SAPIENZA
UNIVERSITÀ DI ROMA

Segmentation of satellite maps for the purpose of extracting statistics through Deep Learning

Department of Computer Science
Department of Computer, Control and Management Engineering
Department of Statistics
Department of Information Engineering, Electronics and Telecommunications

Corso di Laurea Magistrale in Data Science
Candidate

Alberto Piva
ID number 1757137

Thesis Advisor

Filomena Maggino

Co-Advisor

Francesco Pugliese

Academic Year 2017/2018

Thesis defended on 12 January 2019
in front of a Board of Examiners composed by:

Prof. ... (chairman)
Prof. ...
Prof. ...
Prof. ...
Prof. ...
Prof. ...
Prof. ...

Segmentation of satellite maps for the purpose of extracting statistics through Deep Learning
Master thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: piva.1757137@studenti.uniroma1.it

Abstract

The development of technology and the consequent increase in data production have led to an ever-greater search in the field of **Big Data**. Big Data more generally, are huge amounts of data, which are generated every day by any software or electronic device, from mobile phones to social networks. The potential of Big Data is enormous, but only recently have been developed algorithms to use all this information that can process, analyze and find objective feedback on different issues, bringing increasingly significant results.
The objective of this work, born and developed within Istat, was focused on the segmentation and classification of satellite images of urban or extra-urban territories to calculate the *land cover*, ie the percentage of land occupied by buildings civil or military, percentages of plantations, trees, etc. Python programming was the basis of the entire project.
To be able to undertake this process they were first analyzed through articles and then used different architectures of Deep Learning models. **Deep Learning** is a sub-category of Machine Learning that is based on a particular type of data learning and is inspired by the way the human brain processes information and learns thanks to neurons. It is characterized by the creation of a multi-level machine learning model, called neural networks, in which the deeper levels take inputs from the previous levels, abstracting and transforming them more and more. Neural networks are trained thanks to the computing power of modern GPUs.
The dataset used for the main work was provided by DigitalGlobe, via the kaggle platform, which includes 1km x 1km high resolution images taken directly from the satellite. The categories to be classified are: buildings, artificial structures, roads, tracks or highways, trees, waterways, lakes or seas, small vehicles and large vehicles.
The project consists of two different sections: the first purely theoretical and the second most practical. The first describes the computer vision in its entirety, starting with the automatic recognition of objects, focusing in particular on the analyzes made for the recognition, classification and segmentation of satellite images. When analyzing data, it is important to study the source environment and the research history. We continue with an in-depth study on Deep Learning, which is the main tool with which we can work within the computer vision.
The second part is more about image analysis. We speak first of the classification in one of the ten categories mentioned above and then the recognition of the areas

within it. This type of analysis allows you to identify the land cover, or the physical characteristics of the territories and then the land use, or how people exploit and use those territories.
The research applications of this can be developed following several paths, from the simplest statistics that can be collected by calculating the different percentages of occupation of the territory, highlighting the possible neighborhoods buildings-rivers or crops-rivers, up to go more specifically , managing to divide the different types of building (houses, condominiums, offices, etc. ..) with a more detailed dataset. With images on a time scale it would also be possible to calculate several indices such as the increase of buildings in a developing country, the calculation of deforestation and its effects and the mapping of regions damaged due to a natural disaster.

The stratum of the thesis is as follows:

The **first chapter** will deal with the extraction of statistics and the Computer Vision, focusing in particular on the history of satellite image analysis.

The **second chapter** talks about the neural networks. Will be introduced all the particular architectures of the models used and all that is related to the regularization of the models.

The **third chapter** concerns Deep Learning with a particular look at the improvements it has brought.

The **fourth chapter** speaks about segmentation and introduces all the improvements it brings.

In the **fifth chapter** the results obtained from the classification and the statistics calculated on sample images will be presented.

In the chapter **future directions** will be described the possible implementations that the model described may undergo in the case of similar studies.

Contents

1 Remote Sensing **1**

1.1 Computer Vision 1

1.2 Supervised learning, unsupervised learning and reinforcement learning 3

1.3 Remote sensing history 5

1.4 Official statistics 7

1.5 Applications 9

1.6 Remote sensing datasets 14

2 Artificial Neural Networks **18**

2.1 Beginning of Artificial Neural Networks 18

2.2 Multilayer Perceptrons 20

2.3 Batch Normalization 21

2.4 Activation functions 21

2.5 Early stopping 27

2.6 Hyperparameters 28

2.7 Neural Networks 29

2.8 Similarities between neural networks and the human visual cortex . 30

2.9 Convolutional layers 33

2.10 Pooling layers 35

2.11 Non-linear layers (ReLU) 35

2.12 Fully connected layers 36

2.13 Net examples 37

2.14 False Positive Reduction 40

3 Deep Learning **42**

3.1 Brief introduction to Deep Learning 42

3.2 Raising of Deep Learning 46

3.3 Tools and libraries 52

4 Segmentation and Automatic Extraction of Statistics from Satellite Maps 55

4.1 Segmentation 55
4.2 U-Net 56
4.3 Data Augmentation 58

5 Results 59

5.1 False Positive Reduction Experiment 59
5.2 Segmentation software 62
5.2.1 Segmentation 62
5.2.2 Train 63
5.2.3 Test 65
5.2.4 Classification 66

6 Conclusions and Future Directions 68

Bibliography 70

Chapter 1

Remote Sensing

1.1 Computer Vision

Human beings have always relied on sight to understand the complexity of the world, interacting with it simply by opening their eyes. For example, given the image as shown in Figure 1.1, it is easy for anyone to see a cat represented and then categorize and locate the cat in the image (classification plus location activity); locate and label all the animals in the image (object detection) and segment the individual animals that are present in the image (segmentation). The Computer Vision is the science that aims to give first the same ability of the man to the machine and then improve it.

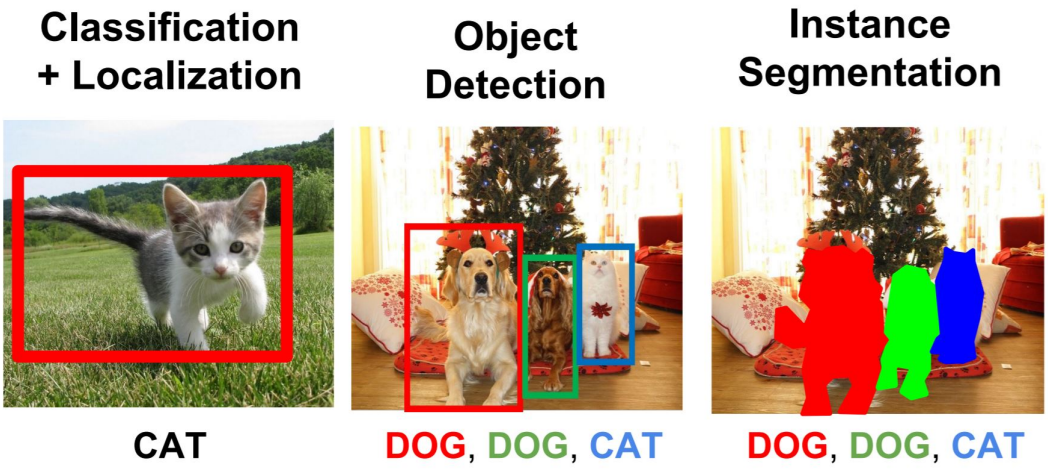

Figure 1.1. Various Computer Vision tasks
(source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf)

Specifically, the main task of Computer Vision is to develop systems and methods able to emulate a property of the human neural visual system, ie to infer the characteristics of the real world in three dimensions using only light, which through the photons objects they reflect on the eye. However, being able to reproduce the complexity of the three-dimensional structure of the world, starting from images that can only offer two dimensions captured digitally, is a particularly complicated work.

Recent studies[1] have revealed mathematical techniques capable of reproducing the three-dimensional shape and the space within the image with the various subjects. A three-dimensional object can in fact be rebuilt from a set of photos that depict it from many different angles. In fact, thanks to the studies supported so far by Computer Vision, and exploiting the various models that make up the state of the art up to now the representation of the object is always more respectful to the original. However, the road to reach a level of understanding of objects in the real world and images that is comparable to that of man is still long.

Recently the computer vision has had a particular development, also due to the possible benefits that can be obtained. One of these is definitely the fully self-driving car project that for some years has been carrying out Tesla[2]. This idea exploits both the computer vision and the deep learning, in order to create inside the car a system of cameras that allows to recognize instantly the various obstacles and road signs that can be found during a normal drive.

Another certainly useful application is the detection of lung cancer[3]. The complexity of the operations in this case compared to those described above is significantly reduced because the images, although they are very high resolution, have only two dimensions. By breaking pixel-by-pixel the images with segmentation, in some cases it was possible to diagnose the presence of lung cancer in advance of the possibilities of the human eye. The analysis in these cases was able to catch even the most imperceptible imperfections of the investigated photo.

Machine learning is directly related to artificial intelligence and it is referred as the science that is able to learn directly from a dataset without having to be directly programmed. It is also important to say that its purpose is to provide algorithms that allow a computer to improve its accuracy and performance over time. Based on

[1] Venkatesan, R., Li, B. (2018). Convolutional Neural Networks in Visual Computing. Boca Raton: CRC Press.

[2] www.tesla.com

[3] Al-Tarawneh, M. S. (2012). Lung cancer detection using image processing techniques. Leonardo Electronic Journal of Practices and Technologies, 11(21), 147-58.

the experience and skills gained through learning, the machine itself will be able to improve each time. Starting from basic commands and developing through models based on the foundations of machine learning it will be possible to classify objects, decide which choices to take or repeat actions already carried out.

1.2 Supervised learning, unsupervised learning and reinforcement learning

There are three main different methods according to which a machine can perform machine learning, which differs in the way data are accumulated and information is classified. The three possible learning algorithms are: supervised, unsupervised and enforcement

Supervised learning is structured in such a way as to give the computer system of the machine specific and detailed information about the problem, ie data and models that have the task of providing the basic knowledge for future calculations. The data needed in order to train the models are usually inserted in the process in sequence with their own label to be able to classify them. They can be of various kinds, such as images, vectors, sounds or generic inputs. Depending on the different types of data, the model after training will be able to provide different output as response, which can vary from a single binary or multivariate label up to a matrix on a single or multiple levels. There are several architectures of supervised learning function, some of which are: Neural Networks (NN), Support Vector Machines (SVM), decision trees, Bayes classifiers, Random Decision Forests (RDF), logistic regression (LR) and kernel machines.

Unsupervised learning, differently than seen above, provides that the data entered into the model is not coded, in other words there is only the input variable with no label to help the machine classify it correctly. This type of learning offers greater freedom to the machine as it will be itself to exploit all the information it has in them, organize and learn how to model them to discover a valid distribution in order to provide the best results for the different situations that arise. Clustering is the most common unsupervised learning method. Other examples of unsupervised models are: Self-Organizing Maps (SOMs), Gaussian Mixture Models (GMM) and Hidden Markov Models (HMMs).

Halfway between supervised learning and unsupervised learning there is **semi-supervised learning**, when there is a big dataset and only one part of data is labeled.

The most complex learning system is **reinforcement learning** that is able to provide the machine with the tools necessary to improve their learning and to understand the characteristics of the surrounding environment. Some of the support elements that can be supplied to the machine are cameras, sensors or GPS systems. They allow to detect what happens in the surrounding environment and make timely choices for a better adaptation. An example of this type of learning has been previously reported with the self-driving car, which thanks to a complex system of support sensors is able to circulate recognizing the various obstacles.

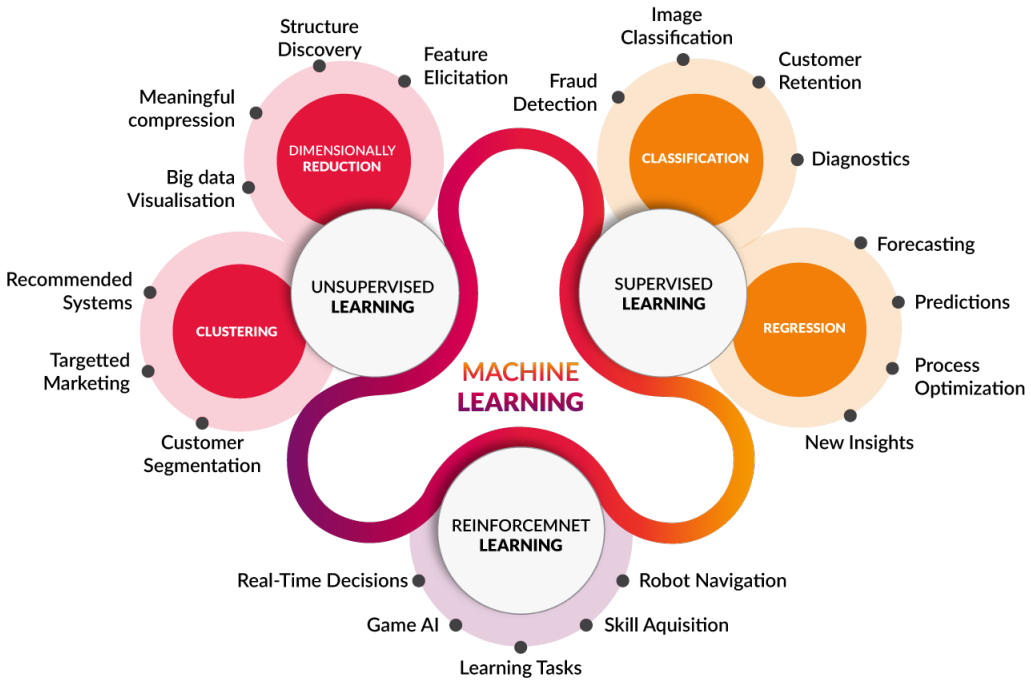

Figure 1.2. Supervised learning, unsupervised learning and reinforcement learning (source: http://www.cognub.com/index.php/cognitive-platform/)

1.3 Remote sensing history

The ultimate goal of automatic image classification is to make the mapping process faster, cheaper, more coherent and more accurate. What distinguishes automatic classification from manual interpretation is that computer algorithms rather than human algorithms determine the confluence of evidence. Automatic classification involves exploring data to establish relationships between images, ancillary data and terrain characteristics. Computer algorithms are then used to classify images based on those relationships.

If the classes in the classification scheme are strongly correlated with the images and the auxiliary data used, most of the work can be successfully automated, producing better results than manual interpretation at a much lower cost, especially for large project areas. However, due to the fact that manual interpretation is always necessary to evaluate the output of computer algorithms and to modify errors, the semiautomatic term, rather than automatic image classification, should be preferred. Semiautomatic methods classify pixels or objects, which are grouped into adjacent spatial groups of color and similar structure. The semi-automatic classification of images began when the pixel classifiers developed in the 1970s have been used to classify digital multispectral satellite images. Initially, there were only two main approaches for pixel to the traditional semiautomatic image classification: supervised and unsupervised approach.The supervised approach mimics manual interpretation, while the unsupervised approach uses statistical clustering.

There are advantages and disadvantages for each approach, and this has led many researchers to find ways to combine them in hybrid approaches. Due to the fact that the pixel classifier occupied at most three elements of the image (color, texture and date), the accuracy of the premature classification by pixel of the thematic maps was often low. In addition, users were not used to having landscapes or functions represented in pixels rather than as polygons or features. The result was that many users of the map rejected the classification of semi-automatic images in favor of long-accepted manual interpretation techniques.

In the late '80s and early' 90s, the incorporation of the position in the classification of semi-automatic images became possible with the introduction of GIS technology and the implementation of GPS technologies and algorithms that allowed precise recording of images. Since it became possible to accurately correct the image, it was also possible to incorporate the position element into the semi-automatic classification. In this way, the accuracy of pixel classifiers is often improved through

the use of location-based rule sets, which relate the position characteristics of a pixel with its probable label. For example, both water bodies and forests on the slopes of the mountains to the northeast can have very low spectral responses in all bands. As a result, it is not uncommon for forest pixels to be erroneously classified as water. But the bodies of water are flat and not noticed on the slopes. If the images are accurately recorded on the ground, you can develop a set of rules that controls that only pixels with a zero slope are classified as still waters, which distinguishes the pixels of the water from the wooded slopes.

As computers improved their processing speed and software gained more accuracy, machine learning algorithms such as random forest and support vector machines were applied to image classification. Whereas supervised and unsupervised traditional classifiers are limited to only classifying continuous data, machine learning algorithms extract images and secondary dataset to establish relationships between the classes of the map and independent continuous and categorical variables, resulting in the development of classification rules more complex and solid.

Almost simultaneously with the advent of high spatial resolution digital images in the late 1990s, new technologies were developed to segment image areas into spectral and textually homogeneous objects and classify these segments rather than each pixel. The pixels represent the arbitrary boundaries of the rectangles on the ground. The segments outline a significant variation across the landscape, which can therefore be linked to the classification scheme. Although powerful in classifying moderate-resolution images (for example, those taken by Landsat and Sentinel satellites), object-oriented classification is critical to the semi-automatic classification of high-resolution aerial or satellite images due to the combination of lighted shadows and pixels within objects and the common need to group pixels together to map vegetation classes (eg forests) instead of individual characteristics (eg trees).

At the beginning of the last century, lidar technologies became operational for topographic mapping. Suddenly, high density DEM and digital surface models (DSM) and their products have become readily available, allowing semi-automatic approaches to incorporate height and finally be able to use all the image elements available for manual interpretation. The classification process groups the data into classes with similar characteristics. Classification algorithms used to automatically classify pixels or objects are very diverse and range from simple set of rules developed heuristically, to sophisticated statistical techniques, to relatively new and complex machine learning methods.

1.4 Official statistics

Starting from September 2016, the National Institute of Statistics (Istat) has launched a new line of research and development based on Deep Learning, with the aim of:

1. Studying and implementing Deep Learning techniques.
2. Investigate the potential of Deep Learning for Statistical Institutes, with a special focus on Big Data projects.
3. Identify promising use cases for the application of Deep Learning to the context of production.

In Istat, Deep Learning has a multiplicity of potential applications since the characteristics extracted automatically from the data can allow the generation of completely automated and massive statistics. Once the satellite images have been acquired (Satellite Imagery Data Sources), to extract the information contained within them, it is necessary to detect the features and understand the structured semantics that are inside the images themselves.

To achieve the goal it is necessary to do two things: calculate the *land cover* and the *land use*:

Land Cover is an automatic processing process that aims to detect the physical characteristics of the territory and to estimate the percentage of land occupied by a certain category of entities: vegetation, residential buildings, industrial areas, forest areas, rivers, lakes, etc.

After the Land Cover calculation, the **Land Use** can be calculated, which is represented by the human activities that take place in the portion of the territory studied. In particular, Land Cover data are combined with census, statistical and numerical data to understand how individuals are using the territory.

Both are clearly linked because changes in land use can change the physical characteristics of the soil, and the other way. The advent of Deep Learning, in the context of Machine Learning, has favored the development of the Convolutional Neural Networks (CNN) as one of the most effective and robust methodological frameworks

in the advanced analysis of large quantities of Big Data Visuals, not structured and heterogeneous. In particular, CNNs are used for the entire production process chain that regards the aggregated data starting from the images.
The two main processes in which the convolutive networks are adopted are:

1. Classification: which consists in the identification of the categories of objects that are in a satellite image [4]: ex. trees, industrial buildings, residential buildings, plantations, etc. VGG-net [5], Wide Residual Networks [6] or the most recent Capsule Networks [7], which are all types of advanced and deep convolutive architectures, are used for the actual classification of image segments.

