POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Master's Degree Thesis

A generative adversarial network approach to single image super-resolution of open-source satellite imagery

Politecnico di Torino

Supervisor

Prof. Roberto FONTANA

Co-supervisor

Mirko ZAFFARONI

Candidate

Filippo BARBA

October 2021

Abstract

Image super-resolution is a widely studied ill-posed problem in computer vision, where the objective is to convert a low-resolution image to a high-resolution one. Conventional methods for achieving super-resolution, such as interpolation-based methods, require a lot of pre/post-processing and optimization.

Thanks to the rise in popularity of Deep Learning methods over recent years, several studies have shown how learning methods such as convolutional neural networks and generative adversarial networks can be used to perform super-resolution tasks with competitive results when compared to prior state of the art methods.

This thesis proposes a focus on the application of super-resolution methods to open-source low-resolution satellite imagery gathered from the Sentinel-2 ESA's satellite in the RGB domain. The open data policy plays an important role in the choice of this dataset, alongside other key characteristics of the Sentinel-2 mission, most notably the high revisitation frequency of the global covered area from 56°S to 84°N, which happens every 10 days under the same viewing angles.

The design of a selection of renowned neural network based models for super-resolution will be subject to analysis, along with a study of the models' applications to the dataset of choice and of the respective performances when using different upscaling factors (2x, 4x, 8x). Furthermore, this dissertation proposes a generative adversarial network architecture with multiple discriminators. The goal of this multi-discriminator model is to optimize the training process of the generative network over different scaling factors in a single training procedure.

Super-resolution applications to the satellite imagery domain are of notable relevance given the impact that well-performing methods can have on existing algorithms relying on this kind of data, such as image classification, object detection, and environmental monitoring among others.

Acknowledgements

First of all, I would like to express my profound appreciation to LINKS Foundation for giving me the opportunity and the resources to work on this enthralling topic.

A particular thank you goes to Ph.D. student Mirko Zaffaroni, whose knowledge and aid have been critical in realizing this thesis work.

I also wish to thank Professor Roberto Fontana for their enthusiasm in guiding me through this experience.

Heartfelt thanks go to my colleagues and those I am proud to call friends for being an essential source of encouragement, celebrating with me through the highs, and motivating me through lows. You helped me strive to reach my full capacities, even when I did not believe in myself.

I would like to thank my family for believing in me and cheering for my every success.

My sincerest gratitude goes to my sister Camilla for being my role model over the years and to my brother Alessandro for inspiring me to try to be among his.

Last but not least, I would like to thank my parents. The endless motivation and patient support they have given me through the years have been fundamental in helping me achieve all I have.

Table of Contents

List of Tables

List of Figures

Acronyms

SR
Super-Resolution

SISR
Single Image Super-Resolution

MISR
Multi-Image Super-Resolution

PSNR
Peak Signal to Noise Ratio

SSIM
Structural Similarity Index Measure

NIQE
Naturalness Image Quality Evaluator

HR
High Resolution

LR
Low Resolution

GAN
Generative Adversarial Network

SotA
State of the Art

Chapter 1

Introduction

In this thesis project we explore the use of Deep Learning models such as Deep Neural Networks, specifically Convolutional Neural Networks and Generative Adversarial Networks, for single image super resolution tasks on images from the satellite imagery domain. This project was carried out with the aid of LINKS Foundation, which proposed the thesis topic and provided the computing facilities necessary to perform the studies contained in this document.

1.1 Motivation

The particular choice of task and data used for this thesis project was driven by the potential implications of successful super resolution techniques applied to satellite images. Applications that make use of satellite images have extremely crucial real-world implications, as many satellite imagery-based algorithms are nowadays used for environmental monitoring, global climate change, natural disaster prediction and land development, urban planning and engineering, agricultural and forestry monitoring to name a few. Most of these algorithms could benefit from using high resolution satellite imagery, data which is usually not open-source nor publicly available due to the high costs involved with both launching and maintaining in orbit satellites that are able to capture such images.
While many private companies can currently provide extremely high resolution images which go up to tens of cm per pixel, some public projects sponsored by space agencies around the globe are fundamental as they aim to provide more openly available data.
In particular, the open-source dataset chosen for this thesis project is the one collected by the European Space Agency's Sentinel mission, specifically by the Sentinel-2 satellites. The Sentinel-2 mission has a few key characteristics, most important of which are frequently up to date collected data and a free and open

data policy, which have been critical in the decision of this dataset for this thesis project.

1.2 The Super-Resolution problem

In most digital imaging applications, high-resolution images are preferred and often required to successfully carry out tasks. Image super-resolution is a widely-studied ill-posed problem in computer vision, where the objective is to generate high-resolution images starting from low-resolution ones. Super resolution algorithms aim to produce details finer than the sampling grid of a given imaging device by increasing the number of pixels per unit area in an image. The problem, where from a low-resolution image, often corrupted by blur, aliasing artifacts, noise and visual distortions, a high-resolution image is generated, is inverse and ill-posed as there does not exist a unique solution. Super resolution techniques can be applied in many scenarios where multiple frames of a single scene can be obtained, or various images of a scene are available from numerous sources.
These techniques can be applied in various fields such as medical imaging where more detailed image details are required on demand, and high-resolution medical images can be used by doctors improve the chance of a correct diagnosis, in homeland security and surveillance where there may be a need to increase the resolution a specific section of interest in a scene (such as zooming on the face of a criminal, or on license plate), in computer vision where they can be used to improve the performance of pattern recognition and other applications such as facial image analysis, text image analysis, biometric identification and fingerprint image enhancement, among others.
Super resolution is particularly of great interest in satellite imaging applications, such as remote sensing, object detection and environmental monitoring, where high resolution images are not always openly available and frequently updated.
In general, image super resolution problems can be classified as belonging to two main categories, according to how the super resolved images are generated.

1.2.1 Single Image Super-Resolution

Single image super resolution (SISR) aims to recover a high-resolution image starting from a single given low-resolution one. Since in most cases there is no underlying ground truth to be used for evaluating the quality of a super resolved image, the significant issue is to generate an image that is visually acceptable, without introducing excessive noise and degradation. Most of the currently best performing SISR algorithms are learning based and aim to complete the missing details of the output super resolved image exploiting relationships between low-resolution and high-resolution images from a training dataset. Single image super

resolution will be the main topic of this thesis project.

1.2.2 Multi-Image Super-Resolution

Multi-image super resolution (MISR) involves the extraction of information from many low-resolution observations of the same scene to reconstruct high resolution images. With the availability of more data from the multiple observations of the scene, it is possible to obtain a more accurate reconstruction than through single-image methods. In recent years, these techniques have been exploited to address super resolution problems in the context of enhancing video sequences, as frames which are temporally close to each other are able to provide the needed multiple observations to carry out the task. However, multi-image super resolution techniques are rarely exploited in the context of satellite imagery. In 2018, the European Space Agency has published a challenge to super-resolve multi-temporal PROBA-V satellite imagery [1]. For the challenge, Salvetti et. al [2] proposed a new architecture that uses an end-to-end learning approach which exploits both temporal and spatial correlations between multiple images.

1.3 Purpose and Goal

This thesis will examine several Deep Learning techniques, focusing on Convolutional Neural Networks and Generative Adversarial Networks, in order to determine which techniques produce the most promising results in super resolution tasks applied to satellite images, comparing them in conjunction with traditional image upscaling techniques. The objectives of this project are to:

- Implement and analyze Deep Learning base methods for single image super resolution based on current State of the Art models using a satellite imagery dataset.

- Propose and implement a Deep Learning based architecture for single image super resolution that directly learns an end-to-end mapping between the low resolution images and high resolution ones, focusing on the possibility of creating a generalized model that is able to single-handedly produce differently scaled super resolved images starting from a single low resolution image.

- Study and determine appropriate metrics to evaluate the super resolved images produced by the proposed architecture and compare them with those generated with State of the Art models.

1.4 Approach and Methodology

Image super resolution is a topic which, thanks to the rise in popularity of machine learning and deep learning, has gathered the interest of many different research studies over recent years. While few research is done focusing on a specific type of images, that of satellite imagery is a domain which has gathered the interest of many researchers and has been subject to analysis in different papers. This interest has been driven by the motivations previously discussed. For this thesis work, the main objective was to propose an easily replicable pipeline, from the creation of a dataset creation to the choice of the appropriate architecture, while proposing relevant improvements to already reliable State of the Art models. The first step of the project regarded the creation of the dataset. There exist many different variations of satellite images, determined by the spectral bands over which data is acquired from satellite, or the type of data in and of itself (e.g., remote sensing data, depth maps, RGB images). The choice of the dataset determined the exact scope of the thesis project: choosing RGB satellite imagery pushed the decision towards a single image super resolution task. After creating the dataset followed a study of the most relevant State of the Art Deep Learning-based models, in order to determine a baseline performance of what current technologies are able to provide in terms of generated super resolved images. While several models have been tested, only the ones relevant to this thesis contributions will be described in detail. Applying the selected models to this specific problem and dataset played a fundamental role in choosing a reference architecture to use as a starting point for proposing improvements and modifications, in conjunction with the hardware and time resources available to run the experiments. In this project, the aspect of optimizing the models and training hyperparameters and procedures have been deemed of secondary importance, mostly because of time and resource constraints which have been the major obstacles in dealing with extremely complex and computationally demanding architectures. Refining the proposed improvements, main topic of this thesis work, is what required the majority of the time in terms of experiments, in order to reach acceptable performances worth mentioning in this report.

1.5 Contributions

The main contributions of this thesis project are the following:

- Propose a satellite imagery dataset for single image super resolution tasks with the objective of training a model that can be used to super resolve open source RGB satellite images collected from the European Space Agency's Sentinel mission.

- Propose a Generative Adversarial Network architecture, based on existing State of the Art models, to perform single image super resolution on three different scales to aim for a generalized model capable of handling multiple super resolution tasks. The aim is to produce a model capable of producing results comparable to those of current State of the Art models.
- Propose a proof of concept application of the impact of super resolution techniques to images used for solar panel detection algorithms by super resolving Sentinel-2 test data.

1.6 Outline

The rest of the report is organized as follows:

- Chapter 2 proposes an overview of the most relevant theory and concepts that are fundamental to the understanding of the work done with this thesis project.
- Chapter 3 proposes an analysis of some of the most relevant State of the Art models for single image super resolution applications.
- Chapter 4 contains the details of how the data used for the project has been gathered, explains how the previously analyzed State of the Art models have been applied to this specific case study, along with the motivations behind the proposed architecture and a comparison of the obtained results.
- Chapter 5 concludes this thesis project with some considerations on the experiments performed and a comment on possible ideas for future work related to what has been proposed.

The results reported in this document appear as they have been obtained without any manipulation and appropriate sources are given credit wherever possible to avoid plagiarism.

Chapter 2

Relevant Theory

In this section we will explore the underlying theory of the main topics touched in this thesis work. We will start by more formally defining the problem of Image Super-Resolution, exploring how it has been tackled via traditional, mathematical approaches and how deep learning techniques have been adapted to this sort of problems.
We will then move onto defining what Deep Learning is, how it relates to Machine Learning and why the Artificial Intelligence field is nowadays playing an increasingly important role in tackling difficult to solve real world problems.
The contents of this chapter are mostly sourced from [3] and [4], with additions from other online sources referenced accordingly.

2.1 Image Super-Resolution

With Super-Resolution we define the problem of creating or recovering a high resolution image starting from a low resolution one. This problem presents significant challenges: it's inherently ill-posed since a multiplicity of solutions exist for any given low-resolution image. By artificially enlarging the resolution of an image we are effectively generating new information regarding such image, which was previously not present, and we must do so in a coherent way, such as not to alter the visual perception of the image while avoiding the introduction of significant noise which would degrade the picture.
This problem has many real world applications, such as medical imaging, surveillance and security, astronomical imaging and satellite imaging. The latter will be the application on which this thesis will focus on.

2.1.1 Interpolation-based Super-Resolution Techniques

These techniques are simple algorithms which have been commonly used until recently for performing image upscaling. Nearest neighbor interpolation is one of the simpler ways of increasing image size, and works by replacing every pixel with the nearest pixel in the output. When upscaling this results in multiple pixels of the same color to be present. This can preserve sharp details in pixel art, but also introduce jaggedness in previously smooth images. 'Nearest' doesn't imply mathematical nearest, one common implementation is to always round down. Rounding this way produces fewer artifacts and is faster to calculate.

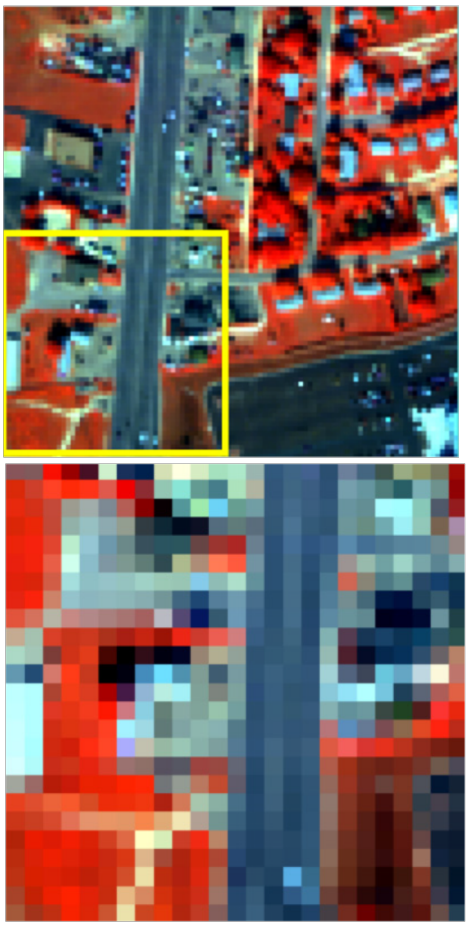

Figure 2.1: Nearest neighbor upscaling [5]

Bilinear interpolation works by interpolating pixel color values, introducing a continuous transition into the output even where the original material has discrete

transitions. It is performed on a 2-dimensional grid using linear interpolation first in a direction and then in the other. Although this is desirable for continuous-tone images, this algorithm reduces contrast (sharp edges) in a way that may be undesirable depending on the image to be scaled.

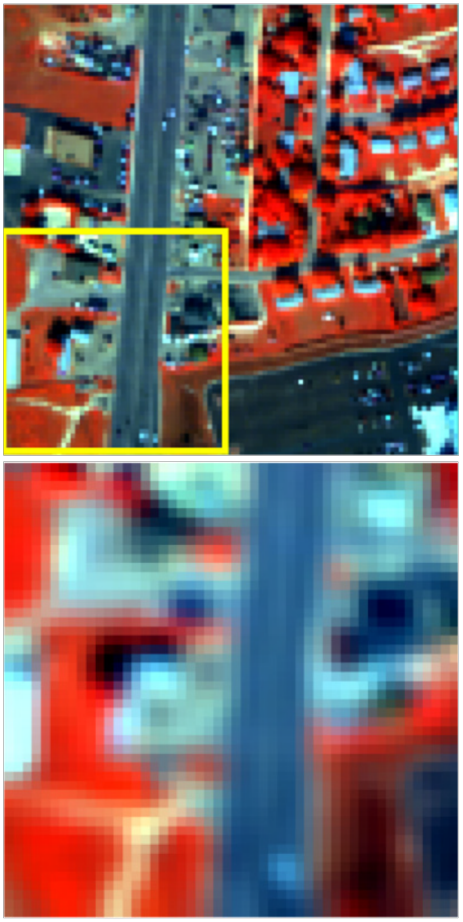

Figure 2.2: Bilinear interpolation [5]

Bicubic interpolation yields substantially better results, with an increase in computational cost.
It is an extension of cubic interpolation on a 2-dimensional grid, and offers smoother results compared to the previous two methods.

There also exist more complex resampling algorithms, such as Sinc and Lanczos. Sinc resampling provides, in theory, the best possible reconstruction for a perfectly band-limited signal. In practice, the assumptions behind the method are difficultly

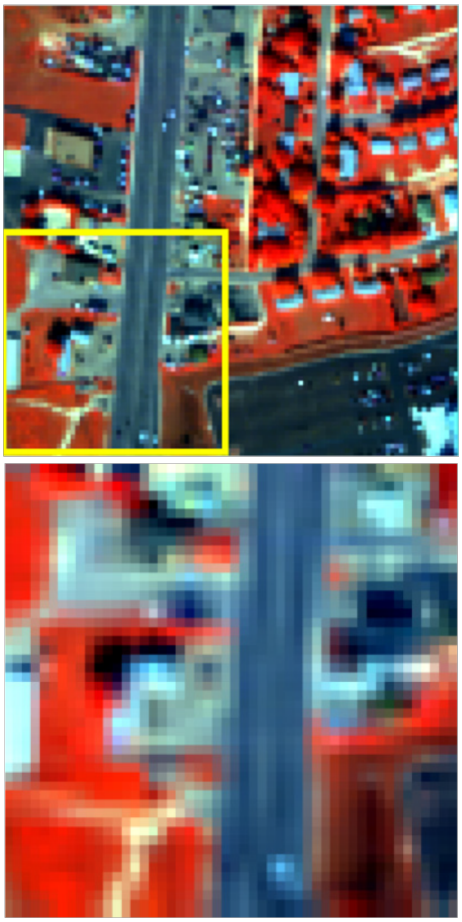

Figure 2.3: Bicubic interpolation [5]

met by real-world images.
Lanczos is an approximation of the Sinc method and usually yields better results. Bicubic interpolation can be regarded as a computationally efficient approximation to Lanczos resampling.

As all of these methods are not learning-based, they lack in generalization and usually introduce visible aliasing artifacts.

