

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH  
University of Mohamed El-Bashir El-Ibrahimi - Bordj Bou Arreridj  
Faculty of Science and Technology  
Department of electronics

# Memory

Presented to get  
THE MASTER'S DIPLOMA

Sector: Electronics

Specialty: Telecommunications systems

By

- Bouraba Nour El Houda
- Rouabeh Lamia

*Intitulé*

## *Study of Image Super Resolution Algorithms*

*Supported on: 22/09/2022*

<i>First name &amp;family name :</i>	<i>Grade :</i>	<i>Quality :</i>	<i>Establishment :</i>
• <i>M.Hacine Gharbi Abd Ennour</i>	<i>MCA</i>	<i>President</i>	<i>Univ-BBA</i>
• <i>M. Messali Zobieda</i>	<i>Prof</i>	<i>Sipervisorrr</i>	<i>Univ-BBA</i>
• <i>M. Hacini Latifa</i>	<i>Prof</i>	<i>Examiner</i>	<i>Univ-BBA</i>



*Aknowlgement :*

I thank God for giving me the strength to accomplish this work to go further.

I would like to thank all the people who contributed to the success of my internship and who helped me during the writing of this thesis.

I would like to express all my gratitude to my thesis director, Mrs. Messali Zoubaida, I thank her for having supervised, guided, helped and advised me and I thank in particular, Mrs. Soumia , Mr. Boudechiche for having given me the extraordinary opportunity to do my fieldwork

I address my sincere thanks to all the professors and to all the people who by their words, their writings, their advice and their criticisms have guided mv reflection.



### ***Didication : Nour el houda***

*I thank my very dear parents, who have always been there for me. I thank my sisters Imane and Siham,,Israa and my brother Sami, for their encouragement.*

*Finally, I would like to thank my friends Chahinez,assia and lamia, who have always been there for me. Their unconditional support and encouragement has been of great help.*

### ***Didication : Lamia***

*I thank my family and my husband for the encouragement who have always been there for me ,and i thank my friend haizia,ibtissam,mouna,louiza, imane for their help.*

***Thanks***

## ***Abstracts :***

Super image resolution (SR) is a group of image processing technologies used in computer vision to improve the resolution of deteriorated images. Deep learning approaches have made great progress in super image resolution in recent years. In this study, we to provide a regular overview of current improvements in image super resolution techniques using deep learning methodologies. Namely, we will describe and implement three SR algorithms : SRCNN, SRGAN and CAR. The comparative study is done in terms of the computation of two criteria peak signal to noise ratio (PSNR) and the structure similarity index (SSIM). The obtained results have demonstrate the efficiency of the three algorithms especially CAR algorithm.

## ***Résumés :***

La super résolution d'image (SR) est un groupe de technologies de traitement d'image utilisées en vision par ordinateur pour améliorer la résolution des images détériorées. Les approches d'apprentissage profond ont fait de grands progrès dans la super résolution d'image ces dernières années. Dans cette étude, nous fournirons un aperçu régulier des améliorations actuelles des techniques de super résolution d'images utilisant des méthodologies d'apprentissage profond. A savoir, nous allons décrire et implémenter trois algorithmes SR : SRCNN, SRGAN et CAR. L'étude comparative est effectuée en termes de calcul du rapport signal sur bruit de crête (PSNR) à deux critères et de l'indice de similarité de structure (SSIM). Les résultats obtenus ont démontré l'efficacité des trois algorithmes en particulier l'algorithme CAR.

## **ملخص**

دقة الصورة الفائقة (SR) هي مجموعة من تقنيات معالجة الصور المستخدمة في رؤية الكمبيوتر لتحسين دقة الصور التالفة. حققت مناهج التعلم العميق خطوات كبيرة في دقة الصورة الفائقة في السنوات الأخيرة. في هذه الدراسة ، سوف نقدم نظرة عامة منتظمة على التحسينات الحالية في تقنيات الصور فائقة الدقة باستخدام منهجيات التعلم العميق. وبالتحديد ، سنقوم بوصف وتنفيذ ثلاث خوارزميات SR: SRCNN و SRGAN و CAR. يتم إجراء الدراسة المقارنة من حيث حساب نسبة ذروة الإشارة إلى الضوضاء (PSNR) ومؤشر التشابه الهيكلي (SSIM). أظهرت النتائج التي تم الحصول عليها كفاءة الخوارزميات الثلاثة ، ولا سيما خوارزمية CAR.

# Summary :

Aknowlgement.....	1
Dedication.....	
Abstract.....	
Table of contents.....	
Table of Tables.....	
General introduction.....	1
▪ <b>Chapter I : Basic Concepts of Deep Learning</b>	
I.1.Introduction.....	4
I.2. Machine Learning.....	4
I.2.1. Approaches to Machine Learning.....	4
I.3.Deep learning.....	4
I.3.1.Neural networks.....	5
I.3.2 Single perceptron.....	6
I.4 Activation function.....	7
I.5 Type of Neural Networks Activation Functions .....	7
I.5.1 Binary Step Function.....	7
I.5.2 Linear Activation Functions.....	7
I.5.3 Non-Linear Activation Functions .....	7
I.5.3.1 Sigmoid (Logistic Activation Function).....	7
I.5.3.2 Tanh Function (Hyperbolic Tangent).....	8
I.5.3.3 ReLU Function.....	8
I.5.3.4 Cost function.....	8
I.6 Gradient descent.....	9
I.7 Data and basic fact labels.....	10
I.8 Convolutional Neural Networks CNN.....	11
I.8.1 Convolutional Layer.....	12
I.8.2 The feature detector.....	12
I.8.3 pooling layer.....	12
1.8.4 Fully-Connected layer.....	13
Conclusion.....	14

- **ChapterII: Basic concepts of super resolution algorithms in deep learning configuration.**

II.1 introduction.....	16
II.2 Problem formulation of image super resolution process.....	16
II.3 Conventional SR algorithm (Bicubic Interpolation).....	17
II.4 SR algorithms based on deep learning networks.....	17
II.5 Super-Resolution Framework.....	18
II.6 Super resolution(SR) algorithms based on deep neural networks.....	20
II.6.1 EDSR algorithm:Enhanced deep super-resolution network.....	20
II.6.2 CAR Algorithm:Content adaptive resample.....	21
II.6.3 SRCNN:Super-resolution Convolution neural network.....	24
II.6.4 SRGAN: Super-resolution generative adversarial network.....	25
II.7 Performance assessment in terms of quantitative parametrs (PSNR,SSIM,FSIM and ISSM) and image quality(IQ).....	26
II.8 Structural similarity.....	26
II.9 Conclusion.....	27

- **ChapterIII: Implementation of image super resolution Algorithms based on Deep Learning**

III.1 Introduction.....	29
III.2 Used data sets.....	29
III.3 Materials.....	29
III.4 Steps of preparing test data sets.....	30
III.6 Implementation of SR algorithms.....	31
III.6.1 SRCNN.....	31
III.6.2 SRGAN.....	32
III.6.3 CAR.....	41
III.7 Concluding Remarues .....	46
General conclusion.....	47
Référence.....	48

## *Liste of Figures*

Figures	pages
<b>Figure I.1:</b> AI is the overall category that includes machine learning and neural networks	4
<b>Figure I.2:</b> Difference between the two types of learning	5
<b>Figure I.3:</b> The architecture of neural networks	6
<b>Figure I.4:</b> schéma du perceptron simple	6
<b>Figure I.5:</b> binary step function	7
<b>Figure I.6:</b> Logistic Activation Function	8
<b>Figure I.7:</b> Tanh function	8
<b>Figure I.8:</b> ReLU Function	8
<b>Figure I.9:</b> weight update by gradient descent in the cost function	10
<b>Figure I.10:</b> A training set	10
<b>Figure I. 11:</b> principle Architecture of CNN	11
<b>Figure I.12:</b> Convolutional layer	12
<b>Figure I.13:</b> Max pooling	13
<b>Figure II.1 :</b> Classification of image SR methods	16
<b>Figure II.2:</b> Super-resolution model frameworks based on deep learning	19
<b>Figure II.3:</b> Architecture of single-scale SR network (EDSR)	20
<b>Figure II.4:</b> Training Objectives (Objective Function)	21
<b>Figure II.5:</b> CAR SR algorithm architecture	21
<b>Figure II.6 :</b> SRCNN structure	25
<b>Figure II.7:</b> Architecture of generator and discriminator network Loss function	25
<b>Figure III.1:</b> Original High resolution images	29
<b>Figure III.2:</b> Generated LR image from HR image based on bicubic algorithm	30
<b>Figure III.3:</b> Workflow of SRCNN	31
<b>Figure III.4:</b> Show the workflow of SRGAN	33
<b>Figure III.5:</b> SRGAN results	40
<b>Figure III.6:</b> CAR SR results	45

## *Listes of tables*

<i>Listes of tables</i>	<i>pages</i>
<b>Table III.1:</b> Illustrates the different original Image	29
<b>Table III.2:</b> The average results of PSNR (dB), SSIM on the Live1 dataset	32
<b>Table III.3:</b> The average results of PSNR (dB), SSIM on the Live1 Set5 and Div2k dataset (SRGAN)	34
<b>Table III.4 :</b> The average results of PSNR (dB), SSIM on the Live1 Set5 and Div2k dataset (SR) CAR	41



## *Liste des abréviations*

<b><i>CNN</i></b>	<b><i>Convolutional Neural Networks</i></b>
<b><i>ANN</i></b>	<b><i>A neural network</i></b>
<b><i>SR</i></b>	<b><i>super-resolution</i></b>
<b><i>HR</i></b>	<b><i>high resolution</i></b>
<b><i>LR</i></b>	<b><i>low resolution</i></b>
<b><i>EDSR</i></b>	<b><i>Enhanced deep super-resolution network</i></b>
<b><i>CAR</i></b>	<b><i>Content adaptive resample</i></b>
<b><i>SRCNN</i></b>	<b><i>Super-Resolution Convolution neural network</i></b>
<b><i>SRGAN</i></b>	<b><i>Super-Resolution generative adversarial network</i></b>
<b><i>PSNR</i></b>	<b><i>Peak Signal-to-Noise Ratio</i></b>
<b><i>SSIM</i></b>	<b><i>Structural Similarity</i></b>

## ***General introduction***

The Super image resolution (SR) refers to the process of recovering high-resolution (HR) images from low-resolution (LR) images, which is an essential class of image processing techniques other than improving the perceptual quality of an image. This problem is complicated and bad in nature because there are many HR images that always correspond to one LR image.

With the rapid development of deep learning techniques in recent years, deep learning based SR models have been actively explored and applied in different area such as computed tomography (CT), biomedical images, synthetic aperture radar (SAR) images and Nondestructive control (NDC) (images [refs]). To solve SR tasks, a range of deep learning approaches have been developed, ranging from the early Convolutional Neural Networks (CNN) based method (e.g., SRCNN) to the more recent Deep Convolutional Neural Networks such as Content Adaptive resample, as we will see in our simulation study .

Generative Adversarial Nets (GAN) have been used in recent promising SR methods (e.g., SRGAN) .In general, the family of SR algorithms based on deep learning techniques differs in the following important aspects: different types of network architectures, loss functions, learning principles including the type of activation function and pooling layer.

### **Super resolution Problem formulation**

The deterioration of high images caused by downsampling and additional noise as shown from equation. The goal of SR algorithms is to recover a SR image from low-resolution one. In our work, we have considered the case of downsampling deterioration only

A focus on deep learning procedure will be detailed in our manuscript.

### **Organization of the manuscript**

. We provide a complete summary of current achievements in image super-resolution with deep learning that we are focused in deep learning based SR techniques

We give a comprehensive review of image super-resolution techniques based on deep learning, a family of SR methods with deep learning, domain-specific SR applications, etc.

Our work is divided into three parts:

The first chapter will devote to deep learning, a more details description on convolutional neural networks (CNNs).

The second chapter will describe the basics concept of image super-resolution. More precisely, at first present problem formulation and the conventional Image super-resolution algorithms

The third chapter will present our simulation with results. More precisely, we implement four algorithms of super-resolution based on deep learning, namely: Content Adaptive Resampler (CAR), SRCNN, and SRGAN algorithms. A performance assessment study is conducted by computing PSNR, SSIM metrics. We will show, through the obtained simulation results that Deep learning SR algorithms surpass the conventional SR algorithms. CAR algorithm has demonstrated a considerable superiority comparatively to the SRCNN and SRGAN algorithms, in terms of computing metrics and visual quality.

# *Chapter I*

## **Basic Concepts of Deep Learning**

### **Abstract:**

In this Chapter, we introduce the main concepts of machine learning and deep learning. More precisely, we define terminology in this field, then we describe linear regression algorithm.

---

### *Acknowledgement*

---

- I.1 Introduction**
- I.2 Machine Learning**
- I.3 Deep Learning**
- I.4 Activation Function**
- I.5 Type of Neural Networks Activation Functions**
- I.6 Gradient descent**
- I.7 Data and basic fact labels**
- I.8 Convolutional Neural Networks CNN**
- I.9 Conclusion**

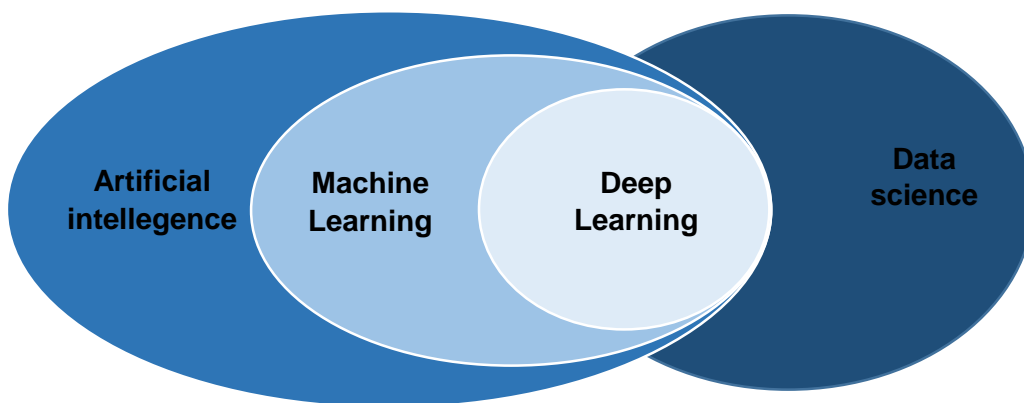
## I.1 Introduction

Deep learning indicates a set of learning technologies aims at designing data using complex structures that includes multiple non-linear transfers. Nerve networks, which are combined to form deep nerve networks, have enabled .these technologies to great progress in the areas of sound and picture processing, where there are several types of structures of nerve networks, which are the oldest and simplest. Convolutional Neural network (CNN) is a type of recurrent neural network that is used to handle sequential data such as text or time series [1].

