IMAGE COMPRESSION METHOD BASED ON LEARNED LIFTING-BASED DWT AND LEARNED ZEROTREE-LIKE ENTROPY MODEL

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

UĞUR BERK ŞAHIN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2022

Approval of the thesis:

IMAGE COMPRESSION METHOD BASED ON LEARNED LIFTING-BASED DWT AND LEARNED ZEROTREE-LIKE ENTROPY MODEL

submitted by **UĞUR BERK ŞAHIN** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering** _____

Assoc. Prof. Dr. Fatih Kamışlı
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Prof. Dr. İlkay Ulusoy
Electrical and Electronics Engineering, METU _____

Assoc. Prof. Dr. Fatih Kamışlı
Electrical and Electronics Engineering, METU _____

Prof. Dr. Alptekin Temizel
Graduate School Of Informatics, METU _____

Assoc. Prof. Dr. Elif Vural
Electrical and Electronics Engineering, METU _____

Assist. Prof. Dr. Osman Serdar Gedik
Computer Engineering, Yıldırım Beyazıt University _____

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Uğur Berk Şahin

Signature :

ABSTRACT

IMAGE COMPRESSION METHOD BASED ON LEARNED LIFTING-BASED DWT AND LEARNED ZEROTREE-LIKE ENTROPY MODEL

Şahin, Uğur Berk
M.S., Department of Electrical and Electronics Engineering
Supervisor: Assoc. Prof. Dr. Fatih Kamışlı

August 2022, 73 pages

The success of deep learning in computer vision has sparked great interest in investigating deep learning-based algorithms also in many image processing applications, including image compression. The most popular end-to-end learned image compression approaches are based on auto-encoder architectures, where the image is mapped via convolutional neural networks (CNNs) into a transform (latent) representation that is quantized and processed again with CNNs to obtain the reconstructed image. The quantized latent representation is entropy coded to obtain a compressed bitstream. To have efficient entropy coding, the probability distribution of the quantized latent representation is also modeled with CNNs. The entire system, including the auto-encoder and the probability model of the latent representation, is trained jointly to minimize the rate-distortion cost function.

A successful traditional image compression system is the Embedded Zerotree Wavelet (EZW) coding algorithm, which is based on the Discrete Wavelet Transform (DWT) and the Zerotrees (ZT) of wavelet coefficients. This thesis explores a similar learning-based compression architecture. In particular, a learned lifting-based DWT and a

learned ZT-like probability model of the transform coefficients are used. Several variations of the ZT-like probability model are explored. The entire system is trained end-to-end to minimize a rate-distortion cost function. The explored system is compared with JPEG2000 and state-of-the-art learned image compression methods.

Keywords: Image Compression, Jpeg2000, Neural Network, EZWT, Lifting Structure, CNN

ÖZ

ÖĞRENİLMİŞ KALDIRAÇ TABANLI DWT VE ÖĞRENİLMİŞ ZEROTREE-BENZERİ ENTROPİ MODELİNE DAYALI GÖRÜNTÜ SIKIŞTIRMA YÖNTEMİ

Şahin, Uğur Berk
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Fatih Kamışlı

Ağustos 2022 , 73 sayfa

Bilgisayarla görüde derin öğrenmenin başarısı, görüntü sıkıştırma da dahil olmak üzere birçok görüntü işleme uygulamasında derin öğrenmeye dayalı algoritmaların araştırılmasına büyük ilgi uyandırdı. En popüler uçtan uca öğrenilen görüntü sıkıştırma yaklaşımları, görüntünün evrişimli sinir ağları (CNN'ler) aracılığıyla nicelenen ve CNN'ler ile tekrar işlenen bir dönüşüm (gizli) temsiline eşlendiği otomatik kodlayıcı mimarilerine dayanmaktadır. Yeniden yapılandırılmış görüntü nicelendirilmiş gizli temsil, sıkıştırılmış bir bit akışı elde etmek için entropi kodludur. Etkili entropi kodlamasına sahip olmak için, nicelenmiş gizli gösterimin olasılık dağılımı da CNN'ler ile modellenmiştir. Otomatik kodlayıcı ve gizli gösterimin olasılık modeli dahil tüm sistem, kod uzunluğu-görüntüdeki bozulma maliyet fonksiyonunu en aza indirmek için ortaklaşa eğitilir.

Başarılı bir geleneksel görüntü sıkıştırma sistemi, Dalgacık katsayılarının Ayrık Dalgacık Dönüşümü (DWT) ve Zerotrees (ZT)'e dayanan Embedded Zerotree Wavelet (EZW) kodlama algoritmasıdır. Bu tez, benzer bir öğrenme tabanlı sıkıştırma mimari-

sini araştırmaktadır. Özellikle, dönüşüm katsayılarının öğrenilmiş bir kaldıraç tabanlı DWT ve öğrenilmiş bir ZT benzeri olasılık modeli kullanılır. ZT benzeri olasılık modelinin çeşitli varyasyonları araştırılmıştır. Tüm sistem, hız bozulma maliyet fonksiyonunu en aza indirmek için uçtan uca eğitilmiştir. Keşfedilen sistem, JPEG2000 ve son teknoloji öğrenilmiş görüntü sıkıştırma yöntemleri ile karşılaştırılmıştır.

Anahtar Kelimeler: Görüntü Sıkıştırma, Jpeg2000, Yapay Sinir Ağları, EZWT, Kaldıraç yapısı, Evrişimsel Sinir Ağları

To science

ACKNOWLEDGMENTS

Firstly, I want to send my most sincere pleasure to Assoc. Prof. Dr. Fatih Kamışlı for being a supportive and helpful supervisor during my master's degree. As a supervisor, he not only tries to help me during my studies but also try to give his best where I can't find any solutions. It has been a great experience for me to be one of his graduate students.

I would express my genuine appreciation to my beloved family and friends for their support and motivation to me during my master's studies. Moreover, I would also like to acknowledge the Scientific and Technological Research Council of Turkey(Tübitak) for their scholarship, with project number 2210-A. Lastly, I would like to express my most sincere thanks to Aselsan Inc. for its support during my master studies.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ANN	Artificial neural network
bpp	Bit per pixel
cdf	Cumulative distribution function
CNN	Convolutional Neural Network
DCT	Discrete cosine transform
EZWT	Embedded Zero Wavelet Tree
GDN	General divisive normalization
MSE	Mean squared error
pdf	Probability distribution function
PSNR	Peak signal to noise ratio
RNN	Recurrent Neural Networks

CHAPTER 1

INTRODUCTION

1.1 Image Compression

Image compression is a set of algorithms to reduce the number of bits to store or transmit an image for a given a quality level. Nowadays, every image we take with our phones or we look at on our phones or computers are compressed in one way or another, because even simple compression algorithms can achieve significant compression ratios, which can save storage, bandwidths, latency and power. For example, the most widely used and supported image compression algorithm JPEG[10], which has a quite simple algorithm compared to other conventional methods, can achieve compression by a factor of 5-10 without noticeable image quality degradation.

There are properties of images that make them compressible. Firstly, spatial and spectral dependencies exist in images and by exploiting these dependencies, compression is achieved. Secondly, the human perceptual system can not differentiate images above a quality level and human perceptual system is more sensitive to low frequency components of an image signal when compared with high frequency components. For instance, there are significant amounts of high-frequency components in image signals, human perception system can not differ the images after neglecting some high-frequency components of these signals.

Transform coding, which is the most common image compression approach, has a common structure, as shown in Figure 1.1. The transform block maps the input image signal to a transform domain to represent the given signal in a more efficient way. Applying a transformation decorrelates the signal, which makes scalar quantization more compression efficient. The encoder block encodes the given representation into

a bitstream, where one of the most popular methods is arithmetic coding or Huffman coding[11].

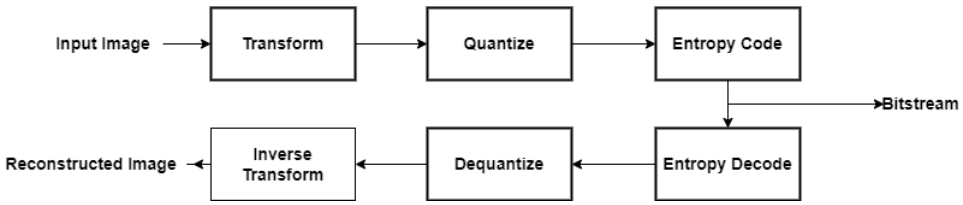

Figure 1.1: Compression model

Image compression can be divided into two main branches, lossy image compression, and lossless image compression. In lossless image compression, the original image can be reconstructed without loss of information at the decoder by using the compressed bitstream. On the other hand, in lossy image compression the original image can not be reconstructed without any distortion, due to loss of information in the quantization block. In general, different conventional methods also adopt the quantization block for frequency components. By using a large quantization step at high frequencies and small quantization step at low frequencies, the compression can be done efficiently with small distortions which could not be understood by the human perceptual system. Moreover, lossy image compression methods generally use the benefit of limitation of human perceptual system.

There are important conventional methods such as JPEG[10], JPEG2000[12], HEVC[13], and BPG which use Fourier, DCT[14], KLT methods which are linear transforms. The power of conventional methods arises from their low computational complexity. However, with the increasing popularity of neural networks, many image processing problems have been revisited using neural network based algorithms such as image denoising [15][16], image reconstruction [17], image super-resolution [18][19] and image compression. In image compression, different from the classical approaches, which have hand-derived coefficients and parameters, the blocks in Figure 1.1 are replaced with neural network based structures whose parameters can be learned from training sets of images.

1.2 Motivation and Problem Definition

As mentioned in Section 1.1, an image compression systems consists of several parts, which are given in Figure 1.1. Conventional methods are generally built by optimizing the block parameters in the given scheme separately and from simple statistical models of image data to get the best result. However, with the introduction of neural networks, joint optimization of system blocks has become easier. There are deep learning models which are better than conventional methods and these models are mostly built using auto encoder networks, which will be given in detail in Section 3.3.3. However, these models are hard to explain by using mathematics because of the nature of neural networks, and they do not depend on any classical method approach.

There are a few works that inspired by the classical methods and these methods replace some parts of classical methods with deep learning based methods [7][8].In [7], the wavelet transform in JPEG2000 is replaced with a neural network based nonlinear transform, which is built using the lifting scheme[3] in which the prediction and update filters are constructured with convolutional neural networks[20]. The rest of the JPEG2000 scheme is preserved while in [8] end-to-end neural network scheme is proposed by replacing the lifting predict and update blocks by CNN blocks, and extract the dependencies of wavelet coefficients by using LSTM[21] blocks to model entropy efficiently, and post-processing blocks are used to decrease the distortions introduced by quantization.

In this thesis, the JPEG2000 structure is mostly preserved, and an end-to-end neural network-based model is proposed at Section 4 with slight modifications to JPEG2000 architecture. One of the main contribution different from the method in [8], the proposed work tries to adapt EZWT[4] to extract the dependencies in between the subbands at different wavelet levels. The results are better than the JPEG2000 approach in terms of both rate and distortion at all compression levels.

In summary, this thesis work is inspired by JPEG2000 architecture and the works in [7][8] mainly, where a trainable end-to-end image compression scheme is developed and proposed. In addition, the results of the proposed method are comparable to BPG

and other autoencoder-based network models.

1.3 Novelties in the Thesis

This section includes the novelties that are introduced in this thesis work. The novelties can be summarized as follows:

1. A neural scaling network rather than a post-processing network as in [8]
2. Exploiting the probabilistic dependency of wavelet transform coefficients across successive wavelet levels, inspired by Embedded Zerotree Wavelet coding (EZWT) [4]
3. Using the probabilistic dependency of wavelet coefficients across successive wavelet levels and in the spatial neighborhood of the same subband, where spatial dependency is extracted using the causal spatial neighborhood of wavelet coefficients (with two different methods) in the same subband

These proposals are investigated in details in this thesis report.

1.4 Outline of the Thesis

In Chapter 2, preliminary information is given related to this work. It includes topics such as image compression, a review of essential points of neural networks, classical approaches, standard blocks that are used in image compression and wavelet transform. This chapter prepares the reader for the methods in the proposed in this thesis.

In Chapter 3, the literature review is done on the topic of image compression, mainly the recent ones with the best results in terms of rate and distortion. The ones, which provided inspiration for this work, are given in detail to prepare the reader for the proposed method.

In Chapter 4, the proposed method is investigated firstly part by part and added up step by step for an end-to-end image compression system.

In Chapter 5, after the introduction of the proposed method, experiments and results are shared to show the positive and negative sides of the proposed approach.

Finally, in the last chapter, Chapter 6, the conclusions are given, and potential future works to improve the model are shared.

CHAPTER 2

BACKGROUND

2.1 Neural Network

Artificial neural networks are studied within the fields of AI and computer science. The main methodology is to model the human neuron systems using nodes, modules, and algorithms. The reason they are called "neural" networks comes from the nature of the human neural system. It tries to mimic the human brain and understand the important points in a set of data. There are different kinds of neural networks:

- Artificial Neural Networks (ANN): Generally used for image classification, loses spatial information [22]
- Recurrent Neural Networks (RNN): Generally used for time series data analysis [23]
- Convolution Neural Networks (CNN): Generally used to extract spatial information in data[24]

ANNs can not preserve most of the spatial information, while CNNs are mainly used to extract spatial information, where the most vital part of RNNs is the capability to preserve sequential information. Neural networks, are optimized using the loss function, where the loss function inherits the neural network's main purpose. In other words, the main task of the neural network is defined by the loss function, and the network tries to reduce the loss function to learn what it is asked for where loss functions differ depending on the task. For example, in image compression, the loss function mostly includes rate and distortion (mean squared error), while in image

classification, the loss function consists of intersection over union. Moreover, the power of neural networks comes from differentiability and non-linearity, which will be investigated in the following sections.

2.1.1 Convolution Neural Networks(CNN)

Convolution neural networks(CNN) are mainly used to extract spatial information. They are more flexible when compared with other classical filtering methods, where CNN can be thought of as a family of filters with learnable weights, where these weights can be modified in the optimization (training) step so that they can extract the necessary information using the spatial dependency.

CNN can be used sequentially, by putting one CNN after another to increase the capability of the model, and it can even be used with fully connected layers depending on the structure. Lastly, the preprocessing for CNNs is generally less than other neural network methods, because of their flexibility.

2.1.2 Nonlinearities

Convolutional filters or any weight parameters can result in a linear operation, while in order to solve a problem that is not linear, non-linearity should be introduced, where these non-linearities are also called activation functions in the neural network literature. In other words, the solution's complexity should be increased.

A neural network model can try to solve a problem as much as it can, up to its complexity limitation. Thus, increasing the complexity up to a level decreases loss function unless other problems occur, such as overfitting. In other words, to see better results, the complexity of a neural network model should be increased by introducing non-linearities, while introducing a non-linearity means putting non-linearity functions after neural networks layers which is mentioned in Section 2.1. It should be noted that increasing the number of non-linearities can decrease performance after a complexity level because of some reasons, such as:

- Overfitting problem: The problem when neural network has become biased

to training data set, so test data set accuracy can decrease while training data set accuracy increases. The capability of generalizability of neural network decreases when this problem occurs.