2.1.2 Deep Learning-based Super-Resolution Techniques

Thanks to the rise in popularity of Machine Learning and Deep Learning, recent research has been extensively performed on neural network-based models applied to image scaling. Neural networks allow to create extremely complex models that, through a learning procedure, are able to produce far better results when compared

Figure 2.4: Original image [6]

Figure 2.5: LR image [6]

Figure 2.6: 4x nearest neighbor upscaling[6]

Figure 2.7: 4x bilinear interpolation[6]

Figure 2.8: 4x bicubic interpolation[6]

to the techniques showcased previously.
In the following sections we will delve into the details of what machine learning and deep learning are, with a focus on the inner workings of neural networks to provide necessary context before analysing these methods. Since this thesis project focuses on this class of methods, the state of the art of deep learning-based super resolution models will be analyzed in detail in the next chapter.

2.1.3 Evaluation metrics for image quality assessment

To evaluate the quality of a super-resolved image, several metrics can be used according to what needs to be measured. A first example is the Peak Signal to Noise Ratio (PSNR), which is the ratio between the maximum possible value, or power, of a signal and the power of distorting noise that affects the quality of its representation. Since many signals have a very wide dynamic range, which is the ratio between the largest and smallest possible values of a quantity, the PSNR is usually expressed in terms of the logarithmic decibel scale. Consider a 2D $m \times n$ image I and its noisy approximation J, the PSNR can be mathematically described

in dB as follows:

$$PSNR = 20\log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \tag{2.1}$$

where MAX_I represents the maximum pixel value of the image I, and MSE indicates the Mean Square Error and is defined as:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}|I(i,j) - J(i,j)|^2 \tag{2.2}$$

PSNR is one of the most commonly used measures for assessing the quality of reconstruction of lossy images. The signal in this case is the original image, and the noise is the error introduced by scaling. Typical values for the PSNR in image super-resolution tasks is between 25 and 35 for State of the Art deep-learning techniques, where higher is better. In case no noise is present, the value of the MSE is zero, and the PSNR is considered infinite. Although a higher PSNR generally indicates that the reconstruction is of higher quality, in some cases it does not translate in a visibly better image. Generally, PSNR has been shown to perform poorly compared to other quality metrics when it comes to estimating the quality of images and particularly images videos as perceived by humans. For this reason there exist many other metrics, called perceptual metrics, that can be considered, either instead of or in conjunction with PSNR, to better evaluate the perceived quality of a super-resolved image.

One of these metrics is the Structural Similarity Index Measure (SSIM). SSIM is a method for predicting the perceived quality of digital images and videos and measures the similarity between two images. It's a perception-based model that considers image degradation as perceived change in structural information, while also incorporating important perceptual phenomena, including both luminance masking and contrast masking terms. The difference with other techniques such as PSNR is that they estimate absolute errors. Structural information relies on the idea that the pixels have strong inter-dependencies when spatially close to each other. These dependencies carry important information about the structure of the objects in the visual scene. Luminance masking is a phenomenon whereby image distortions (in this context) tend to be less visible in bright regions, while contrast masking is a phenomenon whereby distortions become less visible where there is significant activity or "texture" in the image.

Other perceptual scores include the Naturalness Image Quality Evaluator (NIQE)[7], which is a completely blind image quality analyzer that only makes use of measurable deviations from statistical regularities observed in natural images,

without training on human-rated distorted images, and without any exposure to distorted images. It is based on the construction of a quality aware collection of statistical features based on a simple and successful space domain natural scene statistic model. These features are derived from a dataset of natural, undistorted images. A lower NIQE score is an indicator of better perceptual quality.

2.2 Artificial Intelligence

The idea of Artificial Intelligence (**AI**) has antique precursors ranging from myths, stories and legends from ancient civilizations narrating of artificial entities capable of carrying out tasks associated with human intelligence, to classical philosophers pursuing a description of the processes that constitute human thinking.
Thanks to the invention of the programmable digital computer in the 1940s and breakthrough research in fields such as neurology, showing that the brain could be compared to an electrical network of neurons, the field of AI research was founded as an academic discipline in **1956** at the **Dartmouth Workshop** [8], organized by M. Minsky, J. McCarthy and scientists C. Shannon, known as the father of information theory, and N. Rochester. This conference is widely considered the **birth of modern AI**, including fields such as engineering and mathematics, psychology, economics and political sciences.
Because of excessive expectations and limited technological capabilities, AI experienced intermittent highs, fueled by optimism from researchers and matched by conspicuous funding in academia, and lows, known as "AI winters", caused by the limited computing resources, scepticism and criticism, which caused the field to undergo a very slow progression, especially when compared to confident goals of the early days.
In recent years, access to large quantities of data in conjunction with cheaper and faster computing hardware and successful applications of Machine Learning (**ML**) techniques in real case scenarios, allowed AI to gain immense interest from many different fields, boosting it to the position it currently yields today.

2.2.1 Machine Learning

Machine Learning (**ML**) research emerged out of AI, specifically from the interest of researchers to find a way of having machines learn from data. ML algorithms are currently used in a wide variety of scenarios, such as speech recognition, natural language processing and computer vision.
These algorithms are able to improve autonomously via experience and the use of data, or as more formally defined by Tom M. Mitchell: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with

experience E." [9].
Nowadays, while many consider ML to be a subset of AI, others argue that only a portion of ML intersects with the field of AI, considering the two to be separate but overlapping sets.

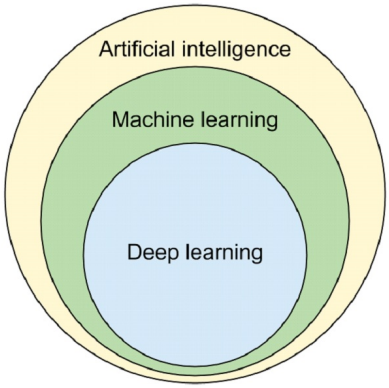

Figure 2.9: ML as a sub-field of AI [10]

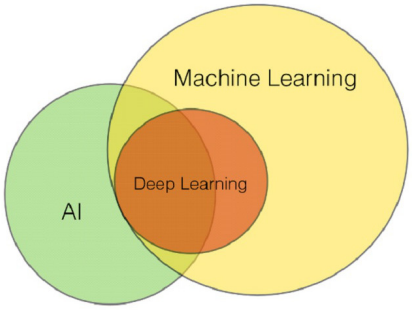

Figure 2.10: Part of ML as a sub-field of AI [10]

Some outline the difference between the two fields in the fact that ML uses passive observations to learn, whereas AI requires an interaction with the environment in order to learn.
ML techniques can be divided into groups, depending on the type of feedback available to the system, but all share some common traits which can be described by the idea of a ML Pipeline. A ML Pipeline describes the common steps of the ML process.

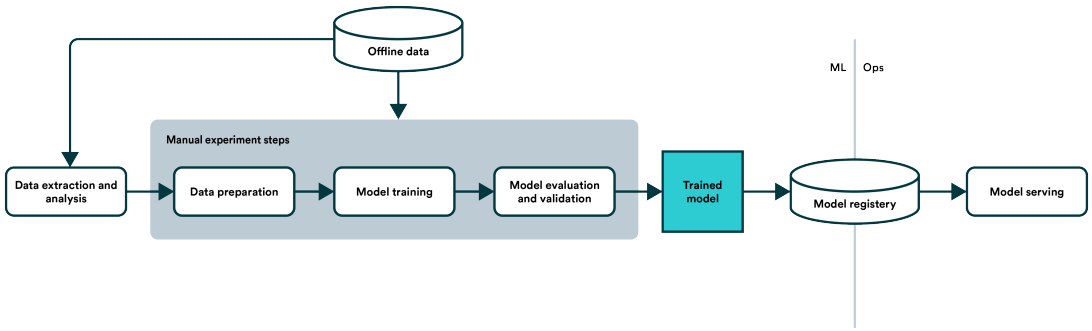

Figure 2.11: ML Pipeline steps [11]

The first steps revolve around the collection, extraction and preparation of the data, which varies according to each specific application scenario. This collected data needs, in general, to be divided into 3 sub-sets: **training set**, **validation**

set and **test set**. This partition is extremely important as it allows to perform the succeeding steps while carrying as little bias as possible, especially in the validation and test phases.
These phases are followed by the creation, training, evaluation and validation of different models, in order to try and find the one that best adapts to the specific requirements, may them be performance, memory footprint, power consumption, etc.
During the training phase the model is trained on the training portion of the dataset. It is then evaluated on the validation set to avoid overfitting on the training dataset: if models were to be tested on the same data used for training, the performance would be artificially perceived as good because the model has already seen that data, usually causing poor performance on newly-seen real-world data. This phase is usually performed in conjunction with cross-validation techniques like K-Fold or Leave One Out: the former consists in dividing the entire dataset in K "folds" (partitions) of equal size and training the model on K-1 folds, using the remaining for validation and cycling through the folds, while the latter works by leaving 1 sample for validation while the rest is used for training.
After having selected the model based on the validation performance during cross-validation, the model can be retrained using both training and validation for a final evaluation using the test set.
Apart from parameters to be learned during training, many models also have tunable hyperparameters. There exist many different performance metrics that can be used to evaluate the goodness of a given model, and the choice of the metric depends mostly on the type of data, the type of the model and the type of ML technique employed: for supervised learning tasks such as classification, some of the most commonly used metrics include accuracy, F1-score and the ROC (receiver operating characteristic) curve, while for unsupervised tasks such as clustering metrics like Rand Index, Mutual Information-based scores and Silhouette are among the most frequently used.
In the following sections we will go into the details and characteristics of the 3 main groups of ML techniques.

Supervised Learning

Supervised Learning algorithms build a model starting from a set of labelled training data, so that through the iterative optimization of an objective function the algorithm can learn to correctly predict the label associated to new inputs. The goal of a supervised learning algorithm is to be able to correctly determine the labels of unseen data, which could be achieved by reasonably generalizing the model during the training phase.
Many different supervised learning algorithms exist, and there does not exist a

single learning method that performs better on all supervised learning problems (No free lunch theorem), which means the choice of the algorithm must be performed on a single case basis, by considering issues such as the bias-variance tradeoff (flexibility = low bias, high variance -> underfitting).

Unsupervised Learning

Unsupervised Learning is a type of task used to find structures in unlabelled data, such as clustering, anomaly detection, dimensionality reduction. Unsupervised learning algorithms try to identify patterns in the data and do not rely on a comparison with a ground truth, like in the case of supervised learning. This is particularly helpful in many real case scenarios in which data labelling is often an extremely time-intensive or simply non-feasible task, while also giving greater freedom by trying to identify previously undetected patterns.
These techniques often require a much greater amount of training data and converge slowly to acceptable performance. Unsupervised learning algorithms are also potentially more susceptible to anomalies in the data that might be considered irrelevant or categorized as erroneous by a human, but are assigned undue importance by the algorithms themselves.

Semi-supervised Learning

This category of algorithms falls in between the previous two. The data used with these algorithms is only partially labelled: the small portion of data provided with a label, when used in conjunction with unlabelled data, allows to greatly improve the learning accuracy of the models.

Reinforcement Learning

Reinforcement learning is an area of ML concerned with how agents should take actions in an environment so as to maximize some cumulative reward. This field is particularly studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms. In ML, the environment is typically represented as a Markov decision process. Many reinforcement learning algorithms use dynamic programming techniques. Reinforcement learning algorithms are often used in scenarios such as autonomous driving or training an AI to play a game against a human opponent.

2.3 The Perceptron

In order to properly introduce Neural Networks (NNs), which are the model this thesis will focus on, it is necessary to explain the role of the **perceptron**.

The perceptron is a supervised learning algorithm used for binary classification: it is a linear classifier that performs predictions based on a linear predictor function by combining a set of weights with a feature vector. This algorithm was invented by Frank Rosenblatt in 1958[12] and it was first intended to be a physical machine rather than a program, to be used for image recognition with 400 photocells randomly connected to the neurons with their weights encoded in potentiometers.

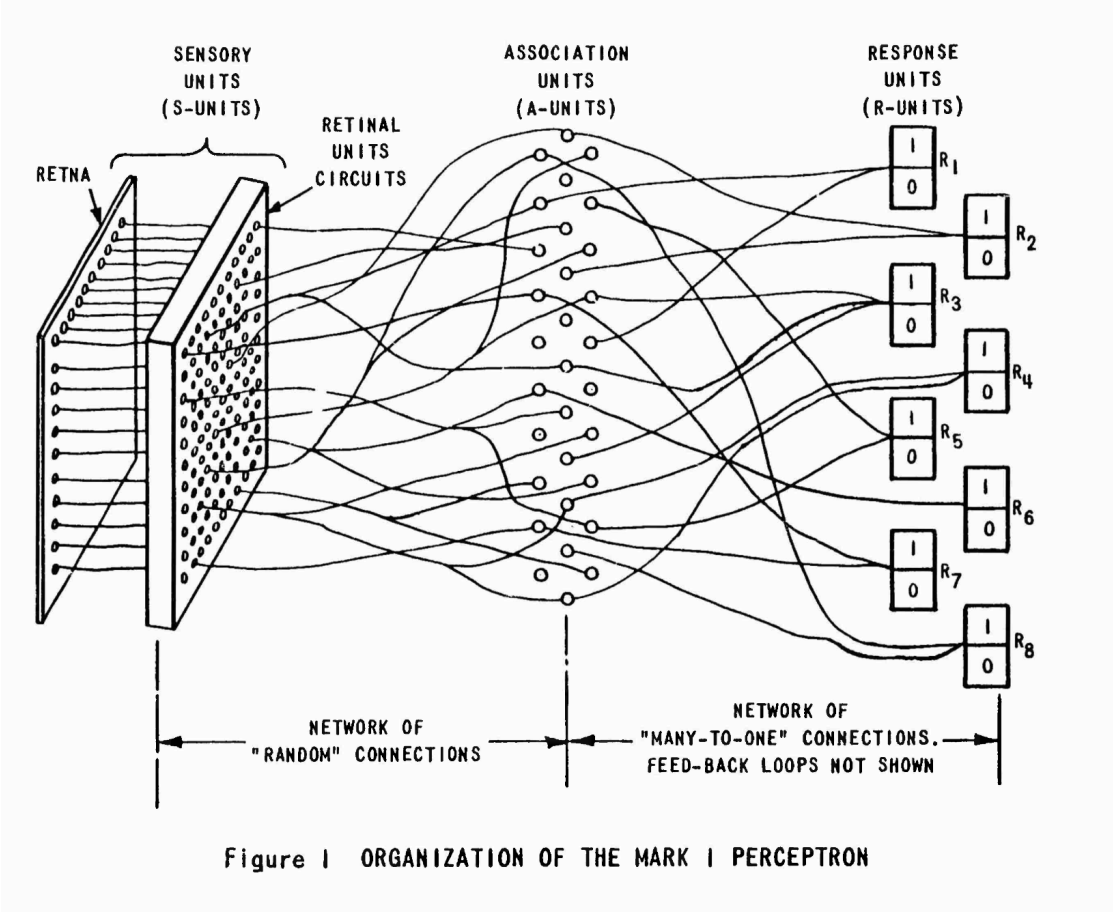

Figure 2.12: MARK I perceptron schema [13]

The perceptron is built around the simplified model of a biological neuron and, in the context of neural networks, plays the role of artificial neuron. It is also known as single-layer perceptron (SLP) to differentiate it from the multi-layer perceptron

(MLP) which, contrary to what it may seem, is a general term indicating a more complex neural network comprising 3 or more layers of neurons. The single-layer perceptron can be considered as the simplest example of feedforward neural network.

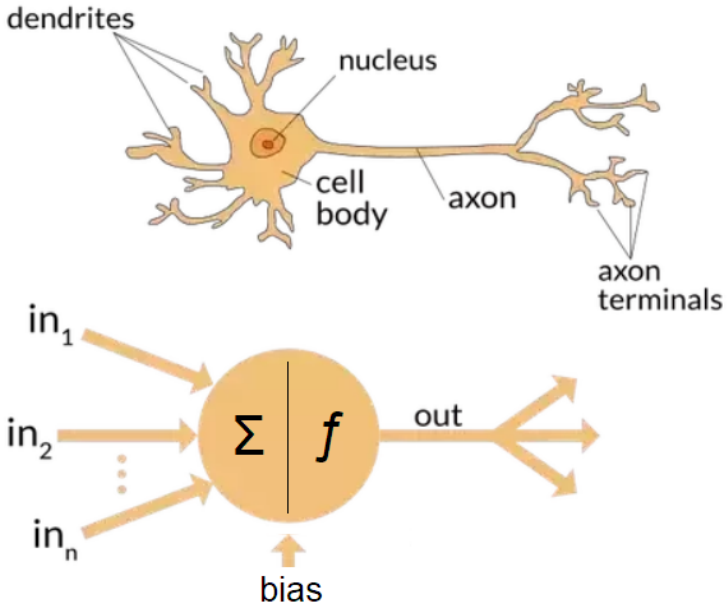

Figure 2.13: Comparison between a biological neuron and the perceptron [14]

Although the perceptron provided initial promising results, it was promptly clear that it could not be trained to recognise many classes of patterns. This caused the research in the field to stagnate for many years, before it was recognised that a MLP had greater capabilities.
During their training phase, perceptrons will gradually learn a linear separation in the data: if the data is not linearly separable, the training task will fail and the perceptron will not converge to a solution where all data is correctly classified. For a classification task with some step activation function, a single node will have a single line dividing the data points forming the patterns. More nodes can create more dividing lines, but those lines must somehow be combined to form more complex classifications. A second layer of perceptrons, or even linear nodes, are sufficient to solve a lot of otherwise non-separable problems.