The fundamental dimensions of learning by networks of neurons are presented in this chapter. First and foremost, we thought it would be interesting to draw a parallel between the biological and artificial neurons that form the foundation of his networks. Second, we use it to distinguish between different layers of a neuron network. In the third place, we'll show you what a function of activation, interest, and specific functions of activation in the fourth place, we we'll define what a loss function is (loss function). Before we wrap up this chapter, we'll go over what an optimizer is and the various sorts of optimizers that are used .

## I.2 Machine Learning

Machine Learning is the science of teaching computers to learn from data using a range of algorithms that iteratively learn from data to improve, explain, and predict outcome it is feasible to generate increasingly exact models based on training data as the algorithms absorb it. A machine learning model is the result of training your machine learning algorithm [2].



**Figure I.I:** AI is the overall category that includes machine learning and neural networks

### I.1.1 Approaches of Machine Learning

Machine learning techniques are required to improve the accuracy of predictive models. Depending on the nature of the business problem being addressed, [3] there are different approaches based on problem being addressed, there are different approaches based on the type and volume of the data. In this section, we discuss the categories of machine learning.

#### Supervised learning:

Supervised Learning is the most popular learning paradigm in Machine Learning and Deep Learning. As the name suggests, this involves supervising machine learning by showing it examples (data) of the task it needs to perform. [4] The applications are numerous: computer vision, regressions, classifications... The vast majority of Machine Learning and Deep Learning problems use supervised learning.

#### Unsupervised learning:

Unsupervised learning is best suited when the problem requires a massive amount of data that is unlabeled. For example, social media applications, such as **twitter**, **Snap chat**, and so on all have large amounts of unlabeled data.

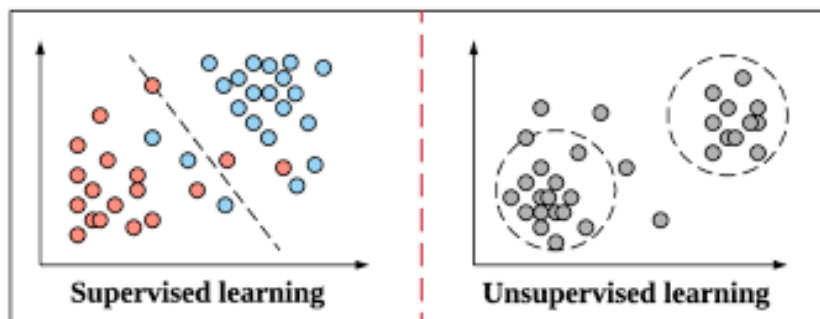


Figure I.2: Difference between the two types of learning [4]

### I.3 Deep Learning (DL)

Deep learning, is a sort of artificial intelligence [5] evolved from machine learning (automatic learning), in which the machine may learn on its own rather than following predetermined rules to the letter.

#### I.3.1 Neural Networks (NNs)

A neural network consists of three or more layers: an input layer one or many hidden layers, and an output layer. Data is ingested through the input layer. Then the data is modified

in the hidden layer and the output layers based on the weights applied to the se nodes. The typical neural network may consist of thousands or even millions of simple processing nodes that are densely interconnected [6].

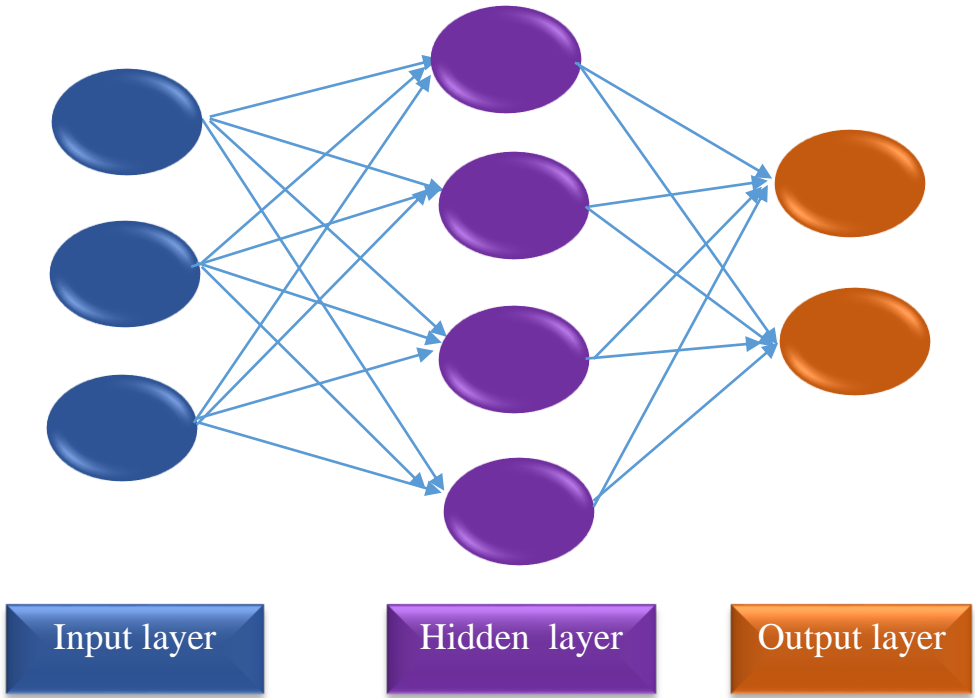


Figure I.3: The architecture of neural networks

**I.3.2 Single perceptron**

It is a binary neuron, that is to say whose output is 0 or 1. To calculate this output, the neuron performs a weighted sum of its inputs (each input has a weight ):

$$Y = (W_1X_1+W_2X_2+W_3X_3) \tag{1, 1}$$

Then apply a threshold activation function: if the sum weighted value exceeds a certain value the output of the neuron is 1, otherwise it is 0. It cannot therefore do only classification, and then prediction therefore does only classification, then prediction.

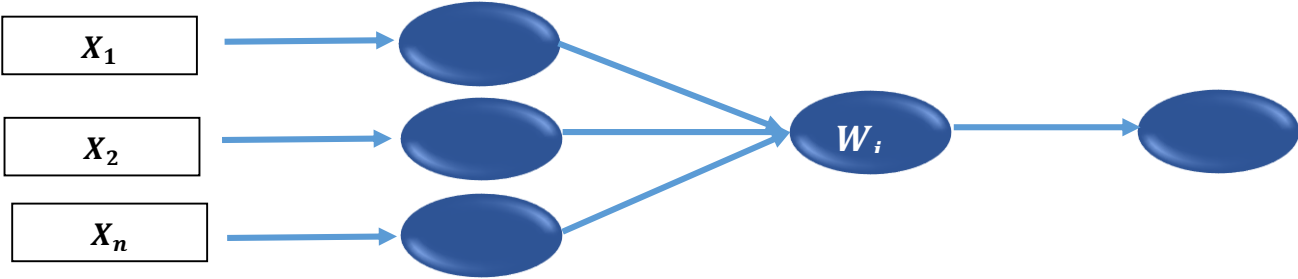


Figure I.4 : Schéma du perceptron simple

## I.4 Activation Function

To improve the model computation time, the activation function plays an important role as it takes any real number as input on the X axis and converts it to a specific output between 0 and 1[7]

## I.5 Type of Neural Networks Activation Functions

### I.5.1 Binary Step Function

The input fed to the activation function is compared to a certain threshold; if the input is greater than it, then the neuron is activated, else it is deactivated, meaning that its output is not passed on to the next hidden layer[8].



Figure I.5: Binary step function

### I.5.2 Linear Activation Function

The function does nothing with the input's weighted sum; it simply returns the value it was given.

$$F(x) = X \quad (1,2)$$

### I.5.3 Non-Linear Activation Functions

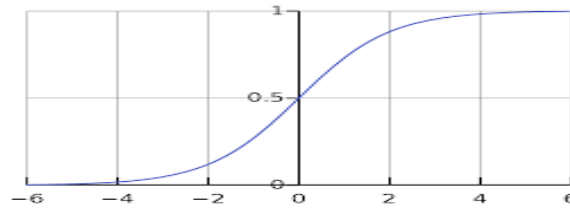
The following drawbacks of linear activation functions are overcome by non-linear activation functions:

#### I.5.3.1 Sigmoid (Logistic Activation Function)

This function accepts any real value as input and returns a value between 0 and 1. It's widely employed in models where the output is a probability prediction. Because anything's probability occurs only between 0 and 1, sigmoid is the best choice due to its range [8].

$$F(x) = \frac{1}{1+e^{-x}} \quad (1,3)$$



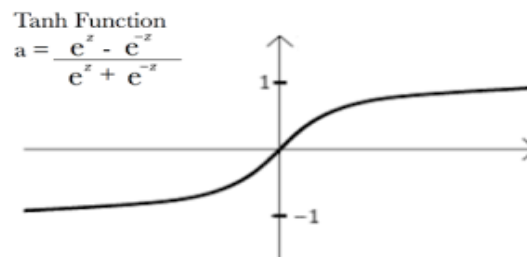


**Figure 1.6:** Logistic Activation Function

### I.5.3.2 Tanh Function (Hyperbolic Tangent)

**Tanh** function is very similar to the sigmoid/logistic activation function, and even has the same S-shape with the difference in output range of -1 to 1.

$$F(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \quad (1, 4)$$



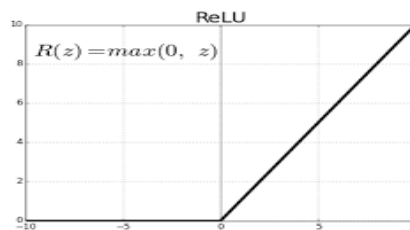
**Figure I.7:** Tanh function [8]

### I.5.3.3 ReLU Function

Although it gives an impression of a linear function, **ReLU** has a derivative function and allows for back propagation while simultaneously making it computationally efficient.

The main catch here is that the **ReLU** function does not activate all the neurons at the same time .

$$F(x) = \max(0, x) \quad (1, 5)$$



**Figure I.8:** ReLU Function [8]

### I.5.3.4 Cost function

We'll utilize supervised learning, or labeled datasets, to train the algorithm as we consider more practical use cases for neural networks, such as image recognition or classification. We'll use a cost (or loss) function to evaluate the model's accuracy as we train it. [9] To train the

logistic regression parameters  $\mathbf{w}$  and  $\mathbf{b}$ . We want to identify  $\mathbf{w}$  and  $\mathbf{b}$  such that the outputs you have ( $\hat{y}$ ) are close to the actual values at least on the training set ( $y$ )

$$L(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{2} (\hat{\mathbf{y}} - \mathbf{y})^2 \quad (1, 6)$$

This function has the drawback of making the optimization problem non-convex, resulting in several local optima. [20]As a result, we construct a novel loss function for logistic regression that solves the optimization problem by providing a convex function.

$$L(\mathbf{y}, \mathbf{y}) = -(\mathbf{y} \log \mathbf{y} + (1 - \mathbf{y}) \log (1 - \mathbf{y})) \quad (1, 7)$$

We want our cost function to be as small as possible. For that, we want our parameters  $\mathbf{w}$  and  $\mathbf{b}$  to be optimized .

Gradient descent a technique that helps to learn the parameters  $\mathbf{w}$  and  $\mathbf{b}$  in such a way that the cost function is minimized).

## 1.6 Gradient descent

A gradient simply measures the change in all weights with regard to the change in error. You can also think of a gradient as the slope of a function. The higher the gradient, the steeper the slope and the faster a model can learn. But if the slope is zero, the model stops learning. In

Here,  $\alpha$  is the learning rate that controls how mathematical terms, a gradient is a partial derivative with respect to its inputs [10]

The updated equation for gradient descent becomes :

$$\mathbf{W} = \mathbf{W}' - \alpha \frac{dJ(\alpha)}{d\mathbf{w}} \quad (1, 8)$$

$$\mathbf{b} = \mathbf{b}' - \alpha \frac{dJ(\mathbf{w}, \mathbf{b})}{d\mathbf{b}} \quad (1, 9)$$

Where:

$\alpha$  : learning rata

Big a step we should take ( after each iteration )

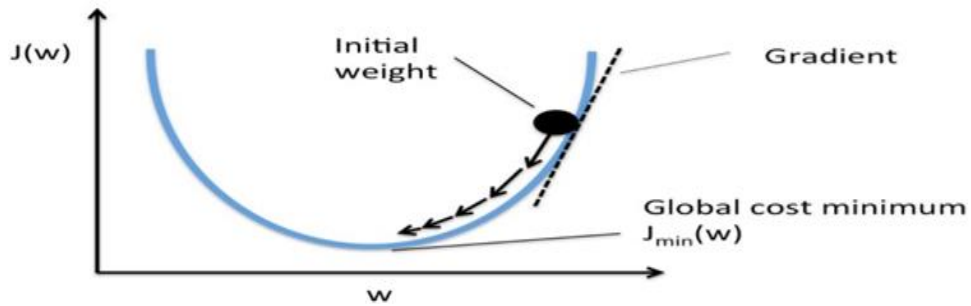


Figure I.9: Weight update by gradient descent in the cost function [11]

## I.7 Data and basic fact labels

Data labels and ground truth are two of the most important components of research that applies deep learning or other machine learning methods. As a well-known proverb that grew up in computer science notes: Accurate collection of data and basic fact markers with which a model is trained and tested is imperative for a successful deep learning project, but obtaining high-quality data can be costly and time-consuming[12].

