

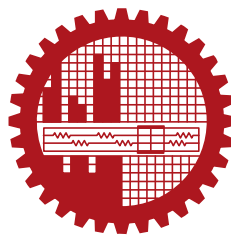
RECONSTRUCTION OF SUPER-RESOLUTION IMAGE USING WAVELET RESIDUAL CONVOLUTIONAL NEURAL NETWORKS

by

TANVIR AHMED

0419062252

MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONIC ENGINEERING



Department of Electrical and Electronic Engineering
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh

July, 2023

The thesis titled, “**RECONSTRUCTION OF SUPER-RESOLUTION IMAGE USING WAVELET RESIDUAL CONVOLUTIONAL NEURAL NETWORKS**”, submitted by **TANVIR AHMED**, Roll No.: 0419062252, Session: April 2019, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Electrical and Electronic Engineering on 26th July, 2023.

BOARD OF EXAMINERS



Dr. S. M. Mahbubur Rahman
Professor
Dept. of EEE, BUET, Dhaka

Chairman
(Supervisor)



Dr. Md. Aynal Haque
Professor and Head
Dept. of EEE, BUET, Dhaka

Member
(Ex-Officio)



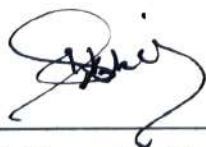
Dr. Mohammed Imamul Hassan Bhuiyan
Professor
Dept. of EEE, BUET, Dhaka

Member



Dr. Shaikh Anowarul Fattah
Professor
Dept. of EEE, BUET, Dhaka

Member



Dr. Md. Hasanul Kabir
Professor
Dept. of CSE, Islamic University of Technology, Gazipur

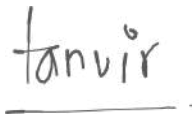
Member
(External)

Candidate's Declaration

This is to certify that the work presented in this thesis entitled, “**RECONSTRUCTION OF SUPER-RESOLUTION IMAGE USING WAVELET RESIDUAL CONVOLUTIONAL NEURAL NETWORKS**”, is the outcome of the research carried out by **TANVIR AHMED** under the supervision of Dr. S. M. Mahbubur Rahman, Professor, Department of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology (BUET), Dhaka-1205, Bangladesh.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma, or other qualifications.

Signature of the Candidate

A handwritten signature in black ink that reads "tanvir". The signature is written in a cursive style and is positioned above a horizontal line.

TANVIR AHMED

0419062252

Dedication

To my loving family.

Acknowledgement

I am immensely grateful for the invaluable support and guidance provided by my thesis supervisor, Dr. S. M. Mahbubur Rahman, Professor of the Department of Electrical and Electronic Engineering, BUET, throughout the entire process of completing this thesis. I also extend my heartfelt appreciation to Professor Rahman for his unwavering belief in my abilities and his commitment to helping me realize my academic potential. His enthusiasm for the research problem and the attention to detail he provided in reviewing and critiquing my work have truly elevated the quality of this thesis. Furthermore, I would like to express my gratitude for never giving up on me when I was going through a difficult time both academically and personally.

I would also like to express my sincere appreciation to the members of my thesis committee, Dr. Md. Aynal Haque, Dr. Mohammed Imamul Hassan Bhuiyan, Dr. Shaikh Anowarul Fattah, and Dr. Md. Hasanul Kabir for their valuable feedback and suggestion on my work.

Finally, I want to thank family members— my wife, my parents, my brother, and my parents-in-law for their unwavering encouragement and unconditional support in this journey. I also thank my friends for their constant presence in my life. Special thanks to my childhood friend Fahim Faisal Niloy his regular comments on my thesis.

Abstract

Image super-resolution is a classic low-level vision and image processing task that aims to generate a high-resolution image from its low-resolution counterpart. As of now, most of the existing methods in image super-resolution use different neural network-based learning algorithms that are usually optimized in the spatial domain. While working in the spatial domain is a perfectly sound approach and produces images with a high peak signal-to-noise (PSNR) ratio, it often leads to producing images with poor frequency domain characteristics, i.e., they lack the natural high-frequency details, resulting in super-resolved images with blurry regions. In this work, we propose a novel wavelet-based residual convolutional neural network architecture, referred to as WaveSRResNet, that learns the image features both in the spatial and wavelet domains. The WaveSRResNet is also optimized by minimizing the loss function in the said both domains in the MAE sense to further harness the power of the discrete wavelet transform. Specifically, we design the WaveSRResNet with the development of a novel wavelet residual block (WaveRB), which is capable of performing the forward and the inverse transform inside the CNN based network. This helps the model to be regularized in the wavelet domain as well, and produce super-resolution images with better high-frequency details.

Extensive experiments are carried out in order to show the effectiveness of the proposed wavelet residual block in the CNN network for its performance in the image super-resolution algorithm. The results demonstrate that the proposed WaveSRResNet outperforms recent image super-resolution methods in terms of both quantitative and perceptual metrics. On average, the WaveSRResNet achieved 4.8% higher PSNR (\uparrow) and 8% higher SSIM (\uparrow) objective scores than the most successful methods for the $8\times$ image super-resolution, which is the most challenging scenario. It also attained 9% lower perceptual score, LPIPS (\downarrow) on average for the $4\times$ up-scaling

factor. Visual outputs also indicate that the proposed WaveSRResNet is able to reconstruct high-quality, edge-preserving images at high-resolution. The outcomes of this research clearly show the effectiveness of the proposed method for the problem of image super-resolution.

List of Symbols

Scalar variables

- $\boldsymbol{x}, \boldsymbol{s}$: Discrete spatial variable (in the vertical direction)
- $\boldsymbol{y}, \boldsymbol{t}$: Discrete spatial variable (in the horizontal direction)
- $\boldsymbol{z}, \boldsymbol{c}$: Discrete spatial variable (in the depth direction)
- $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$: Discrete iteration index
- \boldsymbol{u} : Continuous variable
- \boldsymbol{C} : Number of channels

Scalar Hyperparameters

- \boldsymbol{M} : Image height in pixels
- \boldsymbol{N} : Image width in pixels
- \boldsymbol{P} : Filter height in pixels
- \boldsymbol{Q} : Filter width in pixels
- \boldsymbol{n} : Number of training samples
- \boldsymbol{m} : Mini-batch size
- \boldsymbol{r} : Convolution stride
- \boldsymbol{p} : Convolution padding length in pixels
- \boldsymbol{s} : Up-scaling factor

-
- B_W : Number of wavelet residual blocks
 - B_R : Number of residual blocks
 - ϵ : Learning rate
 - α, ρ, μ : Various algorithmic parameters

Functions

- $f(x)$: A general 1-D function
- $f(x, y)$: A general 2-D function
- $f(x, y, z)$: A general 3-D function
- I_{LR} : A low-resolution RGB image
- I_{HR} : A high-resolution RGB image
- \hat{I}_{HR} : A reconstructed super-resolution RGB image
- \mathcal{G}_s : Proposed super-resolution model, for an up-scaling factor of s
- w : Weight of a convolution layer
- b : Bias of a convolution layer
- \mathcal{L} : Loss function
- ℓ_p : p -norm of a vector
- ψ : Wavelet function
- φ : Scaling function
- T, h : Transform coefficients

Vectors

- \mathbf{a} : An arbitrary 1-D vector
- \mathbf{X} : A mini-batch containing m number of \mathbf{I}_{LR} samples
- \mathbf{Y} : A mini-batch containing m number of \mathbf{I}_{HR} samples
- $\hat{\mathbf{Y}}$: A mini-batch containing m number of $\hat{\mathbf{I}}_{HR}$ samples
- \mathbf{F} : A feature map inside the CNN model
- $\boldsymbol{\theta}$: All learnable parameters of \mathcal{G}
- $\mathbf{g}/\hat{\mathbf{g}}$: Unbiased estimator of a gradient vector

Number Sets

- \mathbb{R} : Set of all real numbers
- \mathbb{N} : Set of all natural numbers
- \mathbb{Z} : Set of all integers

Vector Spaces

- \mathbf{V}_j : A function space spanned by scaling basis functions at scale j
- \mathbf{W}_j : A function space spanned by wavelet basis functions at scale j

Operators

- \mathbb{E} : Expectation operator
- ∇ : Gradient operator

List of Abbreviations

- SR: Super-resolution
- HR: High-resolution
- LR: Low-resolution
- TV: Total variation
- WT: Wavelet transform
- DWT: Discrete wavelet transform
- FWT: Fast wavelet transform
- IDWT: Inverse discrete wavelet transform
- SWT: Stationary wavelet transform
- IB: Interpolation-based
- LB: Learning-based
- IQA: Image quality assessment
- GAN: Generative adversarial networks
- GPT: Generative pre-trained transformer
- PSNR: Peak signal-to-noise ratio
- SSIM: Structural similarity index measure
- CNN: Convolutional neural network
- DTCWT: Dual-tree complex wavelet transform

-
- WaveRB: Wavelet residual block
 - WaveSRResNet: Wavelet super-resolution residual network
 - SISR: Single image super-resolution
 - LPIPS: Learned perceptual image patch similarity
 - ANN: Artificial neural network
 - GPU: Graphics processing unit
 - PReLU: Parametric rectified linear unit
 - MAE: Mean absolute error
 - MSE: Mean squared error
 - SGD: Stochastic gradient descent
 - I.i.d.: Independent and identically distributed
 - ADAM: Adaptive momentum
 - RB: Residual block
 - UB: Up-sampling block
 - MRA: Multi-resolution analysis
 - WaveRBPath: Wavelet residual block path
 - RBPath: Residual block path
 - UBModule: Up-sampling module
 - db: Daubechies
 - RGB: Red, green, and blue

Contents

Abstract	viii
List of Symbols	xiii
List of Abbreviations	xiii
1 Introduction	1
1.1 Introduction	1
1.2 Present state of the problem	2
1.2.1 Related works	2
1.2.2 Challenges	6
1.3 Motivation	6
1.3.1 Why learning-based method?	7
1.3.2 Why regularization method?	8
1.4 Objective	10
1.5 Contributions	11
1.6 Thesis layout	11
2 CNN and DWT: A Brief Review	13
2.1 Introduction	13
2.2 CNN overview	13
2.2.1 Convolution layer	14
2.2.2 PixelShuffle layer	15
2.2.3 Activation function	16
2.3 DWT overview	16
2.3.1 Scaling function	17

2.3.2	Wavelet function	18
2.3.3	Wavelet transform in two dimensions	19
2.4	Loss function	21
2.4.1	Mean Absolute Error Loss	22
2.4.2	Mean Squared Error Loss	22
2.5	Optimization method	23
2.5.1	Stochastic Gradient Descent	23
2.5.2	Adam	25
2.6	Summary	26
3	Wavelet Super-Resolution Residual Network	27
3.1	Introduction	27
3.2	Problem formulation	27
3.3	Architecture	28
3.3.1	DWT layer	29
3.3.2	Wavelet residual block	30
3.3.3	Residual block	32
3.3.4	Upsampling block	33
3.4	Ablation study	33
3.5	WaveSRResNet loss function	36
3.6	Summary	37
4	Experiments and Results	38
4.1	Introduction	38
4.2	Dataset	38
4.3	Experimental design	39
4.4	Performance metrics	41
4.4.1	Objective metrics	41
4.4.2	Perceptual metrics	42
4.5	Methods for comparison	42
4.5.1	$\mathcal{L}_1/\mathcal{L}_2$ loss-based methods	43
4.5.2	GAN-based methods	44
4.6	Results	44

CONTENTS

4.6.1	Objective evaluation	44
4.6.2	Subjective evaluation	46
4.6.3	Perceptual evaluation	46
4.7	Summary	46
5	Conclusion	51
5.1	Concluding remarks	51
5.2	Future works	52

List of Figures

1.1	Image super-resolution model.	1
1.2	2D forward and inverse discrete wavelet transform [1].	9
2.1	PixelShuffle layer with $s = 2$ [2]	16
2.2	Scaling function subspaces.	18
2.3	2D forward and inverse fast wavelet transform.	21
3.1	Wavelet super-resolution residual network (WaveSRResNet) architecture.	29
3.2	The db2 scaling and wavelet functions [3].	30
3.3	DWT layer applied on the ‘red’ channel of the “0149.png” image from the DIV2K dataset. The input image to the DWT layer is of shape (1752, 2040).	31
3.4	Proposed wavelet residual block (WaveRB).	32
3.5	Residual block (RB).	32
3.6	Upsampling block (UB).	33
3.7	Visualization of activation maps number 29, 30, 37, and 38 inside the WaveSRResNet.	35
4.1	Results of visual comparison for the $2\times$ super-resolution on the “Monarch.png” image from the ‘Set14’ dataset. The difference image is shown for the zoomed-in sections.	48
4.2	Results of visual comparison for the $4\times$ super-resolution on the “Haruichiban-NoFukukorox4.png” image from the ‘Manga109’ dataset. The difference image is shown for the zoomed-in sections.	49

4.3 Results of visual comparison for the $8\times$ super-resolution on the “img025.png”
image from the ‘Urban100’ dataset. The difference image is shown
for the zoomed-in sections. 50

List of Tables

1.1	A brief overview of the recent methods	5
3.1	Ablation study of the proposed architecture for the $4\times$ image super-resolution	34
4.1	Description of the datasets used	39
4.2	Results of comparing methods in terms of PSNR and SSIM for the $2\times$ image super-resolution.	45
4.3	Results of comparing methods in terms of PSNR and SSIM for the $4\times$ image super-resolution.	45
4.4	Results of comparing methods in terms of PSNR and SSIM for the $8\times$ image super-resolution.	45
4.5	Results of comparing with GAN-based methods in terms of LPIPS for the $4\times$ image super-resolution.	47

Chapter 1

Introduction

1.1 Introduction

Image super-resolution is the process of increasing the resolution or size of a low-resolution image to a higher resolution or size, while preserving or enhancing its visual quality. In other words, super-resolution (SR) aims to generate a high-resolution (HR) image from a low-resolution (LR) image by estimating the missing details and information. The reconstructed image at the higher resolution is then called a super-resolution (SR) image. Image SR is crucial in image processing and computer vision

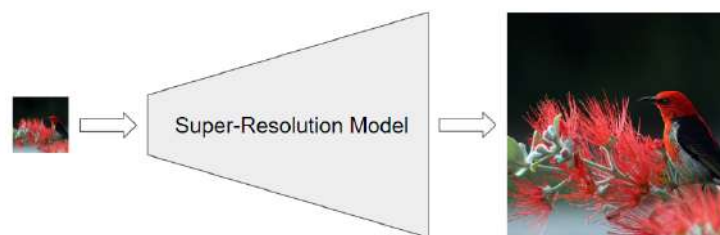


Figure 1.1: Image super-resolution model.

because it has many important real-world applications, such as those in medical imaging, surveillance, forensics, image generation, digital photography, security, remote sensing for satellite images, and computer graphics. In such scenarios, the captured image may have a limited resolution due to various factors like sensor limitations, noise, or transmission loss. Super-resolution techniques can be used to reconstruct the image with higher resolution and more details, which can be beneficial in various tasks such as object detection, recognition, and tracking. However,

the current algorithms of image super-resolution are far from perfect. Occasionally, they produce high-resolution images whose contents are blurry, the edges are over-sharpened or over-smoothed, and contain inconsistent details or textures. We address these issues and propose an elegant solution to them in this thesis. This chapter first dives into the scope of the problem of image super-resolution by outlining the present state, related works, and its challenges. Followed by the approach used and the research contributions.

1.2 Present state of the problem

There are various methods for image super-resolution, including interpolation-based methods, regularization-based methods, learning-based methods, and hybrid methods. Interpolation-based methods involve simple techniques like nearest-neighbor interpolation or bi-linear interpolation. Regularization-based methods use optimization techniques to impose constraints on the image reconstruction process. Learning-based methods, such as deep learning, use a neural network to learn the mapping between low-resolution and high-resolution images from a large dataset of paired images. However, because of the generational leap in hardware and software for learning-based algorithms in the past decade, this method has now become the most popular way to reconstruct SR images. And these methods mainly differ in their neural network architectures and in the loss functions that optimize these networks.

1.2.1 Related works

Image super-resolution can be categorized into different categories based on the underlying techniques, algorithms, and methods used for the task. A non-exhaustive classification of methods for image super-resolution can be listed as follows:

- **Interpolation-based (IB) methods:** These methods use interpolation techniques such as bicubic interpolation [4], Lanczos interpolation [5], or nearest-neighbor interpolation [6] to estimate the missing information in a low-resolution image. These are some of the most prominent interpolation-based SR models that are built upon the sampling theory. As they are deterministic mod-

els, they cannot produce or generate new information, which produces high-resolution images with jagged artifacts.

