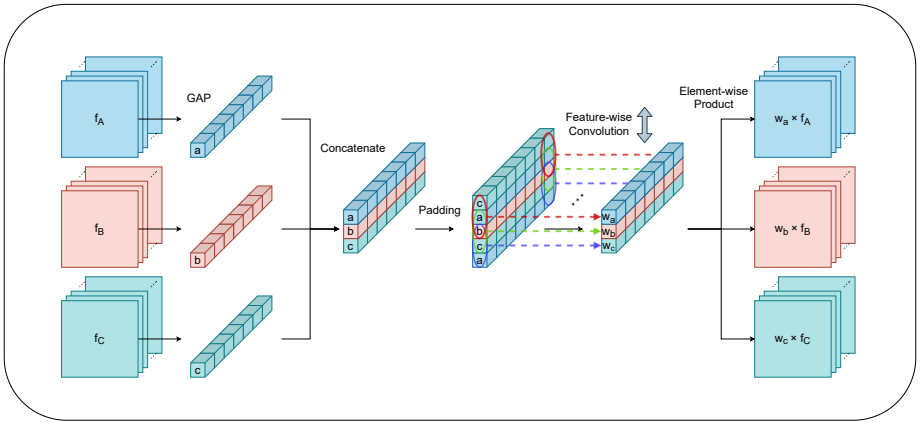
feature contains more information. Therefore, it is necessary to introduce a module that identifies the characteristics of given images and determines each importance before adding feature maps with different information.

To solve this problem, we introduce the Feature Attention Module. Before feature maps are added, the relative weights of importance are estimated by our Feature Attention Module. We calculated weight of importance in a vector form with the dimension of the channel in feature maps, considering that each channel has different importance.

Figure 3.2 shows the structure of our feature attention module. First, we concatenate vectors from each feature map by using the global average pooling layer. Then we pad the stack of vectors and feed them into convolution in the feature-wise direction rather than the channel-wise way. We padded the vectors to maintain the output dimension and make the convolution to compute every feature evenly. By multiplying each feature map in an element-wise way, we could finally obtain a weighted sum of features depending on their importance.



(a) Channel Attention



(b) Feature Attention

Figure 3.2: Feature Attention Module

3.1.3 Network Architecture

Figure 3.3 shows the structure of our proposed network for image denoising. We designed a novel denoising network by combining our multi-scale edge detecting module and feature attention module into the basic structure of RCAN. Our network extracts features from three parts: head, body, and the multi-scale edge filtering module.

The head of the network consists of only one convolutional layer without any activation function. It extracts simple low-frequency information, e.g., shape of objects. Using only one layer allows our model to extract low-level features without distorting the data by minimizing the process of the image. On the other hand, the body of the network consists of 16

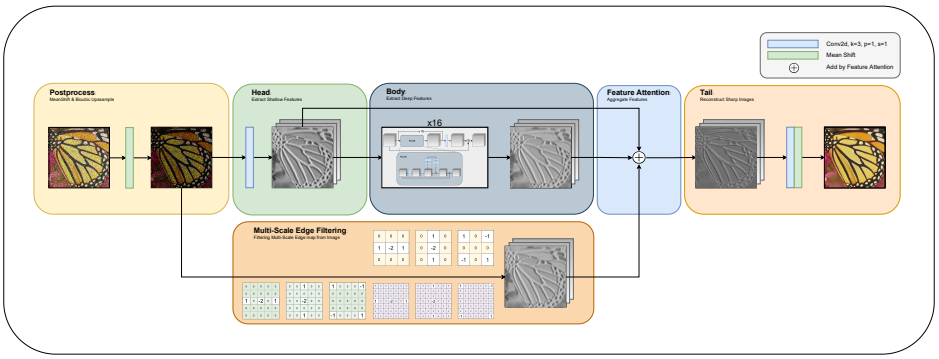


Figure 3.3: Network Architecture of Our Proposed Model

Residual Channel Attention Blocks. It extracts complex and abstract high-frequency information, such as textures or sharp patterns. These high-level features help our model understand the image and provide the intuition necessary to reconstruct areas corrupted by severe noises. However, since the body contains a significant number of convolutions, these high-level features inevitably include artifacts such as checkerboard or blurs which make it difficult to restore sharp images from high-level features only. In this reason, our model uses a combination of high-level features containing more abstract information and low-level feature with more intuitive information.

Our model also utilizes features from our multi-scale edge filtering module in addition to low-level and high-level features from the head and body of the network. To combine three feature maps with different information, we use the feature attention module described in Section 3.1.2. By focusing on more important feature maps depending on the characteristics of given images, our model could successfully remove the noise map from images. Experimental results show that our network achieves notable performance improvement, combined with our multi-scale edge filtering and feature attention module. Detailed results are shown in Section 3.2.

3.2 Experiments

This section shows our experimental results of our model on the synthetic noisy dataset and real noisy dataset. We used DIV2K [2] image with additional white Gaussian noise as a synthetic noisy dataset, and the SIDD [1] as a real noisy dataset.

3.2.1 Training Details

In the training phase, we trained our model for 200 epochs with Adam optimizer and initial learning rate 10^{-4} with learning rate decay by 0.99 for every $1,000$ steps. For each iteration, 16 batches with 192×192 sized image patches cropped original large images were used. Lastly, L1 function was used for the loss function.

3.2.2 Experimental Results on DIV2K+AWGN dataset

We first applied our model to a synthetic noisy dataset generated by adding white Gaussian noise with $\sigma = 10$, 30, and 50 to DIV2K dataset, respectively. In the training phase, we optimized our model for every variance of the noise at once. Using different kinds of noise together, our model has become flexible to more diverse noise levels.

Table 3.1 shows a comparison of the results of our model and other learning-based models using PSNR and SSIM scores. Our proposed model proved the best performance in most cases, while interestingly, our model without feature attention module showed the best result where $\sigma = 50$. This means that it is hard for the network to determine which feature map the model should attend when the input image feature maps are severely damaged. Hence, if the input image is harshly corrupted or contains very complex features, we need to remove the feature attention module from our model and make the structure more intuitive.

Table 3.1: Comparison of denoising results in PSNR and SSIM scores on DIV2K + AWGN dataset. Best scores marked in bold.

Method	**DIV2K + AWGN**		
	$\sigma = 10$	$\sigma = 30$	$\sigma = 50$
Noisy Images	32.95 / 0.7037	23.41 / 0.3280	18.97 / 0.1909
DnCNN [40]	30.28 / 0.8753	26.74 / 0.6389	23.25 / 0.4343
MemNet [32]	33.36 / 0.8815	29.67 / 0.6619	22.46 / 0.3733
FFDNet [41]	30.06 / 0.8208	29.04 / 0.7795	27.32 / 0.6892
DHDN [25]	35.31 / 0.8900	29.74 / 0.7401	26.61 / 0.6168
Ours w/o Edge	**38.64** / 0.9476	31.63 / 0.8325	27.71 / 0.7039
Ours w/o FeaAtt	38.62 / 0.9475	31.65 / 0.8313	**28.14** / **0.7352**
Ours	**38.64** / **0.9483**	**31.67** / **0.8361**	27.69 / 0.7074

(a) DIV2K/806.png + AWGN ($\sigma = 10$) – PSNR: 40.18, SSIM: 0.9629

(b) DIV2K/884.png + AWGN ($\sigma = 10$) – PSNR: 38.12, SSIM: 0.9621

(c) DIV2K/887.png + AWGN ($\sigma = 10$) – PSNR: 37.35, SSIM: 0.9663

Figure 3.4: Denoising results with PSNR and SSIM scores for DIV2K dataset with AWGN ($\sigma = 10$). From left to right: noisy image, our result, and ground-truth. Best viewed on screen.

(a) DIV2K/806.png + AWGN ($\sigma = 30$) – PSNR: 33.85, SSIM: 0.8872

(b) DIV2K/884.png + AWGN ($\sigma = 30$) – PSNR: 31.67, SSIM: 0.8611

(c) DIV2K/887.png + AWGN ($\sigma = 30$) – PSNR: 30.35, SSIM: 0.8931

Figure 3.5: Denoising results with PSNR and SSIM scores for DIV2K dataset with AWGN ($\sigma = 30$). From left to right: noisy image, our result, and ground-truth. Best viewed on screen.