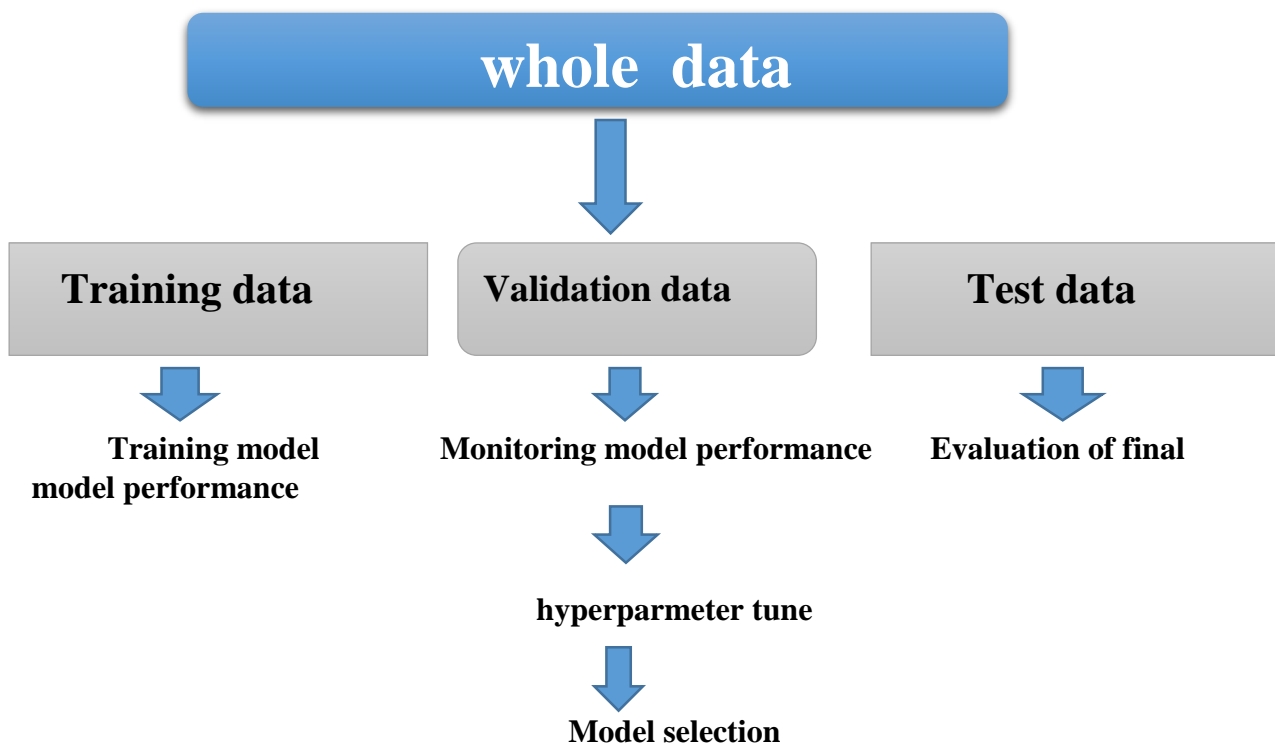


Figure I.10: A training set

Available data are usually separated into three groups: training, validation, and test set (Fig. I.10). A training set is used to train a network, with forward propagation used to compute loss values and reverse propagation used to update learnable parameters.

The validation set is used to assess the model during training, fine-tune hyperparameters, and pick the best model. The test set should be used only once at the end of the project to assess the performance of the final model that was defined and chosen during the training phase with the training and validation sets.

### I.8 Convolutional Neural Networks CNN

The higher performance of convolutional neural networks with picture, speech, or audio signal inputs sets them apart from conventional neural networks.

They have There different types of layer:

- ✓ Layer of convolution
- ✓ Layer of POOLING
- ✓ FC (FULLY CONNECTED) Layer

A convolutional network's first layer is the convolutional layer. While further convolutional layers or pooling layers can be added after convolutional layers, the fully-connected layer is the last layer. The CNN becomes more complicated with each layer, detecting larger areas of the image. Earlier (colors and edges) as the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object layers concentrate on basic elements.

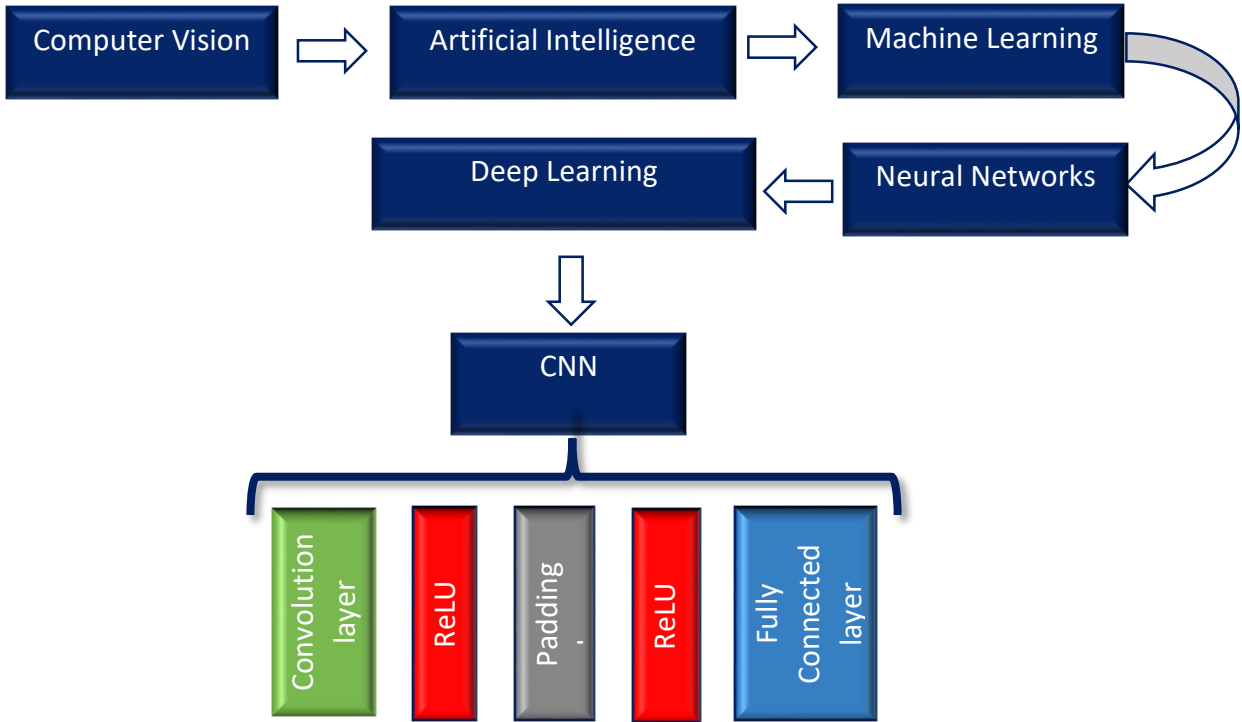


Figure I. 11: Principal Architecture of CNN

## I.8.1 Convolutional Layer

The input will be a color image, which is made up of a matrix of pixels in 3D. This means that the input will have three dimensions a height, width, and depth which correspond to RGB in an image. We also have a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution. [12]

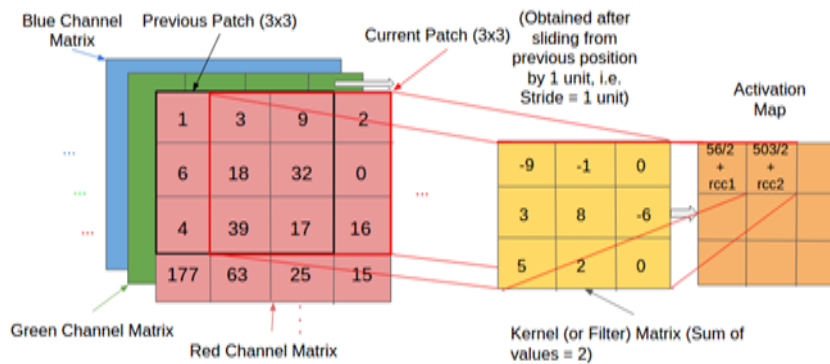


Figure I.12: Convolutional layer [12]

## I.8.2 Feature detector

Is a two-dimensional (2-D) array of weights, which represents part of the image. While they can vary in size, the filter size is typically a 3x3 matrix; this also determines the size of the receptive field.

- ✓ **Valid padding:** This is also known as no padding. In this case, the last convolution is dropped if dimensions do not align
- ✓ **Same padding:** This padding ensures that the output layer has the same size as the input layer
- ✓ **Full padding:** This type of padding increases the size of the output by adding zeros to the border of the input

A CNN adds a Rectified Linear Unit (ReLU) adjustment to the feature map after each convolution operation, introducing nonlinearity to the model.

## I.8.3 Pooling Layer

Down sampling, also known as pooling layers, is a dimensionality reduction technique that reduces the number of factors in the input the pooling process sweeps a filter across the

entire input, similar to the convolutional layer, however this filter does not have any weights. Instead, the kernel uses an aggregation function to populate the output array from the values in the receptive field. POOLING Can be divided into two categories :

**Max pooling:** As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.

**Average pooling:** As the filter moves across the input, it calculates the average value within the receptive field to send to the output array [13]

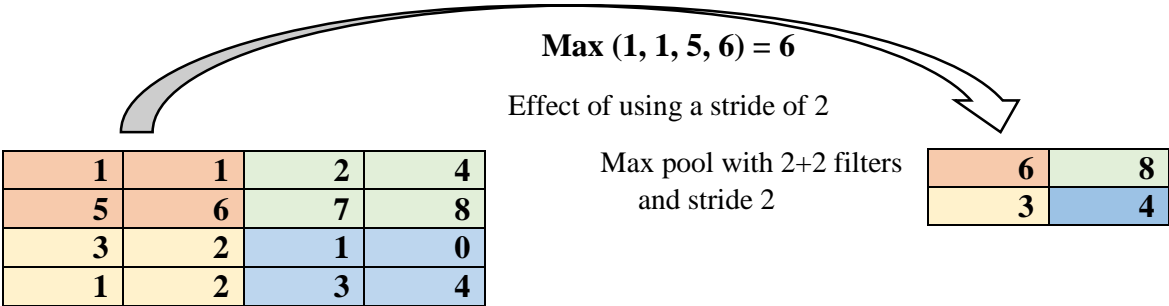


Figure I.13: Max pooling

### I.8.4 Fully-Connected Layer

The full-connected layer's name is self-explanatory. In partially linked layers, the pixel values of the input image are not directly connected to the output layer, as previously stated. Each node in the output layer, on the other hand, connects directly to a node in the previous layer in the fully-connected layer. This layer performs classification tasks based on the features retrieved by the previous layers and their various filters. While convolutional and pooling layers typically utilize ReLu functions to categorize inputs, FC layers typically use a soft max activation function to produce a probability from 0 to 1.

#### Choice of hyper parameters

CNN uses more hyperparameters than a standard MLP. Even though the rules usual for learning rates and regularization constants apply always, it is necessary to take into account the notions of number of filters, their form and the form of max pooling

## **I.9 Conclusion**

In this chapter we have presented the important notions that are related to deep learning (definition, Architectures... etc.). Also that a general vision on deep learning, all giving in detail the method chosen in our work of research who is the CNNs. The next chapter deals with the details of the design, as well as the method and the tools used for the realization of our application.

# *Chapter II*

## **Basic Concepts of Super resolution Algorithms in Deep Learning Configuration**

### **Abstract**

This Chapter deals with SR image process. First of all, we will introduce the main concepts of SR process. Then we will describe three SR algorithms based deep learning. Two metrics will be defined for performance assessment

---

### ***Acknowledgement***

---

#### **II.1 Introduction**

#### **II.2 Problem formulation of image super resolution process**

#### **II.3 Conventional SR algorithm (Bicubic Interpolation)**

#### **II.4 SR algorithms based on Deep Learning Networks**

#### **II.5 Super-Resolution Framework**

#### **II.6 Super resolution (SR) algorithms based on deep neural networks**

#### **II.7 Performance assessment in terms of quantitative parameters (PSNR, SSIM, FSIM and ISSM) and image quality. (IQ)**

#### **II.8 Structural similarity SSIM**

#### **II.9 Conclusion**



## II.1 Introduction

Image super-resolution (SR) is an important task in image processing methods that improve the resolution of input images. In the last two decades [14], significant progress has been made in the field of super-resolution (SR), especially by utilizing deep learning methods. This chapter describes the basic concepts of image super-resolution. More precisely, we first present problem formulation and the conventional Image super resolution algorithms. A focus on Bicubic SR algorithm will be discussed. The SR conventional algorithms will be classified into three categories [15], i.e., conventional algorithms, supervised learning-based algorithms and unsupervised learning based algorithms. SR algorithms based on deep learning will be presented in this Chapter ; namely, EDSR, CAR, SRCNN and SRGAN algorithms. [16]

A theoretical comparative study will be established to highlight the advantages of deep learning networks in SR problem. Two image quality metrics will be considered and computed in our simulation study, namely Peak signal to noise ratio (PSNR) and Structural similarity (SSIM).

