PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

Mohamed El-Bachir El-Ibrahimi University - Bordj Bou Arreridj

Faculty of Science and Technology

Department of Electronics

Memory

Presented for the fulfilment of MASTER degree

FILIERE: Telecommunication

Specialty: Telecommunications System

By

Mr. MEGUELLATI Khaled

Mr. DIBEL Alaa-Eddine

Entitled

Super-Resolution Using a Deep Convolutional Network

Supported on: …/…/2024

Before the jury composed of:

Name	*Grade*	*Quality*	*Establishment*
M. F. KHERRAT	MAA	President	Univ-BBA
M. C. ERRIDIR	MCA	Examiner	Univ-BBA
M. SID AHMED Soumia	MCA	Supervisors	Univ-BBA
M. Messali Zoubeida	Pr	Sub-supervisor	Univ-BBA

University school year 2023/2024

Acknowledgements

In the name of Allah, the Most Gracious, the Most Merciful.

We extend our sincere gratitude to Allah, who blessed us with health, strength, determination, and patience to complete this work. We thank our esteemed supervisor, ***Dr.SID AHMED Soumia*** *for her invaluable guidance and support throughout our academic journey. Special thanks to our assistant supervisor,* ***Pr. MESSALI Zoubeida****,And* ***Saoudi Rania*** *for their continuous assistance.*

We express our deep appreciation to the honorable members of the examination committee for their valuable time and constructive feedback. We are also grateful to all the respected professors who enriched our academic path with their knowledge and guidance.

We cannot overlook the contribution of all those who directly or indirectly supported us in this endeavor. Particularly, we would like to thank our friends, ***SEGUENI OUSSAMA*** *and* ***Mr Zerroug Omar*** *, for their significant support and encouragement.*

This achievement wouldn't have been possible without the efforts and dedication of everyone involved. We pray that Allah rewards each and every one of you abundantly and places this work among our good deeds.

Peace be upon you all.

Dedication

To my dearest family and friends,

Today is a milestone in my life, and I attribute this achievement to the grace of God and the incredible support and love from all of you.

***Mom** and **Dad**, your unconditional love, sacrifices, and guidance have been the cornerstone of my success. I dedicate this achievement to you with deep gratitude and love.*

***Grandma** and **Grandpa**, your wisdom and prayers have always been a source of strength and inspiration to me. Thank you for your unwavering support.*

*My Uncle **Mebarak** and his **wife**, your encouragement and belief in me has meant more to me than words can express. Thank you for standing by my side every step of the way.*

*My siblings, **Hamza**, **Othman Boubakar**, and **Safia**, your love, support, and companionship have been my source of strength. Each and every one of you has played a vital role in this journey, and I am grateful to all of you.*

***Hadjar**, your innocence and bright spirit always warms my heart. You remind me of the beauty of dreams and possibilities.*

To all my uncles and aunts, thank you for your constant support and love. You have all contributed to my growth and success in countless ways.

*Special thanks to my partner, **Alaa eddine Dible***

*And my friend **Sobhi Mohamed ali**, for your constant support and encouragement. Your friendship has made this journey even more meaningful.*

This achievement is a reflection of the collective love, support, and guidance from each of you. I am forever grateful to all of you.

With all my love and appreciation.

Khaled Meguellati

Dedication

*Dear **my parents**, without their support and encouragement, I would not be where I am today. This achievement is the fruit of your hard work and sacrifices, so you have all my love and gratitude. I dedicate this work to you with pride and honor*

dear my sisters.

I dedicate this work to you with love and gratitude, as an expression of how much I love and appreciate you as my dear sisters

***dear my family**, the source of love, tenderness and unlimited support, and the strong support in all stages of my life*

***dear my friends** who have shared moments of joy and sorrow, stood by me in times of difficulty and success, and have been my help and support, never skimping on their advice and encouragement.*

*Special thanks to my partner **MEGUELLATI KHALED***

In the end, I thank God Almighty for his success and help, and I thank everyone who supported me and supported me in my life from near and far

DIBEL Alaa eddine

Table of Contents

Chapter 3* : *Deep learning-based super-resolution optimization algorithms

List of Figures

List of Tables

List of Equation

Acronyms and Abbreviations

AI	**Artificial Intelligence**
CAR	**Content Adaptive Reshaping**
CNN	**Convolutional Neural Networks**
DL	**Deep learning**
DWSR	**Deep Wavelet Super Resolution**
EDSR	**Enhanced Deep Residual Networks for Single Image Super-Resolution**
HR	**High-Resolution**
HRSB	**High Resolution Sub-Bands**
ISR	**Image super resolution**
LR	**Low-Resolution**
LRSB	**Low Resolution Sub-Bands**
ML	**Machine learning**
MRI	**Magnetic Resonance Imaging**
MSE	**Mean Squared Error**
PSNR	**Peak Signal-to-Noise Ratio**
ResamplerNet	**Resampler Generation Network**
ReLU	**Rectified Linear Unit**
ROI	**Region Of Interest**
SRCNN	**Super-Resolution Convolutional Neural Network**
SRGAN	**Superior-Resolution Generative Adversarial Network**
SR	**Super-Resolution**
SRSB	**Super Resolution Sub-Bands**
SSIM	**Structural Similarity Index Metric**
VDSR	**Very Deep Super Resolution**
2D DWT	**Two-Dimensional Discrete Wavelet Transform**
2D IDWT	**Two-Dimensional Inverse Discrete Wavelet Transform**

Abstract

This master thesis deals with Super Resolution (SR) which is a set of image processing techniques used in computer vision to improve the quality of degraded images We focus on the differences between conventional image interpolation algorithms and deep learning based algorithms that have made significant progress in image quality improvement technology We will implement Conventional interpolation techniques and then we will implement deep learning algorithms SRCNN, VDSR,DWSR and EDSR Then the comparative study is performed in terms of calculating the peak signal-to-noise ratio PSNR and the structural similarity index SSIM.

Résumés

Ce mémoire de master traite de la Super Résolution (SR) qui est un ensemble de techniques de traitement d'images utilisées en vision par ordinateur pour améliorer la qualité des images dégradées Nous nous concentrons sur les différences entre les algorithmes d'interpolation d'images conventionnels et les algorithmes basés sur l'apprentissage profond qui ont fait des progrès significatifs dans la technologie d'amélioration de la qualité des images Nous mettrons en œuvre des techniques d'interpolation convolutives et ensuite nous mettrons en œuvre des algorithmes d'apprentissage profond SRCNN, VDSR ,DWSR et EDSR Ensuite, l'étude comparative est effectuée en termes de calcul du rapport signal-bruit maximal PSNR et de l'indice de similarité structurelle SSIM.

ملخص

تتناول هذه الرسالة الرئيسية هذه رسالة الماجستير الاستحالة الفائقة (SR) وهي مجموعة من تقنيات معالجة الصور المستخدمة في الرؤية الحاسوبية لتحسين جودة الصور المتدهورة نركز على الاختلافات بين خوارزميات الاستيفاء التقليدية للصور والخوارزميات القائمة على التعلم العميق التي حققت تقدماً كبيراً في تقنية تحسين جودة الصورة سنقوم بتطبيق تقنيات الاستحالة الفائقة ثم تطبيق خوارزميات التعلم العميقSRCNN , VDSR, EDSR و DWSR ثم يتم إجراء دراسة مقارنة من حيث حساب نسبة ذروة الإشارة إلى الضوضاء PSNR ومؤشر التشابه الهيكلي SSIM.

Key words: Super Resolution , deep learning, CNN, Wavelet

General Introduction

General Introduction

Super Resolution (SR) refers to the process of restoring high-resolution (HR) images from low-resolution (LR) images a fundamental task in image processing. The challenge involves accurate matching between low-resolution (LR) and high-resolution (HR) images, which is common in medical imaging. The demand for high-resolution images has increased with the rapid development of imaging devices and the increasing use of multimedia applications. However, capturing these images can be difficult and expensive. Therefore, ultrasound imaging techniques have been developed to improve the resolution of low-resolution images and deliver visually appealing and detailed results. Interpolation-based algorithms have long been used in SR techniques as a conventional method [1]. These methods are characterized by their computational efficiency and ease of implementation. They rely on interpolation techniques to estimate low-resolution missing high-frequency information in images. However, they often have limited performance in preserving fine details and producing realistic textures.

New methods based on deep learning are revolutionizing many areas of computer vision [2], including image super-resolution. Deep algorithms make significant improvements in reconstructing fine details and optimizing visual results. The algorithms rely on deep neural networks to learn subtle patterns within images and create high-resolution versions of low-resolution images. Thanks to rapid advances in deep learning, deep learning models are actively used in various fields such as computed tomography, medical images, radar images, and others. A large range of deep learning-based methods have been developed, from early convolutional neural networks such as SRCNN to newer deep convolutional neural networks such as VDSR[7], DWSR[8], and EDSR[10].

Our work is organized into three chapters:

In the first chapter, we discuss the basic concepts of image super decomposition, review traditional methods, and then focus on deep learning-based algorithms

The second chapter will cover deep learning concepts in detail, including analyzing the different layers of deep learning networks. We will provide a detailed explanation of deep learning networks that utilize CNN.

In the third chapter, we will study and discuss the results obtained more specifically.

We will implement the traditional Bilinear, Bicubic, and Nearest Neighbor interpolation methods as well as the following super-resolution deep learning networks: SRCNN, VDSR, DWSR, and EDSR. We especially focus on DWSR in terms of its effectiveness in medical images.

Chapter 1 : Basic Concepts of Single Image Super Resolution

Summary

In this chapter, we will introduce the key concepts of super-resolution image enhancement, known as image resolution enhancement (ISR). We will describe conventional image enhancement methods, and then focus on explaining four deep learning-based image enhancement algorithms. We will define the performance metrics used in this context

Table of Contents

1.1 Introduction

Image Super Resolution is an important task in image processing reconstruct HR images from LR images. Significant progress has been made in the field of SR [1], especially using deep learning (DL) techniques [15]. SR therefore represents a difficult challenge for image processing, as it aims to increase the resolution of the image beyond its original resolution.

There are a wide range of applications that benefit from SR technologies, including medical imaging [14], surveillance , etc. ., In recent years , progress in the field of deep learning has led to the development of effective super-resolution methods, enhancing its potential for widespread use in various fields based on a variety of deep learning networks, such as VDSR [7], Deep Wavelet Prediction for Super-resolution (DWSR) neural networks [8] and Convolutional (CNN) and deep learning-enhanced super-resolution (EDSR) [10].

In this chapter, we will explore the principle of image super-resolution process as well as the techniques used to improve image quality. We will discuss both conventional interpolation methods and deep learning-based super-resolution strategies, highlighting the advantages of these advanced techniques.

1.2 Principal of image super-resolution (ISR)

Super-resolution is a popular image processing technique used in many disciplines to convert LR images into HR images. Its applications extend to fields such as medical imaging, where the acquisition of high-resolution MRI images comes up against constraints linked to scan time, spatial coverage and SNR. This method overcomes these obstacles by generating high-resolution MRI images from their low-resolution counterparts. In image processing, super-resolution facilitates detection, identification and facial recognition from low-resolution security camera images. It is also used in digital photography and other fields [16].

The SR process encompasses a variety of methods, relying primarily on image processing algorithms to estimate missing or masked details in LR images. Among the most common approaches are interpolation-based super-resolution and deep learning-based super-resolution techniques, both of which aim to improve the quality and fidelity of images beyond their original resolution.

1.3 Mathematical Modeling

Super-resolution image processing can be mathematically modelled as an inverse problem, where the objective is to estimate a high-resolution image from a low-resolution image and additional information. Specifically, if we have a high-resolution image (HR) $\boldsymbol{x}$ that has been downsampled and blurred to obtain a low-resolution image (LR) $\boldsymbol{y}$, this downsampling and blurring process can be represented mathematically using a convolution operator H, which accounts for blurring and resolution reduction. The mathematical model can therefore be expressed as follows:

$$y = H * x + n \tag{1.1}$$

Where $\boldsymbol{y}$ is the low-resolution image,

- $\boldsymbol{x}$ is the high-resolution image,
- $\boldsymbol{n}$ is the added noise, and
- * Denotes convolution operator.

The objective of super-resolution image processing is to find an estimation of the high-resolution image $\boldsymbol{x}$ from the low-resolution image $\boldsymbol{y}$ and the convolution operator H. This can be achieved using image processing algorithms that estimate the inverse transfer function of $H(H^{-1})$ known as the deconvolution operator, to recover the high-resolution image $\boldsymbol{x}$.

In some cases, additional information may be available to help solve the inverse problem. [3].

The degradation model can be expressed by the equation:

$$g_k = DM_k\, B_k f + \varepsilon_k \tag{1.2}$$

Where g_k represents the set of k low-resolution images, $\boldsymbol{D}$ is the downsampling operator, M_k denotes the shift operator (translation, rotation), B_k is the blur matrix, $\boldsymbol{f}$ is the high-resolution image, and ε_k represents system noise. Figure 1.1 summarizes the generation of low-resolution images from a high-resolution image.

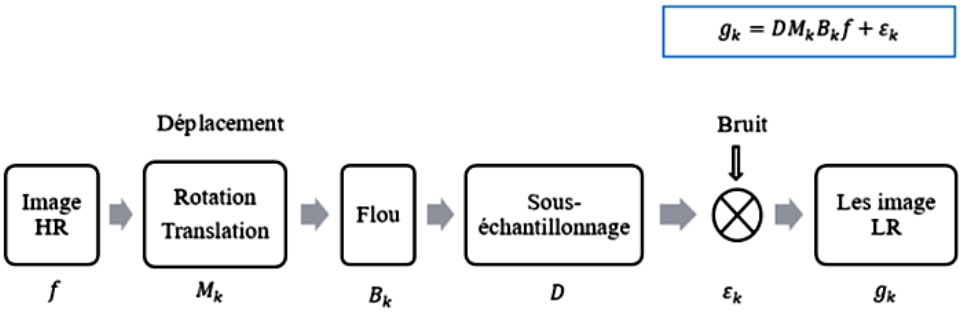


Figure 1.1: The control model that connects LR images to HR images [3]

1.4 Image Super Resolution Methods

Interpolation-based super-resolution techniques involve estimating a high-resolution (HR) image from one or more low-resolution (LR) images. They are based on the assumption that the high resolution and low-resolution images are linked by an interpolation function. The image super-resolution approach can be categorized into two broad classes [4].

- interpolation-based methods such as bilinear, bicubic, and nearest interpolation.
- learning-based methods, including machine learning and deep learning.

The classification of the different super-resolution methods is illustrated in Figure 1.2

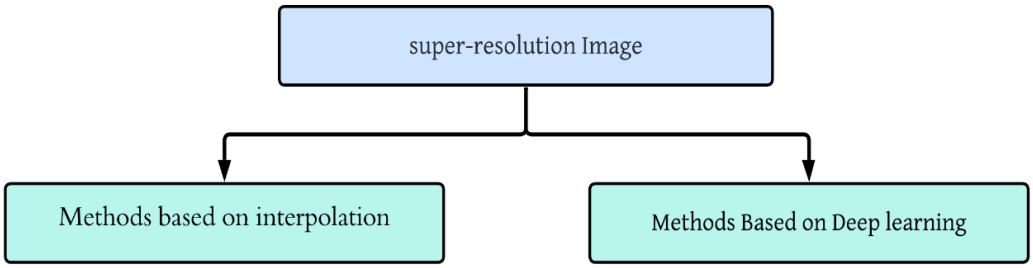


Figure 1.2:Classification of the image super-resolution approaches.

1.4.1 Methods based on interpolation

Super-resolution techniques relying on interpolation involve the estimation of a high-resolution image using one or multiple low-resolution images. These approaches operate under the assumption that there exists an interpolation function connecting high-resolution and low-resolution images.

Numerous interpolation-based super-resolution methods exist, encompassing various approaches, namely:

- Bilinear interpolation.
- Bicubic interpolation.
- Nearest neighbor interpolation.

1.4.1.1 Bilinear interpolation

Bilinear interpolation is a 2D optimization method used to estimate the values of points within a regular grid based on the known values at the grid points. This method assumes that the variation of the function between grid points is linear in each direction. Specifically, when presented with a regular grid of points with known values at the grid nodes and needing to estimate the function's value at a point not on the grid, bilinear interpolation employs a linear combination of the four nearest grid points to estimate the function's value at that point.

to achieve this, the four closest points to the interpolation point are first identified. Next, a linear function is calculated for two variables that pass through these four points. Finally, this linear function is evaluated at the interpolation point to obtain an estimate of the value of the function at that point. The process involves determining the weights of the four closest grid points based on their distances from the interpolation point and using these weights to calculate the interpolated value. This method is especially useful for tasks such as resizing an image and optimizing image resolution [5]. Figure 1.3 illustrates the principle of bilinear interpolation.

Bilinear interpolation can be mathematically represented as follows:

$$f(x,y) = (1-u)(1-v)f(0,0) + u(1-v)f(1,0) + (1-u)vf(0,1) + uvf(1,1) \quad (1.3)$$

Where:

- *f (x, y)* is the interpolated value at point (x, y).
- *f (i, j)* Represents the known values at the grid points.
- u and v are the interpolation coefficients, typically between 0 and 1, representing the relative distances between the interpolation point and its nearest grid points in the x and y directions, respectively.

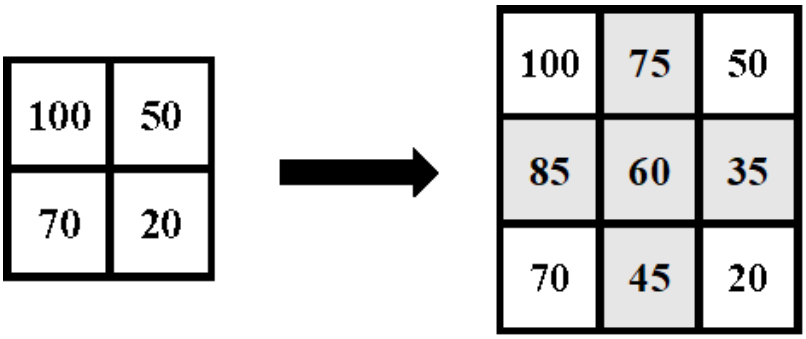


Figure 1.3: Bilinear interpolation

1.1.1.1 Bicubic Interpolation

Cubic interpolation is a surface interpolation method that employs a cubic function to estimate pixel values missing in an image. This technique is commonly utilized for increasing image resolution or for achieving a more precise interpolation compared to bilinear interpolation.

Cubic interpolation operates by fitting a cubic function to each set of four neighboring pixels in the image. This cubic function is then utilized to estimate the value of any point in the image that is undefined. The cubic functions are adjusted to ensure that the surface of the interpolated image is smooth and continuous.

While cubic interpolation offers higher precision than bilinear interpolation, it can also be more computationally expensive. Moreover, cubic interpolation may introduce artifacts in the interpolated image, especially if the input data is noisy.

Mathematically, cubic interpolation can be expressed as follows:

$$f(x,y) = \sum_{j=0}^{3}\sum_{j=0}^{3} a_{ij}x^i y^j \qquad (1.4)$$

Where:

- $f\ (x,y)$ is the interpolated value at point (x, y).
- a_{ij} are the coefficients of the cubic function.
- x and y represent the coordinates of the point being interpolated.

Figure 1.4 illustrates the principle of cubic interpolation.

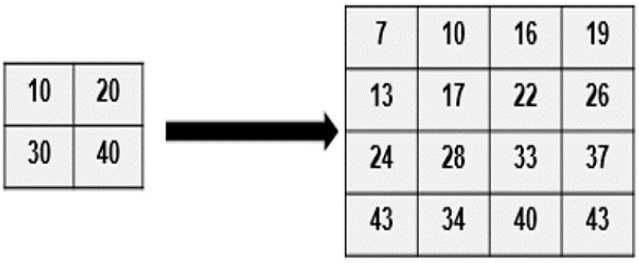


Figure 1.4: Bicubic Interpolation

1.4.1.2 Nearest neighbor interpolation

Nearest-neighbor interpolation is an image interpolation method that involves replacing each missing pixel in an image with the value of the nearest pixel in the original image. In practice, this method entails identifying the nearest pixel to the position of the missing pixel and assigning it the same value as that pixel. While straightforward and quick to implement, this method can yield images with lower quality compared to other interpolation methods. Specifically, it may produce images with sharp edges and pronounced contours, leading to a sense of pixilation and lack of smoothness in transitions.

