



République Algérienne Démocratique et Populaire  
University of Abou Bekr Belkaid– Tlemcen  
Science Faculty  
Computer Science Department

**Master Thesis**

**For obtaining the Master's degree in Computer Science**

**Option:** Intelligent Models and Decision

# Theme

## Explainable convolutional networks for covid-19

**Realized by:**

- **DJELLOULI Aymen**

Presented on 2 July 2022 before the jury composed of:

- Mr MERZOUG Mohamed (President)
- Mr HADJILA Fethallah (Supervisor)
- Mr SMAHI Mohamed Ismail (Examiner)



# Acknowledgments

First of all, I thank God Almighty for giving me strength and patience to complete this humble work.

On this occasion, I would like to express my sincere gratitude to my family, especially my parents, where without them this work would not have been possible, and I am also grateful for their sincere support and encouragement during my studies. I also thank my brothers for their help.

With great appreciation I would like to thank my supervisor and director of this research, Dr. HADJILA Fethallah: I would to say thank you for all the attention you have given me, whether during my studies with you or during my work with you on this research, I am also very grateful for your valuable advices and observations that helped nourish my thinking. Thank you for everything and I hope to always meet your expectations.

I also thank the members of the jury who agreed to read and evaluate this work.

I also thank my teachers who were the reason for my success during my university studies. I also thank my friends and everyone who contributed from near or far to help me accomplish this work.

## Table of Contents

GENERAL INTRODUCTION .....	5
CHAPTRE. I: IMAGE CLASSIFICATION.....	8
I.1.    Introduction.....	9
I.2.    Computer vision.....	9
I.3.    Applications of computer vision.....	10
I.4.    Image classification.....	12
I.4.1.    Types of image classification.....	13
I.4.2.    Datasets for image classification.....	13
I.4.3.    Confusion matrix.....	17
I.4.4.    Image Classification Metrics.....	18
I.5.    Conclusion .....	19
CHAPTRE. II: DEEP LEARNING .....	20
II.1.    Introduction.....	21
II.2.    What is deep learning?.....	21
II.3.    Deep learning applications.....	22
II.4.    A brief History of Deep Learning .....	23
II.5.    Artificial Neural Networks.....	25
II.5.1.    Biological neurons .....	25
II.5.2.    The Perceptron .....	26
II.5.3.    The multilayer perceptron .....	27
II.5.4.    Activation function.....	28
II.5.5.    Loss function.....	32
II.5.6.    Gradient descent:.....	33
II.5.7.    Variants of gradient Descent:.....	34
II.5.8.    Adaptive Learning Rate Optimization techniques for Gradient Descent.....	35
II.5.9.    Backpropagation .....	36
II.6.    Deep neural network: .....	37
II.7.    Convolutional Neural Networks .....	38
II.7.1.    Convolutional layers .....	39
II.7.2.    Pooling layers.....	41
II.7.3.    Fully connected layers.....	42
II.7.4.    History of CNNs .....	42
II.7.5.    The famous state-of-the-art CNNs .....	43
II.8.    Transfer Learning.....	43
II.9.    Explainability in convolutional neural networks .....	44

II.9.1.	Importance of explainability in a CNN model .....	45
II.9.2.	Preliminary Methods .....	46
II.9.3.	Activation based methods .....	48
II.9.4.	Gradient based methods .....	49
II.10.	Conclusion .....	55
CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION .....		56
III.1.	Introduction .....	57
III.2.	Related works .....	57
III.3.	Dataset used in this work .....	58
III.4.	Development tools .....	59
III.5.	Implementation workflow .....	60
III.6.	Conclusion .....	71
GENERAL CONCLUSION .....		72
REFERENCES AND BIBLIOGRAPHY .....		74

## List of figures

### Chapter I:

Figure I- 1: Some examples of fundamental computer vision tasks. [6].....	10
Figure I- 2: daily new confirmed COVID-19 deaths per million people in 26 may 2022 .....	11
Figure I- 3: daily new top 7 confirmed COVID-19 cases deaths per million people in 26 may 2022.....	12
Figure I- 4: samples from each class in CIFAR-10 dataset. [10] .....	14
Figure I- 5: samples from ImageNet [16] .....	16

### Chapter II:

Figure II- 1: Diagram showing the relationship between deep learning, machine learning, and artificial intelligence .....	21
Figure II- 2: a model of a biological neuron [27].....	25
Figure II- 3: an example of a perceptron.....	26
Figure II- 4: an example of MLP .....	27
Figure II- 5: Sigmoid function graph .....	29
Figure II- 6: Tanh function graph.....	29
Figure II- 7: ReLU function graph .....	30
Figure II- 8: LReLU function graph.....	30
Figure II- 9: PReLU function graph ( $\alpha = 0.1$ ) .....	31
Figure II- 10: ELU function graph ( $\alpha = 1$ ) .....	31
Figure II- 11: example of convolutional neural network [51].....	39
Figure II- 12: example of RGB image with $(4 \times 4 \times 3)$ in size.....	39
Figure II- 13: example of a convolution operation .....	41
Figure II- 14: Example of an average and max pooling operations with stride length of 2 and a kernel size $2 \times 2$ [85].....	42
Figure II- 15: AlexNet first convolutional layer kernels visualization [47].....	46
Figure II- 16: higher layers filters visualization [67] .....	47
Figure II- 17: example of two dimensional plot of some ImageNet samples where the images are grouped based on their classes for example the dog class in the red box [16].....	48
Figure II- 18: example of saliency via occlusion method [69].....	49
Figure II- 19: example of saliency maps [70] .....	50
Figure II- 20: Different methods of propagating back through a ReLU nonlinearity [72] .....	51
Figure II- 21: Visualization of the data gradient [72] .....	51
Figure II- 22: guided backpropagation example [73].....	52
Figure II- 23: CAM overview [74].....	53
Figure II- 24: Grad CAM overview [73].....	54
Figure II- 25: example of guided backpropagation, Grad-CAM, and guided Grad-CAM for two classes 'Cat' and 'Dog' [73] .....	55

### Chapter III:

Figure III- 1: some samples from the dataset (the first row is for covid-19 samples the second is for lung opacity samples, the third for normal samples and the fourth is for viral Pneumonia samples) .....	59
Figure III- 2: VGG16 training accuracy (top) and loss (bottom) for train and validation dataset .....	61

Figure III- 3: MobileNetV2 training accuracy (top) and loss (bottom) for train and validation dataset .....	62
Figure III- 4: EfficientNetV2L training accuracy (top) and loss (bottom) for train and validation dataset .....	63
Figure III- 5: ResNet101V2 training accuracy (top) and loss (bottom) for train and validation dataset .....	64
Figure III- 6: examples of Grad-CAM results for MobileNetV2.....	65
Figure III- 7: examples of Grad-CAM results for ResNet101V2 .....	66
Figure III- 8: examples of Grad-CAM results for ResNet101V2 with 30 epochs .....	68
Figure III- 9: examples of Grad-CAM results for ResNet101V2 with 30 epochs and alpha=0.569	
Figure III- 10: our web application home page.....	70

## List of tables

### Chapter I:

Table I- 1: list of CIFAR-100 classes [10].....	15
Table I- 2: the template of binary classification the confusion matrix.....	17
Table I- 3: example of a 2×2 confusion matrix.....	17
Table I- 4: actual and predicted labels .....	18
Table I- 5: example of a 4×4 confusion matrix for multi class classification .....	18

### Chapter III:

Table III- 1: training results .....	60
Table III- 2: training results .....	66
Table III- 3: the model's confusion matrix .....	67
Table III- 4: image classification metrics for the model .....	67

# **GENERAL INTRODUCTION**



## **GENERAL INTRODUCTION**

### **Context**

Coronavirus, also called COVID-19, is one of the deadliest viruses the world has seen and there is no doubt that it left its mark on it. This virus originated in China, specifically in the city of Wuhan, in December 2019 and spread frighteningly throughout the world. So far, it has infected about 528 million people and killed more than 6,283,923 since now (as of 26 May 2022) [1]. This virus is characterized by its rapid spread, which made most countries face a shortage of testing means and tools, and the methods available at that time took time to give results and required direct contact between the potential patient and the hospital staff, which exposes them throughout their work period to the risk of infection.

These circumstances made many researchers think about exploiting deep learning algorithms, especially convolutional neural networks, which have proven their efficiency in the field of image processing and computer vision, as they have become the standard in this field after their superiority over traditional image processing methods.

Among the most widely used applications of convolutional neural networks in computer vision is the image classification based on the features extracted and learned by the network during the training phase. It is also used in objects detection, localization, image segmentation, and many other applications. More details will be covered in the coming chapters.

Deep learning is powerful enough to make it be invested in all areas of medicine, from drug discovery to medical advice and decision-making. Also, the success of convolutional neural networks in computer vision in recent years contributed to making it suitable for medical image processing, which would be a helpful tool for radiologists, as it provides satisfactory results in a short time and at a lower cost.

### **Problematic**

The architecture of neural networks makes them capable of performing various tasks in various fields. With different neural structures, the function of the network varies. But the complexity of the neural networks structure makes them difficult to interpret compared to other machine learning models such as decision trees, as the

## **GENERAL INTRODUCTION**

model can provide accurate results, but on a wrong basis, perhaps because of the presence of noise in the data, using biased dataset or other factors.

This makes neural networks like a black box that its working principle is difficult to predict, making its results unreliable or even dangerous in some critical fields and it cannot be used alone without the presence of an expert, regardless of their accuracy.

Our objective is to design a deep convolutional network for covid 19 detection, which is both accurate and explainable.

## **Contributions**

Convolutional neural networks also suffer from interpretation problems. Fortunately there is a method for generating visual interpretations called Grad-CAM [2] to make convolutional neural networks more transparent and explainable.

The goal of this master thesis is to build a system capable of accurately predicting whether a person is healthy, infected with the Coronavirus, or has another medical condition (Lung Opacity, Viral Pneumonia) only through a chest x-ray, besides the possibility of visually explaining what the system sees and what made it predict the result that it predicted using Grad-CAM .

## **Manuscript plan**

This master thesis is divided into three chapters.

### **Chapter 1:**

In the first chapter, the contribution of artificial intelligence methods in the field of computer vision, especially in the processing of medical images, will be addressed.

### **Chapter 2:**

The second chapter aims to present the basics of deep learning and neural networks.

### **Chapter 3:**

The last chapter will present the steps and methods used to achieve the system.

**CHAPTRE. I:  
IMAGE  
CLASSIFICATION**

### I.1. Introduction

The computer vision field has been rapidly developing, finding real-world applications, and even surpassing humans in solving some of the visual tasks. All of this thanks to the recent advances in artificial intelligence and deep learning.

### I.2. Computer vision

Computer vision is a field of artificial intelligence (AI) that aims to simulate the human visual system to make computers able to extract, analyze and understand meaningful information automatically from visual inputs such as images and videos to make decisions based on that information.

One of the most important techniques used in computer vision is convolutional neural networks (CNNs), which need a lot of data, as the network is trained on it in order to distinguish the differences and eventually recognize the images. Recurrent Neural Networks (RNNs) is also used for video applications to make computers able to understand how frames relate to each other. [3] [4]

The importance of computer vision appears through real-world applications in various domains and even in daily life. The availability of visual information used in training computer vision applications through smart phones, security systems, traffic cameras and other sources contributed to the growth of these applications. Some examples of fundamental computer vision tasks:

- **Image classification:** is a subset of classification problem, where the input image is accurately categorized into its own category.
- **Object detection:** Image classification and localization are used to identify the object class of an image and then it discovers and frames its positions in the image or video by bounding boxes.
- **Object tracking:** this task is often used with videos by detecting and tracking the wanted object.
- **Image segmentation:** where an image is partitioned into different area of pixels called image objects.