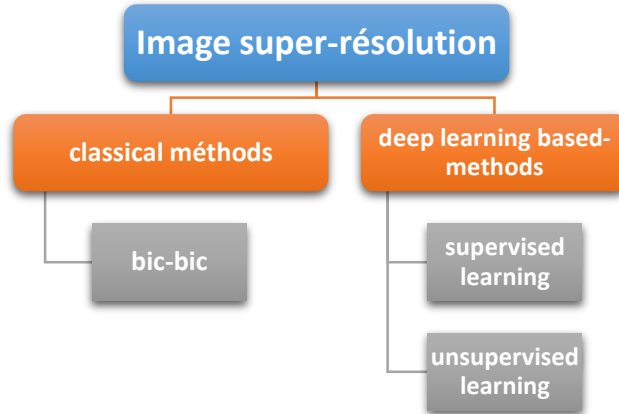


Figure II.1: Classification of image SR methods

## II.2 Problem formulation of image super resolution process [17]:

The image SR focuses on the recovery or the reconstruction of a high resolution (HR) image from a low resolution (LR) image input , The LR image denoted by  $I_{xLR}$  can be represented as the output of the degradation function, as shown in equ (2.1) :

$$I_{xLR} = d(I_{yHR}, \delta) \quad (2.1)$$

Where  $d$  is the SR degradation function that is responsible for the conversion of HR image to LR image,  $I_{yHR}$  is the input HR image (reference image), whereas  $\delta$  depicts the input parameters of the image degradation function. Degradation parameters are usually scaling factor, blur or noise. In practice, the degradation process and dependent parameters are unknown, and only LR images are used to obtain HR images through SR algorithms. The SR process is responsible for predicting the inverse of the degradation function  $d$ , such that :

$$\mathbf{g}(I_{xLR}, \delta) = d^{-1}(I_{xLR}) = I_{yE} \approx I_{yLR} \quad (2.2)$$

Where  $\mathbf{g}$  is the SR function,  $d$  depicts the input parameters to the function  $\mathbf{g}$ , and  $I_{yE}$  is the estimated HR corresponding to the input  $I_{xLR}$  image. The degradation process for the input LR images is unknown, and this process is affected by numerous factors such as sensor-induced noise, artifacts created because of lossy compression, speckle noise, motion blur, and misfocused images. In the literature [18], most of the studies have used a single downsampling function as the image degradation function:

$$d(I_{yHR}, \theta) = (I_{yHR}) \downarrow s_f, \{s\} \subseteq \theta \quad (2.3)$$

Where  $\downarrow s_f$  is a downsampling operation with the scaling factor  $s$ . In our simulation experiments only a downsampling operation degradation is discussed. We will fix  $s$  to 2 or 4 because the used datasets in this study are built based on this pattern and these two values. The most commonly used downsampling operation is bicubic interpolation as it will be considered in Chapter 3. The case of noisy images will not be discussed in our study.

### II.3 Conventional SR algorithm (Bicubic Interpolation) :

Methods based on interpolation - Image interpolation (image scaling) refers to the scaling of digital images and is widely used in image-related applications. The traditional methods include nearest neighbor interpolation, linear interpolation, linear interpolation, cubic interpolation, etc.

Bicubic Interpolation - Similarly, cubic interpolation (BCI) performs cubic interpolation on both axes compared to BLI, BCI takes 4 x 4 pixels into account, and results in smoother results with fewer artifacts but a much lower speed. Refer to this for a detailed discussion[18]

### II.4 SR algorithms based on Deep Learning Networks :

The process of deep learning is to recover a super resolution image for which a loss function is minimized. It's an optimization problem.

in the SR network training. To do fair comparisons with the EDSR, we only adopt the L1 norm loss as the restoration metric as suggested by. The L1 norm loss defined for SR is:

$$\mathcal{L}(\hat{\mathbf{I}}) = \frac{1}{N} \sum_{p \in I} |\mathbf{p} - \hat{\mathbf{p}}| \quad (2.4)$$

Where  $\hat{\mathbf{I}}$  is the SR result,  $\hat{\mathbf{P}}$  and  $\mathbf{P}$  represent the ground-truth and reconstructed pixel value, N indicates the number of pixels times the number of color channels.

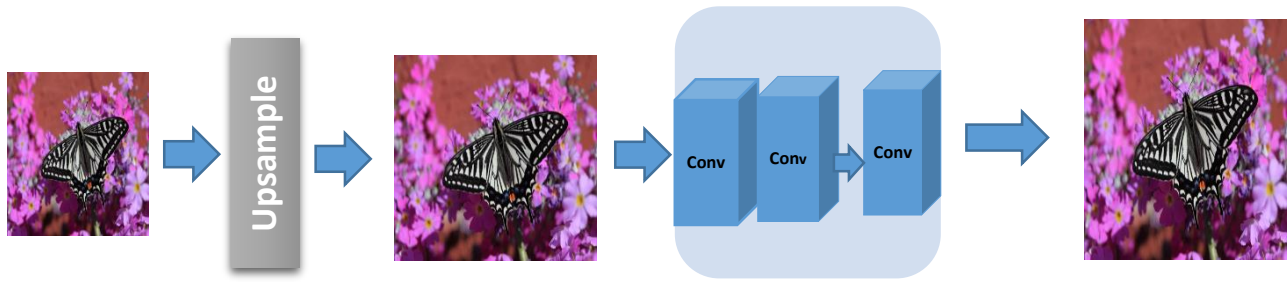
As we have mentioned, two categories of data sets will be considered: X2 and X4.

**X2:** Data Set Scale 2(X2): If the test image is  $M \times N$ , X2 will be  $M/2 \times N/2$

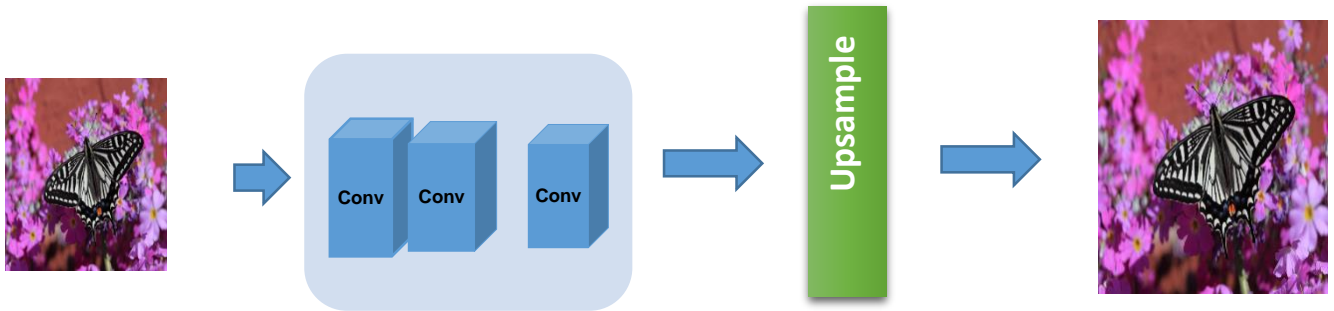
**X4:** Data Set Scale 4(X4): If the test image in  $M \times N$ , X4 will be  $M/4 \times N/4$

## II.5 Super-Resolution Framework :

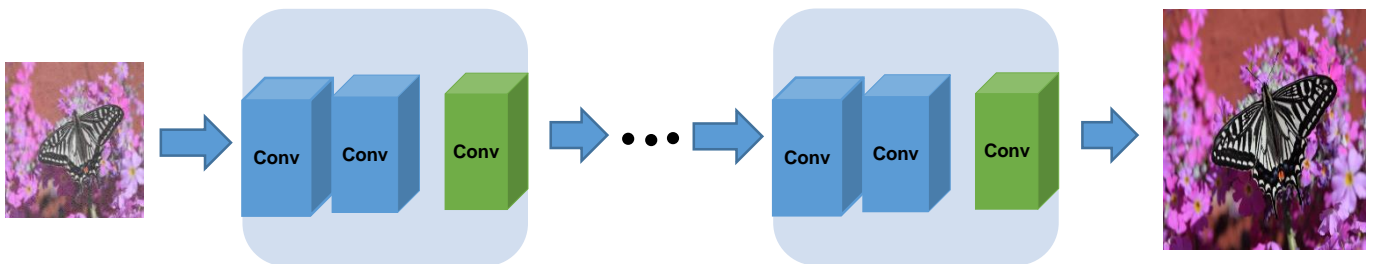
Figure II.3 represents the different SR framework[19] The cube size represents the output size. The gray ones denote predefined upsampling ,while the green, yellow and blue ones indicate learnable upsampling, downsampling and convolution layers, respectively..In our work, we will use Iterative SR framework as we will describe in ChapterIII.



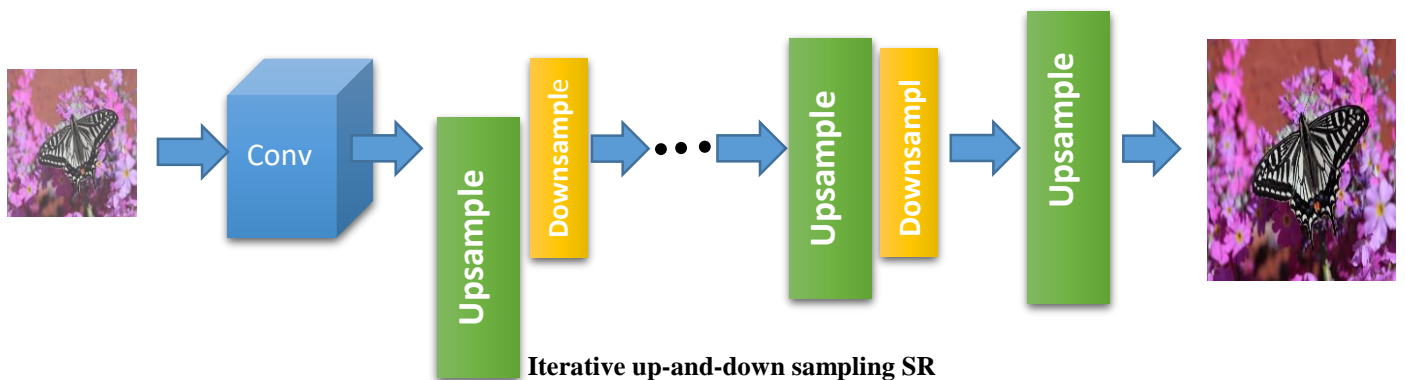
**Pre-Upsampling SR**



**Post-upsampling SR**



**Progressive Upsampling SR**



**Iterative up-and-down sampling SR**

**Figure II.2:** Super-resolution model frameworks based on deep learning

## II.6 Super resolution (SR) algorithms based on deep neural networks

### II.6.1 EDSR algorithm : Enhanced deep super-resolution network

EDSR algorithm is a deep learning SR algorithm network [20]. The significant performance improvement of this network is due to optimization by removing unnecessary modules in conventional residual networks. The performance is further improved by expanding the model size while we stabilize the training procedure.

EDSR algorithm is represented in Figure II.4

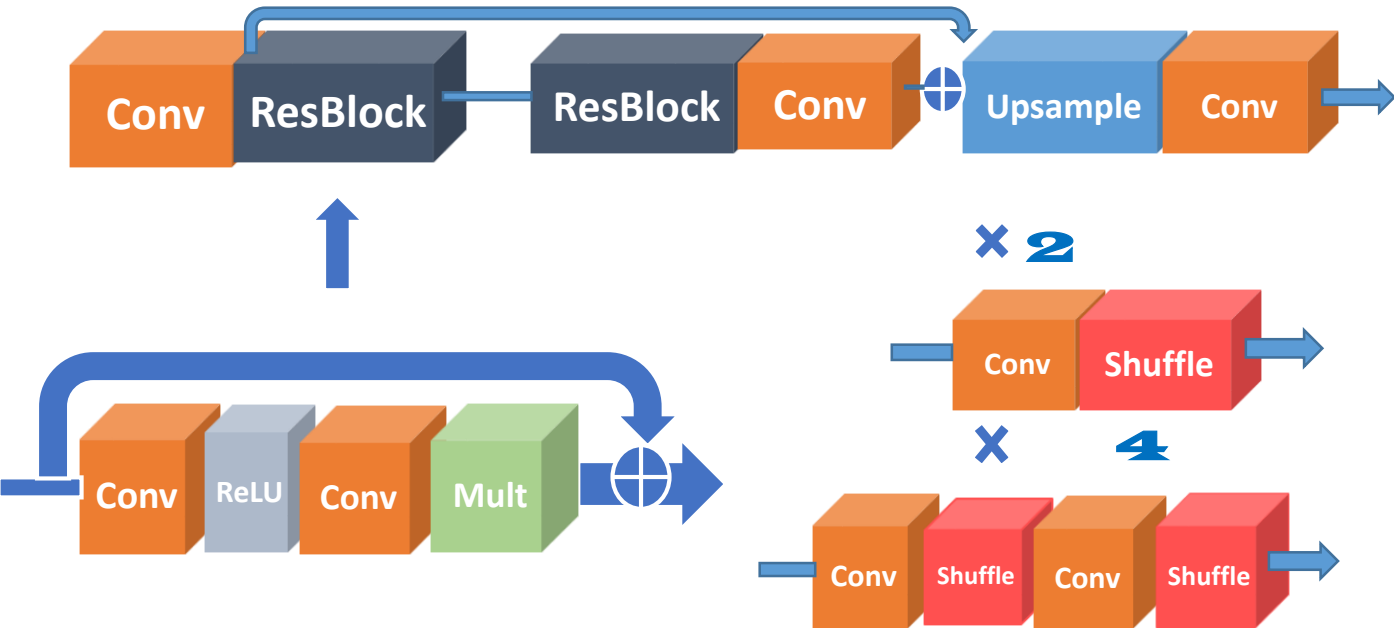
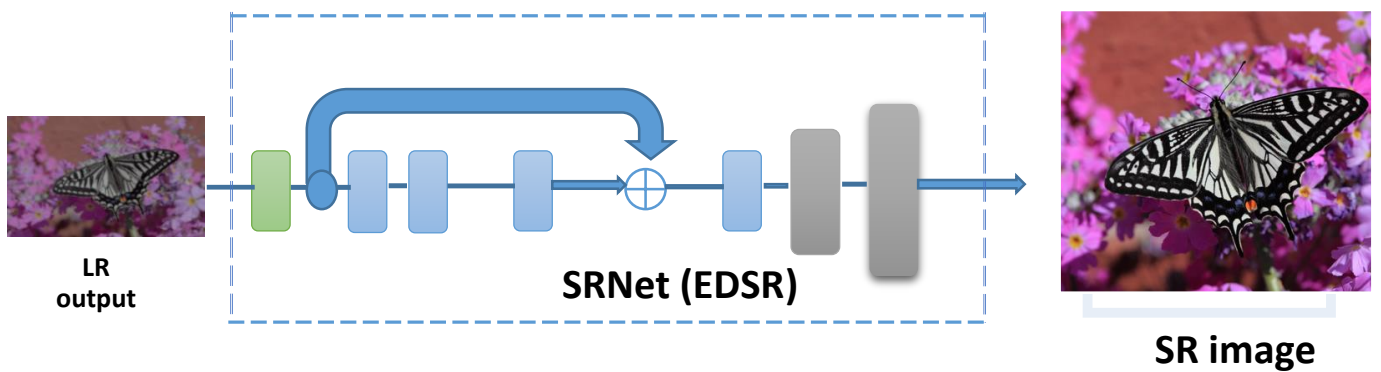