- **Regularization-based (RB) methods:** These methods use optimization techniques to impose constraints on the image reconstruction process, such as smoothness or sparsity constraints. Examples of regularization-based methods include Total Variation (TV) [7, 8] and Wavelet Transform (WT) [9, 10, 11].

The total variation is one of the most prime regularization-based methods that try to minimize the total variation of the image, based on the assumption that images with excessive and spurious detail have high total variation, where total variation is simply the integral of the absolute image gradient.

The wavelet transform has extremely good feature extraction quality and is a very useful tool for image processing. An in-depth discussion about wavelets can be found in the second chapter of this thesis. In image super-resolution, the wavelet transform is primarily used as a pre-processing tool, and very few models actually take full advantage of the wavelet transform. In [12, 13, 14, 15] works, the DWT is used only for the input images but not for the subsequent activations inside the neural networks. Very recent work in this category is [16], which deals with the high-frequency and the low-frequency components separately using the stationary wavelet transform (SWT) [17]. SWT is a variant of DWT, where the downsampling and upsampling operations are omitted to reduce the noise injected by the said operations. Although SWT is a redundant WT scheme, it overcomes the translation-variance property of the DWT significantly.

- **Learning-based (LB) methods:** These methods use machine learning and deep learning techniques to learn the mapping between low-resolution and high-resolution images from a large dataset of paired images. Compared to the traditional image super-resolution models, the learning-based methods demonstrate far better performance. LB methods try to learn the task of reconstructing super-resolution images from the data. The breakthrough LB model for ISR is [18] which used IB technique with LB techniques. It was an end-to-end CNN model capable of reconstructing HR images better than any other IB

method. However, it was a shallow network, so [19] proposed a very deep super-resolution network employing residual connections that outperformed [18]. In the residual networks regime, SRResNet [20] was the revolutionary algorithm that inspired many different algorithms, including the model proposed in this thesis. It used a residual block module in a cascading fashion to learn the LR to HR mapping. In [21], the authors tweaked the SRResNet model by getting rid of the batch-normalization layer to make the model more memory efficient and facilitate training. Later, in [22], the authors proposed a very deep residual network utilizing a method called channel attention. The authors in [23] propose a new technique in the domain of ISR, using a module called the second-order channel attention. Another particularly interesting approach in ISR is the use of generative modelings, such as GAN [24]. [20] and [25] are two of the most prominent GAN-based ISR model which can generate realistic-looking HR images. A relatively new work, [26], incorporated the transformer architecture for the problem of image super-resolution and achieved marvelous results. However, the common issue with all of the existing LB methods is that they tend to over-smooth the generated SR images [20, 27]

- **Hybrid methods:** These methods combine different techniques from the above categories to improve the performance of super-resolution. For example, some methods combine interpolation-based or regularization-based methods with deep learning techniques to achieve faster, better, and more accurate super-resolution. [12, 13, 14, 15, 16, 17] are some of the popular hybrid methods.

The following Table 1.1 gives an overview of some of the most recent and successful image super-resolution methods. In this table it is observed that the methods can be broadly categorized into two groups, namely pixel-loss-based methods and perceptual loss-based methods.

Table 1.1: A brief overview of the recent methods

Authors	Model	Year	Architecture	Loss
R. Keys [4]	Bicubic	1981	An extension of the cubic spline interpolation technique for image interpolation	Not a learning-based model
B. Lim et al. [21]	EDSR	2017	CNN residual network without the batch-normalization	Pixel loss (MAE)
Y. Zhang et al. [22]	RCAN	2018	Residual-in-residual block with skip-connection and channel attention technique	Pixel loss (MAE)
P. Liu et al. [14]	MWCNN	2019	U-net [28] like structure with DWT layers replacing the pooling operation	Pixel loss (MSE)
T. Dai et al. [23]	SAN	2019	CNN with second-order channel attention module	Pixel loss (MAE)
J. Liang et al. [26]	SwinIR	2021	Transformer with swin blocks	Pixel loss (MAE)
C. Ledig et al. [20]	SRGAN	2017	Residual CNN as the generator along with a CNN based discriminator for adversarial training	GAN and perceptual loss
X. Wang et al. [25]	ESRGAN	2018	Relativistic GAN with residual-in-residual dense block	Relativistic GAN and perceptual loss
J. W. Soh et al. [29]	NatSR	2019	GAN with naturalness prior in the low-level domain with natural manifold discrimination	GAN and naturalness loss
C. Ma et al. [30]	SPSR	2021	GAN with learnable gradient guidance	GAN and gradient loss

1.2.2 Challenges

Despite significant advances in image super-resolution in recent years, several challenges remain that must be addressed in order to improve the performance of existing techniques. The image SR is a challenging problem because of its ill-posed nature [31]. An ill-posed problem is a mathematical problem that does not have a unique solution and the mapping from input to output is non-unique or ill-conditioned. So for image SR, there can be many HR images corresponding to a single LR image. The aim of image SR is then to find an HR image with the highest perceptual quality, from the corresponding LR image.

Image SR becomes even more challenging because higher values of the most common image quality assessment (IQA) metrics such as mean squared error (MSE), peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), multi scale structural similarity index measure (MS-SSIM), michelson contrast (C_M), entropy, measure of enhancement (EME) do not necessarily always correlate with higher perceptual image quality[20]. That is why image SR is still considered an open research problem because there are still rooms for improvement in terms of generating realistic images at higher resolutions.

1.3 Motivation

We tackle the problem of reconstructing super-resolution images from the perspective of hybrid methods— a learning method, blended with a regularization-based technique. The reason for choosing a hybrid approach is very straightforward because from the related works in this field we see that approaches which combine multiple techniques tend to perform better. Moreover, in the hybrid method we have the flexibility to try different combinations of the said approaches.

To find which hybrid method will work best, we need to answer two questions: which learning-based approach and which regularization-based approach are best suited for the problem of image super-resolution. We try to find the answers to these two questions in the following paragraphs.

1.3.1 Why learning-based method?

A learning-based algorithm is a type of algorithm that can learn from data to improve its performance on a specific task, which is image super-resolution in our case.

The basic idea behind a learning-based image super-resolution method is to train it on a data set of paired images, such that it can learn to reconstruct high-resolution images with high perceptual quality when presented with new, low-resolution images. The training process typically involves adjusting the algorithm's internal parameters or weights to minimize a specific objective function or error metric (also called the loss function). The following are some of the most desirable properties of the learning-based methods suitable for this image super-resolution problem:

- should be easy to optimize using gradient-based optimization techniques
- should be easily differentiable for the back-propagation algorithm
- the architecture should facilitate a non-convex loss function optimization
- should be able to utilize hardware accelerators to make simultaneous parallel computations

Convolutional neural networks (CNN) are currently the most popular learning-based method, not only in image super-resolution but also in other computer vision and natural language processing tasks, mostly because of the advancements in hardware accelerators that help optimize them extremely efficiently. Because this hardware can perform multiple, simultaneous computations in parallel, they take advantage of very simple algorithms, such as gradient descent [32, 33, 34] or back-propagation [35, 36] to optimize very deep neural networks. As an example, one of the largest language models currently available, the GPT-3, has 175 billion parameters and reportedly took several weeks to train on a large cluster of hardware accelerators (GPUs) [37]. This would have been impossible without the accelerators.

Residual networks[38], on the the other hand, is an extremely popular deep neural network architecture with skip-connections. These skip-connections make the extremely non-convex loss landscape to a strongly convex one, that speeds up the learning process[39]. That is why–

we choose a residual convolutional neural network architecture as our core learning-based algorithm.

1.3.2 Why regularization method?

While super-resolution methods can improve the resolution of an image, they may not necessarily improve its perceptual quality. SR images may sometimes contain artifacts and blurring which can make them look unnatural while reporting a high IQA value. The most common phenomenon is the unwanted blurring in the reconstructed SR images [20, 27, 40] meaning they fail to generate fine image details at higher resolutions. In signal processing terms, this translates to the reconstructed SR images failing to match the fidelity of the high-frequency characteristics of the ground truth HR images. This often leads to SR images that are overly smooth, even though they might have a high PSNR or SSIM value. However, these overly smoothed SR images should come as no surprise, because most learning-based methods use the \mathcal{L}_2 or \mathcal{L}_1 pixel loss to optimize their networks. Even after achieving a high quantitative metric such as PSNR or SSIM, they fail to match the fidelity required at the high-resolution level. The key to producing SR images with high perceptual image quality basically then lies in faithfully generating the high-frequency components during up-scaling.

To address this issue, we need a good regularization-based technique with great high-frequency feature extraction capabilities to bolster the existing residual convolutional neural network. To this end, the following are some of the most desirable properties of the regularization-based methods suitable for our problem:

- should have a multi-resolution property, meaning it can decompose the image into different frequency bands at different levels of detail
- should have good space-frequency localization capabilities, meaning it should be able to accurately represent high-frequency and low-frequency components of an image in both space and frequency domains
- it should have great compression capabilities, meaning it should be able to represent the image information with lesser number of transform coefficients to handle the memory restrictions of the CNN

- it should be able to reconstruct images perfectly with an inverse process, while being computationally efficient, as we will need to move back and forth to the frequency domain in each CNN layer
- it should avoid redundancy

There are many multi-resolution analysis tools such as the Curvelet Transform [41, 42] and the Dual-tree Complex Wavelet Transform[43] (DTCWT), and the Discrete Wavelet Transform[3, 44] (DWT)– all of which could be a good candidate here. Although these tools satisfy the multi-resolution analysis property, because of the computational efficiency and great compactness capability, we choose the Discrete Wavelet Transform (DWT) as our regularization method.

The Wavelet Transform is basically a multi-resolution signal analysis tool that decomposes an image using wavelets while preserving both spatial and frequency information. The discrete wavelet transform (DWT) is a special case of the WT where the wavelets are discretely sampled [11]. In a 1-level 2D-DWT, the higher

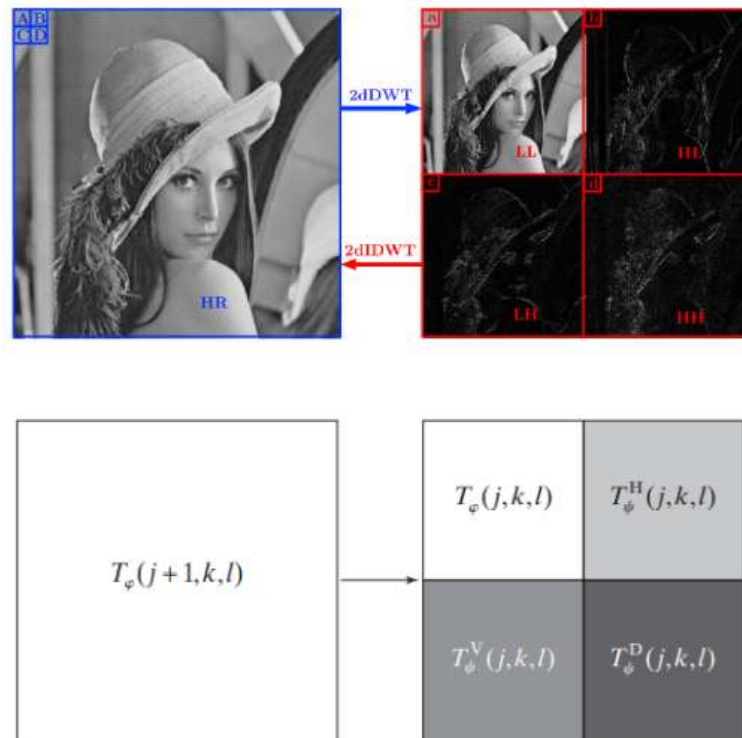


Figure 1.2: 2D forward and inverse discrete wavelet transform [1].

scale image approximation coefficients are decomposed into the lower scale image “approximation” coefficients and three sets of “detail” coefficients– horizontal, ver-

tical, and diagonal. The approximation and detail coefficients of the lower scale can be used to perfectly reconstruct the original approximation coefficients at the higher scale using a 1-level 2D-IDWT[1]. Which brings us to the following insight for the choice regularization method–

if we can properly learn high-frequency components during the reconstruction of the SR images, we can essentially use them to produce HR images from their LR counterparts just by applying the inverse transform.

The existing wavelet-based learning approaches[12, 13, 14, 15, 17] try to predict HR image 2D-DWT coefficients by feeding the networks the LR image 2D-DWT coefficients; which can be basically interpreted as a mere preprocessing tool. But, we think that applying the 2D-DWT to only the input images but not to the subsequent image feature maps inside the CNN is hindering these models from truly harnessing the power of the DWT.

1.4 Objective

The objective is to find a fast, efficient, and robust image super-resolution framework that can produce sharp, crisp, natural-looking high-resolution images with greater perceptual quality from their low-resolution counterparts. We want to design a convolutional neural network (CNN) architecture combining the discrete wavelet transform and residual connectivity. To the best of our knowledge, there is no current SR model that uses the DWT of the image activation features inside the network. Instead, the transform is only used as a pre-processing or post-processing tool.

First, a novel wavelet residual block (WaveRB) is to be designed, which helps the model learn image features in the DWT domain. Second, using these WaveRBs, a novel residual CNN architecture is to be designed. The CNN will be optimized using suitable loss functions both in the pixel domain and the transformed domain that helps it generate SR images from the natural image manifold.

1.5 Contributions

We propose a CNN-based method that is wavelet friendly, meaning that the proposed architecture will help generate HR images with better wavelet “detail” coefficients, resulting in better high-frequency texture details. In this regard, we choose the DWT as our preferred regularization because of its computational efficiency and high compaction capacity. Leveraging the DWT and the inverse DWT (IDWT) of the image features for each activation of the successive layers inside the learning model not only helps it learn the proper frequency characteristics at each convolutional block but also makes the model more robust to generalize to other datasets apart from the training set.

Our specific contributions are hence threefold:

1. We propose a novel CNN architecture, referred to as the *Wavelet Super-Resolution Residual Network* (WaveSRResNet), with the development of a novel *Wavelet Residual Blocks* (WaveRB), for the problem of image super-resolution.
2. We introduce a novel wavelet transform-based loss function for the problem of image super-resolution to guide the network to produce images with better frequency characteristics.
3. We experiment with the proposed WaveSRResNet on five standard benchmark datasets, for $2\times$, $4\times$, and $8\times$ single image super-resolution (SISR) using commonly used performance metrics like PSNR, SSIM [45], and the learned perceptual image patch similarity (LPIPS) [46]. The results show that our proposed model outperforms existing [20, 23, 22, 21, 26] methods for SISR.

1.6 Thesis layout

The rest of the thesis contains the following chapters:

- **Chapter 2** CNN and DWT: A Brief Review
- **Chapter 3** Wavelet Super-Resolution Residual Network
- **Chapter 4** Experiments and Results

- **Chapter 5** Conclusion

In Chapter 2, we briefly give the overview of the necessary theoretical concepts (CNN and DWT) needed to develop the ideas in this thesis. In Chapter 3, we introduce our methodology and proposed algorithm. In Chapter 4, we present the results of the experiments with our proposed algorithm and in Chapter 5, we conclude this thesis by mentioning the challenges faced along with the possible future extensions.

Chapter 2

CNN and DWT: A Brief Review

2.1 Introduction

In the last Chapter, we ended up choosing the hybrid approach for its flexibility and superior performance. Since our hybrid method consists of CNN as the learning-based component and DWT as the regularization component, we devote this chapter to give a brief introduction to them. Specifically in the CNN overview, we discuss convolution, pixel-shuffling, and activation functions and in the DWT overview we discuss scaling functions, wavelet functions, and wavelet transform in two dimensions. Additionally, we also introduce common loss functions and optimization techniques for image super-resolution.

2.2 CNN overview

A convolutional neural network (CNN) is a type of artificial neural network (ANN) that is used extensively in computer vision and image processing. The main building block of a CNN is a mathematical operation called convolution, which replaces the usual matrix multiplication of ANNs.