Nevertheless, in certain specific cases, nearest-neighbor interpolation may be preferable to other interpolation methods. For example, it can be useful for image segmentation or binary image classification. In such scenarios, nearest-neighbor interpolation may provide more accurate and consistent results compared to other methods.

Mathematically, nearest-neighbor interpolation can be represented as follows:

$$f(x, y) = ([x + 0.5], [y + 0.5]) \quad (1.5)$$

$$f(x, y) = f([x + 1], [y + 1]) \quad (1.6)$$

Where:

- f (x, y) is the interpolated value at point (x, y).
- x represents the nearest integer less than or equal to x.

The essence of nearest-neighbor interpolation lies in its simplicity and computational efficiency, making it suitable for specific applications where preserving the exactness of pixel values is paramount. However, its limitations in producing smooth transitions and accurately representing continuous gradients may necessitate the use of alternative interpolation methods in many image processing tasks.

Figure 1.5 represents the principle of nearest-neighbor interpolation.

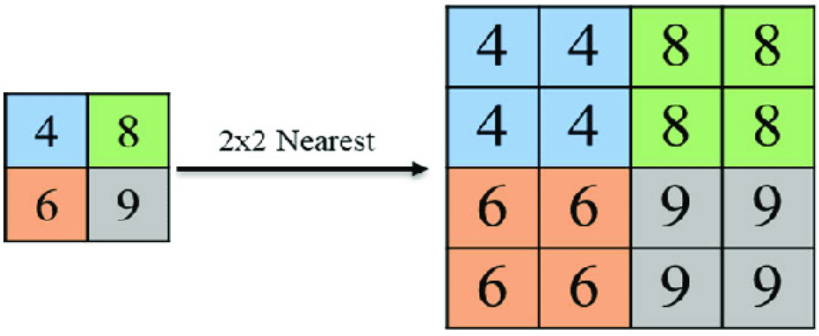


Figure 1.5: Nearest-neighbor interpolation

1.4.2 Methods based on (Deep Learning):

Deep learning-based techniques such as data matching, rectifications, machine learning with CNN [6] is deployed for optimal results. Advanced solutions include EDSR [10], VDSR [7], DWSR, SRCNN. We will present in the next section, the SR based on deep learning.

1.4.2.1 Super Resolution (SR) Algorithms Based on Deep Neural Networks

In recent years, deep learning has witnessed tremendous progress in the field of image processing, and among the applications of deep learning in this field are SR techniques, which are used to improve the quality of low-resolution images such as SRCNN, and SR technology. VDSR technology, Enhanced EDSR and DWSR. In this research, we will discuss each one of them in detail.

1.4.2.2 SRCNN: Super-resolution convolution neural network

The SRCNN is a straightforward architecture for CNN, composed of three critical layers: patch extraction, nonlinear mapping, and reconstruction. The primary function of the patch extraction layer is to extract abundant patches from the input and represent them via convolutional filters. Following this, the nonlinear mapping layer employs 1x1 convolutional filters designated to modify channel numbers and inject non-linearity into operations. Subsequently, a high-resolution image is reconstructed in the final layer of this process.

To effectively train this network mechanism, they utilize Mean Squared Error (MSE) as a loss function criterion. To evaluate the outcomes accurately and ensure the precision of algorithm performance within its operating parameters, a Peak Signal-to-Noise Ratio (PSNR) assessment measurement standard has been put in place. The Super-Resolution Convolutional Neural Network (SRCNN) operates as depicted in the accompanying illustration, the SRCNN consists of the following operation, as shown in Figure 1.6 [6].

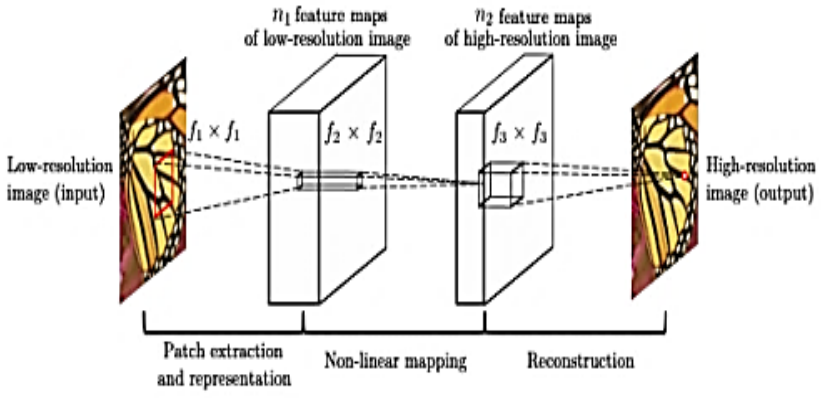


Figure 1.6: Super-resolution convolutional neural network (SRCNN) architecture [6]

- **Preprocessing:**

The process of obtaining the correct low-resolution (LR) image can be a complex task due to variable resolution factors. To achieve a high-resolution (HR) equivalent, meticulous bicubic interpolation is often employed. This technique enlarges the size of the LR image while preserving important details.

- **Feature Extraction:**

This stage uses an expanded dataset created from distinct characteristics extracted from the LR image, simulating high-resolution aspects. Such features may encompass edges, textures, or diverse attributes that capture essential information contained within the image.

- **Non-linear Mapping:**

Non-linear mapping serves as a critical tool that links feature maps to both LR and HR image segments. This invaluable mechanism is proficient at detecting and integrating complex details from various layers, and thus effectively connects disparate elements within the images.

- **Reconstruction:**

With reliance on data obtained from HR patches coupled with results from non-linear mapping. this process recreates a complete HR image that captures both the essence and subtle nuances. Gathering these components yield a High-Resolution output.

These steps are critical in achieving super-resolution transformations of images. Such techniques are essential for enhancing low-resolution imagery, utilizing state-of-the-art technologies and principles that are fundamental to the process of image reconstruction.

The primary goal of SRCNN is to generate high-resolution images from low-resolution inputs through classical super-resolution methods usually based on interpolation techniques such as cubic interpolation or nearest neighbor interpolation. These methods relied on simple mathematical operations and did not involve learning complex patterns from the data. Although it was somewhat effective, it often produced blurry results and was less able to preserve high-frequency detail.

1.4.2.3 Very Deep Super Resolution (VDSR)

VDSR is a deep learning-based method for image super-resolution based on convolutional neural network architecture (CNN). Deep convolutional networks are used to generate high-resolution images from low-resolution images. By stacking multiple convolutional layers, VDSR captures the complex relationships between low- and high-resolution images, enhancing fine details and sharp edges. During the training process, VDSR reduces the differences between predicted and actual images.

VDSR is versatile, handling various image enhancement operations, and takes advantage of parallel computing to achieve faster processing. It has shown exceptional results, making it valuable for applications such as digital imaging, medical imaging, and video processing [7]. In Figure 1.7, the structure of the VDSR network is shown, where $\boldsymbol{x}$, $\boldsymbol{r}$, and $\boldsymbol{y}$ represent the low-resolution, residual, and high-resolution images, respectively.

The model consists of successive layers of convolutional transformations and nonlinear functions (such as the ReLU activation function). The low-resolution image is added to the extracted image to obtain the high-resolution image.

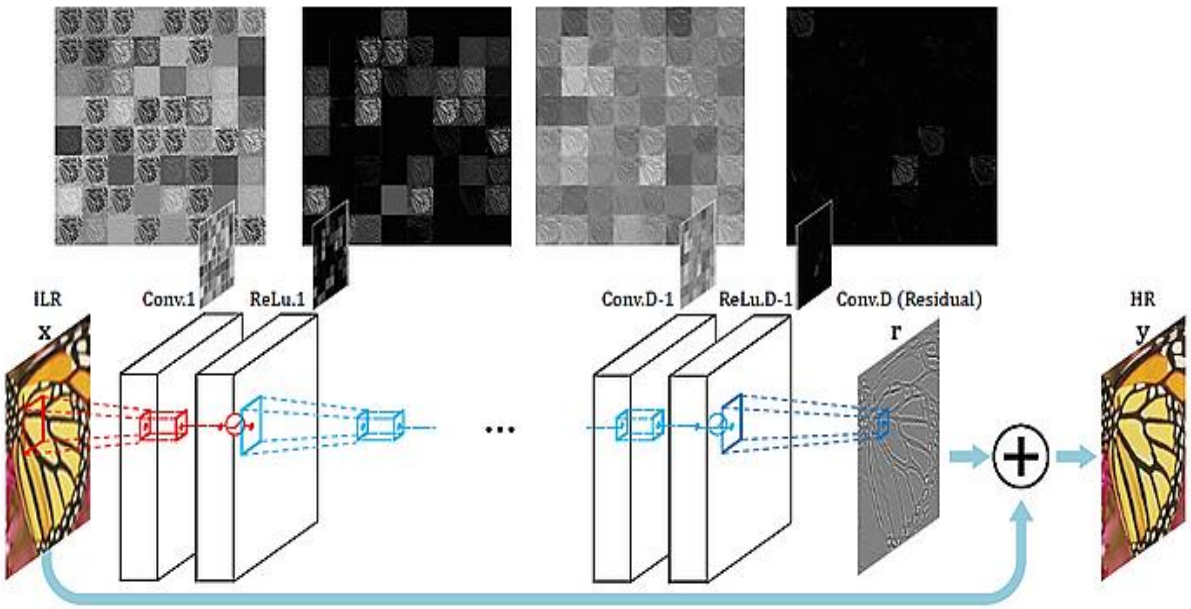


Figure 1.7 : VDSR Network Architecture [7]

1.4.2.4 Deep wavelet super resolution (DWSR)

Deep Wavelet Super Resolution represents a significant advancement in image processing and resolution enhancement. Unlike traditional deep learning-based algorithms that operate in the spatial domain, this method combines the power of deep learning with the efficiency of wavelet transforms to enhance image resolution, producing high-resolution images from low-resolution originals.

Using frequency analysis, Super Deep Wave Resolution technologies can break down low-resolution images into different frequency components, and then analyze and optimize them using deep models. This process improves image quality, it increases clarity and detail. Such advancements expand the potential applications of image processing in fields like medical imaging and photography.

For the sake of clarity, we will introduce discrete transform before presenting the DWSR network.

1.4.2.4.1 Discrete Wavelet Transformation

A wave is a wave oscillation that transforms non-stationary signals. This technique is used to overcome the issues with the Fourier method, which deals with frequencies but does not provide specific temporal details. The wavelet transform, on the other hand, can achieve high frequency resolution and high temporal resolution at the same time. This method starts by using fundamental waves such as Haar, and then translates the signal into a graded set of fundamental waves. This wavelet transform is useful in analyzing low- and high-frequency non-stationary signals, as it can achieve high frequency resolution for low-frequency components and high temporal resolution for high-frequency components.

Wavelet analysis is used to divide the information in an image (signals) into two distinct components: approximations and details (sub-signals). Approximations capture the general low-frequency information of an image, providing a consistent representation. Details, on the other hand, represent the high-frequency components, capturing the finer, more complex features and edges of the image. This decomposition allows for efficient image compression, noise reduction, and feature extraction, enhancing various image processing tasks.

The signal is passed through two filters: high-pass and low-pass filters. The image is then decomposed into high-frequency components (detail) and low-frequency components (approximation). At each level, we get 4 sub-signals. Rounding shows an overall trend of pixel values and details as horizontal, vertical, and diagonal components.

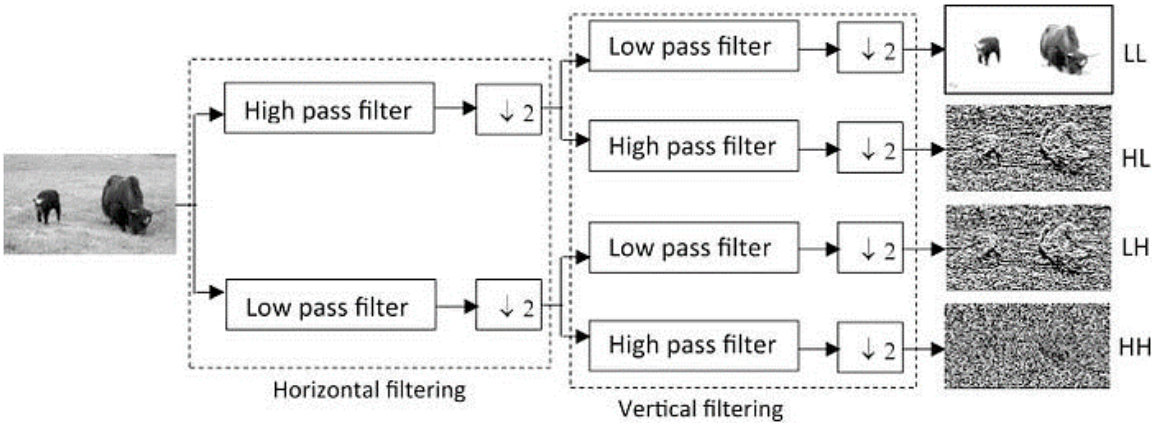


Figure 1.8 : The procedure of 1-level 2dDWT decomposition.

1.4.2.4.2 Deep Wavelet Prediction for Super-resolution (DWSR)

The SR can be viewed as the problem. of restoring the details of the image given an input LR image. This viewpoint can be combined with wavelet decomposition. As shown in Figure 1.9, if we treat the input image as an LL output of 1-level 2dDWT, predicting the HL, LH and HH sub-bands of the 2dDWT will give us the missing details of the LL image. Then one can use 2dIDWT to gather the predicted details and generate the SR results [9].

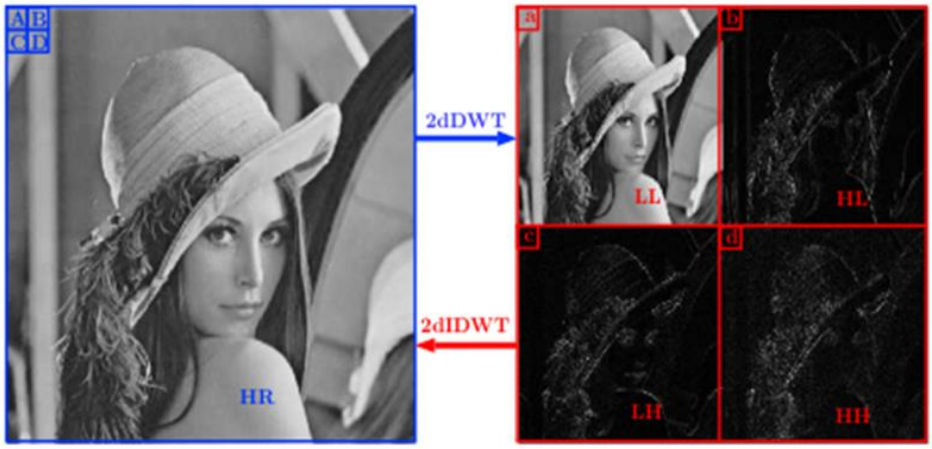


Figure 1.9 : 2D Discrete Wavelet Transform (2D DWT) and its inverse (2D IDWT)

This approach effectively combines super-resolution, wavelet transform, and deep learning. The low-quality image is first divided into sub-bands using wavelet transform. Then, each wavelet subband, representing different frequency details, is processed using a trained deep learning model for super-resolution. The deep learning model learns to predict the high-frequency details of each wavelet sub-band, effectively enhancing the accuracy of that particular frequency band. After processing all wavelet sub-bands, the optimized results are combined (by IDWT) to reconstruct the final high-resolution image.

This approach capitalizes on the strong points of wavelet transforms and deep learning: Wavelet transform is used to analyze multi-scale images and deep learning is used to learn complex relationships between low-quality images and high-quality images. Overall, this integration provides a powerful framework for improving the resolution of images while preserving their critical features, allowing for more accurate and detailed reconstruction of high-resolution images from low-quality inputs.

1.4.2.4.3 The architecture of the network

The network structure is thoroughly described, where it is specified that the proposed network is based on a deep structure similar to the residual network, with two layers of inputs and outputs, each with 4 channels. While most deep learning-based SR network methods rely on only one input-output channel, the proposed network considers four input channels and produces four corresponding output channels. [21]

The details show that there are 64 filters in the first layer of size $4 \times 3 \times 3 \times 3 \times 3$, and 4 filters in the last layer of size $64 \times 3 \times 3 \times 3 \times 3$. In the middle part of the network, each hidden layer contains N layers of the same size, with 64 filters of size $64 \times 3 \times 3 \times 3 \times 3 \times 64$ each.

The outputs of each layer except the output layer are activated using a ReLU activation function to generate a non-linear activation map. Zero padding is used in each layer to keep the output size equal to the input size, to avoid loss of information

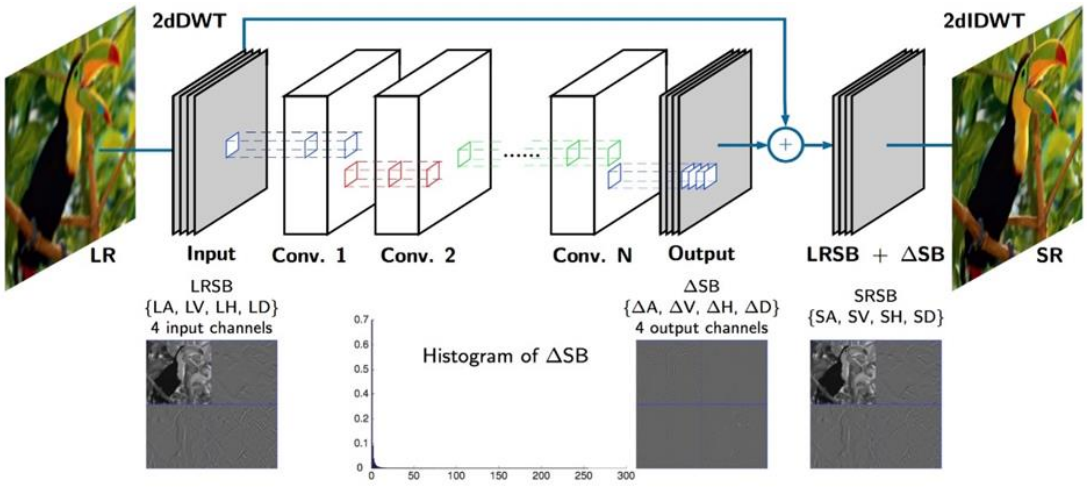


Figure 1.10 : Wavelet-based super-resolution network architecture with contrast visualization

1.4.2.4.4 Training Procedure:

To train the network, we start by enlarging the low-resolution training images using bicubic interpolation based on the original zoom factor. These magnified LR images are then subjected to a 2D discrete wavelet transform (2D DWT) using Haar waves wavelet , generating four LR sub-bands referred to as (LRSB):

$$LRSB = \{LA, LV, LH, LD\} = 2dDWT\{LR\} \tag{1.7}$$

The sub-bands LA, LV, LH, and LD contain wavelet coefficients representing the average, vertical, horizontal, and diagonal details of the LR image, respectively. The 2dDWT {LR} notation signifies the application of the two-dimensional discrete wavelet transform (2D DWT) to the LR image. Similarly, this transformation is also performed on the corresponding HR training images to generate four HR wavelet sub-bands (HRSB):

$$HRSB = \{HA, HV, HH, HD\} = 2dDWT\{HR\} \tag{1.8}$$

The sub-bands HA, HV, HH, and HD represent the wavelet coefficients indicating the mean, vertical, horizontal, and diagonal details of the HR image, respectively. Then, the difference ΔSB (residual) between the LRSB sub-coefficient and the corresponding HRB is:

$$\begin{aligned} \Delta\mathrm{SB} &= \mathrm{HRSB} - \mathrm{LRSB} \\ &= \{\mathrm{HA} - \mathrm{LA}, \mathrm{HV} - \mathrm{LV}, \mathrm{HH} - \mathrm{LH}, \mathrm{HD} - \mathrm{LD}\} \\ &= \{\Delta A, \Delta V, \Delta H, \Delta D\} \\ \Delta A &= HA - LA \\ \Delta V &= HV - LV \\ \Delta H &= HH - LH \\ \Delta D &= HD - LD \end{aligned} \tag{1.9}$$

The goal of the network is to produce the difference (ΔSB) between the LR and HR wavelet sub-bands (LRSB and HRSB, respectively) when given the input LRSB. This difference is the target output, and the feedforward procedure is denoted as f(LRSB).[9]

1.4.2.4.5 Generating SR Results

To achieve SR results, the bicubic magnified LR inputs are transformed by 2dDWT to produce LRSB images applying as follows Equation (1.8). The LRSB is then fed forward through

a network trained to produce ΔSB. Adding LRSB and ΔSB together generates four sub-bands of SR wavelets (SRSB) referred to as:

$$\mathrm{SRSB} = \mathrm{LRSB} + \Delta\mathrm{SB}$$

$$\mathrm{SRSB} = \{\mathrm{SA}, \mathrm{SV}, \mathrm{SH}, \mathrm{SD}\}$$

$$\mathrm{SRSB} = \{\mathrm{LA} + \Delta\mathrm{A}, \mathrm{LV} + \Delta\mathrm{V}, \mathrm{LH} + \Delta\mathrm{H}, \mathrm{LD} + \Delta\mathrm{D}\} \qquad (1.10)$$

In the final step, the two-dimensional inverse discrete wavelet transform (2D IDWT) is applied to generate the super-resolution (SR) image results:

$$\mathrm{SR} = 2\mathrm{dIDWT}\{\mathrm{SRSB}\} \qquad (1.11)$$

1.4.2.5 Enhanced Deep Residual Networks for Single Image Super-Resolution (EDSR)

The EDSR algorithm is a super-resolution network powered by deep learning that has been highly acclaimed for its impressive performance improvements.