Figure II.3 : Architecture of single-scale SR network (EDSR)

It consists of three parts, the ResamplerNet, the Downscaling module, and the SRNet. The ResamplerNet is designed to estimate content adaptive resampling kernels and its corresponding offsets for each pixel in the downsampled image. The SRNet, can be any form of differentiable upsampling operations, is employed to guide the training of the ResamplerNet by simply minimizing the SR error. The entire framework is trained end-to-end by back-propagating error signals through a the differentiable downscaling module. The composition of each building block is detailed on the blue dashed frame.

The SR image is generated from Low resolution (LR) image as it is shown in **Figure.II.5**



**Figure II.4:** Training Objectives (Objective Function)

In the SR network training. To do fair comparisons with the EDSR , we only adopt the L1 norm loss as the restoeation metric [21]

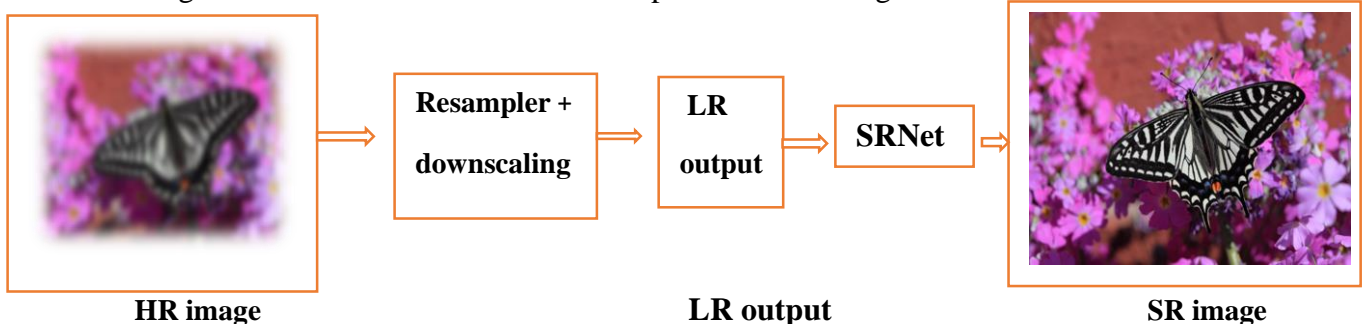
### II.6.2 CAR Algorithm : Content Adaptive Resample

CAR employs state-of-the-art SR model EDSR as the image upscaling module to guide the training of the proposed CAR model.

In this methods ,we propose a learned image downscaling method based on content adaptive resampler (CAR)[22]with consideration on the upscaling process. The proposed resampler network generates content adaptive image resampling kernels that are applied to the original HR input to generate pixels on the downscaled image.

Moreover, a differentiable upscaling (SR) module is employed to upscale the LR result into its underlying HR counterpart. By back-propagating the reconstruction error down to the original HR input across the entire framework to adjust model parameters, the framework achieves a new state-of-the-art SR performance through upscaling guided image resamplers which adaptively preserve detailed information that is essential to the upscaling.

Figure II.6 summarizes the main concepts of CAR SR algorithm



**Figure II.5:** CAR SR Algorithm architecture

The gradient descent algorithm adjusts the parameters of the resampler generation network to produce better resampling kernels which make the downsampled image can be super-resolved more easily. The ResamplerNet consists of downscaling blocks, residual blocks and upscaling blocks.

The ResamplerNet consists of downscaling blocks, residual blocks plus upscaling blocks.

The role of downscaling blocks with the residual blocks is to get the feature maps upscaling blocks consists of interpolation step based on bicubic algorithm.

HR image as a set of feature maps. Then ,two upscaling blocks are used to learn the content adaptive resampling kernel weights  $\mathbf{K} \in \mathbf{R}^{\left(\frac{h}{s}\right) \times \left(\frac{w}{s}\right) \times m \times n}$  ,offsets in the horizontal direction  $\Delta Y \in \mathbf{R}^{\left(\frac{h}{s}\right) \times \left(\frac{w}{s}\right) \times m \times n}$  and offsets in the vertical direction  $\Delta X \in \mathbf{R}^{\left(\frac{h}{s}\right) \times \left(\frac{w}{s}\right) \times m \times n}$  , respectively. h and w are the height and width of the input HR image,S is the downscaling factor,and m,n represent the size of the content adaptive resampling kernel. Each kernel is normalized so that elements of kernel are summed up to be 1.

## Forward pass

Firstly, we need to position each resampling kernel, associated with the pixel of the downsampled image, on the HR image .it can be achieved using the projection operation defined as :

$$(\mathbf{u}, \mathbf{v}) = (\mathbf{x} + \mathbf{0.5} \cdot \mathbf{Y} + \mathbf{0.5}) \times \mathbf{scale} - \mathbf{0.5} \quad (2.5)$$

Where  $(\mathbf{x}, \mathbf{y})$  is the indices of a pixel at the x-th row and y-th column in the downsampled image,scale represents the downscaling factor, and the resulting  $(\mathbf{u}, \mathbf{v})$  is center of downsampled pixel  $\mathbf{P}_{\mathbf{x}, \mathbf{y}}(\text{LR})$  projected on the input HR image.

Equation (2.5) assumes that pixels have a nonzero size, and the distance between two neighboring samples is one pixel.

Then, each pixel in the downsampled image is created by local filtering on pixel in the input HR image with the corresponding content adaptive resampling kernel as follows :

$$\mathbf{p}_{\mathbf{x}, \mathbf{y}}^{\text{LR}} = \sum_{\mathbf{l}=0}^{\mathbf{m}-1} \sum_{\mathbf{j}=0}^{\mathbf{n}-1} \left( \mathbf{K}_{\mathbf{x}, \mathbf{y}}(\mathbf{i}, \mathbf{j}) \cdot \mathbf{S}^{\text{HR}}(\mathbf{u} + \mathbf{i} - \frac{\mathbf{m}}{2} + \Delta \mathbf{X}_{\mathbf{x}, \mathbf{y}}(\mathbf{i}, \mathbf{j}), \mathbf{v} + \mathbf{j} - \frac{\mathbf{n}}{2} + \Delta \mathbf{Y}_{\mathbf{x}, \mathbf{y}}(\mathbf{i}, \mathbf{j})) \right) \quad (2.6)$$

Where  $\mathbf{K}_{x,y} \in \mathbf{R}^{m \times n}$  is the resampling kernel associated with the downsampled pixel at location  $(\mathbf{x}, \mathbf{y})$ ,  $\Delta \mathbf{X}_{x,y} \in \mathbf{R}^{m \times n}$  and  $\Delta \mathbf{Y}_{x,y} \in \mathbf{R}^{m \times n}$  are the spatial offset for each element in the  $\mathbf{K}_{x,y}$ . The  $S^{HR}$  is the sample point value of the input HR image. Due to the location projection and fractional offsets,  $\mathbf{S}^{HR}$  can refer to non-lattice point on the HR image, therefore,  $\mathbf{S}^{HR}$  is computed by bilinear interpolating the nearby four pixels around the fractional position :

$$\begin{aligned} S_{u',v'}^{HR} = & (1 - \alpha)(1 - \beta) \cdot P_{[u'],[v']}^{HR} + \alpha(1 - \beta) \cdot P_{[u'],[v']+1}^{HR} + (1 - \alpha)\beta \cdot P_{[u'],[v']}^{HR} + \\ & \alpha\beta \cdot P_{[u'],[v']+1}^{HR} \end{aligned} \quad (2.7)$$

Where  $u'$  and  $v'$  are fractional position on the HR image,  $\alpha = u' - [u']$  and  $\beta = v' - [v']$  are the bilinear interpolation weights.

## Backward pass

From the SR Net through the resampling operation. The partial derivative of the downsampled pixel with respect to the resampling kernel weight can be formulated as :

$$\frac{\partial p_{x,y}^{LR}}{\partial K_{x,y}(i,j)} = S^{HR}(\mathbf{u} + \mathbf{i} - \frac{\mathbf{m}}{2} + \Delta \mathbf{X}_{x,y}(i,j), \mathbf{v} + \mathbf{j} - \frac{\mathbf{n}}{2} + \Delta \mathbf{X}_{x,y}(i,j)) \quad (2.8)$$

The partial derivative of downsampled pixel with respect to the element in the resampling kernel is simply the interpolated pixel value. Equation (2.8) is derived with a single channel image and can be generalized to the color image by summing up values calculated separately on the R ,G and B channels.

The partial derivative of downsampled pixel with respect to the kernel element offser is computed as:

$$\begin{aligned} \frac{\partial p_{x,y}^{LR}}{\partial \Delta X_{x,y}(i,j)} &= \frac{\partial p_{x,y}^{LR}}{\partial S_{u',v'}^{HR}} \cdot \frac{\partial S_{u',v'}^{HR}}{\partial \Delta X_{x,y}(i,j)} \\ \frac{\partial p_{x,y}^{LR}}{\partial \Delta Y_{x,y}(i,j)} &= \frac{\partial p_{x,y}^{LR}}{\partial S_{u',v'}^{HR}} \cdot \frac{\partial S_{u',v'}^{HR}}{\partial \Delta Y_{x,y}(i,j)} \end{aligned} \quad (2.9)$$

Because we employ bilinear interpolation to compute  $S_{u',v'}^{HR}$ , therefore, the partial derivative of downsampled pixel with respect to the kernel offset is defined as:

$$\begin{aligned} \frac{\partial p_{x,y}^{LR}}{\partial \Delta X_{x,y}(i,j)} &= K_{x,y}(i,j) \cdot (1 - \beta) \cdot \left( P_{[u'],[v']+1}^{HR} p_{[u'],[v']}^{HR} \right) + \beta \cdot \left( p_{[u']+1,[v']}^{HR} - \right. \\ & \left. p_{[u']+1,[v']}^{HR} \right) \end{aligned} \quad (2.10)$$



$$\frac{\partial p_{x,y}^{LR}}{\partial \Delta Y_{x,y}(i,j)} = K_{x,y}(i,j) \cdot (1 - \alpha) \cdot (P_{[u'],[v'] + 1}^{HR} p_{[u'],[v']}^{HR}) + \alpha \cdot (p_{[u'] + 1, [v']}^{HR} - p_{[u'] + 1, [v']}^{HR}) \quad (2.11)$$

Also because equation ( is defined with a single color image , we need to sum up partial derivative of each color component with respect to the offset to obtain the final partial derivative of downscaled pixel with respect to the kernel element offset.

The pixel value of the downscaled image created using equation (2.12) is inherently continuous floatin point number. However , common image representation describes the color using integer number ranging from 0 to 255 . thus, a quantization step is required . since simply rounding the floating point number to ... its nearest integer number is not differentiable,we utilize the soft round method [22] .to derive the gradient un the back-propagation during the training phase.

The soft round function is defined as :

$$\mathbf{round}_{\text{soft}}(x) = x - \alpha \frac{\sin 2\pi x}{2\pi} \quad (2.12)$$

Where  $\alpha$  is a tuning parameter used to adjust the gradient around the integer position . Note that in the forward propagation ,the non-differentiable round function is used to get the nearest interger value.

### **Image upscaling :**

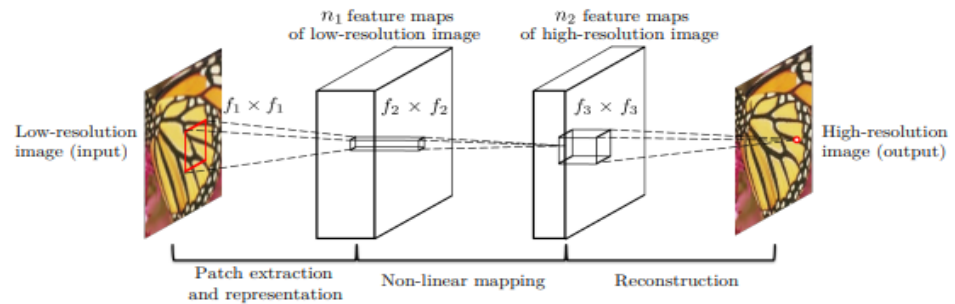
The proposed CAR model is trained using back-propagation to maximize the SR performance. The image upscaling module can be of any form of SR networks,even the differentiable bilinear or bicubic upscaling operation.