2.3.1 Learning and convergence

The objective of the perceptron algorithm is to learn what is called a threshold function, which maps the input data x to an output value $f(\mathbf{x}) = \varphi(\langle \mathbf{w}, \mathbf{x} \rangle + b)$

such that:

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \langle \mathbf{w}, \mathbf{x} \rangle + b > 0, \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

where $\langle \mathbf{w}, \mathbf{x} \rangle$ indicates the inner product between the weights of the perceptron w and input data x, b is the bias, which shifts the decision boundary from the origin, without altering its orientation, and φ is the activation function, i.e. the Heaviside step function $H(x)$. The bias b effectively represents the threshold which determines the activation of the artificial neuron.
The value of $f(x)$, either 0 or 1, is used to classify the data in a binary classification problem.
In the case of single-layer perceptron, the learning algorithm operates according to the following steps:

1. Initialize weights and bias, i.e. $w(0) = 0,\ b(0) = 0$

2. For each example i in the training dataset:

   if $\mathbf{y_i}[\langle \mathbf{x}, \mathbf{w} \rangle + b] \leq 0$ then

   $w(t+1) = w(t) + \eta \mathbf{x_i}$, where η is the learning rate with $\eta \in (0,1]$
   $b(t+1) = b(t) + \mathbf{y_i}$

   define Y

Since the perceptron is a linear classifier, it will never converge to a state in which all the input vectors are classified correctly if the training set is not linearly separable, which means that the two classes of the data must be separable by means of an (n-1)-dimensional hyperplane. Instead if the data is linearly separable, the algorithm is guaranteed to converge in a number of steps that is $O(R^2/b^2)$. define R For this reason, if linear separability is not known a priori, a variant should be used, where a small number of misclassified data is permitted to guarantee convergence.

What follows is an example of the updates that progressively determine the position and rotation of the decision boundary.

It is important to be aware that given two separable classes, there exist infinitely many boundaries dividing them, and the perceptron algorithm is not aware of which decision boundary represents the best one. This problem was later solved by the linear Support Vector Machine (linear SVM) model in which the algorithm chooses the decision boundary that maximizes the margin (i.e., distance) between the two classes and the boundary itself.

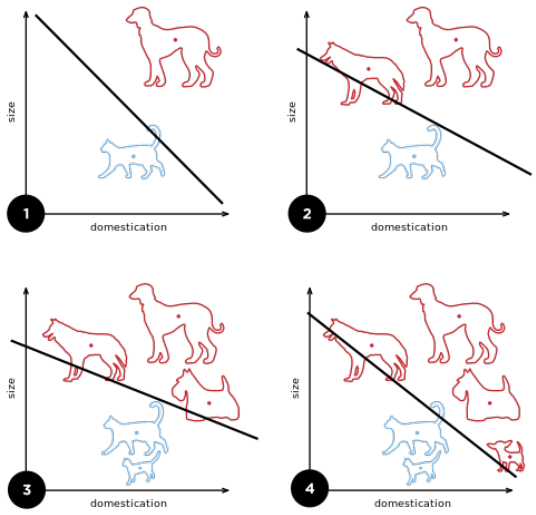

Figure 2.14: Progression of the definition of a decision boundary [15]

2.4 Neural Networks

Neural networks (NNs), also known as artificial neural networks (ANNs), are complex computing models which are based on the original idea of the perceptron and consist of a collection of connected units, called nodes or artificial neurons, which loosely simulate the role of neurons in a biological brain. There exist two main categories of ANNs, based on the differences in the connections between nodes of different layers: feedforward neural networks (FFNNs), in which connections and neurons effectively form a directed acyclic graph and nodes of a layers are only connected to nodes of the next layer; and recurrent neural networks (RNNs), in which connections between nodes of the same or previous layers are allowed, creating loops that can feed information back into the network. In the context of Neural Networks, data is treated as tensors, which are practically equivalent to n-dimensional arrays.

2.4.1 Learning

During the learning process the network adapts to better handle a given tasks by learning from samples. Learning involves adjusting the weights and biases of the network to improve the accuracy of the produced results: this is done by minimizing errors, until the network is not able to usefully improve in reducing the error rate. This parameters updating procedure happens by first defining a cost function that is evaluated periodically during learning: as long as its output continues to decline,

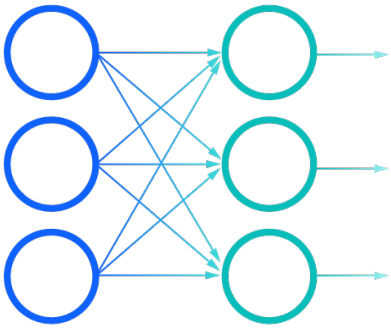

Figure 2.15: Schema of a FFNN [16]

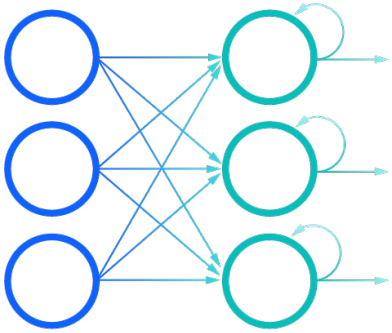

Figure 2.16: Schema of a RNN [16]

the learning process continues. The cost is frequently defined as a statistic, and some examples include the Mean Squared Error (MSE), Binary Cross Entropy (BCE), etc. The choice of the loss function needs to happen on a per case basis, according to the task that has to be learned.
The MSE is a measure of the difference between two quantities, which in a learning scenario are the predicted values and the ground truth.

$$MSE_i = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i; W))^2 \tag{2.4}$$

where $f(x_i; W)$ represents the values predicted by the network with weights W taking as input data x_i, and y_i indicates the ground truth, or label.

The Cross Entropy loss function, also called Binary Cross Entropy in binary classification tasks, is a measure of the difference between two distributions.

$$BCE_i = -\frac{1}{n}\sum_{i=1}^{n} y_i \log\left(f(x_i; W) + (1 - y_i)\log\left(1 - f(x_i; W)\right)\right) \tag{2.5}$$

Once a loss function $\mathcal{L}$ is defined, the goal of the learning task is to find the set of parameters, weights W^* and biases b^*, that minimize the function:

$$W^* = \arg\min_{W} \mathcal{L}(W) \tag{2.6}$$

A common method for minimizing the loss function is called gradient descent. To better understand the gradient descent algorithm, consider the following hypothetical representation of a simple loss function $\mathcal{L}(w_0, w_1)$, where each pair (w_0, w_1) is associated to a value of the loss function.

The idea behind the algorithm is that, starting from any point in the loss function space, one could follow the direction that allows to locally minimizes

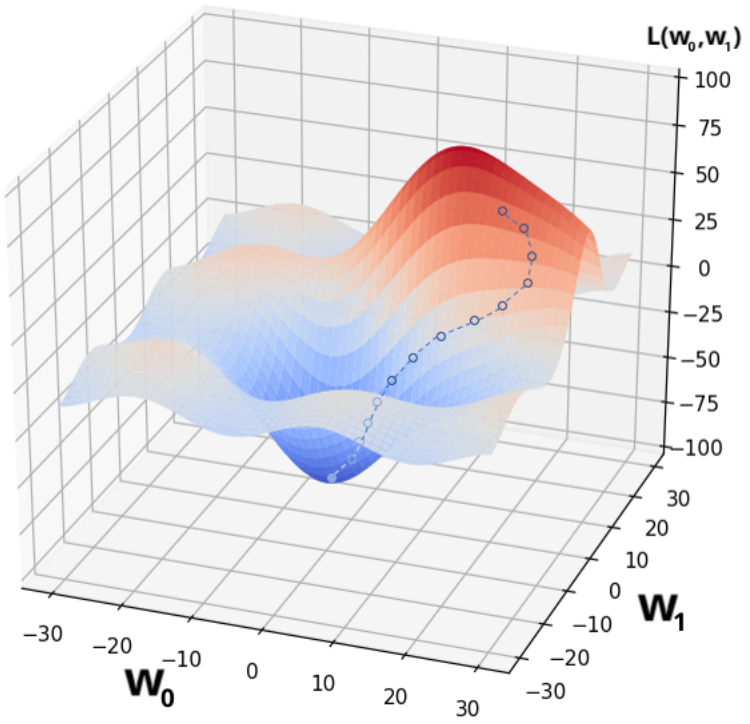

Figure 2.17: Example of gradient descent [17]

the function $\mathcal{L}(w_0, w_1)$ by following the direction of the most rapidly decreasing gradient $-\nabla\mathcal{L}$. While gradient descent is a fairly basic optimization algorithm, in principle it operates similarly to other more complex ones. The gradient descent algorithm is not particularly efficient in avoiding local minima, which would cause the learning process to stagnate and stop improving the model: for this reason there exist other methods that take into account the possibility of getting stuck in these local minima, such as Stochastic Gradiend Descent, Adam, AdaGrad.
Once an optimization algorithm is defined, the gradient is computed by means of what's commonly called backpropagation (BP). The BP algorithm computes the gradient as a function of the network's weights via the chain rule. The chain rule is a formula that expresses the derivative of the composition of two differentiable functions in terms of the derivatives of those two functions. Consider a simple Multi Layer Perceptron with a single hidden layer h, having as input x and output $\hat{y}$, with weights w_0 between the input layer x and hidden layer h, and weights w_1 between the hidden layer h and the output layer $\hat{y}$. Since improving the network is effectively achieved by updating the weights, it's necessary to compute the gradient with respect to each set of weights. While the gradient with respect to w_1 can be

directly calculated as:

$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_1} \tag{2.7}$$

the inner weights can not. This is where the chain rule comes into play, as it allows us to express the gradient in terms of w_0 as:

$$\frac{\partial \mathcal{L}}{\partial w_0} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_0} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h} \frac{\partial h}{w_0} \tag{2.8}$$

In this way, it's possible to calculate all gradients, from output to input layers. An important hyperparameter used in the learning phase is the learning rate. The learning rate defines the size of the steps that the model takes to adjust for errors. Taking as an example the previously explained gradient descent, one could think of the learning rate as the distance between two updates of the loss function.
A high learning rate shortens the training time, but with lower ultimate accuracy, while a lower learning rate increases the time required for training the network, but with the potential for greater accuracy. It is common to use schedulers which adaptively change the value of the learning rate to start the learning process by correcting with large steps, and systematically decreasing it to improve accuracy once the model has reached a stagnating accuracy. Some learning rate schedulers also employ the concept of momentum: it's an hyperparameter that allows to balance the weight update by considering both the current and previous gradient, in order to maintain a dependence from previous updates. A momentum close to 0 neglects the previous gradient, while a value close to 1 emphasizes the last change.

2.4.2 Deep Learning and Deep Neural Networks

Deep Learning (DL) is a class of ML methods based on neural network models. These learning methods can be supervised, semi-supervised or unsupervised. The term Deep Learning generally refers to the use of multiple layers in the neural network: research quickly showed that linear perceptrons were not suitable to be used in universal classification task, but a network with non-polynomial activation functions with multiple hidden layers between the input and output ones was. This led to an increase in the depth (i.e., number of hidden layers) of the networks, also thanks to the extreme increase in computational power achieved in recent years.
Deep Learning models are nowadays extensively used in fields including computer vision, speech recognition, natural language processing, bioinformatics, medical image analysis, board game AI, where they have produced results comparable to and in some cases surpassing human expert performance, as in the case of AlphaGo, Google's AI trained to play the game of Go.

2.5 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a specialized type of feedforward NN usually employed in computer vision tasks (i.e., image classification, image segmentation), recommender systems and natural language processing to name a few. CNNs took their inspiration from biological processes as the connectivity pattern between neurons resembles the organization of neurons of the animal visual cortex. CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns to optimize the filters for feature extraction through automated learning: this independence from prior knowledge and redundancy of human intervention are among the most important advantages of these models.
In the following sections we will focus on the principal components of CNN architectures.

2.5.1 Convolutional Layer

Convolutional layers are the fundamental building block of Convolutional Neural Networks. These layers usually take as input a 4-dimensional tensor (number of inputs, height, width, channels), apply a convolution operation by means of a filter and pass their result to the next layer. The filter, whose size is parametrically specified, slides over the entirety of the input tensor with a given stride (i.e., step size when moving the filter), performing a convolution operation on the portion of the input tensor covered by the filter.
The convolution operation between two matrices A, B of the same size can be defined as $\sum_{i=1,j=1}^{m,n} A_{i,j}B_{i,j}$. This operation generates a feature map, also called activation map, whose size can be derived from the convolutional layer parameters and the input tensor size: the number of output tensors will be the same as the number of input tensors, the output width will be determined by the filter width, the horizontal stride HS and eventual padding P as $W_{out} = (\frac{W_{in}+2P-K_{width}}{HS}) + 1$, the output height will be determined similarly to the width with them being equal in case of square input tensors, and the number of filters determines the depth, or number of channels, of the activation map.

Although fully connected feedforward neural networks can be used to learn features and classify data, they are generally impractical for larger inputs such as high resolution images because they would require a very high number of neurons, even in a shallow architecture, due to the large input size of images, where each pixel is a relevant input feature. For example, a fully connected layer for a relatively small-sized image of size 100 x 100 has 10,000 weights for each neuron in the second layer. The convolution reduces the number of free parameters, allowing the network to be deeper.

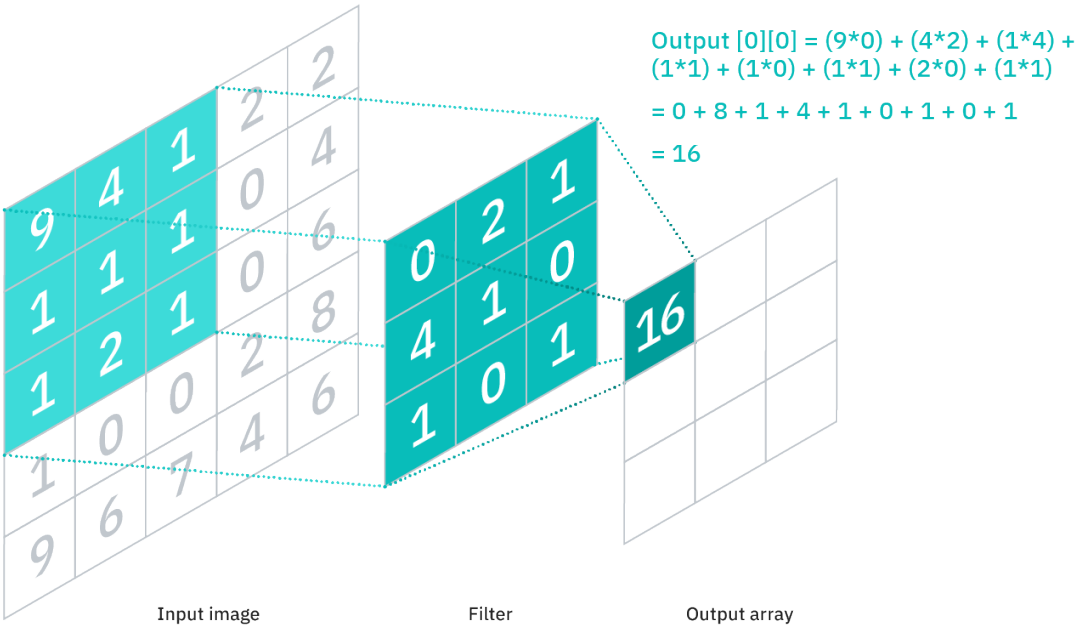

Figure 2.18: Convolutional layer [18]

For example, regardless of image size, using a 5 x 5 filter, each with the same shared weights, requires only 25 parameters to be learned. Using regularized weights over fewer parameters avoids the vanishing gradients and exploding gradients problems seen during backpropagation in traditional neural networks: when dealing with deep architectures, the gradient can easily converge to 0 or explode to infinity. Convolutional neural networks are also ideal for grid-like data such as images, because spatial relations between separate features are taken into account during convolution and/or pooling.

2.5.2 Activation Function

After a convolutional layer, usually a non-linear activation function is applied. A common non-linear activation function is the Rectified Linear Unit (ReLU). This activation function is a piece-wise activation function $f(\mathbf{x})$ that brings to 0 negative values, without varying positive ones.

$$f(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} < 0, \\ \mathbf{x} & \text{if } \mathbf{x} \geq 0 \end{cases} \tag{2.9}$$

One of the biggest downsides of the ReLU activation function is the risk of neurons "dying": in case of exclusively negative values, they all are set to 0 rendering

the neuron incapable of propagating information. To solve this issue it's possible to use a variant, the leaky ReLU which assigns a small, negative value to negative values instead of setting them to 0. For example:

$$f(\mathbf{x}) = \begin{cases} 0.01\mathbf{x} & \text{if } \mathbf{x} < 0, \\ \mathbf{x} & \text{if } \mathbf{x} \geq 0 \end{cases} \tag{2.10}$$

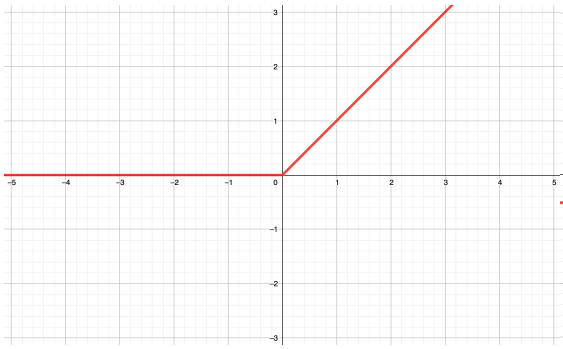

Figure 2.19: ReLU

Figure 2.20: Leaky ReLU

Other common activation functions are the Sigmoid $\sigma(\mathbf{x}) = (1 + e^{-\mathbf{x}})^{-1}$ and the hyperbolic tangent $\tanh(\mathbf{x})$.