These improvements arise from an optimization strategy that entails the elimination of unnecessary units found in traditional residual networks. Furthermore, the EDSR algorithm benefits from expanding the size of its model, while maintaining stable training protocols, which contributes to its effectiveness [10]. The EDSR network architecture is illustrated in Figure 1.11

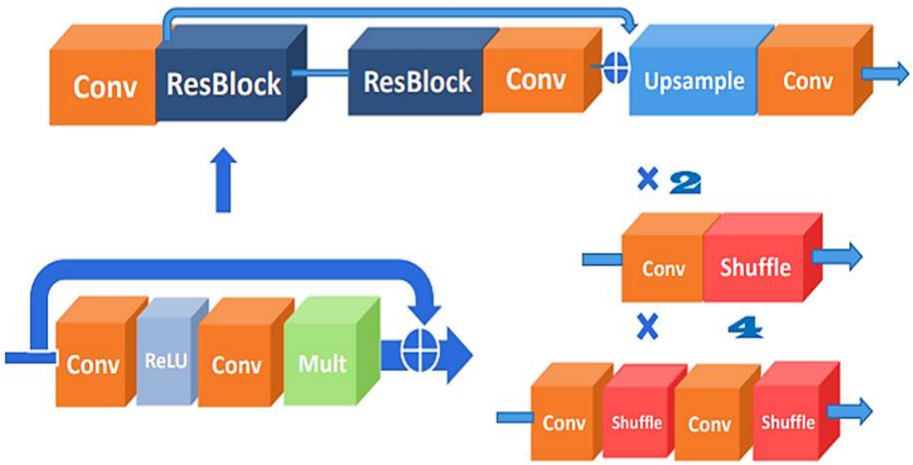


Figure 1.11 : Architecture of single-scale SR network (EDSR)

The system comprises three components: the ResamplerNet, the Downscaling module, and the SRNet. The ResamplerNet is tasked with estimating content-adaptive resampling kernels and their corresponding offsets for every pixel in the downscaled image. To facilitate the training of the ResamplerNet, the SRNet, capable of any differentiable upsampling operation, is employed. Its role is to minimize the SR error. The entire framework undergoes end-to-end training by back-propagating error signals through a differentiable downscaling module. The specific configuration of each building block is outlined within the blue dashed frame. [11]

1.5 Performance assessment in terms of quantitative parameters

Evaluation of performance using quantitative parameters such as PSNR and SSIM is commonly used to measure image quality in these methods.

1.5.1 PSNR

The quality of the reconstructed image is often assessed by comparing each pixel of the original image with that of the reconstructed image. This comparison is used to measure the difference between the two images. This is generally done using the MSE, which calculates the average of the squares of the differences between the values of each pixel in the original image and those in the reconstructed image. [12]

To interpret this difference in terms of quality, it is compared to the maximum amplitude of the signal, which represents the maximum possible value in the dynamic range of the image. For example, for an 8-bit coded image, the maximum value would be 255.

The signal-to-noise ratio is then used to determine the quality of the results. This ratio is calculated by taking the ratio between the maximum amplitude of the signal (represented by the maximum possible value) and the mean square error. More precisely, it can be calculated directly from the mean square error (MSE) using the following formula:

$$PSNR = 10 \times log_{10}\left(\frac{I^2{}_{max}}{MSE}\right) \tag{1.12}$$

$Imax$ is the maximum possible pixel value. This PSNR ratio is expressed in decibels (dB) and provides a measure of the quality of the reconstructed image compared with the original.

A higher $PSNR$ indicates better reconstruction quality, as it means a higher signal-to-noise ratio and therefore greater fidelity between the original and reconstructed images.

1.5.2 SSIM

The Structural Similarity Index (SSIM) quantifies the similarity between two images by comparing their luminance, contrast, and structure. Here's a detailed explanation of how SSIM works with equations [13]:

- **Luminance Comparison (l):** The luminance comparison measures how well the mean intensities of the original image u and its reconstruction û match. It is calculated as:

$$l(u, \hat{u}) = \frac{2u_u u_{\hat{u}} + c_1}{u^2{}_u + u^2{}_{\hat{u}} + c_1} \tag{1.13}$$

  - μ_u and $\mu_{\hat{u}}$ are the mean intensities of images u and uˆrespectively.
  - c_1 is a constant introduced for numerical stability.

Contrast Comparison (c): The contrast comparison evaluates how well the standard deviations of the original image u and its reconstruction û match. It is computed as:

$$C(u,\hat{u}) = \frac{2\sigma_{u\ \sigma_{\hat{u}}+c_2}}{\sigma^2{}_u + \sigma^2{}_{\hat{u}} + c_2} \quad (1.14)$$

Where:

- σu and σû are the standard deviations of images u and uˆrespectively.
- c_2 is a constant introduced for numerical stability.

- **Structure Comparison (s):** The structure comparison measures how well the structures of the original image u and its reconstruction uˆmatch. It is represented as:

$$S(u,\hat{u}) = \frac{2\sigma_{u\ \sigma_{\hat{u}}+c_3}}{\sigma^2{}_u + \sigma^2{}_{\hat{u}} + c_3} \quad (1.15)$$

Where:

- $\sigma u\hat{u}$ is the covariance between images u and $\hat{u}$.
- c_3 is a constant introduced for numerical stability.

Finally, the overall SSIM is calculated by taking the product of these three comparisons:

$$\boldsymbol{SSIM = l(u,\hat{u}) * c(u,\hat{u}) * s(u,\hat{u})} \quad (1.16)$$

The SSIM score ranges between -1 and 1, where 1 indicates perfect similarity between the two images. It considers multiple aspects of image similarity, making it a robust metric for perceptual image quality assessment.

1.6 Conclusion

In this chapter, we have seen in detail at different techniques used in the field of SR. We discuss two main approaches: conventional methods based on interpolation, such as bicubic, bilinear, and nearest neighbor, and deep learning-based methods, including SRCNN and EDSR and VDSR.

Conventional methods are relatively simple and use interpolation algorithms to increase image resolution. In contrast, deep learning-based methods use convolutional neural networks to learn how to generate high-resolution images from low-resolution images. We also discussed evaluation metrics commonly used to measure the performance of super-resolution models, such as PSNR and SSIM. These metrics are used to quantify the quality of the generated image compared with the reference image.

In summary, this chapter has enabled us to understand the different approaches used in image super-resolution, as well as the evaluation tools used to assess their performance.

Chapter 2 : Deep Learning

Summary

This chapter provides an introduction to the fundamental concepts of machine learning and deep learning, including neural networks and convolutional neural networks (CNNs)

Table of Contents

2.1 Introduction

Since its emergence in 2006[53], the advent of deep learning has been a profound paradigm shift in machine learning, forming its own distinct universe within the broader landscape. In the last few years in particular [54], recent strides in deep learning methodologies have extended across machine learning and AI, catalyzing transformative reform in signal and information processing that is particularly useful for tasks involving big data such as deep image learning and image super-analysis as we will use in our study.

This chapter seeks to embark on a comprehensive exploration of the fundamental principles governing AI and machine learning, delving into the structures and architectures that form the basis of deep learning. In addition, we will illustrate the practical applications of deep learning in various industries.

2.2 Machine learning:

Machine Learning involves the process of instructing computers to learn from data through a variety of algorithms. These algorithms iteratively refine their understanding of the data to enhance their ability to explain and predict outcomes. As more data is absorbed by the algorithms, the models generated become increasingly accurate representations of the training data. Essentially, a machine-learning model emerges as the outcome of training a machine-learning algorithm.[22]

The connection between AI and the fields of machine learning, deep learning, data science, and computer vision is shown in Figure 2.1.

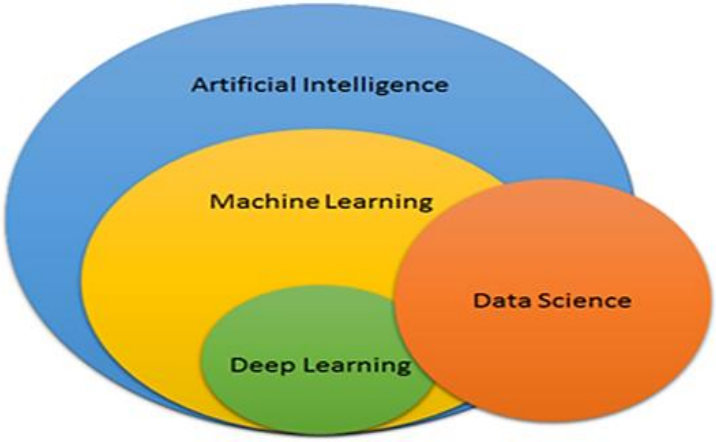


Figure 2.1 : Diagram Representing the Relationships of AI-Related Domains [39]

2.3 Approaches of Machine Learning

Enhancing the precision of predictive models necessitates the utilization of machine learning techniques. The selection of appropriate approaches varies depending on the specific nature of the business problem at hand, as well as the type and quantity of available data. In the ensuing section, we delve into the classifications of machine learning [23]

2.3.1 Supervised learning

Supervised learning stands as the predominant paradigm in both machine learning and deep learning. As the term implies, this approach entails guiding machine learning systems by presenting them with examples (data) about the tasks they are expected to perform. [24]

The applications of supervised learning span a wide range, encompassing areas such as computer vision, regression, and classification tasks. Indeed, the vast majority of problems tackled in both machine learning and deep learning frameworks rely on supervised learning techniques.

2.3.2 Unsupervised learning

Unsupervised learning proves most effective in scenarios where copious amounts of unlabeled data are necessary for addressing the problem. For instance, social media platforms (e.g. Twitter, Snapchat) and others amass substantial volumes of unlabeled data, making them prime candidates for employing unsupervised learning techniques [25]. Figure 2.2 shows supervised ML vs unsupervised ML.

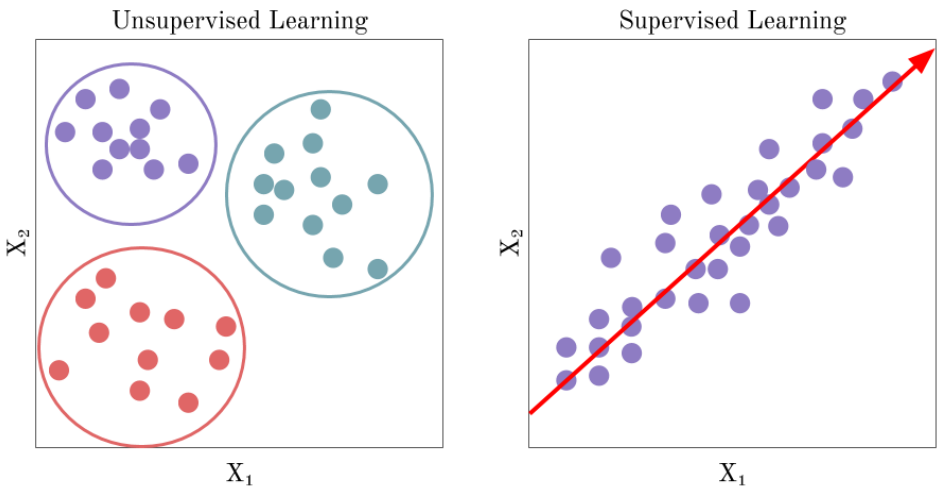


Figure 2.2 : Difference between the two types of learning [25]

2.4 Deep learning

Deep learning (DL) is a subset of machine learning, relies on artificial neural networks characterized by their depth. This depth refers to the presence of multiple layers comprising inputs, outputs, and neurons. Within each layer, modules transform input data into meaningful information utilized by subsequent layers for task-specific predictions. This layered architecture enables the system to autonomously learn to process its data. [26] DL excels in feature extraction, feature selection, and classification tasks.

2.5 Neural Networks (Terminology)

The structure of a neural network consists of different interconnected layers that are responsible for processing information. A typical neural network architecture includes the following:

- **Input Layer:** This is the initial layer that receives input data from the network. Each neuron in this layer corresponds to an input feature or variable. For example, in image processing, each neuron in this layer may represent a pixel.
- **Hidden layers:** Hidden layers are located between the input and output layers and perform intermediate computations on the received inputs. Each hidden layer consists of many neurons that process the input data and transmit the results to the neurons in the next layer. The number and configuration of hidden layers is designed according to the complexity of the task at hand.
- **Output layer:** This last layer generates the results of the neural network. Each neuron in this layer represents a class, category, or output value. For example, in image classification, neurons may refer to specific categories such as dog, cat, or car.[27].

  As shown in Figure 2.3

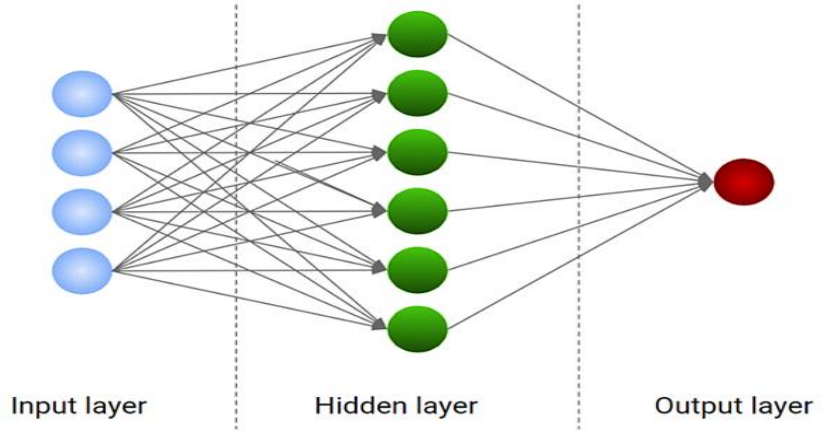


Figure 2.3 : The architecture of neural Networks [42]

2.5.1 Single perceptron

The perceptron, a type of binary neuron, produces an output of either 0 or 1. Its output is determined by performing a weighted sum of its inputs, multiplying each input by a corresponding weight. Mathematically, this can be represented as:

$$Y = (w_1 x_1 + w_2 x_2 + w_3 x_3) \quad (2.1)$$

Where:

Y: This is the output value or dependent variable. In different contexts, it can refer to a prediction, a calculated value, or a result from a linear combination of inputs

w_i: These are the weights (coefficients) associated with each input variable, i indicates that there are multiple weights

x_i: These are the input variables or features. Partial letter, i indicates that there are multiple input variables

Subsequently, the neuron applies a threshold activation function: if the weighted sum exceeds a certain threshold value, the neuron outputs 1; otherwise, it outputs 0. Consequently, the perceptron is primarily used for classification tasks, categorizing input data into distinct classes based on learned patterns. While its output is binary, the perceptron can still be used for prediction tasks by assigning one class to represent positive outcomes and the other for negative outcomes. Thus, the perceptron's capabilities extend beyond mere classification, encompassing prediction.

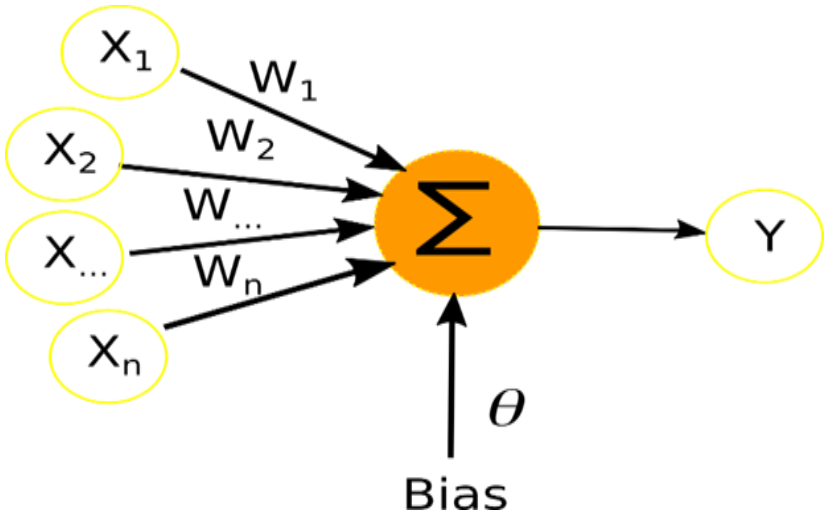


Figure 2.4 : Schematic of the simple perceptron [43]

2.6 Neural Networks Vocabulary

We have outlined the fundamentals of machine learning as a precursor to delving into the realm of deep learning. Yet, grasping the intricacies of algorithmic learning and the vocabulary associated with neural network architecture is paramount. By familiarizing ourselves with these concepts, we gain insight into constructing deep learning models. Expanding on this foundation will facilitate a comprehensive understanding of how deep learning architectures are designed and optimized for various task.

2.6.1 Gradient descent

Gradient Descent is a widely used optimization algorithm in machine learning that aims to minimize the cost function and determine the optimal synaptic weights for a model. Its main objective is to identify the global or local minimum of the cost function by iteratively adjusting the model's weights. The algorithm relies on computing the gradient of the cost function with respect to the model's weights, which quantifies the slope of the cost function and determines the direction in which the weights should be adjusted to minimize the cost function. [29]

As shown in Figure 2.5. The revised equation for gradient descent now reads as follows:

$$w = w' - \alpha \frac{dJ(\alpha)}{dw} \tag{2.2}$$

$$b = b' - \alpha \frac{dJ(w,b)}{db} \tag{2.3}$$

Where:

J: This represents the cost function or loss function, the cost function $J(w,b)$ measures how well the model's predictions match the actual data. It measures the error between the predicted outputs.

α: This is the learning rate. The learning rate is a hyperparameter that determines the size of the steps taken to reach the minimum of the cost function.