(a) DIV2K/806.png + AWGN ($\sigma = 50$) – PSNR: 28.99, SSIM: 0.7872

(b) DIV2K/884.png + AWGN ($\sigma = 50$) – PSNR: 27.74, SSIM: 0.7646

(c) DIV2K/887.png + AWGN ($\sigma = 50$) – PSNR: 26.04, SSIM: 0.7986

Figure 3.6: Denoising results with PSNR and SSIM scores for DIV2K dataset with AWGN ($\sigma = 50$). From left to right: noisy image, our result, and ground-truth. Best viewed on screen.

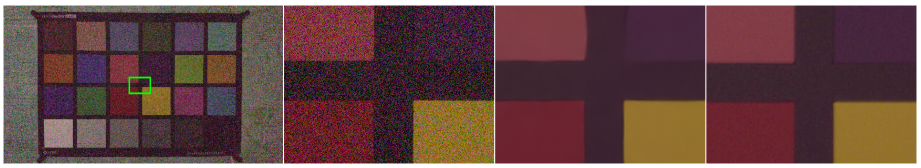
3.2.3 Experimental Results on SIDD dataset

To evaluate our model on real noisy images, we used the Smartphone Image Denoising Dataset, which is often called SIDD [1]. This dataset consists of pairs of real noisy images taken under various conditions using smartphone cameras and ground-truth images, of which defective pixels are corrected manually. We showed that our model could solve real-world denoising problems by training and evaluating our model on the SIDD dataset.

Table 3.2 shows a comparison of the results of our model and other learning-based models on the SIDD dataset.

Table 3.2: Comparison of denoising results in PSNR and SSIM scores on SIDD dataset. Best scores marked in bold.

Method	**SIDD**	
	PSNR	**SSIM**
Noisy Images	34.19	0.5472
DnCNN [40]	43.60	0.9275
MemNet [32]	44.24	0.9249
DHDN [25]	46.99	0.9677
Ours w/o Edge	47.14	0.9692
Ours w/o FeaAtt	47.12	**0.9693**
Ours	**47.21**	**0.9693**



(a) SIDD/0015 – PSNR: 36.09, SSIM: 0.8700

(b) SIDD/0050 – PSNR: 36.44, SSIM: 0.7811

(c) SIDD/0145 – PSNR: 39.78, SSIM: 0.9088

(d) SIDD/0200 – PSNR: 42.60, SSIM: 0.9501

Figure 3.7: Denoising results with PSNR and SSIM scores for SIDD dataset. From left to right: noisy image, our result, and ground-truth. Best viewed on screen.

Chapter 4

Image Deblurring

In this chapter, we propose a novel network structure for image deblurring. Our image deblurring network shares the basic structure with our denoising network from Section 3.1.3, but is more capable of image analysis on multiple scales due to several modifications. We first introduce some techniques added for image deblurring in Section 4.1, and present experimental results on REDS and Flickr2K dataset in Section 4.2.

4.1 Proposed Methods

This section introduces some techniques that we added to our denoising network to solve the image deblurring problem. We propose a kernel blind deblurring method that sharpens blurry images without information about the blur kernel. To this end, we introduce a neural network architecture that can identify the global context as well as local patches in the image. While many studies try to estimate blur kernel [5], and use the predicted blur kernel to make the difficult blind problem easier [4], we chose to solve the deblurring problem without estimating or using any information about the blur kernel. Instead of predicting the blur kernel, our model estimates the relationship between the blurry image and the target image by analyzing the global context.

4.1.1 Multi-Scale Feature Analysis

Image deblurring requires an understanding of the local and global context of the blurry images. To this end, we modified the downsampling-upsampling structure of U-Net [27] to let our network recognize images at various scales. Unlike U-Net, however, which pools feature maps to obtain downsampled features, we made our network take 1/2 and 1/4 sized input images downsampled by bicubic interpolation. While damage such as checkerboard artifacts are frequently observed in commonly used max-pooling or strided convolution, we adopted bicubic interpolation because it is relatively free from such corruption and thus maintains the context of input images better than other methods.

We designed a neural network that analyzes images resized in multiple scales, leading to extracting features in wider ranges than other models. In addition, our model generates output images in multiple scales, corresponding to sizes of reshaped input images. This allows our model to learn how to restore images in various scales, which makes our model flexible so it can be applied not only to deblurring but also to super-resolution.

4.1.2 Network Architecture

Figure 4.1 shows the structure of our proposed network. Our deblurring model first resizes image into three different scales, then extracts low-level and high-level features through its head and body for each scale, respectively. The heads of the network consist of one convolutional layer each, like our denoising model, while the bodies are composed of a different number of Residual Channel Attention Blocks depending of the scale; from largest to smallest scale, each body consists of 4, 16, and 64 blocks, respectively. Here, we let bodies on smaller scales have more blocks with deeper layers because they use less GPU memory as their computation is relatively lower.

To take full advantage of deeply stacked bodies on smaller scale, we combined low-level features from smaller scales with those from larger scales through our feature attention module before we put them into each

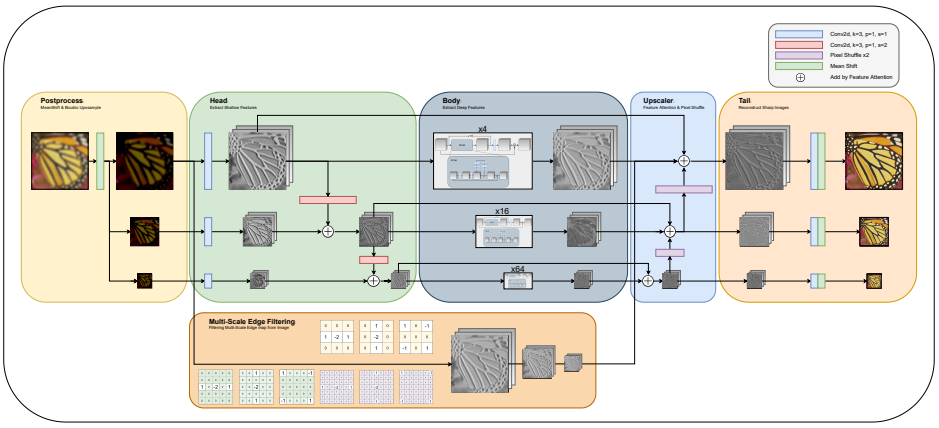


Figure 4.1: Network Architecture of Our Proposed Model for Image Deblurring

body. This allows deeper bodies for smaller scales take more diverse features and generate richer high-level features.

After the bodies extract high-level features, we combine high-level features with low-level and multi-scale edge filtering module for each scale by feature attention. Here, we upscale and combine high-level features from smaller scale to larger scales, which allows the tails for larger scale take more features from diverse scales. To send features to different scales, we use strided convolution for downscale and pixel shuffle to upscale the features.

In the training phase, we compute the errors by comparing 1/2 and 1/4 sized output with bicubicly downsampled versions of ground-truth of corresponding sizes. This can be expressed as following equation:

$$\mathcal{L} = L_1(I_{HR}, I_{SR}) + \lambda_2 L_1(I_{HR_{\times 1/2}}, I_{SR_{\times 1/2}}) + \lambda_4 L_1(I_{HR_{\times 1/4}}, I_{SR_{\times 1/4}}) \tag{4.1.1}$$

where weights are set proportional to the number of pixels in each scale, that is, $\lambda_2 = (1/2)^2 = 0.25$ and $\lambda_4 = (1/4)^2 = 0.0625$.

4.2 Experiments

This section shows our experimental results of our model on the two dataset consisting of real and synthetic blurry images, respectively. We used Flickr2K [20] image with randomly chosen blur kernel from set of isotrocpic and anisotropic Gaussian blurs as a synthetic blurry dataset, and the REDS dataset from "NTIRE 2021 Image Deblurring Challenge - Track2. JPEG Artifacts" as a real blurry dataset.

4.2.1 Training Details

In the training phase, we trained our model for 800 epochs for small image patches and 20 epochs for large image patches with Adam optimizer and initial learning rate 10^{-4} decayed by multiplying 0.99 for every $1,000$ steps. For each iteration, we used 16 batches with 192×192 sized cropped patches for epochs with small images where one batch with 1280×720 sized image was used for epochs with large images. Lastly, we used the L1 function to compute the loss of our prediction.