There are many desirable properties of CNNs that make them so popular for visual tasks:

- CNNs are shift-invariant, meaning it can detect the same pattern anywhere on the input image

- They provide local connectivity, as well as share parameters. So the number of trainable parameters is significantly reduced
- They are able to detect local features as well as global features because the receptive field increases as we stack multiple convolution layers
- With the GPU implementation of the CNNs, they are now more powerful than ever, as the parallel computation capabilities of GPUs enable to train of deeper CNN networks

2.2.1 Convolution layer

Mathematically, the 2-D convolution between an input $f_{in}(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{M \times N}$ and a kernel/weight/filter $\mathbf{w}(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{P \times Q}$ is defined as:

$$f_{out}(\mathbf{x}, \mathbf{y}) = f_{in} \star \mathbf{w}(\mathbf{x}, \mathbf{y}) = \sum_{s=-\frac{P-1}{2}}^{\frac{P-1}{2}} \sum_{t=-\frac{Q-1}{2}}^{\frac{Q-1}{2}} f_{in}(s, t) \mathbf{w}(\mathbf{x} - s, \mathbf{y} - t) \quad (2.1)$$

where we assume that P and Q are odd integers and s, t are dummy variables for convolution, so that we have the output $f_{out}(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{(M+P-1) \times (N+Q-1)}$.

In general, the input the convolution have multiple channels, for example an RGB image has 3 channels, one for each red, green, and blue colors. So extending the 2-D convolution operation over C_{in} channels, where $f_{in}(\mathbf{x}, \mathbf{y}, z) \in \mathbb{R}^{M \times N \times C_{in}}$ and a kernel/weight/filter $\mathbf{w}(\mathbf{x}, \mathbf{y}, z) \in \mathbb{R}^{P \times Q \times C_{in}}$, we have:

$$f_{out}(\mathbf{x}, \mathbf{y}) = f_{in} \star \mathbf{w}(\mathbf{x}, \mathbf{y}) = \sum_{k=0}^{C_{in}-1} \sum_{s=-\frac{P-1}{2}}^{\frac{P-1}{2}} \sum_{t=-\frac{Q-1}{2}}^{\frac{Q-1}{2}} f_{in}(s, t, k) \mathbf{w}(\mathbf{x} - s, \mathbf{y} - t, k) \quad (2.2)$$

where the output of the convolution over volume is still a 2-D function. For a convolution operation with C_{out} number filters, where $\{\mathbf{w}_i\}_{i=0}^{C_{out}-1}$, we stack the outputs for each convolution along the third dimension, so that the output becomes $f_{out}(\mathbf{x}, \mathbf{y}, z) \in \mathbb{R}^{(M+P-1) \times (N+Q-1) \times C_{out}}$ as follows:

$$\begin{aligned} f_{out}(\mathbf{x}, \mathbf{y}, i) &= f_{in} \star \mathbf{w}_i(\mathbf{x}, \mathbf{y}, i) \\ &= \sum_{k=0}^{C_{in}-1} \sum_{s=-\frac{P-1}{2}}^{\frac{P-1}{2}} \sum_{t=-\frac{Q-1}{2}}^{\frac{Q-1}{2}} f_{in}(s, t, k) \mathbf{w}_i(\mathbf{x} - s, \mathbf{y} - t, k) \end{aligned} \quad (2.3)$$

for $i = \{0, 1, 2, \dots, C_{out} - 1\}$.

With these tools, we are now ready to define a typical convolution layer of a CNN model:

- Input: $f_{in} \in \mathbb{R}^{M \times N \times C_{in}}$
- Number of filters in the layer: C_{out}
- Filters: $w_i \in \mathbb{R}^{P \times Q \times C_{in}}$, for $i = \{0, 1, 2, \dots, C_{out} - 1\}$
- Bias: $b_i \in \mathbb{R}$, for $i = \{0, 1, 2, \dots, C_{out} - 1\}$
- Stride: $r \in \mathbb{N}$
- Padding: $p \in \mathbb{N}$
- Output: $f_{out} \in \mathbb{R}^{\lfloor \frac{M-P+2p}{r} \rfloor + 1 \times \lfloor \frac{N-Q+2p}{r} \rfloor + 1 \times C_{out}}$

where stride is the step size for the convolution (which is generally 1), and padding is the number of additional pixels added to each four side of the image for a better control of the shape of the output.

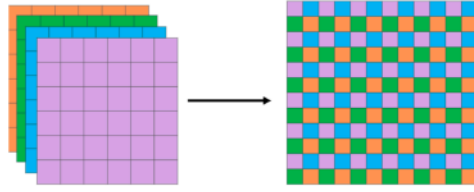
$$f_{out} \equiv Conv(f_{in}) = Conv(f_{in}; w_i, b_i, r, p) = f_{in} \star w_i + b_i \quad (2.4)$$

For $i = \{0, 1, 2, \dots, C_{out} - 1\}$. Here, the (learnable/trainable) parameters are $\{w_i, b_i\}_{i=0}^{C_{out}-1}$, which we learn via an optimization algorithm, and the hyperparameters are $\{C_{out}, r, p\}$, which we do not learn but use to control the learning process of the parameters.

2.2.2 PixelShuffle layer

The pixel shuffling layer [2] rearranges the input volume to increase the spatial dimension (height, width) at the expense of the depth dimension. So for an upscaling factor of $s \in \mathbb{N}$, the PixelShuffle layer is just the transformation:

$$PixelShuffle_s(\cdot) : \mathbb{R}^{M \times N \times (C_{in} \times s^2)} \rightarrow \mathbb{R}^{(M \times s) \times (N \times s) \times C_{in}} \quad (2.5)$$

Figure 2.1: PixelShuffle layer with $s = 2$ [2]

2.2.3 Activation function

In artificial neural networks, each output from a layer has to pass through a certain function (usually a non-linear function), which are called the activation functions, and the output coming from the activation functions are hence called “activation map” or just “activation” for short. Some common activation functions are discussed in the following discussion.

Parametric Rectified Linear Unit (PReLU)

The element-wise parametric rectified linear unit is defined as [47]:

$$PReLU(u) = \max(0, u) + \alpha * \min(u) \quad (2.6)$$

or

$$PReLU(u) = \begin{cases} u & \text{if } u \geq 0 \\ \alpha u & \text{otherwise} \end{cases} \quad (2.7)$$

where α is a learnable parameter.

Hyperbolic Tangent (Tanh)

The element-wise hyperbolic tangent is defined as:

$$Tanh(u) \equiv \tanh(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}} \quad (2.8)$$

2.3 DWT overview

The discrete wavelet transform (DWT) is a popular multi-resolution signal analysis tool that is concerned with the representation and analysis of those signals (in our case, images) at multiple resolutions [44]. At the heart of a wavelet transform are two main components: a scaling function and a wavelet function. The scaling function

creates a series of approximations of a signal or an image that differ by a factor of 2 in resolution, while the wavelet function encodes the differences or errors between two adjacent level approximations. The discrete wavelet transform represents a function or an image as a linear combination of the wavelets and the scaling functions. The Daubechies [3] and the Haar [48] are some of the common bases for the DWT. The following discussions are excerpts from the [1], which shows the development of the DWT.

2.3.1 Scaling function

If $\varphi(\mathbf{x})$ is a real, square-integrable function (called the *father scaling function*), we consider the set of basis functions composed of all integer translations and binary scalings of the father scaling functions $\{\varphi_{j,k}(\mathbf{x}) | j, k \in \mathbb{Z}\}$ where

$$\varphi_{j,k}(\mathbf{x}) = 2^{j/2} \varphi(2^j \mathbf{x} - \mathbf{k}) \quad (2.9)$$

Here, the integer \mathbf{k} determines the position of $\varphi_{j,k}(\mathbf{x})$ along the \mathbf{x} -axis, and the integer j determines its shape (both width and amplitude). If the function space spanned by bases $\{\varphi_{j,k}(\mathbf{x})\}$ for $j = j_0$ is denoted as \mathbf{V}_{j_0} , then increasing j_0 increases the spanned functions in \mathbf{V}_{j_0} , or in other words, functions with smaller variations and finer details can also be represented by the bases *in addition* to \mathbf{V}_{j_0} .

To be a valid scaling function for the discrete wavelet transforms, it has to obey the following four fundamental requirements of multi-resolution analysis (MRA) [44]:

1. it is orthogonal to its integer translates
2. function space spanned by it at low scales are contained within those spanned at higher scales
3. the only function representable at every scale is $\mathbf{f}(\mathbf{x}) = \mathbf{0}$
4. as $j \rightarrow \infty$, all measurable, real, square-integrable functions can be represented by their linear combinations

Under these conditions, $\varphi(\mathbf{x})$ can be represented as a linear combination of the

double-resolution copies of itself (also called the refinement of the dilation equation):

$$\varphi(x) = \sum_{k \in \mathbb{Z}} h_\varphi(k) \sqrt{2} \varphi(2x - k) \quad (2.10)$$

where the $\{h_\varphi(k) | k = 0, 1, 2, \dots\}$ are the expansion coefficients or sometimes called the scaling function coefficients. If the scaling functions are orthonormal, then the coefficients can be calculated by the following inner product:

$$h_\varphi(x) = \langle \varphi(x), \sqrt{2} \varphi(2x - k) \rangle \quad (2.11)$$

2.3.2 Wavelet function

For any father scaling function that meets the MRA requirements from the previous section, there exists a mother wavelet function $\psi(x)$, whose integer translations and binary scalings $\{\psi_{j,k}(x) | j, k \in \mathbb{Z}\}$ span the differences between any two immediate scaling spaces spanned by the scaling functions, where

$$\psi_{j,k}(x) = 2^{j/2} \psi(2^j x - k) \quad (2.12)$$

If \mathbf{W}_{j_0} is the function space spanned by the wavelet functions at scale $j = j_0$, i.e. $\{\psi_{j_0,k} | k \in \mathbb{Z}\}$, then

$$\mathbf{V}_{j_0+1} = \mathbf{V}_{j_0} \oplus \mathbf{W}_{j_0} \quad (2.13)$$

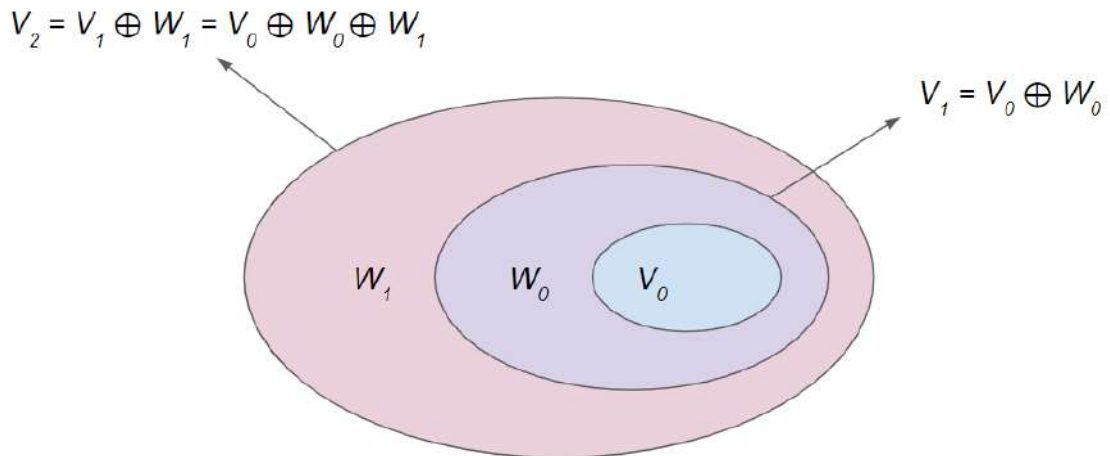


Figure 2.2: Scaling function subspaces.

where \oplus is the union of function spaces. In $\mathbf{V}_{j_0 + 1}$, the orthogonal component of \mathbf{V}_{j_0} is \mathbf{W}_{j_0} , and the scaling functions that are the basis of \mathbf{V}_{j_0} are orthogonal to the wavelet functions that are the basis of \mathbf{W}_{j_0} .

Since the wavelet function spaces at a certain scale reside inside the higher scaling function space, the wavelet function $\psi(\mathbf{x})$, like the scaling function, can be written as a weighted sum of shifted, double-resolution scaling functions as:

$$\psi(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}} \mathbf{h}_\psi(\mathbf{k}) \sqrt{2} \varphi(2\mathbf{x} - \mathbf{k}) \quad (2.14)$$

where the $\{\mathbf{h}_\psi(\mathbf{k}) | \mathbf{k} = \mathbf{0}, \mathbf{1}, \mathbf{2}, \dots\}$ are the wavelet coefficients, sometimes called the wavelet function coefficients. It can be shown that the relationship between the scaling and the wavelet coefficients is given by the following equation:

$$\mathbf{h}_\psi(\mathbf{k}) = (-1)^k \mathbf{h}_\varphi(1 - k) \quad (2.15)$$

2.3.3 Wavelet transform in two dimensions

The 1-D wavelet transforms can easily be extended to 2-D functions like images. In 2-D, a 2-D scaling function $\varphi(\mathbf{x}, \mathbf{y})$, and three 2-D wavelets, $\psi^H(\mathbf{x}, \mathbf{y})$, $\psi^V(\mathbf{x}, \mathbf{y})$, and $\psi^D(\mathbf{x}, \mathbf{y})$, are required. Each is the product of two 1-D functions, hence the resulting 2-D functions are separable.

$$\begin{aligned} \varphi(\mathbf{x}, \mathbf{y}) &= \varphi(\mathbf{x})\varphi(\mathbf{y}) \\ \psi^H(\mathbf{x}, \mathbf{y}) &= \psi(\mathbf{x})\varphi(\mathbf{y}) \\ \psi^V(\mathbf{x}, \mathbf{y}) &= \varphi(\mathbf{x})\psi(\mathbf{y}) \\ \psi^D(\mathbf{x}, \mathbf{y}) &= \psi(\mathbf{x})\psi(\mathbf{y}) \end{aligned} \quad (2.16)$$

The three “directionally sensitive” wavelets measure functional variations (image intensity variation in our case) along different directions: ψ^H measures variations along columns, ψ^V , captures the variations along rows, and ψ^D corresponds to variations along diagonals.

If we have separable 2-D scaling and wavelet functions, an extension of the 1-D DWT to 2-D DWT is straightforward. First, we define the scaled and translated basis functions:

$$\begin{aligned} \varphi_{j,s,t}(\mathbf{x}, \mathbf{y}) &= 2^{j/2} \varphi(2^j \mathbf{x} - \mathbf{s}, 2^j \mathbf{y} - \mathbf{t}) \\ \psi_{j,s,t}^i(\mathbf{x}, \mathbf{y}) &= 2^{j/2} \psi^i(2^j \mathbf{x} - \mathbf{s}, 2^j \mathbf{y} - \mathbf{t}), \quad i = \{H, V, D\} \end{aligned} \quad (2.17)$$

Here, the i is just a superscript shorthand to specify the directionality of the wavelets. The discrete wavelet transform of a 2-D image $\mathbf{f}(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{M \times N}$ is

then:

$$T_\varphi(j_0, s, t) = \langle f(x, y), \varphi_{j_0, s, t}(x, y) \rangle = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \varphi_{j_0, s, t}(x, y) \quad (2.18)$$

$$T_\psi^i(j, s, t) = \langle f(x, y), \psi_{j, s, t}^i(x, y) \rangle = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \psi_{j, s, t}^i(x, y) \quad (2.19)$$

Here, $T_\varphi(j_0, s, t)$ are the approximate coefficients at an arbitrary scale j_0 , and $T_\psi^i(j, s, t)$, $i = \{H, V, D\}$ are the horizontal, vertical, and diagonal detail coefficients for $j \geq j_0$, for a 2-D function $f(x, y)$. We usually set $j_0 = 0$, $N = M = 2^J$ so that $j = 0, 1, 2, \dots, J - 1$, and $s = t = 0, 1, 2, \dots, 2^j - 1$. For convenience, let us define the $DWT(\cdot)$ operator as:

$$\begin{aligned} DWT(f) &\equiv DWT_{\varphi, \psi}(f) \\ &= \{ \langle f, \varphi_{j_0, s, t} \rangle, \langle f, \psi_{j, s, t}^H \rangle, \langle f, \psi_{j, s, t}^V \rangle, \langle f, \psi_{j, s, t}^D \rangle \} \\ &= \{ T_\varphi, T_\psi^H, T_\psi^V, T_\psi^D \} \end{aligned} \quad (2.20)$$

Now, given these coefficients, T_φ and T_ψ^i , the original $f(x, y)$ is obtained by the inverse discrete wavelet transform as:

$$\begin{aligned} f(x, y) &= \frac{1}{\sqrt{MN}} \sum_s \sum_t T_\varphi(j_0, s, t) \varphi_{j_0, s, t}(x, y) + \\ &\quad \frac{1}{\sqrt{MN}} \sum_{j=j_0}^{\infty} \sum_s \sum_t T_\psi^H(j, s, t) \psi_{j, s, t}^H(x, y) + \\ &\quad \frac{1}{\sqrt{MN}} \sum_{j=j_0}^{\infty} \sum_s \sum_t T_\psi^V(j, s, t) \psi_{j, s, t}^V(x, y) + \\ &\quad \frac{1}{\sqrt{MN}} \sum_{j=j_0}^{\infty} \sum_s \sum_t T_\psi^D(j, s, t) \psi_{j, s, t}^D(x, y) \end{aligned} \quad (2.21)$$

or more compactly:

$$\begin{aligned} f(x, y) &= \frac{1}{\sqrt{MN}} \sum_s \sum_t T_\varphi(j_0, s, t) \varphi_{j_0, s, t}(x, y) \\ &\quad + \frac{1}{\sqrt{MN}} \sum_{i=H, V, D} \sum_{j=j_0}^{\infty} \sum_s \sum_t T_\psi^i(j, s, t) \psi_{j, s, t}^i(x, y) \end{aligned} \quad (2.22)$$

For convenience, let us also define the $IDWT(\cdot)$ operator as:

$$\begin{aligned} IDWT_{\varphi, \psi}(\{T_\varphi, T_\psi^H, T_\psi^V, T_\psi^D\}) \\ &\equiv \frac{1}{\sqrt{MN}} \sum_s \sum_t T_\varphi(j_0, s, t) \varphi_{j_0, s, t}(x, y) \\ &\quad + \frac{1}{\sqrt{MN}} \sum_{i=H, V, D} \sum_{j=j_0}^{\infty} \sum_s \sum_t T_\psi^i(j, s, t) \psi_{j, s, t}^i(x, y) \end{aligned} \quad (2.23)$$

Similar to the 1-D case, the 2-D discrete wavelet transform (DWT) and the inverse discrete wavelet transform (IDWT) can be implemented using digital filters, downsamplers, and upsamplers. If the 2-D scaling and wavelet functions are separable, we can just take the 1-D FWT of the rows of $\mathbf{f}(\mathbf{x}, \mathbf{y})$, followed by the 1-D FWT of the resulting columns, and mirror the operation for the synthesis filter banks, using the 1-D IFWT operation.