## CHAPTRE. I: IMAGE CLASSIFICATION

- **Content-based image retrieval:** to retrieve images from large data stores based on its content automatically instead of using metadata tags associated manually with them. [3] [5]

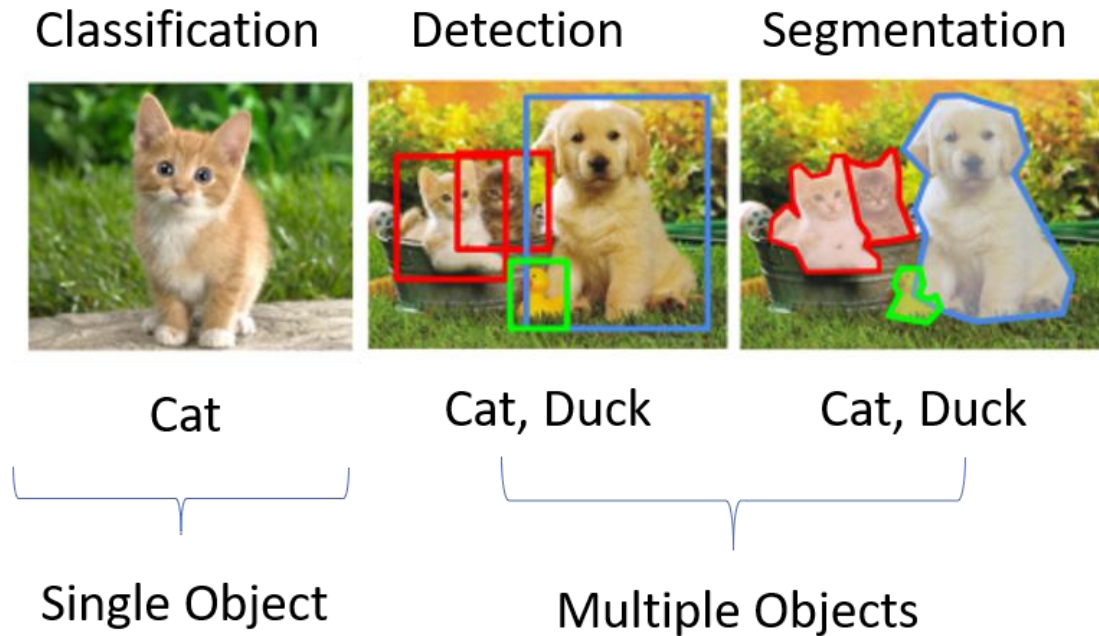


Figure I- 1: Some examples of fundamental computer vision tasks. [6]

### I.3. Applications of computer vision

Computer vision is used in many different areas, whether they are complex or even related to daily life, as its applications have greatly contributed to improving performance in these areas. Uses of computer vision can be found in any field, for example, applications of computer vision are present in the field of manufacturing where it can be used to track productivity analytics in order to obtain data that provides an overview of time management, workplace collaboration, and employee productivity. Another application is visual inspection of equipment, especially personal protective equipment, such as mask detection or helmet detection, in order to monitor compliance with safety protocols at construction sites or factories. It is also used in automated visual inspection for quality control and management in factories. [7]

Another field in which computer vision can be found is the field of transportation where computer vision technic are used for self-driving cars, vehicle classification for traffic analysis systems, moving violations detection in order to automate the detection of traffic violations and to reduce unsafe driving behavior, detect parking occupancy using visual parking monitoring, automated license plate recognition and extraction

## CHAPTRE. I: IMAGE CLASSIFICATION

from either images or videos, The detection of pedestrians that used in autonomous driving, traffic management, and law enforcement, and traffic sign detection and recognition. [7]

The health care field has also benefited from the impressive development of computer vision technologies, as these technologies are used in many medical applications to help the medical staff do their work to the fullest. Among the common examples is the use of computer vision to help doctors make early diagnosis of cancer such as breast and skin cancer by recognizing input images such as magnetic resonance imaging (MRI) scans and then discovering the differences between cancerous and non-cancerous images in order to classify these images, and it also helps to detect brain tumors in MRI. Doctors were also able to detect musculoskeletal and neuromuscular diseases such as strokes, walking and balance problems by analyzing the patient's movement.

With the spread of the Corona virus in the world and its transformation into a global pandemic, it is necessary to develop methods to combat this epidemic.

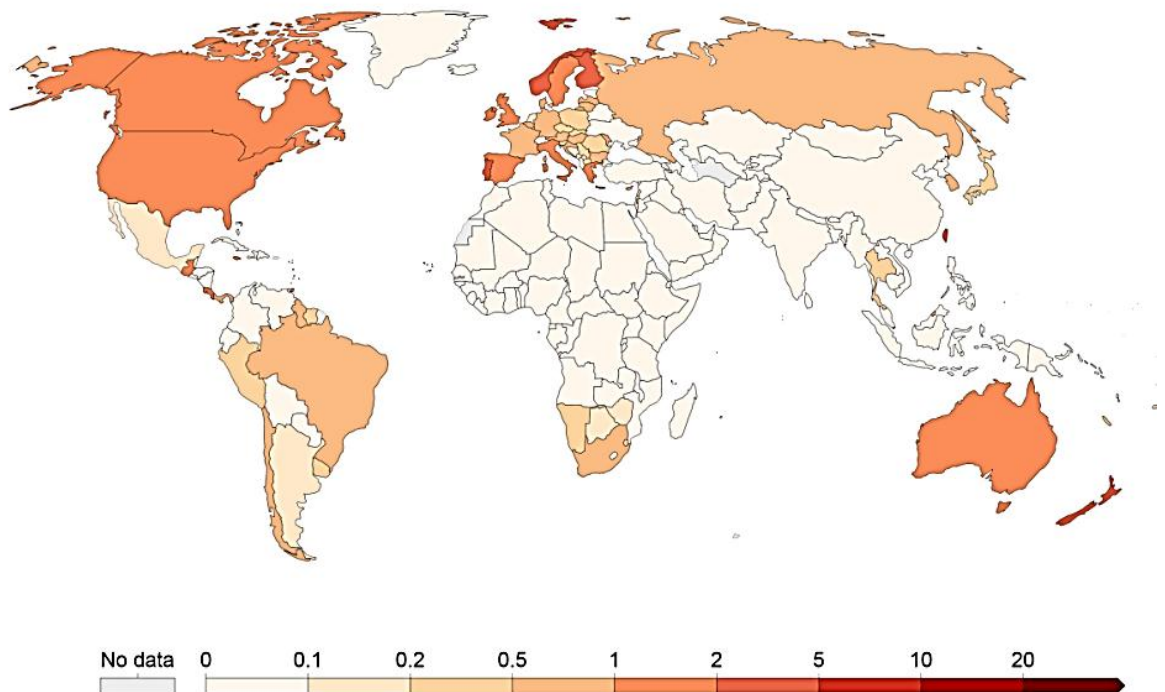


Figure I- 2: daily new confirmed COVID-19 deaths per million people in 26 may 2022 [1]

## CHAPTRE. I: IMAGE CLASSIFICATION

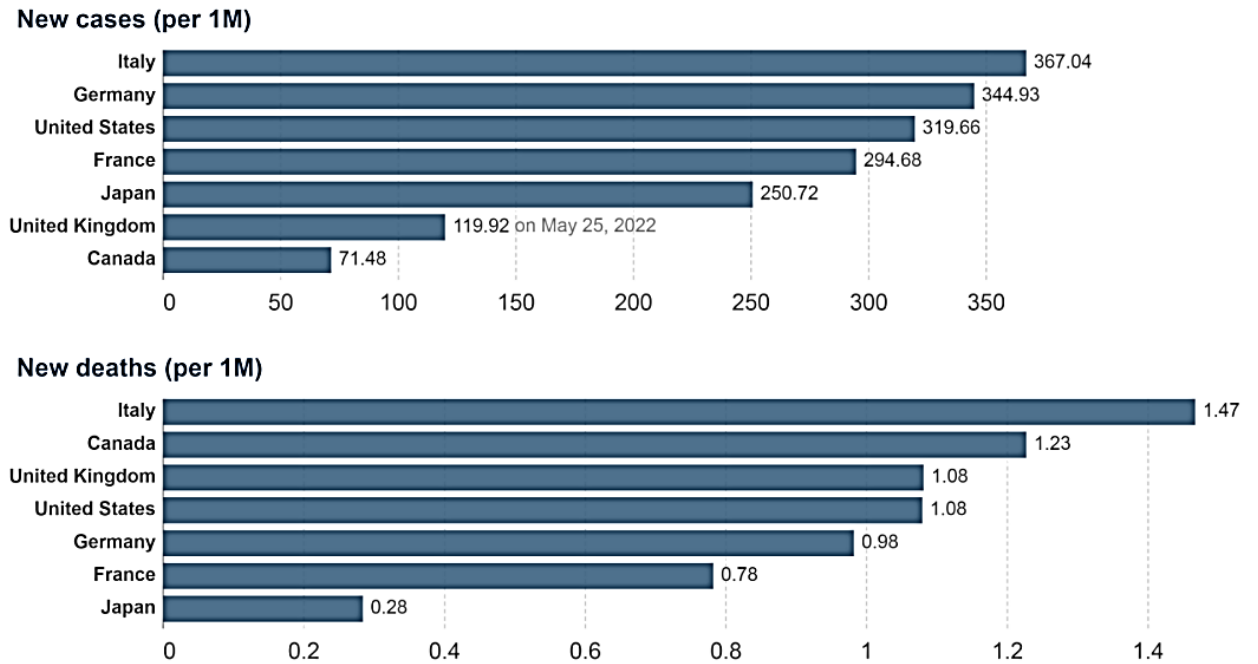


Figure I- 3: daily new top 7 confirmed COVID-19 cases deaths per million people in 26 may 2022 [1]

Computer vision technologies have helped doctors in their fight against the epidemic by providing the possibility of diagnosing infection in a short time. Where one of the most famous models used in diagnosing the Corona virus by examining chest x-rays is a model called COVID-Net. Computer vision systems can also be used to detect the use of masks to limit and control the spread of the Corona virus.

### I.4. Image classification

Among the major tasks involved in the field of computer vision such as image localization, image segmentation, and object detection, image classification is the fundamental task as it forms the basis of the other tasks in this field.

Image classification applications can be found in many areas, such as medical imaging, traffic control systems, monitoring systems, self-driving cars, and many more applications.

The aim of image classification is to assign a specific label or more labels (in case of multi label classification) to an image based on the features extracted from this image. Classification problems are mainly divided into two categories: supervised and unsupervised classification techniques.

## **CHAPTRE. I: IMAGE CLASSIFICATION**

Supervised image classification algorithms like neural networks use labeled samples to train the model and gradually it will be able to classify new unknown data, on the other hand in unsupervised image classification or clustering the algorithms discover the patterns that can be used to differentiate between the different classes and cluster the images based on it. Among the most popular algorithms used for unsupervised image classification there are K-means that groups images into K clusters based on the similarity of their characteristics, and ISODATA or Iterative Self-Organizing Data Analysis Technique where a different number of clusters is allowed instead of fixed number like in the case of K-means. In the two categories to get a good classification the used algorithm must maximize the similarity and minimize the distance between the data points in each class or cluster.