- Vanishing gradient problem: In neural networks' optimization, chain rule, which will be mentioned in Section 2.1.4, is used for gradient flow. The multiplication of values that are near to zero, results in smaller values in each multiplication through chain rule procedure, which results in very small gradients at the final stage, and these low gradients are inefficient to update the weights, where this phenomenon is called vanishing gradient problem.

- Computational complexity problem: In neural networks, as complexity increases or as the network gets deeper or wider, more parameters are used, which results in heavier process on processing units and the process takes longer.

Non-linearities are used to increase the complexity of a neural network model, by putting a non-linearity between layers, each level is forced to learn a different level of information of the training data. In general, only one non-linearity is used in between two layers, where the most popular non-linearities in literature are as follows:

- Relu
- LeakyRelu
- GDN
- Mish
- Sigmoid
- Swish
- Tanh

Formula for each activation function can be seen below in corresponding sections.

2.1.2.1 Relu and Leaky Relu

Relu and LeakyRelu are the two most popular activation functions in literature. Equation 2.1 shows the Relu activation function. It is linear on the positive side of the axis and zero valued on the negative side.

$$f(x) = max(x, 0) \tag{2.1}$$

Equation 2.2 shows the mathematical expression for LeakyRelu, it is the same as input on the positive side and has a different slope on the opposing side. The main difference between Relu and LeakyRelu is that LeakyRelu has non-zero values on the negative side of x-axis.

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ ax, & \text{otherwise} \end{cases} \tag{2.2}$$

2.1.2.2 Mish

Mish is getting more popular recently because it is continuous at zero and is non-linear at the negative side of x-axis, while it is bounded at the negative side.

$$f(x) = x * tanh(softplus(x)) \tag{2.3}$$

2.1.2.3 Sigmoid

Sigmoid function is one of the early activation functions.

$$f(x) = \frac{1}{(1 + e^{-x})} \tag{2.4}$$

2.1.2.4 Swish

Google Brain team discovered swish in 2017, a modified version of the sigmoid function. It should be noted that when $\beta = 0$, it becomes a linear function. Thus, it can be used as a linear or non-linear activation function, depending on the β parameter.

$$f(x) = x * \sigma(\beta * x) \tag{2.5}$$

2.1.2.5 Tanh

Tanh, called hyperbolic function, is one of the most used in image compression because after normalization of images around 0 mean, the range will be $-0.5, 0.5$ which makes tanh efficient for image processing.

$$f(x) = \frac{e^x - e^{-x}}{e^{-x} + e^x} \tag{2.6}$$

2.1.3 Data Set Selection

Data set selection is one of the most important part of neural network based models. The reason is that neural networks tend to learn what they have been trained on. In order to have a successful model, it should be trained using a diverse data set. Otherwise, the learning capability of neural networks can not be used efficiently. Thus, a network that is trained using a non-diverse dataset might give unexpected results and may not show the capability of a model. In addition, to compare network models, they should be tested using the same dataset. In this work, mainly the Vimeo Triplet Dataset is used for training and the Kodak dataset is used for testing.

2.1.4 Optimization of Neural Networks and Back Propagation

In deep learning, the purpose of the model is defined using a function called the loss function. The neural network model tries to minimize the loss function by using the gradients flowing through the network. In mathematics, derivatives are used to find minimum or maximum values of a function. Similarly, in neural networks, derivatives are used to extract information about the sensitivity to change of a function

concerning for the given argument.This information is used in optimizer algorithms in neural networks, such as gradient descent[25]), to decrease the loss function for better performance.

Back Propagation[26] is one of the fundamental algorithms that is used in neural networks, which was introduced nearly 40 years ago to use a gradient descent type algorithm for loss minimization. The chain rule method is the main idea because the output of a neural network depends on many cascaded layers or operations. The main aim of the network is to obtain the minimum cost value by updating weight and bias parameters using gradient descent. As it can be observed in the same figure, the initial weight parameters correspond to a point on the loss function, while at each update, gradients and the learning rate are used to obtain the incremental step.

Thus, chain rule is used to obtain the gradients depending on different weight parameters. After calculation of derivatives, weights(w) and bias(b) parameters can be updated using learning rate, shown by using ϵ parameter, the update procedure can be observed in Equations (2.7) and (2.8), where learning rate is the step size in the direction of derivative.

$$w = w - \epsilon * \frac{\partial Y}{\partial w} \tag{2.7}$$

$$b = b - \epsilon * \frac{\partial Y}{\partial b} \tag{2.8}$$

It should be noted that if the learning rate is much bigger than the desired value, the cost value will oscillate between the hills of the cost function because the weight parameter changes significantly. On the other hand, if learning rate is much lower than the desired value, then cost function value will settle after a lot of iteration because weight values change slightly in each iteration, which is not time efficient. Thus, the learning rate should be set to a reasonable value for an efficient learning process.Those phenomenons can be observed the using the Equations (2.7), (2.8).

In order to use back propagation, all operations in the network should have well-defined gradients. Moreover, there exist different gradient update algorithms, in

addition to gradient descent, where the most popular ones are Stochastic Gradient Descent[27] and ADAM optimizers.

2.1.5 Residual Networks

As it is mentioned in the previous section, the gradient flow is one of the most important properties of neural network optimization. However, as the network becomes deeper, derivatives get lower, because according to the chain rule, as it is mentioned, the final gradient value is calculated by multiplying partial derivatives and multiplication of low values in between $[-1, 1]$ results in lower values in magnitude. In other words, as deep learning models get deeper, the gradient gets smaller, which results in the problem of "vanishing gradients"[28]. In order to get rid of this problem, Residual Networks[1] are proposed. Advantages of residual networks are as follows:

- Preserving gradients through deep networks
- Faster to train
- Increases the performance when compared without using residual connections

The schematic of a simple residual network can be seen in Figure 2.1. The shortcut connection of x is the main branch to preserve gradients because the information and the gradients will be carried through. By using this method, gradients can be preserved for even much deeper networks. In addition, shortcut connections do not add new parameters or increase complexity while preserving gradients and information.

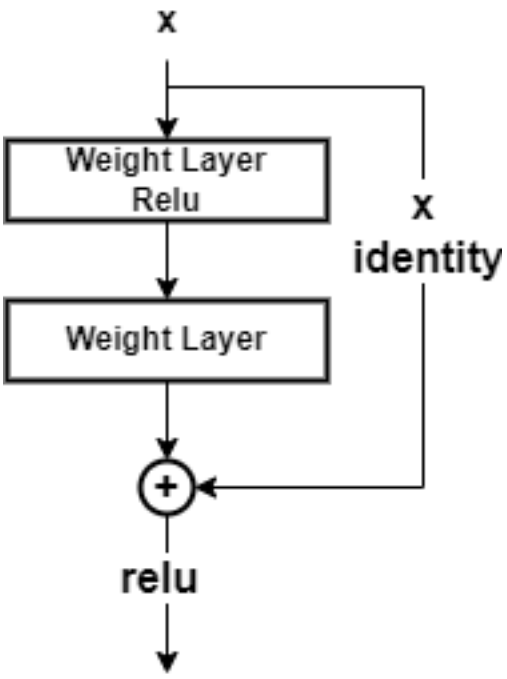

Figure 2.1: Residual block (redrawn from [1])

2.2 Data Compression

Data compression or source coding is the method that aims to represent data by using as few bits as possible for a given reconstruction fidelity or quality. Same approach is valid for the images because images have their data or information in their pixel values. Image compression can be divided into two main categories, similar to data compression.

- Lossy Image Compression: Reconstructed image from the bitstream is not the same as the original image; information loss occurs.
- Lossless Image Compression: Image can be reconstructed from the bitstream without loss of any information.

The main steps in image compression can be summarized as follows:

1. Image pixel values are represented as $x \in \mathbb{R}$.

2. Encoder transforms the pixel values to another domain (g_a, can be called analysis), called forward mapping.

3. The transform domain representation is quantized (with scalar quantization) and the quantized variables are entropy coded

4. The decoder decodes the coded information and reconstructs the image, called inverse mapping.

2.2.1 Quantization

Quantization[29] is a mapping from a continuous domain (or discrete domain with larger dynamic range) to a discrete domain, and the difference between the original value and quantized value is called quantization error. This quantization error is the reason for the loss in lossy image compression. Rounding or truncation to the nearest integer are two basic examples of quantization. After quantization, since the original information is lost, reconstruction can not be equal to the original signal before quantization.

As quantization intervals get larger, the number of necessary bits decreases, but quantization noise increases, which will be mentioned in detail in Section 2.2.3.

One of the important quantization methods in compression is Dead Zone[30] quantization, whose name comes from the region around zero quantized value. Depending on the application, the Dead Zone region can be set to any width. In compression since the main purpose is to preserve the important features while eliminating the less necessary ones, efficient use of dead zone region improves the performance by mapping insignificant features to zero where JPEG2000[12], one of the most popular classical method, uses Dead Zone for quantization. As mentioned before, the dead zone region can be modified using the width parameter in dead zone quantization. The formula for dead zone quantization can be seen in Equation (2.9). The w parameter can be changed for different dead zone region widths[31].

$$[h]y = sgn(x) * max(0, \lfloor(\frac{|x| - w/2}{\delta} + 1)\rfloor) \tag{2.9}$$

2.2.2 Entropy Coding

Entropy coding is one of the essential parts of image compression. Information can be mapped to another domain called symbols to represent the information efficiently in memory. Entropy can be considered the average of self-information, the mathematical expression for entropy can be seen in Equation (2.9) where $p(x_i)$ is the probability of each symbol and $I(x_i)$, given in Equation (2.10), is the self-information of each symbol. There are different methods for entropy coding, and the following sections will give information about one of the most popular methods, arithmetic coding[11].

$$I(x_i) = \frac{1}{\log(p(x_i))} \tag{2.10}$$

$$H(x) = \sum_{n=1}^{n} \frac{p(x_i)}{\log(p(x_i))} \tag{2.11}$$

2.2.2.1 Arithmetic Coding

Arithmetic coding is a reversible coding method. In arithmetic coding, the information is distributed in the range [0,1], and the shortest unique value is encoded into a bit stream. The method is illustrated in Figure 2.2. First the letters b, r, k are split in the range $(0, 1)$ based on their probabilities. As it can be seen from the chart, the probability of "a" is 0.3, so when "b" is coded the next interval will be $0 - 0.3$, then "r" is coded and lastly "k" is coded. After these steps, depending on the last code word, the code will be revealed, so one bit-array corresponds to a codeword. After the last information is coded, the range will be used to turn into the bit stream. For example, assume the last coding word is "a" so the range will be in between 0.144 and 0.180 where 0.144 can be taken for minimum use of bits after omitting the $0.$, where 10010000 equals to 144.

Range coder is similar to arithmetic coding, except the range coding upper and lower limit is not [1,0], respectively. The values are mapped to integer, so it can be considered an integer mapped limited type of arithmetic coding.

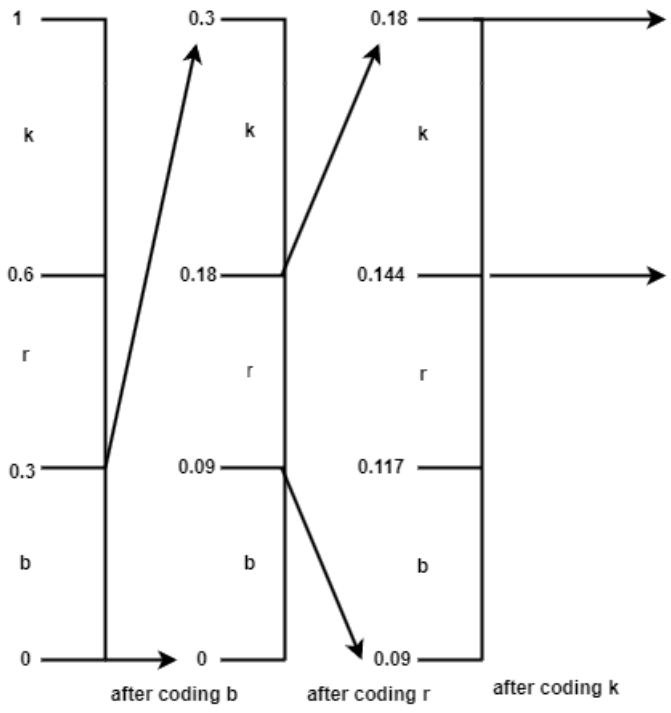

Figure 2.2: Arithmetic coding scheme

2.2.3 Relation between Rate and Distortion

In lossy image compression, there exists a trade-off between the rate(R) and the distortion(D), as can be seen from Equation 2.12. The λ, Lagrangian parameter, is used to find a solution for different rate values at different distortions or vice versa. The given equation will be used in the proposed method with slight changes.

$$L = D + \lambda * R \tag{2.12}$$

In classical methods, the best point for rate distortion is tried to be solved using iterative methods[32]. The reason for this is if a small quantization step size is used, less information will be lost at the quantization step of the compression algorithm because of high resolution of quantization, but this will increase the rate results in a longer bitstream. However, suppose big quantization step size is used. In that case, less information will be preserved, and the reconstructed image has fewer details when compared with the previous case, but rate will be smaller because of obtaining

a shorter codeword. The red curve shows the optimal points for each rate-distortion pair, depending on the efficiency of quantization.

2.3 Wavelet Transform

Wavelet transform is a kind of transformation that extracts both local and temporal information. Since Fourier Transform[33] captures global frequency information, it is not efficient at capturing local information sparsely. However, wavelet transformation can investigate different sets of intervals of frequencies where the advantages of wavelet transform as follows:

- Extract spatial information at the same time,
- Different kinds of wavelet transforms exist to choose from depending on the aim of the transform

In 2D wavelet transform, such as image transform, each level can be represented with three different sub-bands and the last wavelet level is represented as four different sub-bands. The extra sub-band in the last level is the LL band, which represents the low-frequency information in the image. Wavelet steps are as follows:

1. For the first level wavelet, decompose the image into four sub-bands LL, HL, LH, and HH.
2. Other than the first level, decompose the LL component into four sub-bands to obtain the next levels in wavelet transform.

LL of each level has the significant (low frequency) information while the other sub-bands have details, high-frequency components, like edges and corners. Specifically, HL has horizontal details, LH has vertical details and HH has diagonal details.

As it can be seen from Figure 2.3, each level's width and height are halved to obtain the next level sub-bands. Thus, in the wavelet, the next level has filtered information of the previous layer's LL sub-band, which can be used for entropy coding and modeling.

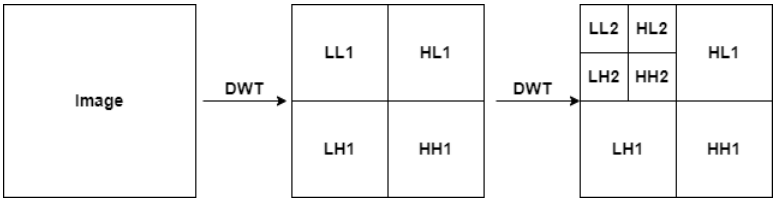

Figure 2.3: 2 level wavelet transforms (redrawn from [2])

2.3.1 Lifting Scheme

Lifting scheme[34] is a method that can be used to construct a wavelet transform. The main difference from the classical construction is that it does not rely on the Fourier Transform. The basic idea behind the lifting scheme is to use the correlation in the data to remove redundancy. The lifting scheme has three main steps called as follows:

1. Split: In this step, the input array (image) separated into odd and even elements
2. Predict: Predict block is used to approximate the dataset and difference between predicted values from even samples is the high frequency(H) output
3. Update: In update phase, an update filter is used on the high-frequency output samples so that missing details are added to even elements, giving the low frequency(L) output

In Figures 2.4, 2.5, predict first forward and inverse lifting transform architecture can be seen, respectively.