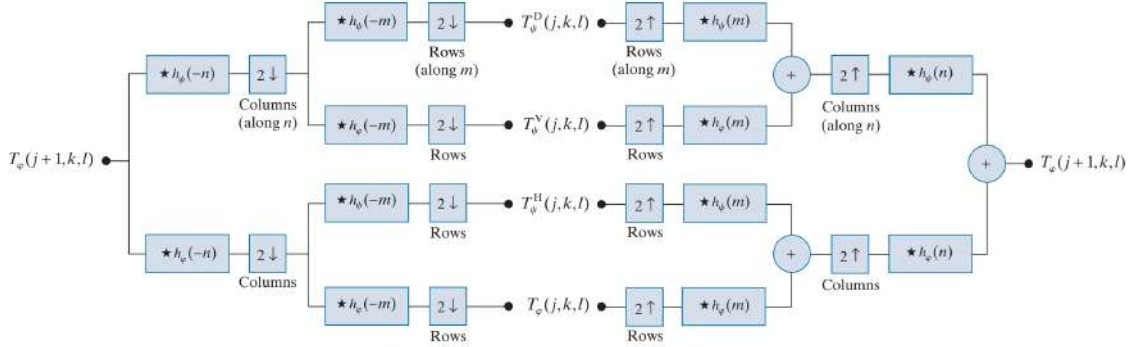


Figure 2.3: 2D forward and inverse fast wavelet transform.

2.4 Loss function

In deep learning, a loss function is a measure of how well the neural network model is performing on the given set of samples, on average. It gives an idea of whether the predicted output of the model matches the actual output. Specifically, it is a mathematical function that takes in the predicted output of the network and the actual output as two arguments and outputs a scalar value that represents the difference between the two.

The goal of a learning-based model is to minimize this loss function during the training (or learning) process. This is accomplished by adjusting the model's (learnable/trainable) parameters, such as weights and biases, in order to decrease the loss across all samples in the available dataset. Depending on the type of problem the neural network is trying to solve, the loss function may vary. In the following discussion, we will cover the two most common regression loss functions, namely the \mathcal{L}_1 and the \mathcal{L}_2 losses, also known as the ‘‘Mean Absolute Error (MAE)’’ and the ‘‘Mean Squared Error (MSE)’’, respectively. Before discussing in detail, we define–

$\mathbf{Y} = \{\mathbf{I}_{HR}^{(i)}\}_{i=1}^m$ as the actual outputs and $\hat{\mathbf{Y}} = \{\hat{\mathbf{I}}_{HR}^{(i)}\}_{i=1}^m$ as the predicted outputs of the model, both corresponding to the inputs $\mathbf{X} = \{\mathbf{I}_{LR}^{(i)}\}_{i=1}^m$.

2.4.1 Mean Absolute Error Loss

The mean absolute error is defined as:

$$MAE(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{m} \sum_{i=1}^m |\hat{\mathbf{I}}_{HR}^{(i)} - \mathbf{I}_{HR}^{(i)}| \quad (2.24)$$

In mathematics, a p -norm of a d -dimensional vector $\mathbf{a} = [\mathbf{a}^{(1)} \mathbf{a}^{(2)} \dots \mathbf{a}^{(d)}]^T$, $\mathbf{a} \in \mathbb{R}^d$, ℓ_p -norm, is defined as:

$$\|\mathbf{a}\|_p := \left(\sum_{i=1}^d |\mathbf{a}^{(i)}|^p \right)^{1/p} \quad (2.25)$$

If $\hat{\mathbf{Y}} = [\hat{\mathbf{I}}_{HR}^{(1)} \hat{\mathbf{I}}_{HR}^{(2)} \dots \hat{\mathbf{I}}_{HR}^{(m)}]^T$ and $\mathbf{Y} = [\mathbf{I}_{HR}^{(1)} \mathbf{I}_{HR}^{(2)} \dots \mathbf{I}_{HR}^{(m)}]^T$, the MAE is then the scaled ℓ_1 -norm of the vector $\hat{\mathbf{Y}} - \mathbf{Y}$:

$$MAE(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{m} \|\hat{\mathbf{Y}} - \mathbf{Y}\|_1 \quad (2.26)$$

If the data samples are independent and identically distributed and $m \rightarrow \infty$, then from the law of large numbers[49], we write \mathcal{L}_{MAE} or \mathcal{L}_1 expression in terms of the expectation value and MAE as:

$$\begin{aligned} \mathcal{L}_1(\hat{\mathbf{Y}}, \mathbf{Y}) &\equiv \lim_{m \rightarrow \infty} MAE(\hat{\mathbf{Y}}, \mathbf{Y}) \\ &= \lim_{m \rightarrow \infty} \frac{1}{m} \|\hat{\mathbf{Y}} - \mathbf{Y}\|_1 \\ &= \mathbb{E}_{\mathbf{X}, \mathbf{Y}} \|\hat{\mathbf{Y}} - \mathbf{Y}\|_1 \end{aligned} \quad (2.27)$$

2.4.2 Mean Squared Error Loss

The mean squared error is defined as:

$$MSE(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{m} \sum_{i=1}^m (\hat{\mathbf{I}}_{HR}^{(i)} - \mathbf{I}_{HR}^{(i)})^2 \quad (2.28)$$

Using a similar argument as MAE, the MSE is just the scaled ℓ_2 -norm of the vector $\hat{\mathbf{Y}} - \mathbf{Y}$:

$$MSE(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{m} \|\hat{\mathbf{Y}} - \mathbf{Y}\|_2^2 \quad (2.29)$$

And from the law of large numbers, we write \mathcal{L}_{MSE} or \mathcal{L}_2 expression in terms of the expectation value and MSE as:

$$\begin{aligned}\mathcal{L}_2(\hat{Y}, Y) &\equiv \lim_{m \rightarrow \infty} MSE(\hat{Y}, Y) \\ &= \lim_{m \rightarrow \infty} \frac{1}{m} \|\hat{Y} - Y\|_2^2 \\ &= \mathbb{E}_{X, Y} \|\hat{Y} - Y\|_2^2\end{aligned}\tag{2.30}$$

2.5 Optimization method

The goal of optimizing a neural network is to find the parameters θ for the network that significantly reduces the loss function \mathcal{L} evaluated on the entire training dataset.

We can optimize a neural network using only a single example (called stochastic), or the whole training set (called batch). Most deep learning optimization algorithms fall somewhere in between, using more than one sample but less than the whole training set, which is now commonly referred to as the minibatch method. They also require computing the gradient of the loss function, $\mathcal{L}(\hat{Y}, Y; \theta, X)$ with respect to the parameters, θ . If the samples of the entire training set, $\{I_{LR}^{(i)}\}_{i=1}^n$ are i.i.d., by sampling a minibatch of size m , it can be shown that computing the gradient of the loss with respect to the parameters for that sampled minibatch, is an unbiased estimator of the exact gradient of the data. We define

$$g \equiv \frac{1}{m} \nabla_{\theta} \mathcal{L}(\hat{Y}^{[j]}, Y^{[j]}; \theta, X^{[j]})\tag{2.31}$$

where \mathcal{L} is the loss calculated for the j 'th minibatch: $\{X^{[j]}, Y^{[j]}\}$.

We will now discuss two of the most common minibatch, gradient-based, deep learning optimization algorithms: SGD and Adam.

2.5.1 Stochastic Gradient Descent

Stochastic gradient descent (SGD) is one of the foundational optimization algorithms used in training machine learning models, particularly in the context of neural networks and deep learning. The SGD is a variant of the traditional gradient descent algorithm and is specifically designed to handle large datasets efficiently.

In traditional gradient descent, the algorithm computes the gradients of the loss function with respect to all training examples in the dataset and then takes a step

towards the minimum of the loss surface. This process requires calculating the average gradient over the entire dataset, which can be computationally expensive for large datasets. Stochastic Gradient Descent takes a different approach. Instead of computing the average gradient over the entire dataset, it randomly selects a single data point (or a small batch of data points) at each iteration and calculates the gradient only with respect to that specific data point or batch. This stochastic nature introduces randomness into the optimization process but allows the algorithm to update the model parameters more frequently and efficiently.

Algorithm 1 The stochastic gradient descent (SGD) optimization

Require: Learning rate ϵ_k **Require:** Initial parameter θ **while** stopping criterion not met **do** Sample a mini-batch j of m examples from the training set $\mathbf{X}^{[j]} = \{I_{LR}^{(i)}\}_{i=1}^m$
 with corresponding targets $\mathbf{Y}^{[j]} = \{I_{HR}^{(i)}\}_{i=1}^m$ Compute gradient estimate $\hat{\mathbf{g}} \leftarrow \frac{1}{m} \nabla_{\theta} \mathcal{L}(\hat{\mathbf{Y}}^{[j]}, \mathbf{Y}^{[j]}; \theta, \mathbf{X}^{[j]})$ Apply update: $\theta \leftarrow \theta - \epsilon_k \hat{\mathbf{g}}$ **end while**

An important parameter for the SGD algorithm is the learning rate. Practically, we gradually decrease the learning rate as the learning progresses, so we denote the learning rate at iteration k as ϵ_k .

A sufficient condition to guarantee the convergence of SGD is that

$$\sum_{k=1}^{\infty} \epsilon_k = \infty \tag{2.32}$$

and

$$\sum_{k=1}^{\infty} \epsilon_k^2 < \infty \tag{2.33}$$

It is worth mentioning that the gradient estimator introduces a source of noise by randomly sampling a minibatch, hence the gradient does not vanish when we reach at the minimum.

2.5.2 Adam

In optimizing neural networks, the learning rate is probably one of the most difficult hyperparameters because it significantly impacts the network’s performance. Recently, researchers have introduced optimization algorithms that adapt the learning rate of the model parameters. The name “ADAM”, is derived from “adaptive moments”. It combines the concepts of two other optimization algorithms, AdaGrad [50] and RMSprop [50], to achieve efficient and adaptive learning rates for each parameter during the training process. Adam not only incorporates both first and second-order moments of the gradient but also corrects the bias in their estimates, to efficiently update the parameters [51]. This approach helps to converge faster and handle sparse gradients more effectively, making it well-suited for a wide range of deep learning tasks.

Algorithm 2 The Adam optimization

Require: Step size ϵ .

Require: Exponential decay rates for moment estimates, ρ_1 and ρ_2 in $[0, 1)$.

Require: Small constant δ used for numerical stabilization.

Require: Initial parameter θ

Initialize 1st and 2nd moment variables $\mu_1 = \mathbf{0}$, $\mu_2 = \mathbf{0}$

Initialize time step $t = 0$

while e stopping criterion not met **do**

 Sample a mini-batch j of m examples from the training set $\mathbf{X}^{[j]} = \{I_{LR}^{(i)}\}_{i=1}^m$
with corresponding targets $\mathbf{Y}^{[j]} = \{I_{HR}^{(i)}\}_{i=1}^m$.

 Compute gradient estimate $\mathbf{g} \leftarrow \frac{1}{m} \nabla_{\theta} \mathcal{L}(\hat{\mathbf{Y}}^{[j]}, \mathbf{Y}^{[j]}; \theta, \mathbf{X}^{[j]})$

 Update biased first moment estimate $\mu_1 \leftarrow \rho_1 \mu_1 + (1 - \rho_1) \mathbf{g}$

 Update biased second moment estimate $\mu_2 \leftarrow \rho_2 \mu_2 + (1 - \rho_2) \mathbf{g} \odot \mathbf{g}$

 Correct bias in first moment: $\hat{\mu}_1 \leftarrow \frac{\mu_1}{1 - \rho_1^t}$

 Correct bias in second moment: $\hat{\mu}_2 \leftarrow \frac{\mu_2}{1 - \rho_2^t}$

 Compute update: $\Delta \theta = -\epsilon \frac{\hat{\mu}_1}{\sqrt{\hat{\mu}_2 + \delta}}$

 Apply update: $\theta \leftarrow \theta + \Delta \theta$

end while

2.6 Summary

In this chapter, we discussed the preliminary concepts of CNN, the Discrete Wavelet Transform, loss functions, and optimization methods for learning-based architectures. In the next chapter, we introduce Wavelet Super-Resolution Residual Network and define its various components.

Chapter 3

Wavelet Super-Resolution Residual Network

3.1 Introduction

Using the preliminaries from the previous Chapter, first, we formulate the problem of image super-resolution and then develop the architecture of the proposed WaveS-RResNet. Specifically, We discuss each component of the network in depth with mathematical reasoning. Finally, we conclude this chapter with the formulation of a compound loss function comprising of the regular pixel loss, SSIM loss, and the discrete wavelet transform loss that can optimize our proposed image super-resolution network.

3.2 Problem formulation

We aim to find the optimum super-resolution model \mathcal{G} , denoted from here on as \mathcal{G}^* , which can successfully reconstruct high-resolution images from their low-resolution counterpart, given that the generated high-resolution images are of greater perceptual quality:

$$\hat{I}_{HR} = \mathcal{G}^*(I_{LR}) \quad (3.1)$$

Where \mathcal{G}^* is computed by solving the following optimization problem:

$$\mathcal{G}^* = \underset{\theta}{\operatorname{arg\,min}} \mathcal{L}(\hat{Y}, T; \theta, X) \quad (3.2)$$

With this formulation, we start with the model architecture in the following section.

3.3 Architecture

Our WaveSRResNet model architecture is adopted from a very popular structure for CNN-based super-resolution models, [20], as depicted in the figure 3.1. The key idea is to infuse the “wavelet-based” regularization method to the existing “learning-based” image super-resolution models, which is implemented in the wavelet residual blocks (WaveRB). The wavelet residual blocks are able to decompose the incoming feature map to its DWT components and are also able to synthesize the outgoing feature maps. These wavelet residual blocks not only facilitate the regularization process but also makes the images sharper and edge-preserving.