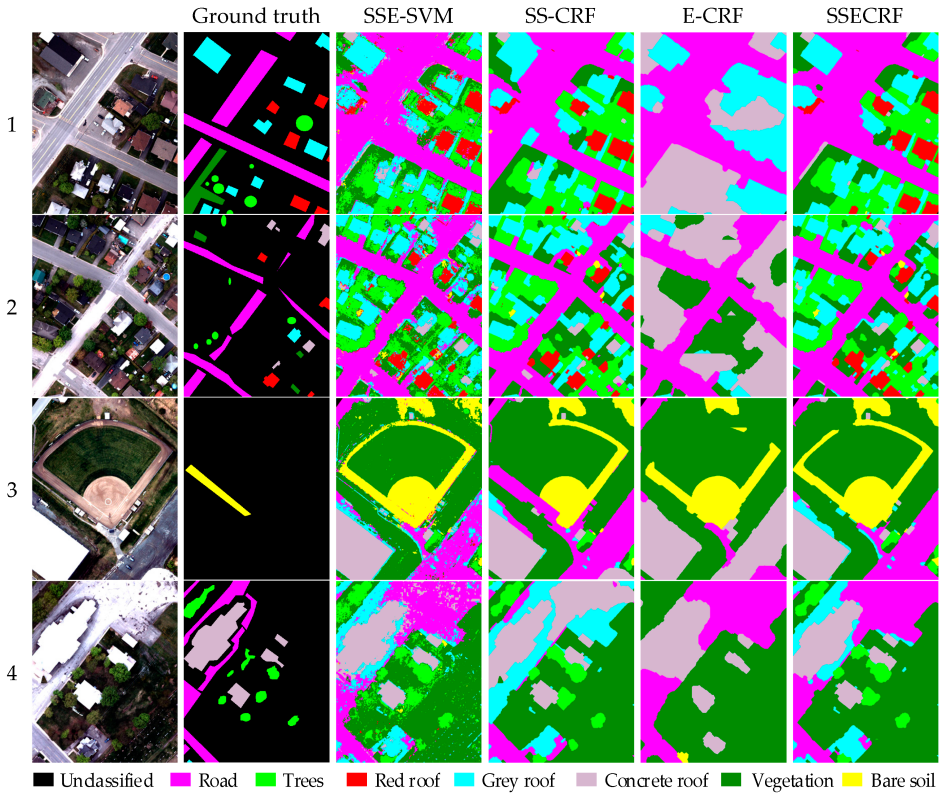

Figure 1.3. Satellite image segmentation techniques and classification - source: Zhong, Y., Jia, T., Zhao, J., Wang, X., & Jin, S. (2017).

[4] Helber, et. al., 2017.
[5] Simonyan, & Zisserman, 2014.
[6] He, et. al., 2016
[7] Sabour, et. al., 2017

2. Segmentation: which consists in the subdivision of the images to identify the areas of the portions of the soil to which the identified classes can be associated or to focus the classifier on specific "features" (discriminating characteristics) of the images. Figure 1.3 shows a typical example of segmentation including also classification. For the segmentation process it is possible to use the same CNN in the "Fully Connected" version [8] or the Fast R-CNN [9] or the U-net [10] which are a specialization of the CNN

1.5 Applications

Computer Vision, together with artificial intelligence and in particular Deep Learning with its convolutive neural networks, are the perfect combination for recognizing and classifying objects within digitally acquired images.
To underline the importance of Computer Vision, consider the percentage of accuracy that dermatologists have in recognizing a benign skin cancer, from a malignant one. According to a study done by a group of researchers from France, United States and Germany, dermatologists identified 86.6% of melanomas, while an algorithm based on convoluted neural networks has guessed in 95% of cases. This is a case in which artificial intelligence surpasses the human being.

Turning to the analysis of satellite images, the main areas of research of the computer vision are:

Change Detection: For a certain period of time, the satellite takes images of the exact same place, so as to make a comparison between the images of the past and the present, and then also to predict how the same environment will change in the future. This makes it possible to identify automatically the patterns of features that reveal the occurrence of phenomena of landscape change. For example, Figure 1.4 shows two photographs taken from the Landsat 5 satellite, left in 1984 and right in 2011. The expansion of the city of Denver, Colorado, and the construction of Denver International Airport is shown here. In this context, artificial intelligence has measured the rate of construction of the country linked to its economic growth and has done so completely automatically thanks to the information captured by the images with the Computer Vision.

[8] Castelluccio, et. al., 2015
[9] Girshick, 2015
[10] Ronneberger, et. al., 2015

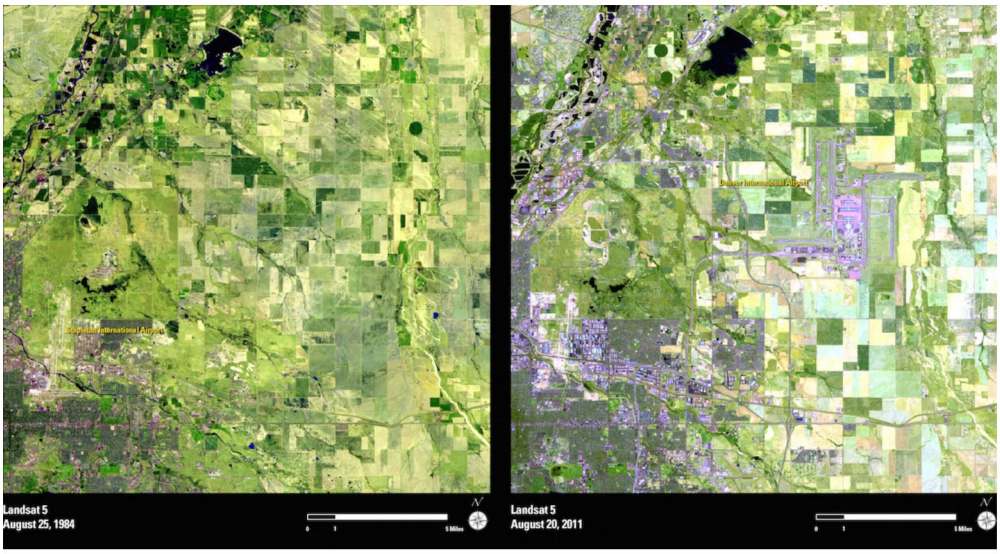

Figure 1.4. Change detection in Denver, CO - credits: USGS/NASA

Urban development: the Orbital Insight company operating in San Francisco is very active for projects involving Deep Learning and Computer Vision. The Californian company has a team of aircraft that sends on the cities to monitor the places that present problems to study and solve. For example, in Figure 5, the city of Nanjing in China was photographed to perform the car counting in order to plan and improve traffic management.

Figure 1.5. Car-counting in Nanjing, China - credits: DigitalGlobe/Orbital Insight

Environmental monitoring: Orbital Insight is also testing and implementing Deep Learning algorithms that analyze high resolution satellite images to detect patterns that could indicate the imminent deforestation. The system that the company is currently developing is called Global Forest Watch (GFW). It is a monitoring and reporting system that is accessible to all, enabling people all over the world to receive and share the information they need to help and conserve forest landscapes. Global Forest Watch uses Deep Learning technology to provide timely and accurate health information on forest landscapes such as the one shown in Figure 1.6, in Uganda. GFW is also able to give alarms almost in real time regarding large areas of a forest where a recent deforestation has been probable.

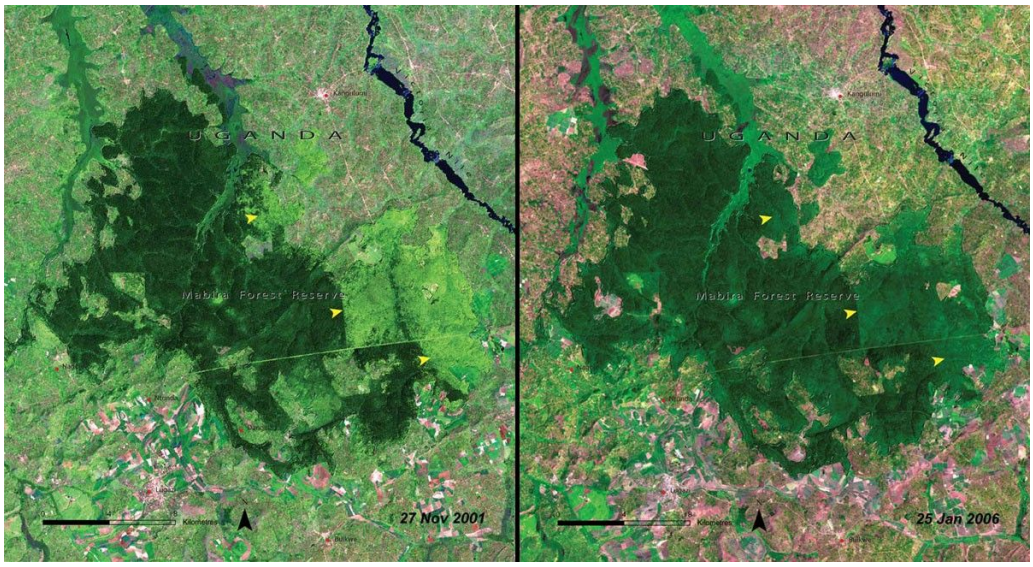

Figure 1.6. Deforestation in Uganda - credits: DigitalGlobe/Orbital Insight

Detection of Landslides: The Sudden Landslide Identification Product, SLIP, program of NASA, uses satellite imagery and data regarding rainfall and soil conditions to determine where a landslide is likely to occur. This project is currently focusing on Nepal, where high amounts of rainfall in a short period of time combined with steep hillsides are major factors that combine to create a high amount of risk to local agricultural efforts. The project uses images from the Landsat 8 satellite as well as information on localized topography from the Shuttle Radar Topography Mission and the Advanced Spaceborne Thermal Emissions and Reflection Radiometer.(Borneman, 2016). In figure 1.7, the image processed on the right shows the areas in red indicating a likely landslide and the yellow areas showing a possible landslide.

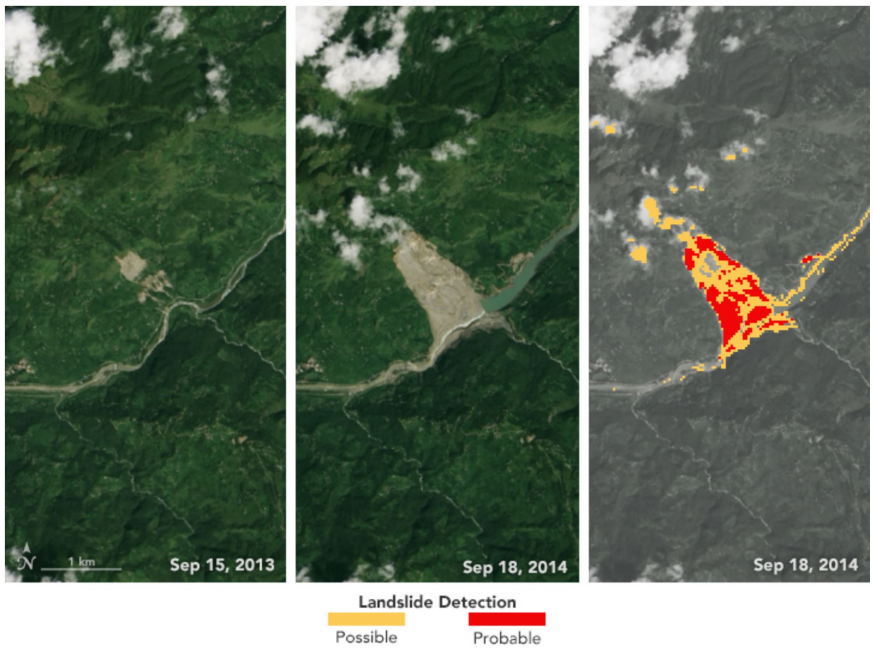

Figure 1.7. Landslide detection in Nepal - credits: NASA

Natural disasters: the data and skills of NASA provided valuable information for the response underway to the earthquake in central Italy in Italy on 24 August 2016, magnitude 6.2. The earthquake has caused significant damage in the historic city of Amatrice. To help disaster response efforts, NASA, the California Institute of Technology, in collaboration with the Italian Space Agency (ASI), has generated this image of the region most affected by the earthquake.
The Damage Proxy Map (DPM) (65-for-120 km) was derived from two consecutive frames of the interferometric radar with synthetic aperture (InSAR) of the Japanese aerospace band (JAXA) satellite data from the ALOS -2, and the DPM (40-for-50 km) was derived from InSAR data from the COSMO-SkyMed X-band satellite of the Italian Space Agency (red rectangle). Both DPMs cover the historic city of Amatrice, revealing serious damage in the western part of the city (panels on the right). Each pixel in the damage proxy map is about 30 meters in diameter. The SAR data were developed by the Advanced Rapid Imaging and Analysis (ARIA) team at JPL Caltech.
The technique uses a prototype algorithm to quickly detect surface changes caused by natural or man-made damage. The evaluation technique is the most sensitive to the destruction of the built environment. When the radar image areas have minimal or no destruction, the pixels in the image are transparent. The increase in the

opacity of the radar image pixels reflects the damage, with the red areas reflecting the heaviest damage to cities. The color variations from yellow to red indicate an increasingly significant change in the surface of the ground.

Preliminary validation was performed by comparing the DPMs with a damage assessment map produced by the Copernicus emergency management service, which is based on the visual inspection of high resolution aerial images before and after the indicated extension with gray squares in the left panel.

ARIA is a project funded by JPL and NASA developed by JPL and Caltech. It is building an automated system to provide fast and reliable GPS and satellite data to support local community monitoring and response to hazards locally, nationally and internationally. Using spatial images of disasters, ARIA data products can provide rapid assessments of the geographic region affected by a disaster, as well as detailed images of the locations where the damage occurred.

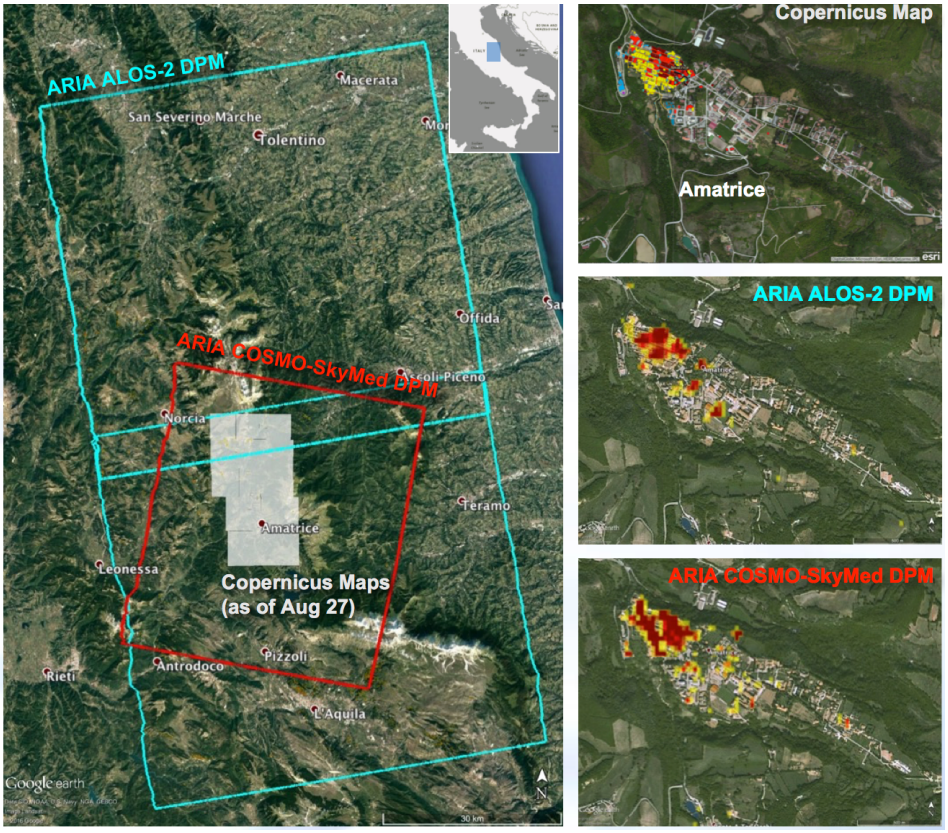

Figure 1.8. Analysis of the earthquake in Amatrice in 2016 - credits: NASA

1.6 Remote sensing datasets

As often happens in supervised machine learning, the performance of classification systems depends heavily on the availability of high quality data with an appropriate set of classes. In particular, considering the recent success of the deep neural networks (CNN) in this kind of problems, it is essential to have a large amount of training data to train this network. Unfortunately, the current public land cover and land use data sets are small-scale or based on data sources that do not allow the aforementioned domain applications.
Currently government programs such as ESA's Copernicus (European Space Agency) and NASA's Landsat (National Aeronautics and Space Administration) are making significant efforts to make such data freely available for commercial and non-commercial purposes with the intention of fueling innovation and entrepreneurship. Trying to improve the situation, various researchers have used Google Earth, which is not freely available, to manually create new datasets. These collections use images with a spatial resolution up to 30 cm per pixel. Since the creation of a labeled dataset is extremely time-consuming, these also consist of a few hundred images per class.
Under the Copernicus program, ESA operates a number of satellites known as Sentinels. Since the launch of Sentinel-1A in 2014, the European Union has initiated a process to put in orbit a constellation of about 20 satellites before 2030. Each sentinel takes care of a service. The six thematic flows of Copernicus services are monitoring the atmosphere, climate, marine waters, territories, climate, emergencies and security.
Data are also accumulated thanks to sensors placed on the ground, in the sea and in the atmosphere with which they communicate the Sentinels, generating topographic and satellite maps statistics. The most important thing is that the data is openly and freely accessible and can be used for any application (commercial or non-commercial use). By making his data available freely, Copernicus has had the overall impact of generating the creation of new markets and new jobs.
It is estimated that in 2030, thanks to this program, there will be 48,000 new jobs that will bring innovations and economic growth.
The data used to calculate the land cover and land use in this project are derived from WorldView-3 Satellite Sensor (0.31m) and provided by the platform kaggle. The WorldView-3 satellite sensor was licensed by the National Oceanic and Atmospheric Administration (NOAA).

The WorldView-3 satellite was successfully launched on August 13, 2014. It is the first multi-payload, super-spectral, high-resolution commercial satellite sensor operating at an altitude of 617 km. WorldView-3 satellite provides 31 cm panchromatic resolution, 1.24 m multispectral resolution, 3.7 m short wave infrared resolution (SWIR) and 30 m CAVIS resolution. The satellite has an average revisit time of <1 day and is capable of collecting up to 680,000 km^2 per day [11].
The dataset for the project is provided with 25 satellite images with shape 1km x 1km in both 3-band and 16-band formats. The 3-band images are the traditional RGB natural color images. The 16-band images contain spectral information by capturing wider wavelength channels. This multi-band imagery is taken from the multispectral (400 – 1040nm) providing 8 band (red, red edge, coastal, blue, green, yellow, near-IR1 and near-IR2) and short-wave infrared (SWIR) (1195-2365nm) range with other 8 band.
The main purpose of the work is to be able to correctly classify the various categories that can appear or not within an image, which are:

1. **Buildings** - large building, residential, non-residential, fuel storage facility, fortified building
2. **Misc. Manmade structures**
3. **Road**
4. **Track** - poor/dirt/cart track, footpath/trail
5. **Trees** - woodland, hedgerows, groups of trees, standalone trees
6. **Crops** - contour ploughing/cropland, grain (wheat) crops, row (potatoes, turnips) crops
7. **Waterway**
8. **Standing water**
9. **Vehicle Large** - large vehicle (e.g. lorry, truck,bus), logistics vehicle
10. **Vehicle Small** - small vehicle (car, van), motorbike

In order to classify correctly the different categories within the image, there is another document that allows the segmentation of each single element of each class.

[11] Satellite Imaging Corporation

This document provides for every object class a simple list of polygons, in which one or more elements are indexed through the coordinates of the corners within the image in the form of "Polygons" or "MultiPolygons".