4.2.2 Experimental Results on Flickr2K dataset

We first trained and evaluated our deblurring model on Flickr2K dataset [20]. While REDS consists of similar images from several daily videos, Flickr2K contains different images with various objects and detailed patterns. Therefore, it is suitable for an extensive experiment to show that our model can be applied to images with more diverse information.

To create blurry images in various conditions, we applied randomly chosen blur kernels from set of isotropic and anisotropic Gaussian kernels of various sizes and angles to randomly cropped and rotated patches. By augmenting the blurry images, our model could observe and learn the various blurring conditions on the limited images.

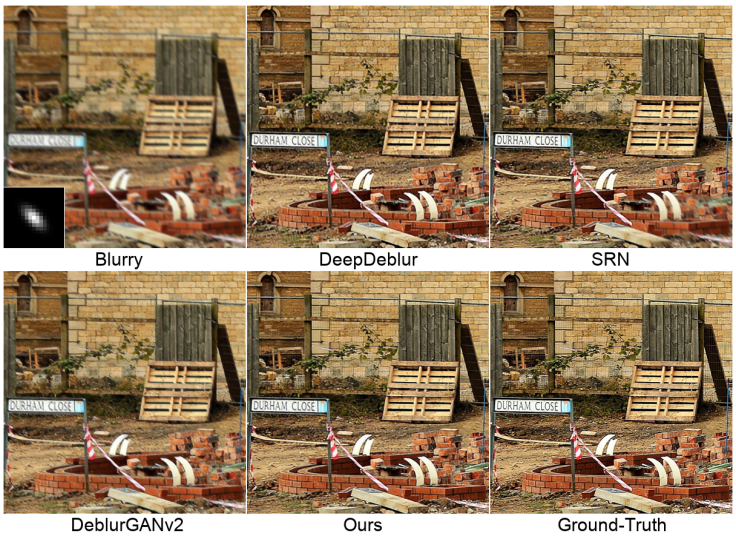
Table 4.1 shows our model achieves the state-of-the-art results on Flickr2K dataset.

Table 4.1: Comparison of deblurring results in PSNR and SSIM scores on Flickr2K dataset. Best scores marked in bold.

Method	**Flickr2K**		
	PSNR	**SSIM**	**MS-SSIM**
Blurry Images	29.19	0.7655	0.9368
DeepDeblur [22]	33.75	0.8990	0.9841
SRN [33]	34.90	0.9070	0.9858
DeblurGANv2 [17]	30.78	0.8546	0.9746
Ours w/o Edge	36.32	0.9253	0.9903
Ours w/o FeaAtt	36.36	0.9252	0.9902
Ours	**36.38**	**0.9264**	**0.9905**

(a) Flickr2K image with isotrocpic Gaussian blur

Figure 4.2: Visual Comparison of SISR methods on Flickr2K validation data. Best viewed on screen.



(a) Flickr2K image with anisotrocpic Gaussian blur

(b) Flickr2K image with anisotrocpic Gaussian blur

Figure 4.3: Visual Comparison of SISR methods on Flickr2K validation data. Best viewed on screen.

4.2.3 Experimental Results on REDS dataset

This section shows our experimental results on the REDS dataset from "NTIRE 2021 Image Deblurring Challenge - Track2. JPEG Artifacts".

Table 4.2 shows comparison of deblurring results of various state-of-the-art models. We measured the performance of results in PSNR, SSIM, and Multi-Scale SSIM scores.

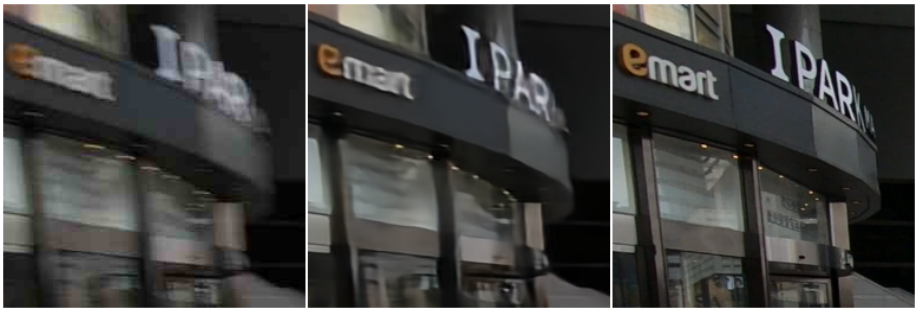
We also provide the results of ablation studies that evaluate the effect of our Multi-Scale Edge Filtering and Feature Attention Module. Ablations studies show that our proposed model achieves top scores at PSNR, SSIM and MS-SSIM. Considering that PSNR measures absolute errors and SSIM measures the perceived change in structural information, based on luminance and contrast of images, it can be inferred that the Feature Attention Module helps the model understand the structural information.

Figure 4.4, 4.5, and 4.6 show some selected deblurring results of our proposed model on REDS dataset. Our model successfully reconstruct objects that are difficult to identify from the blurry image to identifiable levels.

Table 4.2: Comparison of deblurring results in PSNR and SSIM scores on REDS - JPEG dataset. Best scores marked in bold.

Method	**REDS - NTIRE 2021 JPEG Artifact**		
	PSNR	**SSIM**	**MS-SSIM**
Blurry Images	26.51	0.7063	0.8739
DeepDeblur [22]	28.68	0.7690	0.9148
SRN [33]	28.60	0.7588	0.9129
DeblurGANv2 [17]	26.82	0.7220	0.8875
Ours w/o Edge	29.56	0.7901	0.9309
Ours w/o FeaAtt	29.78	0.7959	0.9340
Ours	**29.80**	**0.7966**	**0.9342**

(a) REDS/val_blur_jpeg/001/00000019.jpg – PSNR: 25.72, SSIM: 0.6696



(b) REDS/val_blur_jpeg/004/00000099.jpg – PSNR: 27.95, SSIM: 0.7700

(c) REDS/val_blur_jpeg/006/00000099.jpg – PSNR: 30.45, SSIM: 0.6756

Figure 4.4: Deblurring results with PSNR and SSIM scores for REDS validation dataset from NTIRE 2021 Challenge - Track 2. JPEG Artifacts. From left to right: blurry image, our result, and ground-truth. Best viewed on screen.

(a) REDS/val_blur_jpeg/008/000000529.jpg – PSNR: 26.67, SSIM: 0.7074

(b) REDS/val_blur_jpeg/022/00000049.jpg – PSNR: 22.85, SSIM: 0.5783

(c) REDS/val_blur_jpeg/023/00000009.jpg – PSNR: 27.08, SSIM: 0.7140

Figure 4.5: Deblurring results with PSNR and SSIM scores for REDS validation dataset from NTIRE 2021 Challenge - Track 2. JPEG Artifacts. From left to right: blurry image, our result, and ground-truth. Best viewed on screen.

(a) REDS/val_blur_jpeg/014/00000069.jpg – PSNR: 24.93, SSIM: 0.5693

(b) REDS/val_blur_jpeg/028/00000079.jpg – PSNR: 31.40, SSIM: 0.8666

(c) REDS/val_blur_jpeg/029/00000029.jpg – PSNR: 29.76, SSIM: 0.8294

Figure 4.6: Deblurring results with PSNR and SSIM scores for REDS validation dataset from NTIRE 2021 Challenge - Track 2. JPEG Artifacts. From left to right: blurry image, our result, and ground-truth. Best viewed on screen.

Chapter 5

Single Image Super-Resolution

In this chapter, we introduce a novel learning-based method for Single Image Super-Resolution. By combining methods discussed in Chapter 3 and 4 with techniques that we will introduce in this chapter, we could obtain state-of-the-art results. We first introduce some novel methods and network structures for SISR in Section 5.1, and then compare the results of our method with other state-of-the-art methods on a various dataset in Section 5.2.

5.1 Proposed Methods

This section introduces a novel structure of deep neural network and training schemes for state-of-the-art SISR. The structure of neural network for SISR and its internal modules that process images are mostly the same as those of our deblurring model. Only a small structural modification in the head part of the model has been make because the size of the input and the target images are different in SISR.