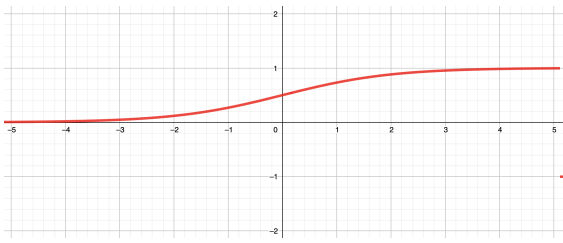

Figure 2.21: Sigmoid

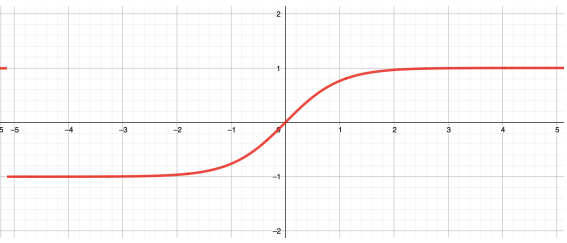

Figure 2.22: Hyperbolic tangent

2.5.3 Pooling Layer

Pooling layers are used to reduce the dimensionality of the data by combining a number of output neurons of a layer into a single neuron to be passed to the next layer. Pooling could be either local, working on a sub-area of the tensor, or global when working on its entirety. Similarly to how the filter slides over the input tensor, during pooling a square matrix slides over a tensor to reduce its covered area to a single value. The two most common pooling methods are Max Pooling and Average Pooling.
In Max Pooling, only the maximum value inside the pooling window is conserved. This is done to preserve the most relevant information, discarding what should be less important.
In Average Pooling, the average of the values inside the pooling window is preserved, to keep bits of information of the whole input tensor.

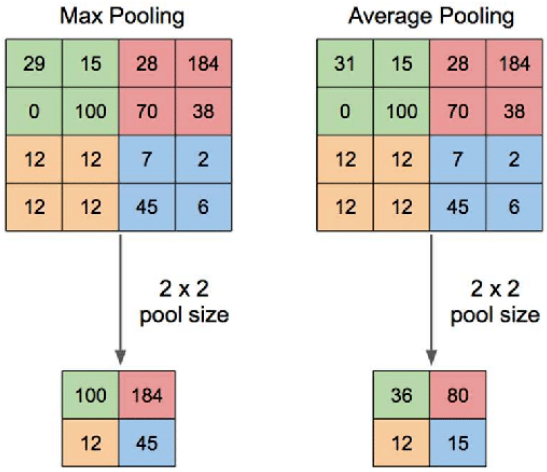

Figure 2.23: Examples of Max and Average Pooling [19]

2.5.4 Batch Normalization Layer

Most neural networks perform best when data is normalized. The problem with CNNs is that after being passed through a convolutional or fully connected layer, tensors are no longer normalized, causing the learning process to slow down and negatively influence the accuracy of the model.
To mitigate this issue, Batch Normalization layers are often used in most NN architectures, re-centering and re-scaling tensors.

2.5.5 Fully Connected Layer

Similarly to traditional multi-layer perceptrons, fully connected layers connect every neuron of a layer to every neuron of another layer. These layers are usually placed just before the output nodes of models used for image classification, as they are effectively used to learn a (usually) non-linear function in the feature space described by the previously placed convolutional layers as to divide the data according to the extracted features.

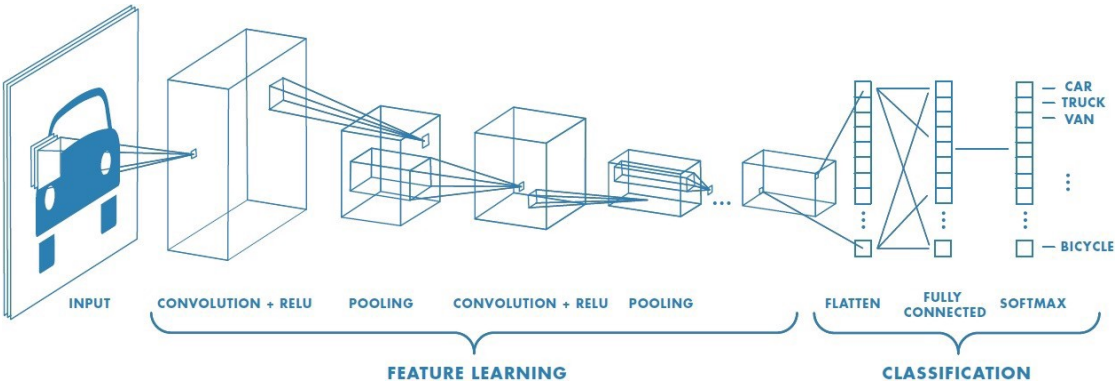

Figure 2.24: Example of a Convolutional Neural Network architecture for image classification [20]

2.6 Residual Neural Networks

Residual Neural Networks (ResNets) are another important class of feed forward neural networks. ResNets are inspired by constructs known from pyramidal cells in the cerebral cortex: they utilizing skip connections, or shortcuts, to jump over some layers in the architecture. Typical ResNet models are implemented with double or triple layer skips, usually over non-linear activation layers and batch normalization. The two main reasons to add skip connections are to avoid the problem of vanishing gradients, or to mitigate the Degradation problem. The latter indicates an issue that manifests when adding more layers to a suitably deep model, leading to higher training error. During training, the weights adapt to mute the upstream layer and amplify the previously-skipped layer. In the simplest case, only the weights for the adjacent layer's connection are adapted, with no explicit weights for the upstream layer. This works best when a single nonlinear layer is stepped over, or when the intermediate layers are all linear.

Skipping speeds learning by reducing the impact of vanishing gradients, as there are fewer layers to propagate through. The network then gradually restores the skipped layers as it learns the feature space.

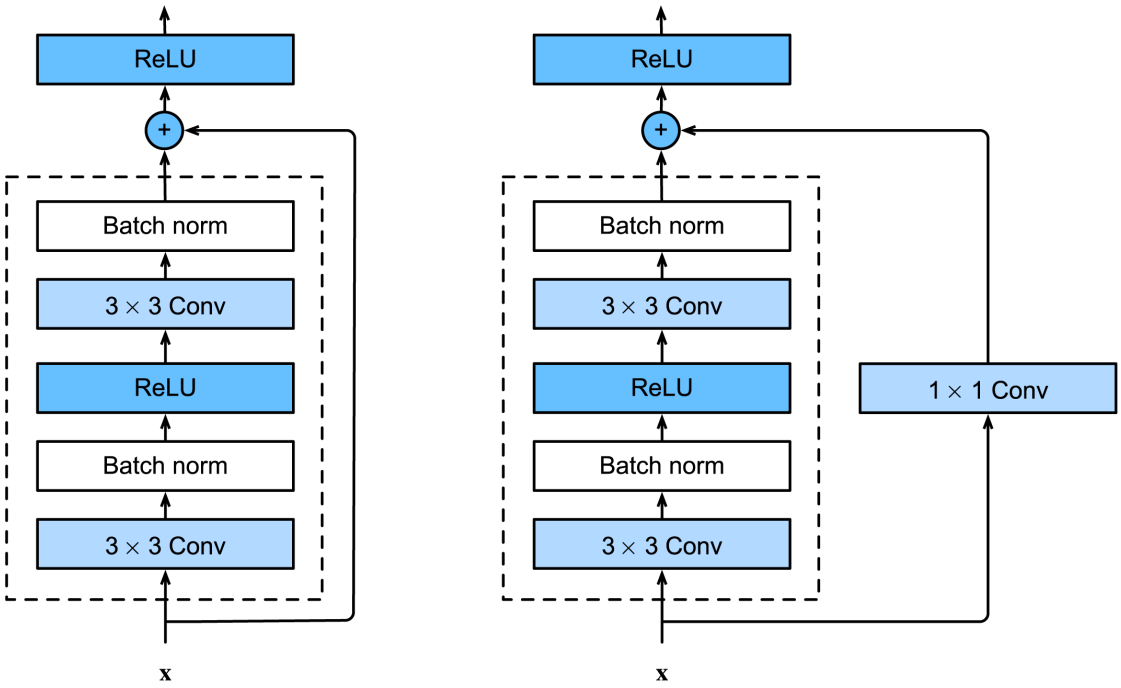

Figure 2.25: Examples of skip connections in a ResNet architecture [21]

2.7 Generative Adversarial Networks

Generative Adversarial Networks (GANs) is an architecture first proposed by Goodfellow et al.[22] in 2014. They are a framework in which two neural networks, a generator G and a discriminator D, are in competition with each other in a zero-sum game.
The generative network G creates candidates which are to be evaluated from the discriminator D, and the training process continues until the game between the two networks reaches an equilibrium in which the discriminative network D is no longer able to accurately distinguish between samples from the original dataset and samples created by the generator.
Usually, G learns to map from what is known as a latent space to a data distribution of interest, which should be similar to that of the training set; while D distinguishes candidates produced by the generator from the true data distribution.
The generative network's training objective is fool the network D by producing candidates that the network D thinks belong to the original dataset, as to increase the error rate of the discriminator.

The discriminator needs to be first trained to be presented with samples from

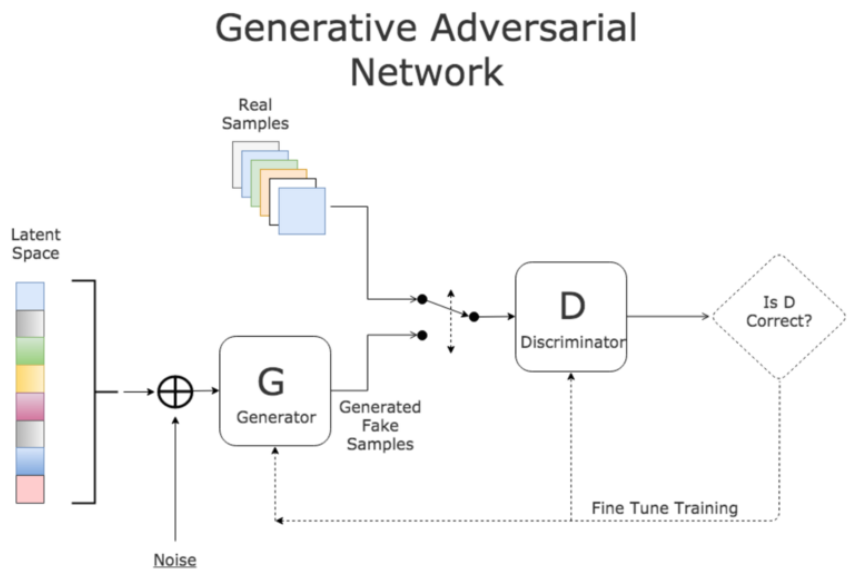

Figure 2.26: Schema of the learning process of a GAN architecture [23]

the training dataset, until it reaches an acceptable accuracy. The generator trains based on whether it succeeds in fooling the discriminator. Typically the generator is provided with randomized input that is sampled from a predefined latent space (e.g. a multivariate normal distribution). Subsequently, candidates synthesized by G are evaluated by the network D. The two networks are independently trained via backpropagation so that the generator is able to produce better samples, while the discriminator becomes more skilled at recognizing synthetic samples.
When used for image generation tasks, the generator is usually a deconvolutional neural network, while the discriminator is a convolutional neural network. GAN architectures have increasingly been the focus of various research in the computer vision field, as they have demonstrated to be particularly suitable in tasks that require the generation of realistic looking images.

2.7.1 The generator

At the beginning of the training process, the generative network G takes as input fixed-length random vectors casually sampled from a Gaussian distribution that the network uses to generate a sample in the domain of the dataset to imitate, also called destination domain. After a successful training procedure, the points of the multi-dimensional vector space should match points in the destination domain, effectively representing a compressed representation of the dataset's probability distribution. As previously stated, this space is called latent space, which is a space

composed by latent variables.
Latent variables represent important non-directly observable variables in a domain. Generators in a GAN architecture can take advantage of a latent space by sampling from it to generate sample which resemble data coming from a distribution of choice. After having trained the network in an adversarial fashion, it can be used to generate new samples that, even if close to the data distribution used for training, are not copies but original products of the network.

2.7.2 The discriminator

The discriminative network D simply takes as input both samples from the original dataset and newly generated data coming from the generator G. Its role is to determine whether a data point comes from a distribution or the other, usually labelled 0 and 1, in a binary classification setting. Once the training process is completed, the discriminator network does not serve additional purposes in the generation process of the network G, but it still can be used in other tasks that require extracting features of the dataset used for training, or of the new generated data.

2.7.3 Training a GAN architecture

The two networks are trained simultaneously in a supervised setting: the generator creates samples and improves its generative process by comparing the products to data coming from the dataset of choice, while the discriminator classifies the data coming from the two sources and trains to improve its classification of the real samples. Generator and discriminator are effectively in competition with each other: while the first aims at deceiving the discriminator, the latter continually trains to recognize when it's being deceived by the generator. The competition between the two can be described as a zero sum-game.
In game theory, a zero-sum game is a mathematical representation of a situation in which the advantage that a player gains is lost by the other. In a GAN architecture, the two networks are the players, and the zero-sum means that when the discriminator successfully identifies the class to which a sample belongs, it is rewarded by maintaining its parameters, while the generator is penalized with an update of its weights, and vice versa. The objective of the whole architecture is to reach an equilibrium state in which the generator consistently creates convincing samples, and the discriminator is not able to accurately distinguish the two classes. In particular, given a latent space z with an a priori distribution $p_z(z)$:

- G can be described as a differentiable function $G(z; \Theta_g)$ that produces data according to a distribution p_g, where Θ_G represents the model's parameters.

- D can be described as a differentiable function $D(z; \Theta_D)$ that produces as output the probability p_{data} that input data x belongs to the training dataset, where Θ_D are the model's parameters.

The goal of G is to imitate as close as possible p_{data}, so that D struggles with distinguishing p_G from p_{data}. The learning procedure of a classic, or vanilla, GAN consists of optimizing the following min-max problem:

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{x \sim p_z(z)} [\log (1 - D(G(z)))] \tag{2.11}$$

Training is effectively achieved by updating the two networks' weights via backpropagation, in a similar fashion to what has been described previously.

Chapter 3

State of the Art analysis

In this chapter we will investigate the current State of the Art (SotA) solutions to the Super-Resolution problem, focusing on neural network-based methods which currently offer the best performance thanks to their computational power and complexity. As previously mentioned, Deep Learning-based solutions are the main topics of research in the super resolution field. Deep Learning architectures allow to create extremely complex models that, through a learning procedure, are able to produce far better results compared to the traditional interpolation-based techniques showcased in the previous chapter.
The State of the Art models showcased in this chapter are all Deep Learning-based models,

3.1 Deep Convolutional Neural Network-based Super-Resolution methods

3.1.1 Super Resolution Convolutional Neural Network

The Super Resolution Convolutional Neural Network (SRCNN) is a fully convolutional model and it's one of the first important milestones in Deep Learning applied to Super Resolution problems introduced by Dong et al. [24].

The idea behind this model is based on the intuition that prior SotA, consisting mostly of methods adopting sparse-coding-based strategies, have a pipeline that loosely resembles a deep CNN architecture.

Since only a selected number of steps of previous methods were studied for optimization and these methods were rarely optimized as a unified pipeline, with the SRCNN model the aim was to consolidate the learning process by developing a CNN able to generate an end-to-end mapping between low-resolution and high-resolution images while proposing a simple yet accurate model able to compete

with already existing methods with little to no pre/post processing needed. The SRCNN network consists of 3 layers, whose main roles are patch extraction and representation, non-linear mapping and reconstruction.

Given a low-resolution image, it is first upscaled using bicubic interpolation. The first layer is responsible for extracting overlapping patches from the image and representing each patch as a high-dimensional vector that comprises a set of feature maps. Each of these high dimensional vectors is further mapped into another high dimensional vector by the second layer. The new vectors comprise another set of feature maps, which conceptually represents the patches of a high-resolution image. The final layer, responsible for the reconstruction, aggregates the previously generated high-resolution patch representations to create a final high resolution image which is expected to be the ground truth image.

The authors show that thanks to its lightweight structure the SRCNN model demonstrates state-of-the-art restoration quality, is able to achieve fast speed for practical online usage, functions on three channels simultaneously and performs better than the SotA methods chose for comparison.

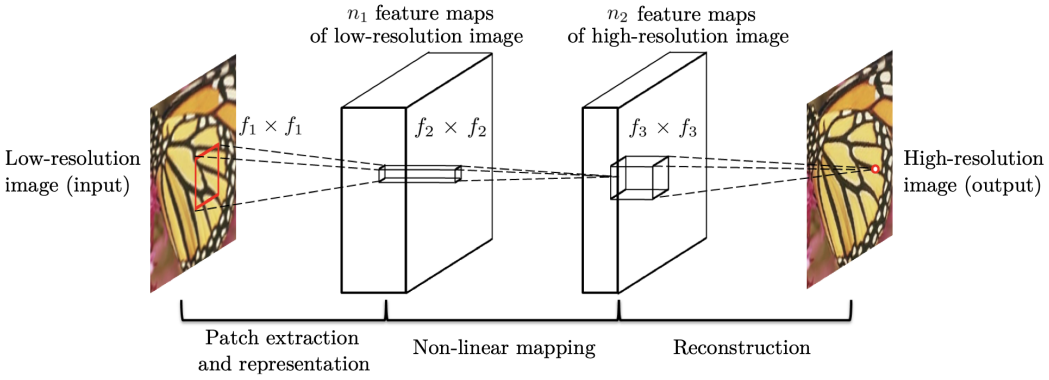

Figure 3.1: Simplified overview of the SRCNN architecture [24]

3.1.2 Enhanced Deep Super Resolution network

The Enhanced Deep Super Resolution (EDSR) network is a model proposed by Lim et al.[25] to improve on the SRResNet architecture developed by Ledig et al.[26]. While the SRResNet model was proposed as a part of the SRGAN model, again proposed in the same paper by Ledig et al.[26], where it performs the role of generative network.