Using wavelet residual blocks (WaveRB), residual blocks (RB), and upsampling blocks (UB); we design the WaveSRResNet following a post-upsampling framework, as described in [52]. First, a low-resolution input RGB image (of shape $M \times N \times 3$) to the model is passed through a 2D convolution layer (using **64** number of 3×3 filters), followed by a PReLU layer. The feature maps coming from the last layer is then passed through two distinct learning-paths, namely the wavelet residual block path (WaveRBPath, consisting of **16** number of cascading WaveRB) and the residual block path (RBPath, consisting of **16** number of cascading RB). The key idea is that, by doing so, we let the model learn the image feature maps both in the transform domain (in the WaveRBPath) and the spatial domain (in the RBPath).

$$\begin{aligned} \mathbf{WaveRBPath}_{B_w} \equiv & \mathbf{WaveRB}_{B_w} \\ & \circ \dots \circ \mathbf{WaveRB}_3 \circ \mathbf{WaveRB}_2 \circ \mathbf{WaveRB}_1(\cdot) \end{aligned} \quad (3.3)$$

$$\mathbf{RBPath}_{B_R} \equiv \mathbf{RB}_{B_R} \circ \dots \circ \mathbf{RB}_3 \circ \mathbf{RB}_2 \circ \mathbf{RB}_1(\cdot) \quad (3.4)$$

A global skip connection is also used, following the literature [20]. The output from these three paths is added together and passed onto the UBs. As each UB increases the spatial height and width by a factor of **2**; the total number of cascading UBs in the model is dependent on the up-scaling factor, \mathbf{s} . Consequently, for the $2\times$, $4\times$, and $8\times$ super-resolution models, namely \mathcal{G}_2 , \mathcal{G}_4 , and \mathcal{G}_8 ; one, two, and three UBs are connected in series, respectively. For any up-scaling factor \mathbf{s} (where $\mathbf{s} = 2^k$, for

$k \in \mathbb{N}$), the upsampling block module (UBModule) is then:

$$UBModule_s \equiv UB_{\log_2 s} \circ UB_{\log_2 s-1} \circ UB_{\log_2 s-2} \dots \circ UB_3 \circ UB_2 \circ UB_1(\cdot) \quad (3.5)$$

The output of which is finally passed through a final 2D convolution layer (with three 3×3 filters) and a hyperbolic tangent layer [53], to finally produce a super-resolution (SR) image (of shape $sM \times sN \times 3$). In general, a WaveSRResNet

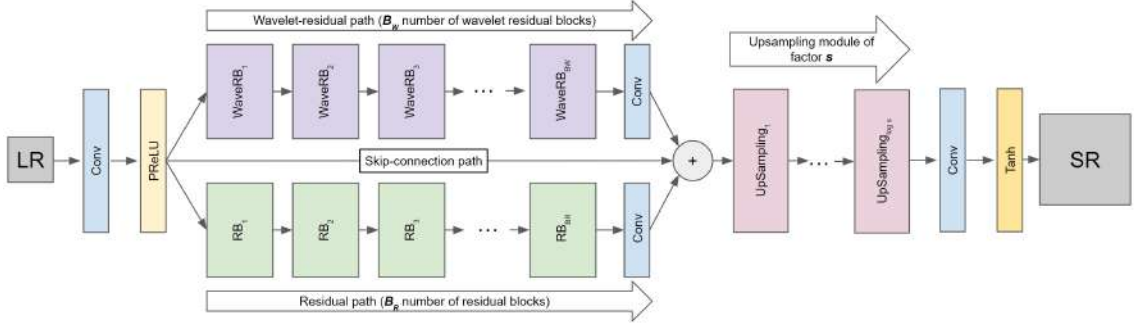


Figure 3.1: Wavelet super-resolution residual network (WaveSRResNet) architecture.

with upscaling factor s (where $s = 2^k$, for $k \in \mathbb{N}$), with B_w and B_R number of WaveRBs and RBs in the wavelet residual path and the residual paths respectively can be written as (figure 3.1):

$$\begin{aligned} \mathcal{G}_s \equiv & \text{Tanh} \circ \text{Conv} \circ UBModule_s \\ & \circ (\text{Conv} \circ \text{WaveRBP}_{B_w} + \text{Conv} \circ \text{RBP}_{B_R} + 1) \\ & \circ \text{PReLU} \circ \text{Conv}(\cdot) \end{aligned} \quad (3.6)$$

In the following subsections, we describe the DWT layer, wavelet residual block, residual block, and upsampling block in detail.

3.3.1 DWT layer

We define a 1-level 2D discrete wavelet forward transform operation using the Daubechies-2 (db2) [3] scaling ($db2_\varphi$) and wavelet ($db2_\psi$) function as:

$$DWT \equiv DWT_{db2_\varphi, db2_\psi}(\cdot) \quad (3.7)$$

and similarly, a 1-level 2D discrete wavelet inverse transform operation using the db2 scaling and wavelet function as:

$$IDWT \equiv IDWT_{db2_\varphi, db2_\psi}(\cdot) \quad (3.8)$$

In this thesis, only the db2 wavelet was used, because of the shorter filter size (4). The only shorter possible wavelet and scaling function is the Haar [48] (2), which is not suitable for natural images. The time complexity of a convolutional layer is $O(P \cdot M \cdot C_{in}^2)$ [54], so to keep the model lightweight and fast the shortest length Daubechies wavelet and scaling functions, the Daubechies-2 (db2) were used. The 1-D (db2) scaling ($db2_\varphi$) and wavelet ($db2_\psi$) function is shown in figure 3.2.

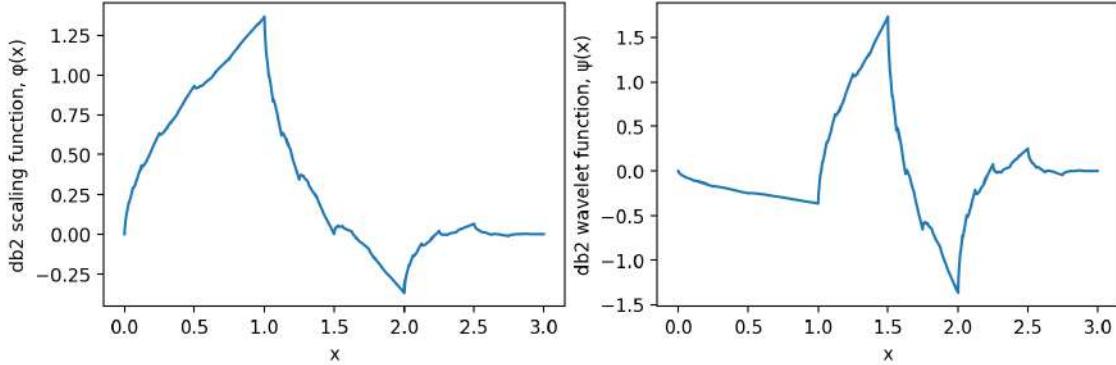


Figure 3.2: The db2 scaling and wavelet functions [3].

The DWT layer splits each channel of the input feature map to its approximate, vertical, horizontal, and diagonal coefficients and stacks them as: $\{T_\varphi, T_\psi^H, T_\psi^V, T_\psi^D\}$ (figure 3.3).

3.3.2 Wavelet residual block

Deep CNN networks are the foundation of computer vision and image processing tasks. CNNs are great for visual tasks because the convolutional layers inside extract features gradually. The shallow networks learn the simple shapes such as lines and edges, while the deeper networks learn more complicated image features existing in an object [55].

The forward and inverse DWT layers are capable of decreasing and increasing the resolution of an input image by a factor of 2. We propose a novel WaveRB which basically aids a mini super-resolution action at every convolutional step. Since the deeper the convolutional layers, the more the features learned are complicated. Hence, the WaveRB is used in parallel with the residual path in successive layers. This helps the model become more robust and generate better-quality super-resolution images.

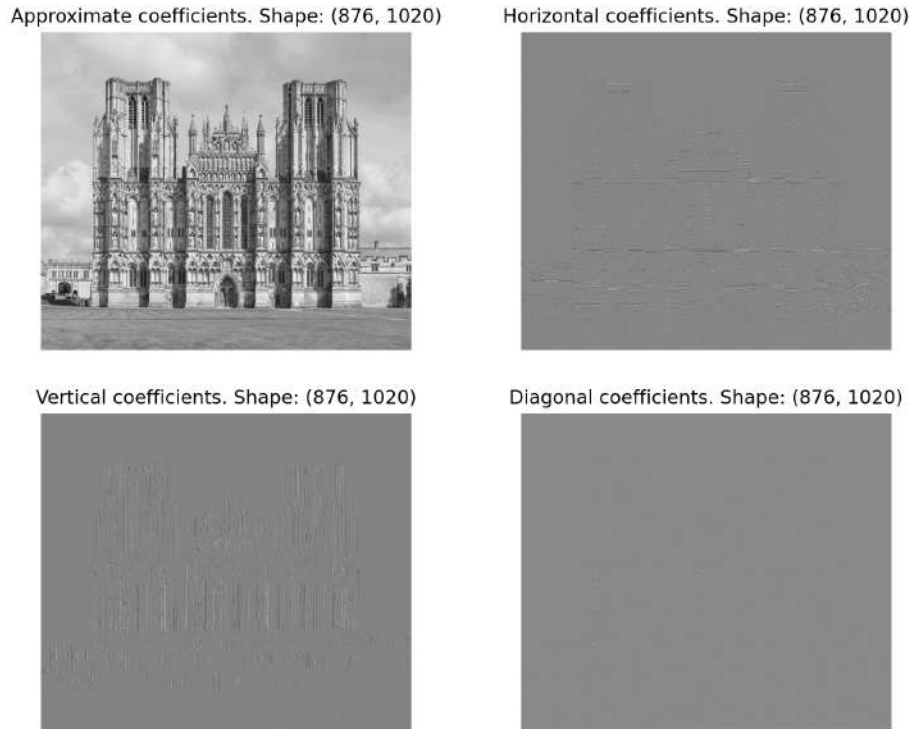


Figure 3.3: DWT layer applied on the ‘red’ channel of the “0149.png” image from the DIV2K dataset. The input image to the DWT layer is of shape $(1752, 2040)$.

Each WaveRB is capable of decomposing the input features to the block into the 2D DWT domain using the db2 wavelet and scaling functions, as shown in figure 3.4. The decomposed 2D-DWT coefficients then first pass through a 2D convolution layer, with **64** filters of shape 3×3 and parametric rectified linear unit (PReLU) [47] non-linearity. The output then passes through another identical set of 2D convolution and PReLU layers. The final transformed 2D-DWT coefficients coming from the previous PReLU layer are then converted back to the spatial domain again, by performing a 2D-IDWT using the db2 wavelets. The initial features input to the WaveRB is finally added to the 2D-IDWT output, using a local skip-connection, before leaving the block.

Assuming, the input to the WaveRB is F_{in} , mathematically, the output from

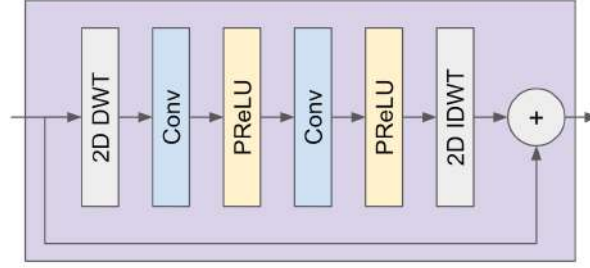


Figure 3.4: Proposed wavelet residual block (WaveRB).

WaveRB, F_{out} , can be written as:

$$F_{out} = IDWT(PReLU(Conv(PReLU(Conv(DWT(F_{in})))))) + F_{in} \quad (3.9)$$

So, the WaveRB operator can be defined as follows:

$$WaveRB \equiv (IDWT \circ PReLU \circ Conv \circ PReLU \circ Conv \circ DWT + 1)(\cdot) \quad (3.10)$$

3.3.3 Residual block

Each residual block has two sets of convolution layer, followed by a PReLU non-linearity. It also has a residual connection from the input to the output, which is being added to the output from the last PReLU layer (Figure 3.5). Every 2D convolution layer in the RB has **64** filters of shape 3×3 .

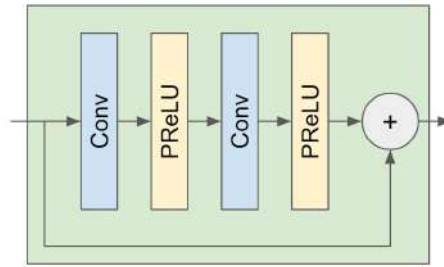


Figure 3.5: Residual block (RB).

Assuming, the input to the RB is F_{in} , mathematically, the output from RB, F_{out} , can be written as:

$$F_{out} = PReLU(Conv(PReLU(Conv(F_{in})))) + F_{in} \quad (3.11)$$

So, the RB operator can be written as follows:

$$RB \equiv (PReLU \circ Conv \circ PReLU \circ Conv + 1)(\cdot) \quad (3.12)$$

3.3.4 Upsampling block

Following the work of [2], which first proposed this technique for image super-resolution, we use a pixel-shuffling layer after a 2D convolution layer with **64**, 3×3 filters to up-sample feature maps inside the network by a factor of **2**. Since the up-sampling block only increases the spatial resolution by $2 \times$, for $4 \times$ super-resolution **2** such blocks should be cascaded and for the $8 \times$ super-resolution, it should be cascaded **3** times.

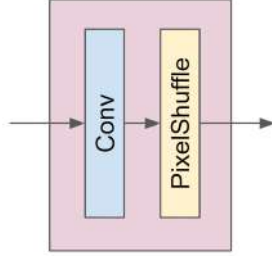


Figure 3.6: Upsampling block (UB).

Assuming, the input to the RB is F_{in} , mathematically, the output from UB, F_{out} , can be written as:

$$F_{out} = PixelShuffle_2(Conv(F_{in})) \quad (3.13)$$

So, the UB operator can be denoted as follows:

$$UB \equiv PixelShuffle_2 \circ Conv(\cdot) \quad (3.14)$$

3.4 Ablation study

An ablation study was carried out for the proposed novel architectural components of the WaveSRResNet. First, the effectiveness of the addition of wavelet residual path along with the pixel-domain residual path is investigated. Second, different loss functions are also included in the investigation such as loss of pixels and DWT coefficients, and SSIM-loss to find their impact on the network's performance. Third,

a visualization of what the WaveSRResNet learns at different stages is also provided for the interpretation of the model.

For $4\times$ image super-resolution, a WaveSRResNet is trained for **100** epochs with **1000** steps per epoch on the **DIV2K** dataset from scratch, and evaluated on the **Set5** dataset. The objective metrics PSNR and SSIM, and the perceptual metric LPIPS are monitored during the study. The results obtained in this study are presented in Table 3.1. Similar results are obtained for $2\times$ and $8\times$ image super-resolution but are not shown for the sake of brevity.

Table 3.1: Ablation study of the proposed architecture for the $4\times$ image super-resolution

Network components		Loss function components			Performance		
Residual path	Wavelet residual path	Pixel	DWT	SSIM	PSNR (\uparrow)	SSIM (\uparrow)	LPIPS (\downarrow)
Yes	No	Yes	No	No	31.65	0.8822	0.0496
Yes	Yes	Yes	No	No	31.10	0.8881	0.0464
Yes	Yes	Yes	Yes	No	31.97	0.8891	0.0489
Yes	No	Yes	Yes	Yes	31.99	0.8912	0.0436
Yes	Yes	Yes	Yes	Yes	32.38	0.8916	0.0454

From the table, it is evident that the inclusion of the wavelet residual path generally yields better results. Also, by computing the loss function in the pixel- and the wavelet-domain, together with the SSIM-loss gives the best results in terms of PSNR and SSIM, and the second best result in terms of LPIPS. It is also suggested by the study that the proposed wavelet residual path helps the model achieve better performance with this combination of loss function components. Using this key insight, we develop a customized loss function \mathcal{L}_{total} that is detailed in the following section.

A separate investigation was also carried out to visualize the activation maps inside the WaveSRResNet for a low-resolution patch. In Figure 3.7 we present the output activation maps number 29, 30, 37, and 38 of the *PReLU* layer, *WaveRB*₁, and *RB*₁. It is clear from figure that, additional image features such as the different edges are learned inside the WaveRB alongside the regular RB. These additional high-frequency edge information are crucial for reconstructing sharper and better super-resolution images.