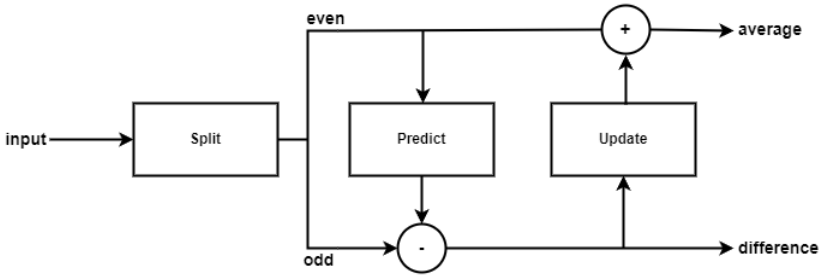

Figure 2.4: Classical lifting scheme (Forward) (redrawn from [3])

Lifting architecture can be used for lossless transforms because the lifting scheme can be inverted lossless. In the inverse lifting scheme, the lifting steps are performed in

reverse order and the addition and the subtraction operations are swapped. In other words, first, update filter is used to obtain even elements, then predict filter is used to obtain original odd elements, and finally, merge is used to have the original image, which is split in forward lifting transform (see Figure 2.5).

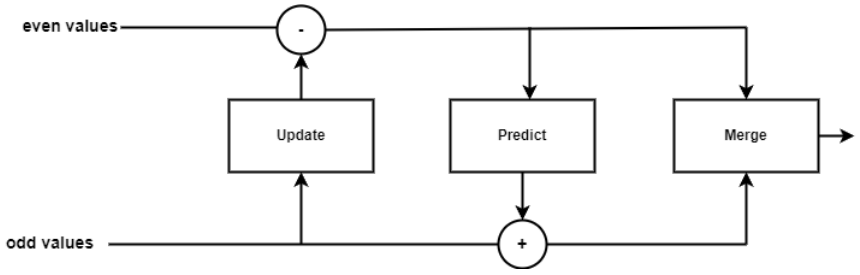

Figure 2.5: Classical lifting scheme (inverse) (redrawn from [3])

2.3.2 Embedded Zero Wavelet Tree(EZWT)

Classical methods are primarily good at high bit-rate values, while for low bit-rate, they are not efficient as expected because insignificant information for low bit-rate could not be eliminated correctly. Embedded Zero Tree is a method that is proposed in [4] to entropy code the quantized wavelet coefficients more efficient. It is a method to generate bit streams according to their order of importance. The embedded zero tree algorithm depends on wavelet transform because it needs a hierarchical sub-band decomposition to use ancestor and descendants. In Figure 2.6 ancestors and descendant relation can be observed. As it is mentioned before, wavelet transforms decomposes their input into four different sub-bands with halved spatial resolution (both width and height), where In Embedded Zero Tree, this is used as a dependency in between each level of transform. As it can be seen from Figure 2.6, the higher level sub-bands are actually a mapping of $2x2$ region of previous layers of the same sub-band. In other words, a pixel value of HL3 is obtained using the $2x2$ region of HL2, or a pixel in HH3 is a mapping from $2x2$ region of HH2; the same is valid for LH sub-band and for the other wavelet transform levels. After understanding this dependency, EZWT algorithm suggests putting a threshold(T), and the pixel values under that T value are named as insignificant pixels [4]. A pixel in a wavelet level can be called a zero tree root if all its descendants are lower than the T value. Zero tree

root and its descendants are put as insignificant pixels into a Significance Map used in coding.

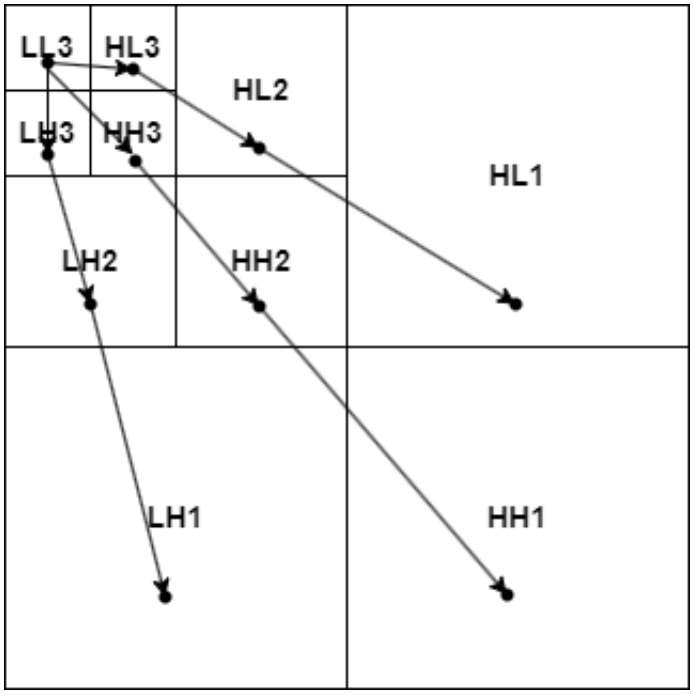

Figure 2.6: EZWT ancestor and descendant relation (redrawn from [4])

CHAPTER 3

RELATED WORK

3.1 Overview

This chapter will investigate methods and ideas that are recently popular in lossy image compression, especially the neural-networks-based ones, will be investigated. Firstly, comparison between classic and neural-network-based methods in terms of advantages and disadvantages will be given. Then, neural-network-based methods will be investigated. In each subsection, part of the general structure will be reviewed by giving the main ideas and benefits of the literature. After investigating the necessary literature works, the baseline for the proposed method will be given. The algorithm that is proposed in this thesis work is mainly inspired by the proposed methods in [8] and [35] which are mentioned in Section 3.3.4. The details of the proposed algorithm will be given in Section 4.

3.2 Classical and Neural Network Based Methods

The most popular classic image compression methods are JPEG, JPEG2000, H264 and H265. These methods are handcrafted, based on mathematics, signal processing, and optimization.

On the other hand, with the increased demand and knowledge in deep learning, especially convolutional neural networks(CNN) [20], image compression has become a field to use deep learning. General information about neural networks and how they work is given in Section 2, and information more closely related to compression will be investigated in this section. Table 3.1 gives a brief comparison of learning-based

and classical methods.

Table 3.1: Comparison of Classic and Learning Based Methods

Methods	Advantage	Disadvantage
Classic Methods	No need to train with data Easier to explain, depends on signal processing Powerful hardware not necessary Less computational requirements	Problem can be optimized up to a level Strong mathematical background necessary
Learning Methods	Higher level optimization No need to derive mathematical expressions Open to improvement	Needs different data to train to be comprehensive Hard to explain Complexity should be carefully arranged to avoid overfitting

3.3 Lossy End to End Image Compression

Image compression is divided mainly into lossy image compression and lossless image compression. In lossless image compression, the image can be decompressed without any loss of information. In other words, all the information in the image is preserved in this compression scheme. On the other hand, in lossy image compression, the image can not be reconstructed without loss of information because some information is discarded to achieve higher compression ratios. In this thesis, since the main focus is lossy image compression, lossless image compression will not be discussed further.

In learning-based image compression models, there are different groups of algorithms. These algorithms can be divided into two primary groups, called end-to-end image compression methods and partial methods. End-to-end compression methods use learning methods throughout the entire compression scheme. In other words, they are differentiable from the beginning to the end so that gradients can be used to optimize all network parameters jointly. Partial methods mainly optimize parts of a classical compression system separately. For example, in some works, classic methods are preserved, and post-processing networks are added at the end of the decoder as in [36] to increase the reconstructed image's quality. These will be investigated in detail in the following sections.

3.3.1 Non-Linear Transform Coding

Non-linear transform coding is one of the most powerful ideas in learning-based image compression algorithms. Although linear transforms are used for their simplicity in traditional compression systems, they decrease compression performance because of their inferior capability to compactly represent real-world image data. In order to get rid of this problem, Ballé et al. proposed the non-linear transform coding framework[5]. In Figure 3.1, the general structure of non-linear transform coding is summarized. The model is generally optimized using both rate and distortion.

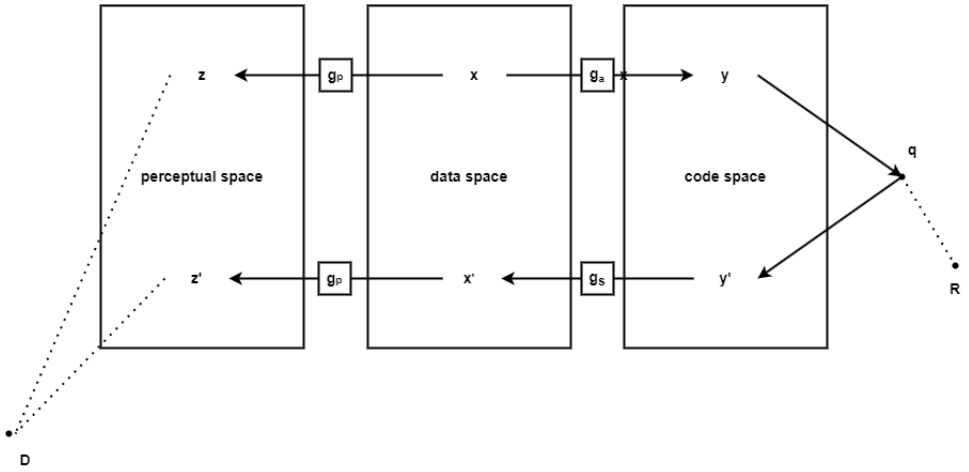

Figure 3.1: Non-Linear Transform Coding Scheme (redrawn from [5])

In Figure 3.1, z stands for perceptual space while x stands for data domain and y stands for code space. The functions g_a and g_s stand for analysis (encoding) and synthesis (decoding) transforms in image compression. In other words $y = g_a(x, \phi)$, where y stands for code vector (latent space), while $x' = g_s(y', \theta)$ where y' stands for quantized code vector, ϕ and θ stands for the parameters of transforms. Encoding and decoding schemes try to optimize both rate and distortion, depending on the given equation for optimization function.

In Figure 3.1, R stands for rate and D stands for distortion. It can be observed that R (rate) is calculated using the quantized code space, while D (distortion) is calculated using the reconstructed image and input image. To summarize, non-linear transform coding uses non-linear analysis and synthesis transforms composed of neural networks to make compression more efficient and accurate in terms of distortion and

rate.

$$loss = \lambda * D + R \tag{3.1}$$

The rate-distortion loss functions given in Eq.3.1 is used to optimize the parameters of the system. The λ parameter stands for the weight of distortion with respect to rate. As λ increases, the importance of distortion increases, meaning the reconstructed image's perceptual quality should increase, i.e. distortion decreases, while the importance of rate decreases, i.e. the rate increases. That means, if high quality reconstructed images are desired, λ should be increased. The following Equation 3.2 shows the loss function with expected value notation, where the expectations can be considered over x, i.e. images in the training set.

$$L[g_a, g_s, P_q] = E[\log(P_q)] + \lambda * E[d(z, \hat{z})] \tag{3.2}$$

3.3.2 Density Modeling and General Divisive Normalization

Density modeling is one of the main aspects of image processing. It is used in different areas such as image denoising, image compression, and image generation. There are different methods to achieve efficient results in density modeling. Since this thesis is based on neural networks, one important property that a density model should have is differentiability of the output w.r.t the input.

Image transform methods are generally designed by analyzing a large and diverse set of data to handle the large variations in the probability distributions of images. At the same time, there are some methods that can convert image data into a Gaussian Distribution. General Divisive Normalization is introduced by Ballé et al. [37] which was inspired from different divisive normalization methods. The name "general" comes from the ability to model different divisive normalization methods in the literature by changing parameters which were released before, like the one proposed in [38], to model local probability of natural images with a continuous and parametric approach.

$$z = Hx \tag{3.3}$$

$$y_i = \frac{z_i}{(B_i + \sum_j \gamma_{ij} |z_j|^{\alpha_{ij}})^{\epsilon_i}} \tag{3.4}$$

A linear transform is applied on the image data(x) as in Eq.(3.4), followed by the GDN operation in Eq.(3.3) can make the distribution of the output(y_i) Gaussian if such multiple operations (linear transform + GDN) are cascaded [37].

The parameters H, β, γ, in equations 3.4, 3.3, are optimized to have an efficient non-linear transform in GDN algorithm. GDN can be used as a nonlinearity, like the ones defined in Section 2.1.2, and as a normalization method instead of batch normalization [39] because batch normalization is not an adaptive method at inference while GDN is adaptive in inference, which increases performance.

3.3.3 Autoencoder Based Models

Autoencoder networks are suitable for image compression, as mentioned in Section 2. These networks map the input image, to a different space, called latent space. In compression, the main goal is to preserve most of the information while using the minimum bitrate for the coding of the latent space representation. There are different autoencoder based methods for compression, while the most popular ones are [5], [6], [40].

In classical methods, three components of transform coding methods, which are transformer, quantizer, and entropy coder, are separately optimized, using simple statistical models of data. To handle the compression problem as a whole, deep learning-based end-to-end compression scheme is proposed by Ballé et al. at [5] where GDN blocks are used as normalization nonlinearity. Analysis (encoder) and synthesis (decoder) part of the proposed network trained simultaneously using stochastic gradient descent. Since the quantization operation, performed in the latent space, has zero derivative almost everywhere, it is replaced by additive uniform noise during training. Rate distortion loss is used as a loss function. It should be noted that the Lagrangian

parameter can be used in front of rate or distortion since it is a scaling term to move on the Rate-Distortion curve as it is mentioned in Section 2.2.3. They suggest that artifacts in the proposed method in [5] is less than the artifacts in JPEG2000 and JPEG, because local features are represented using localized linear basis functions in these classical methods, and independent scalar quantization of the transform coefficients causes an imbalance in these combinations which result in visually disturbing patterns.

Balle et al. proposed an end-to-end trainable variational auto encoder [6] model for image compression, different from other autoencoder based image compression networks. They defined a hyperprior, a concept universal to virtually all modern image codecs, using ANNs [6]. In this work, noise is modeled using i.i.d uniform noise that is proposed in [5], rather than substituting the gradient of quantizer[41] because when i.i.d. noise is used to simulate quantization in the training phase, Kullback-Leibler divergence formula simplifies to rate-distortion problem. Different methods, called hyperprior and factorized prior, are proposed by using a similar architecture, which can be seen in Figure 3.2. AE and AD stand for arithmetic encoder and arithmetic decoder, respectively. The blocks g_a and g_s stand for encoder and decoder parts of the autoencoder network, that are used to map image domain into the latent domain and map the latent space to image domain respectively. The notation "$_a$" is used for "analysis" while "$_s$" stands for synthesis. h_a and h_s stand for the hyperprior analysis and synthesis parts of the hyperprior network, while the factorized model uses the same model shown in the figure except the entropy model networks on the right are different.