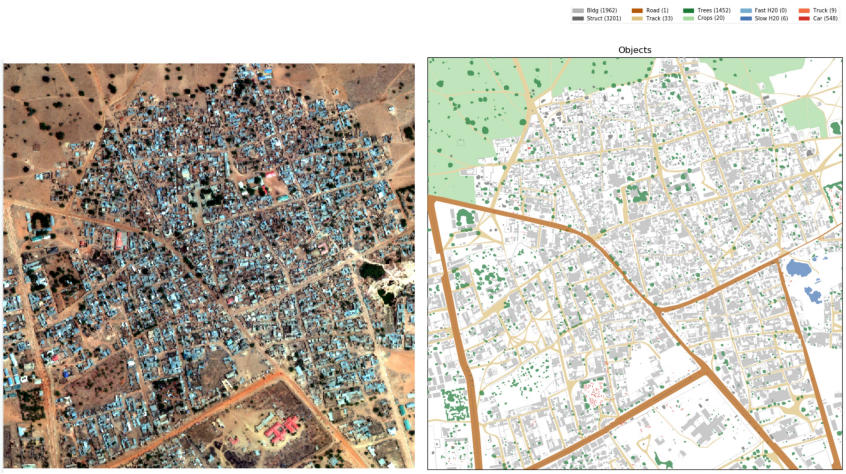

Figure 1.9. Satellite image with polygons representation

In Figure 1.9 it is possible to notice one of the satellite image with the corresponding representation in the form of polygons. It is also important to note by observing the representation of the polygons of the photo that not all categories appear in every single satellite image. In this case there are no waterways. As a result, all classes have different occurrences within the single image and the entire dataset. Figure 1.10 shows the different percentage of area covered in each satellite image by the different categories.

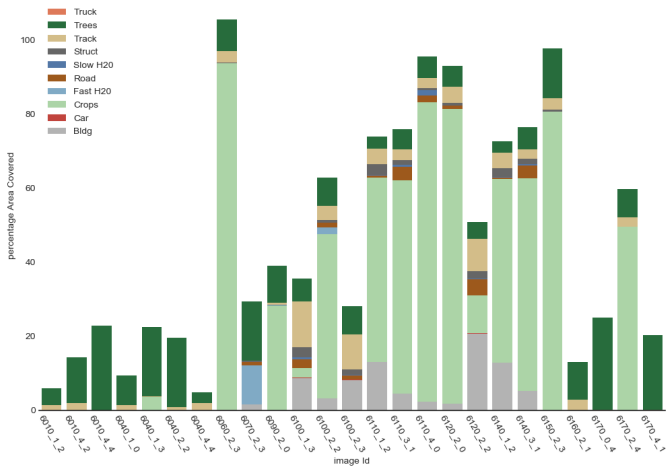

Figure 1.10. Percentage of area covered for each image

To ensure privacy with respect to the dataset, a set of geographic coordinates were created in the range of x = [0,1] and y = [-1,0]. These coordinates are transformed in such a way as to obscure the position from which the satellite images are taken since they come from the same region on Earth.

Chapter 2

Artificial Neural Networks

2.1 Beginning of Artificial Neural Networks

The introduction of artificial neural networks (ANN) as a scientific application of the generalization of neural activity in the logical mathematical form occurred in 1934. [1]

The model introduced is called 'neural networks' and is expressed by the authors through a mathematical formula in different variables. Neural networks are in fact a maximum simplification of the nervous system of a human being and, depending on the different interpretation, they can have a different development and structure according to the different application contexts.

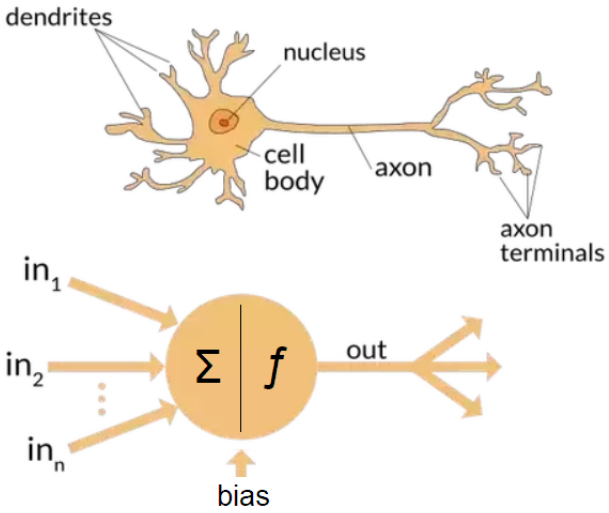

Figure 2.1. Biological and artificial neuron - credits: Nagyfi,R. (2018)

[1] McCulloch, W., S., and Pitts, W., (1943).

The different structures of the neural networks are due to the number and type of connections between the neurons, which constitute them. The neuron, more specifically, is defined by an N number of inputs and an M number of outputs, with two variables, ϕ and θ, that indicate respectively the activation function and the threshold, described in 2.1.
The neuron j^{th} is activated or not depending on the different signals it receives, according to the formula:

$$A_j = \sum_{i=1}^{N} w_{ij} X_i - \theta_i \qquad (2.1)$$

The formula above concerns only the weighted sum of the activations of the single neurons, where in specific w_i is the weight attributed to the connection between the i^{th} and j^{th} neurons, X_i is the input received from the i^{th} neuron and θ_i is the bias of the i^{th} neuron, also known as activation threshold.
The following formula, on the other hand, describes the activation function at the output of the j^{th} neuron:

$$y_j = \phi(A_j) = \phi\left(\sum_{i=1}^{N} w_{ij} X_i - \theta_i\right) \qquad (2.2)$$

In the formula 2.2, ϕ indicates the activation function. The task of the activation function is to define the response given by the neuron according to the different stimuli received. The bias ϕ, in some cases is initialized to a constant value of 1, to act as a weight within the system, relative to a virtual input called X_0. Activation functions will be further investigated in chapter 2.4
It is possible to differentiate two different architectures of neural networks based on the types of connections between neurons:

- **Feed Forward Neural Networks**: The main feature of this architecture is the absence of loops inside the layers. This type of neural networks consists of an input layer and an output layer with a defined number of hidden layers inside. The internal levels are organized according to a hierarchical scale depending on the connections between the layers and the neurons.

- **Recurrent Neural Networks**: Unlike those described above, this type of neural networks allows recurrent connections within the same layer. In fact,

a single neuron can receive as input the same output that it had previously computed.

2.2 Multilayer Perceptrons

A perceptron is defined as a linear classifier, in other words it is an algorithm that is able to classify the input by dividing two different categories through a straight line. It was introduced in 1950 by Frank Rosenblatt[2], and it was created for the first time as an hardware instead of an algorithm, and it had been thought of as a machine that could learn.
The machine built by Rosenblatt was a single-layer perceptron. The architecture of its algorithm within the hardware was not composed of a large number of layers, which would have allowed the machine to classify and index more categories. The neural network was in fact shallow, preventing the perceptron within it to perform a series of more complex classifications, such as non-linear classifications.
The set of individual perceptrons forms the multilayer perceptron (MLP), which is a deep neural network. The MLPs are made, like any neural network, by an input layer that receives the signal and an output layer that provides the final transform. The critical difference compared to single-layer perceptrons is the presence of multiple layers that make up the core of the multilayer perceptron.
MLPs have different applications in the field of supervised learning problems: the models are trained through input and output, being able to learn the different dependencies that distinguish the different classes. In order to minimize the model error it is possible to adjust the weight of the parameters and the bias until the goal is reached.
The main tool available to the neural network to reduce the error is the backpropagation. The backpropagation, as the name suggests, consists in the back propagate through the MPL the partial derivatives of the error function with respect to the various weights and biases. The result of the previous process is a gradient, along which the parameters can be updated while the MLP approaches the minimum of the error.
The optimization algorithm continues the descent of the gradient until the best condition is reached, from which the error can not be further minimized. At the end of this process you will have the bias and the optimal parameters to be used for model training.

[2]Rosenblatt, F. (1958).

2.3 Batch Normalization

Batch normalization rather than simply normalizing the inputs of the network, normalizes the input of the layers within the network favoring the increase of the learning speed, going to reduce the overfitting by increasing the competitiveness between the layers.
What happens during this phase is substantially an increase in competitiveness of adjacent neurons between different kernels, favoring a better activation by reducing the shift of internal covariance.
The increase in competitiveness forces each layer of the neural network to train better, being more autonomous, being able to learn by using less of the other layers that surround it.
The batch normalization allows you to use much higher learning rates and without particular attention regarding the initialization of the weights.

2.4 Activation functions

In real life, the human brain often finds itself in situations where it is presented with various information in a short period of time, being forced to analyze information and classifying them as useful and not useful information.
Since neural networks are completely inspired by the human brain, they also need a specific mechanic to help classify incoming information as useful or not useful. Since information is not always useful to the network, as some are only classified as noise, having a mechanism that helps the network during the training phase is very important.
This task is performed by the activation functions that help the network make this distinction, using only useful information and deleting those irrelevant. The activation functions thus play a key role in artificial neural networks, deciding whether a neuron should be activated or not, specifically decide if the information that is going to be transmitted to the following neuron is relevant given the information provided and therefore must be kept and re-transmitted or it is not relevant and should be ignored.
At the level of the single artificial neuron, is computed a weighted sum of its input with an additional bias, then it is submitted to the activation function to decide

whether it should be dropped or held. Recalling the neuron formula (2.1):

$$A_j = \sum_{i=1}^{N} w_{ij} X_i - \theta_i$$

the activation function is a non-linear transformation that is performed on the input signal. Once processed and finally transformed as output is subsequently conducted to the upper layer of neurons as input again. The range that the transformed value can assume can vary from infinity to infinity. However, not knowing the values that input can take, the activation function is used to decide whether to propagate the output or not. The most important activation functions are presented below:

Step function: It is a threshold-based activation function; it is activated only if the value Y exceeds a certain preset value. The step function can therefore be directly related to a binary activation function [0,1] and is mainly used to implement a binary classifier in the model. In fact, when the problem requires having to classify only two classes, the best solution to simplify the network as much as possible is the step function, as it allows, depending on the case, to activate a specific neuron or to leave it at zero.
However, this function is less and less used since neural networks are almost always used for the classification of several classes, exceeding the ability of a binary classifier. The gradient of the step function is in fact equal to zero, making it practically useless in the retro propagation, when the activation functions send the gradients for the calculations of the errors.

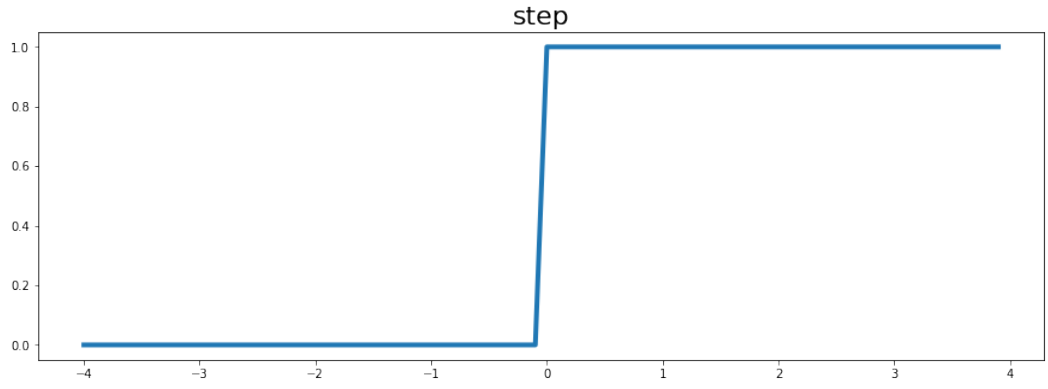

Figure 2.2. Step function

Linear function:

$$f(x) = a + bx$$

It is a straight-line function in which the activation is proportional to the input. The linear activation function allows to choose between a set of activations the one with the maximum weight.
However, the first degree derivative of this function is a constant, not completely solving the problem analyzed with the step function, resulting in a gradient that is always the same at the time of back propagation. A constant result of the gradient in fact does not help the neural network to reduce the margin of error.
Another important limit that the linear function carries with it is the inability to exploit the depth of the network. Whenever an input is transformed, the corresponding output is simply a linear transformation, substantially inhibiting the advantages of the affected layers, which can be directly replaced by a single layer.
This eliminates the need to stack layers and reduce the entire network to a single layer with linear activation.

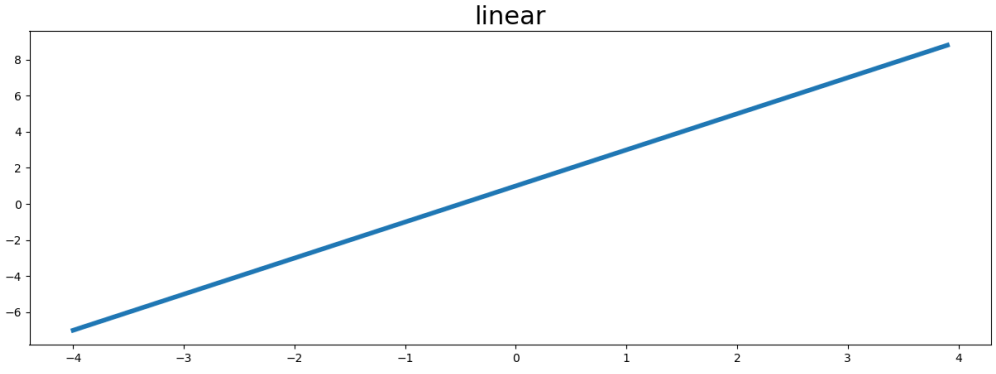

Figure 2.3. Linear function

Sigmoid function:

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

This activation function has many advantages and for this reason it is currently one of the most exploited. Unlike the previous function, it is a non-linear activation, allowing the network to stack layers by performing significant transformations.
If compared to the step function instead, it has an analog activation and an uniform gradient. In the values around zero in the abscissae axis, the function shows a rapid

rise, presenting different Y values at the minimum vary of X. The sigmoid function tends to make the Y values vary until they are brought to both ends of the curve causing considerable differences on the prediction.
The main advantage of the sigmoid function is to always have the output in the range [0,1], thus reducing the values to a precise and constant interval.
The gradient of the sigmoid function is only dependent on the value of X, which makes this function usable without problems during back propagation and the error weights are updated continuously at each iteration.
A disadvantage occurs near both ends of the sigmoid function, where the function tends to assume a more linear trend, and consequently the Y values respond to a lesser extent to the variations of X. In those determined regions of the function the gradient will be small highlighting a problem known as "vanishing gradient". This means that the gradient approaches zero and the network is not really learning. However, there are techniques to work around this problem when training the neural network.

Softmax function:

$$S(X_i) = \frac{Exp(X_i)}{\sum_{j=0}^{k} Exp(X_j)} \qquad i = 0, 1, 2...k$$

It is a particular type of the sigmoid function, but differently, it manages to calculate the probability distribution over all possible events. In fact, the main advantages in the use of the softmax function are that the output will be probabilities. Each value in output will therefore have a value between the interval [0, 1] and the sum of all the probabilities of the possible events will always be equal to one.

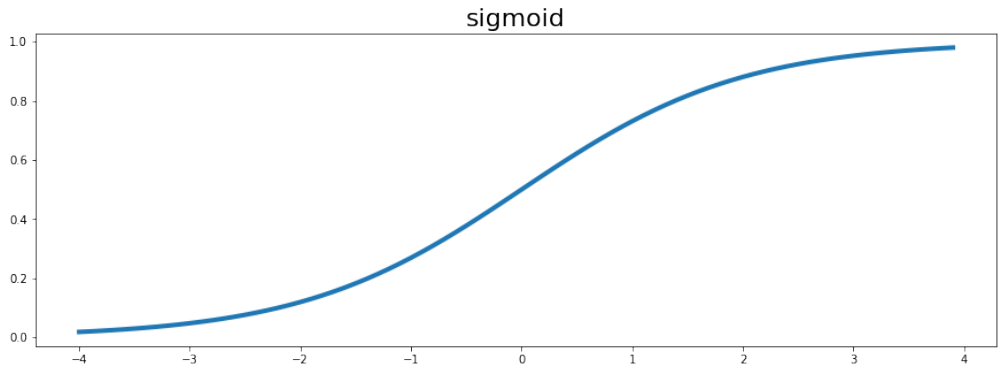

Figure 2.4. Sigmoid function

Tanh function:

$$f(x) = tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

The tanh function is a direct transformation of the sigmoid function and therefore presents its own non-linear shape with all the relative advantages that follow, allowing in fact to stack the layers and maintaining the constraints of the limited interval, albeit with different values [-1 , 1]. There are no activation problems.
The gradient for the tanh function has a stronger behavior than the sigmoid function, but still presents the problem of vanishing gradient.

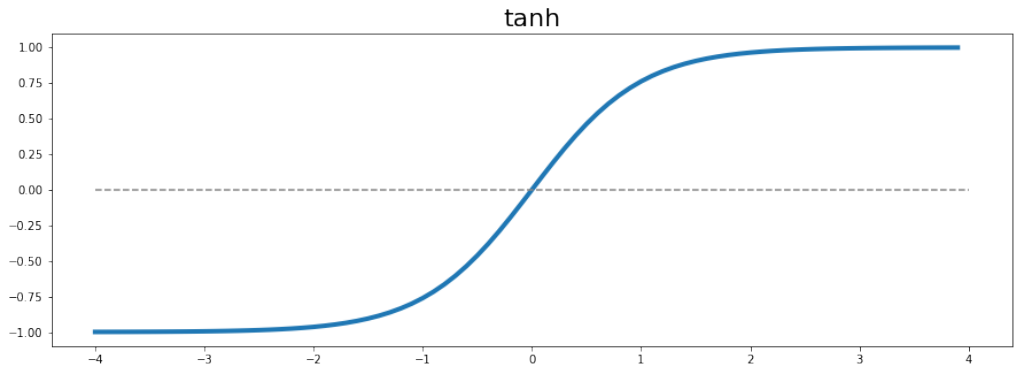

Figure 2.5. Tanh function

ReLU function:

$$f(x) = max(0, x)$$

The ReLU (Rectified Linear Unit) is a function of activating output value equal to that of the input if it is positive and 0 otherwise. Unlike what may result from a first impression, the ReLU function does not have the same problems as the linear function, since it is linear in the positive axis.
The function of ReLU activation is in fact non linear and also the different combinations of ReLU are non linear, resulting in more experiments a good approximator with all kinds of functions.
The range of the ReLU has no higher constraints, while all the values less than or equal to zero are close to zero, the positive values can also be very high [0, inf). In fact, this transformation risks being useless if misused.
A crucial element that this activation function addresses is the scarcity of activation. Given a large neural network with many neurons, the use of a sigmoid function or a tanh function will force all the neurons involved to activate in an analogical

way, resulting in a mass of information that undergo different processing at each layer with the purpose of describing the output from a network. Computationally speaking, the activation becomes very dense and the network would require a huge effort to be trained. Ideally we would like only a few neurons to be activated in the network, thus making sparse and efficient activations.
The advantage of the ReLU is precisely this, given a neural network with random weights, 50% of the network produces the zero activation given the property of the ReLU. In terms of density, the network is less heavy and fewer neurons are activated. The main problem of ReLU can occur in the moment in which the gradient is calculated, because of the negative axis of the function, in fact, the gradient can go to zero, and during the back propagation phase those weights would be no more updated.
This phenomenon can create dead neurons that are never activated and is known as "dying neurons". The main consequence of neuronal death is the possible loss of more or less extensive regions within the neural network, making it inoperative and losing some features. To reduce this problem there are variations of the base ReLU, transforming the negative component into a line that is no longer horizontal.

Leaky ReLU

$$f(x) = \begin{cases} x & if x > 0 \\ 0.01x & otherwise \end{cases}$$

It has the substantial advantage over the normal ReLU function of replacing the horizontal line and removing the zero gradient. In this case the gradient of the left side of the graph is different from zero and therefore no dead neurons will be present in that particular region.