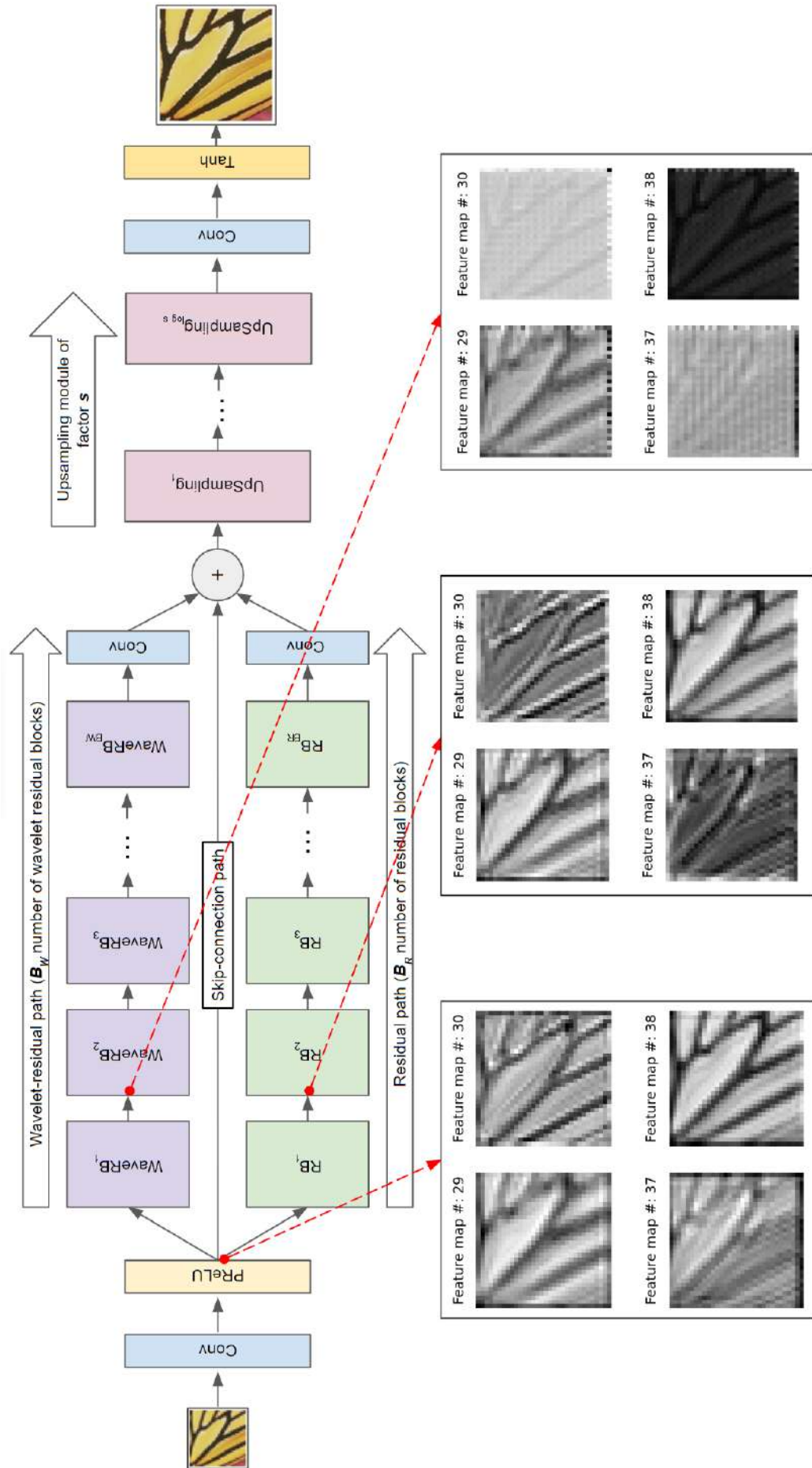


Figure 3.7: Visualization of activation maps number 29, 30, 37, and 38 inside the WaveSRResNet.

3.5 WaveSRResNet loss function

To optimize the WaveSRResNet, we use a weighted combination of the \mathcal{L}_1 reconstruction loss both in the spatial and the wavelet transform domain. Here, the \mathcal{L}_1 pixel loss in the spatial domain is responsible for guiding the model towards color accuracy and general shapes, whereas the \mathcal{L}_1 wavelet loss in the DWT domain is responsible for guiding the model to be edge and structure preserving. Hence, for the mapping $\mathcal{G}_s : I_{LR} \rightarrow I_{HR}$, where the output of the model, \mathcal{G}_s , is $\hat{I}_{HR} = \mathcal{G}_s(I_{LR})$, we then optimize the following loss term:

$$\begin{aligned} \mathcal{L}_{total} &= \mathbb{E}_{I_{LR}, I_{HR}} [\lambda_1 \times \|\hat{I}_{HR} - I_{HR}\|_1 + \lambda_2 \times \|DWT(\hat{I}_{HR}) - DWT(I_{HR})\|_1 \\ &\quad + \lambda_3 \times SSIM(\hat{I}_{HR}, I_{HR})] \\ \therefore \mathcal{L}_{total} &= \mathbb{E}_{I_{LR}, I_{HR}} [\lambda_1 \times \mathcal{L}_{pixel} + \lambda_2 \times \mathcal{L}_{DWT} + \lambda_3 \times \mathcal{L}_{SSIM}] \end{aligned} \quad (3.15)$$

Now let us dissect each term in more depth—

- \mathcal{L}_{pixel} : This term constrains the predicted \hat{I}_{HR} to be as close to the original I_{HR} as possible, in the spatial domain (pixel space). Since the loss is calculated in the spatial domain where each pixel corresponds to the intensity of the corresponding color channels, this term is mainly responsible for guiding the model to color accuracy.
- \mathcal{L}_{DWT} : This term constrains the predicted \hat{I}_{HR} to be as close to the original I_{HR} as possible, in the discrete wavelet transform domain (transform space). This term helps the model learn and reconstruct edges and small small details at higher resolution.
- \mathcal{L}_{SSIM} : This term constrains the predicted \hat{I}_{HR} to be structurally similar to I_{HR} , in the spatial domain.
- λ_1 , λ_2 , & λ_3 : Here, these values are chosen to give the \mathcal{L}_{pixel} and \mathcal{L}_{DWT} similar weight, and be comparable to \mathcal{L}_{SSIM} together. The benefit of setting 50% weight to \mathcal{L}_{SSIM} is that the reconstructed images would be more coherent in terms of image structure. Specifically, they need to obey the following

relations:

$$\begin{aligned}\lambda_1 &= \lambda_2 \\ |\lambda_1 + \lambda_2| &\leq |\lambda_3| \\ \mathbf{0} &\leq |\lambda_1, \lambda_2, \lambda_3| \leq \mathbf{1}\end{aligned}\tag{3.16}$$

3.6 Summary

In this chapter, first, we explored the architecture of our proposed model in depth, including each of its components such as the Wavelet Residual Block and the Upsampling Block. Then we finally wrapped up the chapter by discussing the proposed loss function that can best optimize our proposed Wavelet Super-Resolution Residual Network. In the next chapter, we present the results of experiment with the model on standard datasets and compare the performance with recent image super-resolution models.

Chapter 4

Experiments and Results

4.1 Introduction

This Chapter gives a brief description of the data used for training, validation, and testing. Three performance metrics that are suitable to measure the ability of the model quantitatively are introduced. An ablation study was performed on the architecture of the model to find the effectiveness of each block and each loss function components. After that, the implementation and training details for the WaveSRResNet are presented. Finally, the chapter is concluded by presenting the findings of the experiments for $2\times$, $4\times$, and $8\times$ WaveSRResNet.

4.2 Dataset

In this section, the different datasets that are used for developing and evaluating the WaveSRResNet for the task of image super-resolution are briefly described. Table 4.1 provides a short description of each dataset, the number of samples, the average image resolution, and the contents of the images in them.

For training and validation, the DIV2K dataset [56] was used, which contains 24-bit 800 training and 100 validation 2K-RGB images. The DIV2K data set is used extensively for image resotration tasks including the image super-resolution problem because of its higher resolution and variety of subjects ranging from different household objects to humans playing sports. The WaveSRResNet model was trained for $2\times$, $4\times$, and $8\times$ upsampling factors usign the DIV2K dataset. The LR

Table 4.1: Description of the datasets used

Dataset	Images	Avg. Resolution	Contents
DIV2K	1000	1972, 1437	environment, flora, fauna, handmade object, people, scenery, etc.
Set5	5	313, 336	baby, bird, butterfly, head, woman
Set14	14	492, 446	humans, animals, insects, flowers, vegetables, comic, slides, etc.
BSDS100	100	435, 367	animal, building, food, landscape, people, plant, etc.
Urban100	100	984, 797	architecture, city, structure, urban, etc.
Manga109	109	826, 1169	manga volume

images were generated by applying the bicubic degradation model to the original HR images [22]. The WaveSRResNet was evaluated on common benchmarking datasets including Set5 [57], Set14 [58], Urban100 [59], BSD100 [60, 61, 62], and Manga109 [63]; as listed in Table 4.1

4.3 Experimental design

All the models were implemented using the TensorFlow [64] deep-learning library in Python 3 programming language. The WaveTF [65] wavelet transform library for TensorFlow was used to implement DWT and IDWT layers inside the network. The I_{LR} images input to the model are scaled to the range $[0, 1]$ from $[0, 255]$, while the generated I_{SR} images are scaled back to $[0, 255]$ from $[-1, 1]$ (because of the *Tanh* activation function).

We used a batch-size of **16** for training the WaveSRResNet. For each batch, we randomly crop 128×128 patch size sub-images and iterate through the whole training set (DIV2K) to complete one training iteration (epoch). During training, the *Adam* [51] optimizer was used with default decay rates, $\rho_1 = 0.9$, $\rho_2 = 0.999$ and stabilization constant, $\delta = 10^{-7}$. The initial learning rate was set at $\epsilon = 10^{-4}$, which decayed by **10%** after every **10,000** training steps. **1000** steps were completed to complete an epoch.

A computer system with a 12th Gen Intel(R) Core(TM) i5-12400 2.50 GHz processor, 40 GB RAM, and a 12 GB GDDR5X NVIDIA Titan Xp was used for training the WaveSRResNet. The training time for the $2\times$, $4\times$, and $8\times$ super-resolution model in that system were **9**, **3.5**, and **2** minutes per epoch, respectively.

A separate model for each up-sampling factor was trained using the following hy-

perparameters:

- Up-scaling factor $s \in \{2, 4, 8\}$
- Mini-batch size, $m : 16$
- A low-resolution RGB input sub-image patch, $\mathbf{I}_{LR} \in \mathbb{R}^{128 \times 128 \times 3}$
- A high-resolution RGB ground-truth sub-image, $\mathbf{I}_{HR} \in \mathbb{R}^{(s \times 128) \times (s \times 128) \times 3}$
- A reconstructed super-resolution RGB sub-image, $\hat{\mathbf{I}}_{HR} \in \mathbb{R}^{(s \times 128) \times (s \times 128) \times 3}$
- j 'th input mini-batch, $\mathbf{X}^{[j]} = \{\mathbf{I}_{LR}^{(i)}\}_{i=1}^m \in \mathbb{R}^{16 \times 128 \times 128 \times 3}$
- j 'th ground-truth mini-batch, $\mathbf{Y}^{[j]} = \{\mathbf{I}_{HR}^{(i)}\}_{i=1}^m \in \mathbb{R}^{16 \times (s \times 128) \times (s \times 128) \times 3}$
- j 'th reconstructed mini-batch, $\hat{\mathbf{Y}}^{[j]} = \{\hat{\mathbf{I}}_{HR}^{(i)}\}_{i=1}^m \in \mathbb{R}^{16 \times (s \times 128) \times (s \times 128) \times 3}$
- Scaling basis function in the DWT layer, $\varphi : db2$
- Wavelet basis function in the DWT layer: $\psi : db2$
- Kernel size in the first and the last convolution layer, $(\mathbf{P} \times \mathbf{Q}) \in \mathbb{R}^{9 \times 9}$
- Kernel size in all other convolution layers, $(\mathbf{P} \times \mathbf{Q}) \in \mathbb{R}^{3 \times 3}$
- Convolution padding: *same* padding
- Number of filters in every convolution layer except the last: **64**
- Number of filters in the last convolution layer: **3**
- Number of wavelet residual blocks, (WaveRB's) $\mathbf{B}_W : 16$
- Number of residual blocks, (RB's) $\mathbf{B}_R : 16$
- Number of up-sampling blocks, (UB's) : **1/2/3**
- Weight of \mathcal{L}_{pixel} , $\lambda_1 : 0.5$
- Weight of \mathcal{L}_{DWT} , $\lambda_2 : 0.5$
- Weight of \mathcal{L}_{SSIM} , $\lambda_3 : 1.0$

Here, all tensors are represented in the *channels last* format (where applicable), i.e. (**Height** \times **Width** \times **number of channels**). This is the default tensor representation for TensorFlow.

4.4 Performance metrics

Performance metrics for image super-resolution can be categorized into several groups based on the aspect of image quality they measure. We are interested in how well the learning-based algorithm performs on data that it has never seen before in terms of its objective and subjective qualities. In that note the following two categories of performance metrics for image enhancement:

4.4.1 Objective metrics

- **PSNR:** Peak signal-to-noise ratio (PSNR) is a performance metric used to express the ratio between the maximum possible power of a signal and the power of an error signal that affects the fidelity of its representation.

The PSNR (in units of dB) is defined as:

$$PSNR(\hat{Y}, Y) = 10 \log_{10} \left(\frac{\max(Y)^2}{MSE(\hat{Y}, Y)} \right) \quad (4.1)$$

- **SSIM:** The structural similarity index measure (SSIM) is a method for evaluating the perceived quality of digital images and videos [45]. SSIM is used for measuring the similarity between two images.

The SSIM index is calculated on various windows of an image. The measure between two windows \hat{Y} and Y of a common size is:

$$SSIM(\hat{Y}, Y) = \frac{(2\mu_{\hat{Y}}\mu_Y + \delta_1)(2\sigma_{\hat{Y}Y} + \delta_2)}{(\mu_{\hat{Y}}^2 + \mu_Y^2 + \delta_1)(\sigma_{\hat{Y}}^2 + \sigma_Y^2 + \delta_2)} \quad (4.2)$$

where

- $\mu_{\hat{Y}}$ the pixel sample mean of \hat{Y}
- μ_Y the pixel sample mean of Y
- $\sigma_{\hat{Y}}^2$ the variance of \hat{Y}
- σ_Y^2 the variance of Y
- $\sigma_{\hat{Y}Y}^2$ the covariance of \hat{Y} and Y
- $\delta_1 = (\kappa_1 L)^2$, $\delta_2 = (\kappa_2 L)^2$ two variables to stabilize the division with weak denominator

- L the dynamic range of the pixel-values (typically this is $2^{\#\text{bits per pixel}} - 1$)
- $\kappa_1 = 0.01$ and $\kappa_2 = 0.03$ by default

4.4.2 Perceptual metrics

- **LPIPS:** The Learned Perceptual Image Patch Similarity (LPIPS) is used to judge the perceptual similarity between two images [46]. LPIPS is based on a deep neural network that learns representations of image patches and computes a similarity score based on the differences between these representations. It aims to capture human perception by considering higher-level visual features such as textures, shapes, and objects.

Perceptual metrics like LPIPS are often used in image enhancement tasks to evaluate the quality of the enhanced images from a human perception standpoint. This measure has been shown to match human perception well. A low LPIPS score means that image patches are perceptually similar. By considering perceptual factors, these metrics provide a more comprehensive evaluation of image quality beyond traditional objective metrics like PSNR or SSIM.

4.5 Methods for comparison

The methods for comparison can be broadly divided into two categories for the purposes of this thesis— first category is where only \mathcal{L}_1 or \mathcal{L}_2 loss is utilized to optimize the super-resolution model. The second category is where the methods use gan-based [24] optimization techniques and perceptual loss functions [27]. The quality of the images produced from these two different categories of methods are usually compared via different performance metrics. Methods from the former category are mostly evaluated using objective metrics such as PSNR or SSIM, whereas methods from the latter one are usually evaluated using various perceptual metrics such as the LPIPS, for quantitative performance measure.

4.5.1 $\mathcal{L}_1/\mathcal{L}_2$ loss-based methods

The performance of the WaveSRResNet is compared with the following methods using the PSNR and SSIM performance metrics.