The idea of using deeper networks, employing skip connections and residual blocks stems from prior studies showing that deeper network architectures can lead to a substantial increase in accuracy as the depth allows the model to create

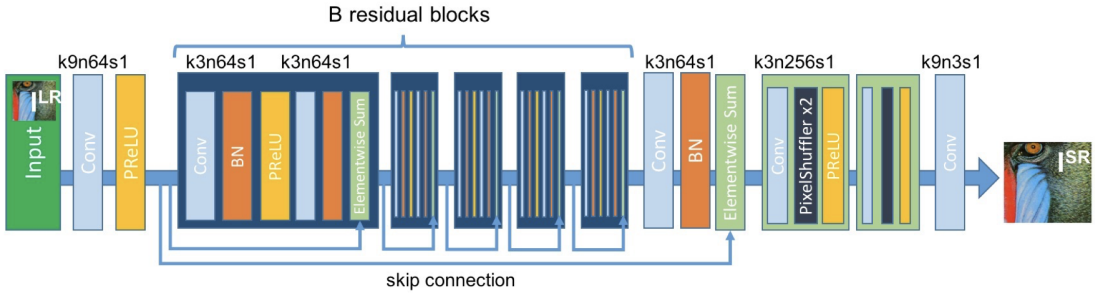

Figure 3.2: Overview of the SRResNet architecture [26]

mappings of high complexity, albeit at the cost of substantial increase in the training difficulty. To alleviate the difficulty in training of such deep networks, batch- normalization is often used to counteract the internal co-variate shift. Deeper network architectures had also been shown to increase performance for Single Image Super Resolution tasks. Skip connections relieve the network architecture of modeling the identity mapping that is trivial in nature, however, potentially non-trivial to represent with convolutional kernels. In the context of SISR it was also shown that learning upscaling filters is beneficial in terms of accuracy and speed. This is an improvement over the SRCNN model where bicubic interpolation is employed to upscale the LR observation before feeding the image to the CNN.

The EDSR architecture aims to improve the already well-performing SRResNet by simplifying the architecture removing unnecessary layers and investigates on the idea of scale-independent training, by analyzing the impact of transfer learning between models used for different SR scales and also proposing a multi-scale variant of the architecture.

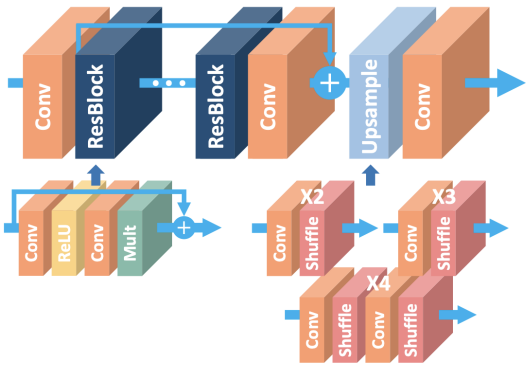

Figure 3.3: Simplified overview of the EDSR architecture [25]

Batch normalization layers normalize the features, getting rid of range flexibility

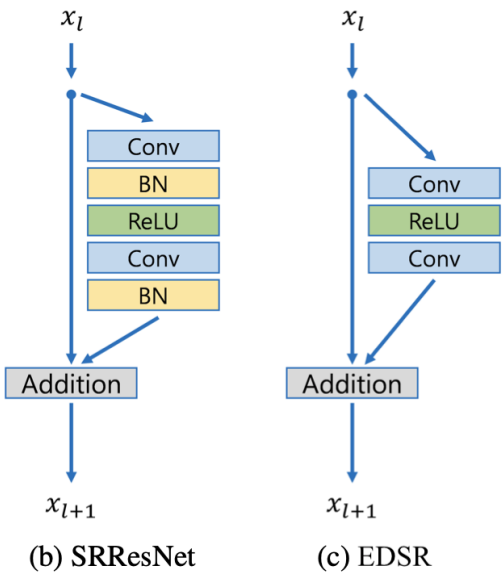

Figure 3.4: Difference between SRResNet and EDSR residual blocks [25]

from networks. By removing them, Lim et al.[25] experimentally show that they are able increases the performance substantially, saving $\approx 40\%$ of memory usage during training, compared to SRResNet.

3.2 Generative Adversarial Network-based Super-Resolution methods

3.2.1 Super Resolution Generative Adversarial Network

In the SRResNet paper, Ledig et al. [26] propose a GAN-based approach with the Super Resolution Generative Adversarial Network (SRGAN). They define a classical GAN architecture using the SRResNet model as the generator G_{θ_G} and define a discriminator D_{θ_D} to solve the min-max adversarial problem:

$$\min_{\theta_G} \max_{\theta_D} \mathbb{E}_{I^{HR} \sim p_{train}(I^{HR})} \left[\log D_{\theta_D}(I^{HR}) \right] + \mathbb{E}_{I^{LR} \sim p_G(I^{LR})} \left[\log \left(1 - D_{\theta_D}(G_{\theta_G}(I^{LR})) \right) \right] \tag{3.1}$$

The general idea is to train a generative model G with the goal of fooling a differentiable discriminator D that is trained to distinguish super resolved images from real images. With this approach, the generator can learn to create images that are highly similar to real ones, thus difficult to classify by D. This encourages

perceptually superior solutions residing in the subspace of natural images, in contrast to solutions obtained by minimizing pixel-wise error measurements, such as the MSE. The generator architecture can be seen in *fig. 3.2* To discriminate real HR images from generated SR samples we train a discriminator network, which can be seen in *fig. 3.5*

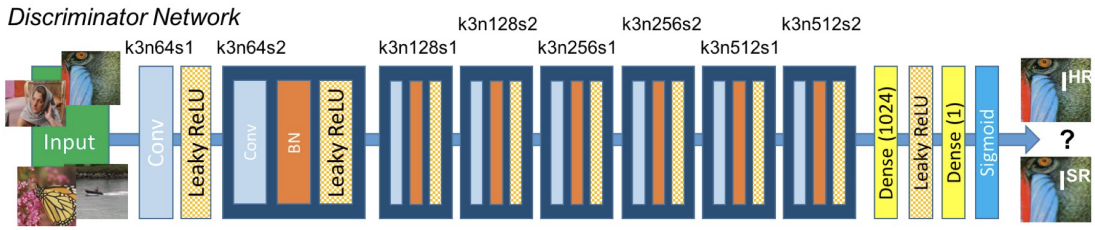

Figure 3.5: SRGAN discriminator

The SRGAN architecture introduces a perceptual loss function, critical to the performance of the generator, formulated as the weighted sum of two losses:

- The content loss which, instead of the classical pixel-wise mean square error (MSE) loss, is a VGG loss based on the activation layers of a pretrained VGG-19 network.
- The adversarial loss, to drive the network towards a more realistic generation of images.

3.2.2 Enhanced Super Resolution Generative Adversarial Network

The Enhanced Super Resolution Generative Adversarial Network (ESRGAN) by Wang et al. [27] proposes a set of improvements to the already aforementioned SRGAN model, by proposing some minor changes to the basic blocks of the SRGAN generator, discarding the classical discriminator in favor of a relativistic one, and by introducing a new metric, the perceptual index, which evaluates the perceived quality of the image rather than relying solely on distortion measures such as PSNR and SSIM.

The modifications to the generator network G include:

- the removal of all batch normalization layers, which has proven to increase performance and reduce computational complexity in different PSNR-oriented tasks, such as super resolution

the replacement of the SRResNet basic block with the Residual-in-residual Dense Block (RRDB), which combines multi-level residual and dense connections. More

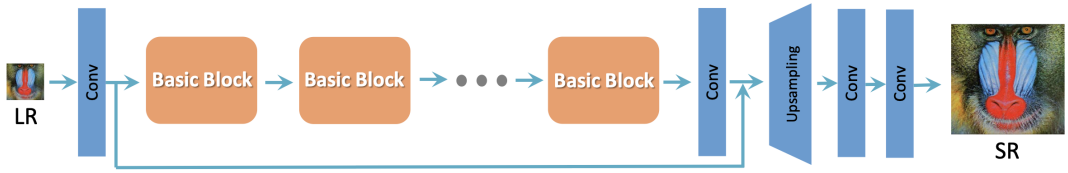

Figure 3.6: Simplified overview of ESRGAN's generator architecture [27]

layers and connections were added to create a deeper network, capable of boosting performance.

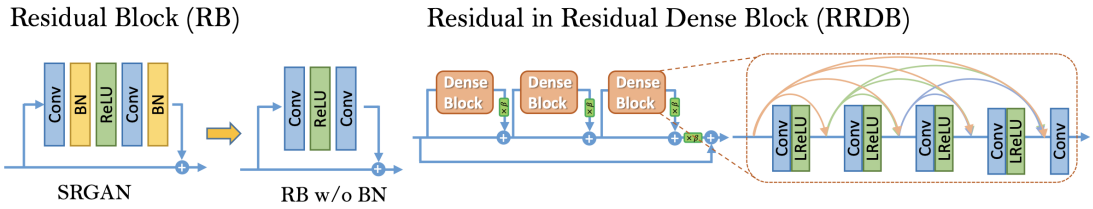

Figure 3.7: Details of the RRDBs of the ESRGAN model [27]

In addition to a modified generator, Wang et al. [27] propose a discriminator based on the Relativistic GAN architecture [28]. While the standard SRGAN discriminator D estimates the probability that an image is real, the proposed Relativistic discriminator D_{Ra} tries to predict the probability that an image x_r is relatively more realistic than a fake image x_f. The relativistic discriminator loss is defined as:

$$L_D^{Ra} = -\mathbb{E}_{x_r}\left[\log D_{Ra}(x_r, x_f)\right] - \mathbb{E}_{x_f}\left[\log\left(1 - D_{Ra}(x_f, x_r)\right)\right] \tag{3.2}$$

while the generator's adversarial loss is symmetrically defined as:

$$L_G^{Ra} = -\mathbb{E}_{x_r}\left[\log\left(1 - D_{Ra}(x_r, x_f)\right)\right] - \mathbb{E}_{x_f}\left[\log D_{Ra}(x_f, x_r)\right] \tag{3.3}$$

where $x_f = G(x_l r)$, $D_{Ra}(x) = \sigma(C(x_r) - \mathbb{E}_{x_f}[C(x_f)]))$, with σ being the sigmoid function, $C(x)$ the non-transformed discriminator output and $\mathbb{E}_{\curvearrowright\circlearrowleft}[\cdot]$ representing the average of all fake data in the mini batch. In this way the generator benefits from both gradients: one from generated data and the other from real data. Wang et al. [27] also propose a more effective perceptual loss by constraining on features before activation rather than after as in the SRGAN model. Thus, the total generator loss appears in the form of:

$$L_G = L_{percep} + \lambda L_G^{Ra} + \eta L_1 \tag{3.4}$$

where L_1 is the content loss evaluated on the 1-norm distance between a super resolved image and its ground truth, and λ, η are coefficients used to balance the different terms.

Here follows a comparison between the four mentioned architectures and the traditional bicubic interpolation method.

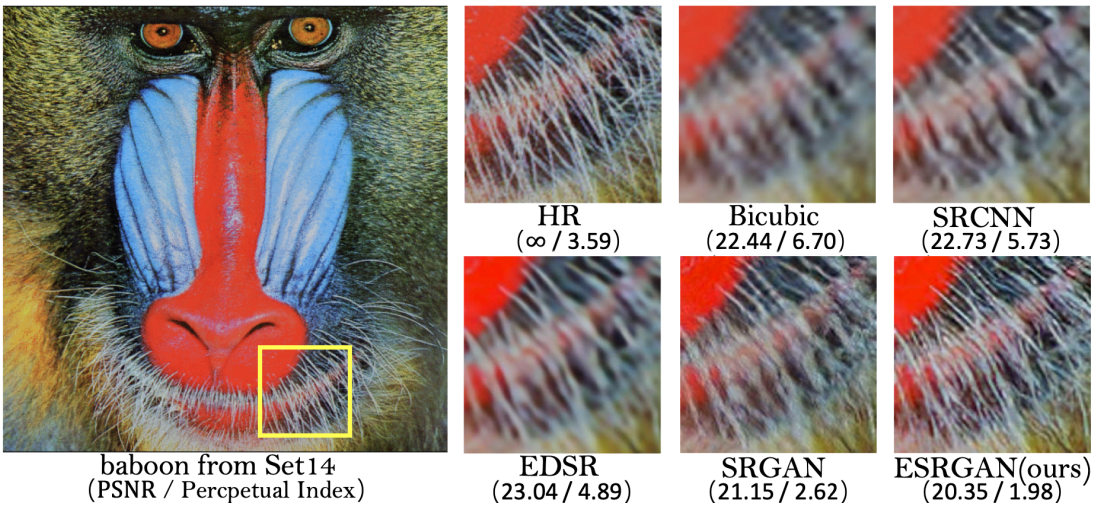

Figure 3.8: Comparison of SR images produced via different techniques [27]

Chapter 4

Super-Resolution techniques applied to satellite images

The contents of this chapter will go over the details of this particular thesis project, from the motivations behind the dataset choice and its creation, to the models analyzed for this specific task and a new proposed architecture, based on the generative adversarial approach presented by Wang et al.[27] with their ESRGAN model.

4.1 The dataset

The choice of the satellite imagery dataset for this specific thesis work was performed considering the possible implications of successfully applying super-resolution techniques to this domain of images. Many algorithms for solving important real-world issues, such as environmental monitoring, natural disaster prediction and land development to name a few, are based on the analysis of satellite images.
Most of these algorithms could benefit from using high resolution satellite imagery, which is usually not open-source nor publicly available due to the high costs involved with both launching and maintaining in orbit satellites that are able to capture such images.
While many private companies, such as Maxar and Planet, can currently provide extremely high resolution data which go up to 46cm per pixel and 50cm per pixel respectively, the Sentinel project is developed and operated by a public organization, the European Space Agency.
The Sentinel-2 mission in particular systematically captures optical imagery over land and coastal waters, thanks to its constellation of two satellites: Sentinel-2A and Sentinel-2B, while a third one, Sentinel-2C is currently being tested in preparation for its launch in 2024. The key characteristics of the Sentinel-2 mission are the

following:

- Systematic global coverage of land surfaces from 56°S to 84°N
- 10-day revisit period under the same viewing angles. At high latitudes, some regions will be observed twice or more every 10 days, but with different viewing angles.
- Spatial resolution of 10m per pixel, 20m per pixel and 60m per pixel. While these resolutions are significantly lower than private competitors, they are still extremely valid in many use cases.
- Free and open data policy

Even at their highest resolution of 10m per pixel, it is quite clear that upscaling images gathered from the Sentinel-2 mission to ground resolutions comparable to the ones offered by the aforementioned private companies poses a non-negligible challenge: most super resolution state of the art models can comfortably operate on a 4x upscaling factor, seldomly analyzing 8x upscaling tasks and rarely exploring the challenges implied with upscaling by a factor of 16, rendering a 20x scaling from the 10m per pixel ground resolution of the Sentinel data to levels comparable to the 46cm per pixel a very open problem. The aim of this thesis project is to propose an architecture capable of producing super resolved images, upscaled by a factor of up to 8x, which are competitive with existing state of the art models. Training super resolution models in a supervised fashion still implies the necessity of a dataset which includes a high resolution ground truth, up to 1.25m per pixel given the 8x upscaling goal.

4.1.1 Analysis of available services

Since the Sentinel-2 data reaches a maximum ground resolution of 10m per pixel, it is quite clear that we need to look elsewhere in order to obtain a dataset which can be used for the supervised training of the model. Several options have been considered for gathering high resolution satellite imagery, specifically Maxar, Planet, Google Maps and Bing Maps from Microsoft.
While all of these providers offer a fairly diversified dataset, both in terms of quality and projections, Maxar and Planet are especially restrictive in terms of freely accessible material they provide, as they only offers a very limited number of extremely high resolution samples of business grade solutions they offer. Because of this both services have been deemed unnecessarily expensive and excessively restrictive in their free tier form for the scope of this work.
Google Maps and Bing Maps offer fairly similar services, both in terms of quality and accessibility. Both providers expose APIs that allow users to collect both

maps and satellite imagery via HTTPS requests, by specifying different parameters for tailoring the request to one's particular needs. Google's service provides on-demand API access, which is free for mobile applications, and is priced at 2$ per 1000 requests otherwise. Bing Maps on the other hand offers, as specified in the Bing Maps API Terms of Use [29], a free API tier for educational and non-profit applications which use up to 50,000 billable transactions within a 24-hour period. Given the similarities between Google's and Microsoft's services and the fairly simple needs of this project, Bing Maps ended up being the provider of choice mostly because of the free tier offer, as the dataset of this project is not intended for commercial use.

4.1.2 Dataset creation

Since oceans cover approximately 70% of Earth's surface, randomly choosing the coordinates to gather satellite imagery may result in a significant portion of the collected images to include water only. These images would lack important features and would hinder the training process. For this reason, the coordinates used to generate this dataset have been randomly picked from the World Cities Database, offered by SimpleMaps.com. In its basic format it allows commercial use under a Creative Commons Attribution 4.0, providing a list of about 41 thousand locations, mostly prominent cities such as capitals and large cities, in CSV format.
The dataset contains a considerable number of features, most important of which:

- *city*: contains the name of the city as a Unicode string. Also available as an ASCII string under the *city_ascii* field.
- *lat*: indicates the latitude of the location.
- *lng*: indicates the longitude of the location.