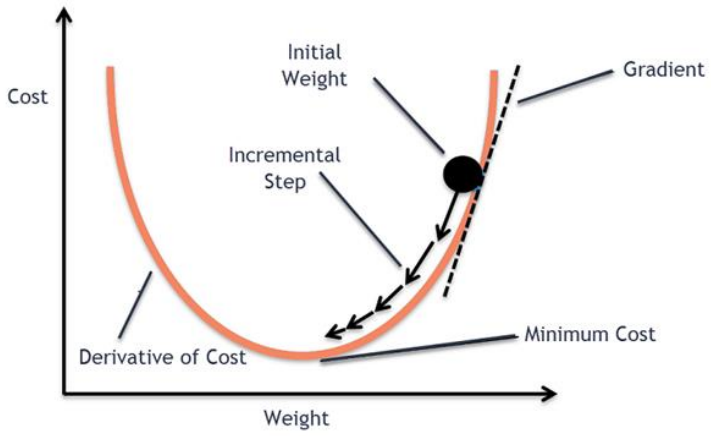


Figure 2.5 : Weight update by gradient descent in the cost function [29]

2.6.2 Backpropagation

In the realm of neural networks, the pivotal task of updating weights and biases is achieved through the sophisticated mechanism known as Backpropagation. This intricate process unfolds after the computation of predicted outputs and subsequent error assessment. Through Backpropagation, the error gradient traverses backward across the network layers, systematically unveiling the contribution of each layer to the overall error. This dynamic chain of gradients enables the computation of gradients for weights and biases at every layer, laying the groundwork for iterative parameter updates.

Leveraging optimization algorithms like gradient descent, these updates are orchestrated meticulously to navigate the vast parameter space, steering towards the minimization of the cost function. This strategic pursuit not only refines the model's performance but also fortifies its ability to generalize to unseen data, thus encapsulating the essence of continual learning and adaptation in neural networks. As shown in Figure 2.6.

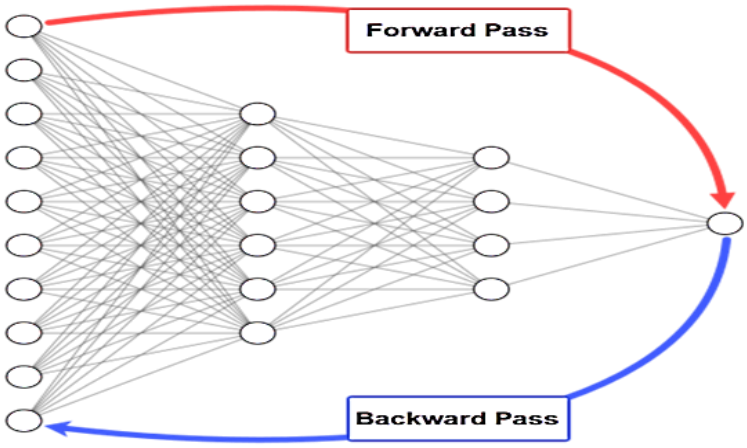


Figure 2.6 : Illustration of backpropagation [45]

Where:

Forward pass is: Calculate the output of the network from input data.

Backward pass is: Calculate the output error relative to the expected output and then go back in the network and update the weights using gradient descent.

2.6.3 Learning Rate:

Learning rate is one of the critical parameters to be tuned when training neural networks using backpropagation and gradient descent. If the learning rate is set too low, it may cause slow training progress as very small updates are made to the weights.

Conversely, if it is set too high, it may lead to undesirable divergent behavior in the loss function. Therefore, it is necessary to choose an appropriate learning rate that allows achieving a balance between the speed of progress in training and stability in improving network performance [30].

2.6.4 Batches

When training a neural network, the training data is typically divided into mini-batches, each containing a specified number of samples known as the batch size. These mini-batches play a crucial role in updating the model parameters during Backpropagation [31].

This division of data into mini-batches enables efficient parameter updates and facilitates training. Moreover, the depicted figure demonstrates a dedicated data block for learning, where each data sample corresponds to a batch. This approach ensures a systematic and effective training process, enhancing the model's performance.

2.6.5 Epochs

The integration of backpropagation and forward propagation forms an epoch in the training of the network, indicating the number of iterations the algorithm executes [32]. This equates to the number of times batches of data traverse through the network, providing crucial insights into the learning progress [33]. Monitoring the passage of training data through mini-batches, coupled with weight updates, serves as a vital gauge for halting the learning process judiciously. Essentially, this aids in validating learning efficacy on test data (test set). For instance, if we consider a training set comprising 500 images and opt for a batch size of 25 images, the resulting epoch value would be 20. This signifies 20 iterations involving both forward and backward propagation, leading to 20 updates for parameters. Among these operations in neural networks, some are classified as hyperparameters and parameters:

- Hyperparameters: Neural networks have a wide range of hypermeters to be set, including the number of layers, the number of neurons in each layer, the learning rate, the Batch Size, the Number of Epochs, the strength of the organization and gradient descent [34]
- Parameters: Neural networks have a wide range of parameters that need to be set, including the Weights and Biases, the Activation function and Feature Extractors.

2.7 Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) are specialized neural networks designed to process data with a grid-like topology, particularly effective in image and video recognition and classification tasks, such as identifying faces, objects, and traffic signs, and aiding in self-driving

cars Moreover [35], CNNs have recently shown effectiveness in various natural language processing tasks, including sentence classification. Structurally, CNNs are a type of feed-forward neural network [36] characterized by one or more layers of hidden neurons (hidden layers) positioned between input and output layers. Each layer is connected to the subsequent layer, facilitating linear signal propagation without cycles or interconnections, thereby providing the network with a global perspective and enhancing neuron interactions. These networks are distinguished by their superior performance with inputs like images, speech, or audio signals and typically comprise three main types of layers:

- convolutional layers
- pooling layers
- fully connected (FC) layers

The convolutional layer, serving as the initial layer, analyzes input data using convolution operations, while subsequent layers may include additional convolutional or pooling layers. The final layer is typically fully connected. As data progresses through the network, it undergoes increasing complexity, starting with the identification of simple features like colors and edges in earlier layers. With each subsequent layer, the CNN identifies larger elements or shapes within the input data until the intended object is recognized. As shown in Figure 2.7 Principal Architecture of CNN. as shown in Figure 2.7 Principal Architecture of CNN.

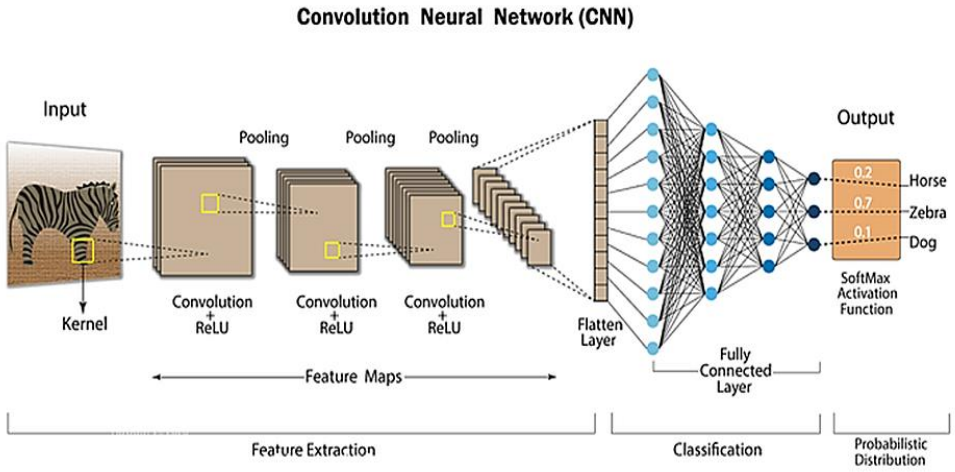


Figure 2.7 : Principal Architecture of CNN[49]

2.7.1 Convolution layers

Color images serve as inputs to Convolutional Neural Networks (CNNs), comprising a 3D matrix of pixels representing height, width, and depth, corresponding to the RGB channels. In this context, the depth dimension signifies the presence of color channels, allowing for the representation of a wide spectrum of colors. Concurrently, within the CNN architecture, feature detectors referred to interchangeably as kernels or filters navigate across the receptive fields of the image, examining the presence of specific features through a process termed convolution.

This convolutional process involves overlaying the feature detector onto various parts of the image to detect patterns or features of interest, enabling the CNN to progressively learn hierarchical representations of the input data. Through iterative convolutional operations, the network discerns increasingly complex features, essential for tasks such as object recognition and image classification [37]. As shown in Figure 2.8.

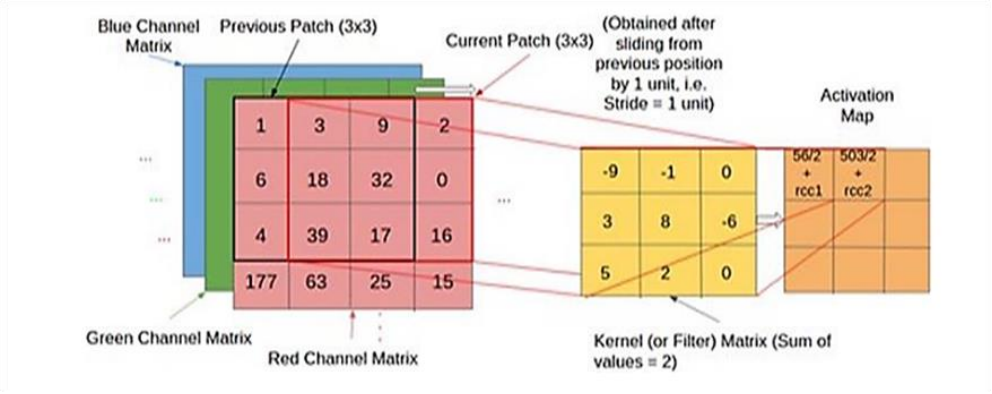


Figure 2.8 : Convolution layer [36]

2.7.2 Feature detector

In Convolutional Neural Networks, feature detectors are represented as 2D arrays of weights, capturing portions of the input image. Typically, these filters are 3x3 matrices, determining the size of the receptive field. Padding is employed to manage image dimensions during convolution. Three common types of padding include:

- valid padding (no padding) also referred to as no padding, which entails discarding the last convolution if dimensions fail to align.
- same padding (ensuring output size matches input) guarantees the output layer matches the size of the input layer.
- full padding (increasing output size by adding zeros) expands the output size by zero-padding the input's border.

After each convolution operation, a Rectified Linear Unit (ReLU) adjustment is applied to the feature map, introducing nonlinearity to the model. The ReLU function is a widely used activation function in neural networks, allowing for the passage of positive values without alteration while setting negative values to zero, thereby enhancing the network's ability to learn complex patterns from the data.

- **ReLU Function**

Although it appears to be a linear function, ReLU has a derivative function that facilitates backpropagation, preserving computational efficiency. Its distinguishing feature is that not all neurons are activated at the same time As shown in Figure 2.9. Also its equation as follows:

$$f(x) = max(0, x) \tag{2.4}$$

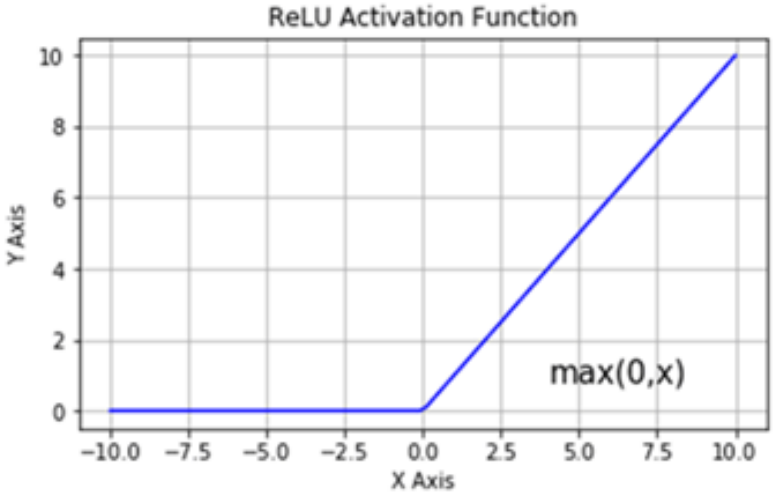


Figure 2.9 : ReLU Function

2.7.3 Pooling Layer

Pooling layers, also known as upsampling layers, reduce the spatial dimensions of the data entering (CNNs) while preserving the underlying information. These layers divide the input data into smaller regions known as pooling windows or receptive fields, and perform pooling within each window, effectively reducing the size of the feature maps. Using downsampling, pooling layers reduce dimensions to reduce factors in the input data. Similar to a convolutional layer, the pooling process involves scanning a filter across the entire input, but without weights. Instead, the kernel uses an aggregate function to fill the output matrix based on the values in the receptive field. This method falls into two main categories, each serving distinct purposes within the CNN architecture.

- **Max Pooling:** it is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after the max-pooling layer would be a feature map containing the most prominent features of the previous feature map. [38] As shown in Figure 2.10.

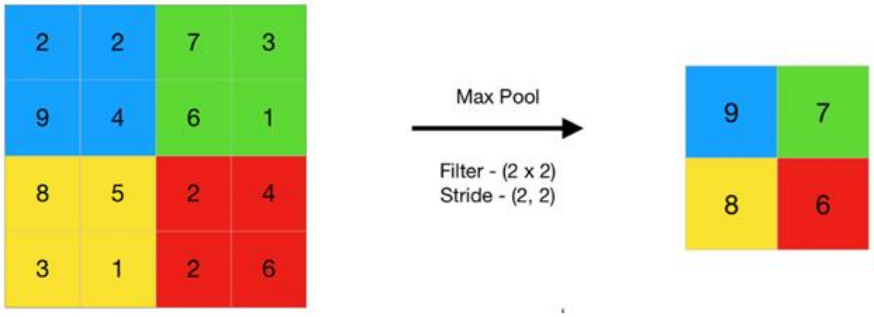


Figure 2.10 : Max Pooling [37]

2.7.4 Fully-Connected Layer

The term "fully-connected layer" is quite descriptive. In contrast, in partially connected layers, the input image's pixel values don't directly link to the output layer, as previously mentioned. Instead, each node in the output layer connects directly to a node in the preceding layer in the fully connected layer. This layer is responsible for performing classification tasks by leveraging features extracted by earlier layers and their respective filters. While convolutional and pooling layers commonly employ ReLU (Rectified Linear Unit) functions for input categorization, fully-connected layers typically utilize a SoftMax activation function to generate probabilities ranging from 0 to 1.

SoftMax ensures that the output values represent valid probabilities, which can be interpreted as the likelihood of the input belonging to each class in a classification problem. This final layer's outputs can be further processed using techniques such as cross-entropy loss to train the neural network during the optimization process, enabling it to learn to make accurate predictions. As shown in Figure 2.11.

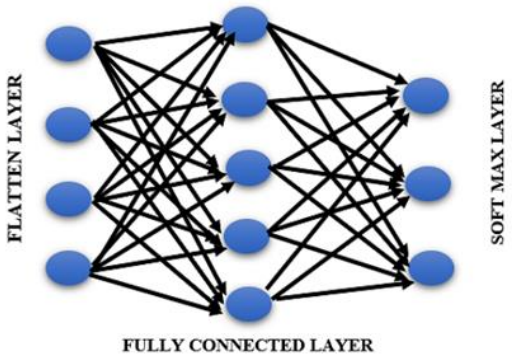


Figure 2.11 : fully connected layer

2.8 Conclusion

Important concepts in the field of deep learning are reviewed in this chapter, including its definition and the architecture of deep models such as deep artificial neural networks (CNNs). In addition, an overview of the benefits of deep learning and its various applications was provided, with a focus on the superior performance of deep neural networks in the field of image processing, which is one of the most important technologies in this field.

The next chapter aims to provide more details on the design of the models and tools used within the research framework, where the methods and techniques used to implement the specific objective will be explored. Using CNNs as a basis for the research work, the focus will be on analyzing model design, and clarifying the methods used to implement deep learning with high accuracy for image super-resolution, using tools and techniques appropriate for the purpose, which contributes to achieving the desired results efficiently and effectively.

Chapter 3 : *Deep learning-based super-resolution optimization algorithms*

Summary

This chapter is dedicated to presenting and discussing the results of our simulation.

We will apply traditional interpolation algorithms and deep learning-based SR

algorithms and then a quantitative comparison study are performed in terms of PSNR and SSIM

Table of Contents

3.1 Introduction

This chapter presents the results of our study on super-resolution (SR) algorithms, where the focus was on evaluating and exploring a variety of these algorithms. We started by using conventional SR methods as a reference to compare with more sophisticated algorithms based on deep learning, such as SRCNN, VDSR, EDSR, and most importantly the DWSR method.

To verify the effectiveness of the algorithms we used, we made sure to have the original high-resolution (HR) images. We then reduced the resolution by factors of x2 and x4. After that, we used one of the proposed methods to restore the HR images and compared them with the originals. This process was essential to ensure a fair comparison with previous research.

Our evaluation involved both quantitative and qualitative analyses using metrics such as peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM), as well as visual quality assessments. By calculating the average PSNR and SSIM values from all experiments, we gained an overview of the performance of the different SR algorithms studied.

3.2 Computational Environment

In our experimental setup, we used Python 3.9.7 with the Visual Studio code editor on an HP computer with an i5 vPro processor, operating within the Windows 10 Professional environment. For our computational tasks, particularly those related to deep learning, we used PyTorch, a library specifically designed for such purposes. Additionally, we employed scikit-image to handle various image processing functions.

To handle large datasets and for additional computational power, we utilized both Google Colab and Kaggle platforms. These platforms facilitated the execution of essential source code, particularly for processing extensive datasets, by utilizing the online GPU resources available through Google Colab. Furthermore, we employed MATLAB R2022a in certain experiments to conduct comparisons and validate the obtained results.

3.3 Data sets

For the datasets used, we selected a variety of well-known datasets. For training purposes, we utilized the DIV2K [50] and T91 datasets. For model testing, we employed several datasets including Set5[51], Set14[52], and LIVE1. In addition, we used a specialized dataset, referred to as 'Medical'[55], for testing models in healthcare-related image processing. Details of these datasets are summarized in the table below.

Table 3.1 : Details of Datasets Utilized in Our Experiments.