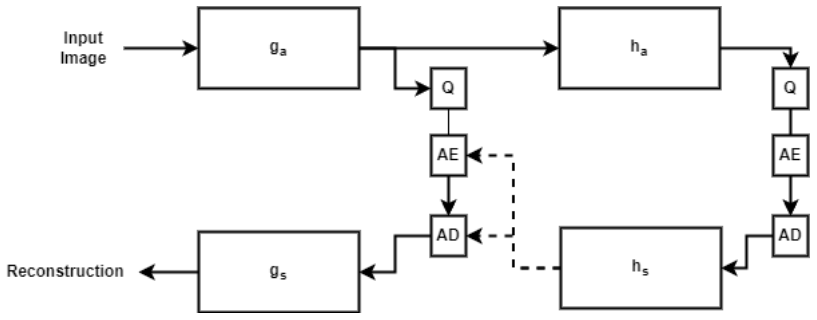

Figure 3.2: Hyperprior model architecture (redrawn from [6])

3.3.4 Lifting Wavelet Based Models

These are the methods that are the baselines for the proposed method in this thesis work. The main purpose is to have a similar architecture like JPEG2000 while surpassing its performance in terms of bit rate and PSNR.

3.3.4.1 iWave

Iwave is the name of the learned non-linear wavelet transform based compression architecture in [7]. JPEG2000 architecture is taken as baseline for the given method. The wavelet transform is replaced by a neural network model which is based on the lifting scheme, while the same quantization and entropy coding methods in JPEG2000 are used without change. The main reason behind this work is that used linear filters in the wavelet transforms are not good enough at transforming natural images. Wavelet transforms for two-dimensional images are usually performed by two steps of one-dimensional transform along with the horizontal and vertical directions, respectively, which causes inefficiency[7] because important features do not have to be vertical or horizontal. Furthermore, directional features in images result in large magnitude high-frequency coefficients, which decreases the compression performance. CNN's have some significant benefits over the classical approach, which are

- CNNs can extract information in spatial dimensions (width and height)
- Ensemble of different layers makes it possible to process different local features simultaneously
- Large number of images are used to optimize the weights rather than designing filters manually

In [7], different from the classical wavelet that is used in JPEG2000, CNNs are used to model lifting filters, together with non-linear activation functions. Since the traditional wavelet transforms have coefficients that are handcrafted using the ideal distribution assumptions, they do not give expected results because raw images are not ideal as they are assumed by the theory.

The lifting scheme is reviewed in Section 2.3.1, it can be separated as update first or prediction first depending on the positions of update or predict filter. In [7], authors use the update first scheme, which can be seen in Figure 3.3. Moreover, it is shown that only addition, subtraction and division operations are done, so inverse lifting transform is lossless. Using CNNs in the update and predict filters rather than using the filters with hand derived coefficients, do not change lifting's property because the same CNN filters are used in forward and backward transform. In other words, an image that is transformed using the lifting scheme can be reconstructed using inverse lifting if quantization is not performed on the lifting output.

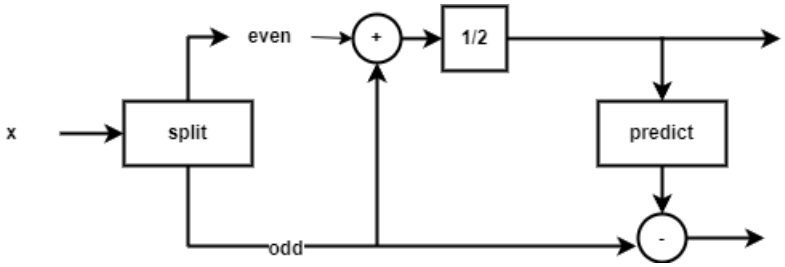

Figure 3.3: Update first lifting scheme, forward pass (redrawn from [7])

In Figure 3.3, split block is used to separate the image into odd and even rows and columns, while update filter is a mean or averaging filter. Output of each lifting stage gives x_c, coarse features, and x_d, detail features. Mathematical expressions for x_c and x_d can be seen at equations 3.5 and 3.6 respectively.

$$x_c = 0.5 * (x_e + x_o) \tag{3.5}$$

$$x_d = predict(x_c) - x_o \tag{3.6}$$

In this work [7], tanh is used as non-linearity rather than Relu because, according to the observation in the paper, the discontinuity at zero decreases performance, and the range of tanh is the best fit for the proposed method. In order to use the same quantization methods and entropy coding models, each sub-band output is scaled to fit into the JPEG2000 algorithm because the output of proposed lifting steps do not fit into the JPEG2000 scheme without the use of these scaling factors where these factors are found by experimenting and then setting the correct values.

According to the given results, slightly better performance is observed in terms of PSNR value, where iWave [7] showed a better energy compaction than JPEG2000 while it could not reach the performance of BPG and the other popular learning based models.

3.3.4.2 End to End Lifting Based Image Compression

The proposal of [7] showed that using CNNs in wavelet transform in lifting architecture has significant potential, but it is not an end-to-end approach. After the proposal of iwave, the same researchers proposed an end-to-end lifting-based compression scheme that surpasses, according to their results, even the best auto-encoder-based network models in compression. The new approach is called "iwave++"[8] which is an end-to-end image compression scheme with predict first lifting architecture. They suggest that using iwave++ architecture, lossy and lossless image compression can be done with the same logic by modifying the quantization method. Different from the proposed methods in the previous sections, iwave++ has the following properties:

- Prediction first lifting scheme
- End to end network
- Rate-distortion loss: $L = R + \lambda * D$, where λ is a Lagrangian parameter
- Uniform Scaling
- Both lossless and lossy compression
- Introducing a post-processing network

The general architecture of "iWave++" can be seen in figure 3.4. Forward and inverse transform use the same parameters in the update and predict filters. One important thing is to notice the "DeQuantization" block. This block is called a post-processing block to decrease distortion and increase the PSNR while preserving the same rate. This block helps to increase the quality of the image at the same bit-rate.

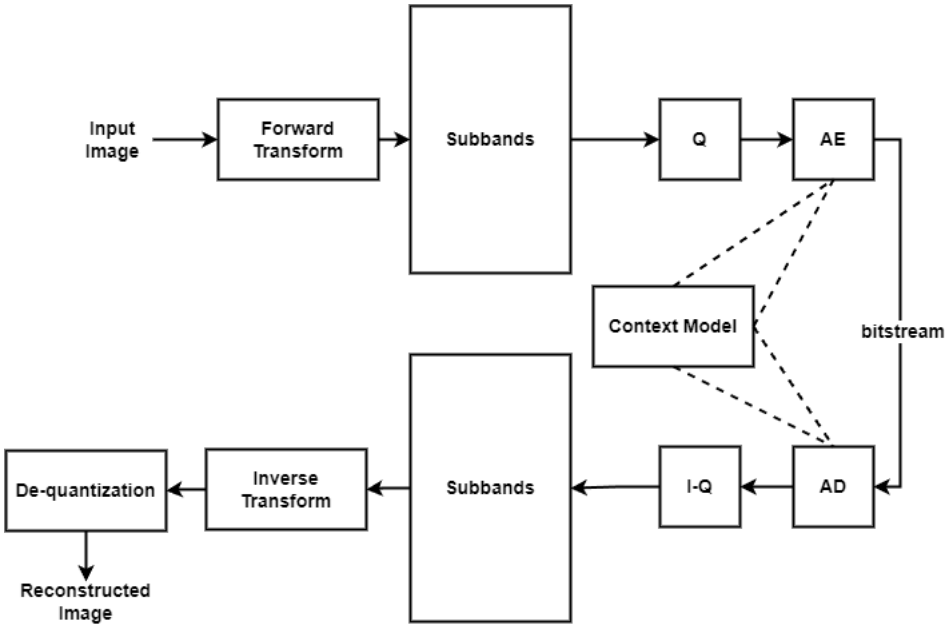

Figure 3.4: iWave++ architecture (redrawn from [8])

The lifting scheme in [8] is given in Figure 3.5. The letters and explanations are as follows:

- "S" stands for split
- "P" stands for predict filter
- "U" stands for update filter
- x_e and x_o stand for even and odd index of images after split respectively
- l_i and h_i stand for low and high frequency components after each lifting step respectively

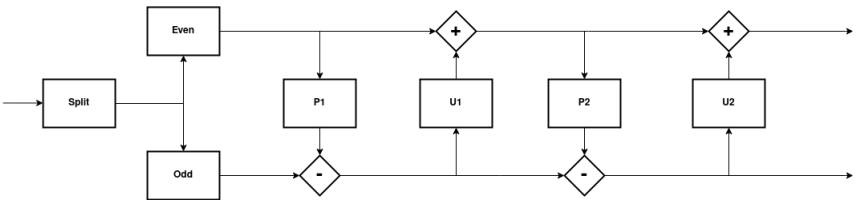

Figure 3.5: iWave++ lifting scheme (redrawn from [8])

The filters P and U are composed of CNNs, where their general structure can be seen in Figure 3.6. Some important points of design are as follows:

- Tanh linearity used as activation function
- Residual connections are used to preserve gradient and increase performance
- Same P and U filters are used in forward and inverse lifting transform

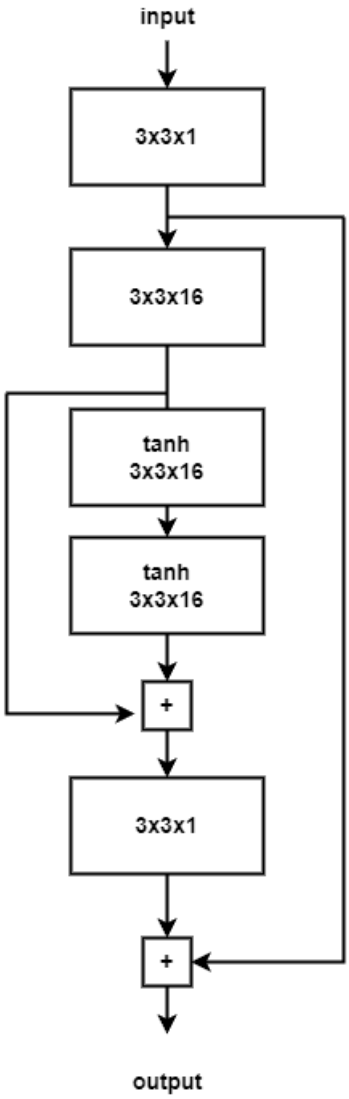

Figure 3.6: Predict and update filters in iWave++ (redrawn from [8])

Figure 3.7 shows how to use CNNs for both row and column-wise transform. It should be noted that there can be more than one lifting step in each lifting level. Figure 3.7

shows only 1 level lifting transform. First-row wise split is done to even and odd indices, then passed through N lifting steps, then resulting arrays are separated.

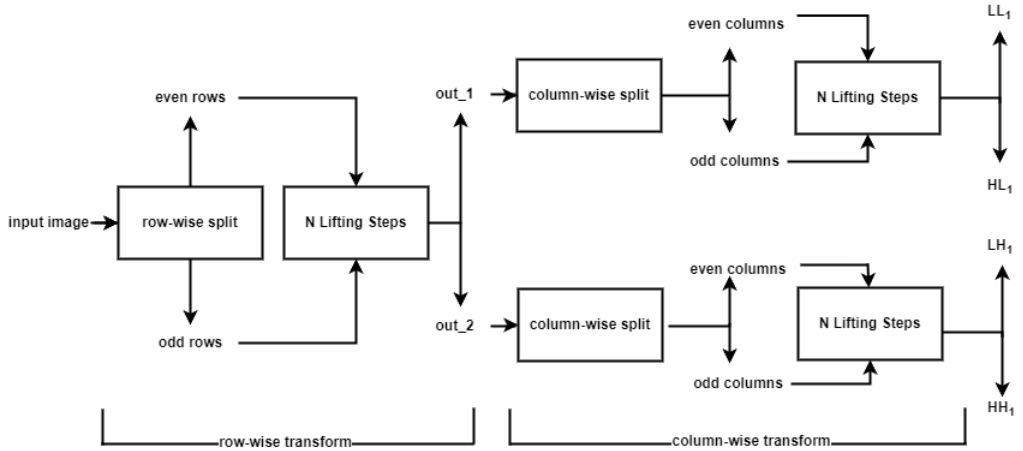

Figure 3.7: 1 level lifting forward transform (redrawn from [8])

Entropy modeling is complicated and heavy in [8]. LSTM network is used to model the entropy using the previous layers, which are called a context model. This context model outputs a set of entropy parameters for a probability model. For long-term context, LSTM model is used while for causal information, masked convolution is used. These models' outputs are fused at the context fusion model. This general structure will be similar in the proposed model to this model in the thesis work.

Lastly, the "dequantization" module, which is used for post-processing. The reason of using the dequantization module is that the filters in forward and inverse lifting transform are the same, and this constraint results in suboptimally reconstructed images [8]. B number of residual blocks are used in an internal network. This block increases quality in a lossy compression scheme, where quantization noise results in distortion.

Finally, they share that post-processing increases the performance significantly. The model with post-processing Iwave++ is better than HyperPrior-Mean model proposed in [40] while without post-processing, iwave++ is comparable with BPG-444.

3.3.5 Post Processing Networks

In lossy image compression, the distortion due to quantization is called quantization error or quantization noise. Quantization error is the main reason for distortion, which decrease in PSNR value. There are different methods to reduce this error. It can be split into two branches as follows:

- Post-processing isolated from rest of the compression system
- Post-processing in an end-to-end neural network scheme

Post-processing networks are generally used to increase the quality of the reconstructed image. To understand the effect of post-processing networks, some examples from the literature will be investigated.

One example for the post-processing network that is used for classical method HEVC is that, in [42], the authors proposed a post-processing network to increase the performance of the HEVC intra-coding scheme is proposed. HEVC is an adaptive block-based compression method, where JPEG is a block-based too, but it is not designed as an adaptive method. Different from the proposed networks for the JPEG post-processing, in [42], there are concatenation modules to capture information from different block sizes. It is suggested that using different sized CNN layers and concatenating them increases the performance, which boosts network's performance for different resolutions.

They observed around $\%4.6$ better BD-rate[43](in luminance), where BD-rate allows the measurement of the bitrate reduction offered by a codec or codec feature, while maintaining the same quality as measured by objective metrics.

In addition, in [44], the authors investigated the use of residual networks for the inception networks. According to their observations, there should be a multiplication factor that multiplies the main branch before two branches added, if residual networks have a high number of feature maps (channels) because it results in unstable performance in terms of gradients. Some observations are done in [45]. Thus, residual scaling with a factor of 0.1 is used. The same strategy is used in [8], for post-processing blocks.

In [36], the attention module, which has become very popular recently, is used for post-processing at low bit rate by adding the post-processing block after the classical compression schemes.

Attention module in this work uses both channel and spatial attention[46]. Channel attention is responsible for selecting semantic attributes, while spatial attention is responsible for finding important regions in the spatial domain. After both attention modules are used, the outputs are multiplied with the input image and concatenated. After concatenation CNNs and Relu blocks are used to improve performance, and lastly, a skip connection is added to make it a residual type network block. Multiple such blocks are cascaded.

According to [36], the proposed attention module for post-processing resulted in an increase in performance which is about %1 increase at the same rate.