Other fields include information about population and density, timezone, whether the city is a capital or not, etc. For the scope of this project, the only relevant information is the one contained in the former list. The whole dataset is composed of 1000 images, 800 of which will belong to the training set, 100 will be used for validation and the remaining 100 for testing. 1000 cities and relative coordinates are chosen at random from the World Cities Dataset, and satellite images of the selected locations are collected via API calls.
This is the format of the HTTPS request sent to the Bing Maps API service:

```
http://dev.virtualearth.net/REST/v1/Imagery/Map/Aerial/
   {lat},{lon}/{zoom}?mapSize={width},{height}&fmt=png&key={
   api_key}
```

where `lat`, `lon` indicate the latitude and longitude respectively and are used to determine the center of the captured image, `zoom` is an integer that indicates the zoom level, which determines the ground resolution, `width` and `height` indicate the width and height of the requested image in pixels, `png` indicates the requested image format, and `api_key` is the API key provided by Bing Maps to successfully perform API calls. These API calls have been performed using the `requests` Python package. The ground resolution is determined as:

$$resolution = \frac{156543.04 \cdot \cos(lat)}{2^{zoom}} \tag{4.1}$$

Since we know that Sentinel-2 images are captured at a ground resolution of 10m/pixel, we can derive the zoom level needed to acquire images from Bing Maps at the desired resolution, as to create different ground-resolution versions of the same dataset
It is quite clear from *eq. 4.1* that increasing the zoom level by 1 halves the ground resolution, thus increasing the level of detail, while decreasing it by 1 doubles it, since lower ground resolutions result in higher-definition images. Several versions of the dataset have been created, as to accommodate for different upscaling tasks: the ground truth dataset is composed by the highest resolution images, at $\approx$ 1.25m/pixel, while the low resolution counterpart depends on the desired upscaling factor. In total, 5 different versions of the dataset were created:

- a $\approx$ 1.25m/pixel dataset, which will be used as ground truth for all training tasks.
- a $\approx$ 2.5m/pixel dataset, used as low resolution images for training the models on 2x upscaling.
- a $\approx$ 5m/pixel dataset, used as low resolution images for training the models on 4x upscaling.
- a $\approx$ 10m/pixel dataset, used as low resolution images for training the models on 8x upscaling.

Low resolution images formally constitute the training datasets, while the highest-resolution ground truth dataset is used to train the models in a supervised fashion, where super-resolved images are the predictions of the model and the ground truth is used for comparison to train the neural-network based models.
The low resolution datasets could also have been generated by downscaling the original image. This methodology has been considered and tested, but since lower resolution images were natively available from Bing Maps and the quality of the downscaled images largely depends on the algorithm used for downscaling and the types of images of choice, it was rejected for this use case.

Figure 4.1: ≈ 1.25m/pixel sample

Figure 4.2: ≈ 2.5m/pixel sample

Figure 4.3: ≈ 5m/pixel sample

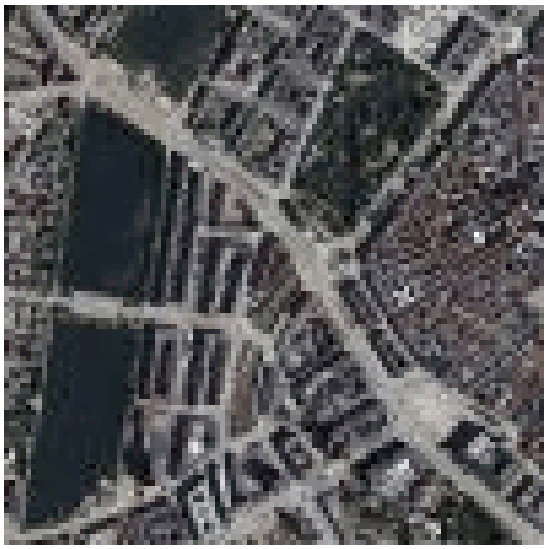

Figure 4.4: ≈ 10m/pixel sample

As it can be seen from the samples from *fig.4.1* to *fig.4.4*, all versions span over the same spatial area, differing only in resolution. While all images apart from the smallest in size contain a watermark from Bing Maps for copyright reasons, this should not interfere in the processing phases as only patches of different sizes will be used as training data. The patches will be taken at random each time an image is passed to the neural network, and random rotations and flips are also applied with a 50% chance in order to perform some data augmentation. This is done when

training every model so that the training dataset is enlarged by a factor of 100, so that it will effectively contain 80,000 fairly different images.

Georeferencing

When performing API calls to obtain images from Bing Maps, an additional `mmd` argument can be set to 1 in order to retrieve the metadata of the API call, which include, among others, information about the requested image in terms of pixel size and, most importantly, bounding box. The bounding box is a list of 4 coordinates, which indicate the left-bottom-right-top coordinates of the image. While in a simple super resolution task georeferencing the images is not relevant, it may still be needed for other previously mentioned algorithms that work satellite imagery. Georeferencing allows to correctly position and overlay satellite images in a GIS (Geographic Information System). To correctly do so, extra data is needed to determine the coordinate system used to map the pixels to geographic coordinates. The most common format for georeferenced images is GeoTIFF, which is a public domain metadata standard which allows georeferencing information to be embedded within a TIFF file. The additional information can include map projection, coordinate reference systems (CRS), and everything necessary to establish the exact spatial reference for the image. A widely used CRS is the WGS (World Geodetic System), and its latest revision (WGS84, proposed by the United States National Geospatial-Intelligence Agency in 1984 and last revised in 2014) is use by the Global Positioning System (GPS).

Test data from Sentinel-2

Since the objective of this project is to analyze and propose methods for successfully upscaling Sentinel-2 images, data from this satellite has been used for building the test portion of our dataset. Test data should not be similar to training or validation data, as to accurately and realistically measure the goodness of the models in a scenario which has to be as close as possible to a real use case.

As it can be seen from *fig.4.5*, images collected from the Sentinel-2 satellite are very different from the ones collected via the Bing Maps service. The main difference comes from the physical sensors used for capturing the images, which results in visually different images while maintaining the characteristic features of satellite imagery. While ESA does provide free access to Sentinel data, Sentinel-2 images have been collected via the SentinelHub platform by Sinergise, which exposes a set of APIs that allow for requesting satellite data according to ones specific needs and exact criteria. Sentinel Hub is an engine for processing of petabytes of satellite data. It makes Sentinel, Landsat, and other Earth observation imagery easily accessible for browsing, visualization and analysis. Apart from offering commercial grade features for data analytics, SentinelHub provides free plans and free trials

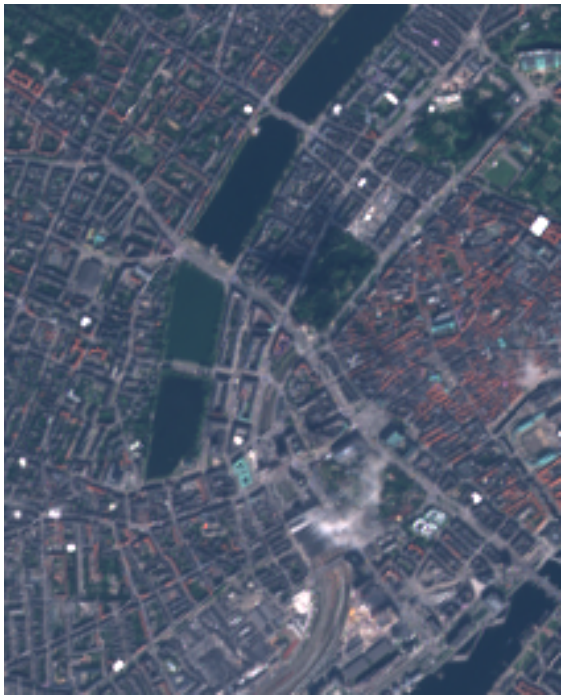

Figure 4.5: Sentinel ≈ 10m/pixel sample

which can be used to use the APIs for non-commercial uses. SentinelHub can be used via the homonym Python package which allows to directly configure the APIs and to easily request satellite data. After having configured the package with SentinelHub's client ID and client secret, the `SentinelHubRequest`[30] class can be used for requesting data, by specifying the data collection (Sentinel-2 in this case), the time interval to consider when requesting the image, the mosaicking order which can be mostRecent, leastRecent or leastCC (least cloud coverage: this is the option used to avoid excessive cloud coverage which would render the image unusable), output format (usually TIFF or PNG: the latter has been used for this application), the desired bounding box and image size. Even when requesting images with the least amount of cloud coverage, results are extremely inconsistent depending on the location. To further avoid excessive cloud coverage, only images having less than 5% of white pixel percentage have been kept. This has been done by counting the number of pixels whose value in the R, G and B channels are all above 250. A value of 255 on all 3 channels represents pure white, but since we are dealing with real world data, pixels may not necessarily be exactly white, hence the limit has been lowered to 250 to calculate the white pixels ratio.

4.2 The models

In order to propose a contribution, two of the previously mentioned state of the art models have been chosen for analysis in this specific use case: EDSR and ESRGAN. These models have been chosen as they have been deemed as a fairly advanced basis to study and propose improvements on. While research in the field is constantly producing new results, such as ultra dense GANs[31] and siamese networks[32], the former methods still represent a reliable and efficient solution in most use cases, while allowing room for improvements and new proposals. These two models have been tested starting from the code provided by Wang's *BasicSR* library[33], which provides a variety of tools for image and video restoration using PyTorch, such as super-resolution, denoising, deblurring, JPEG artifacts removal. The proposed architecture is built using the PyTorch framework, and the code is available on GitHub[1]. All models have been trained in a distributed fashion on a shared server hosted by LINKS Foundation, using as much hardware resources as possible at a given time. Usually, this meant the model were trained using 2 NVIDIA GTX 1080 Ti GPUs. All experiments have been monitored via Weights and Biases(W&B)[2], which is a machine learning platform that provides tools for monitoring and running machine learning related code. W&B offers lightweight tools to track experiments, versions, evaluate model performance, and visualize results via a cloud hosted dashboard, similar in concept to TensorFlow's TensorBoard. W&B can be used via its easy to configure python package, which allows to log the desired info during an experiment by simply allowing a user to log into their profile, and linking an experiment to a project.

4.2.1 Training the EDSR model

The EDSR model has been trained only for a 4x upscaling super resolution task to preliminarily evaluate its performance. As it was not used for proposing improvements on the architecture, the analysis of this model has been limited because of time constraints. As high resolution ground truth, the Bing Maps dataset with $\approx$ 1.25m/pixel ground resolution was used, whereas the low resolution dataset was composed again of Bing Maps images, but of $\approx$ 5m/pixel ground resolution. Training low resolution data is presented to the network as RGB patches of 32x32 pixel in size, augmented with random horizontal and vertical flips, 90 degree rotations and Gaussian blur to further degrade the images. All images are pre-processed by subtracting the mean RGB value of the training dataset dataset. The model

[1]https://github.com/FilBr/Multi-scale-ESRGAN

[2]https://wandb.ai/

is trained with ADAM optimizer, by setting $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 10^{-8}$. The batch size is set at 10 per GPU. The learning rate is initialized as 10^{-4} and halved at every 2×10^5 batch updates. The model is initialized with weights of an EDSR model pretrained on the DIV2K dataset. Training is performed over 100,000 iterations, which along with the number of GPUs determines the number of training epochs as:

$$\text{iterations per epoch} = \left\lceil \frac{\#\text{ training data} \times \text{enlarge factor}}{\text{batch size per GPU} \times \#\text{ of GPUs}} \right\rceil \tag{4.2}$$

$$\#\text{ of epochs} = \left\lceil \frac{\text{total iterations}}{\text{iterations per epoch}} \right\rceil \tag{4.3}$$

The model is trained using an L1 loss with mean reduction to evaluate the quality of the image in the pixel space. The L1 loss is simply defined as:

$$l_n = |x_n - y_n| \tag{4.4}$$

and is equivalent to the mean absolute error. The training progress was tracked every 100 iterations, while validation has been performed every 5000 iterations. During validation, the 3 different evaluation metrics, PSNR, SSIM and NIQE are calculated on the 100 validation images. The model is fairly light when it comes to computational resources, requiring only ≈ 8 hours to train.

4.2.2 Training the ESRGAN model

Several experiments have been performed with the ESRGAN model, testing its capabilities on the 2x, 4x, and 8x super resolution tasks. The generator model has been maintained consistent through the experiments, varying only the second upsampling layer's scale factor to accommodate for the different tasks. The discriminative network employed was the same as the one from the original ESRGAN paper, a VGG-19 network for the binary classification of 128x128 pixel images. Similarly to the procedure followed for training the EDSR model, the ≈ 1.25m/pixel Bing Maps data has been used as ground truth for all super resolution tasks, whereas the low resolution images were chosen according to the scaling:

- the ≈ 2.5m/pixel data has been used as low resolution data for 2x upscaling
- the ≈ 5m/pixel data has been used as low resolution data for 4x upscaling
- the ≈ 10m/pixel data has been used as low resolution data for 8x upscaling

Again, low resolution data has been fed to the generator in the form of square patches (64x64 pixels for 2x scaling, 32x32 pixels for 4x scaling, 16x16 pixels for 8x

scaling), randomly cut out from the whole images and randomly rotated, flipped and blurred so that the training dataset could reach an enlargement of a factor of 100. Differently from the original paper, the architecture has been trained on 100,000 iterations instead of 1,000,000 because of time and computational constraints. Both the generator and the discriminator have been optimized using separate Adam optimizers, both with $\beta_1 = 0.9, \beta_2 = 0.99$ and an initial learning rate of 10^{-4}. The learning rates have been halved every 5000 iterations, until reaching 30,000 iterations. These values have been reduced by an order of magnitude to accommodate for the reduction in total number of iterations. The batch size per GPU has again been set to 10, and the total number of training epochs can be derived from *eq. 4.2* and *eq. 4.3*. For all tasks, the generator has been initialized with the weights pretrained on a 4x upscaling task on the dataset DIV2K. While this may seem counterproductive when dealing with different scaling factors, a pretrained network greatly reduces the needed training time as it allows to take advantage of the already trained feature extraction portions of the network, resulting in a need to only finetune during training. Using a network pretrained on satellite data may help towards reducing the training time while also improving the performance of the architecture. The architecture uses 3 different loss functions:

- a pixel-related loss function, in the form of an L1 loss.

- a perceptual loss function which aims at optimizing super resolution in a feature space, based on a finetuned VGG network. This loss is in the form of an L1 loss, but operates on the feature space. This loss also has the option to be used in style GAN scenarios.

- a relativistic discriminator loss, in the form of a Binary Cross Entropy with Logits, which is used to try to predict the probability that a real image x_r is relatively more realistic than a fake one x_f

Similarly to the training procedure used for the EDSR model, the values of the three losses have been tracked every 100 iteration on the W&B platform. Validation is also performed in a similar manner, every 5000 iterations on the entire validation dataset of 100 images. This model is fairly more complex than EDSR, and this translates in a considerable inflation in the training times, requiring $\approx$ 12 hours to completely train the architecture using 2 GPUs for 2x super resolution, $\approx$ 16 hours for the same training on a 4x super resolution task, and $\approx$ 16 hours for 8x super resolution. These high training times are what motivated the choice for reducing the total training iterations by an order of magnitude.

4.2.3 Proposal: a multi-scale approach

Starting from the ESRGAN architecture, this thesis proposes a multi-scale approach to optimize the training of a single model over different scaling factors. While using the same main body as the ESRGAN architecture, with 23 residual in residual dense blocks (RRDBs), with thesis project we propose an architecture using a generator which splits 3-way when generating the super resolved images: one 2x, one 4x and one 8x scaled super resolved images. All 3 splits are independent from each other, as to train each section so that it specializes on upscaling on a single scaling factor.

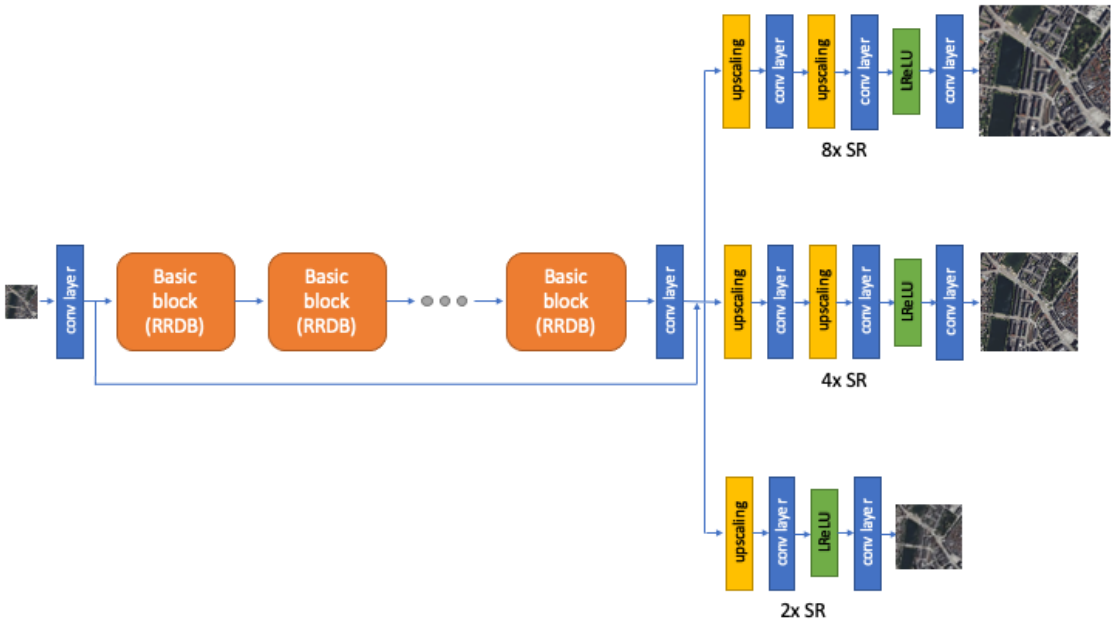

Figure 4.6: Multi-scale generator architecture

where, similarly to the original ESRGAN model, the Basic blocks, also known as Residual-in-Residual Dense Blocks, are composed of the following layers:

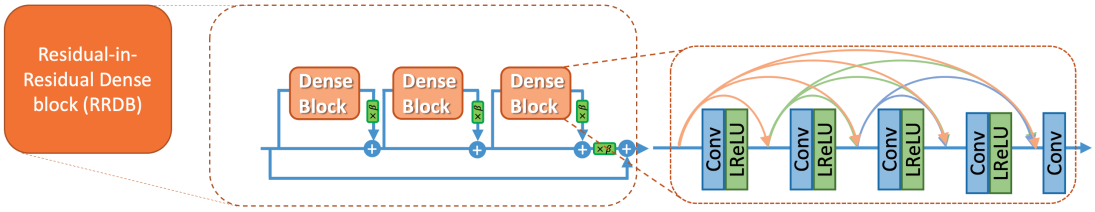

Figure 4.7: Residual-in-Residual Dense Blocks

The idea behind this proposal is to aim for a more generalized architecture able to produce high-quality super resolved images on different scales. By using 3 independent branches in the generator, and by using 3 independent discriminators to judge the results of each product of the first network, the goal is to create a network which is capable of extracting the features from an image independently of the upscaling factor, while the last separated layers dedicated to increasing the resolution are independent from each other in order to avoid possible interference between training the layers for a scale or others.