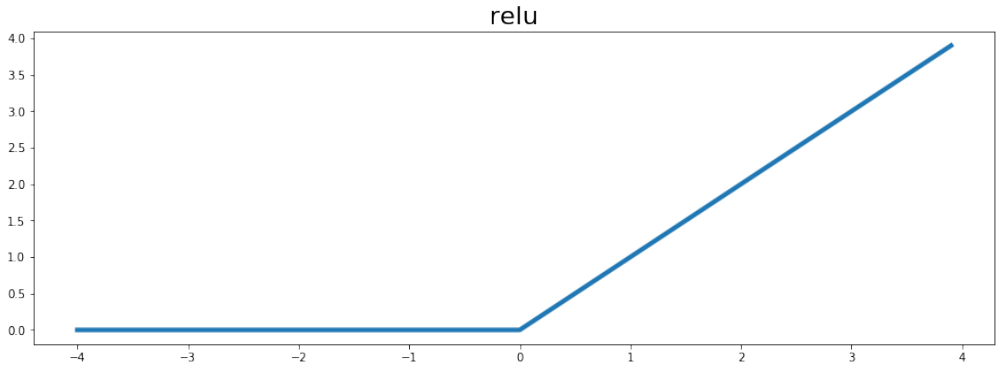

Figure 2.6. ReLU function

2.5 Early stopping

When a model has good performance on the set of training, but it drastically drops with the data not visible, it means that there is an overfitting problem. To avoid overfitting in gradient-based iterative algorithms it is applied early stopping. This is achieved by evaluating performance on a validation set maintained at different iterations during the training process.
The training algorithm can continue to improve on the training set until performance on the validation set does not improve. Once there is a decrease in the generalization capacity of the learned model, the learning process can be stopped or slowed down as shown in Figure 4.3.
Finally, best model weights are saved.

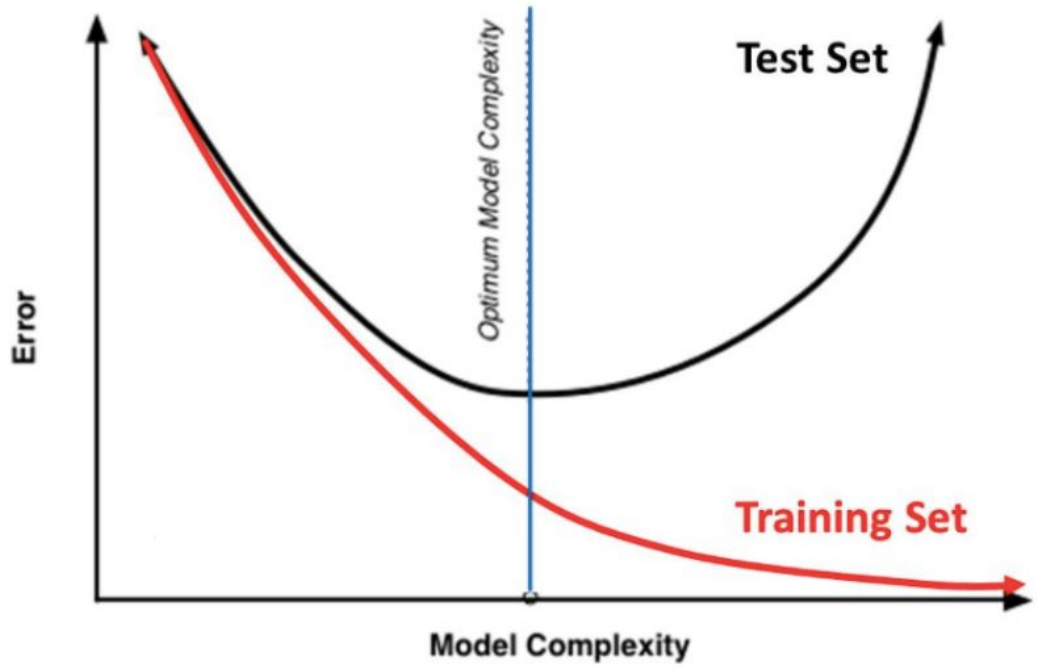

Figure 2.7. Early stopping - credits: Jain, S. (2018)

2.6 Hyperparameters

The learning rate is a hyperparameter that controls how much the network weights are adapting to the loss gradient. The lower the value, the slower the path along the descent. Using a low learning rate to make sure that no local minimum is lost, it could also mean that it would take a long time to converge, especially if it gets stuck on a plateau region.

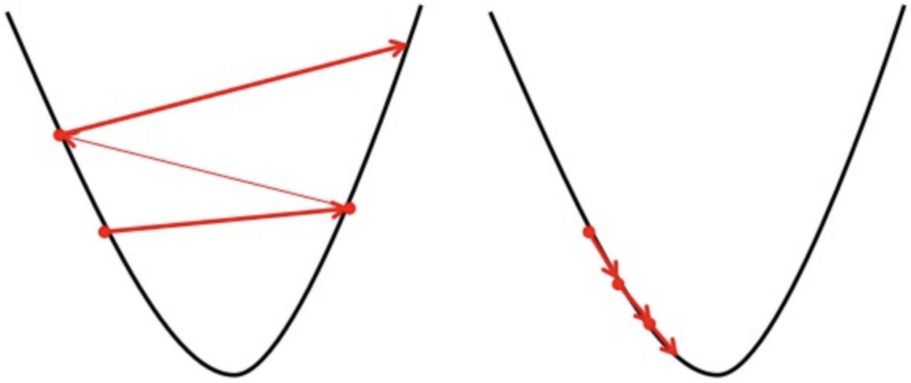

Figure 2.8. Gradient descent

Looking at Figure 2.8, given the formula

$$\theta_1 := \theta_1 - \alpha \frac{\gamma}{\gamma\theta_1} J(\theta_1) \qquad (2.3)$$

it is possible to define two different situations when alpha assumes values that are too big or too small

- When α is too big, as shown on the left side, the gradient descent can overshoot the minimum. It is also possible the it fail to converge or even diverge.
- When α is too small, on the right side, the gradient descent will be slower and may require a long time to converge.

The rate of learning directly affects the speed at which the model can converge to a local minimum. Appropriate choices made before training the model can drastically reduce the time required for the training.
During the training of a model there are a series of hyperparameters to be monitored to verify if the accuracy increases and in what percentage. Some of the most important values to monitor are the following:

Train accuracy: in the training phase it indicates the number of elements correctly classified.
Validation accuracy: in the test phase it indicates, on the basis of the rules established in the training, how many new elements have been correctly classified.
Train loss: in the training phase it indicates the number of elements not classified correctly.
Validation loss: in the test phase it indicates, on the basis of the rules established in the training, how many new elements have not been correctly classified. The loss function is related to the effectiveness of the learning rate

The accuracy of the training provides important information on the model overfitting. When the accuracy is greater than the validation accuracy, we are in the presence of overfitting and in this case we resort to regularization methods to reduce the overfitting of data

2.7 Neural Networks

Convoluted neural networks, also known as ConvNets or CNN, are a category of deep neural networks that have proven to be very effective in areas such as image recognition and classification. Figure 3.1 illustrates the relationship between artificial vision, machine learning, human vision, deep learning, and CNN.

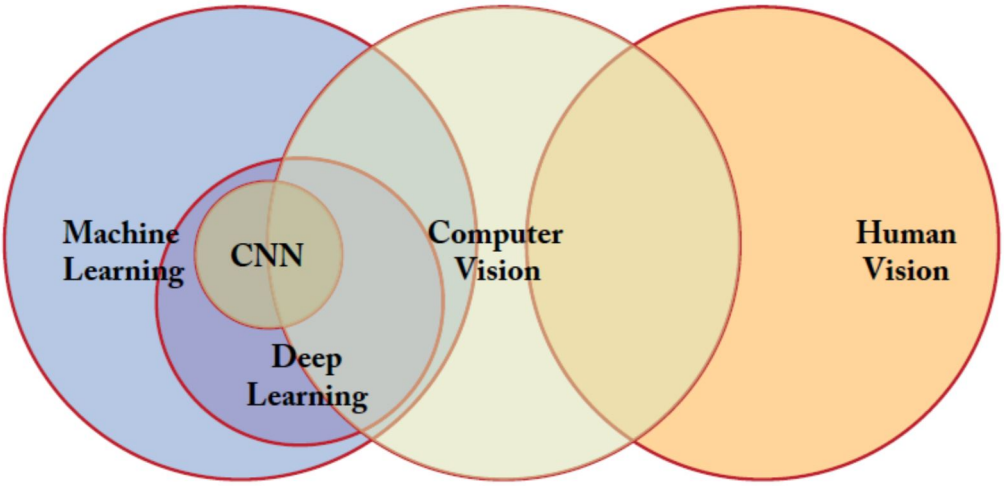

Figure 2.9. Relationship between artificial vision, machine learning, human vision, deep learning and CNN - credits: A Guide to CNN for Computer Vision.

Because of the lack of training data and computing power in the early days, it was difficult to train a large CNN with high capacity without using a turbocharger. After the rapid growth in the amount of data available and recent improvements in the strengths of the graphics processing units (GPUs), research on CNNs has emerged rapidly and has achieved cutting-edge results in various artificial vision tasks.

There are a number of reasons why convolutional neural networks are becoming important. In traditional models for model recognition, features pullers are designed by hand. In CNNs, the convolutional layer weights used for feature extraction and the fully connected layer used for classification are determined during the training process. The structures of CNN networks allow to save on memory requirements and on calculation complexity requirements.

2.8 Similarities between neural networks and the human visual cortex

The first thing humans do when they are born is to start recognizing or identifying things. The visual cortex of the brain plays an important role in this identification. In 1988 Yann Le Cun was inspired by this concept to create the first version of the convolutional neural networks.

The visual cortex obtaining a more direct signal from the eyes calculates simple characteristics such as borders or color, then sends the output of that calculation to the next visual area. Each visual area calculates a slightly more complicated feature of the area from which the input is received simply because the input it receives has already been transformed from the previous area, when these calculations reach the decisional areas of the brain the recognition of the object.

Going even more specifically, as shown in Figure 3.2, after the image is observed in the real world through the eyes, it undergoes the first transformation process at the level of the lateral geniculate nucleus (LGN), first used to analyze the characteristics of movement and depth of the visual field and then for the analysis of shapes and colors.

The processed image is then sent to the primary visual cortex (V1), which recognizes static objects and repetitive characteristics. The successive layers of the visual cortex, V2 and V4, are closely related to V1 and perform the last transformations before the recognition of the image.

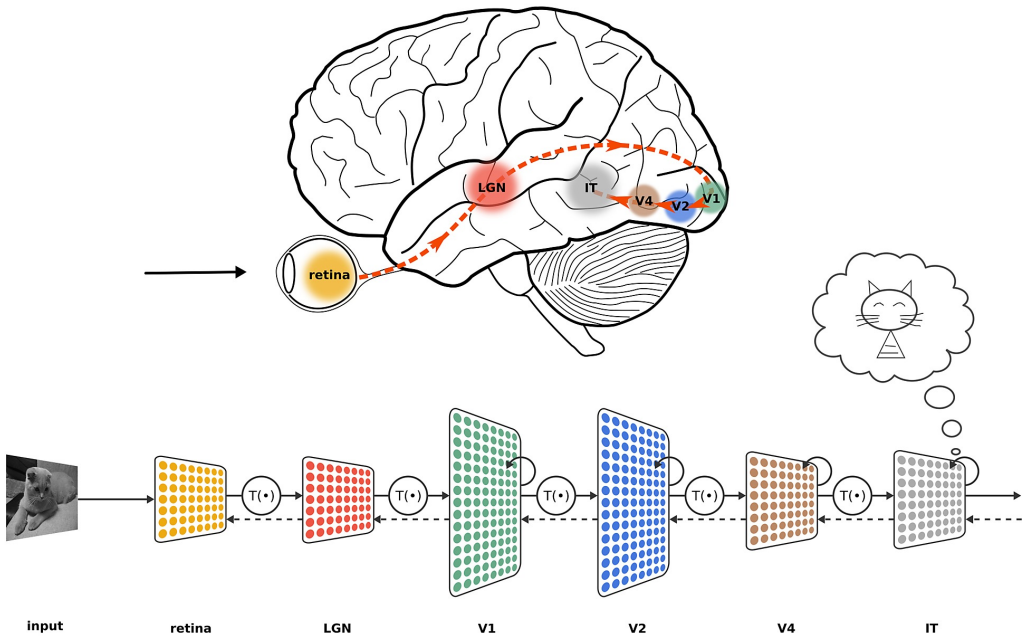

Figure 2.10. Processing steps of visual system and simple neural network - credits: Kubilius, Jonas (2017).

The visual cortex contains many cells that are responsible for detecting light in small, overlapping regions of the visual field, which are called receptive fields. These cells act as local filters on the input space and the more complex cells have larger receptive fields. The convolution layer in a CNN performs the function that is performed by cells in the visual cortex.

The neural network, as in the case of the visual cortex, is composed of multiple levels of "neurons" that detect features. Each level has many neurons that respond to different combinations of input from previous levels.

A first difference between neural networks and the visual cortex derives from the fact that neural networks can have more or less levels of abstraction within them depending on the operations they have to perform. The less deep neural networks are called "shallow" and can have only two or three layers, while the deeper ones can reach more than one hundred layers.

As shown in Figure 3.3, there is the simplest neural network, which is structured with a single hidden layer, which directly receives the inputs from the first level, processes them according to appropriate transformations and passes them directly to the output layer.

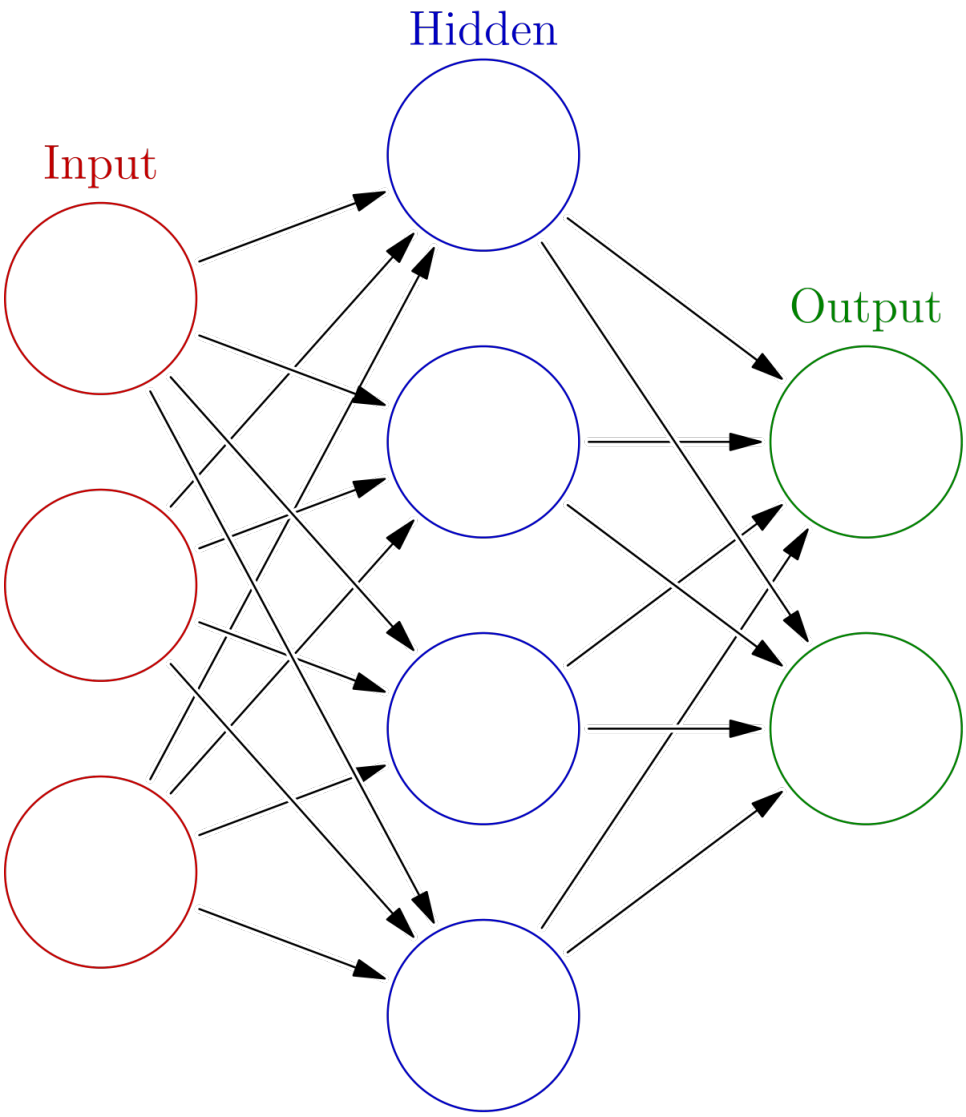

Figure 2.11. Neural network - source: Wikipedia

The most common types of CNN layers are:

- Convolutional layers
- Pooling layers
- Non-linear layers
- Fully connected layers

2.9 Convolutional layers

The convolution operation extracts different characteristics from the input. The first convolution layer extracts low-level features such as borders, lines and angles. The upper layer layers extract superior features.
Figure 3.4 illustrates the 3D convolution process used in convolutional neural networks. The input with size 32 x 32 x 3 (width, height, and RGB color) of the image got a smaller size receptive field, let's suppose 5 x 5 x 3, that has the task of exploring the complete depth of the input volume.
The receptive field is the extent of the connectivity along the depth axis that is always equal to the depth of the input volume. It is critical to underline this asymmetry in how the spatial dimensions are managed (width and height) and the RGB dimension: The connections with the input are local in width and height, but always full along the full depth of the input.
When the process begin, each filter is convoluted through the width and height of the input shape and it is computed the dot product between the filter entries and the input at any position. During this phase the filter is moved over the width and height of the volume of the input and the result will be a 2-dimensional activation map that returns the computations of that filter at every position.
The network will return filters that activate in presence of some type of visual feature, like a particular characteristic of an edge or a specific combination of color on a single or more layers, or possibly entire repetitive patterns of objects and colors on top layers of the network.

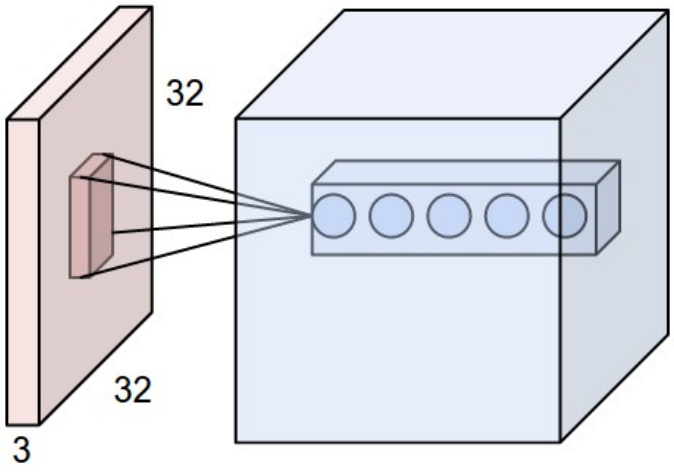

Figure 2.12. Convolutional layer - credits: Karpathy, A. (2016)

In addition to the connectivity between neurons that regulate the connections with the input, it is also important to mention how neurons regulate the volume of the output. The output volume is controlled by three hyperparameters:

1. **Depth**: indicates the number of filters you want to use, each of which is learning to look for different characteristics from the input. For example, if the first convolutional layer receives as input the image not yet analyzed, then depending on the parameter, a different number of neurons are activated along the depth dimension at particular oriented edges, or color spots.

2. **Stride**: Indicates the size, in pixels, of the displacement that the receptive field must do. For example, when the stride is equal to one, the filters move one pixel at a time, when it is two, the filters move two pixels at a time.