Furthermore, due to the large information loss of input images compared to deblurring, the training required for SISR neural networks is

generally more difficult. Therefore in this section, we mainly focus on methods to increase the training efficiency of our model by recognizing and weighting regions that are more difficult to train or contain more critical information.

5.1.1 High-Pass Filtering Loss

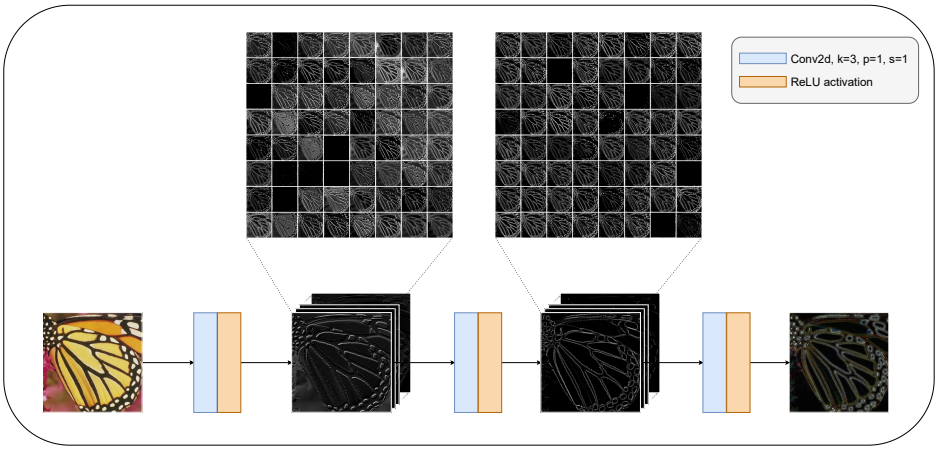
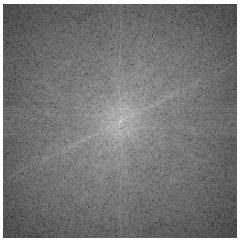


Figure 5.1: High-Pass Filtering with CNN Model

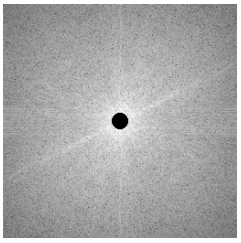
The concept of perceptual loss was first introduced by Johnson et al. [13] who tried to solve the image transformation problem by comparing content and style discrepancies between two images. They used VGG-16 [30] pre-trained for image classification as the loss network and measured perceptual differences of output and ground-truth images. Motivated by their method, we propose a loss function that compares feature differences in the high-frequency domain instead of comparing per-pixel differences in a color space.

The commonly used perceptual loss uses the VGG-16 network pre-trained on ImageNet dataset. However, the network trained on image classification is optimized to extract feature representation that contains information about the class of objects in images. The objective of network

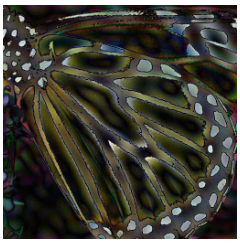
(a) Original Image



(b) Frequency Spectrum of Original Image



(c) High Pass Filtered Frequency Spectrum



(d) High Pass Filtered Image

(e) Convoluted Image

Figure 5.2: A visual example of high-pass filtering. (a) Original image. (b) Frequency spectrum in the polar form where the spectrum is shifted to place zero frequency at the center. (c) High-pass filtered Frequency spectrum. (d) High-pass filtered image, or inverse Fourier transform of (c). (e) High-frequency domain of original image from our model.

is to figure out what objects are in the image, not to extract detailed patterns or complex high-frequency information. What we need, however, is neural networks that extract such detailed patterns and high-frequency information and feature representations of the networks that are needed to extract such information. Because the commonly used perceptual loss is not appropriate to our problem solving, we have trained a neural network that is optimized to high-frequency extraction.

We first extracted high-frequency signals by applying a high-pass fil-

ter to the image transformed into the frequency domain via Fast Fourier Transform. Then, we trained a simple three-layered CNN, or high-pass filtering network, which takes images as input and generates high-pass filtered signals. Figure 5.2 shows an example of high-frequency signals extracted by a traditional high-pass filter using FFT and our high-pass filtering network. Figure 5.1 shows a visualization of feature maps produced by intermediate layers of the network during extracting high-frequency signals.

We utilized this high-pass filtering network as the loss network and defined the high-pass filtering loss function as following:

$$\mathcal{L}_{hf}(I_{SR}, I_{HR}) = loss^{\phi}_{conv.0}(I_{SR}, I_{HR}) + loss^{\phi}_{conv.1}(I_{SR}, I_{HR}) \qquad (5.1.1)$$

where ϕ denotes the high-pass filtering network. The loss network ϕ analyzes images from various perspectives to generate high-frequency signals where intermediate layers give us abstract feature maps, including edges. We measure the feature difference of I_{SR} and I_{HR} by feed-forwarding two images to fixed ϕ where the feature difference is trainable by back-propagation as it is generated through convolutional layers. By minimizing the high-pass filtering loss, high-frequency features are added to the I_{SR}, allowing us to obtain sharper images.

5.1.2 Gradient Magnitude Similarity Map Masking

The local perceptual quality of output images often varies by region. In general, the lower perceptual quality is more observed in areas containing detailed and irregular patterns, but these results depend on which model is used. To learn models that perform evenly, we need to know which part of the resulting image is poor, and therefore more training is needed.

Here, we adopted the Gradient Magnitude Similarity (GMS) map [38] to evaluate the local quality of images. The gradient magnitude of given image I is computed as follows:

$$GM(I) = \sqrt{(I * G_x)^2 + (I * G_y)^2} \qquad (5.1.2)$$

(a) Hard Gradient Magnitude Similarity Map Masking

(b) Soft Gradient Magnitude Similarity Map Masking

Figure 5.3: Visual examples of hard and soft version of Gradient Magnitude Similarity map masking. From left to right: GMS map, binarized GMS map, GMS map masked image, Hard/Soft GMS map masked image, and Original Image.

where G_x and G_y denote prewitt filters given in Equation (3.1.2). With the gradient magnitudes of I_{HR} and I_{SR}, we compute the GMS map as follows:

$$GMS(I_{HR}, I_{SR}) = 1 - \frac{2GM(I_{HR})GM(I_{SR}) + c}{GM(I_{HR})^2 + GM(I_{SR})^2 + c} \tag{5.1.3}$$

where we set $c = 170$ for pixel values in $[0, 255]$. Note that the value of the GMS map is closer to zero where two images are similar while it is closer to one where two images are different.

To give information about which area is more damaged and thus training should be weighted to the loss function, we multiply I_{HR} and I_{SR} with the GMS map before we put them into our loss functions. However, since the GMS map is calculated pixel-wise, it can be computed high for some lucky locations with similar pixel values even where they are contained in severely corrupted regions. So we first binarized the GMS map and then remove tiny or trivial regions using the opening which is defined as erosion followed by dilation.

In the mathematical morphology, the opening of a binary image A by the structuring element B is expressed as follows:

$$\textit{Erosion:} \quad A \ominus B = \bigcap_{b \in B} A_{-b} \tag{5.1.4}$$

$$\textit{Dilation:} \quad A \oplus B = \bigcup_{b \in B} A_b \tag{5.1.5}$$

$$\textit{Opening:} \quad A \circ B = (A \ominus B) \oplus B \tag{5.1.6}$$

where A_b denotes the translation of A by b. The opening is often applied to coarse images to remove pixel-wise outliers and make them locally smooth. Here, adopting the opening to the coarse GMS map allows us to eliminate pixel noise and acquire more smooth labels while maintaining information about the locally damaged area inside the image.

Two images on left side of Figure 5.3(a) shows an visual example of applying the opening to GMS map. We can observe that the map distinguishes between well-reconstructed and poorly-reconstructed area smoother when the opening followed by thresholding is applied to the coarse GMS map. Here, we use the opened-binarized GMS map, or the hard GMS map, to mask images to let our network re-train only on poorly-reconstructed areas.

5.1.3 Soft Gradient Magnitude Similarity Map Masking

The hard GMS map assigns each pixel a hard label whether to train or not. In practice, however, it is more reasonable to express with score or probability of how much pixel should be trained. Therefore, we transform the discretized hard GMS map into soft GMS map so it represent the pixel-wise score.