### **I.4.1. Types of image classification**

Image classification problems can be divided into two types based on the classification task complexity, where the single label classification is the first type and the most common in supervised image classification problems. It aims to assign one label to each image for this reason its output is a vector with number of elements equal to the number of classes where each element represents the probability that the input image belongs to each class, and the maximum value of the probabilities is taken as label for the image. The single label classification can be either binary classification (two classes) or multiclass (more than two classes) classification.

The second type is multi label classification, as the name suggests in this type an image can have more than one label which increases the complexity of the problem. For example multi label classification appears in the domain of medical imaging where a patient can be diagnosed with more than one disease from its X-ray images. [8]

### **I.4.2. Datasets for image classification**

The image classification process is highly dependent on the training data, as the quality of the dataset used in the model training contributes to its accuracy. Comparatively, due to the simplicity of the work the process of collecting a data set used in image classification may be somewhat faster compared to other tasks such as image segmentation. In addition to collecting images, for the supervised learning approach, each image in the dataset must be attached with a label indicating the class of



## CHAPTRE. I: IMAGE CLASSIFICATION

this image, or organize the dataset in the form of folders where each folder represents a class and includes all images belonging to the same class.

The most challenging part at this stage is ensuring that the data set is bias free and is balanced to ensure the model's prediction accuracy.

Here are some of the most popular data used to classify images

- **MNIST:** stands for Modified National Institute of Standards and Technology, it's a dataset contains handwritten digits gray scale images. MNIST is a subset of a larger dataset called NIST created in 1998, the images size in MNIST is fixed to  $28 \times 28$  and the digits are scaled and centered in each image. It contains 60,000 images for training and 10,000 images for testing [9].
- **CIFAR-10:** stands for Canadian Institute For Advanced Research, it is one of the most popular dataset used in computer vision and machine learning algorithms, collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. It contains 60,000 color images with size  $32 \times 32$  where there are 50,000 images for training and 10,000 images for testing, divided into 10 classes where each class contains 6,000 images. Its classes include airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks [10] [11].

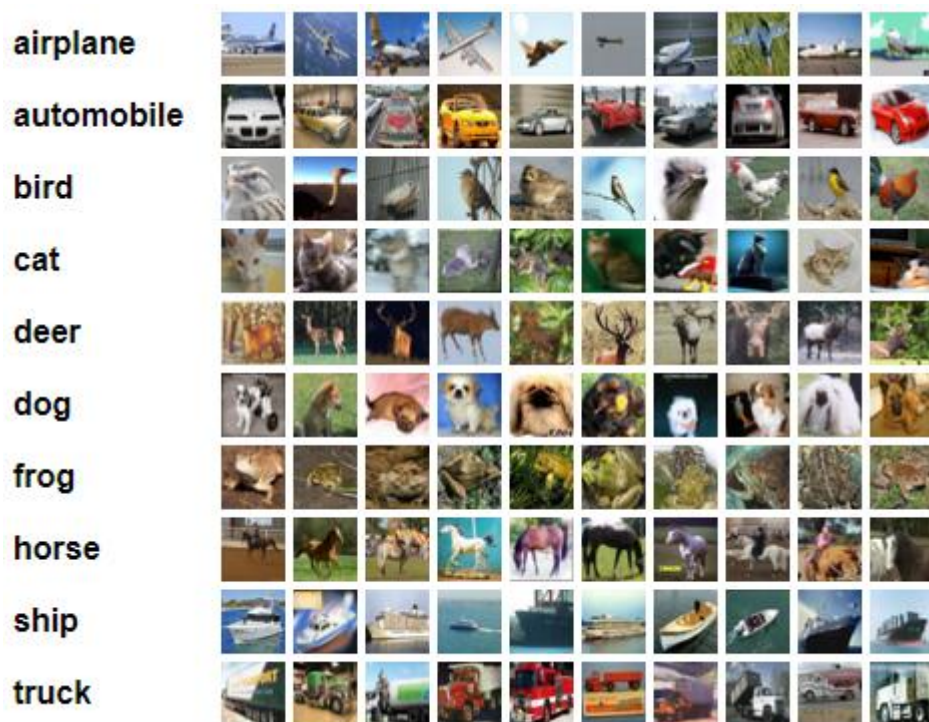


Figure I- 4: samples from each class in CIFAR-10 dataset. [10]

## CHAPTRE. I: IMAGE CLASSIFICATION

- **CIFAR-100:** also collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Like CIFAR-10, it contains 60,000 color images with size  $32 \times 32$ , but this time it has 100 classes grouped into 20 super classes and each class contains 600 images where there are 500 images for training and 100 images for testing for each class.

Superclass	Classes
aquatic mammals	beaver, dolphin, otter, seal, whale
Fish	aquarium fish, flatfish, ray, shark, trout
Flowers	orchids, poppies, roses, sunflowers, tulips
food containers	bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household furniture	bed, chair, couch, table, wardrobe
Insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
People	baby, boy, girl, man, woman
Reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
Trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

Table I- 1: list of CIFAR-100 classes [10]

- **FER-2013:** stands Facial Expression Recognition, it is a dataset for emotion detection training based on the facial expression, it consists of 35,887 gray scale images of faces with size of  $48 \times 48$  pixel. It has seven classes include the following: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral. It is downloadable on kaggle [12].

## CHAPTRE. I: IMAGE CLASSIFICATION

- **ImageNet:** it is a huge dataset contains more than 14 million images categorized into 22,000 classes presented by Fei-Fei Li in 2009 [13]. The average resolution of these images is 469x387 and usually cropped to 256x256 or 224x224 in the image preprocessing step [14]. In 2010 Fei-Fei Li suggested using the ImageNet to create an annual model training challenge called ILSVRC or ImageNet Large Scale Visual Recognition Challenge. ILSVRC uses only 1000 classes and approximately 1.2 million images for training, 50,000 for validation and 100,000 images for testing. And since then ILSVRC became the standard for image classification, and it has contributed to the emergence of convolutional neural networks and deep learning technologies since they dominated the leaderboard for this challenge since 2012 [15].



Figure I- 5: samples from ImageNet [16]

## CHAPTRE. I: IMAGE CLASSIFICATION

### I.4.3. Confusion matrix

Confusion or error matrix is a simple table that visualize the model performance, where its rows represent the actual (true) classes and its columns represent the predicted classes. In the case of binary classification the confusion matrix (the  $2 \times 2$  matrix in the blue box) will take the following template:

		Predicted label	
		Positive (P)	Negative (N)
Actual label	Total Population= $P+N$		
	Positive (P)	True Positive (TP)	False Negative (FN)
	Negative (N)	False Positive (FP)	True Negative (TN)

Table I- 2: the template of binary classification the confusion matrix

When the actual label is positive and if the model predict it as positive (belongs to the same class) this called true positive otherwise is called false negative and vice versa when the actual label is negative and if the model predict it as negative this called true negative else is called false positive.

For better understanding let say that a model has to predict if an input image is a dog or not, the test dataset contains ten images five images for dogs and five images for cats. Assuming that the model predictions say there are seven dog images and only three cat images. The confusion matrix of this model will be like this:

		Predicted label	
		Positive (P)=7	Negative (N)=3
Actual label	$P+N=10$		
	Dog or P=5	5	0
	Cat or N=5	2	3

Table I- 3: example of a  $2 \times 2$  confusion matrix

The same logic can be applied to the multi class classification, let say instead of predict if an image is for a dog or not the model has to classify this image into four classes Dog, Cat, Rabbit, and Fish. The following table represents the test dataset (contains ten images) labels and the model predictions:

## CHAPTRE. I: IMAGE CLASSIFICATION

Actual labels	Cat	Dog	Dog	Fish	Rabbit	Cat	Rabbit	Fish	Dog	Cat
Predicted labels	Cat	Dog	Cat	Fish	Dog	Cat	Rabbit	Fish	Dog	Dog

Table I- 4: actual and predicted labels

The confusion matrix will be like this:

		Predicted label			
		Population=10	Cat =3	Dog = 3	Fish = 2
Actual label	Cat = 3	2	1	0	0
	Dog = 3	1	2	0	0
	Fish = 2	0	0	2	0
	Rabbit = 2	0	1	0	1

Table I- 5: example of a 4×4 confusion matrix for multi class classification

### I.4.4. Image Classification Metrics

In order to ensure accurate results, the process of model evaluation and monitoring is essential to determine its performance and to compare it with other models. In supervised classification problems this evaluation can be done using some statistical metrics based on the confusion matrix.

- **Accuracy:** it is the most metric used to evaluate the machine learning models. It measures the ratio of correct predictions to the total number of predictions without taking into account the type of error [17]. For binary classification the accuracy formula is the following:

$$\frac{TP+TN}{Total\ Population} \dots\dots\dots Eq\ I- 1$$

For multi class classification the formula will be like this:

$$\frac{correct\ predictions}{Total\ Population} \dots\dots\dots Eq\ I- 2$$

When the class is unbalanced the accuracy can be misleading in the quality of the model. Therefore, using other metrics would be useful [5].

## CHAPTRE. I: IMAGE CLASSIFICATION

- **Precision:** it measures for each class the ratio of correct positive predictions to the total number of positive predictions (how many predictions that an image belongs to a class are correct). It is calculated as follows:

$$\frac{TP}{TP+FP} \quad \dots\dots\dots\text{Eq I- 3}$$

- **Recall:** it measures for each class the ratio of correct positive predictions to the total number of the class labels (how many correct predictions that an image belongs to its class). It is calculated as follows:

$$\frac{TP}{TP + FN} \quad \dots\dots\dots\text{Eq I- 4}$$

- **F1 Score:** it takes the advantage of both precision and recall. It is calculated as follows:

$$\frac{2 \times Precision \times Recall}{Precision + Recall} \quad \dots\dots\dots\text{Eq I- 5}$$

### I.5. Conclusion

In this chapter, we discussed the definition of computer vision and its applications, and we also highlighted one of its basic tasks, which is the classification of images.

Due to the growth of computer power and with the expansion of deep learning in 2012, deep models have been adapted in the task of image classification and it gave significantly better performance compared to the classic methods, and the goal of the next chapter it is to introduce the field of deep learning, in particular convolutional neural networks, and how to explain them visually.

# **CHAPTRE. II: DEEP LEARNING**

### II.1. Introduction

In this chapter, we will address an introduction to the field of deep learning, a brief historical view of its development, the basic concepts in deep learning, and their role in bringing deep learning to what it is today. Then we will address the base of this field, the artificial neural networks, after that we pass to one of its most popular variant convolutional neural network then we will conclude this chapter by some method that used to make this networks more explainable.

### II.2. What is deep learning?

Deep learning is a subfield of machine learning, which is a subfield of artificial intelligence. Its algorithms depend on the artificial neural networks (ANN) that are mathematical models aim to simulate the behavior of the human brain.