3. **Zero-padding**: Allow to control the output volume by padding it with zero around the boarder

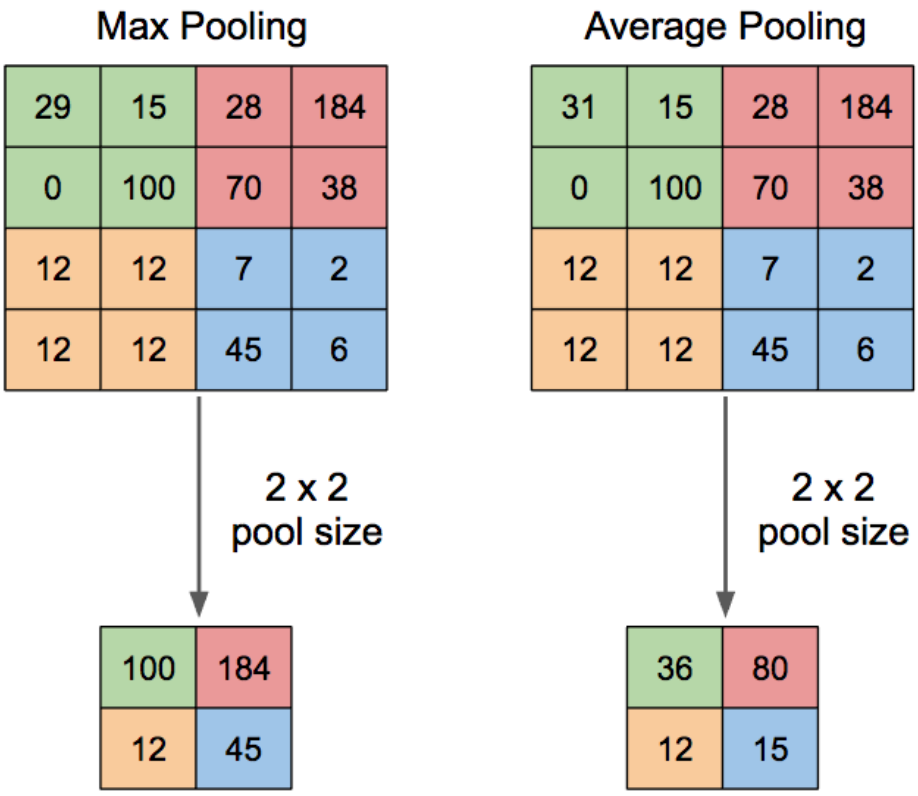

Figure 2.13. Pooling layer - credits: Deshpande, M. (2017)

2.10 Pooling layers

The pooling layer reduces the resolution of features. Commonly it is placed in-between successive convolutional layers. Pooling layer makes robust features against noise and distortion. There are two ways to perform pooling: maximum pooling and average pooling. In both cases, the input is divided into two-dimensional non-overlapping spaces. For example, in Figure 3.5, if the input feature is 32 x 32, it is divided into 16 x 16 and 2 x 2 regions. For average pooling, it is calculated the average of the four values in the region, and for maximum pooling, it is selected the maximum value of the four values.

2.11 Non-linear layers (ReLU)

As seen before, ReLU implements the function y = max (x, 0), so the input and output dimensions of this layer are the same. It increases the non-linear properties of the decision-making function and of the global network without affecting the convection-layer receptive fields. Compared to other nonlinear functions used in CNN (for example, hyperbolic tangent, absolute of the hyperbolic tangent and sigmoid), the advantage of a ReLU is that the network trains much faster. The effect of the ReLU is shown in Figure 3.6.

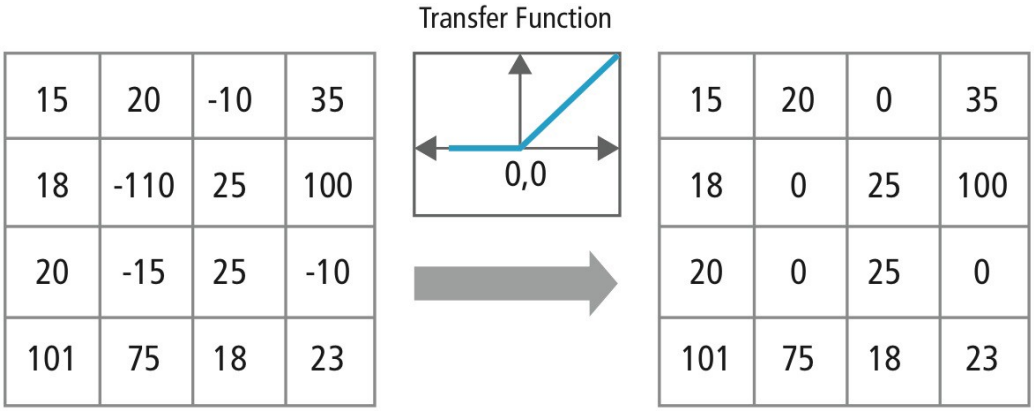

Figure 2.14. ReLU - credits: Harsh, S. (2017)

2.12 Fully connected layers

The fully connected layer (FC) is definitely appropriate to suggest its name: it is completely connected to the output of the previous level. The fully connected layers is useful typically in the later stages of CNN for connectors at the output level and construct the desired number of outputs.

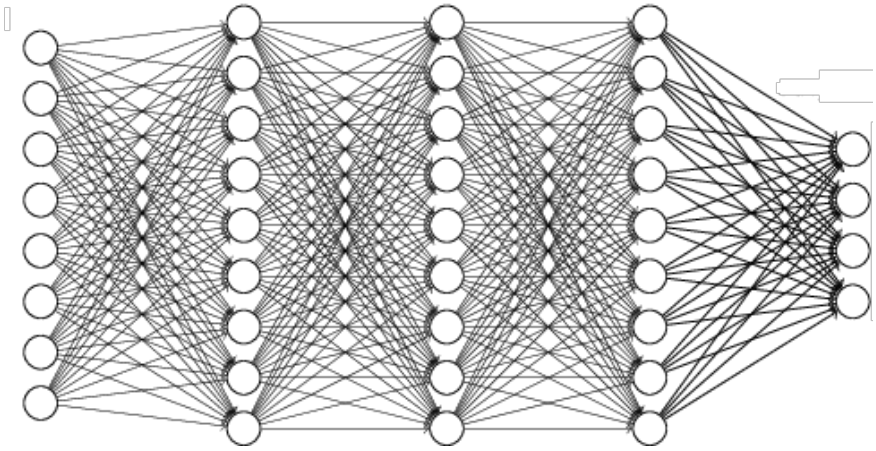

Figure 2.15. Fully connected layers - credits: Port, A. (2016).

Here there is a representation of a simple model of a ConvNet, where are highlighted convolutional layer, pooling layer and ReLU. It is simple to catch the different transformations that are implemented layer after layer.

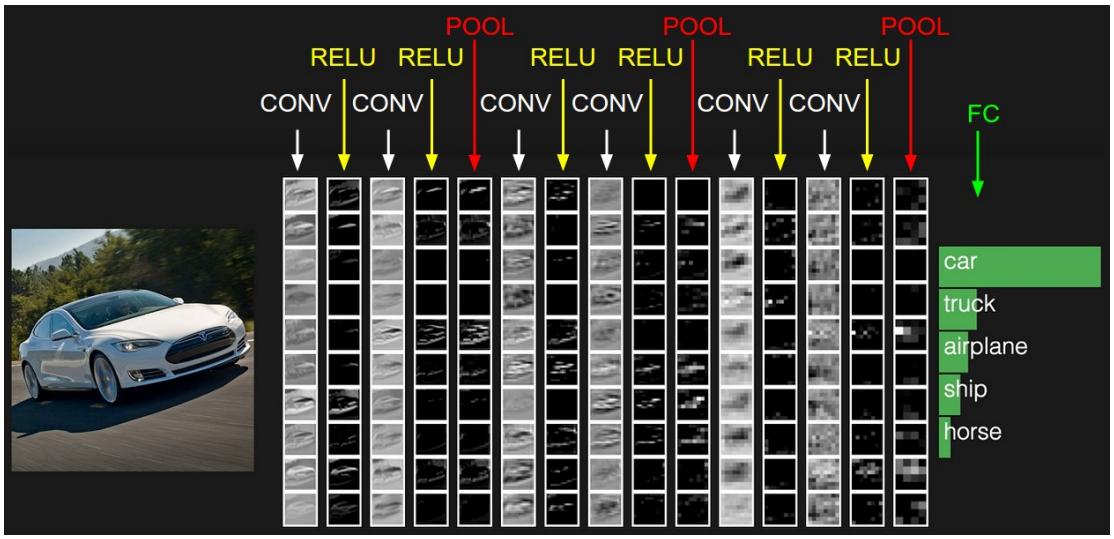

Figure 2.16. ConvNet - credits: Karpathy, A. (2016)

2.13 Net examples

The ImageNet project runs a large-scale annual competition, ImageNet Large ScaleVisual Recognition Challenge (ILSVRC). In this competition software programs compete to correctly classify and detect objects and scenes. The main competing architectures of ILSVRC are shown in Figure 3.7

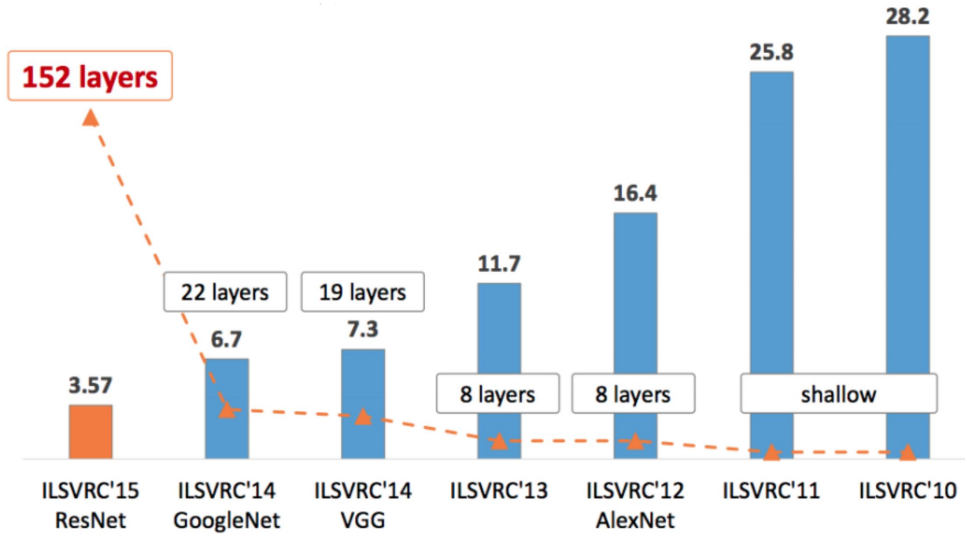

Figure 2.17. ILSVRC - credits: He, K., Zhang, X., Ren, S., & Sun, J. (2016).

LeNet-5, a 7-level pioneering network, developed by LeCun in 1998, which classifies the figures and has been used by several banks to recognize handwritten numbers on checks digitized in 32x32 pixel images. The ability to process high resolution images requires larger and more convolutional layers, so this technique is limited by the availability of computing resources.

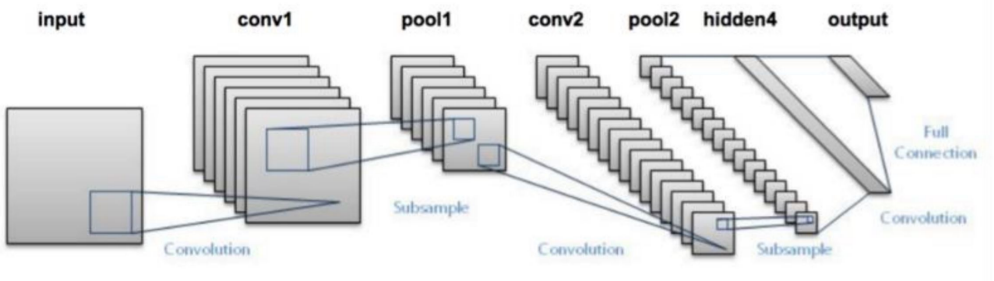

Figure 2.18. LeNet-5 - credits: Yann LeCun et. al. 1998

AlexNet was designed in 2012 by the SuperVision group, composed by Alex Krizhevsky, Geoffrey Hinton and Ilya Sutskever. AlexNet significantly surpassed all previous competitors and won the challenge by reducing the top-5 error to 15.3%. The top-5 error is the rate at which, given an image, the model does not issue the correct label with its 5 main predictions. In second place there was a top-5 error rate, which was not a change in CNN of around 26.2%.
The network had an architecture very similar to the LeNet of Yann LeCun, but it was deeper, with more filters per layer and with superimposed convolutional layers. AlexNet was trained simultaneously on two Nvidia GeForce GTX 580 GPUs, which is why the network is split into two pipelines.

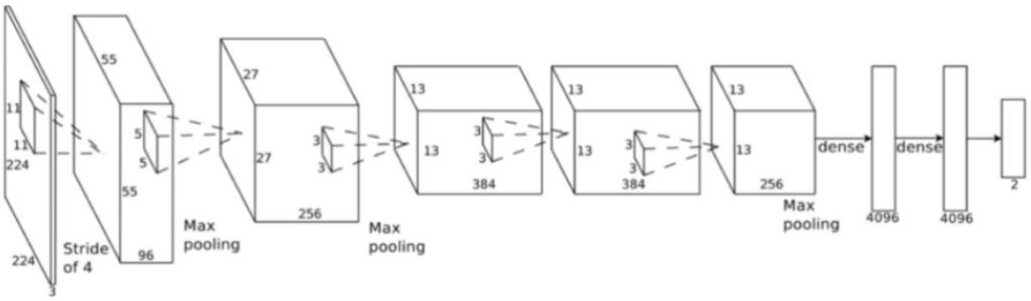

Figure 2.19. AlexNet - credits: Krizhevsky et al., 2012

The winner of the ILSVRC 2014 competition was **GoogleNet (Inception)** by Google achieving a top-5 error of 6.67%. This rate was very close to the performance of a human that challenge organizers were now forced to evaluate. Apparently, the goal was actually rather difficult to achieve and required human training to beat GoogLeNets accuracy. After a few days of training, expert Andrej Karpathy managed to achieve a 5% top-5 error rate.
The network used a CNN inspired by LeNet but implemented a new element that was dubbed an initial module. This module is based on numerous very small convolutions in order to drastically reduce the number of parameters. Their architecture consisted of a 22-layer deep CNN, but reduced the number of parameters from 60 million (AlexNet) to 4 million.

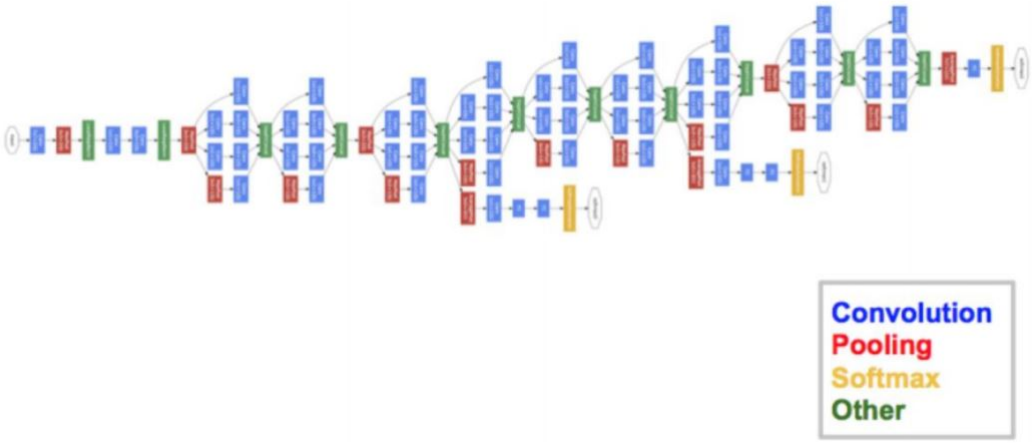

Figure 2.20. GoogleNet - credits: C. Szegedy et al., 2014

The second place in the ILSVRC 2014 competition was **VGGNet**, developed by Simonyan and Zisserman. VGGNet consists of 16 convolutional layers and is very captivating thanks to its very uniform architecture. Performs only 3 × 33 times 33 × 3 convolutions and 2 × 22 times 22 × 2 pooling all the way down. It is currently the preferred choice in the community to extract functionality from images. The configuration of VGGNet is publicly available and has been used in many other applications and challenges as a basic feature extractor. However, VGGNet includes 140 million parameters, which can be difficult to manage.

Figure 2.21. VGGNet - source: http://machinethink.net

Finally, at the ILSVRC 2015, the so-called residual neural network, **ResNet** of Kaiming He introduced the innovative architecture with "skip connections" and presents a heavy batch normalization. These skip-connections are also known as closed units or recurring units with gating and bear a strong resemblance to the recent success factors applied in the RNNs. Thanks to this technique they were able to train a neural network with 152 levels while having a lower complexity than

VGGNet. It achieves an error rate higher than 5 of 3.57% which beats human performance on this dataset.

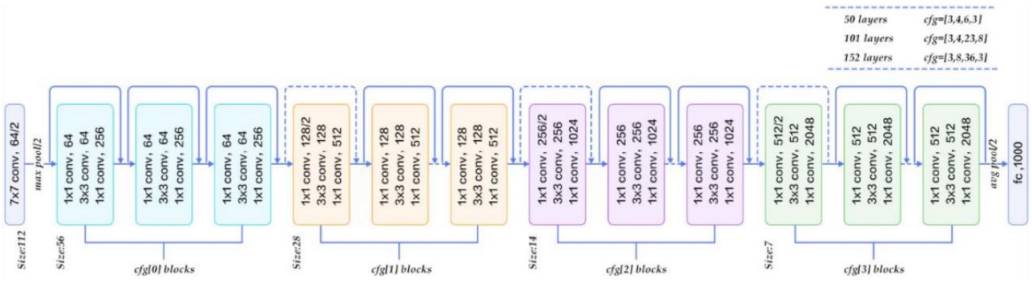

Figure 2.22. ResNet - source: ImageNet Large Scale Visual Recognition Challenge, 2015

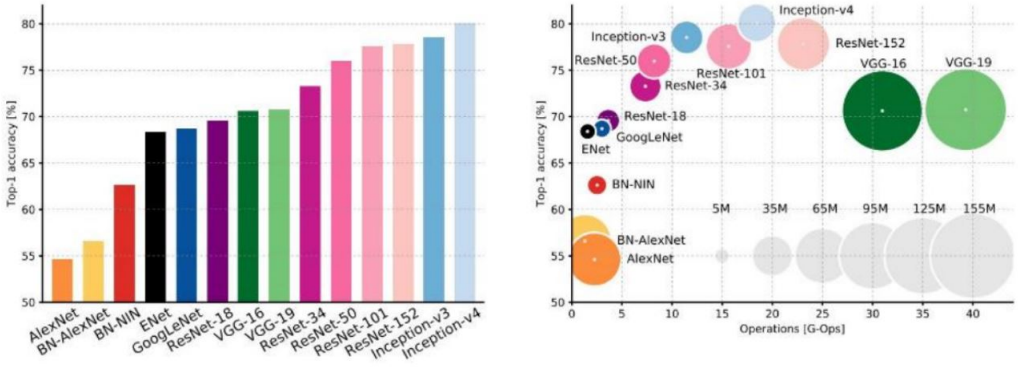

Figure 2.23. Comparison between various models - credits: Karpathy, A. (2016)

2.14 False Positive Reduction

False positive reductions refer to the ability of the network to tolerate the presence of erroneously indexed elements within the model classes.
It has been established that[3], given a dataset relative to a number of patients who are tested positive or negative cancer by CAT, and having to analyze which nodules of cancer patients were actually infected, we would inevitably face the presence of false positives. The various labels are in fact assigned to the nodules based on the patient's situation, so all the nodules of a positive patient are labeled as positive, and vice versa if the patient is negative to the cancer.