Training

For training this architecture, 3 different ground truth sets have been used. While the Bing Maps images of $\approx$ 10m/pixel resolution are still used as low resolution training data, each split necessitates its own ground truth to perform different scaling tasks:

- the $\approx$ 1.25m/pixel dataset is used as ground truth for the 8x scaling
- the $\approx$ 2.5m/pixel dataset is used as ground truth for the 4x scaling
- the $\approx$ 5m/pixel dataset is used as ground truth for the 2x scaling

To each output of the generator network is linked an independent discriminator. Just like in the original ESRGAN architecture, the discriminators are all based on a VGG-19 architecture for feature extraction, and are inspired by the Relativistic average GAN (RaGAN)[28], which learns to judge whether one image is more realistic than the other, rather than whether one image is real or fake. By using 3 discriminators for training, they all contribute to updating weights in the common portion of the generator architecture, which is responsible for feature extraction, while independently contributing each one to a specific super resolution branch. For each batch of the dataset, the generator is trained by computing the pixel-space loss, the feature-space loss and via the feedback provided by all 3 discriminators on the 3 produced images, and performing a backpropagation on a linear combination of these losses. Then, only the discriminators are trained individually as to improve their recognition of the super resolved images on their respective scale of interest. Training is performed in a similar fashion to that of the ESRGAN model. Low resolution data has been fed to the generator in the form of 16x16 pixels square patches, randomly cut out from the whole low-resolution images and randomly rotated, flipped and blurred so that the training dataset could reach a size 100 times that of the original dataset. Again, the architecture has been trained on 100,000 iterations because of time and computational constraints. Both the generator and the discriminators have been optimized using separate Adam optimizers, both with $\beta_1 = 0.9, \beta_2 = 0.99$ and an initial learning rate of 10^{-4}. The learning rates have

been halved every 5000 iterations, until reaching 30,000 iterations. As when training the ESRGAN model, these values have been reduced by an order of magnitude to accommodate for the reduction in total number of iterations from the original paper. The batch size per GPU has again been set to 10, and the total number of training epochs can be derived from *eq. 4.2* and *eq. 4.3*. For all tasks, the generator has been initialized with the weights pretrained on a 4x upscaling task on the dataset DIV2K. Only the 4x super resolution branch has been initialized with pretrained weights, while the 2x and the 8x ones have been initialized to standard values. This architecture is considerably harder to train from the computational resources standpoint, as it effectively involves two extra neural networks used for discriminating super resolved images, and the generator presents extra layers as well, requiring 3 backpropagations to train. The training procedure takes about double the time needed to train the original ESRGAN, lasting ≈ 42 hours to train on 2 GPUs.

4.3 Results

4.3.1 EDSR

While this model produced the highest values of average PSNR scores, the the super resolved images lack detail, are excessively smooth and do not appropriately represent some of the characteristic features of satellite images. While this architecture may work well on other image domains, satellite images are rich in high contrast feature that describe the different elements of the image: streets in crowded city areas are poorly recovered, and buildings in groups are indistinguishable from each other most of the time. The poor perceptual results of this network drove this thesis project towards GAN architectures, which have been at the center of research in the field for the past few years and, at least in other contexts, seemed to produce far more convincing results.

Figures 4.10, 4.8, 4.9 represent a sample taken from the validation dataset, which has a high resolution ground truth for comparing the super resolved image and calculating the PSNR and SSIM metrics.

Figures 4.11, 4.12 refers to a Sentinel-2 test sample and its relative super resolved image produced by the network.

4.3.2 ESRGAN

While this model produced lower metric scores compared to the ones obtained on the DIV2K dataset analyzed in the original paper[27], it produced far more convincing results compared to the EDSR model. The model is able to convincingly

Figure 4.8: LR sample (val)

Figure 4.9: EDSR 4x SR image (val)

Figure 4.10: GT sample (val)

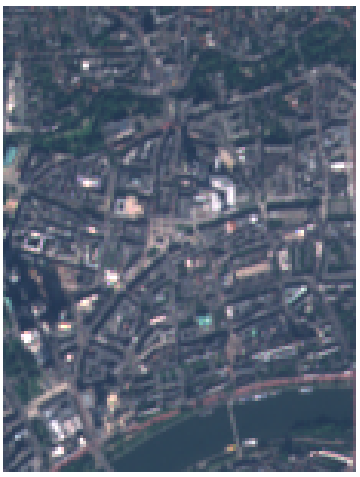

Figure 4.11: Low resolution sample (test)

Figure 4.12: EDSR 4x upsampled SR image (test)

recreate most of the features of the image, producing promising results on every tested upsampling scale (i.e., 2x, 4x, 8x).

Figures 4.19, 4.20, 4.21 refer to the ground truths for the 2x, 4x and 8x super resolution tasks respectively, *figures 4.13, 4.13, 4.13* represent the low resolution images used for the 2x, 4x and 8x super resolution tasks respectively, while *figures 4.16, 4.17, 4.18* represent the super resolved images generated starting from their lower resolution counterparts. These figures belong to the validation dataset, gathered from Bing Maps, and have a ground truth which allows to compute all metrics: PSNR, SSIM and NIQE.

Figures 4.22, 4.23, 4.24, 4.25 refer to a Sentinel-2 test sample and its relative super resolved image produced by the architecture by a 2x, 4x and 8x super resolution tasks respectively.

Figure 4.13: LR sample for 2x SR(val)

Figure 4.14: LR sample for 4x SR (val)

Figure 4.15: LR sample for 8x SR (val)

Figure 4.16: ESRGAN 2x upsampled SR image (val)

Figure 4.17: ESRGAN 4x upsampled SR image (val)

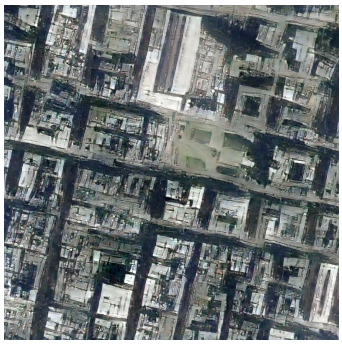

Figure 4.18: ESRGAN 8x upsampled SR image (val)

Figure 4.19: GT sample for 2x SR (val)

Figure 4.20: GT sample for 4x SR (val)

Figure 4.21: GT sample for 8x SR (val)

The images belonging to the test dataset do not have a ground truth to compare the results to. Only the NIQE metric can be applied to this images for evaluation,

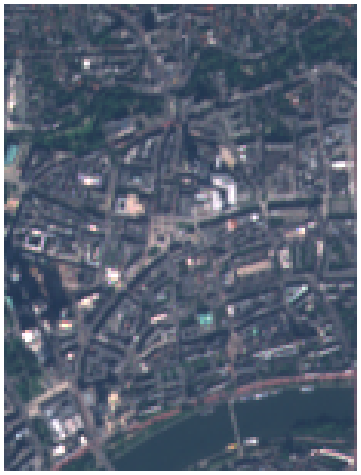

Figure 4.22: Low resolution sample (test)

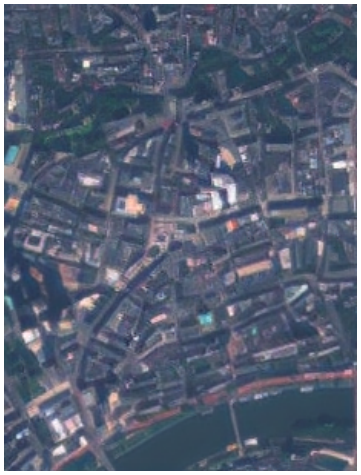

Figure 4.23: ESRGAN 2x upsampled SR image (test)

Figure 4.24: ESRGAN 4x upsampled SR image (test)

Figure 4.25: ESRGAN 8x upsampled SR image (test)

since it's the only blind metric that does not require a ground truth.

4.3.3 Multi-scale ESRGAN

The proposed architecture produces fairly promising results, matching the visual quality of samples produces by the standard ESRGAN architecture on the dataset chosen for this project. Starting from a single low resolution sample, the architecture produces 3 different

Figure 4.26 represents the low resolution image used to generate super resolved

samples, *figures 4.27, 4.28, 4.29* indicate the ground truths for the 2x, 4x and 8x super resolution tasks respectively, while *figures 4.30, 4.31, 4.32* represent the produced super resolved images. The samples were taken from the validation dataset, which has high resolution ground truths for comparing the super resolved images and calculating the PSNR and SSIM metrics, which give a score based on the comparison of two images.

Figure 4.26: LR sample (val)

Figure 4.27: GT sample for 2x SR (val)

Figure 4.28: GT sample for 4x SR (val)

Figure 4.29: GT sample for 8x SR (val)

Figure 4.33 shows a low resolution sample taken from the Sentinel-2 test dataset, while *figures 4.23, 4.24, 4.25* refer to the super resolved images produced by the proposed multi-scale architecture.

When comparing the evaluation metrics for the 3 models we see that ESRGAN and the multi-scale proposed architecture perform fairly similarly, with the proposed architecture producing slightly lower in terms of SSIM metric scores. The difference in performance does not produce noticeable differences in the produced images. Metrics for the multi-scale architecture have been estimated as the average of the metrics on the 3 different scales.

Figure 4.30: Multi-scale ESRGAN 2x SR image (val)

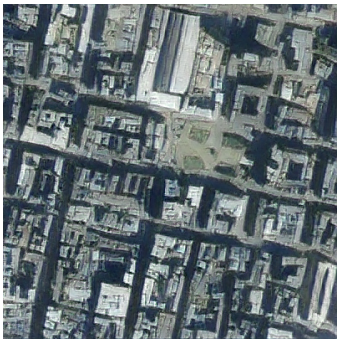

Figure 4.31: Multi-scale ESRGAN 4x SR image (val)

Figure 4.32: Multi-scale ESRGAN 8x SR image (val)

Figure 4.33: Low resolution sample (test)

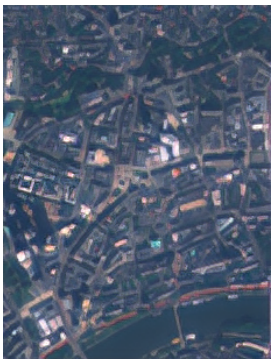

Figure 4.34: Multi-scale ESRGAN 2x SR image (test)

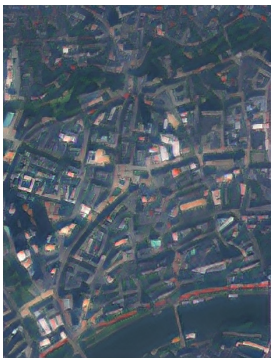

Figure 4.35: Multi-scale ESRGAN 4x SR image (test)

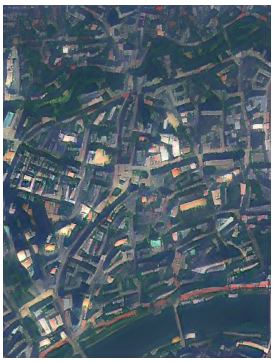

Figure 4.36: Multi-scale ESRGAN 8x SR image (test)

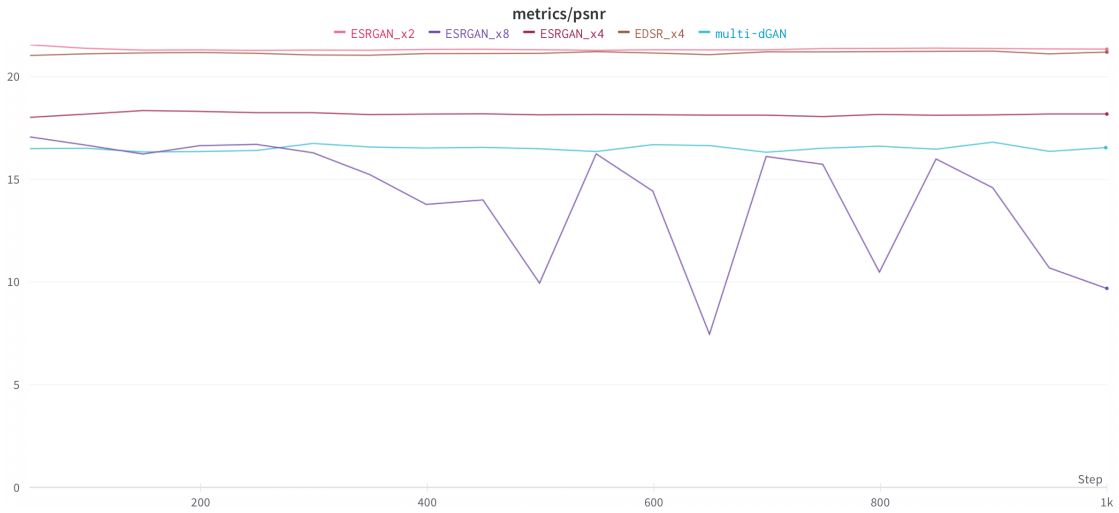

Figure 4.37: Comparison of PSNR scores between the 3 models

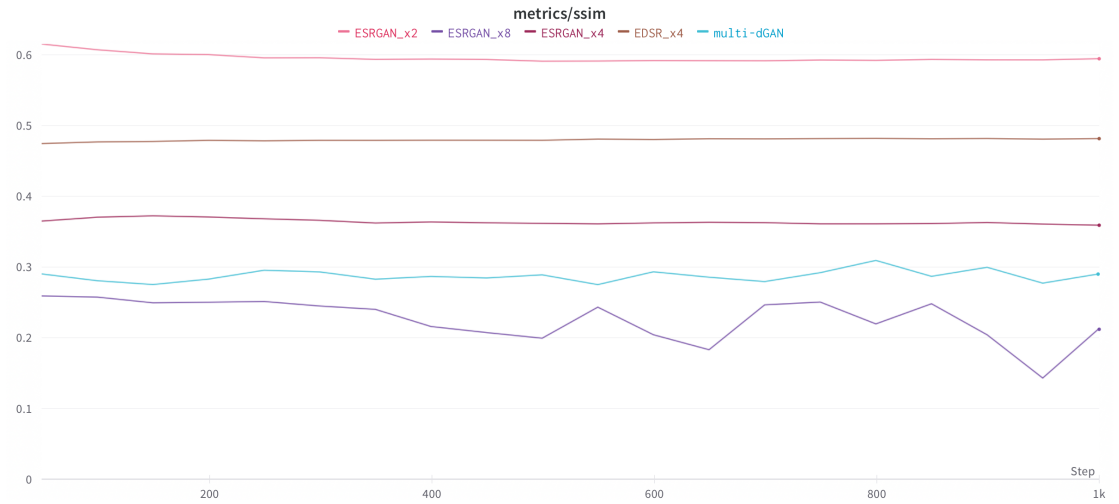

Figure 4.38: Comparison of SSIM scores between the 3 models

Both GAN-based architectures produce PSNR scores which are much smaller than those produced by the EDSR model, which applies an excessive smoothing action causing the super resolved images to lack important details. The GAN-based architectures produce much more realistic images, with the proposed multi-scale architecture generating super resolved images which are very similar to the ESRGAN network trained on a single scale. It is quite clear from analyzing these metrics' behaviors that, even though they are useful in providing insight between different methods, they are not the right choice for evaluating the quality of satellite images. Noise does not seem to affect to heavily the quality of the produced images, and the

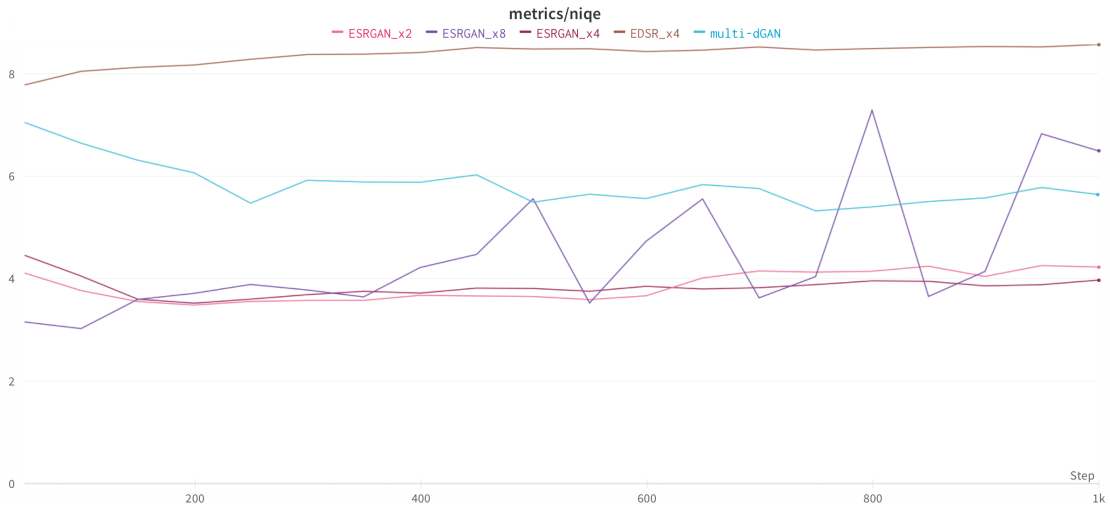

Figure 4.39: Comparison of NIQE scores between the 3 models

NIQE blind metric seems to be the only metric score producing results which are in line with the perceived quality of the produced images on such images, without being negatively impacted by the noise scores.
Here follows a side-by-side comparison of the images produced by the three models (EDSR, ESRGAN, multi-scale ESRGAN) on a 4x super resolution task, along with an example of bicubic interpolation used for a similar 4x upscaling task.