3.3.6 Conditional Probabilities

Conditional probability is the likelihood of an event occurring given another event (conditional event) occurred. In image compression, the spatial dependency of pixel values can be used with conditional probabilities via the chain rule of probability, to obtain efficient compression results. One of the milestone contributions regarding dependency between pixel values is "Pixel Recurrent Neural Networks", which is proposed by Van Oord, Aaro and Kalchbrenner in their paper [9], while the paper mainly focuses on image reconstruction. However, the methodology can be used in image compression, too.

3.3.6.1 Probability Estimators: PixelRNN, PixelCNN and Gated PixelCNN

PixelRNN and PixelCNN are the methods proposed in [9], mainly for the image generation, using the dependency between pixel values. It depends on the idea of conditional probability because the probability of an image pixel value can be evaluated conditioned on known (obtained) previous pixels, which can be seen from Equation 3.7. In addition, the masked CNN approach can be observed in Figure 3.8. On the

left side of the Figure 3.8, known causal pixels are marked with blue while the red one, x_i, is the current pixel for which the probability distribution is calculated, in the middle of the Figure 3.8 an example for a multiscale context is given where pixels can be conditioned on subsampled image pixels with using dilation, and at the right of the dependency usage between different color channels can be observed where R values depend on just R values, G values depend on G and R values, and finally, B values depend on R, G, B values.

$$[h]p(x) = \prod_{i=1}^{N} p(x_i|x_{i-1}, ..., x_1, h) \tag{3.7}$$

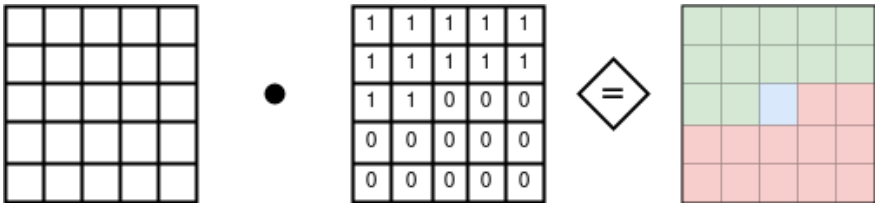

Figure 3.8: Masked CNN, the green ones for Mask CNN A, and when blue one is included it becomes Mask CNN B (redrawn from [9])

PixelRNN uses both LSTM modules[21] and masked convolutions to get the probabilistic information, while PixelCNN uses masked convolutions. PixelRNN is much slower than PixelCNN because it needs to process the pixel values individually, while PixelCNN can process pixel values in parallel. However, the PixelRNN can preserve all the spatial information along with the image, while PixelCNN preserves the regional information. However, in PixelCNN a problem called "blind spot" occurs, and the solution is proposed in [47]. In order to get rid of the "blind spot" problem, they proposed a method to carry the information along with the spatial domain while using CNN, called "Gated PixelCNN". They used different convolution blocks, one of them called "vertical block" and the other one called "horizontal block" and by using these blocks the information on previous rows and columns can be carried out using PixelCNN. All these methods are used to extract the probabilistic information of each pixel value, where this information can be used for different purposes. The masked CNN idea is an inspiring technique for the proposed thesis work, which will be mentioned in Section 4.

CHAPTER 4

PROPOSED METHOD

4.1 Overview

Compression methods are generally based on a structure given in Figure 1.1 that is given in Section 1.1 at the beginning of the thesis report. In this thesis, an end-to-end compression scheme is proposed, and it can be observed from Figure 4.1, the blocks in conventional image compression are replaced with neural networks. These blocks are investigated in detail in Figure 4.2 where 4 level neural wavelet transform and proposed architectures are shown. Different learning based methods are investigated for different parts of the proposed scheme, in this chapter, the one with the best performance is given.

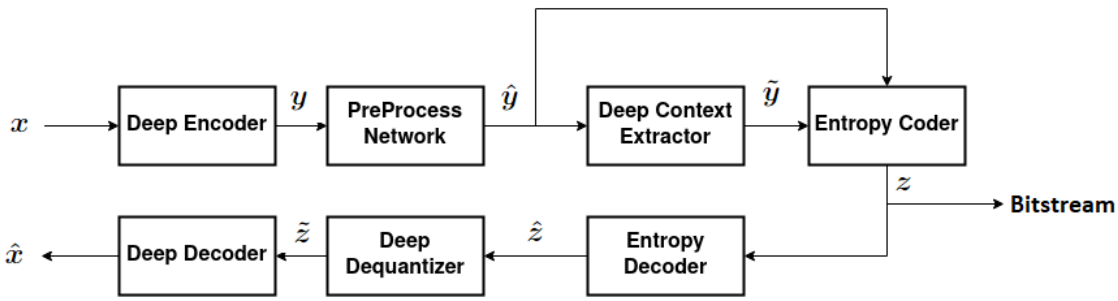

Figure 4.1: General structure of proposed scheme

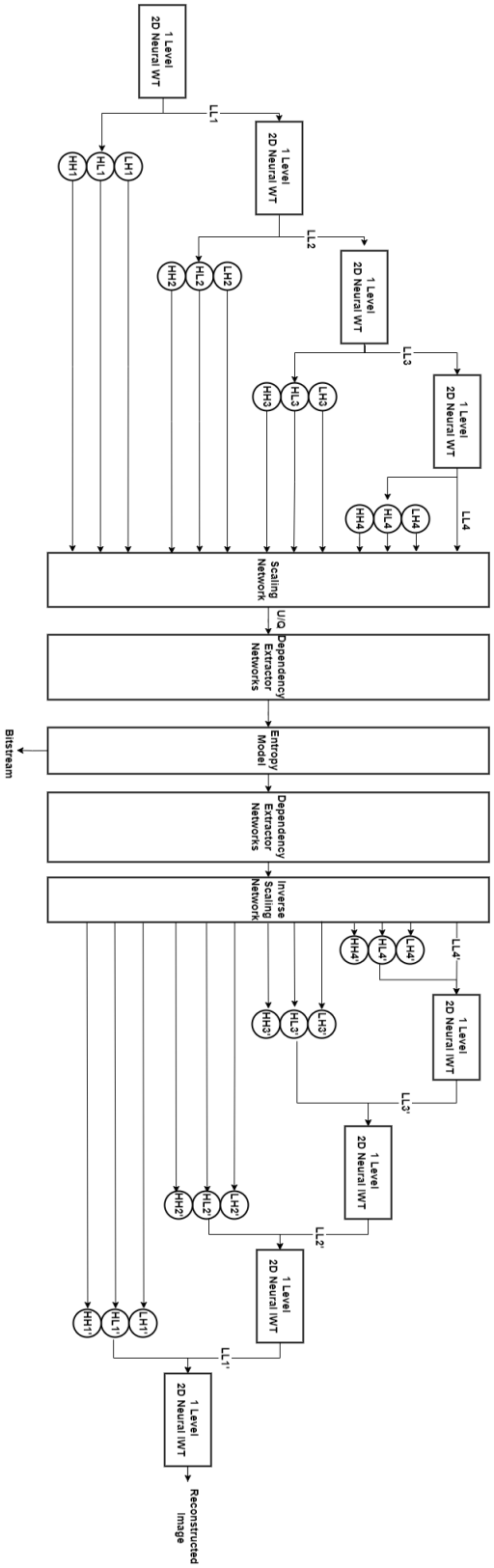

Figure 4.2: Detailed general structure of the proposed scheme

Short review of each block in Figure 4.1 is as follows:

- Deep Encoder/Decoder Blocks: CNN blocks inspired from lifting scheme are used in deep encoder and decoder.

- Preprocess Network Block: A deep CNN layer is used to obtain the accurate scales because output encoder and input of decoder scale must be accurate for efficient performance.

- Deep Context Extractor: Same and different level spatial dependencies are extracted and fused in this block with the help of mask CNN and EZWT structure by modeling neural networks.

- Entropy Coder: During training, i.i.d. noise, is used to model quantization, which is explained in [5], while in the test phase, rounding to the nearest integer is used. In addition, arithmetic coder is used as entropy coder during inference. Moreover, to model entropy, a Gaussian Conditional probability distribution is used.

Neural network models for all blocks will be given in the forthcoming sections, and all the proposed networks in the next sections, can be put in the blocks in the Figure 4.1 to obtain the final end-to-end neural network.

4.2 Architecture

4.2.1 Predict and Update Blocks

The lifting scheme, as it is mentioned in Section 2.3.1, is a method to perform wavelet transform. In Figure 4.3, a detailed structure of predict and update block is given. It should be noted that the general structure of predict and update block in[8] is preserved with the change of filter size. The given model is used as a block to replace the prediction and update blocks in the lifting scheme.

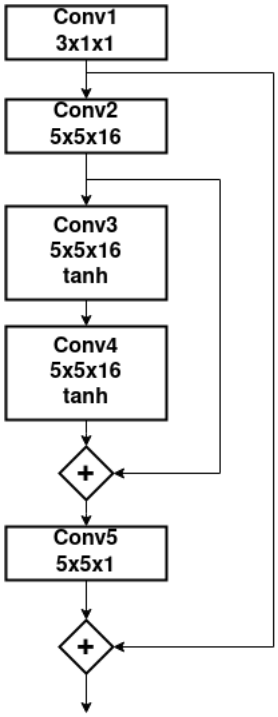

Figure 4.3: Predict and Update block scheme

4.2.2 Encoder and Decoder Blocks

In the proposed network, lifting steps are obtained by adding or subtracting prediction of update results, the complete lifting transform is invertible. In other words, original input can be reconstructed by going in reverse direction and addition with subtraction and vice versa. In Figure 4.4 1 level lifting wavelet transform consisting of 2 predict and update blocks can be seen. Since the operation consists of two stages, two different P and U networks are used and added serially, where these networks are used in the inverse transform (decoder) block, too.

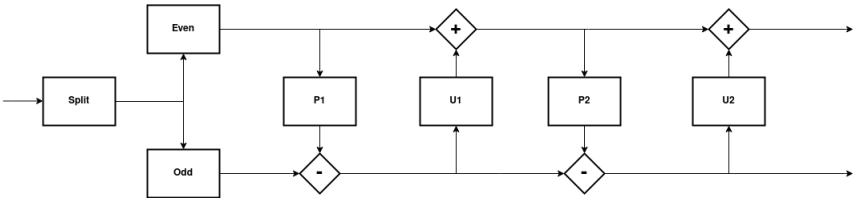

Figure 4.4: 2 stage lifting Transform 1 Level Wavelet

4.2.3 Scaling Network

Wavelet transform extracts different frequency components in each level, and the range of subbands in each level is different. In other words, the pixel value range of first-level wavelet subbands(LL1, HL1, LH1, HH1) differ from second-level wavelet subbands(LL2, HL2, LH2, HH2). In the proposed thesis work, quantization noise is modeled by using i.i.d uniform noise, where the pixel value ranges should be scaled in order to obtain the similar ranges in quantization. Otherwise, the i.i.d uniform noise can not model the quantization in rounding, so the model can not be trained to have good performance at the test phase. In Figure 4.5, the network architecture for "preprocessing" blocks can be observed. Both "a)" and "b)" are tested, as it can be observed that "a)" is a classic CNN while "b)" is a scalar CNN where group parameters of CNN is set to channel number of input, so each subband is scaled with the corresponding $1x1$ filter value. It is observed that choosing "b)" has less complexity when compared with "a)" while "a)" increases performance, so the structure "a)" is used in the proposed scheme. The CNNs, in right-hand side of the given Figure 4.4, are modeled using separable convolution along channel and spatial dimensions by setting the groups parameter in pytorch implemented convolution layers. The designs of scaling blocks are conducted using a trial and error approach. Non-linearity is introduced to increase the complexity of the network, while $3x3$ filters are selected to use the minimum number of parameters in each convolution block.

The scaling network is used in both encoder and decoder part. The scaling network is used in the encoder part in order to fine-tune the neural wavelet outputs. In addition, the scaling network in decoder part is used in order to reduce the quantization noise which is introduced by quantization block. The quantization noise comes from the uniform noise in training and rounding in inference phase.

4.2.4 Deep Context Extractor

A deep context extractor is a block that consists of two different deep learning blocks that can be seen in Figure 4.6. The first block is responsible for extracting causal dependencies, while the second block is a levelwise dependency extractor that ex-

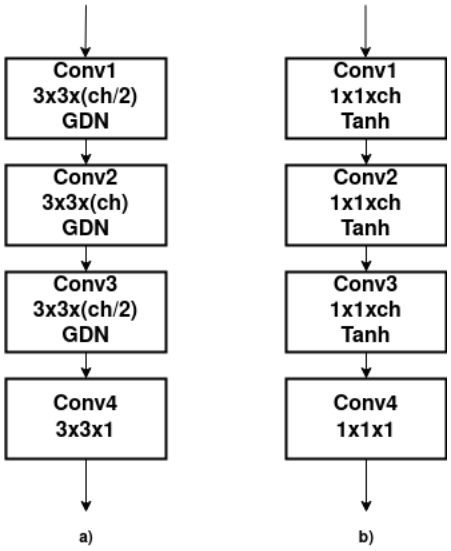

Figure 4.5: Preprocessing Blocks, a) Traditional CNNs, b) 1x1 scaling CNNs

tracts the dependency of same subbands in different levels of the wavelet transform, this structure is inspired from the classical method called Embedded Zero Tree, Wavelet(EZWT) coding[4]. Similar works are generally focused on more complicated methods such as RNN, or hyperprior. In the proposed scheme, an idea that is used in conventional methods, JPEG2000, is tried to be modeled using the deep learning methods in order to increase the efficiency compared to the conventional method.

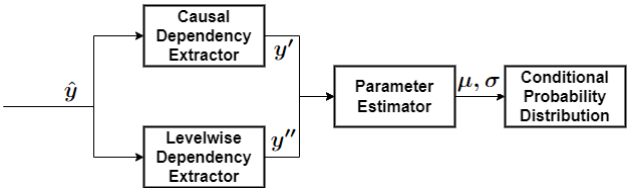

Figure 4.6: Deep context extractor block

4.2.4.1 Causal Dependency Extractor

In this block, mask CNN approach is used which can be observed in Figure 4.7, which is mentioned in details in Section 3.3.6.1 where the reason for using mask CNN is to carry the dependency of the neighbors to current pixel by using a CNN with a masked

approach, where its name comes from. In the proposed work, both "Mask A" and "Mask B" are used to obtain causal dependencies in spatial and channel domains, and the output is used as an input to the "Parameter Estimator" block. The main reason of using Mask CNN blocks rather than CNN is that in compression at the decoder side the pixel values should be decoded without using the values of unknown pixels, (i.e. in a causal order) so Mask CNN preserves causality.

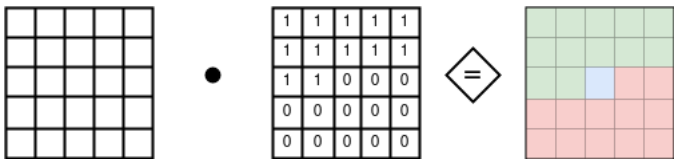

Figure 4.7: Mask CNN approach

The proposed network for "Causal Dependency Extractor" block can be seen in Figure 4.8. The reason of using first "Mask A" then using "Mask B" is because "Mask A" does not take the information of current pixel and in a compression scheme the probability of the current pixel should be obtained by using the previous pixels. After "Mask A", "Mask B" can be used because at the next layer, the value at the current pixel is not the actual value, it is the value obtained by the "Mask A" by using the neighbors which is showed in Figure 4.7, with a filter size of 5. The channel number depends on the desired network model, and can be chosen based on the desired complexity and success trade-off. In addition, the proposed design, using one Mask Conv A and four Mask Conv B can be changed according to expected performance and network depth. A few experiments are conducted to find an optimum solution. First Mask Conv A block is used to avoid the current pixel information, and after that non-linearity is used for better data fitting. Then, Mask Conv B filter is put to increase the order of the proposed architecture. When the Mask Conv B layers are removed, it is observed that performance is decreased by approximately $\%2$.