[3]Setio, A. A. A., Ciompi, F., Litjens, G., Gerke, P., Jacobs, C., Van Riel, S. J., ... & van Ginneken, B. (2016).

When this operation is performed, false positives are introduced, because the starting group were the CATs, transformed by segmentation into the various nodules. Consequently, when the model is trained, positive labels are attributed to all the individual nodules of a positive patient.
Experimentally, it has been proven that the neural networks that are resistant to the introduction of false positives by a certain percentage. If the false positives are not too high in fact, the final accuracy of the model does not undergo any particular changes thanks to the discriminating capacity of the network.

Chapter 3

Deep Learning

3.1 Brief introduction to Deep Learning

While machine learning algorithms have been around for a long time, the possibility of automatically applying complex mathematical calculations to large-scale data has recently been developed.
This is because the increased power of today's computers, in terms of speed and memory, has helped machine learning techniques to evolve to learn from a large training dataset. For example, with more computing power and a fairly large memory, it is possible to create neural networks of many layers, which are called deep neural networks, or deep neural networks.
We can attribute the birth of Deep Learning to different historical moments that led first to the discovery of deep neural networks and then to the resolution of problems thanks to these models.
Regarding the trend of research conducted on Google about the words "Machine Learning" and "Deep Learning", the graph of Figure 2.1 shows a growing trend since October 2015, the month in which the program developed by DeepMind called Google AlphaGo won the strategic game Go against the european champion Mr Fan Hui.
Go is strategy board game for two players. The aim of Go is to conquer more territory than the opponent. Even if its rules are not so difficult, Go is really tricky. Some of its difficulties consist in the greater size of the game board compared to similar games (ie chess), the different strategies that can be adopted and the all possible moves to consider at each move. The number of legal board positions in Go has been estimated to be 2 x 10^{170}.

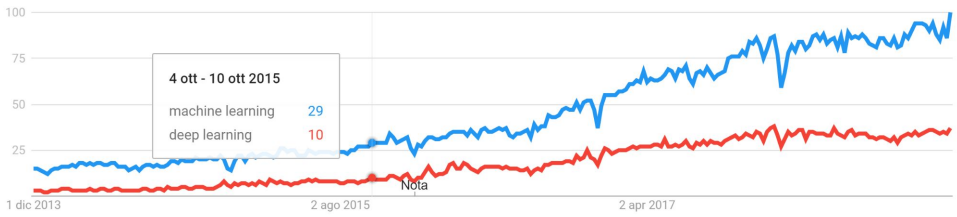

Figure 3.1. World trend over 5 year of 'machine learning' and 'deep learning' - credits: Google

Since that time, Deep Learning has passed under the eyes of the media. This has consolidated its popularity and as can be seen from the chart, the research has begun to have an ever-increasing trend. The second peak in the chart can be attributed to DeepMind and in particular to the AlphaGo 4-1 victory in Seoul, South Korea, in March 2016 which has had a television audience of over 200 million people worldwide.
Even in Italy these news have not gone unnoticed. In fact, in Figure 2.2, we can see the same trends that took place worldwide.

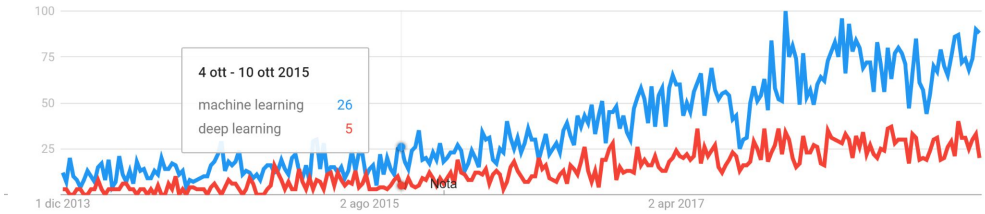

Figure 3.2. Italian trend over 5 year of 'machine learning' and 'deep learning' - credits: Google

The Google DeepMind system combined the "Monte-Carlo Tree Search" algorithm with Deep Neural Networks that have been trained in a supervised manner. In particular, the term "Deep Learning" was used to describe the way in which DeepMind won. Figure 2.3 helps to better understand the differences between Deep Learning, Machine Learning and Artificial Intelligence

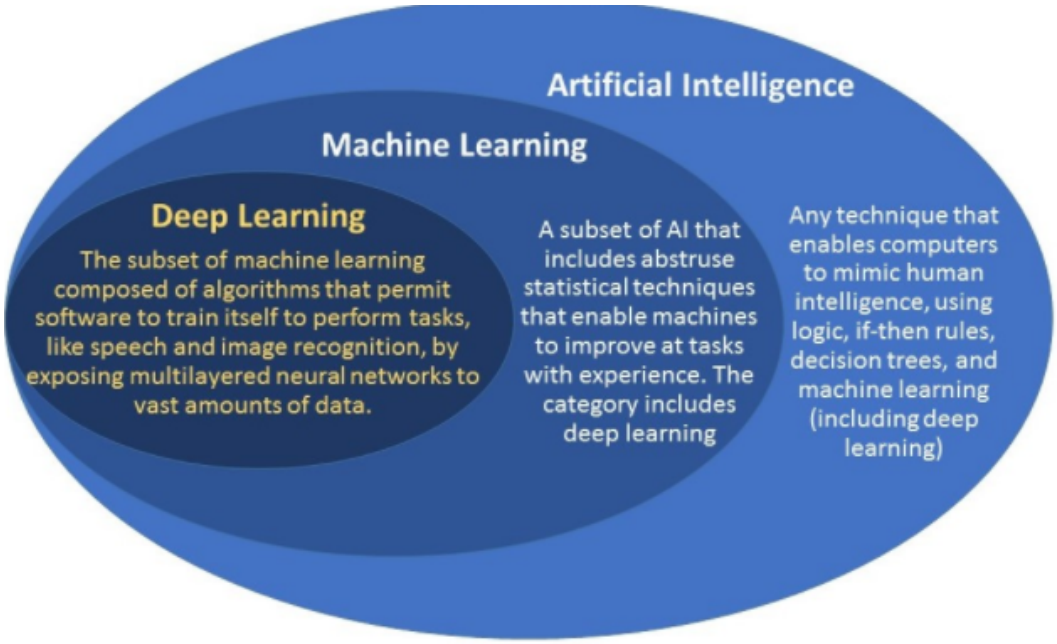

Figure 3.3. Comparison between Deep Learning, Machine Learning and Artificial intelligence - credits: deeplearningitalia.com

The set-theoretical vision of scientific areas helps us to understand the subsets that have been created in the world of Artificial Intelligence. Artificial Intelligence represents the techniques that allow the computer to copy human intelligence, using logic and different models, including Machine Learning and Deep Learning.
Machine Learning is a subset of Artificial Intelligence, including techniques that allow the machine to learn from a dataset. Deep Learning is the extension of a particular technique of Machine Learning (Neural Networks), which allows the machine to learn independently. This technique uses Deep Networks (Deep Learning) and a large dataset (Big Data) to solve problems such as recognition of images and language processing.
Deep Learning is an area of Machine Learning, which uses the Neural Networks as a basis but with the addition of several layers to improve the learning of the model.
The Neural Networks are bio inspired by the functioning of the visual cerebral cortex. Each neuron can connect to any other neuron within a certain physical distance. Artificial Neural Networks have discrete levels of connections and directions of data propagation.
Until a few years ago the Neural Networks, as a Machine Learning area, had been completely ignored by the research community. The problem was the intensive computational calculation that also required for the simplest models. Until, the

research group led by Geoffrey Hinton at the University of Toronto has finally parallelized the algorithms for the Neural Networks. The final turning point was Andrew Ng who added layers and neurons to the Net to train the system on huge amounts of data. In the case of Andrew Ng, they were images taken from 10 million videos on YouTube. Andrew Ng puts the term "Deep" to describe the huge number of levels that these Neural Networks have.
Thus, the first requirement of a Deep model is to have a large dataset available for training. This is because there are a large number of parameters that need to be understood. The reasons why Deep Learning has become so popular and used are: Big Data and Graphic Processing Units (GPUs). For example, a Deep Learning algorithm could be instructed to "learn" to recognize a dog. To distinguish the details that differentiate a dog from a wolf, we need a huge dataset on which to train the model, and consequently power computing (GPU).
The ability to correctly learn the models depends on the dataset, because to avoid unexpected results it is necessary that, both in the dataset and on the labels we assign, there are no errors, so that the model does not learn the wrong rules that it will put into practice.
There are also several examples of the use of Deep Learning in business such as: fraud detection, graphic recognition, image search, speech recognition. To date, the recognition of images trained through Deep Learning, in some scenarios, is better than human beings. Machine Learning and Deep Learning are two sides of the same coin, composed of the same common nucleus of AI. According to some researchers once you know well the Deep Learning you can solve many problems in areas such as health, business, marketing, etc.

Andrew Yan-Tak Ng, former Baidu Data Scientist, who led Baidu's Artificial Intelligence team in May 2016, wrote in a Twitter post:

"AI is the new electricity! Electricity transformed countless industries; AI will now do the same."[1]

[1] https://twitter.com/AndrewYNg/status/735874952008589312

3.2 Raising of Deep Learning

In recent years, deep neural networks have achieved considerable progress in the search for many fields such as Artificial Intelligence and Machine Learning. Deep Learning has been successfully applied to speech recognition[2] and many other areas of Machine Learning[3].
So far, in all experiments, the resulting performance was far better than other available machine learning techniques. One of the properties that give Deep Learning a special place is the ability to extract meaningful concepts (with a human perspective) from data, combining features based on data structure. The concepts extracted sometimes have a clear interpretation and it seems that the machines have actually "learned something". The reasons for this success derive from the three advantages of the Deep Learning approach:

1. **Simplicity**: Instead of problem-specific tweaks and measure-to-measure detectors, deep networks offer basic architectural blocks, network layers, that are repeated multiple times to generate large networks.

2. **Scalability**: Deep learning models are easily scalable in huge datasets. Other competing methods, for example, kernel machines, encounter serious computational problems if the datasets are huge.

3. **Domain Transfer**: A model learned about a task is applicable to other related activities and the features learned are general enough to be used on a range of activities that may have limited data available.

Because of the tremendous success in learning as a result of the development of deep neural networks, Deep Learning techniques are currently at the forefront for the detection, segmentation, classification and recognition (ie identification and verification) of objects in images.
It is also important to underline that such a significant improvement in the last few years is due above all to a directly proportional improvement in the technology that allowed the development of techniques that required increasingly advanced systems to manage and analyze data.
The improvements that have been made, although they all belong to the technological

[2] Dahl et al., 2012
[3] Li Deng et al., 2013

sphere, have been of various kinds.

One of the most important improvements has certainly been the advent of powerful GPUs making it possible to train very large neural networks with over 60 million parameters. A Graphical Processor Unit (GPU) is a processor that renders images, animations and videos to the computer screen. GPUs are usually found on computer video cards and perform parallel operations. Basically, GPUs are SIMD (Single Instructions Multiple Data) architectures that fall into the array processor category. Thanks to this parallelism, GPUs have led to the emergence of GPU-accelerated computing, which is the use of GPUs together with a CPU to accelerate scientific, analytical, engineering, consumer and business applications.

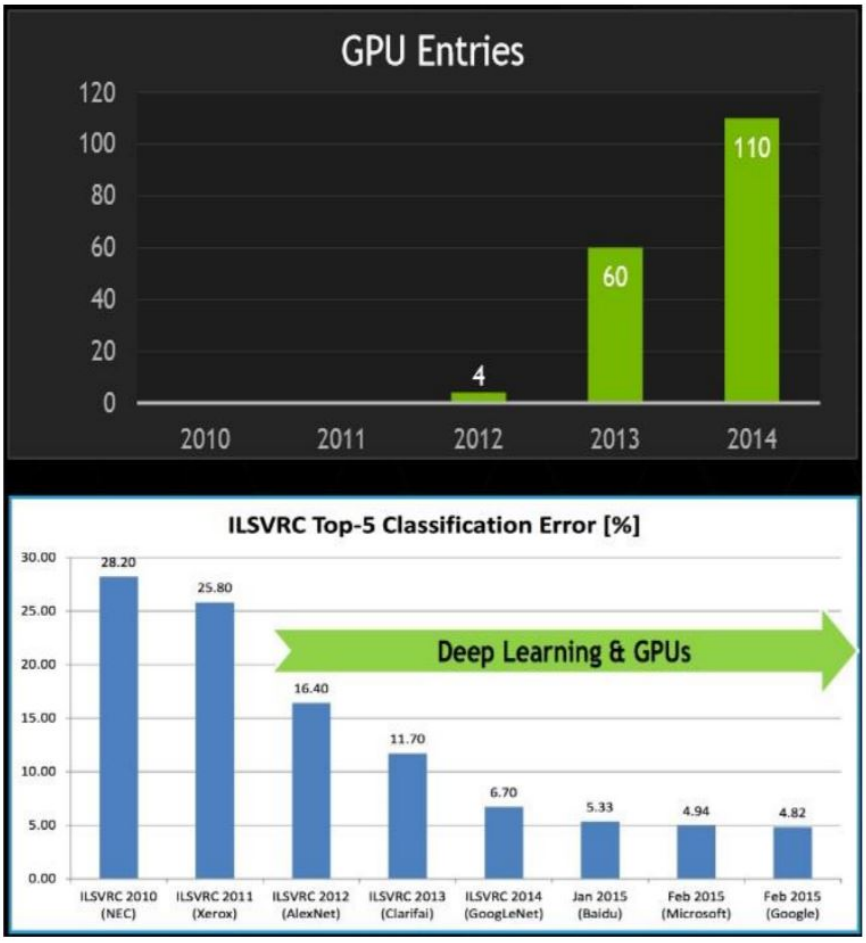

Figure 3.4. GPU improvements - credits: NVIDIA

An easy way to understand the difference between a CPU and a GPU is to compare the way they process tasks. A CPU is made up of a few cores optimized for serial processing, while a GPU has a massively parallel architecture consisting of thousands of smaller and more efficient cores designed to handle multiple tasks simultaneously. GPU accelerated computing delivers unprecedented application performance by downloading intensive processing portions from the application to the GPU, while the rest of the code still runs on the CPU. From the user's point of view, applications are executed much faster.
A pioneer in 2007 was NVIDIA, now GPU accelerators power energy-efficient data centers in government labs, universities, businesses and small and medium-sized businesses around the world. GPUs are accelerating applications on platforms ranging from cars, mobile phones and tablets. For these reasons, accelerated processing by GPU has become the reference platform for a computationally expensive methodology such as Deep Learning. Figure 2.4 shows a graph that certifies the impressive improvement of the performance achieved by deep neural networks adopted during the "Image Recognition Challenge" promoted by Imagenet.
Imagenet competition consists in training the machine vision algorithm of artificial vision through a dataset of millions of images divided into 1000 categories. As can be seen, after the use of GPUs from 2012 (from 4 to 110), the winning teams witnessed a drastic reduction in the error rates of the neural network adopted. Figure 2.5 shows the acceleration of the GPU based on the different data dimensions compared to the CPU times, these data refer to the deep neural network used in the 2012 Imagenet competition.

Batch Size	Training Time CPU	Training Time GPU	GPU Speed Up
64 images	64 s	7.5 s	8.5X
128 images	124 s	14.5 s	8.5X
256 images	257 s	28.5 s	9.0X

Figure 3.5. GPU acceleration - credits: NVIDIA

The reasons why GPU acceleration calculation is so well suited to deep neural networks is that:

1. Neural network structures are inherently parallel and based on basic matrix algebra operations. GPUs can calculate the massive operations required with greater accuracy, faster results, smaller footprint, lower power and lower costs.

2. Thanks to the increasing number of challenges, there is now a bigger availability of "Big Data", that is to say a new one generation of broader training and testing sets. For example, datasets with millions of tagged and unlabeled images. Until recently, the image datasets were relatively small, in the tens of thousands of images, but in recent years a new generation of datasets with millions of images has arrived. Some of popular datasets are shown below:

    - **Norb**: contains images of 50 toys belonging to 5 generic categories: four-legged animals, human figures, airplanes, trucks and cars.
    - **Caltech-101**: images of objects belonging to 101 categories. Approximately 40-80 images by category. Most categories have about 50 images (300x200).
    - **Caltech-256**: the same data set as before but with 256 categories. Here is a total of 30607 images.
    - **CIFAR-10/100**: CIFAR-10 and CIFAR-100 are labeled subsets of the dataset of 80 million lowercase images that vary from animal species to means of transport. The CIFAR-10 dataset consists of 60,000 32x32 color images in 10 classes, with 6000 images per class. There are 50,000 training images and 10,000 test images. The CIFAR-100 dataset is just like the CIFAR-10, except that it has 100 classes containing 600 images each. There are 500 training images and 100 test images per class. The 100 classes of the CIFAR 100 are grouped into 20 superclasses.
    - **MNIST**: the MNIST database of handwritten digits has a training set of 60,000 examples and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The figures were normalized and centered in size in an image of a fixed size.
    - **LabelMe**: consists of hundreds of thousands of images in lightning segments. The goal of LabelMe is to provide an online annotation tool to create image databases for artificial vision research.

- **ImageNet**: ImageNet is a data set organized according to the hierarchy of WordNet. This figure consists of over 15 million high-resolution images labeled in over 22,000 categories.

3. Better model regularization techniques such as "Dropout" [4] or "Data Augmentation" [5] have been discovered that limit the effect of *overfitting*.
   Overfitting is a condition that usually occurs during model training and is a major problem of modern neural network training. Typically, an artificial neural network uses "hidden" levels for learning feature detectors that help predict the correct output. If this relationship is complicated and the network has enough hidden units to model it accurately, there will be many different weight settings that can shape the training set almost perfectly, especially if there is only a limited amount of labeled training data.
   Each of these weight vectors will make different predictions about untested test data and almost all of them will worsen both on test data and on training data because the feature detectors have been optimized to work well together on the training data, but not on the data test, and this is the situation where overfitting occurs.
   Using the dropout technique half of the feature detectors on each training case can be randomly omitted, and overfitting can be greatly reduced. Basically the output of each hidden neuron is set to 0 with probability 0.5. In this way, the dropped out neurons do not contribute to the passage forward and to the learning process. In this way, this technique reduces complex adaptations of the neurons, since a neuron can not count on the presence of particular other neurons. It is therefore, forced to learn more robust features that are useful in conjunction with many different random subsets of the other neurons. An example of how dropout works is shown in Figure 2.6
   Another way to reduce over-processing on data is to artificially increase the number of items of the dataset using label retention transformations called data increases. For example, in the case of the training set of images there are two distinct forms of data increment: the first consists in generating horizontal translations and reflections, while the second consists in altering the intensity of the RGB channels in the training images.

[4] Hinton et al, 2012.
[5] Krizhevsky et al., 2012.

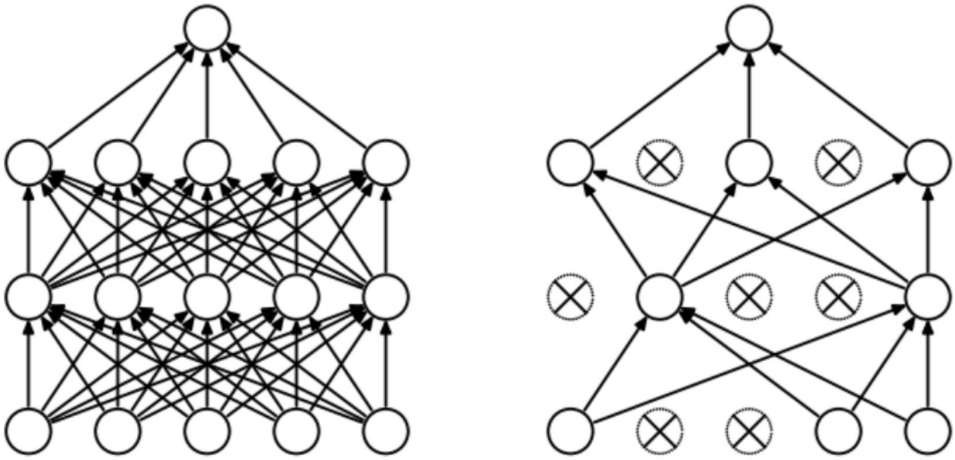

Figure 3.6. Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting", JMLR 2014

4. Another strong advantage of Deep Learning is the recent development of new visualization techniques that provide insights into the function of internal layers in the model[6].