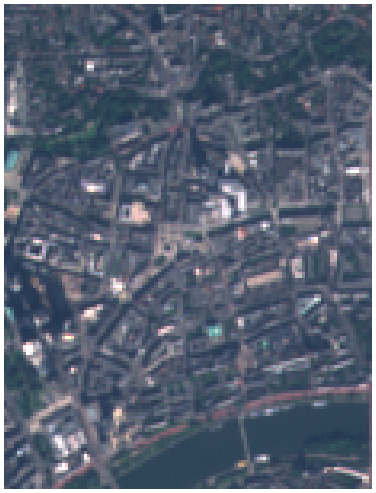

Figure 4.40: LR sample

Since low resolution images without high resolution ground truth were used as test dataset, the only metric that can be evaluated is the Naturalness Image Quality

Figure 4.41: Bicubic 4x upsampled SR image

Figure 4.42: EDSR 4x upsampled SR image

Figure 4.43: ESRGAN 4x upsampled SR image

Figure 4.44: Multi-scale 4x upsampled SR image

Evaluator (NIQE), which allows to measure deviations from statistical regularities observed in natural images, where a smaller NIQE score indicates better perceptual quality.

From the results reported in *table 4.1* it's possible to observe that bicubic interpolation and the EDSR model actually produce perceptually worse looking super resolved images when compared to the baseline value calculated on the low resolution datasets. The GAN-based architectures are the best performing ones, providing results which are in line with analysis often performed in research. The proposed multi-scale architecture is able to produce results which are comparable

Model	NIQE
Low resolution test images	5.43
Bicubic 4x	8.92
EDSR 4x	8.32
ESRGAN 2x	4.07
ESRGAN 4x	3.03
ESRGAN 8x	3.34
Multi-scale ESRGAN average	3.67
Multi-scale ESRGAN 2x	4.87
Multi-scale ESRGAN 4x	3.05
Multi-scale ESRGAN 8x	3.11

Table 4.1: NIQE metrics evaluated on the Sentinel-2 single image test dataset

in perceptual quality to those of the classic ESRGAN model when trained on a single scale, confirming the successful creation of an architecture able to operate on multiple scales simultaneously while maintaining State of the Art quality in the produced super resolved images.

4.4 A proof of concept: solar panel detection

After having determined the acceptable performance of the proposed architecture by comparing the evaluation metrics scores to those obtained with the original ESRGAN model, we study the application of the model on images used by LINKS Foundation, the company who provided the computing facilities and offered support for developing this thesis project.
LINKS uses a high definition dataset of satellite images of large areas surrounding the Alessandria and Asti provinces to perform solar panel detection. While the high resolution dataset has not been disclosed, the bounding boxes of the covered areas and annotations of the detected solar panels have been shared for demonstrating the practical applications of the architecture proposed in this thesis work.
With the intention of performing the same solar panel detection task starting with the low resolution Sentinel-2 satellite images, with thesis project we propose an approach to performing such a task, briefly discussing the visual results of the produced images of interest. Starting with the definition of the geographical bounding box, a Python scripts gathers both the high resolution imagery from Bing Maps (to provide a visual reference to what a high resolution version of the produced images would look like) and low resolution samples from Sentinel-2, similarly to how its been described in the previous *Dataset creation* section. In this task, the georeferencing aspect plays an important role, as images are visualized as

a mosaic of smaller images using GIS software, in particular QGIS.

Starting from the north-west corner of the bounding box, the Python script gathers the georeferenced data in GeoTIFF format, scanning the entirety of the bounding box until the whole geographical area is covered. The gathered satellite images represent small geographical areas described using the WGS84 standard set of coordinates. The low resolution georeferenced images are then transformed in a raster format, PNG, since georeferencing data is not needed for producing the super resolved images. Low resolution images are fed to the proposed trained multi-scale generator network described in previous sections, and the three super resolved images are produced. These images are then transformed back into GeoTIFF for georeferenced visualization in QGIS, which allows to import the generated images and automatically tiles them in the correct locations, thanks to the geographical metadata contained in the GeoTIFF images. Unfortunately, the starting low resolution sentinel images lack important details, which do not allow for an optimal reconstruction of the missing features characteristic of solar power plants, such as the distinctive rows or grids in which solar panels are positioned.
Here follow some examples of the low resolution Sentinel-2 data used for generating the super resolved images, as well as some high resolution examples taken from Bing Maps to show what an ideal ground truth would look like.

Figure 4.45: LR sample (Alessandria)

While images from Bing Maps and Sentinel-2 are in very different domains because they have been captured in different time instances, it is quite clear that the super resolved images are lacking detail. *Fig. 4.45* represents a solar power plant, as it can be better seen from the high resolution images collected from Bing Maps, while *fig. 4.52* is an image of industrial buildings with solar panels on their roof. Even if they represent an improvement from the initial low resolution Sentinel-2 images as it can be seen in *table 4.2*, in the super resolved products solar panels are still hard to discern from the surrounding environment, especially in the case

Figure 4.46: SR sample 2x upscaled (Alessandria)

Figure 4.47: SR sample 4x upscaled (Alessandria)

Figure 4.48: SR sample 8x upscaled (Alessandria)

Figure 4.49: GT sample for 2x SR (Alessandria)

Figure 4.50: HR sample for 4x SR (Alessandria)

Figure 4.51: HR sample for 8x SR (Alessandria)

Figure 4.52: LR sample (Asti)

Figure 4.53: SR sample 2x upscaled (Asti)

Figure 4.54: SR sample 4x upscaled (Asti)

Figure 4.55: SR sample 8x upscaled (Asti)

Figure 4.56: GT sample for 2x SR (Asti)

Figure 4.57: HR sample for 4x SR (Asti)

Figure 4.58: HR sample for 8x SR (Asti)

of solar panels on industrial buildings were they are completely indistinguishable from a generic roof. An idea to produced higher perceived quality images for this specific task could be to train the architecture using a dataset which focuses on the presence of solar panels in the proposed imagery, as to aim for a model specialized in the reconstruction of the key features of solar panels, or to use a starting low resolution dataset having a better ground resolution than that of Sentinel-2.

Scale	NIQE
Low resolution images	4.62
Multi-scale ESRGAN average	4.19
Multi-scale ESRGAN 2x	4.00
Multi-scale ESRGAN 4x	3.96
Multi-scale ESRGAN 8x	4.60

Table 4.2: NIQE metrics evaluated on the Sentinel-2 solar panels dataset

Chapter 5

Conclusions

In this document, we presented a more generalized multi-scale Generative Adversarial Network for performing super-resolution tasks on different scale, trained exclusively on satellite images. Along with the neural network architecture, we propose a satellite imagery dataset for single image super resolution tasks with the objective of training a model that can be used to super resolve open source satellite images collected from ESA's Sentinel-2, and we implement a streamlined pipeline that, starting from the raw collected data, performs data augmentation by means of image transformations and prepares the training samples to be used for training the analyzed models. The open data policy of the test dataset plays an important role in the implications of successful applications. We highlight the poor efficiency of simple Convolutional neural network-based solutions, that tend to produce less defined super resolved images which are less amiable to the human visual perception, even though they seem to perform better in terms of the evaluation metrics employed in this project. The work proposed with this thesis project is also motivated by the fact that single-scale super resolution architectures still generate a significant amount of coverage in research, but usually lack the flexibility of producing convincing results on different scale simultaneously. It's also quite clear that, even though interpolation-based techniques are extremely less impacting on hardware resources when compared to training deep learning models, after having overcome the obstacle imposed by training, the deep learning models consistently produce much better looking and visually coherent results. The difficulties of training such complex models are still of great relevance, as they still present an important hindrance. To demonstrate the usefulness of performing super-resolution on satellite imagery, a proof of concept study focusing on images used by LINKS Foundation for solar panel detection is performed. We conclude that Generative Adversarial Networks can lead to aesthetically pleasing SR images, even when considering all the issues related to the training procedures, if carefully

designed with appropriate loss functions. The use of suitable metrics and architectures geared towards the application domain hold an important role towards generating more realistic images. The proposed multi-scale approach performs similarly to established State of the Art models of similar nature, hinting at the possibility of employing this model successfully for real-world tasks.

5.1 Future work

Different evaluation metrics - As it can be observed by the results of the experiments, the PSNR and SSIM metrics are not the best indicators for judging the quality of the produced images. As discussed in the introduction of these metrics, optimizing for PSNR in particular tends to over-smoothen the images produced by the model, resulting in a lack of important detail. These metrics also seem poor indicators for judging the quality of satellite images, as noise, when not to excessive, or differences in the qualities judged by the SSIM are not necessarily indicators of poorly super resolved satellite images. In satellite images, it's of prime importance the ability to distinguish building from roads, or different types of land (e.g., green areas, waters, urban areas, deserts, forests, mountains, etc.). A study of more relevant evaluation metrics could be of relevance for improving the work proposed in this thesis project.
While the NIQE scores are more in line with the perceptual quality of the results, PSNR and SSIM provide negligible insights on the actual quality of the produced images.

Different loss functions - While the original ESRGAN model had been studied and validated on more generic datasets (e.g., DIV2K), it may be worth investigating alternative hyperparameters and loss functions used to train the model on a dataset of only satellite imagery. Satellite images have very distinctive features which are rarely found in a generic dataset, which may have caused the degradation in performance metrics compared to the results obtained in the model's paper [27].

Deeper generator architecture - Because of temporal and computational limits, the State of the Art architectures and the proposed ones have either been trained for less time than may have actually been needed to improve results, or have been considered in a limited-depth scenario, in order to have reasonable training times and successfully fit the models in memory. With a better hardware infrastructure, it would be worth investigating the effect of longer training procedures or the use of deeper models. Especially for the proposed architecture, working on 3 different scaling tasks, the limited size of the branches specialized for branching may have caused the delta in metrics observed from the produced results.

Different test cases - While solar panel detection is an important use case, it may not be the ideal example to validate the proposed architecture's performance. Even though solar power farms are vast in surface area, the single rows or sections of solar panels are very limited in size, and because of Sentinel-2's ground resolution images of such sites are originally captured so that they lack important features that could help produce better super resolved images. This issue could be solved by either training the architecture specifically on images containing solar panels, or by using a source with ground resolution higher than the 10 meters per pixel offered by Sentinel-2. Other applications may be better suited to take advantage of super resolved images produced by the proposed architecture.

Domain adaptation - The quality and yield of satellite imagery is extremely dependent on the physical sensor used to capture said images. These differences are mostly noticeable in the differences in colors between different sources, which can be observed between the Bing Maps dataset and the Sentinel one. This results in a slight change in colors when performing super resolution on Sentinel-2 data, as Bing Maps colors are more vibrant. For this reason, it could be worth investigating the effect of a GAN-based architecture for applying domain adaptation to produce more accurate results color-wise. Domain adaptation has also been considered for this thesis project as a tool to train a super resolution model using synthetic datasets that could be gathered from video-games such as Microsoft Flight Simulator, which offers realistic satellite imagery.

Bibliography

[1] European Space Agency. *PROBA-V Super Resolution*. URL: https://kelvins.esa.int/proba-v-super-resolution/ (cit. on p. 3).

[2] Francesco Salvetti, Vittorio Mazzia, Aleem Khaliq, and Marcello Chiaberge. «Multi-Image Super Resolution of Remotely Sensed Images Using Residual Attention Deep Neural Networks». In: *Remote Sensing* 12.14 (2020). ISSN: 2072-4292. DOI: 10.3390/rs12142207. URL: https://www.mdpi.com/2072-4292/12/14/2207 (cit. on p. 3).

[3] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning : from theory to algorithms*. 2014. ISBN: 9781107057135 1107057132. URL: http://www.worldcat.org/search?qt=worldcat_org_all&q=9781107057135 (cit. on p. 7).

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016 (cit. on p. 7).

[5] Xinyu Dou, Chenyu Li, Qian Shi, and Mengxi Liu. «Super-Resolution for Hyperspectral Remote Sensing Images Based on the 3D Attention-SRGAN Network». In: *Remote Sensing* 12.7 (2020). ISSN: 2072-4292. DOI: 10.3390/rs12071204. URL: https://www.mdpi.com/2072-4292/12/7/1204 (cit. on pp. 8–10).

[6] Wikipedia. *Comparison gallery of image scaling algorithms*. URL: https://en.wikipedia.org/wiki/Comparison_gallery_of_image_scaling_algorithms (cit. on p. 11).

[7] R. Soundararajan A. Mittal and A. C. Bovik. *Making a Completely Blind Image Quality Analyzer, submitted to IEEE Signal Processing Letters*. 2012 (cit. on p. 12).

[8] John McCarthy, Marvin L. Minsky, Nathaniel Rochester, and Claude E. Shannon. «A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955». In: *AI Magazine* 27.4 (Dec. 2006), p. 12. DOI: 10.1609/aimag.v27i4.1904. URL: https://ojs.aaai.org/index.php/aimagazine/article/view/1904 (cit. on p. 13).

[9] Tom M. Mitchell. *Machine Learning*. New York: McGraw-Hill, 1997. ISBN: 978-0-07-042807-2 (cit. on p. 14).

[10] Wikipedia. *Machine Learning*. URL: https://en.wikipedia.org/wiki/Machine_learning (cit. on p. 14).

[11] Valohai. *What is a Machine Learning Pipeline?* URL: https://valohai.com/machine-learning-pipeline/ (cit. on p. 14).

[12] F. Rosenblatt. «The perceptron: A probabilistic model for information storage and organization in the brain.» In: *Psychological Review* 65.6 (1958), pp. 386–408. ISSN: 0033-295X. DOI: 10.1037/h0042519. URL: http://dx.doi.org/10.1037/h0042519 (cit. on p. 17).

[13] Inc. Cornell Aeronautical Laboratory. *Mark I Perceptron operators' manual*. Feb. 1960. URL: https://ia600106.us.archive.org/11/items/DTIC_AD0236965/DTIC_AD0236965.pdf (cit. on p. 17).

[14] Quora. *What is the differences between artificial neural network (computer science) and biological neural network?* URL: https://www.quora.com/What-is-the-differences-between-artificial-neural-network-computer-science-and-biological-neural-network (cit. on p. 18).

[15] Wikipedia. *Perceptron*. URL: https://en.wikipedia.org/wiki/Perceptron (cit. on p. 20).

[16] IBM. *Recurrent Neural Networks*. URL: https://www.ibm.com/cloud/learn/recurrent-neural-networks (cit. on p. 21).

[17] InteractiveChaos. *Gradient Descent*. URL: https://interactivechaos.com/en/node/910 (cit. on p. 22).

[18] IBM. *Convolutional Neural Networks*. URL: https://www.ibm.com/cloud/learn/convolutional-neural-networks (cit. on p. 25).

[19] Muhamad Yani, S Irawan, and M.T. S.T. «Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail». In: *Journal of Physics: Conference Series* 1201 (May 2019), p. 012052. DOI: 10.1088/1742-6596/1201/1/012052 (cit. on p. 27).

[20] run:ai. *Understanding Deep Convolutional Neural Networks*. URL: https://www.run.ai/guides/deep-learning-for-computer-vision/deep-convolutional-neural-networks/ (cit. on p. 28).

[21] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. «Dive into Deep Learning». In: *arXiv preprint arXiv:2106.11342* (2021) (cit. on p. 29).

[22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML] (cit. on p. 29).

[23] Medium. *A review of Generative Adversarial Networks - Part 1*. URL: https://medium.com/analytics-vidhya/a-review-of-generative-adversarial-networks-part-1-a3e5757a3dc2 (cit. on p. 30).

[24] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. *Image Super-Resolution Using Deep Convolutional Networks*. 2015. arXiv: 1501.00092 [cs.CV] (cit. on pp. 33, 34).

[25] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. *Enhanced Deep Residual Networks for Single Image Super-Resolution*. 2017. arXiv: 1707.02921 [cs.CV] (cit. on pp. 34–36).

[26] Christian Ledig et al. *Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network*. 2017. arXiv: 1609.04802 [cs.CV] (cit. on pp. 34–36).

[27] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. *ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks*. 2018. arXiv: 1809.00219 [cs.CV] (cit. on pp. 37–39, 41, 53, 68).

[28] Alexia Jolicoeur-Martineau. «The relativistic discriminator: a key element missing from standard GAN». In: *CoRR* abs/1807.00734 (2018). arXiv: 1807.00734. URL: http://arxiv.org/abs/1807.00734 (cit. on pp. 38, 52).

[29] Microsoft. *Microsoft Bing Maps Platform APIs Terms Of Use*. URL: https://www.microsoft.com/en-us/maps/product (cit. on p. 43).

[30] Sinergise. *Documentation of sentinelhub Python package*. URL: https://sentinelhub-py.readthedocs.io/en/latest/ (cit. on p. 47).

[31] Zhongyuan Wang, Kui Jiang, Peng Yi, Zhen Han, and Zheng He. «Ultra-dense GAN for satellite imagery super-resolution». In: *Neurocomputing* 398 (2020), pp. 328–337. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.03.106. URL: https://www.sciencedirect.com/science/article/pii/S0925231219314602 (cit. on p. 48).

[32] Litu Rout, Saumyaa Shah, S Manthira Moorthi, and Debajyoti Dhar. *Monte-Carlo Siamese Policy on Actor for Satellite Image Super Resolution*. 2020. arXiv: 2004.03879 [cs.CV] (cit. on p. 48).

[33] Xintao Wang, Ke Yu, Kelvin C.K. Chan, Chao Dong, and Chen Change Loy. *BasicSR: Open Source Image and Video Restoration Toolbox*. https://github.com/xinntao/BasicSR. 2020 (cit. on p. 48).