### **II.6.3 SRCNN : Super-Resolution Convolution Neural Network**

The CNNs have been developed rapidly in the last two decades. However, its application to solve SISR problem is first introduced, who described a three-layer CNN for super resolution as super-Resolution Convolution neural network (SRCNN).

In this method,CNN has been used to learn the non-linear mapping between the LR and HR images and it significantly outperforms previous non-deep learning methods. The training objective is to find optimal model ,given training set  $\{x^i, y^i\}_{i=1 \dots N}$  .the best mode F then will use to accurately prediets value  $Y=f(x)$ , where x is unobserved examples , the SRCNN consists of following operation, as show in figure [23]

- 1) **Preprocessing:** Upscale LR image to desired HR image using bicubic interpolation.
- 2) **Feature extraction:** extracts a set of feature map from the upscale LR image.
- 3) **Non –linear mapping:** maps the feature maps between LR and HR patch.
- 4) **Reconstruction:** produce the HR image from HR patches.



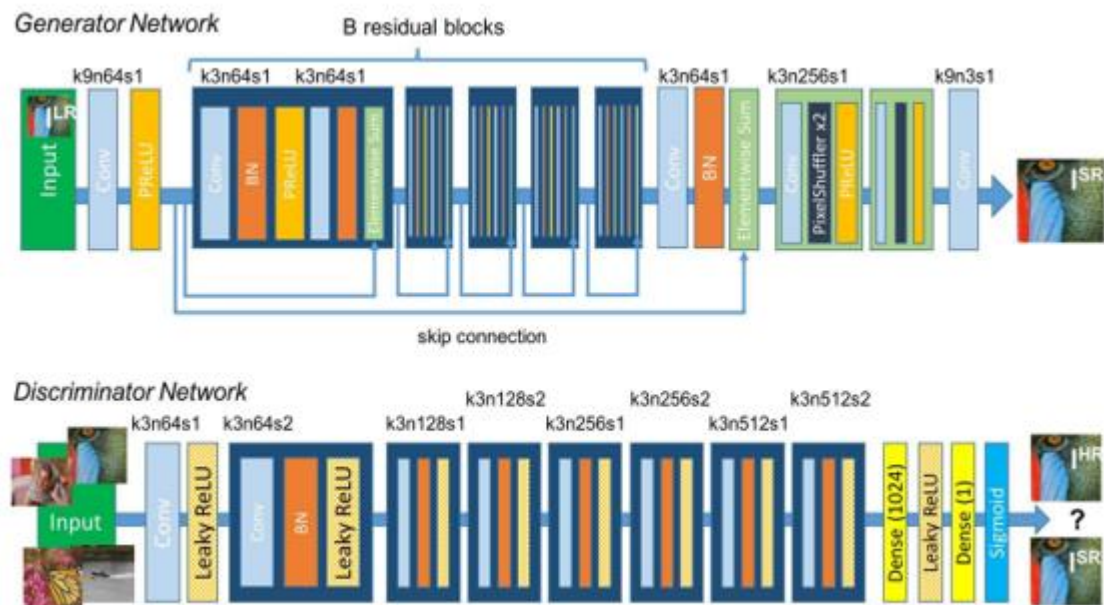
**Figure II.6 :** SRCNN structure[11]

The zero padding also is introduced to avoid the size of feature map reduces quickly through layers of convolution, which appears in deep networks.

## II.6.4 SRGAN : Super-Resolution generative adversarial network

### Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network

The SRGAN SR architecture is summarized in FigureII.8 :



**Figure II.7:** Architecture of generator and discriminator network Loss function

In terms of super-resolution, it is straightforward to adopt adversarial learning, in which case we only need to treat the SR model as a generator, and define an extra discriminator to judge whether the input image is generated or not. Therefore, SRGAN using adversarial loss based on cross entropy, as follows:

$$\mathcal{L}_{\text{gan\_ce\_g}}(\hat{I}; D) = -\log D(\hat{I}) \quad (2.14)$$

$$\mathcal{L}_{\text{gan\_ce\_d}}(\hat{I}; I_s; D) = -\log D(I_s) - \log(1 - D(\hat{I})), \quad (2.15)$$

Where  $\mathcal{L}_{\text{gan\_ce\_g}}$  and  $\mathcal{L}_{\text{gan\_ce\_d}}$  denote the adversarial loss of the generator and the discriminator  $D$  (i.e., a binary classifier) respectively, and  $I_s$  represents images randomly sampled from the ground truths. Besides, the EnhancNet also adopts the similar adversarial loss.

## II.7 Performance assessment in terms of quantitative parameters (PSNR, SSIM, FSIM and ISSM) and image quality (IQ)

PSNR are defined as follows:

$$\text{PSNR} = 10 \log_{10} \cdot \left( \frac{L^2}{\frac{1}{2} \sum_{i=1}^N (I(i) - \hat{I}(i))^2} \right), \quad (2.16)$$

Where  $L$  equals to 255 in general cases using 8-bit representation. Since the PSNR is only related to the pixel-level MSE, only caring about the differences between corresponding pixels instead of visual perception, it often leads to poor performance in representing the reconstruction quality in real scenes, where we're usually more concerned with human perceptions. However, due to the necessity to compare with literature works and the lack of completely accurate perceptual metrics, PSNR is still currently the most widely used evaluation criteria for SR models.

## II.8 Structural similarity SSIM

Considering that the human visual system (HVS) is highly adapted to extract image structures, the structural similarity index (SSIM) is proposed for measuring the structural similarity between images, based on independent comparisons in terms of luminance, contrast, and structures. For an image  $I$  with  $N$  pixels, the luminance  $\mu_I$  and contrast  $\sigma_I$  are estimated as the mean and standard deviation of the image intensity, respectively, i.e.,

$\mu_I = \frac{1}{N} \sum_{i=1}^N I(i)$  and  $\sigma_I = \left( \frac{1}{N-1} \sum_{i=1}^N (I(i) - \mu_I)^2 \right)^{\frac{1}{2}}$  where  $I(i)$  represents the intensity of the  $i$ -th pixel of image  $I$ . and the comparisons on luminance and contrast, denoted as  $C_l(I, \hat{I})$  and  $C_c(I, \hat{I})$  respectively, are given by :

$$C_l(I, \hat{I}) = \frac{2\mu_I\mu_{\hat{I}} + C_1}{\mu_I^2 + \mu_{\hat{I}}^2 + C_1} \quad (2.17)$$

$$C_c(I, \hat{I}) = \frac{2\sigma_I\sigma_{\hat{I}} + C_2}{\sigma_I^2 + \sigma_{\hat{I}}^2 + C_2} \quad (2.18)$$

where  $C_1 = (K_1L)^2$  and  $C_2 = (K_2L)^2$  are constants for avoiding instability,  $K_1 \ll 1$  and  $K_2 \ll 1$ . Besides, the image structure is represented by the normalized pixel values

(i.e.,  $(I - \mu_I) / \sigma_I$ ), whose correlations (i.e., inner product) measure the structural similarity, equivalent to the correlation coefficient between  $I$  and  $\hat{I}$ . Thus the structure comparison function  $C_s(I, \hat{I})$  is defined as :

$$\sigma_{I\hat{I}} = \frac{1}{N-1} \sum_{i=1}^N (I(i) - \mu_I)(\hat{I}(i) - \mu_{\hat{I}}) \quad (2.19)$$

$$C_s(I, \hat{I}) = \frac{2\sigma_{I\hat{I}} + C_3}{\sigma_I\sigma_{\hat{I}} + C_3} \quad (2.20)$$

Where  $\sigma_{I\hat{I}}$  is the covariance between  $I$  and  $\hat{I}$ , and  $C_3$  is a constant for stability.

Finally, the SSIM is given by :

$$\text{SSIM}(I, \hat{I}) = [C_l(I, \hat{I})]^\alpha [C_c(I, \hat{I})]^\beta [C_s(I, \hat{I})]^\gamma, \quad (2.21)$$

Where  $\alpha, \beta, \gamma$  are control parameters for adjusting the relative importance ; since the SSIM evaluates the reconstruction quality from the perspective of the HVS, it better meets the requirements of perceptual assessment, and is also widely used.

## II.9 Conclusion

A bicubic SR algorithm and four SR deep learning algorithms are presented in this Chapter. We mainly discussed the improvement of supervised SR : EDSR, CAR, SRCNN and SRGAN. In Chapter 3, we will present the results of the implementation of CAR an.

# *Chapter III*

## *Implementation of image super resolution Algorithms based on Deep Learning*

### **Abstract:**

This chapter is dedicated to present and discuss our simulation results. We will implement the three SR algorithms based on deep learning: SRGAN, SRCNN and CAR algorithms. A quantitative comparative study is conducted in terms of PSNR and SSIM.

---

### *Acknowledgement*

---

- III.1 Introduction**
- III.2 Used data sets**
- III.3 Original Image**
- III.4 Materials**
- III.5 Step of preparing test data sets :**
- III.6 Implementation of SR algorithms**
- III.7 Conclusion**

### III.1 Introduction

This Chapter is dedicated to the obtained simulation results. Our experiments consist on the implementation of bicubic SR algorithm, SRCNN SRGAN and CAR, for performance assessment, we compute PSNR and SSIM in addition to the recovered image quality. For the sake of clarity, we will use the same datasets for the different implemented SR algorithms. **For performance assessment, we will compute PSNR and SSIM**

### III.2 Used data sets

In our simulation experiments, we have used many data sets .Table 3.1 summarizes these data sets

Data sets	Amount	Format	Resolution
Set 5	5	PNG	313 , 336
DIV 2K	1000	PNG	2.793 , 250

**Table III.1:** Illustrates the different original Image

### III.3 Original Image

Original images are represented on figure III.1. More precisely, the three data sets are represented ; namely : Set 5, Div 2k and Live1. As we can see, there are different contents in these images. In our experiments, we have used



**Figure III.1:** Original High resolution images

### III.4 Materials

The code sources are written and run in python 3.10.7 spyder navigator using a PC DELL windows 10 the packages used in python are Tensor Flow which is oriented to deep learning implementation . Main codes sources are run in Google Collab because of the big data sets.

### III.5 Step of preparing test data sets :

We use the Set5 Live1 DIV2K Dataset, a well-known single-image super-resolution Data set, divided into 800 images for training, 100 images for validation, and 100 images for testing. As a "poor quality" reference, we employ bicubic X2, X4 down sampled photos; we augment the training data with random horizontal flips and rotations.As low resolution images, we use 24x24 RGB input patches we will use two categories of data sets:

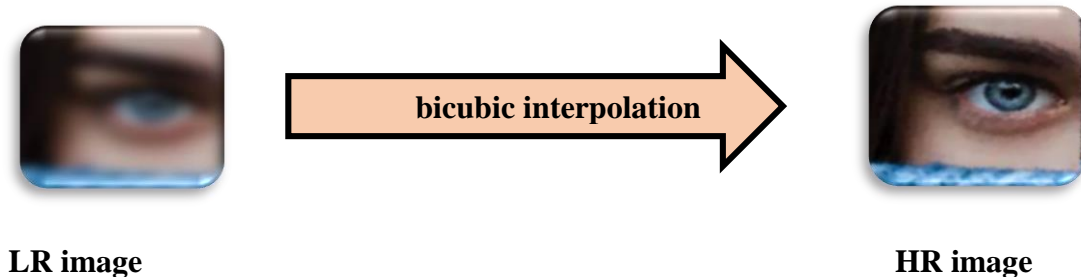
$X_2$  : Which denote the images ( $\dim \frac{X}{2}$ ,  $\dim \frac{Y}{2}$ )

$X_4$ : Which denote the images ( $\dim \frac{X}{4}$ ,  $\dim \frac{Y}{4}$ )

Three evaluation metrics are considered in our study to establish a comparison between the implemented SR algorithms: PSNR, SSIM and **computed time**. In addition to the obtained SR images. Then conventional bicubic interpolation is also implemented to highlight the advantage of deep learning. Many simulation experiments are conducted.

- ✓ To implement the algorithms, the rule used the details for the training data (80%)
- ✓ Test Data (20%)

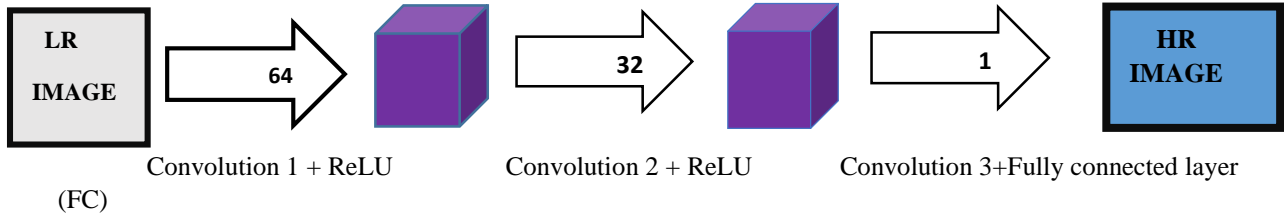
For bicubic algorithm, we first generate the low resolution LR image using a downsamplin step by factor scale 2 or 4:



**Figure III.2:**Generated LR image from HR image based on bicubic algorithm

## III.6 Implementation of SR algorithms

### III.6.1 SRCNN:



**Figure III.3:** Workflow of SRCNN

Shows the workflow of **SRCNN** the architecture of these networks is based on three main operations: convolution and RELU and Full connected layer. 64 and 32 represent the number of layers in each convolutional block.