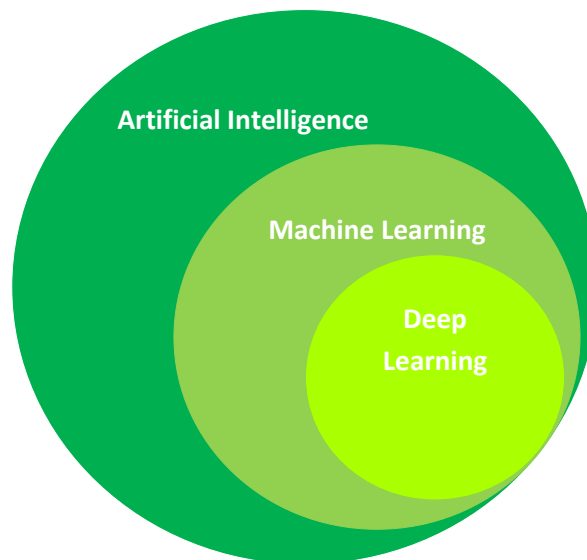


Figure II- 1: Diagram showing the relationship between deep learning, machine learning, and artificial intelligence

It contributes in many applications and services of artificial intelligence that improve the performance and accuracy of tasks automatically without human intervention, whether simple applications such as the digital assistant in our phones, and even complex applications such as self-driving cars.

Deep learning differs from machine learning in methods and data used in learning. Machine learning algorithms use structured labeled data it means that before the training, raw data goes first through some pre-processing by the experts to extract features from it, which is often a complex operation and require a good knowledge of



## **CHAPTRE. II: DEEP LEARNING**

the problem domain. So features extraction in machine learning requires human intervention, however in deep learning there is less dependency on human experts by eliminating this data preprocessing. These algorithms can process unstructured data (such as text, images, and audio), and automate feature extraction.

Many of the foundations and models of deep learning existed in the 1990s, but deep learning didn't really take off until 2012, during the ImageNet computer vision competition, where a team of researchers led by Geoffrey Hinton developed a neural network that performed better than any other algorithm at that time. But why did it take so long for the technologies we have today to emerge?

Well, there are two main reasons for that. The first is that to work well, a neural network needs to be trained on a very large amount of data, sometimes exceeding millions or tens of millions of data. But in the 1990s, this amount of well-categorized and indexed data was not available. After the advent of the Internet, smartphones, and the Internet of things, it has become possible to begin collecting large amounts of data such as images, texts, and audio that can be used in deep learning.

The second reason is that the power of computers in the 80s and 90s did not allow this, where training a neural network requires a lot of time and a lot of computation power. With the development of CPUs and GPUs, it became possible to train neural networks and get good results.

### **II.3. Deep learning applications**

Deep learning has become a part of our daily lives as we deal with its applications in the real world, sometimes without even knowing it. Deep learning applications are used in many fields. Some of these examples include:

Among the most famous deep learning applications that we always deal with in our daily lives are virtual assistants such as Google Assistant, Siri or Alexa, which can manage the device automatically as it learns by interacting with the user more about the user's voice and even his habits as well. Virtual assistants also provide the ability to recognize and transcribe the speech to text or even translate it using natural language processing. Virtual assistants are an extension of the idea of a Chabot used in customer service that also uses deep learning for natural language processing to answer customer questions or route them into a human user. Another example of the applications used daily is the recommendation systems used in movie streaming service platforms, social media, as well as online stores.

## CHAPTRE. II: DEEP LEARNING

Deep learning can be found in special fields, for example in the field of law enforcement, where deep learning algorithms are used to extract patterns that are suspected to be patterns of fraudulent or criminal activity. It is also used in the field of financial services to predict stock prices, assess business risks, and fraud detection.

Another critical area is the healthcare area that has benefited from deep learning capabilities, especially since the digitization of medical records, where image recognition applications help doctors early diagnosis of diseases in a short time. In addition to medical imaging, deep learning is used to analyze genomes and discover new drugs [18] [19].

### II.4. A brief History of Deep Learning

The history of deep learning dates back to 1943, when Walter Bates and Warren McCulloch published a scientific paper entitled: “A Logical Calculus of the ideas immanent in nervous activity” [20] in which they explained how they could create a computer model inspired by the functioning of biological neurons called “Threshold Logic Unit”. This name comes from the fact that their model was designed to process logic inputs (0 or 1). They showed that with this model, it was possible to reproduce certain logic functions, like the AND and the OR gates, and it would be possible to solve any boolean logic problem by connecting several of these functions to each other, like brain neurons. After this announcement, there was an excessive craze for artificial intelligence. Even some people thought that in a few years we could develop artificial intelligence capable of completely replacing human beings! Of course, this was not the case, because although this model lays the foundations for Deep Learning, it has many drawbacks, notably, it didn't have a learning algorithm, and the values of the parameters must be found in advance to use it for real world problems.

In 1957, an American psychologist found a way to improve this model by proposing the first learning algorithm in the history of Deep Learning. He is Franck Rosenblatt, the inventor of the Perceptron [21], an artificial neuron, which is activated when the weighted sum of its inputs exceeds a certain threshold. And it also has a learning algorithm that allows finding the values of the parameters in order to get the desired output. This invention gained huge international attention where the New York Times published an article titled “New Navy Device Learns By Doing” [22]. This led to exaggerated optimism, as it was thought that with perceptron, it would be possible to build machines that could read, speak, walk, and even have a conscience.

## CHAPTRE. II: DEEP LEARNING

A few years later, these promises could not be kept, in 1969, Marvin Minsky and Seymour Papert wrote a book entitled “Perceptrons” [23], which highlighted the many limitations of the Perceptron. For example, because it is a linear model, a single perceptron could not learn the XOR function, which was easily solved by creating a network of them. [24]

From 1974 to 1980, there was no funding for AI research. This period is known as the first winter for artificial intelligence. In 1980s although the concept of back propagation existed in the early 1960s, when Henry J. Kelley developed the basics of the continuous back propagation model in 1960, and in 1962 Stuart Dreyfus developed a simpler version based solely on the chain rule, and in the 1970s, it was developed to use errors in training deep learning models when it became popular after Seppo Linnainmaa wrote his master’s thesis, including code for implementing FORTRAN for it. However back propagation became useful until 1986, when Hinton and Rumelhart, Williams demonstrated back propagation in a neural network which could provide interesting distribution representations. [25] In 1989, at Bell Labs, Yann LeCun provided the first practical demonstration of back propagation. He combined convolutional neural networks with back propagation to read “handwritten” digits. This system was finally used to read the numbers of handwritten checks.

The overestimation of the potential of AI by some overly optimistic individuals in the period of 1985 to 90s or the second winter for artificial intelligence led to breaking expectations and angering investors. Fortunately, some people have continued to work on AI, and they have made some notable progress. In 1995, Dana Cortes and Vladimir Vapnik developed the support vector machine (SVM). Sepp Hochreiter and Jürgen Schmidhuber also developed the LSTM (Long short-term Memory) for recurrent neural networks in 1997.

In 1999, the development of computers contributed to an increase in their speed, and with the development of GPUs, the speed of data processing increased, making neural networks in competition with SVMs. A neural network provided better results using the same data and also has the advantage of continuing to improve as more training data is added.

In 2009, an AI professor at Stanford named Fei-Fei Li, launched ImageNet, a free database of more than 14 million labeled images. Also in 2012, Google Brain released

the results of project known as “The Cat Experiment” which demonstrated difficulties of “unsupervised learning.” And in 2014 the Generative Adversarial Neural Network (GAN) was created by Ian Goodfellow. [26] [24]

### II.5. Artificial Neural Networks

As cited before, the base of deep learning lies in artificial neural networks ANN. These neural networks with many layers (deep) can simulate (to some extent) the behavior of the human brain during the learning process, making it able to learn, identify patterns and classify different information by being exposed to various examples. Its structure is inspired by the biological neurons, the basic unit of the nervous system. So to understand how deep learning works, we have to know how brain neurons work. <https://towardsdatascience.com/what-is-deep-learning-and-how-does-it-work-2ce44bb692ac>

#### II.5.1. Biological neurons

A neuron cell is an electrically excitable cell that includes of four major regions: a cell body, dendrites, an axon, and synaptic terminals. The cell body receives signals that can be excitatory or inhibitory through dendrites where they are integrated and transported across the axon to the synaptic terminals, which forms connections either with the dendrites or the cell body of another neuron. When the sum of these signals exceeds a certain threshold, the neuron will be activated and produces all-or-none electrochemical pulse, called the action potential. [27]

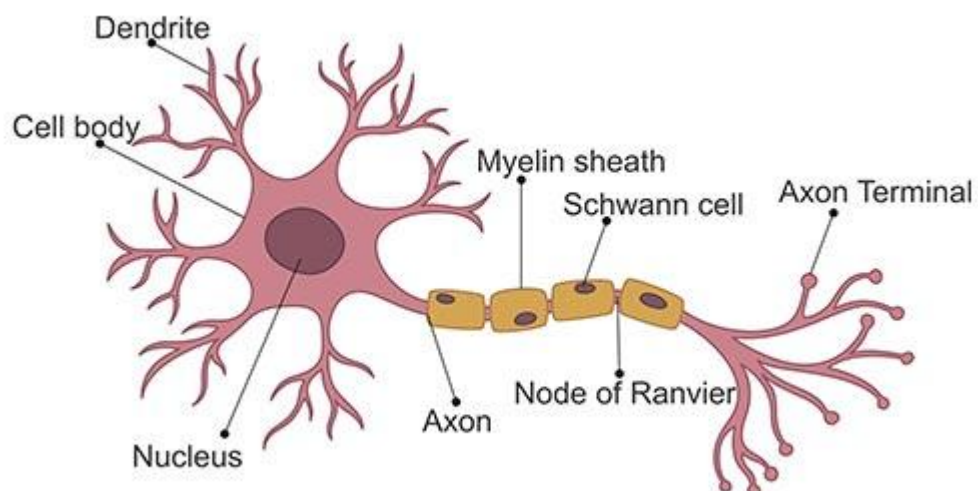


Figure II- 2: a model of a biological neuron [27].

### II.5.2. The Perceptron

As mentioned earlier, Perceptron is a mathematical binary classification model capable of linearly separate between two classes of points. It takes weighted inputs (features) and a bias, performs a dot product between the inputs and its weights and passes the result to a nonlinear function or activation function to produce the outputs.

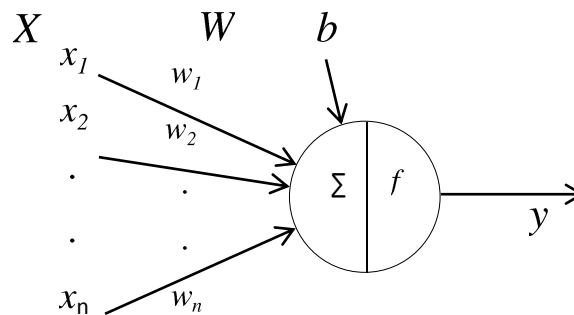


Figure II- 3: an example of a perceptron

This gives the following equation:

$$y = f\left(\sum_{i=0}^n w_i x_i + b\right) \dots\dots\dots \text{Eq II- 1}$$

In addition it has a learning algorithm that allows finding the values of the weights  $w$  in order to get the desired  $y$  output. To develop this algorithm, Frank Rosenblatt was inspired by Hebb's theory which suggests that when two biological neurons are jointly excited, then they strengthen their synaptic links which means strengthen the connections between them. In neuroscience, this is called synaptic plasticity, and this allows our brain to build memory, learn new things, and make new associations. So from this idea, Frank Rosenblatt developed a learning algorithm, which consists in training an artificial neuron on train data  $(X, y)$  so that it reinforces its weights  $w$  each time an input  $X$  is activated at the same time as the output  $y$  present in these data. To do this, he proposed the following formula:

$$W = W + \alpha(y_{true} - y)X \dots\dots\dots \text{Eq II- 2}$$

Where  $y_{true}$  is the desired outputs,  $y$  is the outputs produced by the model,  $\alpha$  is the learning rate,  $W$  is the weights and  $X$  is the inputs. [28]

### II.5.3. The multilayer perceptron

The fact that a lot of problems in real life are not linear makes a single perceptron is not very useful. However according to the idea of McCulloch and Pitts, that by connecting several neurons together, it is possible to solve more complex problems than one neuron does.