Datasets	Number of Images	Format	Resolution	Train/Test
DIV2K	1000	PNG	2040*1356	Train
T91	91	H5	220*187	Train
Set5	05	BMP	512*512	Test
Set14	14	BMP	500*480	Test
LIVE1	29	PNG	768*512	Test
Medical	743	PNG	512*420	Test

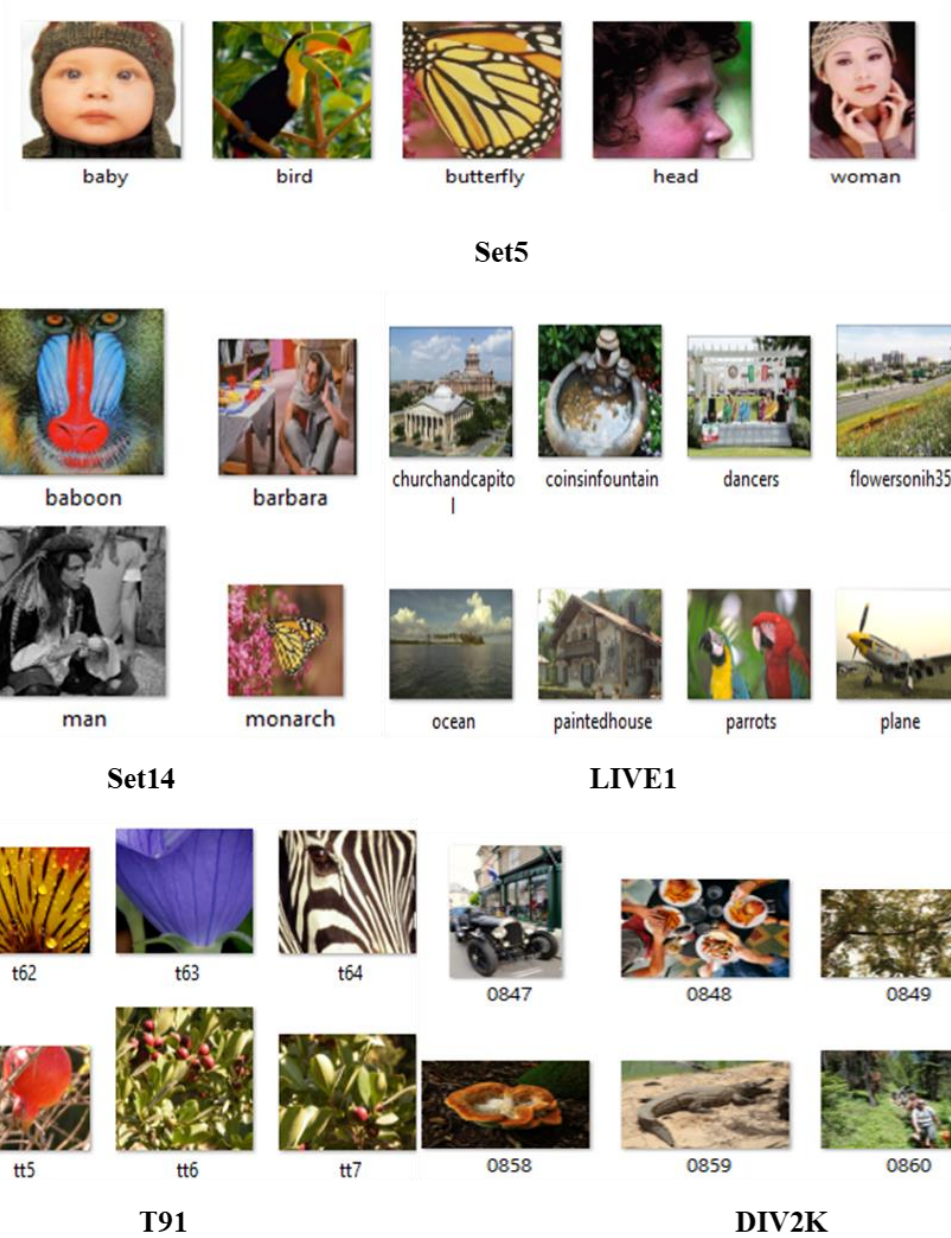


Figure 3.1 : Original high-resolution (HR) images

In our experiments, we utilized X-ray body images. This dataset consists of a collection of X-ray body images commonly used in medical imaging studies and machine learning projects in healthcare. The images are provided in DICOM format, which is a standard for handling, storing, printing, and transmitting medical imaging information. To facilitate accessibility and use in various applications, the images have been converted to PNG format. Figure 3.2 shows samples from the dataset X-ray body images.

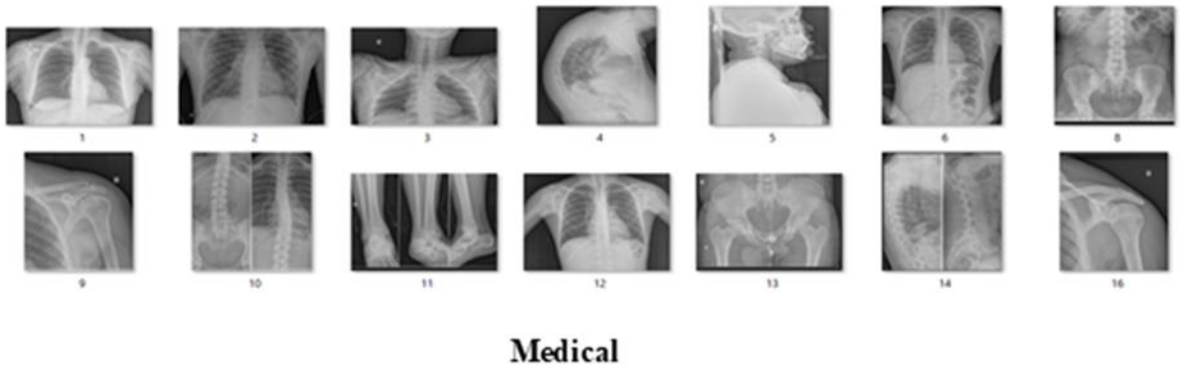


Figure 3.2 : Samples from the dataset X-ray body images.

3.4 Implementation of SR Algorithms

3.4.1 Conventional SR Algorithms

In Figure 3.3, the flowchart illustrates the steps for implementing conventional super-resolution (SR) methods.

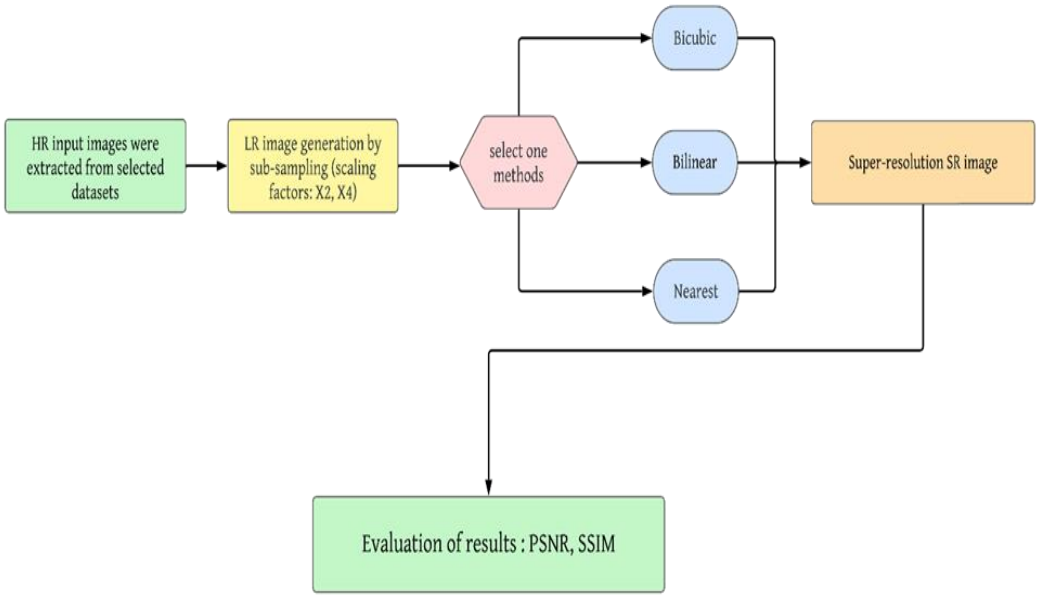


Figure 3.3 : Procedural Flowchart for Conventional Super-Resolution Methods.

Table 3.2 presents the results achieved using bicubic, bilinear, and nearest neighbor methods. In this experiment, images were downsampled by factors of 2 and 4, then reconstructed into HR images using one of these methods. Subsequently, the PSNR and SSIM metrics were calculated to compare the 'HR original' input images with the 'HR obtained' output images.

Table 3.2 : PSNR (dB), SSIM for Set5, conventional methods.

Methods	Scale	Metrices	Bird	Butterfly	Woman
Bicubic	X2	PSNR	**38.32**	**30.28**	**33.34**
		SSIM	**0.90**	**0.95**	**0.96**
	X4	PSNR	**33.52**	**27.22**	**30.14**
		SSIM	**0.72**	**0.82**	**0.87**
Bilinear	X2	PSNR	36.60	30.13	30.40
		SSIM	0.85	0.90	0.93
	X4	PSNR	30.13	26.89	27.01
		SSIM	0.70	0.80	0.85
Nearest Neighbor	X2	PSNR	36.90	30.30	30.80
		SSIM	0.85	0.91	0.94
	X4	PSNR	30.40	27.00	27.35
		SSIM	0.71	0.81	0.87

Figure 3.4 : The SR image obtained by utilizing the Bicubic method on the LR image

Figure 3.5 : The SR image obtained by utilizing the Nearest method on the LR image

Figure 3.6 : The SR image obtained by utilizing the Bilinear method on the LR image

The analysis of the results, as shown in the accompanying table and images, indicates that bicubic interpolation gave higher PSNR and SSIM values compared to the nearest neighbor and bilinear methods. However, in terms of visual quality, the differences among the images produced by these methods were minimal. This observation was consistent across all the test datasets utilized in the study.

3.4.2 Deep Learning-Based Methods for Super Resolution

3.4.2.1 SRCNN

Before presenting the results obtained with the SRCNN method in Figure 3.7, we provide a flowchart that outlines the key stages of our implementation process.

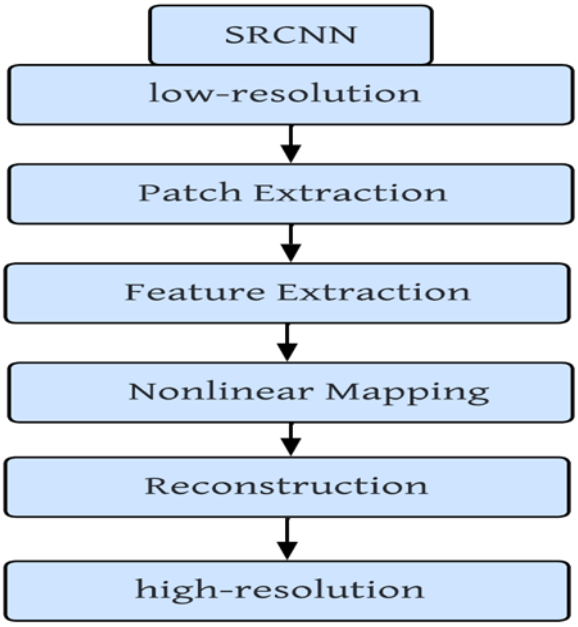


Figure 3.7 : Flowchart of SRCNN

For Deep Learning-Based Methods, we calculated two types of PSNR: $PSNR_{in}$, which measures the difference between the original and the LR image, and $PSNR_{out}$, which compares the original with the HR image obtained. Similarly, we calculated SSIM values for the same comparisons.

Tables 3.3, 3.4, 3.5, and 3.6 summarize the $PSNR_{IN}$, $PSNR_{out}$ and SSIMs results obtained by applying the SRCNN technique on Set5, Set14, LIVE1 Part1, and LIVE1 Part2 respectively. Where $PSNR_{in}$ calculated between the input and output and $PSNR_{out}$

Table 3.3 : Show the $PSNR_{in}$& $SSIM_{in}$ of (LR) images and obtained$PSNR_{out}$ & $SSIM_{out}$ of SRCNN of Set 5.

Name	**X2**				**X4**			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
baby	18.15	38.53	0.70	0.96	17.98	33.12	0.66	0.87
bird	15.11	**40.91**	0.73	**0.97**	14.97	**32.52**	0.64	**0.88**
butterfly	15.29	32.74	0.71	0.95	14.52	25.44	0.62	0.82
head	20.37	35.72	0.69	0.83	20.14	32.44	0.58	0.74
woman	19.48	35.36	0.76	0.96	18.65	28.88	0.63	0.87

The data presented in Table 3.3 reveals significant increases in PSNR values for the Set 5 datasets. Taking the 'baby' image as an example, the PSNR increased by approximately 20.38dB and 15.14dB for downsampling factors of 2 and 4, respectively.

Similarly, the SSIM values improved from 0.70 to 0.96 for a factor of 2, and from 0.66 to 0.87 for a factor of 4. Similar improvements were observed in the other datasets, Set 14, LIVE1 Part 1, and LIVE2 Part 2.

Overall, the SRCNN method has notably enhanced the PSNR values, clearly indicating an improvement in visual quality. The LR and super-resolution SR images generated using the SRCNN method, as detailed in Tables 3.3 to 3.6, are shown in Figures 3.8 to 3.15, alongside their HR counterpart

Table 3.4 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of SRCNN of Set 14

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
bridge	17.88	27.83	0.75	0.86	17.01	23.76	0.54	0.63
foreman	22.36	36.46	0.79	0.96	21.26	32.11	0.77	0.88
face	20.29	35.70	0.67	0.83	20.07	32.38	0.63	0.72
man	25.32	31.04	0.65	0.88	23.03	26.89	0.61	0.72
barbara	18.97	28.63	0.67	0.86	18.35	25.76	0.62	0.70
comic	18.46	28.52	0.62	0.87	17.08	22.69	0.52	0.58
coastguard	20.45	30.82	0.73	0.84	20.28	26.04	0.49	0.53
monarch	16.97	**37.74**	0.81	0.97	16.02	30.22	0.59	0.90
pepper	14.20	36.87	0.61	0.85	14.12	**32.97**	0.58	0.77
lenna	15.72	36.63	0.64	0.87	14.59	31.40	0.61	0.78
zebra	17.35	33.49	0.76	0.90	16.45	27.16	0.55	0.67
flowers	16.21	33.31	0.69	0.86	15.50	26.09	0.57	0.68
ppt3	16.57	31.52	0.77	0.94	15.54	24.80	0.69	0.83
baboon	14.88	25.73	0.72	0.58	13.99	22.73	0.43	0.47

Table 3.5 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of SRCNN of LIVE1 Part 1

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
bikes	20.63	30.06	0.83	0.91	19.08	24.25	0.55	0.67
building2	19.29	25.67	0.79	0.86	17.12	20.69	0.49	0.59
buildings	20.75	25.96	0.70	0.84	18.50	21.64	0.48	0.59
caps	18.22	37.15	0.86	0.93	18.08	**32.79**	0.67	0.84
carnivaldolls	17.50	32.51	0.82	0.93	15.03	27.40	0.71	0.80
cemetry	18.81	29.46	0.69	0.88	17.36	23.68	0.51	0.66
churchandcapitol	20.47	29.40	0.81	0.90	18 .23	24.24	0.64	0.75
coinsinfountain	21.36	32.50	0.83	0.91	20.20	27.27	0.68	0.74
dancers	19.19	27.07	0.78	0.87	17.12	22.79	0.54	0.66
flowersonih35	17.50	25.15	0.79	0.87	16.06	20.88	0.57	0.61
house	20.70	32.96	0.77	0.89	20.15	28.48	0.59	0.71
lighthouse2	21.15	30.85	0.85	0.90	21.00	26.45	0.68	0.74
lighthouse3	21.55	30.03	0.72	0.87	20.30	25.33	0.63	0.70

Table 3.6 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of SRCNN of LIVE1 Part 2

Name	X2				X4			
	$PSNR_{in}$	$PSNR_{out}$	$SSIM_{in}$	$SSIM_{out}$	$PSNR_{in}$	$PSNR_{out}$	$SSIM_{in}$	$SSIM_{out}$
manfishing	21.65	29.84	0.81	0.86	19.45	25.64	0.62	0.70
monarch	16.98	**37.74**	0.85	0.97	16.63	30.22	0.83	0.90
ocean	25.27	33.14	0.69	0.87	24.30	29.66	0.66	0.71
paintedhouse	22.33	29.10	0.70	0.89	20.90	25.13	0.61	0.69
parrots	16.53	38.82	0.87	0.96	16.39	32.13	0.80	0.89
plane	21.02	34.97	0.81	0.92	20.38	30.16	0.76	0.82
rapids	18.32	31.98	0.78	0.89	17.83	26.97	0.63	0.69
sailing1	21.45	29.66	0.76	0.85	20.43	25.82	0.58	0.63
sailing2	24.42	35.03	0.83	0.92	23.16	29.63	0.78	0.81
sailing3	26.23	34.81	0.81	0.92	24.73	29.78	0.80	0.80
sailing4	22.67	31.73	0.69	0.88	21.65	27.49	0.60	0.69
statue	24.62	34.34	0.83	0.93	23.51	29.87	0.78	0.82
stream	19.84	29.96	0.71	0.80	18.47	22.29	0.47	0.51
studentsculpture	21.32	27.65	0.79	0.87	18.93	22.72	0.50	0.58
woman	20.08	29.81	0.75	0.88	19.26	25.85	0.61	0.67
womanhat	18.39	35.85	0.79	0.91	18.18	31.40	0.73	0.78

Figure 3.8 : SRCNN result Set 5, scale factor x2

Figure 3.9 : SRCNN result Set 5, scale factor x4

In Figure 3.10, the region of interest (ROI) for low- and high-resolution images is shown to illustrate the difference between them.

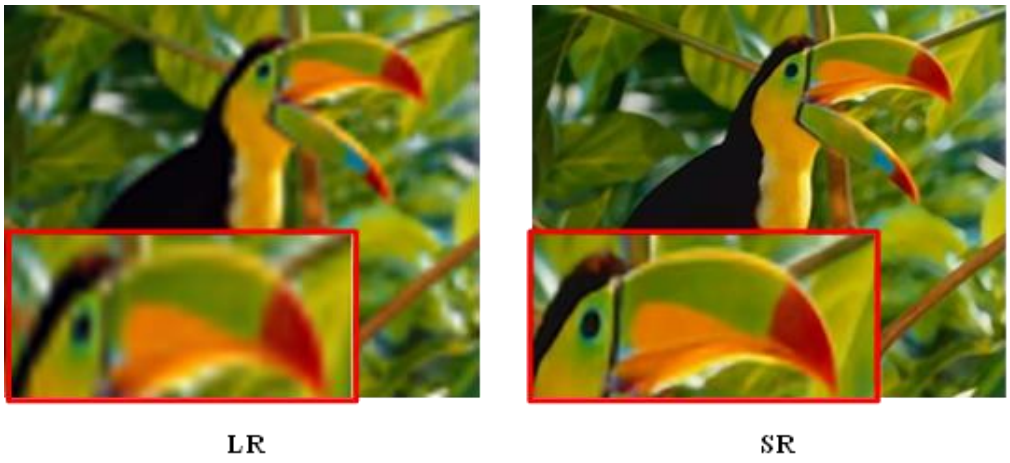


Figure 3.10 : Bird (ROI) of SRCNN

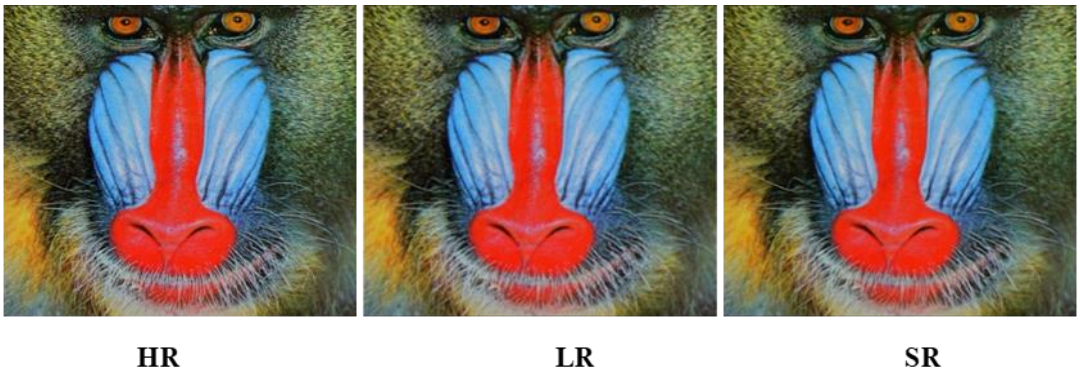


Figure 3.11 : SRCNN result Set 14, scale factor x2

HR **LR** **SR**

Figure 3.12 : SRCNN result Set 14, scale factor x4

HR **LR** **SR**

Figure 3.13 : SRCNN result LIVE1, scale factor x2

HR **LR** **SR**

Figure 3.14 : SRCNN result LIVE1, scale factor x4

LR **SR**

Figure 3.15 : buildings (ROI) of SRCNN

After transitioning from the aforementioned conventional methods to SRCNN, one of the algorithms based on deep learning, the model was trained using lr=1e-4, batch-size=16, and num-epochs=400. These parameters provided us with better results on the test data.

3.4.2.2 VDSR

After implementing the SRCNN algorithm, we subsequently executed the VDSR algorithm and computed the PSNR and SSIM values for all the images obtained as test results for this algorithm, as shown in Tables 3.7, 3.8, 3.9, and 3.10. This assessment enables us to evaluate the effectiveness and performance level of this latter algorithm. Figure 3.16 illustrates the key steps of our implementation in a flowchart format.