To soften the hard GMS map, we smoothed the boundaries between different regions within the hard GMS map by applying blurring with isotropic Gaussian kernel and additional image opening to remove outliers in an iterative manner. In the soft GMS map, pixels at the center of well or poorly-reconstructed area have more confident score close to 0 or 1,

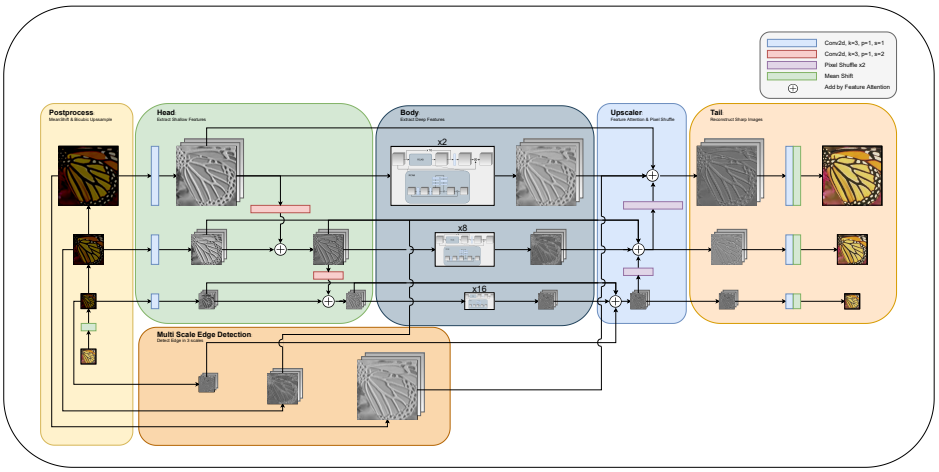


Figure 5.4: Network Architecture of our proposed model

respectively, while scores close to 0.5 are assigned to pixels if they are close to boundaries. Figure 5.3 show examples of masked results using the hard and the soft GMS maps.

5.1.4 Network Architecture

Our proposed SISR model shares the network architecture with our image deblurring network except for the data input part. Figure 5.4 illustrates structure of our proposed SISR model. While our deblurring model downscales the input image, our SISR model upscales the input and then forwards them into the body part to analyze them in multi-scale. Here, we adopted the bicubic interpolation method to resize input images.

Most learning-based models use transposed convolutions or pixel shuffle to upscale the image; pixel shuffle is mainly used recently, as explained in Section 2.3. Our model adopt the pixel shuffle method to upsample image in the tail part of the network. However, our model also includes upsampling process in the preprocessing part.

We compute the loss function by comparing not only desired $\times 4$ sized output but also $\times 1$ and $\times 2$ sized outputs with bicubicly downsampled ver-

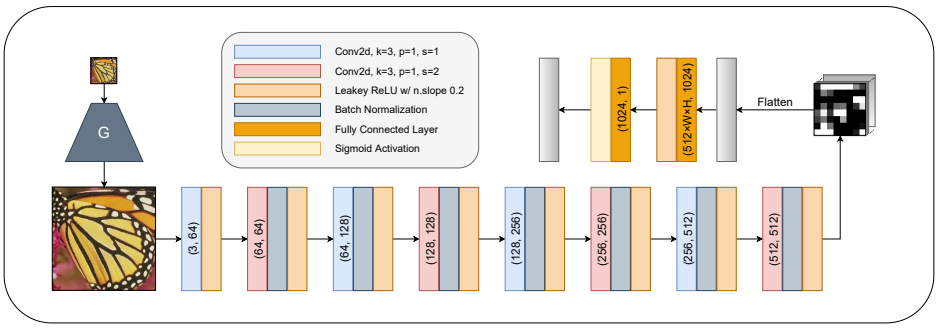


Figure 5.5: Network Architecture of Discriminator model

sions of ground-truth images. This can be expressed as following equation:

$$\mathcal{L} = L_1(I_{HR}, I_{SR}) + \lambda_2 L_1(I_{HR_{\times 1/2}}, I_{SR_{\times 1/2}}) + \lambda_4 L_1(I_{HR_{\times 1/4}}, I_{SR_{\times 1/4}}) \tag{5.1.7}$$

where weights are set proportional to the number of pixels in each resized image, that is, $\lambda_2 = (1/2)^2 = 0.25$ and $\lambda_4 = (1/4)^2 = 0.0625$.

5.1.5 Adversarial Training for Perceptual Generative Model

So far, we have trained our model based on the L1 loss function. This method is useful for obtaining high PSNR scores as it aims to minimize the absolute error of the result and the target image. However, such PSNR-oriented methods often generate result images that are over-smooth and perceptually unnatural when the resolution difference between I_{LR} and I_{HR} is large.

In 2018, Zhang et al. [42] introduced a metric named LPIPS, the Learned Perceptual Image Patch Similarity, to overcome such limitations of PSNR measurement. However, Rad et al. [26] argues that LPIPS is insufficient to assess the perceptual image quality as it shows a similar trend to the traditional distortion-based SSIM method. Besides LPIPS, many researchers have tried to develop suitable algorithms for the perceptual image quality assessment that correlate well with MOS or Mean opinion score. But it remains challenging, and no convincing results have

been made yet.

In this thesis, we add the adversarial training phase to our proposed model using the discriminator (illustrated in Figure 5.5), which enabled our model to generate much more natural images and leave research on finding effective methods for perceptual image quality assessment as future studies.

Algorithm 1: Training Steps of Our Proposed Net

Input: Paired data: LR, HR;
while *not convergent* **do**
  Sample labeled data $\{lr_i, hr_i\}_{i=1}^{m}$;
  Get SR output;
  for $i = 1$ *to* m **do**
    $sr_i = \text{model}(lr_i)$
    if $psnr(sr_i, hr_i) > \theta$ **then**
      $sr_i = \text{Soft GMS Map Masking}(sr_i)$
      $hr_i = \text{Soft GMS Map Masking}(hr_i)$
    end
  end
  Update the model by back-propagating the Loss function:

$$\text{Loss} = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}_1(sr_i, hr_i) + \lambda_{hf}\mathcal{L}_{hf}(sr_i, hr_i) + \lambda_{me}\mathcal{L}_{me}(sr_i, hr_i)$$

end

Algorithm 2: Training Steps of Our Proposed GAN

Input: Paired data: LR, HR;
Load the pretrained model;
while *not convergent* **do**
  Sample labeled data $\{lr_i, hr_i\}_{i=1}^{m}$;
  Get SR output;
  for $i = 1$ *to* m **do**
    $sr_i = \text{model}(lr_i)$
  end
  Update the discriminator with the Loss function:

$$\text{Loss}_D = \frac{-1}{m} \sum_{i=1}^{m} [\log(D(hr_i)) + \log(1 - D(sr_i))]$$

  Update the generator with the Loss function:

$$\text{Loss}_G = \frac{1}{m} \sum_{i=1}^{m} [\mathcal{L}_{per}(sr_i, hr_i) - \lambda_{adv} \log(D(sr_i))]$$

end

5.2 Experiments

This section shows experimental results of our models: PSNR-oriented version (Our Net) and perceptual version (Our GAN). We trained and validated our models on the DIV2K dataset and then tested them on Set5/Set14 dataset. We also applied our model on the REDS dataset from "NTIRE 2021 Image Deblurring Challenge - Track 1. Low Resolution". The results are provided in Section 5.2.2, 5.2.3, and 5.2.4.

5.2.1 Training Details

In the training phase, we trained our model for 800 epochs for small image patches and 20 epochs for large image patches with Adam optimizer and initial learning rate 10^{-4} with learning rate decay by 0.99 by every $1,000$ steps. For each iteration, 16 batches with 192×192 sized cropped patches

from large images were used for epochs with small images where one batch with 320×180 sized input image and 1280×720 sized target image was used for epochs with large images. Lastly, L1 function was used for the loss function.