Causal dependency extracting process can be defined as in equation 4.1, where θ represents the weights of the filters proposed in Figure 4.8.

$$y' = f_c(\tilde{y}; \theta) \tag{4.1}$$

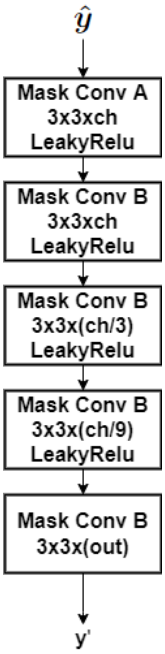

Figure 4.8: Spatial dependency extractor

4.2.4.2 Levelwise Dependency Extractor

It is mentioned that there is a dependency between the subbands at different levels in wavelet transform because wavelet transforms iteratively processes LL subbands which inherit properties of previous level. Levelwise Dependency Block is designed with inspiration from EZWT[4], mentioned in Section 2.3.2, where the dependency between ancestors and descendants is used in probability modeling to increase the efficiency of entropy coding. The deep learning structure that is used in the Levelwise Dependency Extractor block can be observed in Figure 4.9. First the pixel of next wavelet level is up sampled to a $2x2$ block by filling the block with same values which can be observed in Figure 4.10, then two convolution blocks are used to extract the necessary information using the up sampled levels. Lastly, non-linearity is added in between convolution layers to extract nonlinear relations between different levels of the same subbands. The usage of CNN filters and non-linearity in between them is inspired by [4]. In [4], there is a threshold value and this threshold value introduces non-linearity. Rather than using a threshold value, using CNN filters and using non-linearity in between them is expected to behave similar. In addition, further CNN

blocks may increase the performance, but in this design it is taken as minimal as possible.

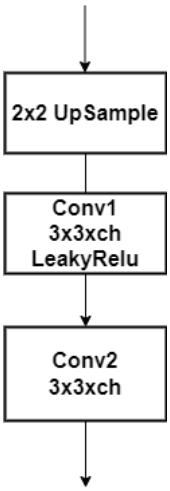

Figure 4.9: Levelwise dependency extractor

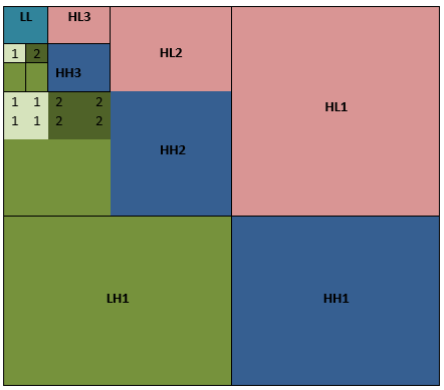

Figure 4.10: Example upsampling

This block can be represented as in Equation 4.2, where γ represents the parameters in the levelwise dependency extractor network.

$$y'' = f_l(\tilde{y}; \gamma) \tag{4.2}$$

4.2.4.3 Parameter Estimator

The previous two sections explain how the casual and levelwise dependencies are extracted while these dependencies are merged in the "Parameter Estimator" block, where the outputs of these blocks are fused to estimate mean and variance for each pixel value, to model the distribution at "Probability Distribution" block. In Figure 4.11, the network is proposed. The reason to use $1x1$ CNNs is again causality. Since the input to this parameter estimation block comes from the causal dependency extractor block, causality is preserved using Type A Masked convolution block or $1x1$ convolution blocks.

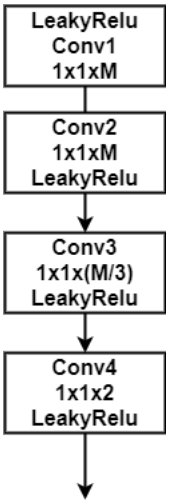

Figure 4.11: Parameter estimator block

Equation 4.3 shows the corresponding mathematical form of the proposed scheme for parameter estimator, as it can be observed from the same equation, outputs of both casual dependency extractor block(y') and levelwise dependency extractor block(y") are used in parameter estimator block where the ρ represents the weight parameters in the parameter estimator block, that is updated during training.

$$\mu, \sigma = f_p(y', y''; \rho) \tag{4.3}$$

4.2.4.4 Probability Distribution

The output of the parameter estimator is μ, σ for each variable of the wavelet transform subbands, which are used to model the probability distribution of each variable with Gaussian distribution. A Gaussian distribution is represented as $\mathcal{N}(\mu, \sigma^2)$. The distribution for each pixel is shown in Equation 4.5. The approach is shown in Equation 4.4 where x_i represents the current wavelet transform variable and X is the vector of values that represents the wavelet transform variables which have information about the current pixel value and are in the receptive field of the levelwise and spatial dependency extractor blocks, the information comes from models that use these dependencies.

$$p(x_i) = f(x_i|X) \tag{4.4}$$

The function $f_{context}$ stands for the deep context extractor block, includes the learnable parameters, and the last partition of context extractor is a conditional Gaussian normal distribution where final probability distribution obtained in this block shown in Figure 4.5.

$$p(x_i) = \mathcal{N}(\mu_i, \sigma_i) \tag{4.5}$$

4.2.5 Quantization and Entropy Coding

In the proposed method, quantization is handled in 2 different ways. As it is mentioned in the Section 2.1.4, the power of neural networks comes from gradient descent based learning, while rounding results in the loss of gradient flow. Thus, in training i.i.d uniform noise in range [-0.5,0.5] is used while in test (inference) phase, rounding is used for quantization. As an entropy coder, an arithmetic coder is responsible to generate a bitstream.

4.3 Post Processing

Post-processing is a method to increase the reconstructed image quality of the compression methods because of the introduced error by quantization noise. There are different methods to increase the quality of images after compression, which can be called denoising networks. In thesis work different denoising networks were tried in order to increase the performance of compression scheme such as [48], [8], [49]. However, it is observed that using a post-processing block helps significantly only at low bit rates if it is trained end-to-end, which means they tend to behave like a low pass filter. Thus, in the proposed scheme, post-processing units are not shown because at high frequencies, it does not show the expected behavior and decreases the proposed model performance at high bit-rate, PSNR values.

CHAPTER 5

EXPERIMENTS AND RESULTS

This chapter gives information about different experiments and the results of the proposed method.

As it is mentioned in the previous sections, the proposed method is mainly inspired by the conventional zerotree-like wavelet transform method and neural network-based implementation of a lifting wavelet transform[8]. One of the main contributions of the proposed work is to use the dependencies of subband pixel values, which correspond to wavelet coefficients, between different levels with the help of deep learning models, which are inspired by the conventional work of EZWT[4]. During thesis work, different methods are tried to achieve a better rate-distortion loss, and these methods are investigated in detail in the forthcoming sections. Firstly, different ideas of entropy modeling methods and networks are investigated. Then, some post-processing models are added to the end of the proposed algorithm to improve the performance of the proposed scheme. Lastly, the comparison of the proposed methods with the state-of-the-art methods in the literature is given in Section 5.2.

5.1 Ablation Study

In this section, different methods to improve the rate-distortion loss will be investigated. Each proposed model is illustrated with the help of figures. The examples for entropy modeling are given by using the LH1 and LH2 subbands, where LH2 stands for the second level wavelet transform coefficients of LH subband while LH1 stands for the first level wavelet transform coefficients of LH subband. All wavelet coefficients are used in the algorithm, but for the visualization of algorithm examples only

LH1 and LH2 are used for simplification. Then, post-processing networks are added at the end of the compression scheme to see the effect of post-processing. After the methods are investigated, a comparison of the proposed methods in terms of PSNR vs bpp is given.

5.1.1 Entropy Modeling Methods

In this section, different entropy modeling networks are investigated while preserving the rest of the end-to-end model remains the same. In Table 5.1, the number of parameters of each proposed compression model with the specified entropy model and number of parameters of state-of-the-art solutions can be observed. It should be noted that because of the causal approach during the spatial dependency extractors, "block-based" and "causal+ezwt" probability modeling approaches are slower than others. In this thesis work, main purpose is to find a different way and propose a new systematic to literature, encoding and decoding steps can be compared with other algorithms after computational optimizations are done.

Table 5.1: Number of parameters for proposed entropy models

Model Definition	**Parameter Number**
Factorized*	9.41*10^6
Block-based EZWT*	11.01*10^6
Only EZWT*	14,94*10^6
Causal + EZWT**	16.94*10^6
mbt-2018_mean,D.Minnen et al. 2018[50]	6.70*10^6
bmshj2018-hyperprior[50]	4.84*10^6

5.1.1.1 Factorized Model

In the factorized entropy model, dependencies between pixel values are not used. Each pixel is entropy coded using its own value. This method is called Factorized Model.

5.1.1.2 Neural EZWT Dependency

In neural EZWT dependency, deep learning models are used only to model the dependency of the ancestors and descendants, as it is mentioned in Section 2.3.2. Different from the model that is proposed in Section 5.1.1.3, the causal dependency (masked CNNs) are not used to observe the effect of spatial (neighborhood) dependencies between the wavelet coefficients at the same subband. The last level wavelet coefficients are used in a factorized entropy model as in the previous section, in other words dependency for last level wavelet coefficients are not used. Only the EZWT kind of dependency is used for the lower levels of wavelet coefficients.

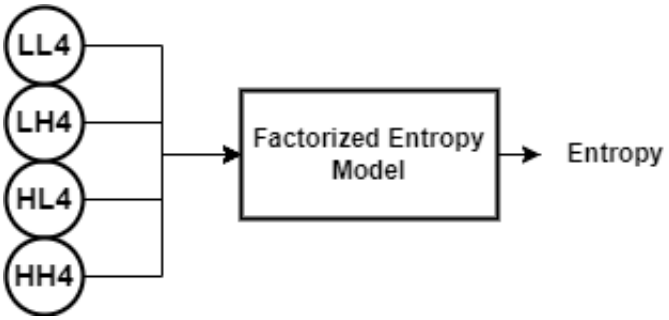

Figure 5.1: Schematic of causal and EZWT dependency extractor for last level wavelet-like coefficients

Except for the last level wavelet coefficients, since they do not have any descendants, other coefficients at different levels are modeled using the method that is shown in Figure 5.2. Only levelwise dependency, which EZWT inspires, is used to obtain the correct mean and scale parameters for the corresponding wavelet coefficient.

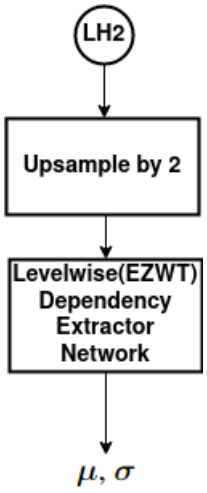

Figure 5.2: Schematic of causal and EZWT dependency extractor other than last level wavelet coefficients

5.1.1.3 Causal and EZWT

This entropy model is the one that is proposed in Chapter 4. In this entropy model, both EZWT kind subband dependency and spatial causal dependency between neighbor wavelet coefficients are used to obtain the best probability model for efficient entropy coding. This approach can be divided into two branches, as seen in Figure 5.3. In Figure 5.3, an example scheme is given to illustrate how the entropy model works for a pair of subbands at different wavelet levels. Firstly, the ancestor pixel (wavelet coefficient) is up sampled to a $2x2$ block by filling the $2x2$ block values with the ancestor wavelet coefficient value, which can be observed in Figure 4.10, where it is called nearest-neighbor interpolation in literature. All the wavelet coefficients at that subband are up sampled by 2. Then neural network models are used to extract levelwise dependency, which is given in Section 4.2.4. Secondly, neural network models consist of mask CNNs are used to model the dependencies between the current pixel and the neighboring pixel values. After those operations, the outputs of the causal dependency branch and levelwise dependency branch are concatenated subbandwise and given as an input to a parameter estimator network, whose output is the mean and scale for the probability distribution for each pixel value.

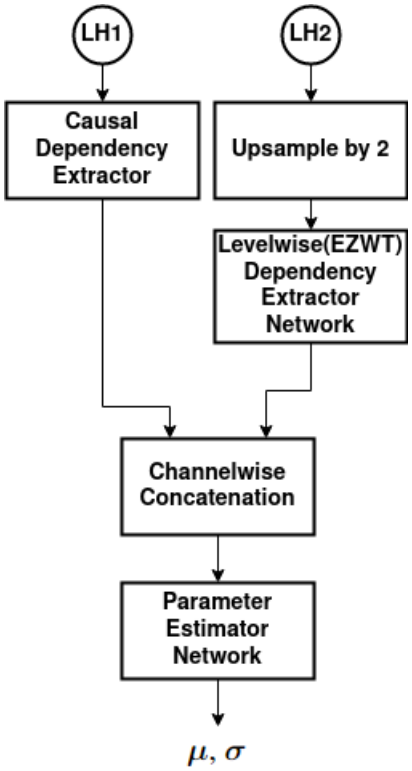

Figure 5.3: Schematic of causal and EZWT dependency extractor

5.1.1.4 Block-based EZWT and Causal Dependency

The previous method, uses both causal and EZWT dependencies and takes long periods to decode because of the dependencies of each wavelet coefficient to another while using mask CNNs. In the decoding side, the process is not highly parallel, which results in long time periods to decode. Thus, a method that makes the process faster than the method given in Section 5.1.1.3 is implemented and proposed. In Block-based EZWT, both causal and EZWT dependencies are tried to be modeled using a different method. The general scheme for block-based entropy model can be seen in Figure 5.4.