In the 1980s, Geoffrey Hinton developed the multilayer Perceptron, the first and one of the most famous artificial neural networks. Its structure is inspired by biological neural networks, where neurons are connected in layers (fully connected dense layers) and neurons in the same layer cannot be connected.

It contains input layer, output layer where there is one neuron for each input, output respectively, and the layers between them are called hidden layers with any number of neurons for each.

The outputs of the neurons of one layer become the inputs of the neurons of the next layer. These networks are called feed-forward neural networks, in which the inputs are forward passed through all the layers to produce an output. This technique is called forward propagating. In the first layer, simple decisions are made based on the input, and it becomes more complex with each layer based on the decisions adopted in the previous layer, and as the network is deep, more accurate decisions appear.

To train the model, the weights  $w_i$  must be adjusted and to do that a technique called Back Propagation is used.

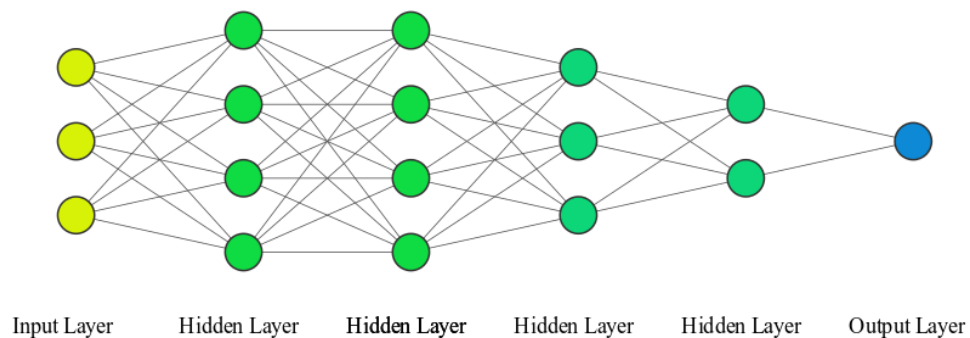


Figure II- 4: an example of MLP

### II.5.4. Activation function

Activation functions are an essential sub-component of neural networks. It is a nonlinear function used in every neuron of the network to decide whether it should be fired or not, and maps its output between 0 and 1 (or -1 and 1 depending upon the function). It helps the network learn the complex patterns in the data and make predictions by introducing nonlinearities into the neural network.

Activation functions have some properties the most important are nonlinearity, differentiability, continuous, bounded and zero-centering. Widely used activation functions face two major challenges: the vanishing gradient problem, which happens when the values of the gradient get closer to zero as the back propagation goes deeper into the network, making the weights change hardly. As a result, the loss stops decreasing, which affects the learning process. And the second problem is the dead neuron problem, which appears when the value of activation function is close to zero, as a result it forces the neurons to be inactive, making them not contribute to the final output. Moreover, the weights may be updated in such a way that the weighted sum of a big part of the network is forced to zero, forcing a large portion of the input to be deactivated. [29] [30] [31]

Below are some of the most frequent activation functions:

- **Sigmoid:**

Sigmoid or logistic function is a non-linear function ranges from 0 to 1 and takes an S shape. Since it ranges from 0 to 1 it is used in probability prediction problems, and its formula is the following:

$$f(x) = \frac{1}{1 + e^{-x}} \dots\dots\dots\text{Eq II- 3}$$

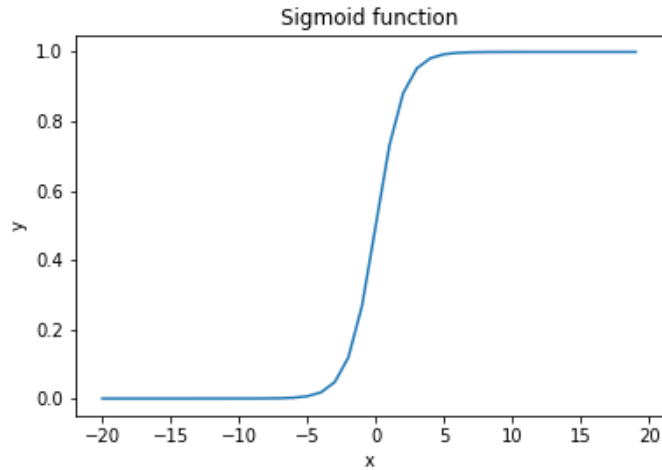


Figure II- 5: Sigmoid function graph

- **Tanh:**

The hyperbolic tangent function is a zero- centered function ranges from -1 to 1, and its formula is the following:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \dots\dots\dots \text{Eq II- 4}$$

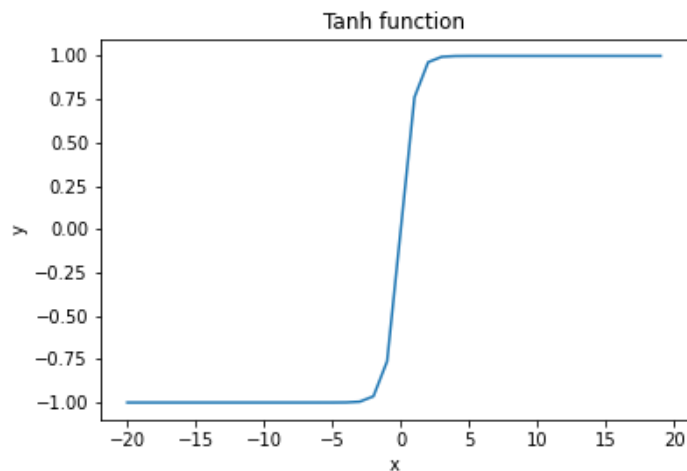


Figure II- 6: Tanh function graph

- **ReLU:**

Rectified Linear Unit or ReLU is one of the simplest activation functions, it outputs zero when the input is negative otherwise it takes the input as an output, it is generally used in the hidden layers. Mathematically it is represented as:

$$f(x) = \max(0, x) \dots\dots\dots \text{Eq II- 5}$$



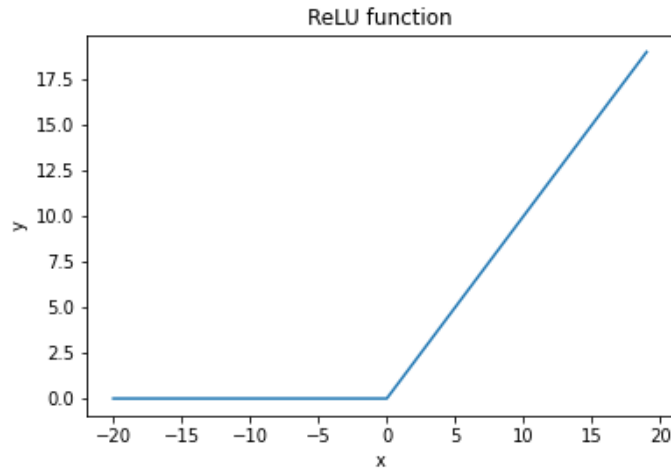


Figure II- 7: ReLU function graph

- **LReLU:**

Leaky Rectified Linear Unit or LReLU is an enhanced version of ReLU for solving the dead neuron problem using a small slope in the negative side of ReLU graph, mathematically it is represented as:

$$f(x) = \max(0.01x, x) \dots\dots\dots \text{Eq II- 6}$$

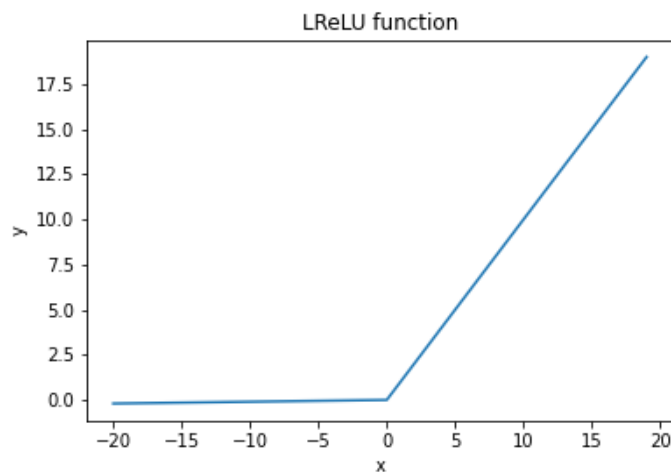


Figure II- 8: LReLU function graph

- **PReLU:**

Parametric ReLU like LReLU it is another version of ReLU that aims to solve the dead neuron problem. It is used when LReLU can't solve this problem, but in this function the slope is provided as a parameter *a*. Its formula is the following:

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ ax & \text{if } x < 0 \end{cases} \dots\dots\dots \text{Eq II- 7}$$

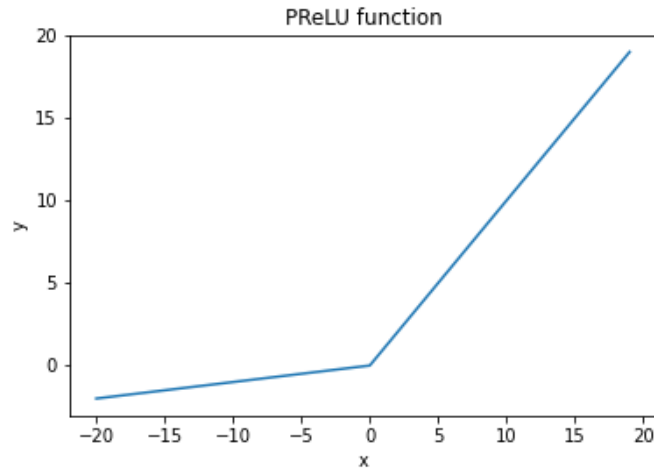


Figure II- 9: PReLU function graph ( $\alpha = 0.1$ )

- **ELU**

Exponential Linear Unit is another variant of ReLU that use a log function in the negative side of ReLU graph instead of the linear function like used in LReLU and PReLU. Its formula is the following:

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha(e^x - 1) & \text{if } x < 0 \end{cases} \dots\dots\dots \text{Eq II- 8}$$

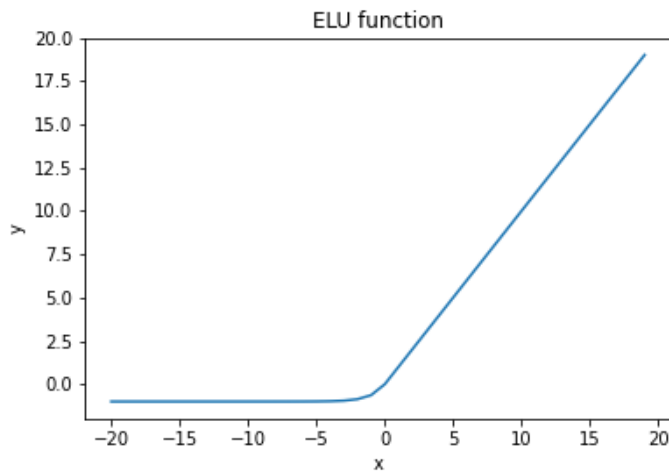


Figure II- 10: ELU function graph ( $\alpha = 1$ )

- **Softmax:**

As mentioned before, sigmoid function is used to deal with probability prediction. However, it can handle binary classification where there are just two probabilities, but how to deal with multi-class classification problem where there are more than two classes and two probabilities. In this case, Softmax function is used, like sigmoid

## CHAPTRE. II: DEEP LEARNING

function it ranges from 0 to 1 but it calculates the probability of each class. It takes a vector  $v = (v_0, v_1, \dots, v_n)$  and returns another vector of probabilities. Its formula is:

$$\text{Softmax}(v_i) = \frac{e^{v_i}}{\sum_{j=0}^n e^{v_j}} \dots\dots\dots \text{Eq II- 9}$$

### II.5.5. Loss function

The loss function is used to measure the performance of a neural network by measuring the distance between its outputs or predictions based on the provided data, and the desired outputs. By using this distance as a feedback signal, the weights and biases can be adjust to minimize this distance or the error between predicted output and desired output and make more accurate predictions. Loss function  $L(w)$  computes the error for one training sample, while the mean of all losses measured for all the training samples constitutes the cost function  $C(w)$  used in the learning process. [5]

Loss functions are classified into two categories based on the problem, where in regression problems regression losses are used, and in classification problems classification losses are used.