- **EDSR** (2017): The authors of the paper titled “Enhanced Deep Residual Networks for Single Image Super-Resolution” [21] improved on the [20] by removing unnecessary batchnormalization module and stabilizing the training procedure. The EDSR model is a CNN based architecture employing residual blocks, optimizing an \mathcal{L}_1 loss in the spatial domain.
- **RCAN** (2018): In their paper “Image Super-Resolution Using Very Deep Residual Channel Attention Networks” [22] the authors proposed a novel residual in residual (RIR) block which allows low-frequency components of the LR image to be bypassed using different skip-connections. This paper also uses channel attention technique to reduce channel information redundancies. The RCAN model is also a CNN based architecture with channel attention, which is also optimized on an \mathcal{L}_1 loss in the spatial domain.
- **MWCNN** (2018): The “Multi-level Wavelet-CNN for Image Restoration” paper [14] proposes a wavelet-based U-net [28] inspired model that resembles the wavelet packet transform (WPT) architecture. Their novel contribution is adding DWT layers in the encoder block and corresponding IDWT layers in the decoder block. Basically, the downsampling and upsampling operations of the pooling and unpooling layers are achieved via the DWT and IDWT layers. The model is optimized using an \mathcal{L}_2 loss in the spatial domain.
- **SAN** (2019): In “Second-Order Attention Network for Single Image Super-Resolution” [23] the authors present a second-order channel attention module to further strengthen the feature correlation learning. The SAN model is optimized using an \mathcal{L}_1 loss on the spatial domain.
- **SwinIR** (2021): The “SwinIR: Image Restoration Using Swin Transformer” [26] paper proposes a transformer architecture for image restoration, by using novel residual swin transformer blocks. The SwinIR model is trained by optimizing \mathcal{L}_1 loss in the spatial domain.

4.5.2 GAN-based methods

SRGAN (“Super-Resolution Generative Adversarial Network”, 2017) [20], **ESR-GAN** (“Enhanced Super-Resolution Generative Adversarial Networks”, 2018) [25], **NatSR** (“Natural and Realistic Single Image Super-Resolution”, 2019) [29], **SFT-GAN** (“Spatial Feature Transform Generative Adversarial Network”, 2018) [66], and **SPSR** (“Structure-Preserving Super Resolution”, 2021) [30] are some of the most prominent GAN-based image super-resolution models that are optimized by optimizing some sort of perceptual loss function, against which we compare the LPIPS values of our proposed WaveSRResNet.

4.6 Results

The results are presented in three different approaches- one using objective metrics, next subjective evaluation using visual outputs, and finally using perceptual metrics.

4.6.1 Objective evaluation

Tables 4.2, 4.3, & 4.4 show the comparison of proposed WaveSRResNet with recent methods: EDSR [21], RCAN [22], MWCNN [14], SAN [23], and SwinIR [26], in terms of PSNR (\uparrow) and SSIM (\uparrow) on the Y channel (i.e., luminance) of in the transformed YCbCr color space. The best results are shown in red, while the second best results are shown in blue.

In 2 out of 5 benchmarking datasets, our proposed method performs better for the $2\times$ image super-resolution. For the $4\times$ and $8\times$ up-scaling factors, the proposed WaveSRResNet performs better on 3 out of 5 benchmarking datasets, and on 5 out of 5 benchmarking datasets, respectively. WaveSRResNet achieved, on average, **4.8%** higher PSNR (\uparrow) and **8%** higher SSIM (\uparrow) objective scores than the most successful and recent methods for the $8\times$ image super-resolution (Table 4.4), the most challenging scenario. Our WaveSRResNet outperforms the most competitive method SwinIR [26] on all datasets for the $8\times$ image super-resolution.

Table 4.2: Results of comparing methods in terms of PSNR and SSIM for the $2\times$ image super-resolution.

Method ($2\times$)	Year	Set5		Set14		BSD100		Urban100		Manga109	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
EDSR	2017	38.11	0.9602	33.92	0.9195	32.32	0.9013	32.93	0.9351	39.10	0.9773
RCAN	2018	38.27	0.9614	34.12	0.9216	32.41	0.9027	33.34	0.9384	39.44	0.9786
MWCNN	2018	37.91	0.9600	33.70	0.9182	32.23	0.8999	32.30	0.9296	-	-
SAN	2019	38.31	0.9620	34.07	0.9213	32.42	0.9028	33.10	0.9370	39.32	0.9792
SwinIR	2021	38.35	0.9620	34.14	0.9227	32.44	0.9030	33.40	0.9393	39.60	0.9792
WaveSRResNet	2023	37.28	0.9468	33.29	0.9071	34.53	0.9472	33.84	0.9293	38.16	0.9681

Table 4.3: Results of comparing methods in terms of PSNR and SSIM for the $4\times$ image super-resolution.

Method ($4\times$)	Year	Set5		Set14		BSD100		Urban100		Manga109	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
EDSR	2017	32.46	0.8968	28.80	0.7876	27.71	0.7420	26.64	0.8033	31.02	0.9148
RCAN	2018	32.63	0.9002	28.87	0.7889	27.77	0.7436	26.82	0.8087	31.22	0.9173
MWCNN	2018	32.12	0.8941	28.41	0.7816	27.62	0.7355	26.27	0.7890	-	-
SAN	2019	32.64	0.9003	28.92	0.7888	27.78	0.7436	26.79	0.8068	31.18	0.9169
SwinIR	2021	32.72	0.9021	28.94	0.7914	27.83	0.7459	27.07	0.8164	31.67	0.9226
WaveSRResNet	2023	32.64	0.9016	29.25	0.8148	30.06	0.8742	28.33	0.8362	32.01	0.9043

Table 4.4: Results of comparing methods in terms of PSNR and SSIM for the $8\times$ image super-resolution.

Method ($8\times$)	Year	Set5		Set14		BSD100		Urban100		Manga109	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
EDSR	2017	26.96	0.7762	24.91	0.6420	24.81	0.5985	22.51	0.6221	24.69	0.7841
RCAN	2018	27.31	0.7878	25.23	0.6511	24.98	0.6058	23.00	0.6452	25.24	0.8029
SAN	2019	27.22	0.7829	25.14	0.6476	24.88	0.6011	22.70	0.6314	24.85	0.7906
SwinIR	2021	27.37	0.7877	25.26	0.6523	24.99	0.6063	23.03	0.6457	25.26	0.8005
WaveSRResNet	2023	27.40	0.7898	25.43	0.7043	27.27	0.7839	25.17	0.7226	26.50	0.7797

4.6.2 Subjective evaluation

Figures 4.1, 4.2, and 4.3 show the subjective comparison of the output of the proposed WaveSRResNet with two methods: EDSR [21] and SwinIR [26]. Difference images for zoomed-in sections are also shown for a better comparison. The difference images is bright for regions where the model performs poorly, and dark for regions where the model performs well, as measured by the ground-truth HR images.

Subjective comparisons of WaveSRResNet to the most conventional method EDSR and the most competitive method SwinIR reveal the superiority of our proposed method for the extremely challenging large up-scaling factors (Figures 4.2, 4.3), while demonstrating comparable performance for small up-scaling factors (Figure 4.1). In Figure 4.2 we can clearly see the edges of the person are not being reconstructed well by EDSR and SwinIR, while our WaveSRResNet is able to reconstruct them well. Lastly, in Figure 4.3, we can clearly see edge artifacts being created between and inside the windows by EDSR and SwinIR, while our WaveSRResNet does a good job of mitigating it.

4.6.3 Perceptual evaluation

Table 4.5 shows the comparison of proposed WaveSRResNet with recent methods: SRGAN [20], ESRGAN [25], NatSR [29], SFTGAN [66], and SPSR [30] in terms of the LPIPS (\downarrow) metric. The best results are shown in red, while the second best results are shown in blue.

It shows that, the proposed WaveSRResNet achieved **9%** lower LPIPS (\downarrow) on average, for the $4\times$ up-scaling factor. All of the methods compared for perceptual evaluation are optimized on a particular form of perceptual metric. It is therefore evident that by adding the DWT path and optimizing in the DWT domain helps the model achieve greater perceptual results.

4.7 Summary

This chapter began with a short description of all the datasets used in this thesis, followed by the experimental design of this research. Hardware and software requirements, training settings, as well as the hyperparameter selection were discussed.

Table 4.5: Results of comparing with GAN-based methods in terms of LPIPS for the $4\times$ image super-resolution.

Method	Year	DIV2K val	Set5	Set14	BSD100	Urban100
		LPIPS (\downarrow)	LPIPS (\downarrow)	LPIPS (\downarrow)	LPIPS (\downarrow)	LPIPS (\downarrow)
SRGAN	2017	0.1263	0.0882	0.1663	0.1777	0.1551
ESRGAN	2018	0.1154	0.0748	0.1329	0.1615	0.1229
NatSR	2019	0.1523	0.0939	0.1758	0.2115	0.1500
SFTGAN	2018	0.1449	0.0890	0.1481	0.1769	0.1433
SPSR	2021	0.1099	0.0644	0.1318	0.1613	0.1184
WaveSRResNet	2023	0.0999	0.0432	0.1045	0.1556	0.1255

After introducing 3 performance metrics (PSNR, SSIM, LPIPS) from 2 categories (objective, perceptual) and the existing methods for comparison, we present our results from 3 evaluation perspectives— objective, subjective, and perceptual. The results show that our proposed WaveSRResNet performs significantly better than all the recent methods in the most challenging scenario (i.e. $8\times$ super-resolution) across the three evaluation criteria.

Since the $2\times$ image super-resolution is fairly simpler in setting than the other larger up-scaling factors such as $4\times$ or $8\times$, during training the model is high likely to be over-fitted. This may be the reason that the WaveSRResNet does not perform well for the $2\times$ image super-resolution. However, for higher up-scaling factors since we have very limited data to work with, over-fitting can be easily avoided, and hence, the model performs significantly better.

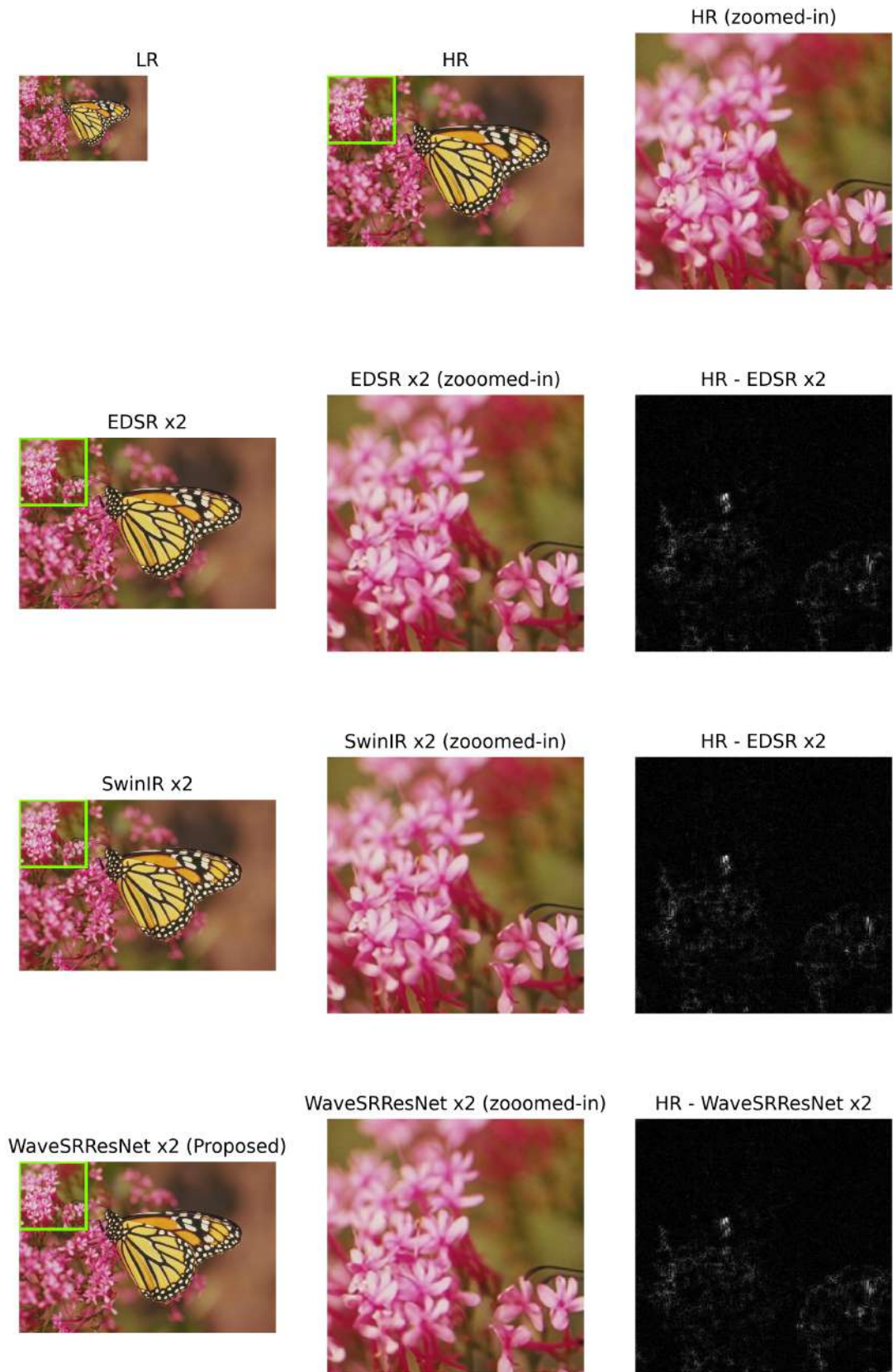


Figure 4.1: Results of visual comparison for the $2\times$ super-resolution on the “Monarch.png” image from the ‘Set14’ dataset. The difference image is shown for the zoomed-in sections.

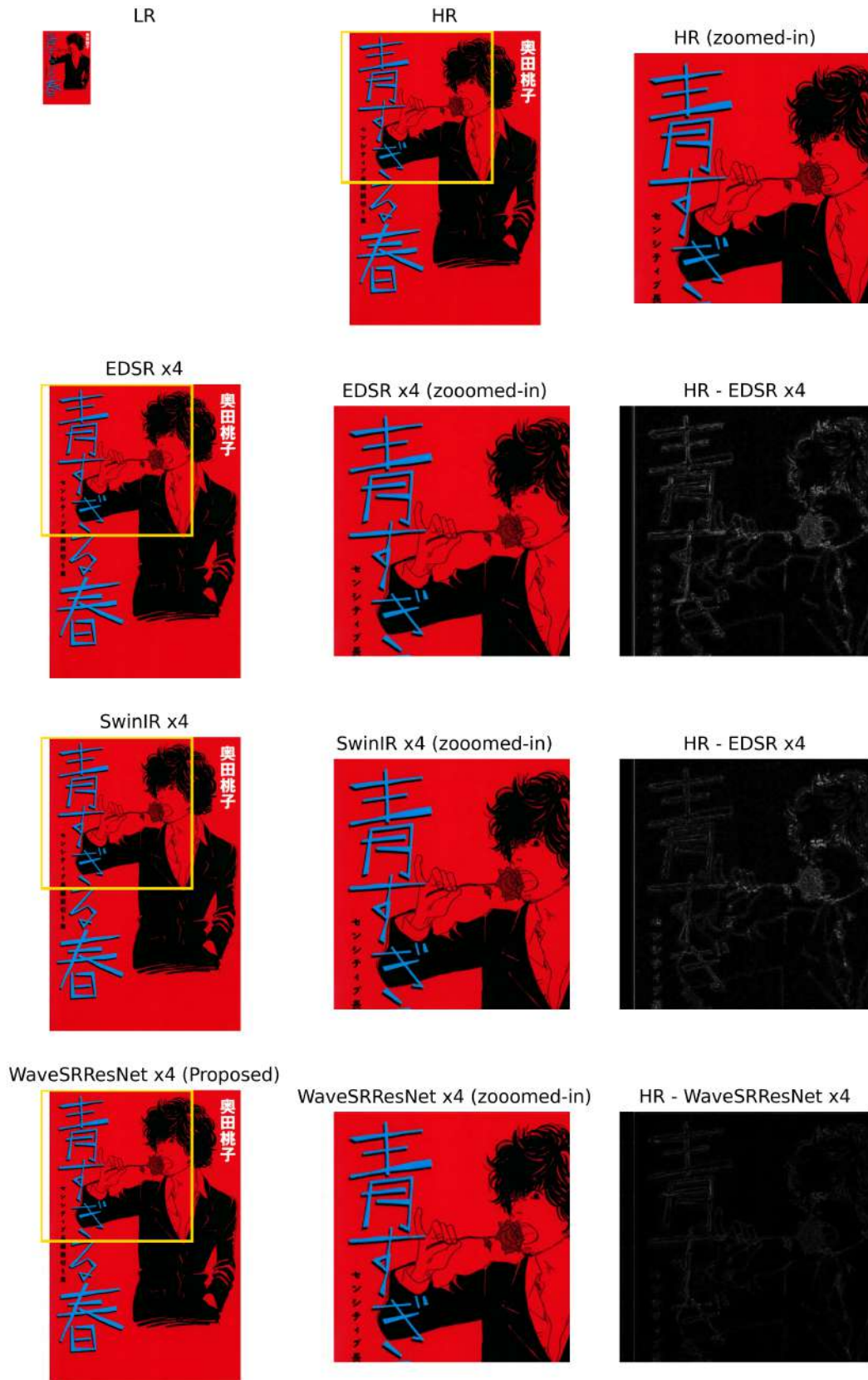


Figure 4.2: Results of visual comparison for the $4\times$ super-resolution on the “HaruichibanNoFukukorox4.png” image from the ‘Manga109’ dataset. The difference image is shown for the zoomed-in sections.