   Basically, these visualization methods are able to reveal input stimuli that excite the individual characteristics of the maps at any level in the model. They allow to observe the evolution of the characteristics during training phase, to diagnose potential problems with the current model and to find optimal architectures.

   Furthermore, visualization techniques make a deep neural network not more a black box as was the case with Artificial Neural Networks, but potentially give a tool to study the behavior of the various learning models even more thoroughly.

   In the case of images, for example, it is easy to recognize the activations of neurons at different levels of abstractions because they focus on different characteristics (color, shape, width, length) of the same image when it is introduced into the model during the training phase.

   Even in the largest companies in the IIT world there is a big debate around the developments of Deep Learning: Google, Facebook, Microsoft, etc. Last year Google bought "Deep Mind", a start-up specialized in Deep Learning for 400 millions.

[6] Zeiler, M. D., & Fergus, R., 2014.

Deep Mind has developed a system that combines methods of Deep Learning and reinforcement learning [7]. In this way, the machine seems able to learn how to play simple computer games in the same way as humans: using the computerized equivalent of coordination " hand-to-eye. " Basically the system has access only to visual information on the game screen and to scores. Based on these two inputs the system learns to understand which moves are good and which ones are bad depending on the situation on the screen.
Note that a human player uses exactly the same information to evaluate his performance and adapt his game strategy. The more than positive result reported shows that the system has been able to master several games and play some of them better than a human player. According to some experts, this result can be seen as a step towards truly intelligent machines.

3.3 Tools and libraries

There was great interest from academics (for example, the University of California Berkeley, New York University, the University of Toronto, the University of Montreal) and industrial groups (for example Google, Facebook, Microsoft) to develop Deep Learning frameworks and libraries. This is mainly due to their popularity in many application domains in recent years.
The key motivation for developing these libraries is to provide an efficient and friendly development environment for researchers to design and implement deep neural networks.
Some of the widely used Deep Learning frameworks are: Caffe, TensorFlow, MatConvNet, Torch7, Theano, Keras, Lasange, Marvin, Chainer, DeepLearning4J. Many of these libraries are well supported, with dozens of active contributors and a multitude of users. Because of the powerful CUDA backends, several of these frameworks are very fast in training deep networks with billions of parameters. The most often used frameworks are briefly described:

- TensorFlow: it was originally developed by the Google Brain team. TensorFlow is written with a Python API on a C / C ++ engine for numerical calculation using data flow graphs. You can find multiple APIs (application programming interface). Full programming control is provided with the lowest level APIs,

[7] Mnih, V., et. Al., 2013

called TensorFlow Core suitable for Deep Learning researchers and others who need a precise level of control over their models.
The higher level APIs are easier to learn and use, compared to TensorFlow Core. TensorFlow offers automatic differentiation features, which simplify the process of defining new operations in the network. Use data flow charts to perform numeric calculations. Graphical nodes represent mathematical operations and borders represent tensors. TensorFlow supports multiple backends, CPUs or GPUs on desktop, server or mobile platforms. It has well-connected links to Python and C ++. TensorFlow also has tools to support reinforcement learning.

- Theano: it is a Python library and a compiler that efficiently defines, optimizes, and evaluates mathematical expressions involving multidimensional arrays. Theano was mainly developed by an automatic learning group at the University of Montreal. Combines the algebra system with an optimizing compiler useful for tasks in which complex mathematical expressions are evaluated repeatedly and evaluation speed is critical.
  Theano provides several implementations for the convolution operation, such as an implementation based on FFT [8] and an implementation based on the open source code of the image classification network [9]. Theano has implementations for most of the avant-garde networks, both in the form of a higher level picture, such as Blocksa and Keras, or pure Theano. However, Theano is somewhat low-level and large models have long build times. However, Theano will no longer be developed by implementing new features after 2018.

- Keras: it is a opensource high-level neural network API, written in Python and able to run on TensorFlow and Theano. Therefore, Keras benefits from the advantages of both and provides a set of abstractions of a higher and more intuitive level, which simplifies the configuration of neural networks, independently of the back-end scientific calculation library. The main motivation behind Keras is to allow rapid experimentation with deep neural networks and to move from idea to results as quickly as possible.
  The library consists of numerous implementations of blocks and neural network tools to simplify work with image and text data. Keras offers two types of deep neural networks including sequences based networks, where inputs flow linearly

[8] Mathieu et al., 2014.
[9] Krizhevsky et al.2012.

across the network, and graph-based networks, where inputs can skip certain levels. Therefore, the implementation of more complex network architectures such as GoogLeNet and SqueezeNet is easy. However, Keras does not provide the most advanced pre-trained models.

Chapter 4

Segmentation and Automatic Extraction of Statistics from Satellite Maps

4.1 Segmentation

The term segmentation of the image indicates the division of an image into the different parts or regions in which it is composed. The ultimate goal of this process is that the different sections of the image are accurate so as to highlight significant areas, which can range from simple objects, people or animals, to the more complex areas of a satellite image such as buildings, seas, roads and crops. Areas can also be delimited by geometric shapes delineated by sets of pixels arranged in a polygon or as a circle. It is possible to talk about segmentation even if the entire image is not completely covered by the areas involved, subdividing them into areas of interest in the foreground and background areas to be ignored.
There are two main objectives of segmentation:

- The first objective is to break down the initial image into smaller regions for in-depth analysis. In cases involving not particularly complex images, it is possible to check the analysis directly to ensure that the segmentation along its process efficiently extracts only the regions that need further analysis. The segmentation process is almost always reliable, some of the possible difficulties could arise when the color gradations between one region of interest and the other are particularly similar. In different cases, concerning the extraction of areas and polygons from a satellite image, more problems may arise due to the

complexity of knowing perfectly the color scales in order not to confuse crops and vegetation according to the different shades of green color. With proper knowledge of domains for segmentation, however, most errors can be avoided.

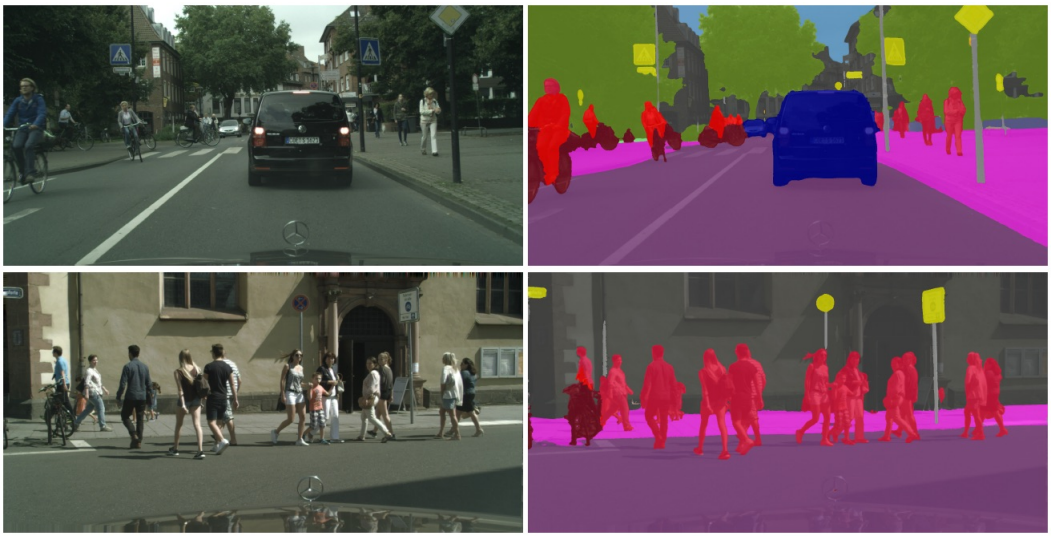

Figure 4.1. Segmentation examples - credits: Kundu, A., Vineet, V., & Koltun, V. (2016).

- The second objective of segmentation based on the change of image representation. The organization of pixels within the image must be organized in units that have a higher level than others, and consequently that they are either more efficient or more significant for subsequent major studies.

4.2 U-Net

U-Net is a special convolutional neural network developed for the segmentation of biomedical images at the Computer Science Department of the University of Friborg in Germany[1] and owes its name to the particular "U" shape. The core of the network is based on a fully convolutional network[2], with topology and architecture modified correctly and updated to work with fewer images during training phase, managing to obtain much more detailed segmentations.

The principal idea of the U-Net is to integrate a base contracting network by successive layers, where pooling layers are replaced by upsampling layers, in order to

[1] Ronneberger, O., Fischer, P., & Brox, T. (2015, October).
[2] Long, J., Shelhamer, E., & Darrell, T. (2015).

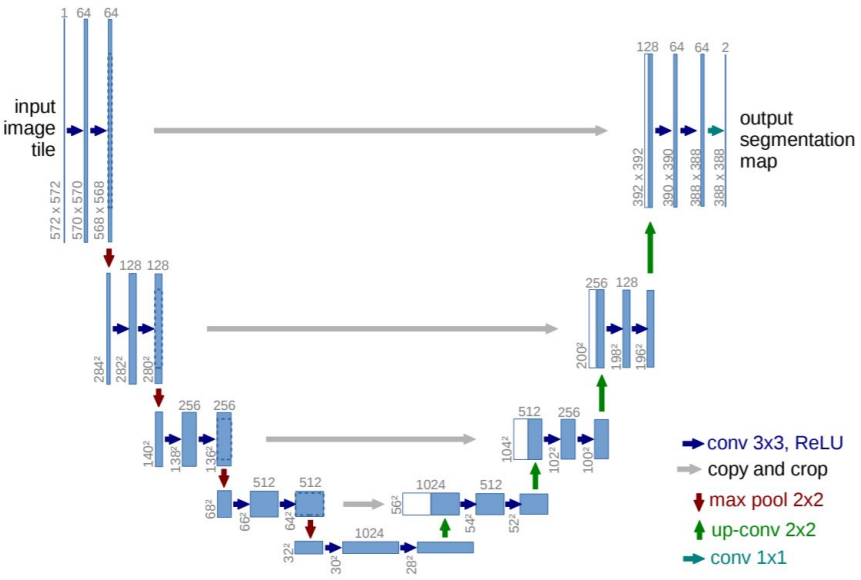

Figure 4.2. U-Net architecture - credits: Ronneberger, O., Fischer, P., & Brox, T. (2015, October).

have an higher resolution in the output. Furthermore, to be able to localize, there is a combination between high resolution features from the contracting path and the output of the previous layer, as a result, a convolution layer has the necessary information to provide a more accurate output.

In the architecture of the model there is also an important innovation that regards the part of upsampling, where the network is able to spread useful information at higher resolution levels through a large number of specific channels. For this reason, the entire network architecture is symmetrical, bringing out the particular "U" shape, s shown in Figure 4.1

Inside the U-Net there are no fully connected layers, since using only the pixels contained in the segmentation map, it already has the whole context available from the input image, in this way it is possible to be able to segment images continuously, even if they are of different sizes.

In the case in which it is necessary to be able to predict a portion of the pixels contained in the edge of the supplied image, the missing part is provided directly by the image itself via mirroring. The mirroring technique becomes even more useful if you need to work with a high resolution image, in order not to overload the GPU during model training.

4.3 Data Augmentation

In the case of models for which limited training samples are available, data augmentation is an essential technique for improving the generalization of models to make them adequately invariant and robust.
When we deal with high-resolution images that represented significantly smaller objects in scale, the data augmentation has an even more crucial role, allowing the image to be submitted to transformations that are useful to allow a more accurate model training.
Another strong point of data augmentation is the ability to avoid overfitting when a model is trained with few images. However, there are cases in which the images are highly correlated even after the data augmentation process.
The features of the data augmentation are:

- Size reorganization
- Sample standardization
- Random rotation
- Shift
- Zoom
- Horizontal and vertical vibration
- ZCA whitening

Chapter 5

Results

5.1 False Positive Reduction Experiment

Before the real work at the center of the project regarding satellite images, it was tried to replicate the condition that occurred in the experiment regarding false positives. Recalling from Ch. 2.14, false positive reduction is the ability of the network to tolerate the presence of erroneously indexed elements within the model classes.

In order to do that, the CIFAR-10 dataset was used, appropriately reduced so as to be able to use only two out of ten present classes, specifically, the first two classes were chosen, representing airplanes and automobiles respectively, with 5000 images for each class.

Once the dataset has been reduced, to be able to recreate conditions similar to those desired, it was decided to "dirty" one class, keeping the other unchanged. The procedure used to dirty one class, consisted in eliminating a pre-established number of images chosen in random positions from the airplanes class, replacing them with the same number of elements always chosen randomly by the automobiles class, and setting a fixed seed for reproducibility.

At the end of the procedure, in fact, there will be two identical classes with 5000 each, with images representing only automobiles, since the images representing airplanes have been gradually eliminated. The number of items chosen to be substituted for each iteration was 500, corresponding to 10% of the entire class, so that you can closely observe the changes in the network at every single step.

The model used for this experiment consisted of two convolutional layer sets, followed by a set of fully connected layers, ending with a softmax classifier.

To be able to visualize the activations of the neurons of the model in order to study the behavior of the neural network in different instants, it was used t-SNE.[1] t-SNE is a technique which allows dimensionality reduction by giving each datapoint a location in a two or three-dimensional map, and is particularly well suited for the visualization of high-dimensional datasets.
At each iteration the value of the best accuracy was saved and its trend is represented in Figure 5.1

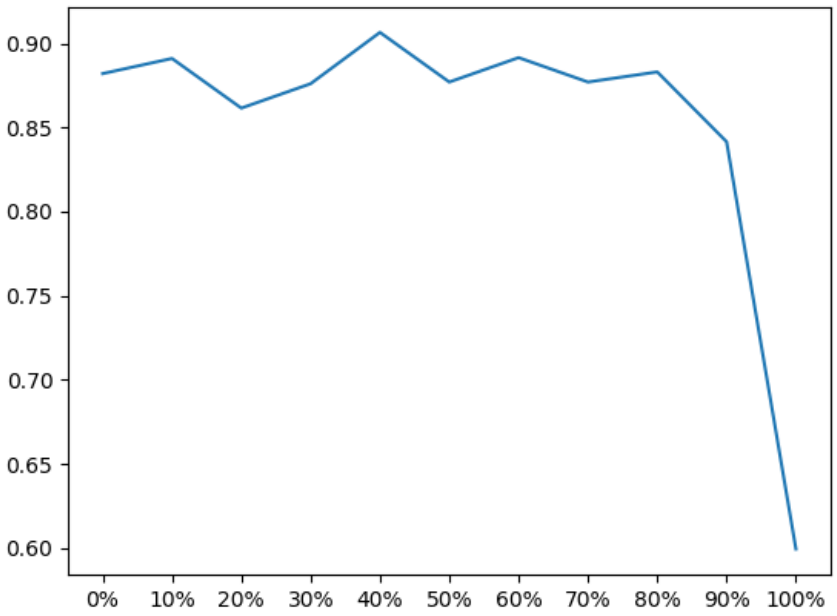

Figure 5.1. Accuracy trend

As is easy to see, the accuracy of the model maintains a very high value until false positives reach 80% of the airplanes class. The model then undergoes a slight drop when a further 10% of images are eliminated and then become completely random once the two classes have actually the same elements.
The discriminative capacity of the model is actually very powerful, because until the two classes are completely identical it manages to maintain a high level of accuracy. The neurons of the model can in fact somehow discern the specific characteristics of all the elements of a class, highlighting even the smallest but significant difference between the two classes.

[1] Maaten, L. V. D., & Hinton, G. (2008).

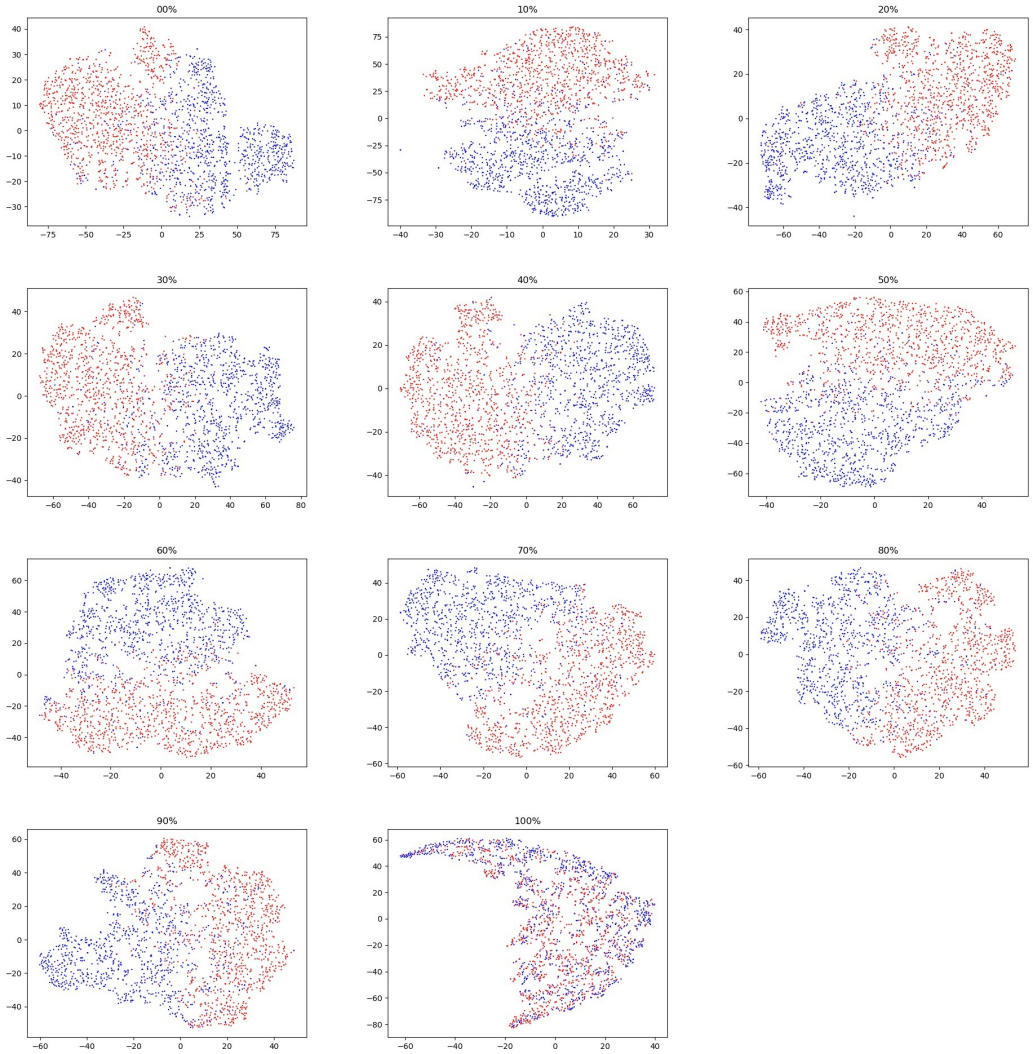

Figure 5.2. Internal representation of neurons

Figure 5.2 shows the internal representation of neurons according to the re-trained neural network each time a percentage of images was eliminated.
Within the neural network the two different colors symbolize the elements of the two different classes. The shape of the net seems to change with each training, while maintaining a clear distinction between the elements of the two classes except for a few cases. This observation is completely overturned when the classes are the same and the network loses all discriminatory capacity, resulting in a big mixture of neurons where it is impossible to divide the two sets.