For regression losses the most frequent cost functions are:

- **Mean Squared Error:**

MSE or Mean Squared Error is the mean of sum of squared differences between desired and predicted output vectors ( $\vec{y}_{true}$  and  $\vec{y}$  respectively).

$$MSE = \frac{1}{n} \sum_{i=0}^n (y_i - y_{true_i})^2 \dots\dots\dots \text{Eq II- 10}$$

- **Mean Absolute Error:**

MAE or Mean Absolute Error is the average of sum of absolute differences between desired and predicted output vectors ( $\vec{y}_{true}$  and  $\vec{y}$  respectively).

$$MAE = \frac{1}{n} \sum_{i=0}^n |y_i - y_{true_i}| \dots\dots\dots \text{Eq II- 11}$$

- **Mean Bias Error:**

MBE or Mean Absolute Error is same as MSE but without taking the squared differences.

$$MBE = \frac{1}{n} \sum_{i=0}^n (y_i - y_{true_i}) \dots\dots\dots \text{Eq II- 12}$$

And for Classification losses there are:

## CHAPTRE. II: DEEP LEARNING

- **Cross Entropy Loss:**

In classification problems the neural network predicts the probabilities of how match an input belongs to each class, and to measure the error between a predicted probability and the desired class label the cross entropy loss function is used.

For binary classification, it is called binary cross entropy loss, and it is calculated as:

$$y_{true} \ln(y) + (1 - y_{true}) \ln(1 - y) \dots\dots\dots \text{Eq II- 13}$$

The formula of the cost function is:

$$-\frac{1}{n} \sum_{i=0}^n y_{true_i} \ln(y_i) + (1 - y_{true_i}) \ln(1 - y_i) \dots\dots\dots \text{Eq II- 14}$$

And for multiclass classification cross entropy loss is calculated for each class label and returns the sum as result. if the labels are provided in a one hot representation it is called categorical cross entropy and if the labels are provided as integers it is called sparse categorical cross entropy. Its formula is the following:

$$-\sum_{i=0}^n y_i \ln(y_i) \dots\dots\dots \text{Eq II- 15}$$

The formula of the cost function is:

$$-\frac{1}{n} \sum_{j=0}^m \sum_{i=0}^n y_{ji} \ln(y_{ji}) \dots\dots\dots \text{Eq II- 16}$$

[32] [33]

### II.5.6. Gradient descent:

As mentioned before, to make a neural network perform better it must minimize the cost function by updating its weights and biases. The way to do this is the gradient descent algorithm.

Gradient descent is an iterative optimization algorithm that aims to find a local (global) minimum of a function  $f(x)$ . It uses the derivative of this function  $f'(x)$  to determine how to change  $x$  to make a small improvement in  $f(x)$  which make it possible to reduce  $f(x)$  by moving  $x$  in small steps with opposite sign of  $f'(x)$ . [34]

Since the cost function value depends on the difference between predicted and desired output vectors  $\vec{y}$  and  $\vec{y}_{true}$  respectively, and since the prediction vector itself is a function of weights and biases, it makes the cost also a function of weights and biases [35].

## CHAPTRE. II: DEEP LEARNING

In deep learning, gradient descent algorithm is used while training the neural network. First, the parameters of the neural network  $p$  (weights and biases) are initialized to random values, then the algorithm updates these parameters in the opposite direction of the cost function gradient with respect to this parameters, and to fix the size of the steps taken to reach the minimum (local or global) a small constant  $\alpha$  called the learning rate is used. Choosing an appropriate value for the learning rate is important because when it is too big the minimum might be missed and if it is too small the convergence to the minimum will take a long time so it must be neither too big nor too small. [36]

$$p = p - \alpha \nabla_p C(p) \dots\dots\dots \text{Eq II- 17}$$

### II.5.7. Variants of gradient Descent:

There are three variants of gradient descent algorithm that differ in the amount data used to adjust the parameters.

#### 1. Batch Gradient Descent:

In this type of gradient descent the parameters get updated after all the samples in the training dataset are forward passed into the network to calculate the loss for each sample and calculate the cost in the end of the epochs, and this processes are repeated for each iteration of the algorithm. For big datasets batch gradient descent can be very slow and computationally expensive.

#### 2. Stochastic Gradient Descent:

In this type of gradient descent the parameters get adjusted after one samples randomly picked is forward passed into the network to calculate the error for each iteration, this can make it faster than batch gradient descent due the frequent adjustment but it causes noisy gradient.

#### 3. Mini Batch gradient descent:

It is a combination of stochastic and batch gradient descent but instead of using one randomly picked training sample or using all the dataset it uses a batch of randomly picked samples per iteration. [36] [37]

### II.5.8. Adaptive Learning Rate Optimization techniques for Gradient Descent

There are various optimization methods are used for the gradient descent algorithm, some of them are discussed below:

- **Momentum method:**

Since stochastic gradient descent causes noisy gradient making it takes time to reach the minimum. To accelerate the algorithm, the Momentum method [38] uses a fraction  $\gamma$  (usually set to 0.9) from the past to the current update vector. The Momentum update formula is represented in the following equation [39]:

$$v_t = \gamma v_{t-1} + \nabla_p C(p) \dots\dots\dots \text{Eq II- 18}$$

$$p = p - v_t$$

- **Nesterov accelerated gradient:**

Momentum [38] method does not take into account which direction it is going, to solve this problem Nesterov accelerated gradient (NAG) [40] calculate the gradient with respect to the approximate future position of the parameters using the following formula [39]:

$$v_t = \gamma v_{t-1} + \nabla_p C(p - \gamma v_{t-1}) \dots\dots\dots \text{Eq II- 19}$$

$$p = p - v_t$$

- **Adagrad:**

Adaptive subgradient methods for online learning and stochastic optimization or Adagrad [41] improves the stochastic gradient descent algorithm by updating the learning rate for each parameter  $p_i$  at every time step  $t$  based on the past gradients of  $p_i$ , which means the size of these updates depends on the importance of  $p_i$ . To update the learning rate it uses the following formula:

$$p_{t+1,i} = p_{t,i} - \frac{\alpha}{\sqrt{G_{t,ii} + \epsilon}} \nabla_{p_i} C(p_{t,i}) \dots\dots\dots \text{Eq II- 20}$$

## CHAPTRE. II: DEEP LEARNING

Where  $G_{t,ii}$  is a diagonal matrix contains the sum of the squares of the previous gradients with respect to  $p_t$  and to avoid division by zero, the term  $\epsilon$  is used [39].

- **Adadelta:**

Since  $G_{t,ii}$  keeps growing the learning keeps decreasing due the growing of the accumulated sum  $G_{t,ii}$ , this causes problem for Adagrad [41] optimizer. Adadelta [42] is a variant of Adagrad that aims to fix this problem by bounding the range of square gradients accumulated to a certain fixed size [39].

- **RMSprop:**

RMSprop is a method of adaptive learning rate proposed by Geoffrey Hinton, like Adadelta [42] it also deals with the problem of Adagrad method. It divides the learning rate by decreasing average exponential square gradients [39].

- **Adam:**

Adaptive Moment Estimation or Adam [43], it is also an optimizer that calculates adaptive learning rates for each parameter. This method combines the advantages of Adadelta, RMSprop and the Momentum method to update the learning rate [39].

### II.5.9. Backpropagation

Backpropagation is an important part of neural network training and in supervised learning in general to minimize the cost function. It is the tool that gradient descent algorithm uses to calculate the gradient of the cost function with respect to each of the neural network parameters, from the last layer to the first.

The formula of the gradient of the cost function with respect to the parameter going from neuron  $j$  in layer  $i - 1$  to the neuron  $k$  in layer  $i$  is the following:

$$\frac{\partial \mathcal{C}}{\partial p_{jk}^{(i)}} \dots\dots\dots \text{Eq II- 21}$$

To calculate the gradient at a specific layer  $i$ , all the gradients of the next layers (layer  $i + 1$  to the last layer) are combined by using the chain rule of calculus, because

## CHAPTRE. II: DEEP LEARNING

forward propagation can be considered as a long chain of nested functions that depends on each other where the cost function depends on the outputs  $y$  that depends on the activation function  $A$  that in turn depends on the function  $Z$  that takes the inputs  $X$  and its weights  $W$  and return the dot product of it make it depends on  $W$ , so the cost function can be expressed by the following formula.

$$C = C(A(Z(XW))) \quad \dots\dots\dots\text{Eq II- 23}$$

So the formula of the gradient using the chain rule will be like this:

$$\frac{\partial C}{\partial p_{jk}^{(i)}} = \left(\frac{\partial C}{\partial A_k^{(i)}}\right) \left(\frac{\partial A_k^{(i)}}{\partial Z_k^{(i)}}\right) \left(\frac{\partial Z_k^{(i)}}{\partial p_{jk}^{(i)}}\right) \quad \dots\dots\dots\text{Eq II- 24}$$

Computing the gradient of the parameters individually using the chain rule can be an expensive task however backpropagation algorithm avoids duplicate calculations. To calculate the gradient in the current layer it uses the last gradient calculation got with the chain rule to avoid recalculating all the expression, and repeat this operation backward from the last to the first layer, this makes the backpropagation algorithm more efficient than calculating gradient individually for all the parameters [44] [45] [25] [46].

### II.6. Deep neural network:

In recent years, deep learning methods using networks of neurons have shown impressive results in multiple areas, such as speech recognition, natural language processing or computer vision. The word “Deep” in the term “deep neural networks “ refers to a network that has multiple hidden layers, and the choice of its number depends on the problem and the dataset’s size, where from these layers deep neural networks derive its efficiency to automatically learn complex patterns from input data.

However deep neural networks are computational expensive and need a huge amount of data, fortunately the availability of large-scale datasets and the invention of the graphics processing units that increase the computing power makes deep learning methods a tool to solve many problems in various domains. [47] [48]

Deep neural networks have varied architectures that used in various applications and the most popular architectures are:



- **Convolutional Neural Networks (CNNs):**

That is used in computer vision such as image processing and recognition, video recognition, and even natural language processing. It will be discussed in more detail in the next section.