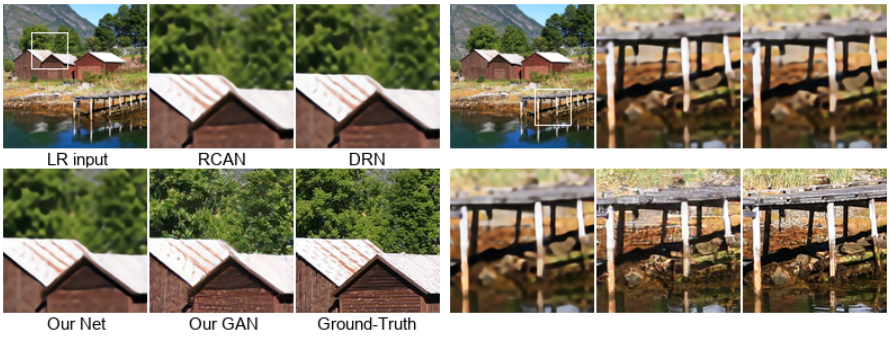
5.2.2 Experimental Results on DIV2K dataset

As the DIV2K dataset provides $\times 2$ and $\times 4$ downsampled images, we trained our model on each case and evaluated the results. Especially when training $\times 2$ images, we weighted the loss term on $\times 2$ sized outputs in the training phase and then extracted them as final results instead of changing our model's structure.

Compared to other blind SISR methods, our proposed net achieves higher PSNR and SSIMS scores while our proposed GAN produces perceptually more natural results. Figures from 5.6 to 5.10 show detailed results of our models.

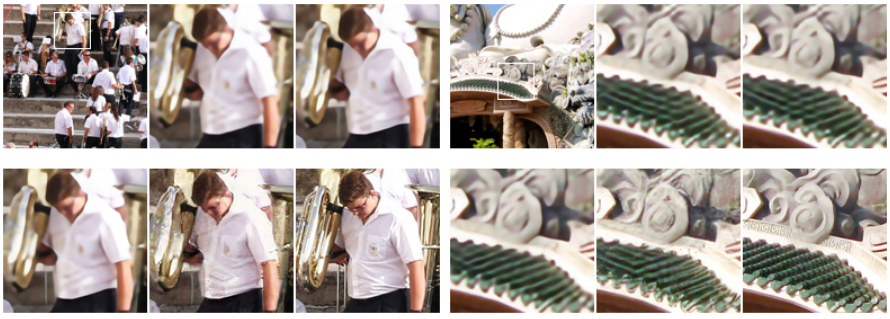
Table 5.1: Comparison of SISR results on DIV2K dataset. Best scores marked in bold.

Method	**Scale**	**DIV2K**		
		PSNR ↑	**SSIM** ↑	**LPIPS** ↓
Bicubic	×2	31.07	0.8662	0.1769
EDSR [20]		34.31	0.9190	0.0679
RCAN [43]		34.68	0.9227	0.0630
Our Net w/o Edge		35.04	0.9251	0.0560
Our Net w/o FeaAtt		**35.11**	**0.9279**	0.0547
Our Net		35.10	0.9270	0.0540
Our GAN		31.97	0.8813	**0.0257**
Bicubic	×4	26.78	0.6839	0.4087
EDSR [20]		28.65	0.7594	0.2451
RCAN [43]		28.93	0.7680	0.2371
DRN [9]		28.91	0.7676	0.2363
Our Net w/o Edge		29.00	0.7705	0.2240
Our Net w/o FeaAtt		29.10	0.7740	0.2239
Our Net		**29.11**	**0.7743**	0.2225
Our GAN		25.79	0.6598	**0.1096**



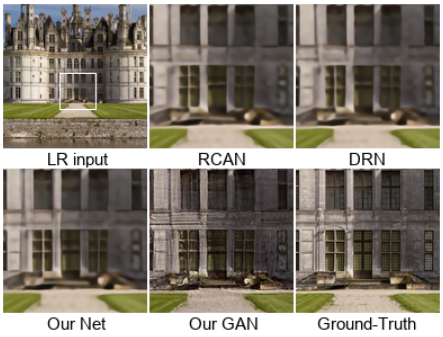
(a) DIV2K/808.png (b) DIV2K/808.png

(c) DIV2K/823.png (d) DIV2K/823.png

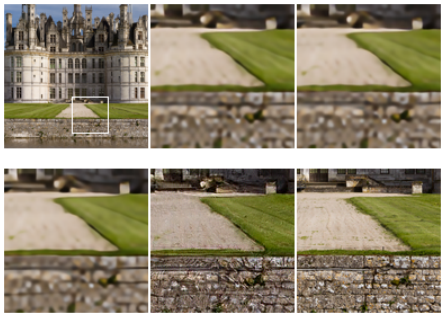


(e) DIV2K/825.png (f) DIV2K/818.png

Figure 5.6: Visual Comparison of SISR methods on DIV2K validation data. Best viewed on screen.



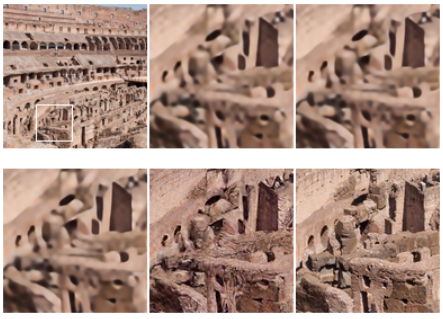
(a) DIV2K/830.png



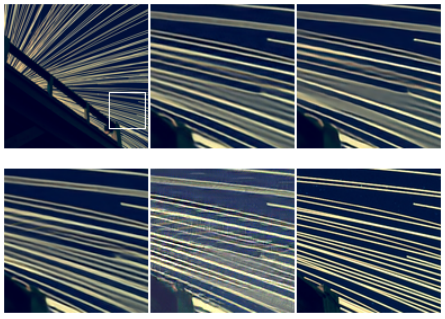
(b) DIV2K/830.png

(c) DIV2K/841.png

(d) DIV2K/841.png



(e) DIV2K/854.png



(f) DIV2K/828.png

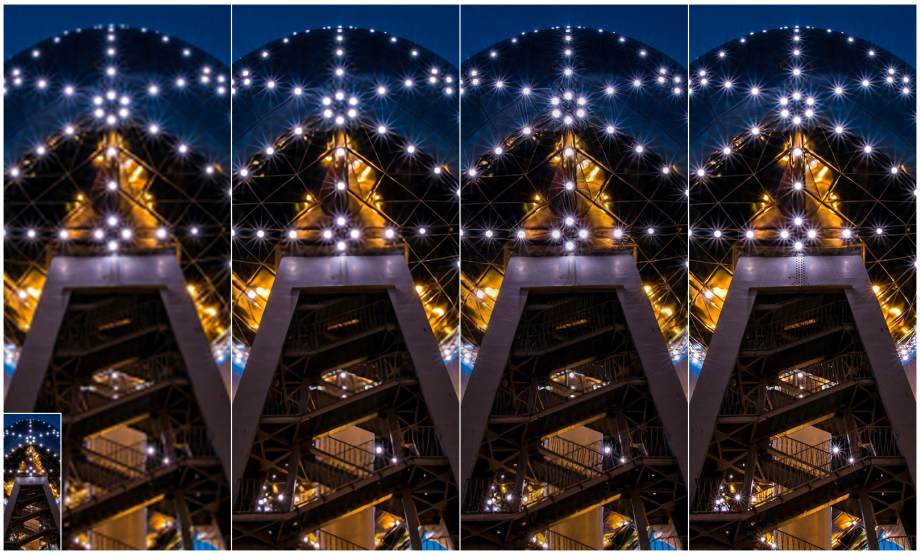
Figure 5.7: Visual Comparison of SISR methods on DIV2K validation data. Best viewed on screen.

(a) DIV2K/806.png

(b) DIV2K/810.png

Figure 5.8: SISR results of our proposed methods on DIV2K validation data. From left to right: bicubic interpolation (with I_{LR} at the lower left corner), our net, our GAN, and ground-truth. Best viewed on screen.

(a) DIV2K/837.png



(b) DIV2K/851.png

Figure 5.9: SISR results of our proposed methods on DIV2K validation data. From left to right: bicubic interpolation (with I_{LR} at the lower left corner), our net, our GAN, and ground-truth. Best viewed on screen.

(a) DIV2K/852.png

(b) DIV2K/861.png

Figure 5.10: SISR results of our proposed methods on DIV2K validation data. From left to right: bicubic interpolation (with I_{LR} at the lower left corner), our net, our GAN, and ground-truth. Best viewed on screen.