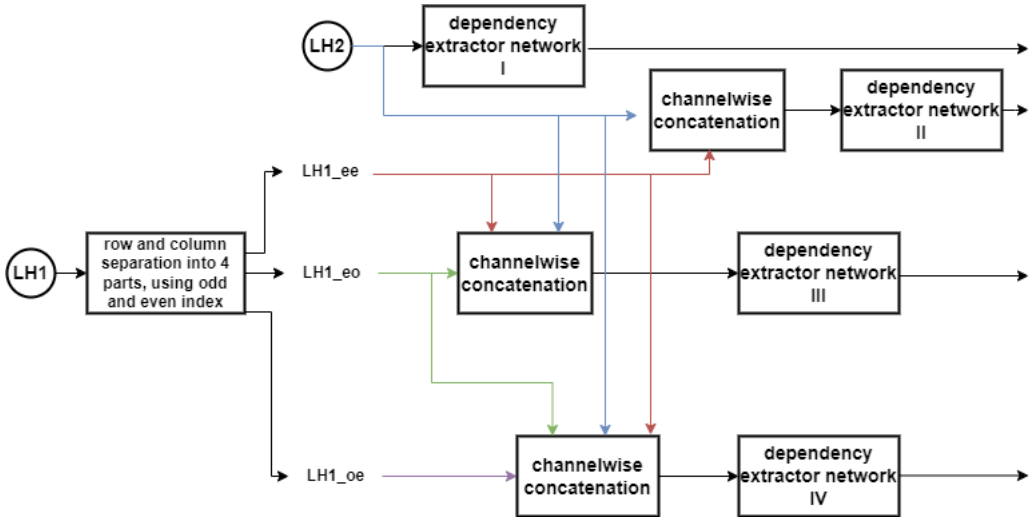

Figure 5.4: Schematic of block-based EZWT and causal dependency extractor

In this method, causal and EZWT dependencies are used differently that can be observed in Figure 5.5. The steps are as follows:

- Firstly, the next level($LH2$) wavelet coefficients are used as input to the dependency extractor network, the output corresponds to the mean and scale of even row and even column(ee) of the $LH1$

- Secondly, $LH2$ and $LH1's$ even row, and even column wavelet coefficients are concatenated and are given to dependency extractor II for the mean and scale of even rows and odd columns of $LH1$

- Thirdly, ($LH2$), $LH1_{ee}$ and $LH1_{eo}$ are concatenated and are given as input to dependency extractor III to obtain the mean and scale of odd rows and even columns of $LH1$

- Lastly, $LH2$, $LH1_{ee}$, $LH1_{eo}$ and $LH1_{oe}$ are concatenated and are given as input to dependency extractor IV to obtain the mean and scale of the odd rows and odd columns of $LH1$

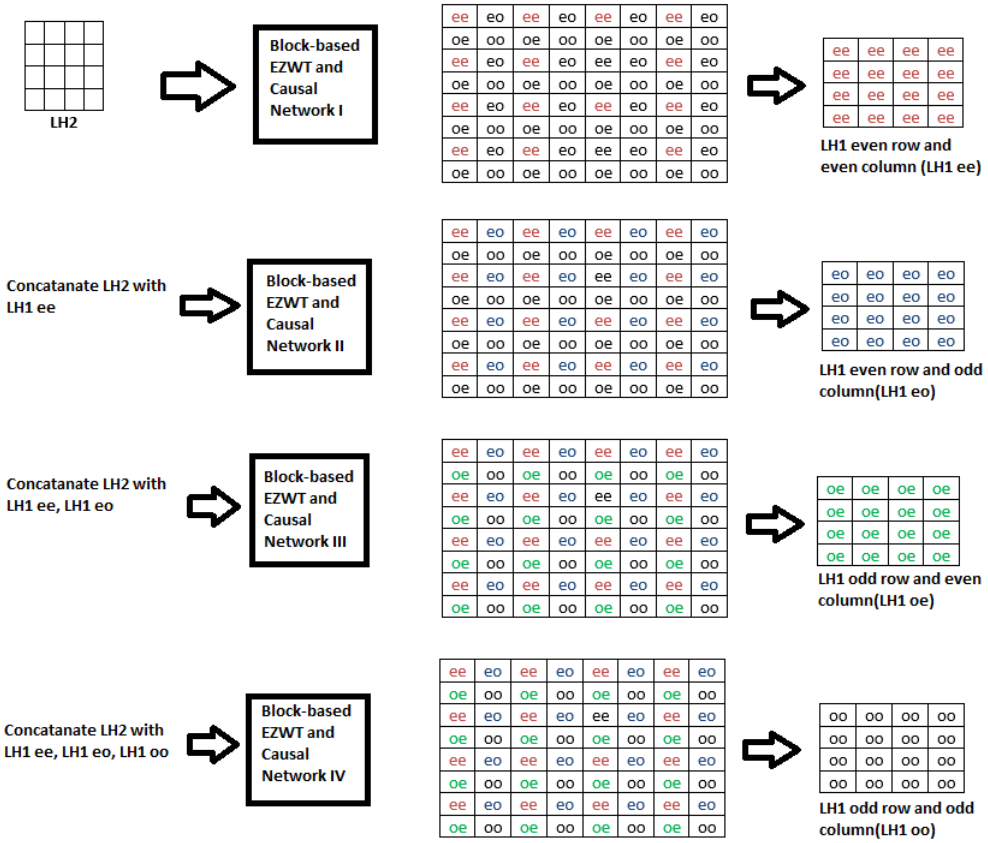

Figure 5.5: Visualization of block-based EZWT and causal dependency extractor

As it can be seen from the above steps, rather than a sequential process, a more parallel process is used in the block-based approach. In other words, the multiples of mxn blocks are estimated at the same time for both mean and scale. Each dependency extractor block consists of two different neural network models, mean estimator and scale estimator, which can be seen in Figure 5.6. The output mean and scale for each block (even/odd rows and even/odd columns) are merged to obtain the original size of the image.

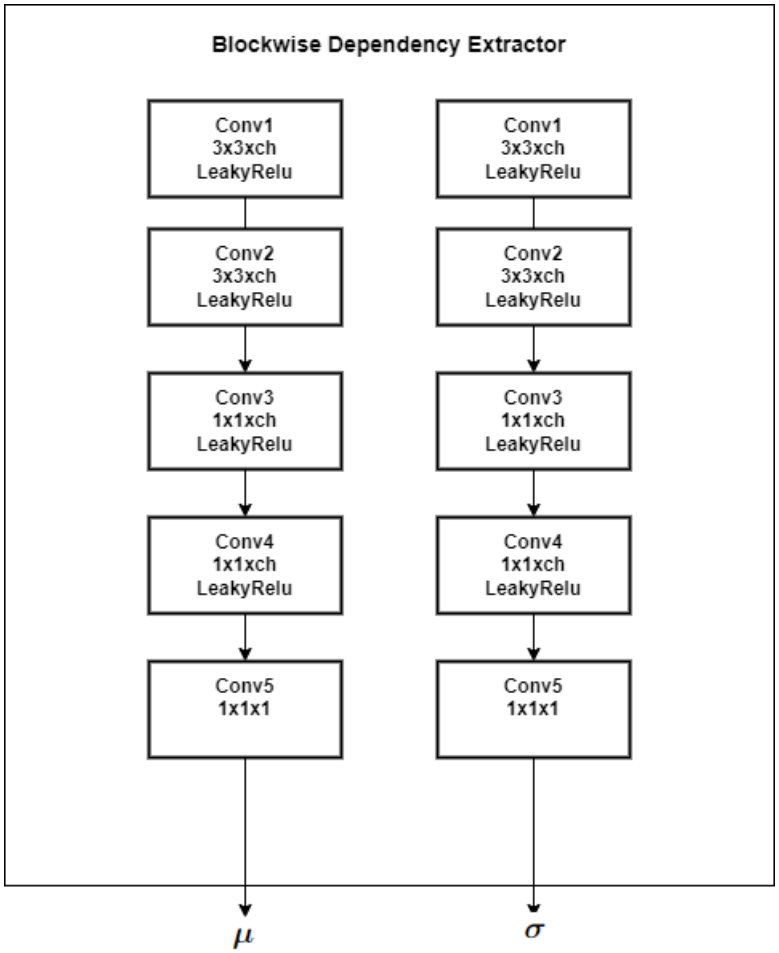

Figure 5.6: Schematic of blockwise causal and EZWT dependency extractor

5.1.2 Post-processing Methods

The reason to use post-processing methods in an image compression scheme is given in Section 3.3.5. During the thesis study, different post-processing networks are used in order to reduce the noise, which results in an increase in PSNR values at the same low bit-rate. Similar methods that are used in [8], [36], [48] are implemented for post-processing model. These methods are tested for image compression performance, and it is observed that using these methods results in similar results. Thus, the investigations in the forthcoming sections are done using a similar architecture which is used in [8] and it can be seen in Figure 5.7.

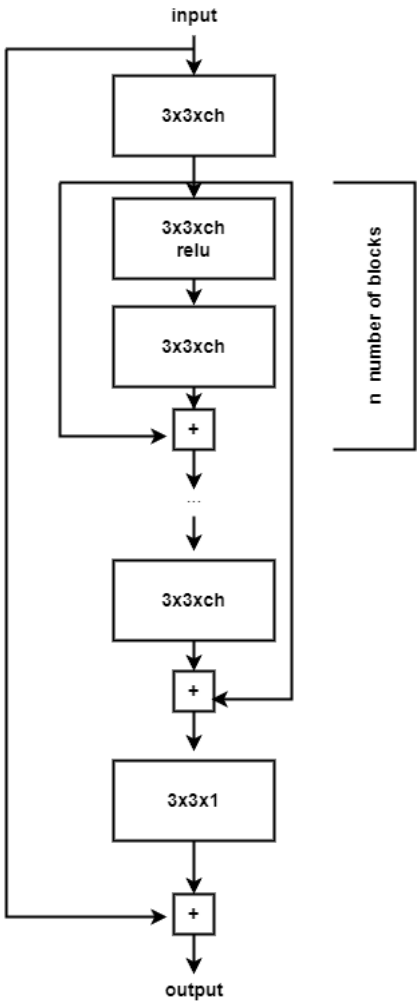

Figure 5.7: Proposed post-process block in [8]

In Figure 5.8, two different training strategies(a, b) are given. Both strategies are investigated in the upcoming sections. According to the observations that are done in this thesis study, using the below strategy(b) gives coherent results, where in literature it is called "transfer learning". However, in experiments, it is observed that using the scaling network that is given in Section 4.2.3 appropriately minimize the noise which is introduced by quantization. Since, the post-processing methods do not increase the performance significantly in the proposed compression scheme, they are not added to the method.

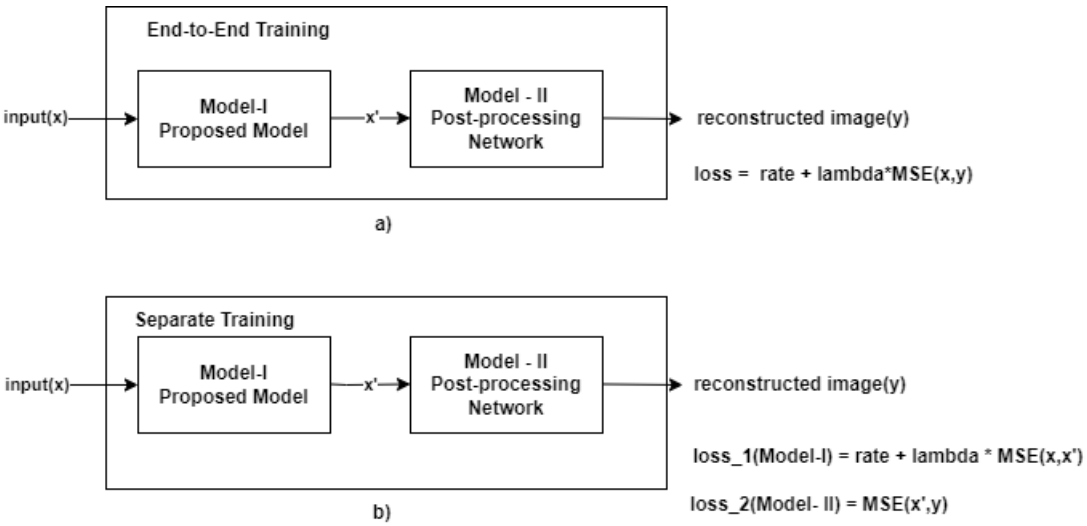

Figure 5.8: Different training strategies for post-processing networks

5.1.2.1 End-to-end Training with Post-processing Model

In Figure 5.8, the above one(a) shows an end-to-end compression scheme where the post-processing network and the proposed compression scheme are trained and optimized using one primary loss function, which is the rate-distortion loss function given in Equation3.1. The motivation for using end-to-end training is to optimize the network by using one loss function. However, the results show that the transfer learning strategy shows a better performance in the proposed compression structure. The expectation is that the noise which is introduced by the convolutional layers, activations and quantization can be minimized with the post-processing unit so that the MSE decreases and PSNR increases. It is observed that at low bit-rate(0.13 bpp) post-processing networks increase PSNR and decrease MSE. In comparison, at high bit-rate(2 bpp), post-processing networks behave like low pass filters, which decrease the PSNR and increase the MSE because at high bit-rate, the high-frequency components are essential for the details of the image.

5.1.2.2 Separate Training with Post-processing Model

In Figure 5.8, the below one(b) shows a two-step training strategy, where the Model-I represents the proposed compression scheme. It is trained and optimized using

the rate-distortion loss function and after the training phase is completed the trainable parameters of the Model-I are frozen, their gradients are cancelled. Then, the post-processing model(Model-II) is trained and optimized by using the MSE as loss function where the output of the proposed model (decompressed image, x') and the original input image(x) are used to calculate the MSE. This strategy is called "Transfer Learning" in literature; basically, after a model is trained, the trainable parameters of the trained model are frozen and fixed. Then another model is attached to the trained model and this model is trained according to the given new loss function, in this case MSE.

It is observed that using this strategy does not reduce the PSNR value at high bit-rate(2 bpp) while increasing the PSNR value at low bit-rate. Rate does not change during the optimization of the second model because in loss function II, the rate term does not exist, so gradients are not used to optimize the rate term.

5.2 Results

5.2.1 Proposed Compression Scheme Method Results on Different Dataset

In compression literature, Kodak, Technic, and Clic datasets are generally used to test performance of proposed methods' performance. In this thesis work, training is done using Vimeo Triplet dataset and testing is done using Kodak dataset. In the below Figure 5.9, the results of 4 different variations of the proposed network with different entropy models, which are given in Section 5.1.1 are compared with the methods that are already proposed by different researchers in the literature and the source for these results is [50]. As it can be observed, the best results in proposed methods belong to the model that uses both causal and EZWT dependency. The factorized model is the one that does not use any probabilistic dependency. It is the base method for entropy modeling in the proposed architecture. All proposed entropy models show better than Factorized Method's performance in terms of PSNR vs bpp on Kodak Dataset. In addition, Factorized Method shows similar performance with JPEG2000. It should be noted that the causal EZWT method shows similar performance to BPG(4:4:4) and the BMSHJ Hyperprior model, which are much better than JPEG2000 in terms

of PSNR vs bpp. Thus, in this thesis work, the neural network model to model EZWT architecture shows better performance in terms of PSNR vs bit-rate when compared with the classical approach.

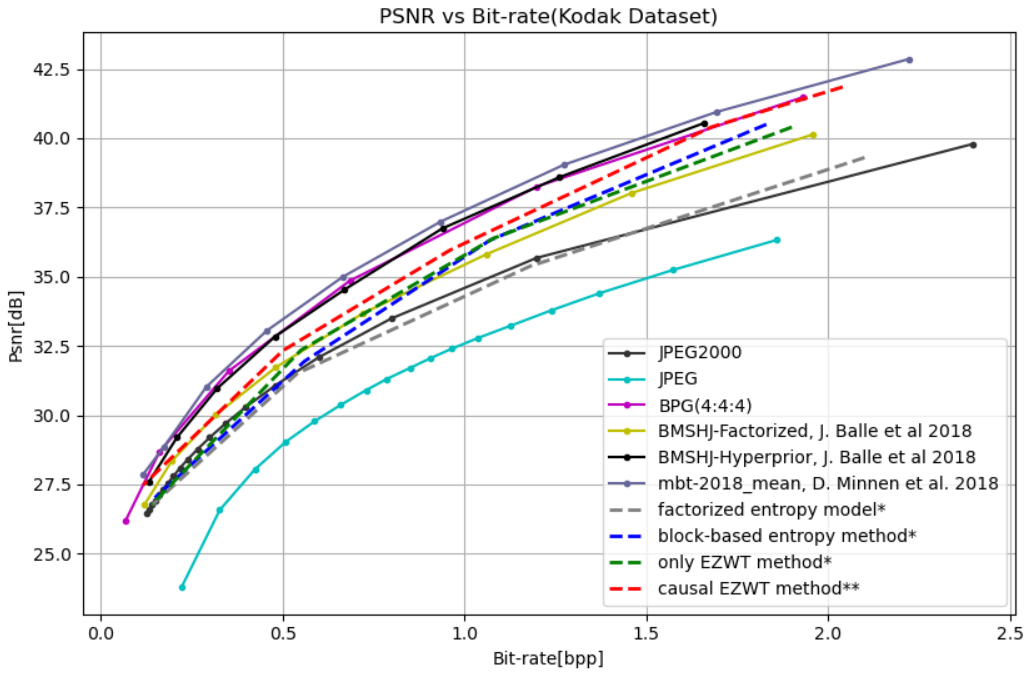

Figure 5.9: PSNR vs bpp, Kodak Dataset

5.2.2 Proposed Compression Scheme Method Visual Results

Two different images are selected from the Kodak image dataset to show the compression results, which can be seen in Figure 5.10. Original Image 1 is an easier example for compression when compared with Original Image 2 because it has less high frequency components. In Original Image 2, there are a lot of edges. Thus, after compression, the rate of Original Image 1 is higher for than Original Image 2 at the same PSNR.