- **Recurrent Neural Networks (RNNs):**

That used in many applications of speech and handwriting recognition. It has connections that feed back into the previous layers or in the same layer rather than the feed-forward connections, allowing it to maintain memory of previous inputs to process sequential data. [49]

### II.7. Convolutional Neural Networks

Convolutional neural networks (CNNs) are feed forward neural networks inspired by the brain's visual cortex and used to process structured data arrays like images by performing an operation called convolution on the input to generate a features map. They are commonly used in computer vision applications. However, they can be found in other domains, such as natural language processing for text classification. [50]

Unlike the old computer vision algorithms, convolutional neural networks don't need to preprocess the inputs to extract features. It can work directly with raw images and detect the different patterns or features in it, such as edges, lines, shapes or even more complex ones like eyes and faces. This property makes the convolutional network appropriate for computer vision.

As with any other artificial neural network a convolutional neural network includes an input layer, hidden layers, and output layers. These networks have special layers called convolutional layers, and each convolutional layer is followed by another type of layers called pooling layers. A convolutional neural network can have many convolution and pooling layers on top of each other, where in the first convolution layer the network learn simple visual patterns in the second layer it will learn more complex patterns based on the patterns of the first layer and so on.

## CHAPTRE. II: DEEP LEARNING

These two types of layers form the features extraction part of the network. In other hand, another part of the network contains a fully connected layers handles the classification process.

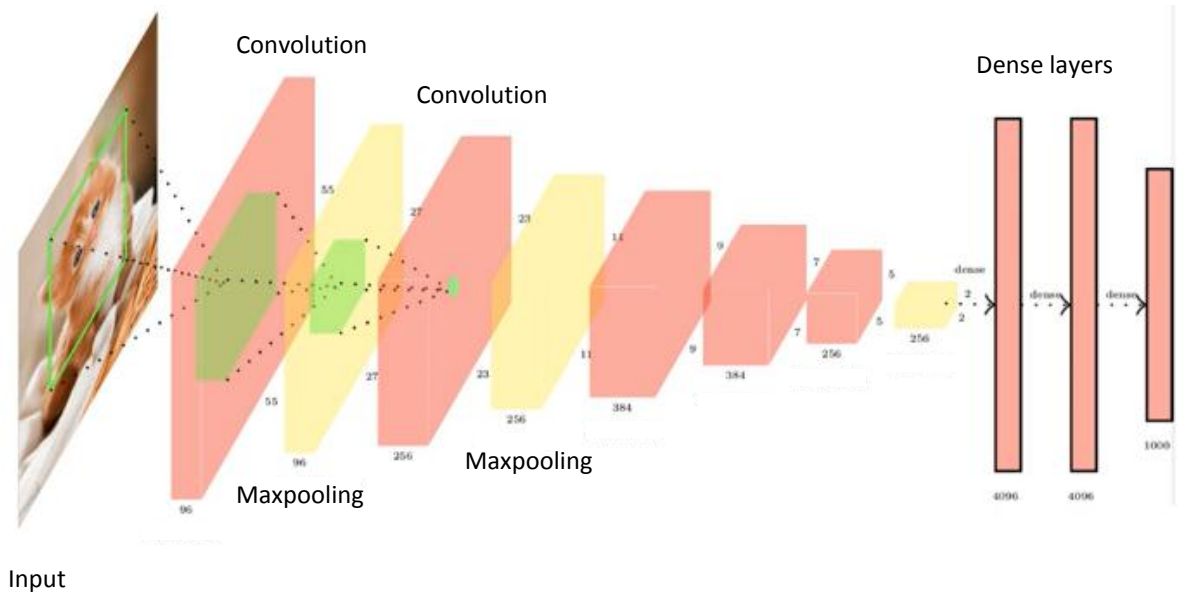


Figure II- 11: example of convolutional neural network [51]

### II.7.1. Convolutional layers

As mentioned before, convolutional neural networks deal with images as inputs. An image is considered as a three-dimensional array of pixels where each entry takes a value between 0 and 255 to represent the brightness intensity. The dimensions of this image represent its width, height, and a depth, which represent the number of channels in the image, where black and white images have one channel representing the gray scale, color images have three channels one for each color (RGB).

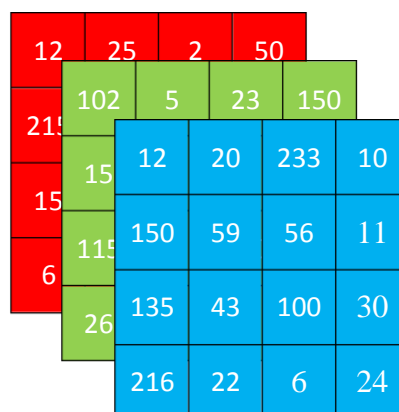


Figure II- 12: example of RGB image with (4×4×3) in size

## CHAPTRE. II: DEEP LEARNING

In the convolutional layers, the network performs a convolution operation between the input image  $I$  and the convolutional filter or kernel  $K$ . The kernel is learnable three-dimensional array of weights with small dimensions (width, height, and depth). The depth here represents the number of channel in the input image.

The convolution operation is a simple dot product between the kernel and a small region of the image that has a size determined by the kernel's width and height called the receptive field. This dot product is fed into an output array then the kernel slides by a stride along the image's width and height to convolve the entire image and produce a feature map.

The stride is the number of pixels that the filter slides over the input image at a time, generally it takes the value of one and a larger stride returns a smaller feature map where for an input image  $I(Ix \times Iy \times d)$  and a kernel  $K(Kx \times Ky \times d)$  and a stride  $s$  the size of the convolution output  $O$  will shrink by the following factor:

$$O\left(\frac{Ix - Kx}{s} + 1 \times \frac{Iy - Ky}{s} + 1 \times d\right) \dots\dots\dots\text{Eq II- 25}$$

Since the network performs multiple convolution operations in cascade, there is a risk of getting an image that is too small and lose all the information. Another problem is that the stride might not fit the image. This prevents to get a full convolution and leads to loss of information in the edges.

To solve these problems, a technique called zero padding is used on the image by adding zero pixels around its edges. Zero padding also makes corner pixels more contributory to feature detection. It has three types:

- **Same padding:** this type preserves the same output size as the input.
- **Full padding:** this type increases the size of the output by adding zeros to the input borders. [52]
- **Valid padding:** this type is considered as no padding.

So using  $p$  as number of padding the size of the convolution output  $O$  will be represented with this formula:

$$O\left(\frac{Ix - Kx + 2p}{s} + 1 \times \frac{Iy - Ky + 2p}{s} + 1 \times d\right) \dots\dots\dots\text{Eq II- 26}$$

## CHAPTRE. II: DEEP LEARNING

After each convolution operation, the network passes the result through a ReLU function to produce the feature map and it is possible to convolve one input image with multiple filters. [53] [54]

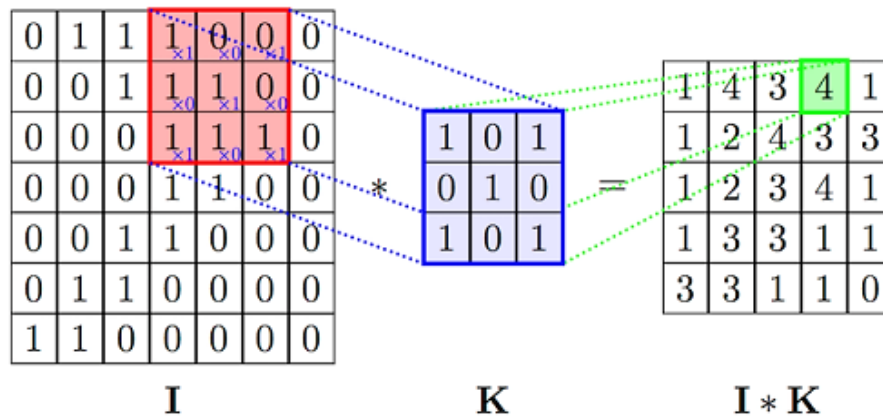


Figure II- 13: example of a convolution operation

As shown in the figure above, the feature map ( $I * K$ ) doesn't map directly to each pixel value in the image  $I$ . Instead, each entry in the feature map (ex: 4) connects to the receptive field only (the red square in the image  $I$ ), which helps to reduce the number of weights. This makes the convolutional layers be referred to as partially connected layers where each filter learns local patterns or features that are translation invariant [34]. It also has the property of parameter sharing, since the kernel remains the same with the entire image during the convolution operation. [5]. [55]

### II.7.2. Pooling layers

After each convolution layer, the convolutional neural network applied another operation on the feature map in order to reduce its dimensions. This reduction is done in a specific layer called pooling or down sampling layer. Like a convolutional layer, it has a kernel that slides over the feature map with a stride, but here the kernel has no weights. Instead, it applies a dot product between the kernel and a small region of the feature map. The kernel applies a function to the values in the target region or the receptive field to be specific, and this function determines the type of pooling where there are two types of pooling:

- **Average pooling:** while the kernel is sliding along the feature map, it calculates the average value of the receptive field, and this average value is fed into an output array.

## CHAPTRE. II: DEEP LEARNING

- **Max pooling:** while the kernel is sliding along the feature map it returns the max value of the receptive field and feed it into an output array. This approach is commonly used more than average pooling.

The pooling operation helps to reduce the complexity of the network and prevent over fitting. Note that the pooling operation can be applied on the entire feature map to reduce each channel in it to one value, either the max or the average of the feature map. Here the operation is called the global pooling.

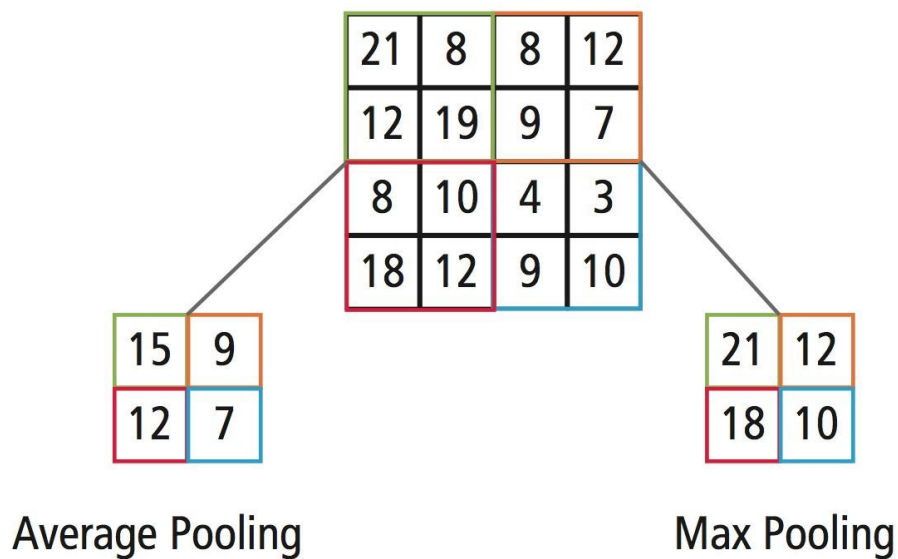


Figure II- 14: Example of an average and max pooling operations with stride length of 2 and a kernel size  $2 \times 2$  [90]

### II.7.3. Fully connected layers

Since the convolution and pooling layers are considered as a feature extractors, a fully connected layers deal with the classification problem where the features are reshaped to a vector by a flatten layer and fed into a multi-layer perceptron to be trained on it, then it passes the output throw softmax to produce predictions. Sometimes global pooling is used in models instead of the flatten layer and in some cases is used to replace the fully connected layer.