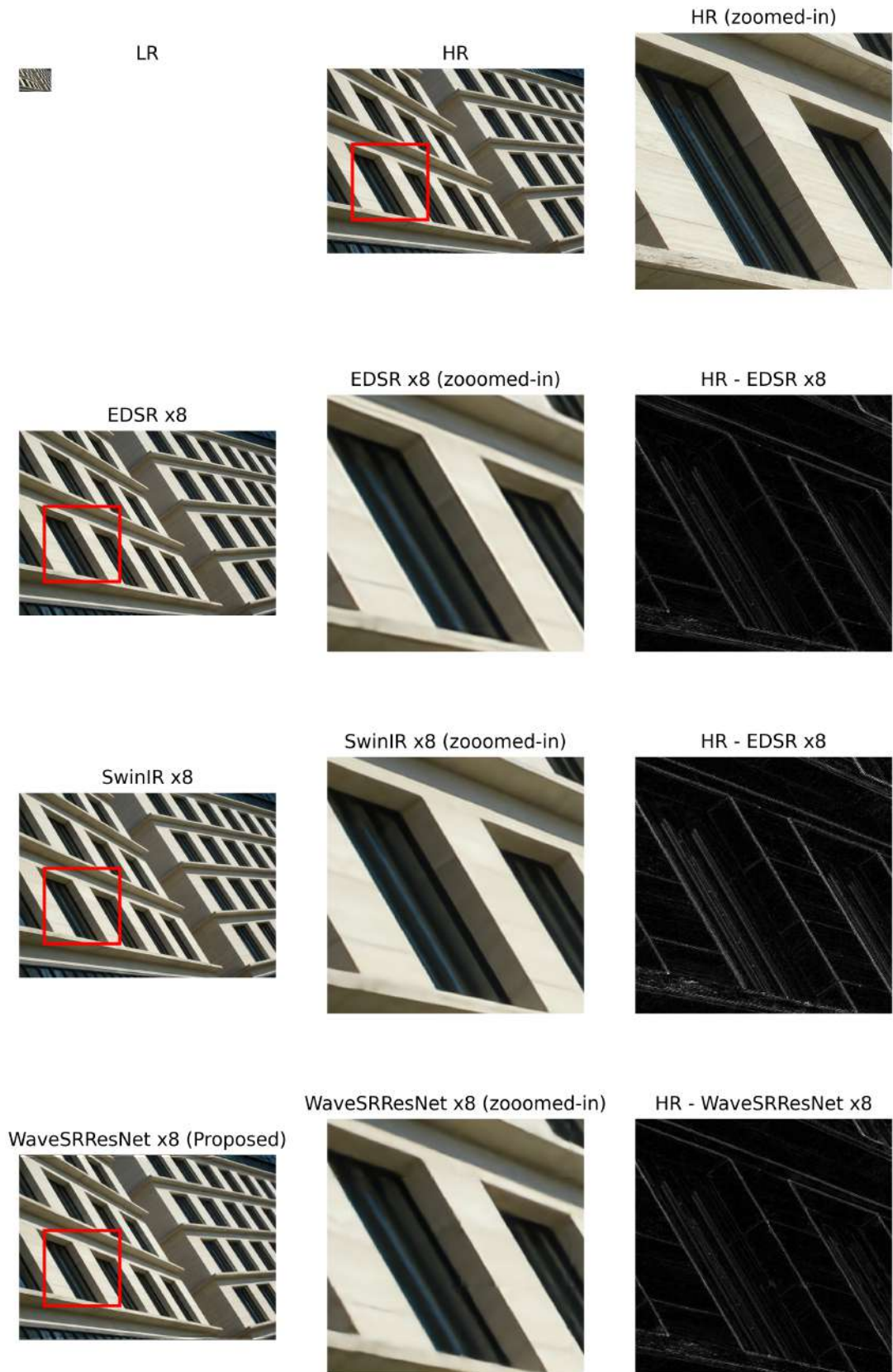


Figure 4.3: Results of visual comparison for the $8\times$ super-resolution on the “img025.png” image from the ‘Urban100’ dataset. The difference image is shown for the zoomed-in sections.

Chapter 5

Conclusion

5.1 Concluding remarks

Image super-resolution is a classic image processing task which has applications in various domains such as security, computer graphics, remote-sensing, image generation, etc. With the improvement of hardware for learning based methods, nowadays, almost all the competitive methods in image super-resolution are CNN based, usually optimized on objective or perceptual loss functions. One of the main challenges in this area is not being able to reconstruct sharper and clearer super-resolved images. We attribute this issue to not being able to reconstruct the high-frequency components properly. To mitigate this issue, this thesis proposes a novel DWT based regularization method that helps the model being edge preserving to produce sharper images. To facilitate the regularization, a novel wavelet residual block is proposed which performs forward and inverse DWT inside the CNN. The model was designed in such a way that the convolutional neural network with residual connection can be trained in an end-to-end fashion. The effectiveness of the said WaveRB block and the proposed regularization loss function term was designed after a proper ablation study.

To validate the premise, various experiments have been carried out. Image super-resolution methods can be broadly categorized into the type of loss functions that they optimize. One category of methods optimize using common objective functions such as the PSNR or SSIM, while the others use different forms of perceptual metrics. We compare the results of our proposed model with both kinds of methods.

Specifically, we compare the PSNR and SSIM values of the proposed WaveSRResNet with the following methods in the first category, namely EDSR [21], RCAN [22], SAN [23], and SwinIR [26] for $2\times$, $4\times$, and $8\times$ image super-resolution. For the $2\times$ up-scaling factor, our proposed method performs better in 2 out of 5 benchmarking datasets. For the $4\times$ and $8\times$ up-scaling factors, the proposed WaveSRResNet performs better in 3 out of 5, and 5 out of 5 benchmarking datasets, respectively. For the $8\times$ image-super resolution, our WaveSRResNet beats the most competitive method SwinIR [26] across all datasets. Subjective comparison of WaveSRResNet with the most traditional method EDSR and the most competitive method SwinIR clearly indicate the superiority of our proposed method for the extremely challenging large up-scaling factors, and comparable performance for small up-scaling factors. The perceptual performance of our model is also compared against 5 methods from the second category: SRGAN [20], ESRGAN [25], NatSR [29], SFTGAN [66], and SPSR [30]. WaveSRResNet performs better than these methods on 4 out of 5 benchmarking datasets.

The critical part of this research was to integrate the DWT layers inside the CNN models in such a way that the resulting learning-based model can be optimized and trained in an end-to-end fashion. Most of the DWT-based super-resolution networks [15] use the DWT as a mere pre-processing tool, which does not allow the said models to be trained end-to-end. This crucial step was overcome by incorporating the WaveTF [65] DWT library with the existing TensorFlow [64] deep learning framework. Thus, by using the forward and inverse DWT layers inside the convolutional neural networks, along with a DWT based regularization loss function, can facilitate the reconstruction of super-resolution images, and improves the quality of the reconstructed super-resolution images on both objective, subjective (visual quality) and perceptual metrics.

5.2 Future works

The results and findings of this thesis can be taken forward in two directions. Firstly, we can investigate the impact of other multi-resolution wavelet-based transforms such as the complex wavelet transform for the problem of image super-resolution.

This is a natural progression of the work because while the discrete wavelet transform takes into direction 3 directions, namely horizontal, vertical, and diagonal, other directional transformation should be able to improve the performance even further. The challenge in that lies in being able to design the model in such a way that it can be trained end-to-end.

Secondly, this proposed super-resolution network architecture can be applied for video super-resolution, which is also an important task for computer vision. For this application, the temporal relationship between the frames also needs to be addressed. In this case, the model also needs to be lightweight.

Bibliography

- [1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing, Global Edition*. Pearson, 2018.
- [2] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, “Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.
- [3] I. Daubechies, *Ten lectures on wavelets*. SIAM, 1992.
- [4] R. Keys, “Cubic convolution interpolation for digital image processing,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 6, pp. 1153–1160, 1981.
- [5] K. Turkowski, “Filters for common resampling tasks,” *Graphics gems*, pp. 147–165, 1990.
- [6] E. Meijering, “A chronology of interpolation: from ancient astronomy to modern signal and image processing,” *Proceedings of the IEEE*, vol. 90, no. 3, pp. 319–342, 2002.
- [7] L. I. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.
- [8] A. Chambolle, “An algorithm for total variation minimization and applications,” *Journal of Mathematical Imaging and Vision*, vol. 20, pp. 89–97, 2004.
- [9] Y. Meyer, *Wavelets and Operators: Volume 1*. Cambridge University Press, 1992, no. 37.

- [10] I. Daubechies, *Ten lectures on wavelets*, ser. CBMS-NSF regional conference series in applied mathematics. Philadelphia, Pa: Society for Industrial and Applied Mathematics, 1992, no. 61.
- [11] S. Mallat, *A wavelet tour of signal processing*. Elsevier, 1999.
- [12] H. Huang, R. He, Z. Sun, and T. Tan, “Wavelet-srnet: A wavelet-based cnn for multi-scale face super resolution,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1689–1697.
- [13] N. Kumar, R. Verma, and A. Sethi, “Convolutional neural networks for wavelet domain super resolution,” *Pattern Recognition Letters*, vol. 90, pp. 65–71, 2017.
- [14] P. Liu, H. Zhang, K. Zhang, L. Lin, and W. Zuo, “Multi-level wavelet-cnn for image restoration,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 773–782.
- [15] F. A. Dharejo, F. Deeba, Y. Zhou, B. Das, M. A. Jatoy, M. Zawish, Y. Du, and X. Wang, “Twist-gan: Towards wavelet transform and transferred gan for spatio-temporal single image super resolution,” *ACM Transactions on Intelligent Systems and Technology*, vol. 12, no. 6, pp. 1–20, 2021.
- [16] W.-Y. Hsu and P.-W. Jian, “Detail-enhanced wavelet residual network for single image super-resolution,” *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–13, 2022.
- [17] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian, “A real-time algorithm for signal analysis with the help of the wavelet transform,” in *Wavelets*, J.-M. Combes, A. Grossmann, and P. Tchamitchian, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 286–297.
- [18] 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, 2015.
- [19] J. Kim, J. K. Lee, and K. M. Lee, “Accurate image super-resolution using very deep convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1646–1654.

- [20] 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,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4681–4690.
- [21] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, “Enhanced deep residual networks for single image super-resolution,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 136–144.
- [22] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, “Image super-resolution using very deep residual channel attention networks,” in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 286–301.
- [23] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, “Second-order attention network for single image super-resolution,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 065–11 074.
- [24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in Neural Information Processing Systems*, vol. 27, 2014.
- [25] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, “Esrgan: Enhanced super-resolution generative adversarial networks,” in *Proceedings of the European Conference on Computer Vision Workshops*, 2018, pp. 0–0.
- [26] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte, “Swinir: Image restoration using swin transformer,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2021, pp. 1833–1844.
- [27] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part II 14*. Springer, 2016, pp. 694–711.

- [28] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.
- [29] J. W. Soh, G. Y. Park, J. Jo, and N. I. Cho, “Natural and realistic single image super-resolution with explicit natural manifold discrimination,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8122–8131.
- [30] C. Ma, Y. Rao, J. Lu, and J. Zhou, “Structure-preserving image super-resolution,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7898–7911, 2021.
- [31] Z. Wang, J. Chen, and S. C. Hoi, “Deep learning for image super-resolution: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3365–3387, 2020.
- [32] C. Lemaréchal, “Cauchy and the gradient method,” *Doc Math Extra*, vol. 251, no. 254, p. 10, 2012.
- [33] J. Hadamard, *Mémoire sur le problème d’analyse relatif à l’équilibre des plaques élastiques encastrées*. Imprimerie Nationale, 1908, vol. 33.
- [34] R. Courant, “Variational methods for the solution of problems of equilibrium and vibrations,” 1943.
- [35] G. W. Leibniz, *The early mathematical manuscripts of Leibniz*. Courier Corporation, 2012.
- [36] S. Linnainmaa, “The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors,” Ph.D. dissertation, Master’s Thesis (in Finnish), Univ. Helsinki, 1970.
- [37] “OpenAI’s GPT-3 Language Model: A Technical Overview,” Jun. 2020. [Online]. Available: <https://lambdalabs.com/blog/demystifying-gpt-3>

- [38] 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*, 2016, pp. 770–778.
- [39] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, “Visualizing the loss landscape of neural nets,” *Advances in Neural Information Processing Systems*, vol. 31, 2018.
- [40] Z. Wang, E. Simoncelli, and A. Bovik, “Multiscale structural similarity for image quality assessment,” in *The 37th Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 2, Nov. 2003, pp. 1398–1402 Vol.2.
- [41] E. J. Candès, D. L. Donoho *et al.*, *Curvelets: A surprisingly effective nonadaptive representation for objects with edges*. Department of Statistics, Stanford University USA, 1999.
- [42] E. Candes, L. Demanet, D. Donoho, and L. Ying, “Fast discrete curvelet transforms,” *Multiscale Modeling & Simulation*, vol. 5, no. 3, pp. 861–899, 2006.
- [43] N. Kingsbury, “Image processing with complex wavelets,” *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 357, no. 1760, pp. 2543–2560, 1999.
- [44] S. G. Mallat, “A theory for multiresolution signal decomposition: the wavelet representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.
- [45] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [46] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 586–595.
- [47] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.

- [48] A. Haar, “Zur theorie der orthogonalen funktionensysteme,” *Mathematische Annalen*, vol. 71, no. 1, pp. 38–53, 1911.
- [49] J. Bernoulli, “Ars conjectandi: Usus & applicationem praecedentis doctrinae in civilibus, moralibus & oeconomicis,” *Translated into English by Oscar Sheynin. Basel: Turneysen Brothers,. Chap*, 1713.
- [50] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization.” *Journal of machine learning research*, vol. 12, no. 7, 2011.
- [51] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [52] Z. Wang, J. Chen, and S. C. H. Hoi, “Deep Learning for Image Super-resolution: A Survey,” Feb. 2020, arXiv:1902.06068 [cs]. [Online]. Available: <http://arxiv.org/abs/1902.06068>
- [53] M. Abramowitz, I. A. Stegun, and R. H. Romer, “Handbook of mathematical functions with formulas, graphs, and mathematical tables,” 1988.
- [54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [55] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part I 13*. Springer, 2014, pp. 818–833.
- [56] E. Agustsson and R. Timofte, “Ntire 2017 challenge on single image super-resolution: Dataset and study,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 126–135.
- [57] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, “Low-complexity single-image super-resolution based on nonnegative neighbor embedding,” 2012.

- [58] R. Zeyde, M. Elad, and M. Protter, “On single image scale-up using sparse-representations,” in *Curves and Surfaces: 7th International Conference, Avignon, France, June 24-30, 2010, Revised Selected Papers 7*. Springer, 2012, pp. 711–730.
- [59] 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*, 2015, pp. 5197–5206.
- [60] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2. IEEE, 2001, pp. 416–423.
- [61] R. Timofte, V. De Smet, and L. Van Gool, “Anchored neighborhood regression for fast example-based super-resolution,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1920–1927.
- [62] —, “A+: Adjusted anchored neighborhood regression for fast super-resolution,” in *Computer Vision—ACCV 2014: 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part IV 12*. Springer, 2015, pp. 111–126.
- [63] Y. Matsui, K. Ito, Y. Aramaki, A. Fujimoto, T. Ogawa, T. Yamasaki, and K. Aizawa, “Sketch-based manga retrieval using manga109 dataset,” *Multimedia Tools and Applications*, vol. 76, pp. 21 811–21 838, 2017.
- [64] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015, software available from [tensorflow.org](https://www.tensorflow.org). [Online]. Available: <https://www.tensorflow.org/>

- [65] F. Versaci, “Wavetf: A fast 2d wavelet transform for machine learning in keras,” in *Pattern Recognition. ICPR International Workshops and Challenges*. Springer International Publishing, 2021, pp. 605–618.
- [66] X. Wang, K. Yu, C. Dong, and C. C. Loy, “Recovering realistic texture in image super-resolution by deep spatial feature transform,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 606–615.