5.2.3 Experimental Results on Set5/Set14 dataset

We used Set5/Set14 dataset to evaluate our models that are trained on the DIV2K dataset. As Figure 5.12, 5.13 and Table 5.2 show, our proposed net achieved the highest PSNR and SSIM scores while our proposed GAN generated the most perceptually natural images.

Table 5.2: Comparison of SISR results on Set5 dataset. Best scores marked in bold.

Method	**Scale**	**Set5**		
		PSNR ↑	**SSIM** ↑	**LPIPS** ↓
Bicubic	×2	33.97	0.9125	0.1093
EDSR [20]		37.68	0.9425	0.0414
RCAN [43]		37.87	0.9421	0.0416
Our Net w/o Edge		38.10	0.9437	0.0378
Our Net w/o FeaAtt		**38.22**	**0.9464**	0.0371
Our Net		38.19	0.9450	0.0374
Our GAN		35.13	0.9158	**0.0136**
Bicubic	×4	26.46	0.7349	0.3157
EDSR [20]		31.69	0.8566	0.1413
RCAN [43]		31.98	0.8615	0.1368
DRN [9]		32.01	0.8619	0.1406
Our Net w/o Edge		32.07	0.8585	0.1379
Our Net w/o FeaAtt		32.04	0.8596	0.1385
Our Net		**32.20**	**0.8645**	0.1364
Our GAN		29.72	0.7869	**0.0599**

Table 5.3: Comparison of SISR results on Set14 dataset. Best scores marked in bold.

Method	**Scale**	**Set14**		
		PSNR ↑	**SSIM** ↑	**LPIPS** ↓
Bicubic	×2	31.19	0.8343	0.1665
EDSR [20]		33.50	0.8742	0.0760
RCAN [43]		33.65	0.8767	0.0748
Our Net w/o Edge		33.76	0.8957	0.0684
Our Net w/o FeaAtt		**33.84**	**0.8966**	0.0689
Our Net		33.80	**0.8966**	0.0679
Our GAN		31.33	0.8508	**0.0349**
Bicubic	×4	25.13	0.6411	0.3908
EDSR [20]		28.17	0.7366	0.2405
RCAN [43]		28.38	0.7427	0.2365
DRN [9]		28.38	0.7429	0.2360
Our Net w/o Edge		28.44	0.7438	0.2304
Our Net w/o FeaAtt		28.46	0.7449	0.2315
Our Net		**28.54**	**0.7473**	0.2312
Our GAN		25.78	0.6510	**0.1192**

(a) Bicubic (29.10, 0.7970, 0.2440)

(b) RCAN [43] (33.73, 0.8692, 0.1680)

(c) DRN [9] (33.69, 0.8684, 0.1707)

(d) Our Net (**33.82**, **0.8703**, 0.1767)

(e) Our GAN (31.91, 0.7730, **0.0787**)

(f) Ground-Truth

Figure 5.11: Visual Comparison of SISR results on Set5 dataset with PSNR, SSIM, and LPIPS scores. I_{LR} at lower left corner of (a). Best scores marked in bold. Best viewed on screen.

(a) Bicubic
(20.91, 0.5326, 0.4812)

(b) RCAN [43]
(23.52, 0.7128, 0.2372)

(c) DRN [9]
(23.53, 0.7155, 0.2194)

(d) Our Net
(**23.68**, **0.7235**, 0.2149)

(e) Our GAN
(20.96, 0.6133, **0.1295**)

(f) Ground-Truth

Figure 5.12: Visual Comparison of SISR results on Set14 dataset with PSNR, SSIM, and LPIPS scores. I_{LR} at lower left corner of (a). Best scores marked in bold. Best viewed on screen.

(a) Bicubic (24.80, 0.6736, 0.3634) (b) RCAN [43] (28.18, 0.7863, 0.1768)

(c) DRN [9] (28.30, 0.7891, 0.1752) (d) Our Net (**28.40**, **0.7940**, 0.1622)

(e) Our GAN (25.42, 0.6889, **0.0822**) (f) Ground-Truth

Figure 5.13: Visual Comparison of SISR results on Set14 dataset with PSNR, SSIM, and LPIPS scores. I_{LR} at lower left corner of (a). Best scores marked in bold. Best viewed on screen.

5.2.4 Experimental Results on REDS dataset

This section shows our experimental results on the REDS dataset from "NTIRE 2021 Image Deblurring Challenge - Track1. Low Resolution". As Figure 5.12, 5.13 and Table 5.2 show, our proposed net achieved the highest PSNR and SSIM scores while our proposed GAN generated the most perceptually natural images.

Table 5.4: Comparison of SISR results on REDS dataset. Best scores marked in bold.

Method	**Scale**	**REDS - NTIRE 2021 Low Resolution**		
		PSNR ↑	**SSIM ↑**	**LPIPS ↓**
Bicubic	×4	24.55	0.6313	0.4855
EDSR [20]		25.26	0.6775	0.3752
RCAN [43]		25.31	0.6797	0.3775
DRN [9]		25.30	0.6791	0.3773
Our Net w/o Edge		27.78	0.7648	0.2542
Our Net w/o FeaAtt		**27.83**	0.7660	0.2550
Our Net		**27.83**	**0.7662**	0.2540
Our GAN		24.19	0.6337	**0.1489**

(a) REDS/val/X4/000/00000069.png

(b) REDS/val/X4/015/00000029.png

Figure 5.14: SISR results of our proposed methods on REDS validation data. From left to right: bicubic interpolation (with I_{LR} at the lower left corner), our net, our GAN, and ground-truth. Best viewed on screen.

(a) REDS/val/X4/021/00000049.png

(b) REDS/val/X4/023/00000079.png

Figure 5.15: SISR results of our proposed methods on REDS validation data. From left to right: bicubic interpolation (with I_{LR} at the lower left corner), our net, our GAN, and ground-truth. Best viewed on screen.

Chapter 6

Conclusion and Future Works

This thesis introduces a novel deep learning method that generates high-resolution images by increasing the quality of given images. We first divide the image quality enhancement problem into denoising, deblurring, and super-resolution and propose a deep learning model optimized for solving each task step by step.

We propose multi-scale edge detection to solve the denoising problem, which extracts high-frequency information from noisy images. This helps our deep neural network perform statistical analysis and reconstruction suitable for each region of the image. We also add feature attention modules to enable the network to determine feature maps containing more important information.

Input images of deblurring and SISR problems are generally more damaged, and the loss of information is more severe. We design a deep neural network with a U-Net structure that can analyze images at multiple scales so the model can learn the local and global contexts of the input image. Furthermore, in the training phase, we introduce a high-pass filtering loss function that compares feature maps generated from a high-pass filtering network computing the high-frequency information of the results and

ground truth images. Finally, our soft GMS masking helps the model identify which areas of the resulting image are more compromised and need to be more focused on additional training processes.

Experimental results show that our model can achieve state-of-the-art PSNR and SSIM scores compared to other learning-based methods. However, when visualizing the results, over-smoothing problems have been observed as in other PSNR-oriented methods. Adversarial training was applied to pre-trained models using a discriminator that distinguishes real and synthetic images, allowing the model to generate much more natural images.

Images generated via GAN have high LPIPS scores, while PSNR and SSIMS scores are very low. This is because the pixels are distorted when the model generates synthetic patches to convert a small-sized low-resolution image into a large-sized high-resolution image. We can overcome this problem to a certain level if the model learns enough information provided in a low-resolution image. In future research, we will study learning-based methods that achieve superior scores in both PSNR and LPIPS by enabling the model to extract sufficient information from low-resolution images.

Bibliography

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown (2018). A high-quality denoising dataset for smartphone cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1692–1700.

[2] Eirikur Agustsson and Radu Timofte (2017). Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 126–135.

[3] Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D Lawrence, and Zhenwen Dai (2019). Variational information distillation for knowledge transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9163–9171.

[4] Sungkwon An, Hyungmin Roh, and Myungjoo Kang (2020). Long-term residual blending network for blur invariant single image blind deblurring. *arXiv preprint arXiv:2007.04543*.

[5] Sefi Bell-Kligler, Assaf Shocher, and Michal Irani (2019). Blind super-resolution kernel estimation using an internal-gan. *arXiv preprint arXiv:1909.06581*.

[6] Charles A Bouman (2013). Model based image processing. *Purdue University*.