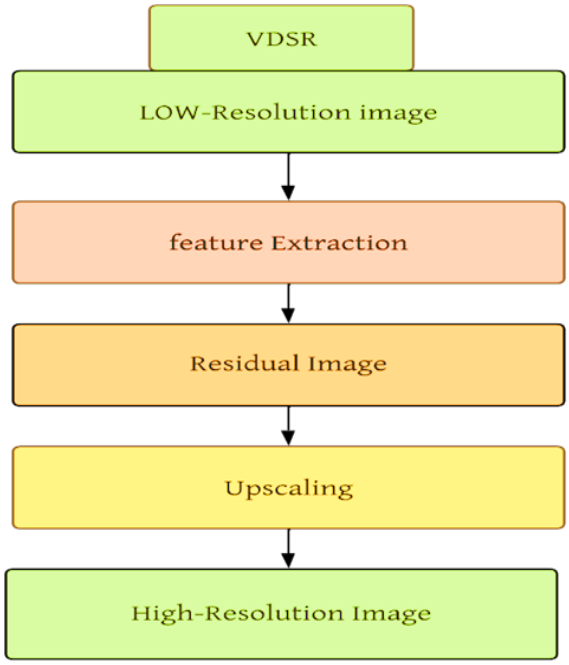


Figure 3.16 : Flowchart of VDSR

Table 3.7 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of VDSR of Set 5

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
baby	18.15	**39.80**	0.70	0.97	17.98	**35.12**	0.66	0.93
bird	15.11	39.49	0.73	0.98	14.97	32.76	0.64	0.92
butterfly	15.29	32.30	0.71	0.96	14.52	25.92	0.62	0.88
head	20.37	36.67	0.69	0.89	20.14	34.24	0.58	0.84
woman	19.48	36.20	0.76	0.98	18.65	30.09	0.63	0.94

Based on the obtained results shown in Table 3.7, which shows test data for our study, we observed a significant improvement in PSNR and SSIM values. For instance, when we consider the "bird" image, we noted an improvement of 24.38 dB and 17. 79 dB for 2 and 4 downsampling factors respectively.

In addition, the SSIM valeus increased from 0.73 to 0.98 and from 0.64 to 0.92 for the same two downsampling factors. This improvement applies to all the data used in the study (Set14, LIVE1-Part1 and LIVE1-Part2), given our results we can say that VDSR method was effective. The LR and SR images generated using the VDSR method, as detailed in Tables 3.7 to 3.10, are shown in Figures 3.17 to 3.24, along with their HR counterparts.

Table 3.8 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of VDSR of Set 14

Name	**X2**				**X4**			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
bridge	17.88	31.29	0.75	0.93	17.01	27.92	0.54	0.85
foreman	22.36	37.02	0.79	0.95	21.26	31.60	0.77	0.94
face	20.29	36.63	0.67	0.89	20.07	**34.23**	0.63	0.83
man	25.32	33.18	0.65	0.95	23.03	29.64	0.61	0.89
barbara	18.97	31.05	0.67	0.92	18.35	28.35	0.62	0.83
comic	18.46	29.66	0.62	0.93	17.08	25.06	0.52	0.80
coastguard	20.45	32.67	0.73	0.92	20.28	28.86	0.49	0.82
monarch	16.97	**37.27**	0.81	0.98	16.02	31.11	0.59	0.94
pepper	14.20	34.51	0.61	0.90	14.12	31.12	0.58	0.85
lenna	15.72	36.63	0.64	0.91	14.59	32.57	0.61	0.85
zebra	17.35	34.19	0.76	0.96	16.45	28.39	0.55	0.88
flowers	16.21	33.13	0.69	0.93	15.50	28.23	0.57	0.81
ppt3	16.57	31.00	0.77	0.97	15.54	26.48	0.69	0.91
baboon	14.88	26.86	0.72	0.71	13.99	24.60	0.43	0.54

Table 3.9 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of VDSR of LIVE 1 Part 1

Nom	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
bikes	20.63	31.35	0.83	0.94	19.08	26.85	0.55	0.84
building2	19.29	27.31	0.79	0.91	17.12	23.46	0.49	0.75
buildings	20.75	28.20	0.70	0.91	18.50	24.78	0.48	0.82
caps	18.22	**38.78**	0.86	0.96	18.08	**34.73**	0.67	0.93
carnivaldolls	17.50	32.57	0.82	0.95	15.03	26.76	0.71	0.90
cemetry	18.81	30.34	0.69	0.92	17.36	26.32	0.51	0.94
churchandcapitol	20.47	30.70	0.81	0.93	18 .23	25.66	0.64	0.93
coinsinfountain	21.36	33.64	0.83	0.95	20.20	31.16	0.68	0.95
dancers	19.19	28.75	0.78	0.92	17.12	23.90	0.54	0.89
flowersonih35	17.50	27.14	0.79	0.91	16.06	23.48	0.57	0.76
house	20.70	34.93	0.77	0.95	20.15	31.27	0.59	0.87
lighthouse2	21.15	32.85	0.85	0.95	21.00	29.32	0.68	0.89
lighthouse3	21.55	31.84	0.72	0.93	20.30	28.34	0.63	0.87

Table 3.10 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of LIVE 1 Part 2

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
manfishing	21.65	31.02	0.81	0.92	19.45	25.36	0.62	0.91
monarch	16.98	37.31	0.85	0.98	16.63	31.15	0.83	0.94
ocean	25.27	35.49	0.69	0.94	24.30	32.62	0.66	0.89
paintedhouse	22.33	31.34	0.70	0.93	20.90	28.23	0.61	0.85
parrots	16.53	39.45	0.87	0.97	16.39	34.47	0.80	0.95
plane	21.02	35.71	0.81	0.96	20.38	31.67	0.76	0.92
rapids	18.32	33.56	0.78	0.94	17.83	29.35	0.63	0.85
sailing1	21.45	31.68	0.76	0.92	20.43	28.77	0.58	0.85
sailing2	24.42	36.15	0.83	0.96	23.16	31.78	0.78	0.92
sailing3	26.23	36.01	0.81	0.96	24.73	32.32	0.80	0.91
sailing4	22.67	33.65	0.69	0.94	21.65	30.21	0.60	0.87
statue	24.62	36.06	0.83	0.96	23.51	32.22	0.78	0.92
stream	19.84	28.40	0.71	0.90	18.47	25.56	0.47	0.79
studentsculpture	21.32	29.31	0.79	0.91	18.93	25.18	0.50	0.76
woman	20.08	32.06	0.75	0.93	19.26	28.65	0.61	0.84

Figure 3.17 : VDSR result Set5, scale factor x2

Figure 3.18 : VDSR result Set5, scale factor x4

Figure 3.19 : Bird (ROI) of VDSR

HR LR SR

Figure 3.20 : VDSR result Set14, scale factor x2

HR LR SR

Figure 3.21 : VDSR result Set14, scale factor x4

HR LR SR

Figure 3.22 : VDSR result Set14, scale factor x2

HR LR SR

Figure 3.23 : VDSR result Set14, scale factor x4

Figure 3.24 : buildings (ROI) of VDSR

3.4.2.3 EDSR

Following the execution of the VDSR algorithm, we proceeded with the implementation of the EDSR algorithm. Subsequently, we calculated the PSNR and SSIM metrics for all images obtained as test results for this algorithm, as documented in Tables 3.11, 3.12, 3.13,3.14 , and 3.15. This evaluation allows us to assess the effectiveness and performance of the EDSR algorithm. Figure 3.25 presents the essential steps of our implementation in the form of a flowchart.

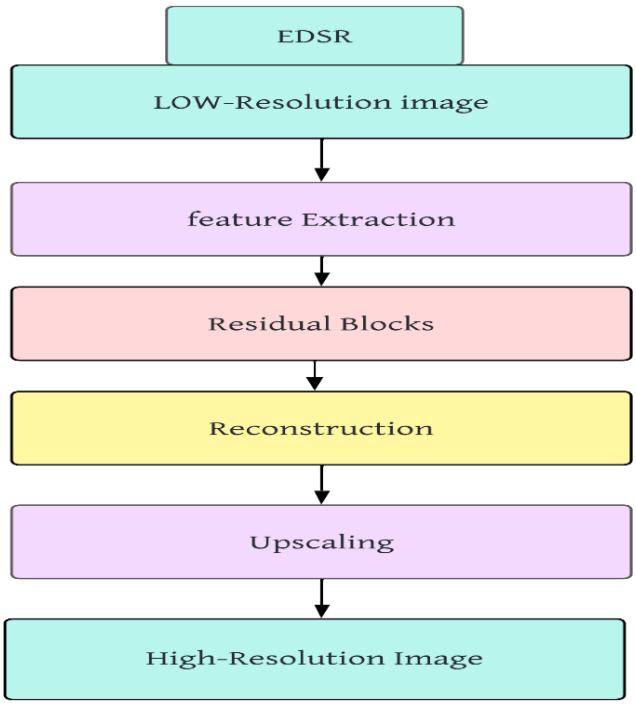


Figure 3.25 : Flowchart of EDSR

Table 3.11 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of EDSR Set 5.

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
baby	18.15	42.15	0.70	0.99	17.98	42.10	0.66	0.98
bird	15.11	**43.15**	0.73	0.99	14.97	**42.76**	0.64	0.99
butterfly	15.29	31.30	0.71	0.99	14.52	30.24	0.62	0.97
head	20.37	33.84	0.69	0.98	20.14	32.55	0.58	0.90
woman	19.48	38.22	0.76	0.99	18.65	37.80	0.63	0.99

Based on the results shown in Table 3.11, which contains a set of test data, we can observe notable improvements in both PSNR and SSIM metrics. Taking the "bird" image as a case study, we noted an increase in PSNR of 28.04 dB and 27.79 dB for downsampling factors of 2 and 4, respectively. Similarly, SSIM values improved from 0.73 to 0.99 at a factor of 2 and from 0.64 to 0.99 at a factor of 4, approaching the maximum value of 1. These enhancements were consistently observed across the various data used in the study, including Set 14, LIVE1-Part1, and LIVE1-Part2. These results indicate the effectiveness of the EDSR algorithm. The EDSR algorithm performed well, which led us to test it with medical data. However, despite its strong performance with the initial test data, the algorithm showed only modest improvements in both metrics and visual quality when applied to medical data. The LR and SR images generated using the EDSR method, as detailed in Table 3.11 to 3.15, are shown in Figures 3.26 to 3.36, along with their HR counterparts.

Table 3.12 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of EDSR set 14.

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
Bridge	17.88	31.07	0.75	0.88	17.01	29.20	0.54	0.79
foreman	22.36	36.19	0.79	0.98	21.26	33.96	0.77	0.96
Face	20.29	33.84	0.67	0.93	20.07	32.57	0.63	0.90
Man	25.32	31.14	0.65	0.85	23.03	28.80	0.61	0.73
Barbara	18.97	32.13	0.67	0.96	18.35	31.55	0.62	0.95
Comic	18.46	29.89	0.62	0.94	17.08	28.27	0.52	0.90
coastguard	20.45	37.03	0.73	0.96	20.28	37.18	0.49	0.93
monarch	16.97	**37.21**	0.81	0.99	16.02	36.27	0.59	0.99
Pepper	14.20	34.62	0.61	0.98	14.12	32.66	0.58	0.98
Lenna	15.72	36.50	0.64	0.99	14.59	36.31	0.61	0.99
Zebra	17.35	36.72	0.76	0.98	16.45	**36.36**	0.55	0.97
Flowers	16.21	34.24	0.69	0.97	15.50	32.42	0.57	0.95
ppt3	16.57	30.48	0.77	0.95	15.54	28.77	0.69	0.56
baboon	14.88	26 .59	0.72	0 .91	13.99	24.94	0.43	0.87

Table 3.13 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of EDSR LIVE1 Part1.

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
bikes	20.63	32.00	0.83	0.96	19.08	31.78	0.55	0.94
building2	19.29	28.19	0.79	0.94	17.12	27.60	0.49	0.91
buildings	20.75	28.50	0.70	0.91	18.50	25.55	0.48	0.86
caps	18.22	38.86	0.86	0.98	18.08	38.34	0.67	0.98
carnivaldolls	17.50	33.25	0.82	0.95	15.03	30.76	0.71	0.90
cemetry	18.81	31.12	0.69	0.96	17.36	30.32	0.51	0.94
churchandcapitol	20.47	30.37	0.81	0.95	18 .23	29.60	0.64	0.93
coinsinfountain	21.36	35.34	0.83	0.97	20.20	35.16	0.68	0.95
dancers	19.19	27.74	0.78	0.93	17.12	26.90	0.54	0.89
flowersonih35	17.50	27.57	0.79	0.95	16.06	27.13	0.57	0.92
house	20.70	35.56	0.77	0.98	20.15	35.22	0.59	0.98
lighthouse2	21.15	33.00	0.85	0.97	21.00	32.54	0.68	0.96
lighthouse3	21.55	31.26	0.72	0.96	20.30	29.77	0.63	0.95
manfishing	21.65	30.42	0.81	0.94	19.45	29.36	0.62	0.91

Table 3.14 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of EDSR LIVE1 Part 2.

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
bikes	20.63	32.00	0.83	0.96	19.08	31.78	0.55	0.94
building2	19.29	28.19	0.79	0.94	17.12	27.60	0.49	0.91
buildings	20.75	28.50	0.70	0.91	18.50	25.55	0.48	0.86
caps	18.22	38.86	0.86	0.98	18.08	38.34	0.67	0.98
carnivaldolls	17.50	33.25	0.82	0.95	15.03	30.76	0.71	0.90
cemetry	18.81	31.12	0.69	0.96	17.36	30.32	0.51	0.94
churchandcapitol	20.47	30.37	0.81	0.95	18 .23	29.60	0.64	0.93
coinsinfountain	21.36	35.34	0.83	0.97	20.20	35.16	0.68	0.95
dancers	19.19	27.74	0.78	0.93	17.12	26.90	0.54	0.89
flowersonih35	17.50	27.57	0.79	0.95	16.06	27.13	0.57	0.92
house	20.70	35.56	0.77	0.98	20.15	35.22	0.59	0.98
lighthouse2	21.15	33.00	0.85	0.97	21.00	32.54	0.68	0.96
lighthouse3	21.55	31.26	0.72	0.96	20.30	29.77	0.63	0.95
manfishing	21.65	30.42	0.81	0.94	19.45	29.36	0.62	0.91

Table 3.15 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of EDSR Medical

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
01	14.88	23.46	0.55	0.59	14.21	23.00	0.55	0.57
02	20.36	27.71	0.58	0.63	20.19	**26.59**	0.54	0.60
03	19.29	26.29	0.60	0.66	18.17	25.61	0.51	0.61
04	16.32	24.84	0.58	0.64	15.03	24.56	0.58	0.63
05	15.97	22.75	0.49	0.55	15.00	22.23	0.45	0.52
06	18.46	26.56	0.54	0.60	16.99	25.70	0.50	0.58
07	19.45	26.74	0.59	0.67	18.97	25.45	0.59	0.64
08	17.90	26.59	0.51	0.56	16.33	25.93	0.44	0.52
09	15.23	**28.32**	0.55	0.60	14.19	25.95	0.50	0.57
10	14.69	24.49	0.67	0.61	13.98	22.42	0.49	0.55
11	16.35	25.07	0.58	0.62	15.44	24.49	0.56	0.61
12	16.80	27.01	0.62	0.68	15.36	26.09	0.57	0.65
13	17.49	26.36	0.59	0.66	16.01	24.66	0.56	0.63
14	15.04	26.15	0.67	0.63	14.03	25.23	0.52	0.59