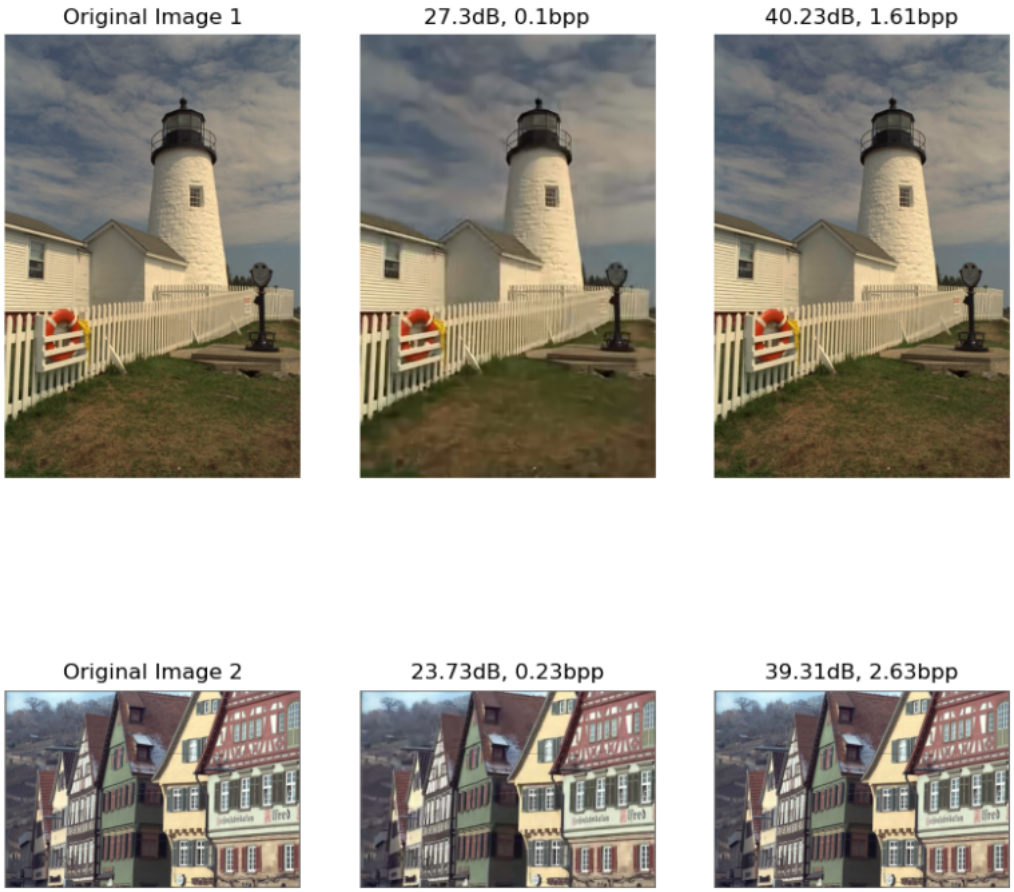

Figure 5.10: Perceptual quality at different bit-rates

5.2.3 Proposed Compression Scheme Method Visual Results of Wavelet Coefficients

In wavelet transform, as it is mentioned in Section 2.3, each level corresponds to 4 different subbands that represent different directional properties, and details of the image. In the proposed method, a neural network model is used in a lifting scheme to extract wavelet coefficients. In order to understand the efficiency of those filters, visualization of the wavelet coefficients of a sample image from Kodak image dataset using the trained weight of the proposed model is shown in Figure 5.11. The original image, that is transformed to obtain wavelet coefficients, can be seen in Figure 5.13. Since this image has a lot of high-frequency components, such as edges, and

corners, it is selected as a sample image to observe the meaning and idea of wavelet coefficients in different wavelet levels.

In Figure 5.11, it can be seen that the wavelet coefficients in the upper left corner, which are low-frequency coefficients of the image, have most of the main components of the image while most of the details, high-frequency components are stored in other subbands. In addition, it can be seen that the distribution of the wavelet coefficients changes at each level because wavelet transformation is an iterative method to transform the image into multiple lower resolutions. When these coefficients are used in an inverse lifting wavelet scheme, an image can be reconstructed without loss. Moreover, in Figure 5.12, wavelet coefficients of the same image from Kodak image dataset at low bit-rate is shown. The main difference between the high bit-rate and low bit-rate wavelet coefficients is that the high frequency wavelet coefficients are not significant for low bit-rate compression while for high bit-rate compression high frequency wavelet coefficients are more prominent. Lastly, it can be seen that the subbands show different features whose directions are different, as it is mentioned in Section 2.3. Thus, using CNNs show similar behavior like classical wavelet transform.

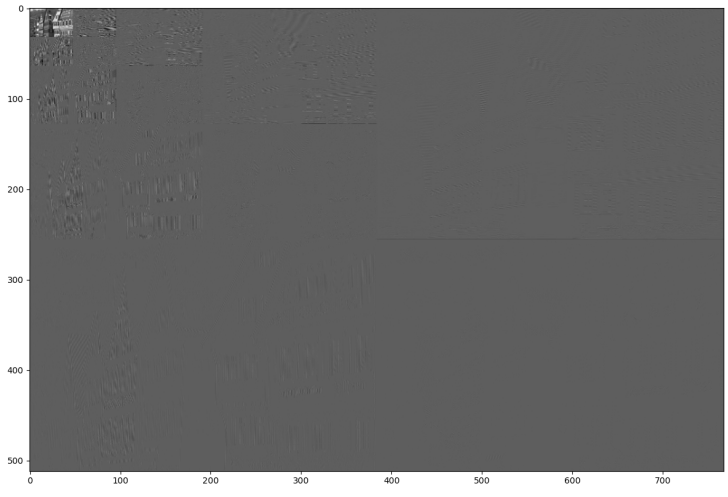

Figure 5.11: Wavelet coefficients that are obtained by the neural network model at high bit-rate for Y channel(2bpp, 40PSNR)

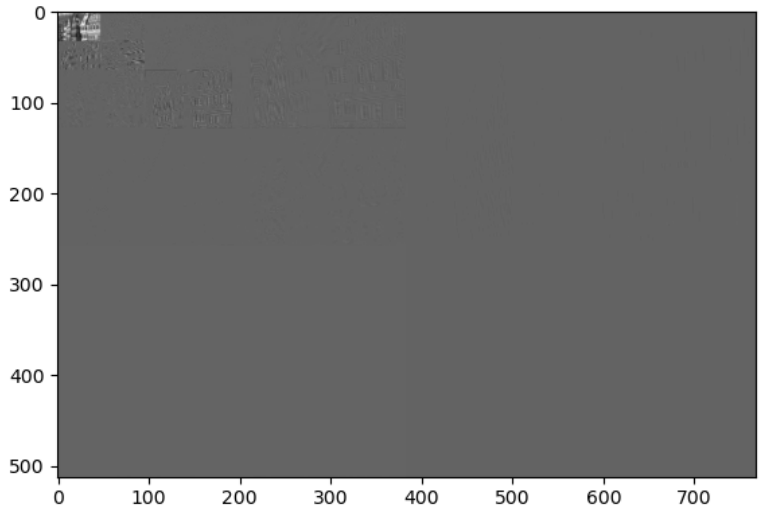

Figure 5.12: Wavelet coefficients that are obtained by the neural network model at low bit-rate for Y channel(0.23bpp, 23.73PSNR)

Figure 5.13: The image which is used to illustrate the wavelet coefficients

CHAPTER 6

CONCLUSION

In conclusion, in the thesis work, firstly, an introduction, in Chapter 1, of image compression is given to the reader; secondly, preliminary knowledge, in Chapter 2, about the methods and algorithms are shared. Thirdly, in Chapter 3, mostly state-of-the-art algorithms and proposed works are investigated in detail. Fourthly, the proposed method is investigated and shared in Chapter 4. Lastly, the ablation study with the experiments and results are shared with the advantages and disadvantages of the proposed method in Chapter 5.

The proposal and the contribution to the literature of thesis work is mainly designing a neural network-based architecture that can fit the general architecture of EZWT approach, which uses a lifting wavelet approach and levelwise subband dependency. The proposed method shows significantly better results than JPEG2000 compared in terms of rate and distortion. It reaches the performance of BPG 4:4:4. The limitations and advantages of the proposed scheme are shared. Moreover, modeling the dependency in between pixel values with the help of the neural EZWT increases the performance when compared with factorized (without dependencies) approach. This means there exists a meaningful levelwise dependency in between wavelet levels. Lastly, this is the first use case of EZWT in compression using learning based methods. The proposed architecture is unique in terms of entropy modeling.

The PSNR value of the proposed method is around $\%5$ better than JPEG2000 at low bit rates, while at high bit rates, this ratio goes up to $\%10$ on Kodak Image Dataset.

6.1 Future Works

Many methods are implemented and tested in order to obtain the proposed end-to-end compression scheme. While some of them show poor performance, some of them show promising performance, where one primary method which can be added to the proposed scheme is an adequate post-processing network, which can reduce the quantization noise which is introduced by the quantization block. However, implemented methods only boost the performance at low bit rates, behaved like a low pass filter, and performance decreases at high bit-rates because at high bit-rates high frequency components are the important ones. In addition, the hyperprior methods show better performance than factorized methods in the literature, so the hyperprior approach can increase performance. Moreover, the causal dependency extractor block forces the network to be sequential, so it takes time at decoding, better approaches can be investigated to decrease decoding time. Furthermore, as a future work, different metrics such as SSIM and LPIPS can be used to test the performance of the proposed scheme. Lastly, hyperparameter tuning can be done to increase the performance. It should be noted that the proposed thesis work is a proposal of a method, by hyperparameter tuning performance can be increased up to a level.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[2] F. Serzhenko, "Fast discrete wavelet transform on cuda," Dec 2019.

[3] N. Jaswal, A. Panghal, and A. Garg, "Compression of image using 2d-dwt based on lifting scheme," 01 2010.

[4] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on signal processing*, vol. 41, no. 12, pp. 3445–3462, 1993.

[5] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," *arXiv preprint arXiv:1611.01704*, 2016.

[6] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," *arXiv preprint arXiv:1802.01436*, 2018.

[7] H. Ma, D. Liu, R. Xiong, and F. Wu, "iwave: Cnn-based wavelet-like transform for image compression," *IEEE Transactions on Multimedia*, vol. 22, no. 7, pp. 1667–1679, 2020.

[8] H. Ma, D. Liu, N. Yan, H. Li, and F. Wu, "End-to-end optimized versatile image compression with wavelet-like transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[9] A. Van Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *International conference on machine learning*, pp. 1747–1756, PMLR, 2016.

[10] G. K. Wallace, "The jpeg still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[11] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, 1987.

[12] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The jpeg 2000 still image compression standard," *IEEE Signal processing magazine*, vol. 18, no. 5, pp. 36–58, 2001.

[13] J. Lainema, M. M. Hannuksela, V. K. M. Vadakital, and E. B. Aksu, "Hevc still image coding and high efficiency image file format," in *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 71–75, 2016.

[14] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[15] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[16] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[17] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9446–9454, 2018.

[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] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3147–3155, 2017.

[20] Y. LeCun, Y. Bengio, *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[21] L. Theis and M. Bethge, "Generative image modeling using spatial lstms," *Advances in neural information processing systems*, vol. 28, 2015.

[22] A. Abraham, "Artificial neural networks," *Handbook of measuring system design*, 2005.

[23] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[24] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 international conference on engineering and technology (ICET)*, pp. 1–6, Ieee, 2017.

[25] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[27] S.-i. Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, no. 4-5, pp. 185–196, 1993.

[28] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[29] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE transactions on information theory*, vol. 44, no. 6, pp. 2325–2383, 1998.

[30] M. W. Marcellin, M. A. Lepley, A. Bilgin, T. J. Flohr, T. T. Chinen, and J. H. Kasner, "An overview of quantization in jpeg 2000," *Signal Processing: Image Communication*, vol. 17, no. 1, pp. 73–84, 2002.

[31] a. M. M. W. Taubman, D. S., *JPEG2000: Image Compression Fundamentals, standards, and Practice.* Springer Science + Business Media, LLC., 2013.

[32] T. Berger, "Rate-distortion theory," *Wiley Encyclopedia of Telecommunications*, 2003.

[33] R. N. Bracewell and R. N. Bracewell, *The Fourier transform and its applications*, vol. 31999. McGraw-Hill New York, 1986.

[34] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *Journal of Fourier analysis and applications*, vol. 4, no. 3, pp. 247–269, 1998.

[35] D. Taubman and M. Marcellin, *JPEG2000 image compression fundamentals, standards and practice: image compression fundamentals, standards and practice*, vol. 642. Springer Science & Business Media, 2012.

[36] Y. Xue and J. Su, "Attention based image compression post-processing convolutional neural network," 05 2019.

[37] J. Ballé, V. Laparra, and E. P. Simoncelli, "Density modeling of images using a generalized normalization transformation," *arXiv preprint arXiv:1511.06281*, 2015.

[38] S. Lyu and E. P. Simoncelli, "Nonlinear image representation using divisive normalization," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, 2008.

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, PMLR, 2015.

[40] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," *Advances in neural information processing systems*, vol. 31, 2018.

[41] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," *arXiv preprint arXiv:1703.00395*, 2017.

[42] Y. Dai, D. Liu, and F. Wu, "A convolutional neural network approach for post-processing in hevc intra coding," in *International Conference on Multimedia Modeling*, pp. 28–39, Springer, 2017.

[43] G. Bjøntegaard, "Calculation of average psnr differences between rd-curves," 2001.

[44] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[45] 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*, pp. 136–144, 2017.

[46] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5659–5667, 2017.

[47] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, *et al.*, "Conditional image generation with pixelcnn decoders," *Advances in neural information processing systems*, vol. 29, 2016.

[48] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[49] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

[50] J. Bégaint, F. Racapé, S. Feltman, and A. Pushparaja, "Compressai: a pytorch library and evaluation platform for end-to-end compression research," *arXiv preprint arXiv:2011.03029*, 2020.