[7] Victor Cornillere, Abdelaziz Djelouah, Wang Yifan, Olga Sorkine-Hornung, and Christopher Schroers (2019). Blind image super-resolution with spatially variant degradations. *ACM Transactions on Graphics (TOG) 38*(6), 1–13.

[8] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang (2015). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence 38*(2), 295–307.

[9] Yong Guo, Jian Chen, Jingdong Wang, Qi Chen, Jiezhang Cao, Zeshuai Deng, Yanwu Xu, and Mingkui Tan (2020). Closed-loop matters: Dual regression networks for single image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5407–5416.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708.

[12] Yuanfei Huang, Jie Li, Xinbo Gao, Yanting Hu, and Wen Lu (2021). Interpretable detail-fidelity attention network for single image super-resolution. *IEEE Transactions on Image Processing 30*, 2325–2339.

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei (2016). Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pp. 694–711. Springer.

[14] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee (2016). Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1646–1654.

[15] Jakob Kruse, Carsten Rother, and Uwe Schmidt (2017). Learning to push the limits of efficient fft-based image deconvolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4586–4594.

[16] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas (2018). Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8183–8192.

[17] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang (2019). Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8878–8887.

[18] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690.

[19] Wonkyung Lee, Junghyup Lee, Dohyung Kim, and Bumsub Ham (2020). Learning with privileged information for efficient image super-resolution. In *European Conference on Computer Vision*, pp. 465–482. Springer.

[20] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee (2017). Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 136–144.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440.

[22] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee (2017, July). Deep multi-scale convolutional neural network for dynamic scene deblurring. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[23] Jinshan Pan, Yang Liu, Deqing Sun, Jimmy Ren, Ming-Ming Cheng, Jian Yang, and Jinhui Tang (2020). Image formation model guided deep image super-resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Volume 34, pp. 11807–11814.

[24] Yingxue Pang, Xin Li, Xin Jin, Yaojun Wu, Jianzhao Liu, Sen Liu, and Zhibo Chen (2020). Fan: Frequency aggregation network for real image super-resolution. *arXiv preprint arXiv:2009.14547*.

[25] Bumjun Park, Songhyun Yu, and Jechang Jeong (2019). Densely connected hierarchical network for image denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0.

[26] Mohammad Saeed Rad, Behzad Bozorgtabar, Urs-Viktor Marti, Max Basler, Hazim Kemal Ekenel, and Jean-Philippe Thiran (2019). Srobb: Targeted perceptual loss for single image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2710–2719.

[27] Olaf Ronneberger, Philipp Fischer, and Thomas Brox (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer.

[28] Taizhang Shang, Qiuju Dai, Shengchen Zhu, Tong Yang, and Yandong Guo (2020). Perceptual extreme super-resolution network with receptive field block. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 440–441.

[29] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang (2016). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1874–1883.

[30] Karen Simonyan and Andrew Zisserman (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[31] Ying Tai, Jian Yang, and Xiaoming Liu (2017). Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3147–3155.

[32] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu (2017). Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE international conference on computer vision*, pp. 4539–4547.

[33] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia (2018). Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8174–8182.

[34] Tong Tong, Gen Li, Xiejie Liu, and Qinquan Gao (2017). Image super-resolution using dense skip connections. In *Proceedings of the IEEE international conference on computer vision*, pp. 4799–4807.

[35] Ruxin Wang and Dacheng Tao (2018). Training very deep cnns for general non-blind deconvolution. *IEEE Transactions on Image Processing 27*(6), 2897–2910.

[36] Zhihao Wang, Jian Chen, and Steven CH Hoi (2020). Deep learning for image super-resolution: A survey. *IEEE transactions on pattern analysis and machine intelligence*.

[37] Norbert Wiener (1964). *Extrapolation, interpolation, and smoothing of stationary time series*. The MIT press.

[38] Wufeng Xue, Lei Zhang, Xuanqin Mou, and Alan C Bovik (2013). Gradient magnitude similarity deviation: A highly efficient perceptual image quality index. *IEEE Transactions on Image Processing 23*(2), 684–695.

[39] Kai Zhang, Luc Van Gool, and Radu Timofte (2020). Deep unfolding network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3217–3226.

[40] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang (2017). Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing 26*(7), 3142–3155.

[41] Kai Zhang, Wangmeng Zuo, and Lei Zhang (2018). Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing 27*(9), 4608–4622.

[42] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595.

[43] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu (2018). Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 286–301.

[44] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu (2018). Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2472–2481.

[45] Rui Zhao, Kin-Man Lam, and Daniel PK Lun (2019). Enhancement of a cnn-based denoiser based on spatial and spectral analysis. In *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 1124–1128. IEEE.

국문초록

본 학위 논문은 단일 영상의 품질 강화를 위한 딥러닝 기법에 대한 연구를 다룬다. 영상 품질 강화를 손상된 이미지의 잡음 제거 및 디블러링과 저해상도 이미지를 고해상도로 변환하는 초해상도 문제로 세분화한 뒤, 각각의 문제 해결에 최적화된 딥러닝 기법을 단계별로 소개한다. 특히, 손상된 영상의 특성을 효과적으로 분석하고 보다 깔끔한 고해상도 영상을 생성하기 위하여 주어진 영상을 다중 스케일로 분석하는 심층 신경망 구조를 제안하였으며, 이외에도 딥러닝 모델이 영상 내 복잡한 고주파수 영역에 대한 정보를 효과적으로 추출하고 재건할 수 있도록 돕는 기법들을 소개한다. 우리는 제안된 기법들을 SIDD, Flickr2K, DIV2K, REDS 등 데이터셋에 적용하여 기존의 딥러닝 기반 기법보다 향상된 성능을 실험적으로 증명하였다. 또한 초해상도 문제 해결을 위해 학습된 심층 신경망에 추가적인 적대적 학습을 적용함으로써 기존 딥러닝 기법들의 한계로 지적되었던 부분 평균화 문제를 극복하고 보다 자연스러운 고해상도 영상을 생성할 수 있음을 보였다.

주요어휘: 단일 영상 초해상도, 영상 강화, 딥러닝 기법, 합성곱 신경망, 영상 디블러링, 영상 잡음 제거
학번: 2015-22685

감사의 글

6년 간의 대학원을 마무리 지으며 이 학위 논문을 완성하게 되었습니다. 박사학위 기간 동안 많은 분들의 도움과 지도, 그리고 배려가 있었기에 가능했습니다. 이렇게 짧은 글로나마 고마움을 담아 모든 분들께 감사의 말씀을 전하려 합니다.

먼저 지도교수님이신 강명주 교수님께 감사의 말씀을 드립니다. 교수님께서 바쁘신 와중에도 항상 많은 관심과 지원을 아낌없이 베풀어주셨기에 좋은 연구 환경에서 모자람이 없는 풍족한 대학원 생활을 보낼 수 있었습니다. 귀중한 시간 내주셔서 논문을 심사해주신 국웅 교수님, Ernest Ryu 교수님, 이병준 교수님, 곽지훈 박사님께도 진심으로 감사의 말씀을 드립니다.

짧지 않은 대학원 생활 동안 소중한 인연을 많이 만났습니다. 그들이 있었기에 행복한 대학원 생활을 보냈고, 또한 즐겁게 마무리를 합니다. 대학원 입학 동기이자 졸업 동기인 성권이에게, 그리고 원조 112호 식구인 수진누나, 경민이형, 상연이형과 경현이에게 감사의 말을 전합니다. 한수형과 현이형, 효제를 비롯하여 계산과학과 수리과학부에서 열심히 연구 중인, 그리고 이미 졸업하고 여러 곳으로 진출하여 활약하고 계시는 우리 NCIA 연구실의 모든 선후배님들에게 고마운 마음을 전합니다. 대학원 생활 동안 정말로 감사하고 본받고 싶은 소중한 인연을 참 많이 만났습니다. 일일이 감사의 말씀을 드리고 싶지만 지면이 부족하기에 짧은 글을 빌어 여러분께 많이 감사하고 항상 응원한다는 말씀을 드립니다.

마지막으로 항상 저를 응원해주고 언제나 든든한 편이 되어주는 우리 가족들, 부모님과 누나에게 가장 감사하고 누구보다 더 사랑한다는 말씀을 드립니다.

2021년 8월
노형민