Figure 3.26 : EDSR result Set5, scale factor x2

Figure 3.27 : EDSR result Set5, scale factor x4

Figure 3.28 : bird (ROI) of EDSR

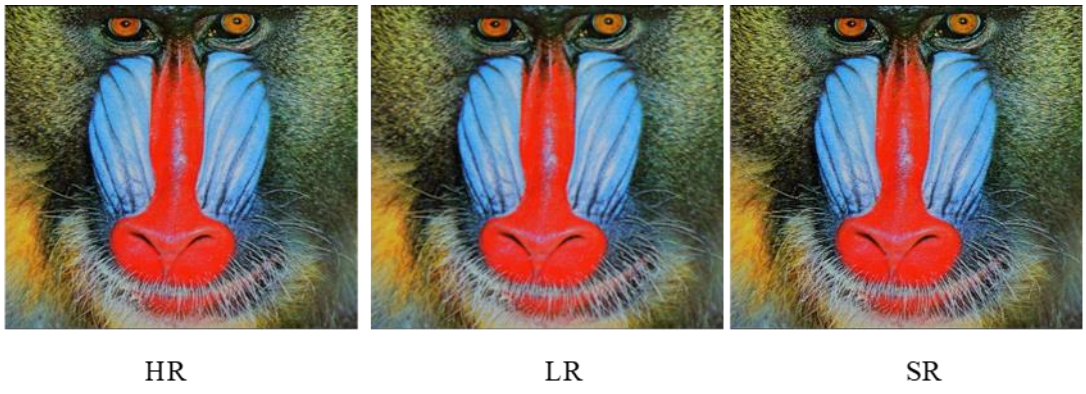


Figure 3.29 : EDSR result Set14, scale factor x2

Figure 3.30 : EDSR result Set14, scale factor x4

Figure 3.31 : EDSR result LIVE1, scale factor x2

Figure 3.32 : EDSR result LIVE1, scale factor x4

Figure 3.33 : buildings (ROI) of EDSR

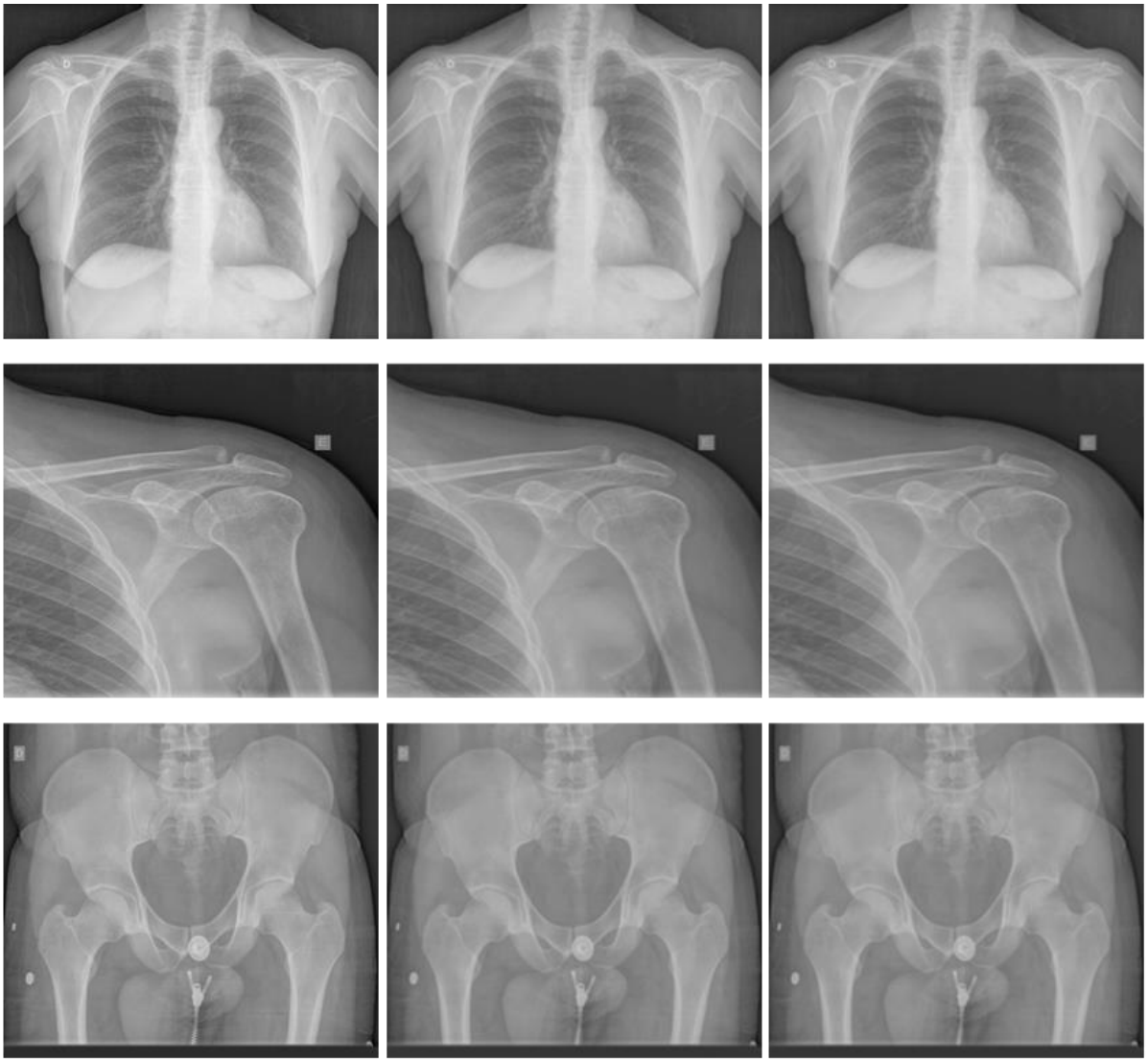


Figure 3.34 : EDSR results Medical, scale factor x2

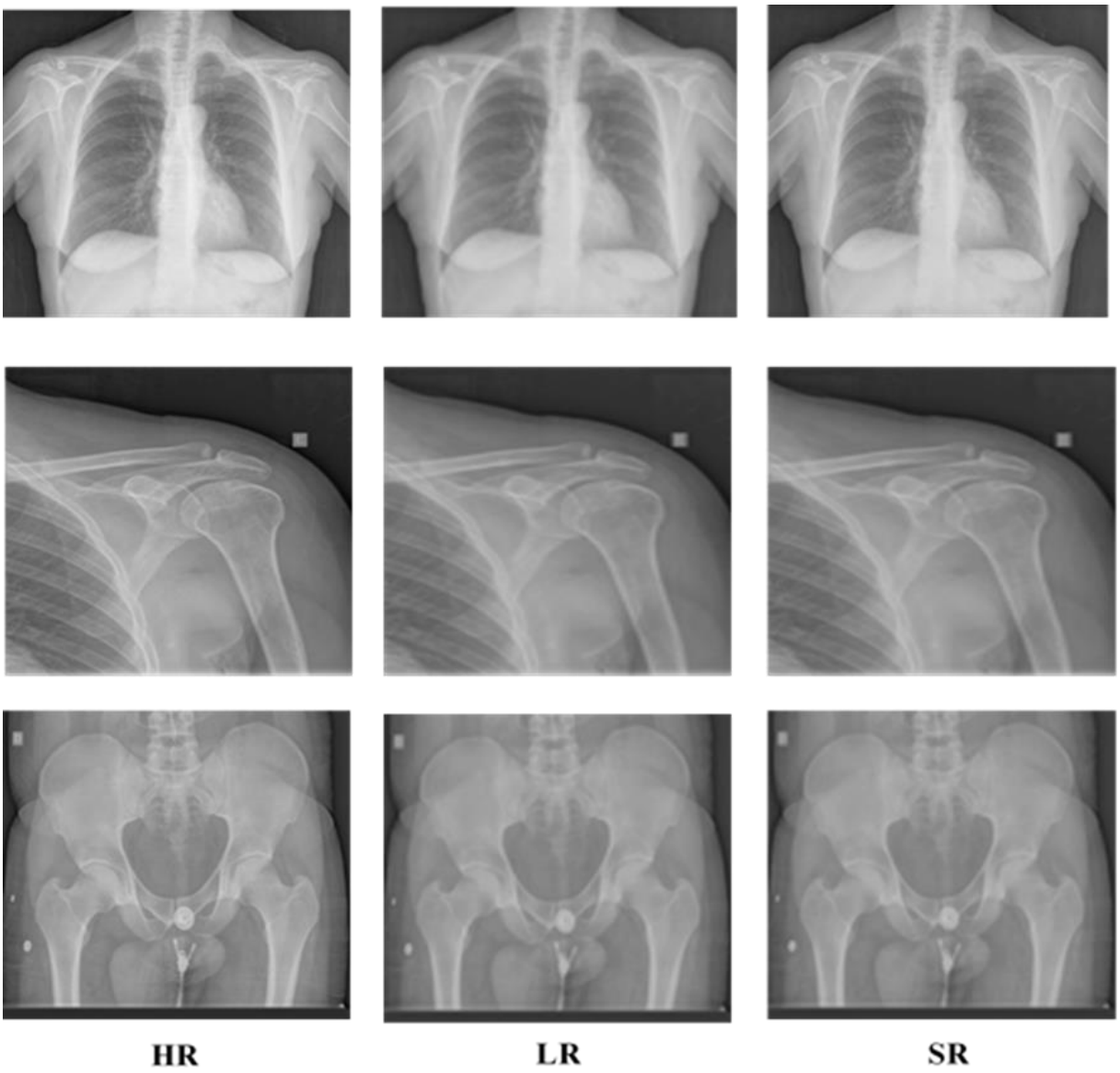


Figure 3.35 : EDSR results Medical, scale factor x4

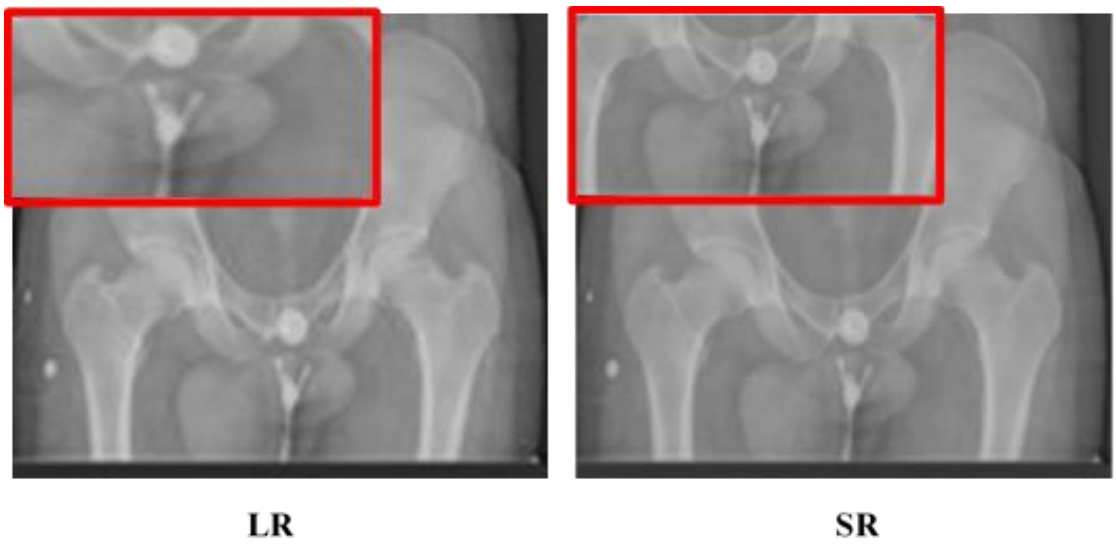


Figure 3.36 :Medical (ROI) of EDSR

3.4.2.4 DWSR

The latest algorithm we implemented was DWSR, where we evaluated its performance by calculating PSNR and SSIM for all its outputs, as shown in the tables 3.6. Figure 3.37 illustrates the key steps of its implementation in the form of a flowchart.

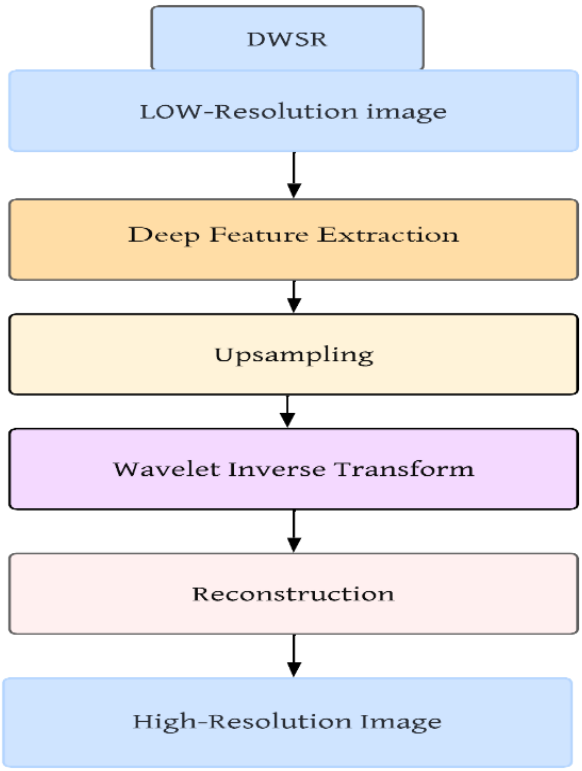


Figure 3.37 : Flowchart of DWSR

Table 3.16 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of DWSR Set5

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
baby	18.15	39.91	0.70	0.96	17.98	31.83	0.66	0.91
bird	15.11	**40.56**	0.73	0.99	14.97	**33.22**	0.64	0.96
butterfly	15.29	34.57	0.71	0.98	14.52	27.41	0.62	0.94
head	20.37	35.10	0.69	0.89	20.14	29.71	0.58	0.81
woman	19.48	36.55	0.76	0.98	18.65	30.51	0.63	0.95

Based on the findings presented in Tables 3.16 and 3.20, which encompass a comprehensive set of test data in our study, notable improvements are observed in both PSNR and SSIM values. Taking the "bird" image as a prime example, enhancements of 25,45 dB and 18,25 dB are recorded for downsampling factors of 2 and 4, respectively. Correspondingly, SSIM values exhibit a substantial increase from 0.73 to 0.99 at factor 2, and from 0.64 to 0.96 at factor 4. This consistent enhancement is evident across most of the datasets used in our study, including Set 14, LIVE1-Part1, and LIVE1-Part2. Particularly noteworthy are the results obtained from the Medical dataset, where the algorithm demonstrates exceptional performance both in terms of metrics and visual quality. For instance, in image" 9", we observe significant PSNR values of 18,33dB and 17,85dB for r downsampling factors 2 and 4, respectively.

Additionally, there is a remarkable improvement in SSIM values from 0.55 to 0.95 and 0.50 to 0.67 for the same two downsampling factors, underscoring the effectiveness of the DWSR method, especially for this image type.The LR and SR images generated using the DWSR method, as delineated in Tables 3.16 to 3.20, are visually depicted in Figures 3.38 to 3.48, alongside their HR counterparts.

Table 3.17 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of DWSR Set 14

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
bridge	17.88	28.98	0.75	0.81	17.01	24.90	0.54	0.50
foreman	22.36	33.30	0.79	0.97	21.26	29.00	0.77	0.92
face	20.29	32.07	0.67	0.89	20.07	29.69	0.63	0.81
man	25.32	30.07	0.65	0.78	23.03	29.94	0.61	0.52
barbara	18.97	29.82	0.67	0.88	18.35	26.12	0.62	0.77
comic	18.46	30.57	0.62	0.93	17.08	28.39	0.52	0.73
coastguard	20.45	29.41	0.73	0.80	20.28	27.44	0.49	0.60
monarch	16.97	**37.38**	0.81	0.99	16.02	**32.99**	0.59	0.97
pepper	14.20	31.63	0.61	0.98	14.12	29.67	0.58	0.97
lenna	15.72	33.75	0.64	0.98	14.59	29.84	0.61	0.97
zebra	17.35	32.77	0.76	0.98	16.45	28.35	0.55	0.91
flowers	16.21	31.11	0.69	0.95	15.50	28.34	0.57	0.84
ppt3	16.57	30.83	0.77	0.90	15.54	26.83	0.69	0.80
baboon	14.88	25.39	0.72	0.83	13.99	23.60	0.43	0.65

Table 3.18 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of DWSR LIVE1 Part1

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
bikes	20.63	29.81	0.83	0.94	19.08	29.47	0.55	0.93
building2	19.29	24.48	0.79	0.87	17.12	24.12	0.49	0.86
buildings	20.75	25.86	0.70	0.88	18.50	25.75	0.48	0.86
caps	18.22	36.45	0.86	0.97	18.08	35.83	0.67	0.97
carnivaldolls	17.50	30.00	0.82	0.92	15.03	29.00	0.71	0.90
cemetry	18.81	28.05	0.69	0.92	17.36	26.94	0.51	0.90
churchandcapitol	20.47	28.31	0.81	0.92	18 .23	27.31	0.64	0.92
coinsinfountain	21.36	30.76	0.83	0.92	20.20	29.95	0.68	0.90
dancers	19.19	25.98	0.78	0.89	17.12	25.01	0.54	0.87
flowersonih35	17.50	23.82	0.79	0.90	16.06	23.45	0.57	0.89
house	20.70	32.10	0.77	0.94	20.15	31.70	0.59	0.93
lighthouse2	21.15	29.88	0.85	0.93	21.00	29.76	0.68	0.92
lighthouse3	21.55	29.33	0.72	0.92	20.30	29.22	0.63	0.91

Table 3.19 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of DWSR LIVE1 Part2.

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
manfishing	21.65	27.98	0.81	0.88	19.45	26.65	0.62	0.85
monarch	16.98	37.36	0.85	0.99	16.63	36.06	0.83	0.99
ocean	25.27	32.11	0.69	0.87	24.30	32.02	0.66	0.83
paintedhouse	22.33	28.02	0.70	0.91	20.90	27.78	0.61	0.90
parrots	16.53	**37.84**	0.87	0.98	16.39	**37.02**	0.80	0.98
plane	21.02	33.56	0.81	0.91	20.38	33.23	0.76	0.89
rapids	18.32	31.20	0.78	0.92	17.83	30.41	0.63	0.91
sailing1	21.45	28.77	0.76	0.91	20.43	28.67	0.58	0.91
sailing2	24.42	34.40	0.83	0.90	23.16	34.03	0.78	0.89
sailing3	26.23	34.56	0.81	0.94	24.73	34.21	0.80	0.93
sailing4	22.67	30.99	0.69	0.90	21.65	30.79	0.60	0.89
statue	24.62	33.34	0.83	0.92	23.51	33.16	0.78	0.92
stream	19.84	24.77	0.71	0.87	18.47	24.70	0.47	0.86
studentsculpture	21.32	26.03	0.79	0.88	18.93	25.65	0.50	0.86
woman	20.08	28.61	0.75	0.89	19.26	28.40	0.61	0.88
womanhat	18.39	34.43	0.79	0.98	18.18	33.69	0.73	0.98

Table 3.20 : Show the $PSNR_{in}$& $SSIM_{in}$ of LR images and obtained $PSNR_{out}$ & $SSIM_{out}$ of DWSR Medical

Name	X2				X4			
	$PSNR_{in}$	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**	**$PSNR_{in}$**	**$PSNR_{out}$**	**$SSIM_{in}$**	**$SSIM_{out}$**
01	14.88	28.31	0.55	0.90	14.21	28.03	0.55	0.64
02	20.36	32.71	0.58	0.91	20.19	31.96	0.54	0.62
03	19.29	31.20	0.60	0.92	18.17	30.73	0.51	0.68
04	16.32	29.66	0.58	0.87	15.03	29.51	0.58	0.65
05	15.97	27.59	0.49	0.89	15.00	27.31	0.45	0.64
06	18.46	31.53	0.54	0.81	16.99	30.84	0.50	0.61
07	19.45	31.66	0.59	0.81	18.97	30.68	0.59	0.64
08	17.90	31.49	0.51	0.92	16.33	31.15	0.44	0.60
09	15.23	**33.56**	0.55	0.95	14.19	**32.04**	0.50	0.67
10	14.69	29.86	0.67	0.94	13.98	28.57	0.49	0.64
11	16.35	29.96	0.58	0.93	15.44	29.56	0.56	0.61
12	16.80	31.94	0.62	0.87	15.36	31.39	0.57	0.75
13	17.49	31.51	0.59	0.96	16.01	30.45	0.56	0.64
14	15.04	31.07	0.67	0.97	14.03	30.61	0.52	0.63