5.2 Segmentation software

The software related to the segmentation and subsequent classification of satellite images was created and written entirely in Python.
The functions it performs are as follows:

- **Segmentation**: for each image are extracted the various polygons that compose it so as to build the final dataset for training and testing.
- **Train**: the neural network is trained for the recognition of the ten different classes following a U-Net model.
- **Test**: the accuracy of the network in recognizing objects is verified by testing the power of the model with unindexed images.
- **Classification**: an image provided as input is processed, and after having divided it into polygons, the land cover is calculated with the relevant statistics.

5.2.1 Segmentation

In order to work on the software that will have to be able to classify the satellite images it was necessary to construct the starting dataset accordingly.
The images correctly indexed with the respective polygons that allowed the segmentation were 25, and they were all aggregated in a single larger image, of size 5x5. The pixel dimensions of the starting images were variable, with an average of 840x850 and multispectral bands with 8 channels.
Once the complete image is obtained, the polygons have been highlighted to allow them to be divided into classes. The division of the polygons into the various classes was stored in a matrix of dimensions equal to that of the larger image, containing 10 channels, each of which contained within it the representation of the pixels relative to the polygons of its category, thus creating the image mask. For example, the first channel contained the pixels that corresponded to the polygons of the buildings.
This procedure allows in fact to be able to subsequently create the datasets that allow the train and the test of the model. Subsequently, to complete the procedure, sub-images of size 160x160 from the large image have been cut out, taking extreme care to also extract the corresponding mask related to the same pixels of the original one.
The final dataset was so complete, with 4594 total images of 160x160 size, divided into 3533 images in the training set and 1061 in the test set.

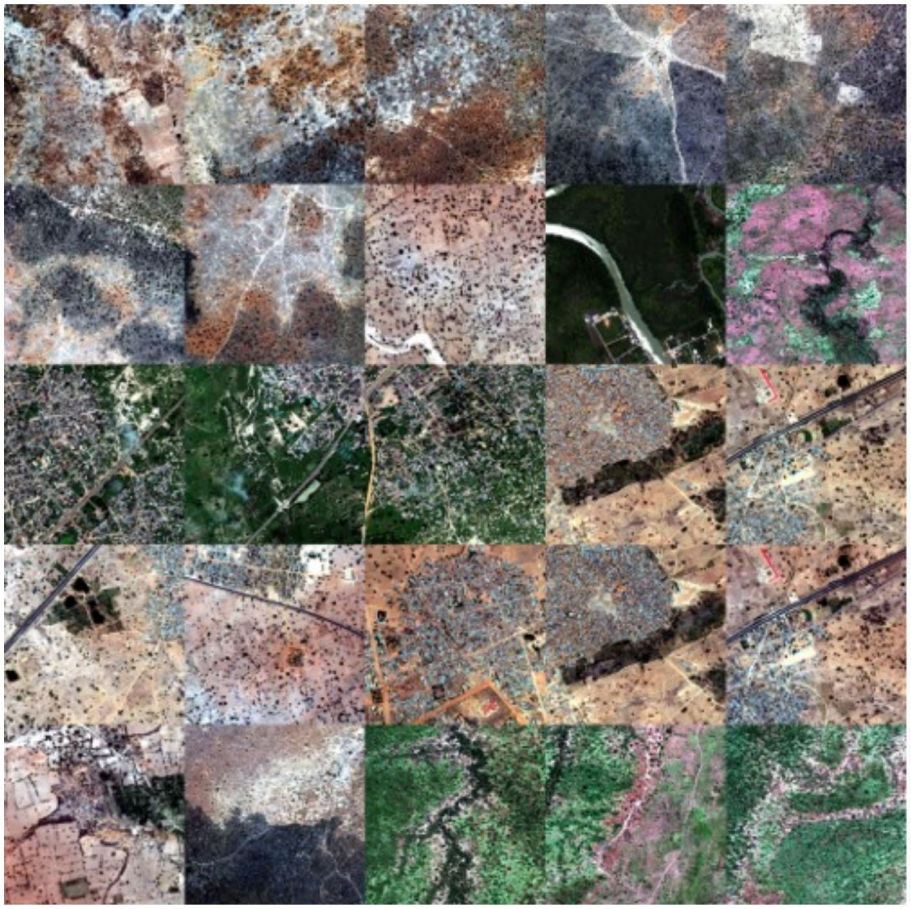

Figure 5.3. Complete dataset

5.2.2 Train

The train of the model is the main interest of this work, as the final goal is to teach the network to recognize the various categories present in a satellite image. The model was trained with the power of a 970gtx GPU which took 240 seconds on average per epoch. One hundred epochs allowed to reach a value of jaccard similarity equal to 0.17. The value of jaccard similarity can vary from 0 to 1, where 0 means complete failure and perfect overlap.
The selected hyperparameters are the following:

- Optimization algorithm = Adam
- Batch size = 4
- Learning rate = 0.01

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 3, 160, 160)	0	
conv1_1 (Conv2D)	(None, 64, 160, 160)	832	input_1[0][0]
conv2d_1 (Conv2D)	(None, 64, 160, 160)	16448	conv1_1[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 64, 80, 80)	0	conv2d_1[0][0]
conv2d_2 (Conv2D)	(None, 64, 80, 80)	16448	max_pooling2d_1[0][0]
conv2d_3 (Conv2D)	(None, 64, 80, 80)	16448	conv2d_2[0][0]
max_pooling2d_2 (MaxPooling2D)	(None, 64, 40, 40)	0	conv2d_3[0][0]
conv2d_4 (Conv2D)	(None, 64, 40, 40)	16448	max_pooling2d_2[0][0]
conv2d_5 (Conv2D)	(None, 64, 40, 40)	16448	conv2d_4[0][0]
max_pooling2d_3 (MaxPooling2D)	(None, 64, 20, 20)	0	conv2d_5[0][0]
conv2d_6 (Conv2D)	(None, 64, 20, 20)	16448	max_pooling2d_3[0][0]
conv2d_7 (Conv2D)	(None, 64, 20, 20)	16448	conv2d_6[0][0]
max_pooling2d_4 (MaxPooling2D)	(None, 64, 10, 10)	0	conv2d_7[0][0]
conv2d_8 (Conv2D)	(None, 64, 10, 10)	16448	max_pooling2d_4[0][0]
conv2d_9 (Conv2D)	(None, 64, 10, 10)	16448	conv2d_8[0][0]
up_sampling2d_1 (UpSampling2D)	(None, 64, 20, 20)	0	conv2d_9[0][0]
cropping2d_1 (Cropping2D)	(None, 64, 20, 20)	0	conv2d_7[0][0]
concatenate_1 (Concatenate)	(None, 128, 20, 20)	0	up_sampling2d_1[0][0] cropping2d_1[0][0]
conv2d_10 (Conv2D)	(None, 64, 20, 20)	32832	concatenate_1[0][0]
conv2d_11 (Conv2D)	(None, 64, 20, 20)	16448	conv2d_10[0][0]
up_sampling2d_2 (UpSampling2D)	(None, 64, 40, 40)	0	conv2d_11[0][0]
cropping2d_2 (Cropping2D)	(None, 64, 40, 40)	0	conv2d_5[0][0]
concatenate_2 (Concatenate)	(None, 128, 40, 40)	0	up_sampling2d_2[0][0] cropping2d_2[0][0]
conv2d_12 (Conv2D)	(None, 64, 40, 40)	32832	concatenate_2[0][0]
conv2d_13 (Conv2D)	(None, 64, 40, 40)	16448	conv2d_12[0][0]
up_sampling2d_3 (UpSampling2D)	(None, 64, 80, 80)	0	conv2d_13[0][0]
cropping2d_3 (Cropping2D)	(None, 64, 80, 80)	0	conv2d_3[0][0]
concatenate_3 (Concatenate)	(None, 128, 80, 80)	0	up_sampling2d_3[0][0] cropping2d_3[0][0]
conv2d_14 (Conv2D)	(None, 64, 80, 80)	32832	concatenate_3[0][0]
conv2d_15 (Conv2D)	(None, 64, 80, 80)	16448	conv2d_14[0][0]
up_sampling2d_4 (UpSampling2D)	(None, 64, 160, 160)	0	conv2d_15[0][0]
cropping2d_4 (Cropping2D)	(None, 64, 160, 160)	0	conv2d_1[0][0]
concatenate_4 (Concatenate)	(None, 128, 160, 160	0	up_sampling2d_4[0][0] cropping2d_4[0][0]
conv2d_16 (Conv2D)	(None, 64, 160, 160)	32832	concatenate_4[0][0]
conv2d_17 (Conv2D)	(None, 64, 160, 160)	16448	conv2d_16[0][0]
zero_padding2d_1 (ZeroPadding2D	(None, 64, 160, 160)	0	conv2d_17[0][0]
conv2d_18 (Conv2D)	(None, 10, 160, 160)	650	zero_padding2d_1[0][0]

Total params: 346,634
Trainable params: 346,634
Non-trainable params: 0

Figure 5.4. U-Net structure

The saving of the model was made on the best epoch, because the trend of jaccard similarity has proved not to be constantly increasing, presenting various oscillations. For this reason the early stopping was avoided by stopping the execution at a certain time because the repeated oscillations could have stopped the training before the model reached its maximum accuracy.

5.2.3 Test

The model was tested with satellite images that were not indexed, ie they did not show polygon segmentation.
As shown in figure 5.4, the model is able to perfectly recognize the class relative to buildings and tracks in the image.

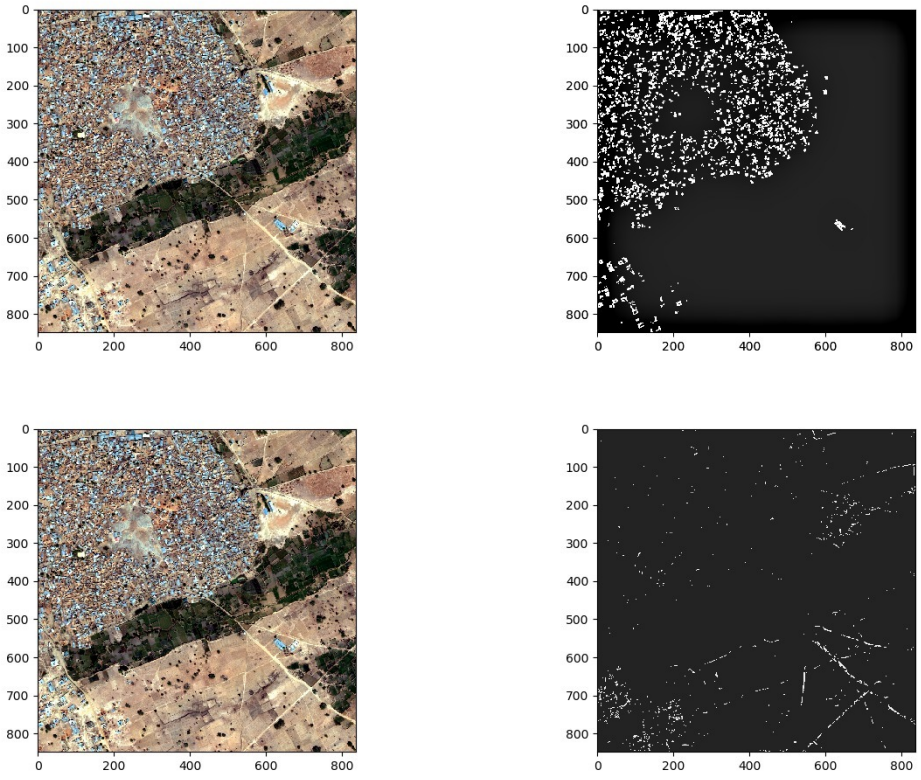

Figure 5.5. Test image

5.2.4 Classification

For the final processing and the consequent classification of a satellite image to recognize the categories within it, a function has been used that allows to scroll an image of arbitrary size through the so-called sliding window.
This technique allows to analyze the high resolution image by sliding a fixed size window on it (for the purpose of the work a 160 x 160 window has been chosen), succeeding in segmenting and classifying each window and finally processing the results of everything that has been recognized within it. The final result will be nothing more than a sum of the percentages calculated on each individual window of the image.

Figure 5.6. Sliding window example

The software eventually returns the image with the relative segmentation in polygons for each single category of elements that has been found inside it. The segmentation related to the image provided in input is also cataloged in an external readable file from which it is possible to obtain the individual polygons of the categories.

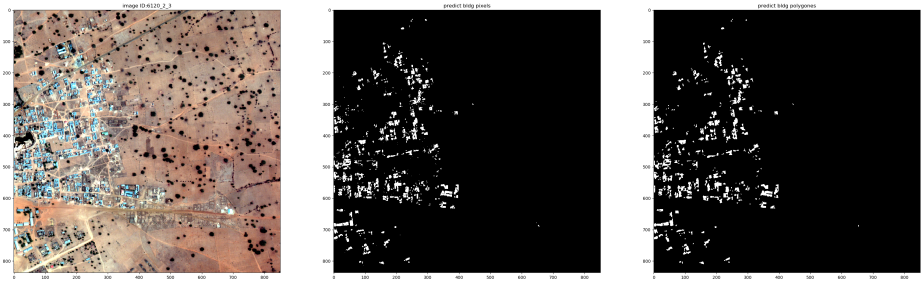

Figure 5.7. Segmentation of the input image

Finally, the results obtained from the segmentation are elaborated as a whole to obtain a pie chart that indicates the actual land cover of the image provided as input.

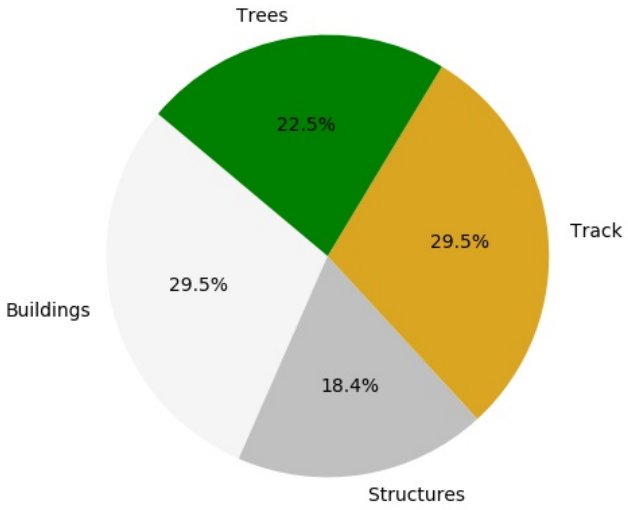

Figure 5.8. Land cover

Chapter 6

Conclusions and Future Directions

The U-Net architecture has proved to be an effective method to analyze remote sensing problems, specifically the ones related to labeling and classification of satellite and territorial images. Thanks to the segmentation the U-Net model is able to recognize the various elements of the images.
The extraction of statistics and information necessary for the case of interest automatically from satellite maps was in fact possible thanks to the deep learning combined with the properties of neural networks.
The introduction of segmentation has played a key role in this project. By scanning the image by means of the sliding window scrolling through all the directions with an "eye" of size fixed 160x160 it has been possible to recognize the various percentages of land occupation. This was a step forward towards the elimination of false positives, because previously, the eye would recognize only the preponderant category within a specific window, labeling it with the recognized one, thus inevitably creating false positives.
Despite this, the methodology used to extract the land cover statistics developed in this thesis project has the potential to be considered innovative, but retains properties that make it approximate for the purposes of calculating the official statistics, as it does not it is still completely able to return the precise percentage of employment of the various classes.

A subsequent approach to satellite image analysis is the introduction of GANs [1](Generative Adversarian Networks). The GANs introduce an image generator inside the model that allows the entire architecture to work with fewer images, producing significant results. The goal of the model through the various epoch will be to generate images more and more congruous to the original ones thanks to a U-Net that acts as a discriminator.
In the near future, with the considerable progress made by NASA through the Copernicus project to make satellite images public, it is estimated that thousands of new businesses and startups will be created with the aim of analyzing the territories photographed by the satellites. The advent of public datasets would also lead to the possibility of studying the territory over the years, with the possibility of being able to determine past changes, present and estimate future ones.

[1]Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014).

Bibliography

[1] Al-Tarawneh, M. S. (2012). *Lung cancer detection using image processing techniques. Leonardo Electronic Journal of Practices and Technologies, 11*(21), 147-58.

[2] Beringer, J., Arguin, J. F., Barnett, R. M., Copic, K., Dahl, O., Groom, D. E., ... & Yao, W. M. (2012). *Review of particle physics. Physical Review D-Particles, Fields, Gravitation and Cosmology*, 86(1).

[3] Bischke, B., Helber, P., Folz, J., Borth, D., & Dengel, A. (2017). *Multi-task learning for segmentation of building footprints with deep neural networks.* arXiv preprint arXiv:1709.05932.

[4] Castelluccio, M., Poggi, G., Sansone, C., & Verdoliva, L. (2015). *Land use classification in remote sensing images by convolutional neural networks.* arXiv preprint arXiv:1508.00092.

[5] Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016, October). *3D U-Net: learning dense volumetric segmentation from sparse annotation. In International Conference on Medical Image Computing and Computer-Assisted Intervention* (pp. 424-432). Springer, Cham.

[6] Electric Cars, Solar Panels & Clean Energy Storage | Tesla. (n.d.). Retrieved from https://www.tesla.com/

[7] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). *Generative adversarial nets. In Advances in neural information processing systems* (pp. 2672-2680).

[8] He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[9] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). *Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups.* IEEE Signal processing magazine, 29(6), 82-97.

[10] Huang, P. S., He, X., Gao, J., Deng, L., Acero, A., & Heck, L. (2013, October). *Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM international conference on Conference on information & knowledge management* (pp. 2333-2338). ACM.

[11] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems* (pp. 1097-1105).

[12] Long, J., Shelhamer, E., & Darrell, T. (2015). *Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3431-3440).

[13] McCulloch, W. S., & Pitts, W. (1943). *A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics*, 5(4), 115-133.

[14] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). *Playing atari with deep reinforcement learning.* arXiv preprint arXiv:1312.5602.

[15] Ng, A. (2016, May 26). "AI is the new electricity!" Electricity transformed countless industries; AI will now do the same. pic.twitter.com/dGFEUMSmVj. Retrieved from https://twitter.com/AndrewYNg/status/735874952008589312

[16] Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems* (pp. 91-99).

[17] Robins, D. L., Casagrande, K., Barton, M., Chen, C. M. A., Dumont-Mathieu, T., & Fein, D. (2014). *Validation of the modified checklist for autism in toddlers, revised with follow-up* (M-CHAT-R/F). Pediatrics, 133(1), 37-45.

[18] Ronneberger, O., Fischer, P., & Brox, T. (2015, October). *U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241).* Springer, Cham.

[19] Rosenblatt, F. (1958). *The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review*, 65(6), 386.

[20] Sabour, S., Frosst, N., & Hinton, G. E. (2017). *Dynamic routing between capsules. In Advances in Neural Information Processing Systems* (pp. 3856-3866).

[21] Setio, A. A. A., Ciompi, F., Litjens, G., Gerke, P., Jacobs, C., Van Riel, S. J., ... & van Ginneken, B. (2016). *Pulmonary nodule detection in CT images: false positive reduction using multi-view convolutional networks. IEEE transactions on medical imaging*, 35(5), 1160-1169.

[22] Simonyan, K., & Zisserman, A. (2014). *Very deep convolutional networks for large-scale image recognition.* arXiv preprint arXiv:1409.1556.

[23] Venkatesan, R., & Li, B. (2017). *Convolutional Neural Networks in Visual Computing: A Concise Guide.* CRC Press.

[24] Wattenberg, M., Viégas, F., & Johnson, I. (2016). *How to use t-sne effectively.* Distill, 1(10), e2.

[25] WorldView-3 Satellite Sensor. (n.d.). Retrieved from https://www.satimagingcorp.com/satellite-sensors/worldview-3/

[26] Zeiler, M. D., & Fergus, R. (2014, September). *Visualizing and understanding convolutional networks. In European conference on computer vision* (pp. 818-833). Springer, Cham.