SRCNN consists on three parts as follows:

- 1. Patch extraction and representation:** this part makes a reference to the first layer, which extracts patches from the low-resolution input image. The operation of the first layer is as follows:

$$F_1(Y) = \max(0, w_1 * Y + B_1) \quad (3.1)$$

Where  $F$ ,  $Y$ ,  $w_1$  and  $B_1$  respectively represent the mapping function, the LR image Bicubic interpolation, filters and biases

- 2. Non-linear mapping:** this part refers to the middle layer, which maps function vectors nonlinearly to another set of function vectors function. ReLU The middle layer operation is as follows:

$$F_2(Y) = \max(0, W_2 * F_1(Y) + B_2) \quad (3.2)$$

- 3. Reconstruction:** This operation combines these HR functionalities to generate The final HR image. The operation of the last layer is as follows:

$$F(Y) = W_3 * F_2(Y) + B_3 \quad (3.3)$$



The configuration of the parameters of the 3-layer neural network that will be exploited

- ✓ The neural network will have a total of 3 layers.
- ✓ The size of the input patches is 33x33 pixels.
- ✓ The first layer contains a total of 64 9x9 space filters of each, this generate a 64-dimensional feature vector.
- ✓ The second layer includes 32 filters of 1x1 size.
- ✓ The third layer is formed by a single filter (since we only work
- ✓ In the luminance channel) of size 5x5.

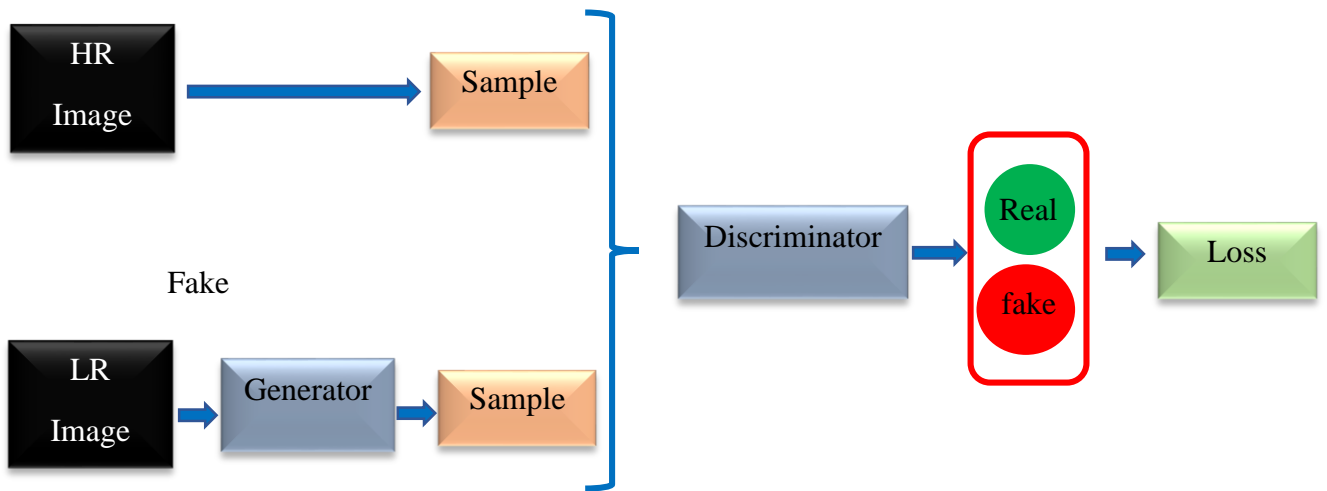
The LR input image are processed by SRCNN .the obtained results are shown on figure 3.5 we easily see that the visual quality of the outputs is considerably enhanced. this result is on concordance with the enhancement of the corresponding PSNR and SSIM criteria shown on Table 3.2. Note that the represented values of PSNR and SSIM are the computed average values

**Table III.2:**The average results of PSNR (dB), SSIM on the Live1 dataset

méthodes	Scale	PSNR	SSIM
SRCNN	2	41.15	0.9521
	4	30.11	0.8532

### III.6.2 SRGAN :

The works SRGAN is shown on figure 3.5 where architectures have two parts: a generator and a discriminator. The generator generates data based on a probability distribution, while the discriminator tries to infer data from the input dataset or generator. After that, the generator attempts to optimize the created data to deceive the discriminator. The generator and discriminator architectural details are listed below.



**Figure III.4:** Show the workflow of SRGAN

### **Generator Architecture:**

Instead of deep, the generator architecture fig 3.7 uses residual networks. This is due to the residual network's use of a connection type known as skip connections. **B** residual blocks (16) were generated by **Reset**. Two convolutional layers with tiny 3x3 kernels and 64 feature maps are utilized in the residual block, followed by batch-normalization layers with Parametric **ReLU** as the activation function.

Instead of deep, the generator architecture uses residual networks. This is due to the residual network's use of a connection type known as skip connections. **B** residual blocks (16) were generated by **Reset**. Two convolutional layers with tiny 3x3 kernels and 64 feature maps are utilized in the residual block, followed by batch-normalization layers with **ParametricReLU** as the activation function.

### **Discriminator Architecture:**

The discriminator's job is to tell the difference between real HR images and manufactured SR the network has eight convolutional layers, each with 33 filter kernels, increasing by a factor of from 64 to 512 kernels. When the number of features is doubled, stridden convolutions are employed to lower the image resolution. To obtain a probability for sample classification, the 512 feature maps are followed by two thick layers with a **leakyReLU** applied between them and a final sigmoid activation function.

### **Loss Function:**

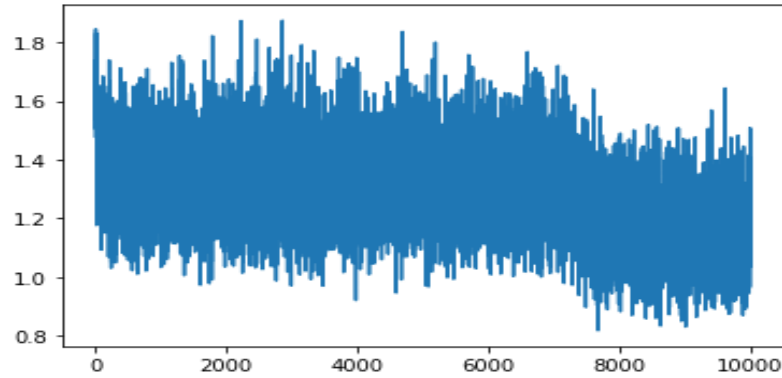
**Content Loss:** This VGG loss is based on the **ReLU** activation layers of the pre-trained 19 layer VGG network.

$$I_{VGG/i,j}^{SR} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta G}(I^{LR}))_{x,y})^2 \quad (3,4)$$

### Résult PSNR and SSIM:

methods	Scale	PSNR	SSIM
SRGAN	2	<b>29.80</b>	<b>0.8465</b>
	4	<b>27.45</b>	<b>0.7.395</b>

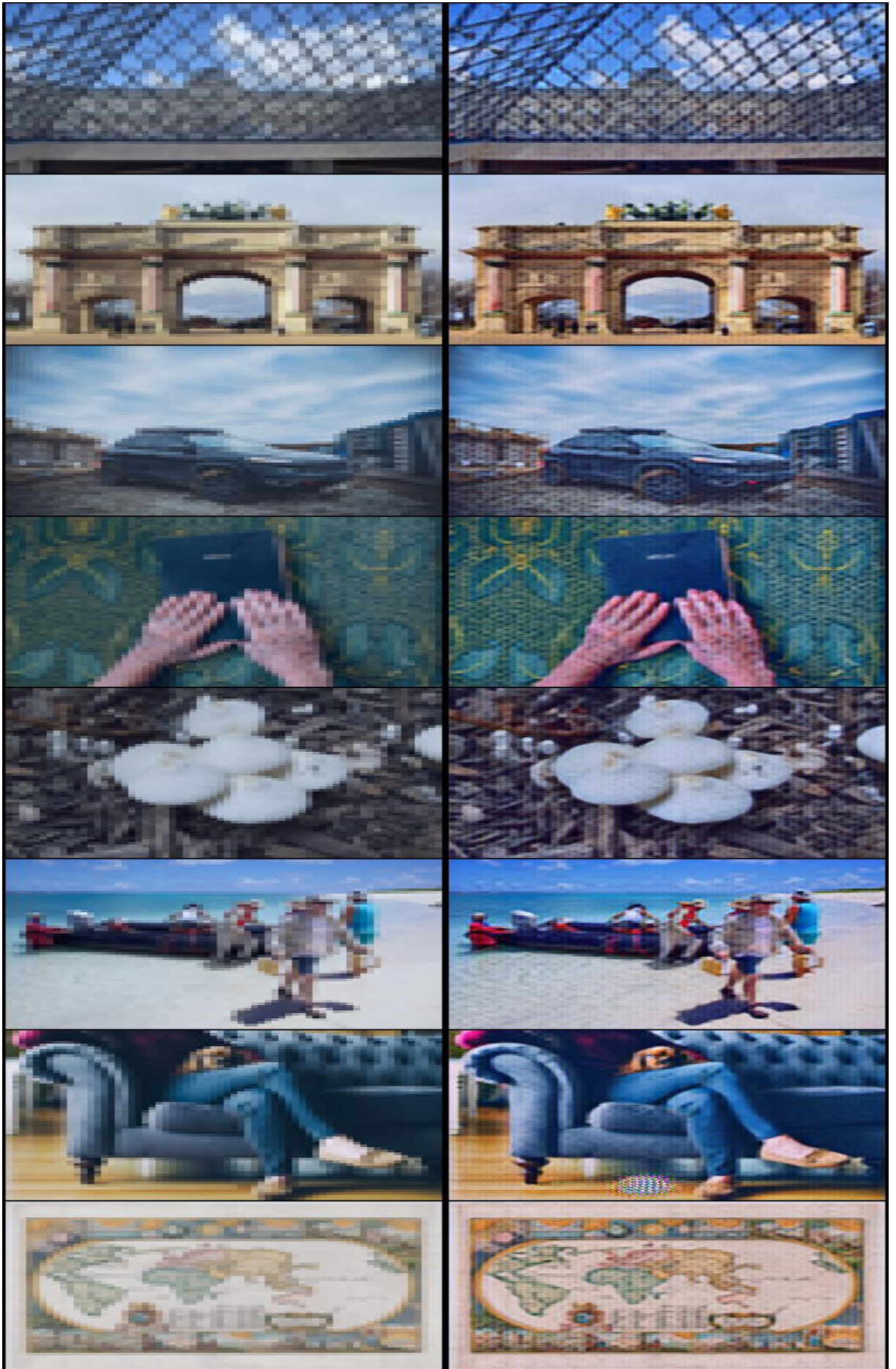
**Table III.3:** The average results of PSNR (dB), SSIM on the Live1 Set5 and Div2k dataset (SRGAN)