Figure 3.38 : DWSR result Set5, scale factor x2

Figure 3.39 : DWSR result Set5, scale factor x4

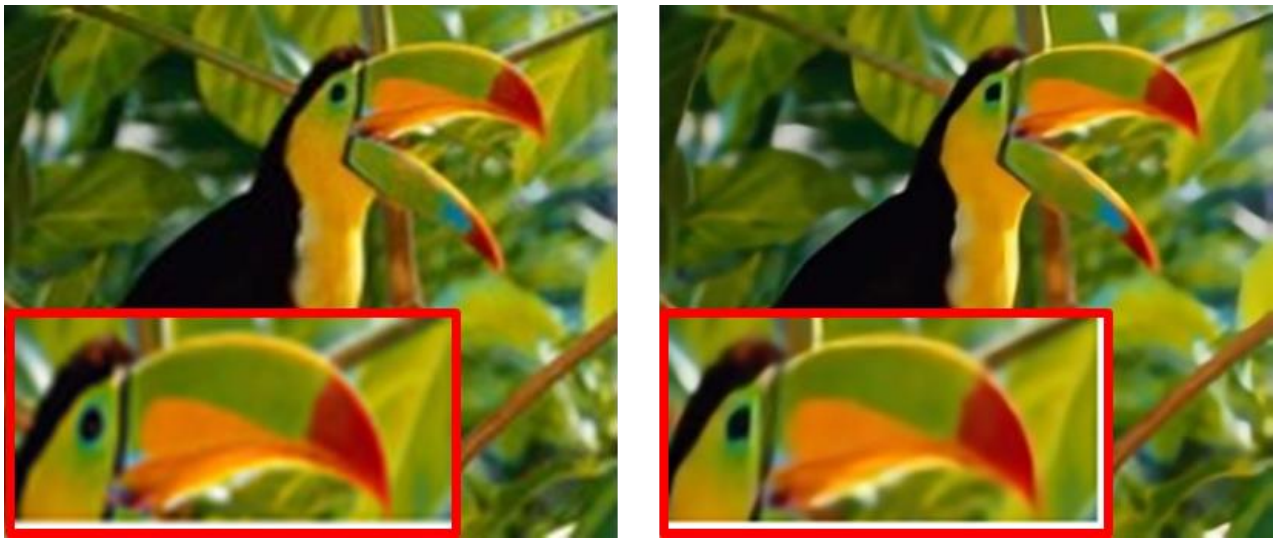


Figure 3.40 : bird (ROI) of DWSR

HR LR SR

Figure 3.41 : DWSR result Set14, scale factor x2

HR LR SR

Figure 3.42 : DWSR result Set14, scale factor x4

HR LR SR

Figure 3.43 : DWSR result LIVE1, scale factor x2

HR LR SR

Figure 3.44 : DWSR result LIVE1, scale factor x4

Figure 3.45 : building (ROI) of DWSR

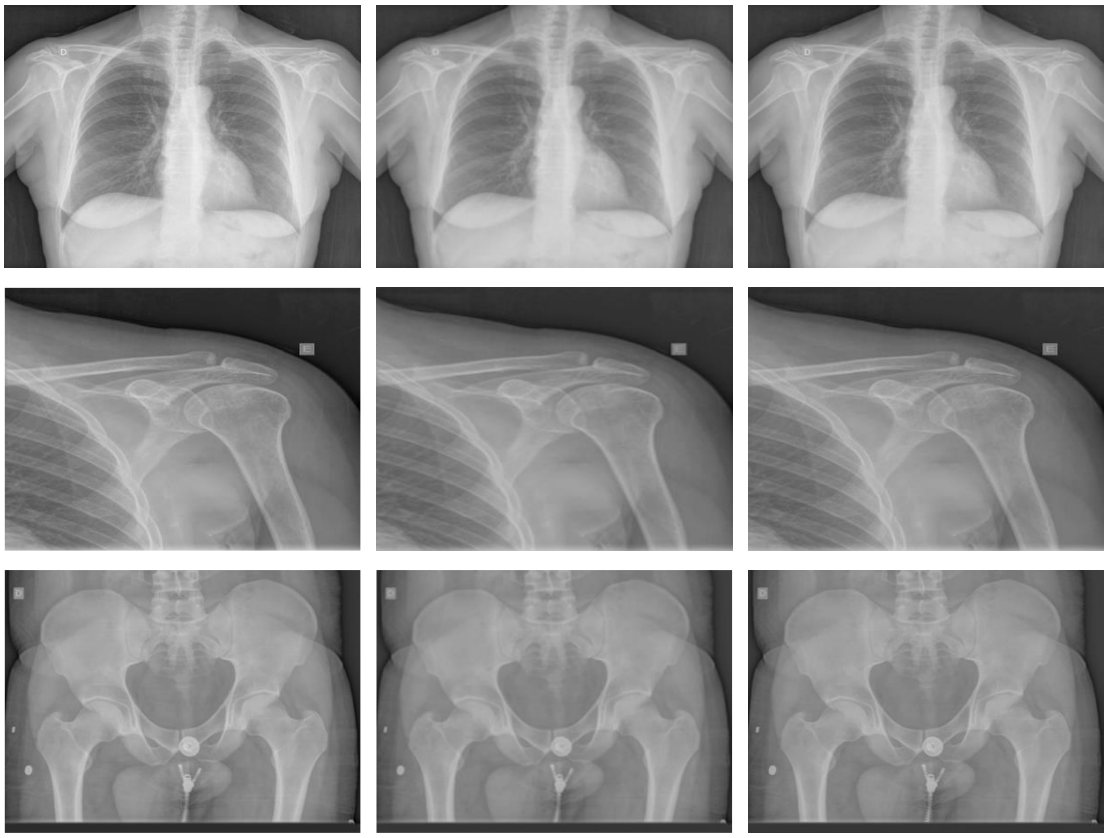
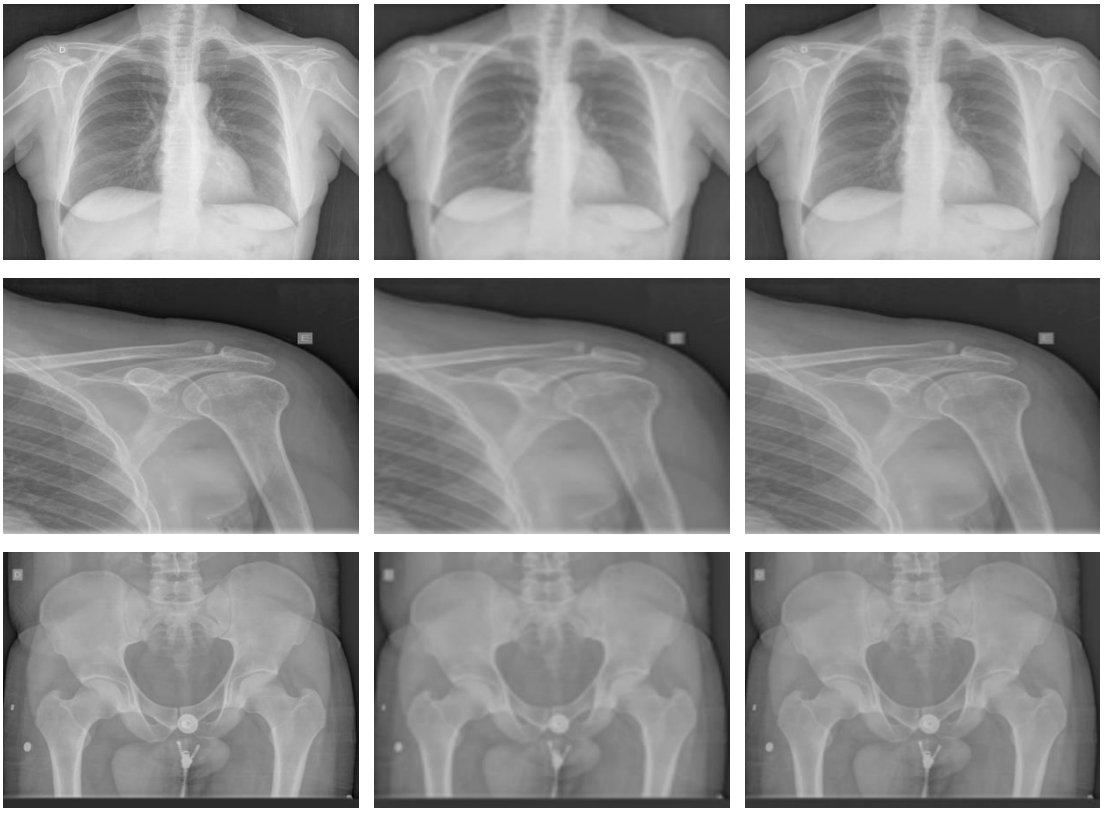


Figure 3.46 : DWSR results Medical, scale factor x2



HR **LR** **SR**

Figure 3.47 : DWSR results Medical, scale factor x4

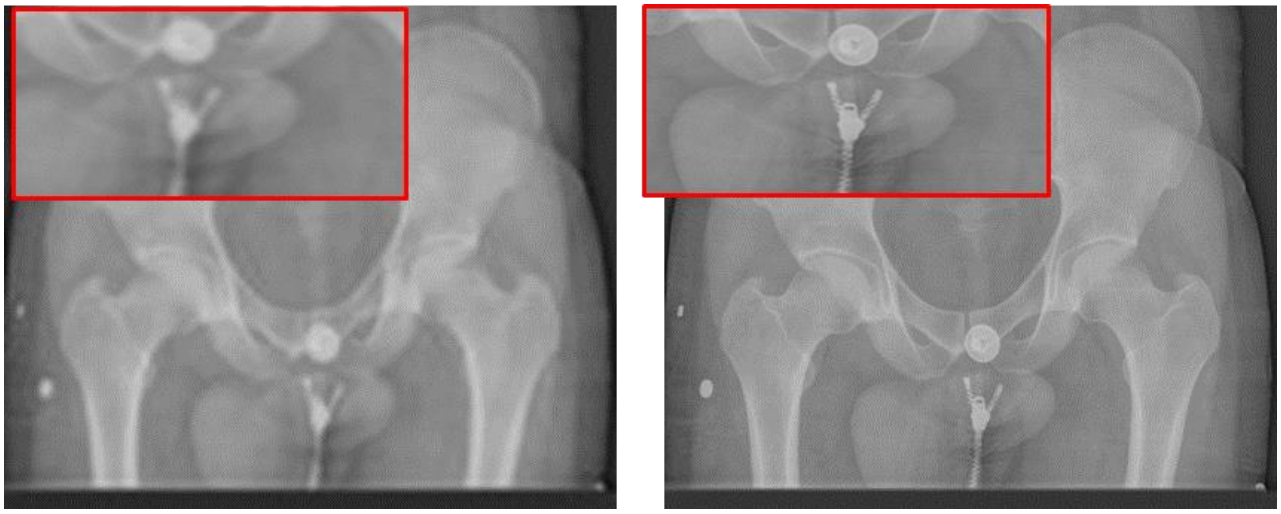


Figure 3.48 : Medical (ROI) of DWSR

Table 3.21 : The average results of PSNR (dB) and SSIM of SRCNN, VDSR, EDSR and DWSR

Dataset	Scale	SRCNN	VDSR	EDSR	DWSR
Set5	**X2**	36.65/0.94	36.89/0.95	37.73/0.98	37.33/0.96
	X4	30.48/0.84	31.62/0.90	37.09/0.96	30.53/0.91
Set14	**X2**	32.44/0.87	33.22/0.91	33.40/0.94	31.22/ 0.90
	X4	27.50/0.70	29.15/0.83	32.09/0.89	28.22/0.78
LIVE1	**X2**	31.48/0.89	32.86/0.93	33.14/0.95	30.30/0.91
	X4	26.57/0.71	28.86/0.87	32.32/0.93	29.79/0.90
Medical	**X2**	--	--	25.88/ 0.62	30.84/0.89
	X4	--	--	24.85/0.59	30.17/0.64

3.5 Discussion

After completing our study on the implementation of classical methods and deep learning algorithms, we recorded several observations:

For the conventional methods, the highest and best value of PSNR and SSIM was recorded when bicubic was implemented on the test data, from which we conclude that it is better performing than its counterpart, where the difference was about 1dB, and this applies to all test data.When visually comparing two images with a difference of about 1dB between them, it is difficult to see this difference in quality easily, so the larger the difference in PSNR value, the better the visualization of the difference.

When examining the results of the deep learning algorithms as shown in Tables 3.6, which represents the average values of PSNR and SSIM for each algorithm and previewing the outputs of the images, it is clear that EDSR performed more effectively than SRCNN, VDSR and DWSR, where we recorded a difference of 2dB versus SRCNN, and compared to VDSR and DWSR there was a difference of 1dB, and this applies to the following test data Set5, Set14 and LIVE1. while the SSIM value was close to 1 in some cases :

- So we can say that it is the best option to get high-resolution images on this type of images.

When our study proved that EDSR is the best option, we decided to test its effectiveness with its counterpart DWSR on Medical images as shown in Table 3.21 The reason we chose DWSR for comparison is that it works in the frequency domain while the other algorithms work in the spatial domain, so we decided to test its effectiveness.

From the results obtained as shown in Table 3.21, we can see that DWSR was more effective on medical test data compared to EDSR, so we can say that it is the best option in improving the image quality on this type of images, we observed a 5 dB difference for medical data at factor 2 and 4, in addition, we recorded a difference in SSIM value of 0.27 and 0.5 for the same two downscaling factors respectively.

In addition, we wanted to make the differences clearly visible so we mapped the region of interest (ROI) on a number of test data as shown in Figure 3.32 and 3.46 where the quality differences were well visualized.

Deep learning-based algorithms have proven to be much more effective than traditional methods in terms of PSNR and SSIM as well as output quality, with a significant improvement in the latter.

3.6 Conclusion

In this chapter of the thesis, we studied and implemented methods and algorithms to improve image quality and resolution, including classical methods (bicubic, nearest neighbor, and bilinear) and deep learning algorithms (SRCNN, VDSR, EDSR, DWSR), where we generated low quality and resolution images from high quality and resolution images, and then evaluated the effectiveness of the latter two by calculating metrics (PSNR, SSIM).

Based on the results obtained with respect to the classical methods, bicubic interpolation proved to be more effective than its conventional counterparts.

As for the results of the neural network algorithms, we found EDSR to be the best choice due to its image output compared to the other algorithms, with better metric values by a significant difference of 2 dB and in some cases 3 dB, for the test data (Set5, Set14, LIVE1).

In the same experiment, we tested the DWSR and EDSR algorithms on medical test data, and the observation was that DWSR performed significantly better both on the quality of the obtained images, where visual inspection was easy due to the obvious difference, and on the metric values.

In conclusion, the above-mentioned neural network-based algorithms proved to be more effective in performance in terms of improving image quality and resolution, as we observed that the majority of the image outputs were of high resolution and were successful and desirable solutions in solving.

General conclusion

General Conclusion

The work involves reconstructing a high-resolution image from a low-resolution image

This thesis deals with the comparison of two methods for improving the accuracy of image interpolation: Traditional interpolation algorithms and deep learning-based super-interpolation algorithms.

First: we studied conventional interpolation algorithms, and three algorithms were implemented: Binary interpolation, cubic interpolation, cubic interpolation, and nearest-neighbor interpolation.

Second: we studied deep learning-based SR image algorithms. Four algorithms SRCNN, VDSR, EDSR, and DWSR were implemented using a set of data including medical images in simulation experiments to evaluate the performance.

The results clearly showed that deep learning-based super-resolution algorithms significantly outperform traditional interpolation methods. These algorithms significantly improved the quality of low-resolution images, as evidenced by the high values of quality metrics such as PSNR and SSIM. Thus, this study confirms the effectiveness and feasibility of solving the issue of detail restoration and improving image resolution using deep learning-based super-resolution methods. In addition, DWSR achieved exceptional results on medical images and was effective in performance.

Recent advances in deep learning have opened up new horizons and exciting perspectives in the field of super-resolution images. Algorithms based on deep learning provide significant improvements in image quality, opening up new horizons in areas such as image processing, image medical, surveillance, and computer vision, among others. Although there are challenges such as performance and compute time, the results obtained so far are promising and point to an exciting future for super-resolution images using deep learning.

Bibliography

Bibliography

[1] K. Nakanishi, S. Maeda, T. Miyato, and D. Okanohara, "Neural Multi-scale Image Compression," 2018.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436–444, 2015.

[3] 2018 IEEE 3rd International Conference on Signal and Image Processing.

[4] M. L. Yahiaoui, "Les techniques de super-résolution SR appliquées en imagerie astronomique et nucléaire."

[5] C. Dong, C. C. Loy, K. He, and X. Tang, "Image Super-Resolution Using Deep Convolutional Networks," 31 Jul. 2015.

[6] R. C. Gonzales and R. E. Woods, Digital Image Processing. Prentice Hall, 2008.

[7] W. Sun and Z. Chen, "Learned Image Downscaling for Upscaling using a Content Adaptive Resampler," in Proc. IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW), 2018.

[8] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 1646-1654, 2016.

[9] S. Mallat, A Wavelet Tour of Signal Processing: The Sparse Way. Academic Press, 2008.

[10] T. Guo, H. S. Mousavi, T. H. Vu, and V. Monga, "Deep wavelet prediction for image super-resolution," in Proc. IEEE Conf. Computer Vision and Pattern Recognition Workshops, pp. 104-113, 2017.

[11] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," IEEE Trans. Comput. Imag., vol. 3, no. 1, pp. 47-57, 2016.

[12] C. Dong, C. C. Loy, K. He, and X. Tang, "Image Super-Resolution Using Deep Convolutional Networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 2, pp. 295-307, Feb. 2016, doi: 10.1109/TPAMI.2015.2439281.

[13] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," IEEE Trans. Image Process., vol. 13, no. 4, Apr. 2004.

[14] J. S. Isaac and R. Kulkarni, "Super resolution techniques for medical image processing," in Proc. Int. Conf. Technologies for Sustainable Development (ICTSD), pp. 1-6, Feb. 2015.

[15] J. H. Hwang, C. K. Park, S. B. Kang, M. K. Choi, and W. H. Lee, "Deep Learning Super-Resolution Technique Based on Magnetic Resonance Imaging for Application of Image-Guided Diagnosis and Surgery of Trigeminal Neuralgia," Life, vol. 14, no. 3, p. 355, 2024.

[16] D. C. Lepcha, B. Goyal, A. Dogra, and V. Goyal, "Image super-resolution: A comprehensive review, recent trends, challenges and applications," Inf. Fusion, vol. 91, pp. 230-260, 2023.

[17] D. Khaledyan, A. Amirany, K. Jafari, M. H. Moaiyeri, A. Z. Khuzani, and N. Mashhadi, "Low-cost implementation of bilinear and bicubic image interpolation for real-time image super-resolution," in Proc. IEEE Global Humanitarian Technology Conf. (GHTC), pp. 1-5, Oct. 2020.

[18] E. Koester and C. S. Sahin, "A comparison of super-resolution and nearest neighbors interpolation applied to object detection on satellite data," arXiv preprint arXiv:1907.05283, 2019.

[19] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," IEEE Signal Process. Mag., vol. 35, no. 1, pp. 53-65, 2018.

[20] W. Sun and Z. Chen, "Learned Image Downscaling for Upscaling using a Content Adaptive Resampler," in Proc. IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW), 2018.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 770–778, 2016.

[22] B. J. Kirsch, Machine Learning for Dummies IBM Limited Edition. Hoboken, NJ: Wiley, s.d.

[23] S. Ray, "Proven Ways for Improving the 'Accuracy' of a Machine Learning Model," 2020.

[24] Z. Wang and J. C. Ye, "Deep Learning for Image Super-Resolution: A Survey," 2021.

[25] "The Difference Between Machine Learning and Deep Learning," Levity AI. [Online]. Available: https://levity.ai/blog/difference-machine-learning-deeplearning#:~:text=Machine%20Learning%20means%20computers%20learning,documents%2C%20images%2C%20and%20text

[26] M. Nielsen, Neural Networks and Deep Learning.

[27] C. Bento, "Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis," 2021. [Online]. Available: https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141

[28] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss function for image restoration with neural networks," arXiv:1511.08861v3 [cs.CV], 2017.

[29] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.

[30] X. Wang, L. Yan, and Q. Zhang, "Research on the application of gradient descent algorithm in machine learning," in Proc. Int. Conf. on Big Data, Artificial Intelligence and Internet of Things Engineering, pp. 1-5, Sep. 2021.

[31] "Setting the learning rate of your neural network."

[32] H. Kinsley and D. Kukieła, Neural Networks from Scratch in Python, p. 658, 2020.

[33] F. Schilling, "The effect of batch normalization on deep convolutional neural networks," 2016.

[34] J. J. Heckman, R. Pinto, and P. A. Savelyev, "Artificial Intelligence, IoT and Machine Learning," Angew. Chem. Int. Ed., vol. 6, no. 11, pp. 951-952, 1967.

[35] A. Romero and A. Romero, "Assisting the training of deep neural networks with applications to computer vision," University of Barcelona, 2015.

[36] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, pp. 1798–1828, 2013.

[37] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, "Subject-independent facial expression recognition with robust face detection using a convolutional neural network," Neural Netw., vol. 16, no. 5-6, pp. 555-559, 2003.

[38] P. Sharma, "A Comprehensive Tutorial to Learn Convolutional Neural Networks," 2020.

*[39]Available:https://miro.medium.com/v2/resize:fit:640/format:webp/1*WKGr-4r4ap4hzLpBXjYxrQ.png*

[41]https://www.redhat.com/rhdc/managed-files/image1_114.png

[42]https://www.yourdatateacher.com/wp-content/uploads/2021/04/neural-network-1024x665.png

[43] https://kongakura.fr/images/image-5d761a9229520.png

[44] C.Bento, "Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis," 2021. [Online]. Available: https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141

[45]https://i0.wp.com/neptune.ai/wp-content/uploads/2022/10/Backpropagation-passes-architecture.png?resize=434%2C414&ssl=1

[46] https://d14b9ctw0m6fid.cloudfront.net/ugblog/wp-content/uploads/2020/12/1-4.png

[47] "CNN Introduction to Pooling Layer," GeeksforGeeks, [Online]. Available: https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/

[48] N. Karayiannis and A. N. Venetsanopoulos, Artificial Neural Networks: Learning Algorithms, Performance Evaluation, and Applications. Springer Science & Business Media, 1992.

[49]A.Pandey, "Basics of CNN in Deep Learning," Analytics Vidhya, Mar. 2022. [Online]. Available:https://www.analyticsvidhya.com/blog/2022/03/basics-of-cnn-in-deep-learning/. [Accessed: May 19, 2024].

[50] Eirikur Agustsson and Radu Timofte. Ntire 2017 challengeon single image superresolution: Dataset and study. In IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 1122–1131, 2017.

[51] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie-Line Alberi Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding.In British Machine Vision Conference, 2012.

[52] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparserepresentations. In Proceedings of the 7th International Conference on Curves and Surfaces, pages 711–730, 2012.

*[53]Hinton, G.E.; Osindero, S.; Teh, Y.-W. A fast learning algorithm for deep belief nets. Neural Comput. **2006**, 18, 1527–1554.*

[54]Taye, M. M. (2023). Understanding of machine learning with deep learning: architectures, workflow, applications and future directions. Computers, 12(5), 91.

[55] https://www.kaggle.com/datasets/ibombonato/xray-body-images-in-png-unifesp-competion