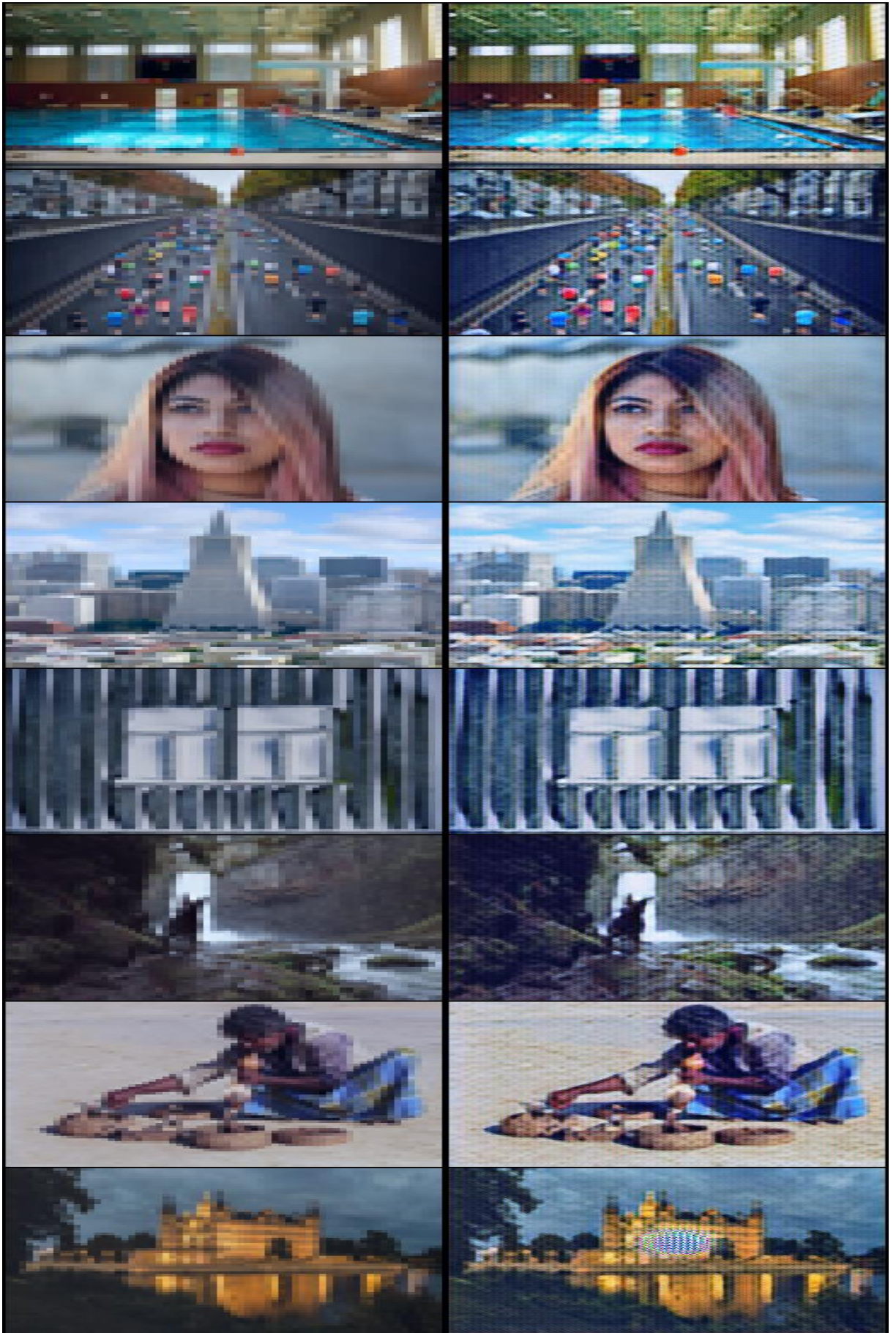


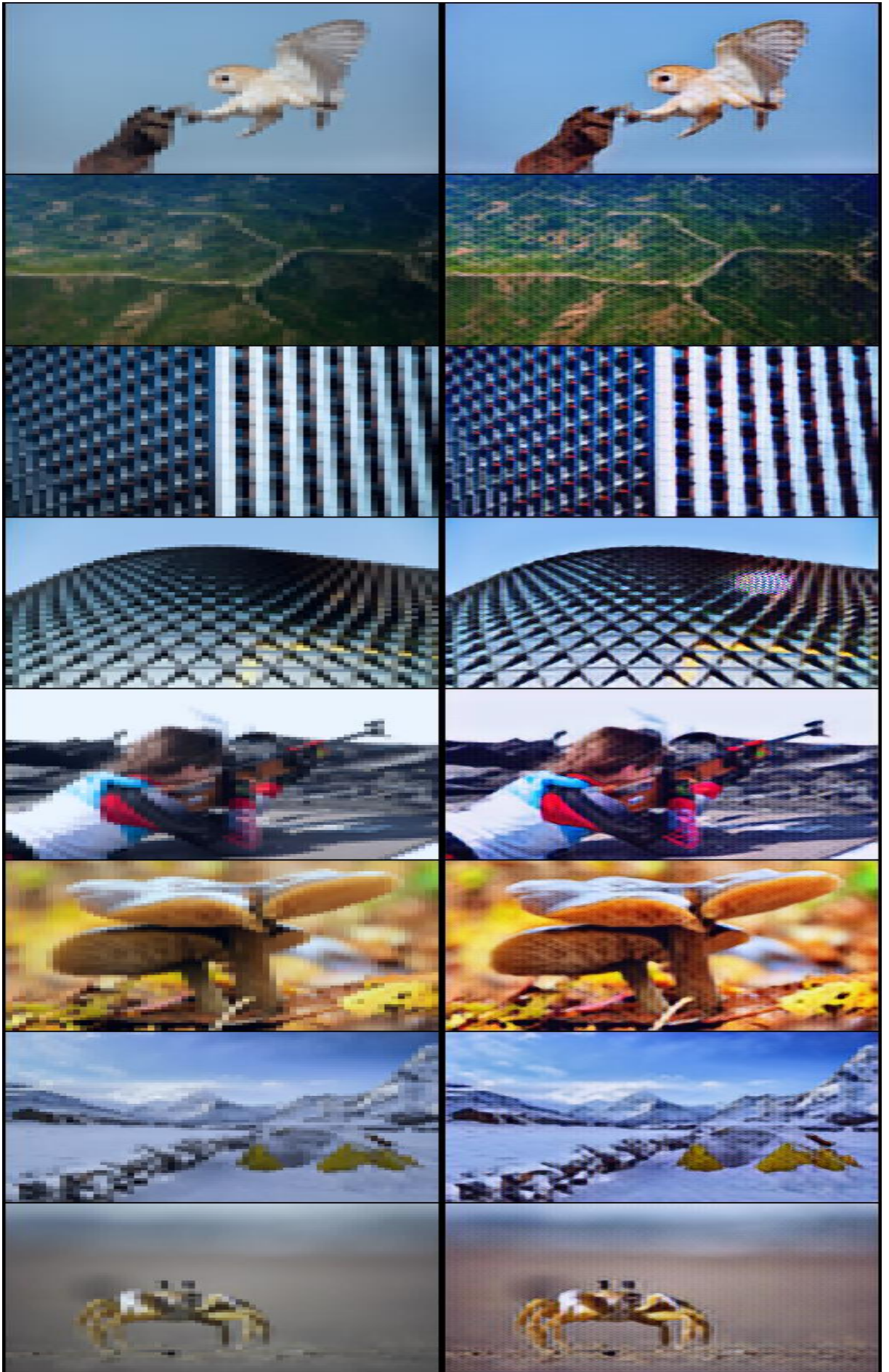
### Obtained SR images

The obtained SR images are shown in Fig The first column represents LR images while the second column represents the obtained SR images.









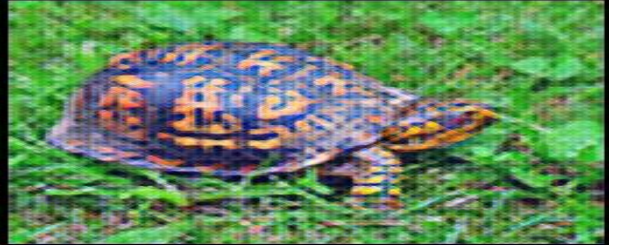
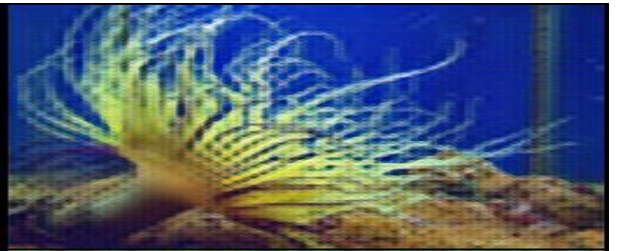






Figure III.5: SRGAN results

### III.6.3 CARS: Continuous Evolution for Efficient Neural Architecture Search

As we have described in chapter II, the architecture of the content-adaptive resampled (CAR) model consists on a ResamplerNet and SRNet block .the framework is made up of two primary components

The ResamplerNet is in charge of predicting content-adaptive resampling kernels based on the input HR image, and then applying the resampling kernels to the input HR image to produce the downscaled image.










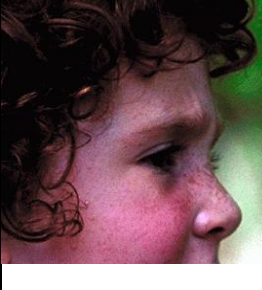

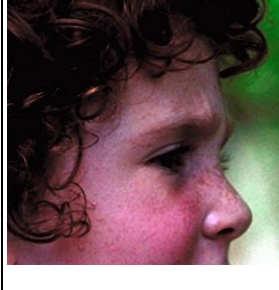



The SRNet ,on the other hand, accepts the downscaled image as input and attempts to reconstruct the original HR image.Table 3 show the computed average values of the two considered metrics PSNR and SSIM. The original images, the generated down sampling images and the obtained SR images are shown on Fig3 As we can see the quality of SR images is considerably enhanced and is closed to the original images.

**Table III.4:** The average results of PSNR (dB), SSIM on the Live1 Set5 and Div2k dataset (SR) CAR
















methods	DATA	Scale	Mean PSNR	Mean SSIM
CAR	Set5 And	X2	34.90	0.9446
	LIVE1	X4	29.71	0.8393

Name images	orig-PNG	down.png	recon.png	Test images Dataset live1(PSNR)	Format
monarch				41.3466 4071896 455	PNG
Lighthouse 3 (scale X2)				35.7669 1473070 391	PNG
house				35.7664 9524651 933	PNG
Caps				40.9641 7477654 1626	PNG

**(DATA set5 X2)**

images	Orig.png	Down.png	Rect.png	Test image dataset set5	Format
Baby				39.644130 382461604	png
bird				44.51 451735299 871	png
butterfly				36.853120 647756185	png
head				36.274054 67660417	png
woman				37.503940 550805304	png

**(Data Set5 X4)**

images	Down.png	Orig .png	Rect .png		format
baby				35.27665 18295533 8	png
bird				37.96306 62569646	png
butterfly				30.95272 3815401	png
head				33.79241 60805307 3	png
Woman				32.84423 94922988 1	png

**Dataset (Live1 x4)**

Nom	Down	orig	rect	Result PSNR	
bikes				27.75517 25549432 58	png
caps				35.45394 33984498 8	png
house				31.52314 21393992 2	png
Lighthouse 2				28.95282 90189013 72	png
Lighthouse 3				30.47500 09525239 88	png

Figure III.6: CAR SR results

### **III.7 Conclusion:**

In our study, three SR image algorithms based on deep learning are considered in addition to the conventional bicubic SR algorithm. We have used two data, namely : sets 5 and LIVE1. The performance assessment of the considered SR algorithms are conducted in terms of two metrics PSNR and SSIM in addition to the visual quality of the obtained SR images. For each experiment, we first generate the LR images from the original ones, in this step we obtain X2 images or X4 images We apply the SR algorithm to get the SR images. From the obtained results, we can easily remarque that CAR highlight the other SR algorithms in terms of visual uality and the computed metrics. Although, the performances of the three SR algorithms allow approximatevely the same enhancement but highlight the bicubic algorithm.

## **General conclusion**

In this work, we have studied SR image algorithms based on deep learning. More precisely, three SR algorithms are implemented :

SRCNN (based on CNN architecture), SRGAN (based on GAN strategy) and CAR algorithm. Two data sets are used in simulation experiments, namely : set 5 and live1.

For performance assessment, two metrics are computed :

PSNR and SSIM. Many simulation experiments are conducted through python under anaconda (navigator spyder) and googlecolab. For the limitation space we have choosen a part of our obtained results.

From the obtained results, we can easely remarque that CAR highlight the other SR algorithms in terms of visual quality and the computed metrics.



## Références

- [1] Bendong Zhao, H. L. (2017). Convolutional Neural Network (CNN) for Time Series Classification. *Journal of Systems Engineering and Electronics* Vol. 28, No. 1, February 2017, pp.162 – 169
- [2] Kirsch, b. J. (s.d.). *Machine learning for Dummies* IBM Limited Edition Hoboken, NJ 07030-5774.
- [3] Ray, S. (2020). Proven Ways for improving the “Accuracy” of a Machine Learning Model. <https://www.analyticsvidhya.com/blog/2015/12/improve-machine-learning-results/>
- [4] BIN QIAN, J. S. (2020). Orchestrating the Development Lifecycle of Machine Learning-Based IoT. arXiv:1910.05433v5 [cs.DC]
- [5] Zhihao Wang, J. C. (2021). Deep Learning for Image Super-Resolution: A Survey. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, VOL. 43, NO. 10, OCTOBER 2021
- [6] Nielsen, M. (2019). *Neural Networks and Deep Learning* ;<http://neuralnetworksanddeeplearning.com>
- [7] Andrew Blais, D. M. (2020). *Neural network*;IBM Education.
- [8] Kadam, S. (2020). Neural network:inside in single neuron. <https://medium.com/analytics-vidhya/neural-network-part1-inside-a-single-neuron-fee5e44f1e>
- [9] Bento, C. (2021). Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis, <https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141>
- [10] delua, j. (2021). Supervised-vs-unsupervised-learning, SME, IBM Analytics, Data Science/Machine Learning
- [11] Togashi, R. Y. (2018). Convolutional neural networks: an overview.<https://doi.org/10.1007/s13244-018-0639-9>
- [12] Sharma, P. (2020). A Comprehensive Tutorial to learn Convolutional Neural. using in,analytics vidhya <https://www.analyticsvidhya.com/blog/2018/12/guide-convolutional-neural-network-cnn/>
- [13] WaqarAhmad, H. Z. (2022). A new generative adversarial network for medical images super. [www.nature.com/scientificreports](http://www.nature.com/scientificreports),<https://www.nature.com/articles/s41598-022-13658-4>
- [14] Zhihao Wang, J. C. (2020). Learning super-resolution jointly from external and internal examples. *IEEE Transactions on Image Processing*, 24(11):4359–4371,
- [15] Syed Muhammad Arsalan Bashir, Y. W. (2021). A comprehensive review of deep learning-based single image super-resolution, <https://peerj.com/articles/cs-621/>
- [17] R.Keys. (1981). Cubic convolution interpolation for digital image processing.in *IEEE Transactions on acoustics ,speech ,and signal processing* ,vol 29 no 6,pp.1153-1160
- [18] Ken M. Nakanishi, S.-i. M. (2019). Neural multi-scale image compression.Machine learning(stat.ML);computer vision and pattern recognition ; <https://arxiv.org/abs/1805.06386>
- [19] J. Kim, J. K. (2016). Accurate image super resolution using very deep convolution networks., <https://arxiv.org/abs/1511.04587>
- [20] Hang Zhao, O. G. (2017). loss function for image restoration with neural networks. arXiv:1511.08861v3 [cs.CV]
- [21] Dong C, L. C. (2016). Image super-resolution using deep convolution networks. In *ECCV*
- [22] Chen, W. S. (2019). Learned Image Downscaling for Upscaling using Content Adaptive. School of Remote Sensing and Information Engineering, Wuhan University; arXiv:1907.12904v2 [cs.CV]
- [23]Chao Dong, C. C. (2015). Image Super-Resolution Using Deep Convolutional Networks In *ECCV*

- [24]C. Dong, C. C. (2014). learning a deep convolution network for image super resolution,in ECCV.
- [25]Bee Lim, S. S. (2017). Enhanced Deep Residual Networks for Single Image Super-Resolution.arXiv:1707.02921
- [26]Almohammad A, G. G. (2010). Stego image quality and reliabilityof PSNR.  
<https://ieeexplore.ieee.org/document/5586786>