### II.7.4. History of CNNs

In the 1950s and 1960s, David Humble and Torsten Wiesel published papers showing that neurons in the visual cortex are restricted to certain parts of the visual field known as receptive field, and the visual processing proceeds in series, from neurons that detect simple patterns to that detect more complex ones. [56]

In 1980, Kunihiro Fukushima invented a type of neural network capable to recognize patterns in the images through learning, inspired by the discoveries of Hubble

## CHAPTRE. II: DEEP LEARNING

and Wiesel, which he called a “neocognitron” [57]. In 1998, Yann LeCun invented LeNet, a convolutional neural network with five layers, where he applied backpropagation to train the network for handwritten zip code recognition. [44]

After AlexNet won the ImageNet Image Recognition competition in 2012, the convolutional neural networks became the first choice and the standard in computer vision applications. [47]

### II.7.5. The famous state-of-the-art CNNs

- **LeNet:** developed by Yann LeCun in 1998. It was the first successful convolutional neural network that was trained by backpropagation to recognize handwritten zip code digits.
- **AlexNet:** developed by Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton [47]. It was the first work that marked the role of the convolutional neural networks in computer vision applications. In 2012 it won the ILSVRC. It had a very similar architecture to LeNet, but it was deeper where many convolutional layers were stacked on top of each other.
- **GoogLeNet.** Developed by Szegedy et al [58] from Google. In 2014 it won the ILSVRC.
- **VGGNet:** Developed by Karen Simonyan and Andrew Zisserman in 2014 [59]. It showed that the depth of the network is important for the model performance. It supports 16 and 19 convolution layers for its versions VGG16 and VGG19 respectively.
- **ResNet:** Residual Network developed by Kaiming He et al [60]. It was the winner of ILSVRC 2015 with an error of only 3.57%.

### II.8. Transfer Learning

The large number of data used in training deep learning models is a key factor in their effectiveness and accuracy of performance, as training data is like fuel for them.

For some problems, the necessary data for training models may not be available, perhaps because the problem is new and not enough data has been collected yet, or the problem is very rare, which makes the process of collecting the dataset very expensive.

## CHAPTRE. II: DEEP LEARNING

Fortunately, the idea of reusing pre-trained models to solve new problems or what is called transfer learning helped reduce the amount of data needed for training. Transfer learning is the process of taking a pre-trained model on a large training dataset like ImageNet and using its parameters to learn on the training data of the problem to be solved.

The idea with Transfer Learning is that this pre-trained model will act as a feature extractor. Then it is adjusted by removing the network classifier (i.e. the fully connected layers) and replacing it with a new classifier depending on the problem. After that, the model parameters of all other layers that it learned earlier are frozen to use it in training the rest of the network instead of training the entire network with random initialization of parameters, this adjustments as fine tuning. The number of frozen layers depends on how similar the problem that the model has previously trained on and the new problem are, where if they are very different it would be better to freeze just a few of the first layers that detect basics features (edges and curves...). [61] [62] [63]

### II.9. Explainability in convolutional neural networks

As mentioned previously, convolutional neural networks play a prominent role in computer vision applications, as they have moved this field to a level that almost rivals human ability to some extent due to the accuracy of its results and in a short time. The reason for the superiority of these networks over the rest of computer vision algorithms is due to their unique structure, which, as mentioned previously, simulates the structure of the human visual system. This gave it the ability to extract features regardless of where they are located in the image, in addition to deducing relationships between image pixels, as well as to elicit relationships between image pixels using convolutional layers.

However, there is a trade-off between accuracy and explainability where the simpler the model is, like classic machine learning models, the more interpretable it is. Although convolutional networks provide accurate performance results in various computer vision applications, their interpretation is difficult and consequently, when the model fails to do something, it is difficult to find the cause, and in some high risk areas like healthcare it is difficult to trust the results of the model to make decisions. This is what makes finding a way to interpret these networks and make them reliable by explaining why they reached what they reached.

### II.9.1. Importance of explainability in a CNN model

The process of interpreting convolutional neural networks is very important, especially in critical areas that require accurate results, for example, a model for detecting cancerous tumors must be accurate in its results with the lowest possible error rate, but how can this model be trusted without explainability and without knowing how did it reached the results it reached, especially in some cases, the model can provide impressive results, but when examining how it reached it, it can be found that it was relied on a wrong basis, perhaps due to a noise in the input like smudge on the scan image or the use of biased data to train the model, which may cause serious consequences in this case it is a matter of life and death for the patient so knowing what the model does and how it makes its decisions about its predictions is very important.

Among the most famous examples that illustrate the role of model interpretation based on convolutional neural networks in understanding their slips and improving their performance is the prevailing legend in the computer vision community about the technical failure that happened with the United States Army [64] [65]. They wanted to take advantage of neural networks to automatically detect camouflaged enemy tanks. Therefore, their researchers assigned to this work collected 200 images for this purpose, 100 images of camouflaged tanks in trees and another 100 images of trees without tanks, and then they trained a neural network on half of the data, i.e. 50 images of camouflaged tanks in trees, and 50 images of trees without tanks, where the model achieved accuracy 100%. Of course, the researchers were happy with this result, but to confirm these results, the researchers tested the model on the remaining 100 images and again the model classified all the images correctly.

After the researchers confirmed the success of their model and that the camouflaged tank detection problem had been successfully solved, the researchers handed over the final work to the Pentagon. But the situation soon changed as after a few weeks, researchers received a call from the Pentagon who expressed their dissatisfaction with the model's terrible performance in the field.

The researchers repeated their experiments on the model over and over again, but to no avail, until they visually examined the data set and then realized that the reason for what happened was that in the data set, images of camouflaged tanks were taken on cloudy days, while images of trees were taken on sunny days. This is what led the



## CHAPTRE. II: DEEP LEARNING

network to bias the cloudy and sunny weather to learn from learning to differentiate between a forest full of camouflaged tanks and an empty forest. Although this remains a legend, it does illustrate the importance of model explainability. [65] [66]

### II.9.2. Preliminary Methods

These methods simply rely on the presentation of the architecture of the model to understand and analyze how it works in general using diagrams and illustrations. Despite this, these schematics require the person to be a specialist to be able to interpret the models, and this method does not help much in the case of deep models where it is not possible to predict how the model will work, making it more like a black box.

Making convolutional neural networks more transparent is by visualizing input parts that that greatly affect predictions. Among the first methods used to make visual explanations for convolutional neural network is visualizing the kernels. The kernels can also be plotted as an image. By visualizing these kernels in each layer, it is possible to understand the work of the kernels in the layer. For example, in the picture below it can be seen that there are kernels to detect edges along different orientations, others to detect corners, others for color based edge detectors and others that detect higher order patterns.

This example is taken from AlexNet kernels visualizing for the first convolutional layer but this is not a characteristic of AlexNet alone, in convolutional neural networks the kernels of the first convolutional layer have almost the same structure where they applied low level image processing (edge, corner, color, blob detectors...).



Figure II- 15: AlexNet first convolutional layer kernels visualization [47]

## CHAPTRE. II: DEEP LEARNING

It is also possible to visualize the kernels of the other higher layers, but due to the fact that these kernels become more complex as the network goes deeper they become difficult to interpret and uninteresting.

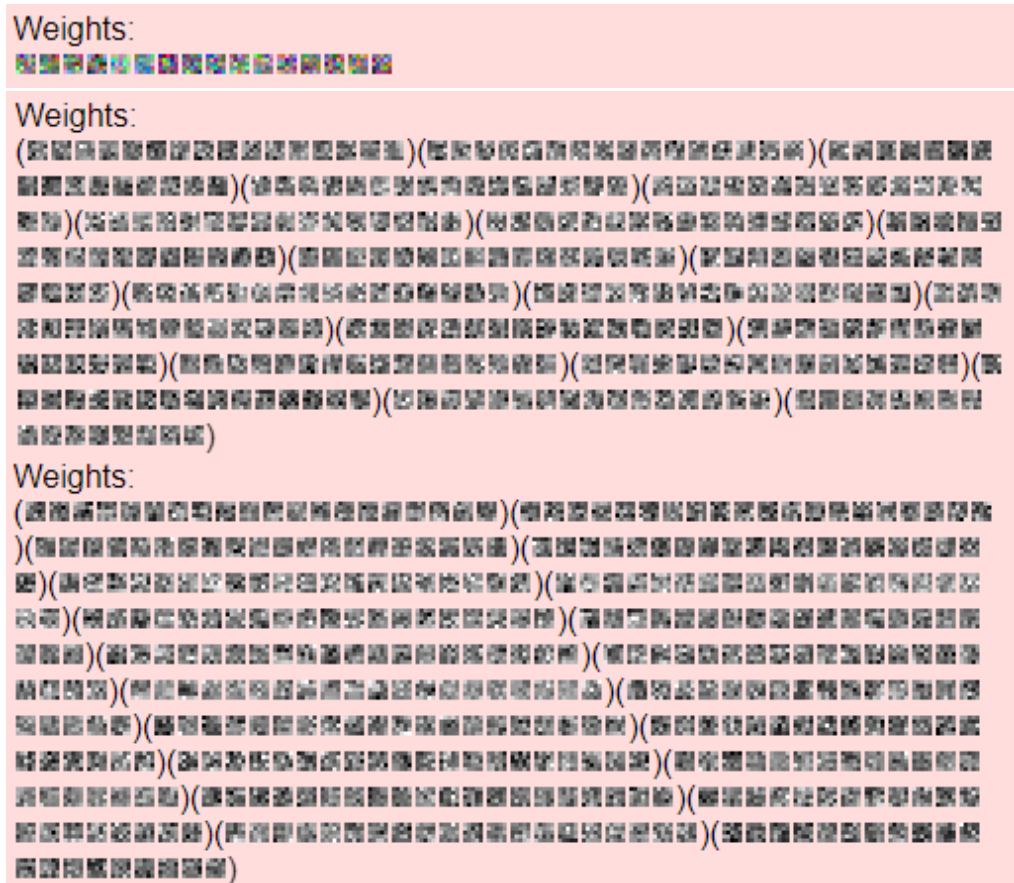


Figure II- 16: higher layers filters visualization [67]

Another method is Instead of visualizing the kernels of the CNN, it is possible to visualize the representation space learned by the CNN by visualizing the feature vector in layer immediately before the classifier after the forward pass of the test dataset. To do this first the dimension must be reduced using dimensionality reduction algorithms such as Principle Component Analysis algorithm (PCA) or t Stochastic Neighborhood Embedding algorithm (t-SNE) [68]



Figure II- 17: example of two dimensional plot of some ImageNet samples where the images are grouped based on their classes for example the dog class in the red box [16]

### II.9.3. Activation based methods

In this methods the visualization is done by visualize the feature maps after applying the kernels over the input image, this method helps to understand where and which input pattern activate which kernel. Also it is possible to choose a kernel in the intermediate convolutional layers to be monitored, forward propagate different input images to the network and visualize which the most image that makes this kernel activated by going backwards through the network until reaching the receptive field that corresponds to this kernel in the input image. This method helps to visualize image patches (the receptive field) that correspond to maximal activations.

Another visualization method that based on kernels activation is saliency via occlusion that aims to visualize the pixels responsible for maximizing the probability of belonging to each class. It helps to understand where the CNN looked (the location of the object)to make its prediction in an image classification problem, for example in a dog image it would be useful to know if the model looked to the dog to predict the class dog or it looked to something else. This method occludes different patches in the image (column “a” in figure I-17 below) and observes the effect on predicted probability of the correct class image (column “d” in figure I-17 below).

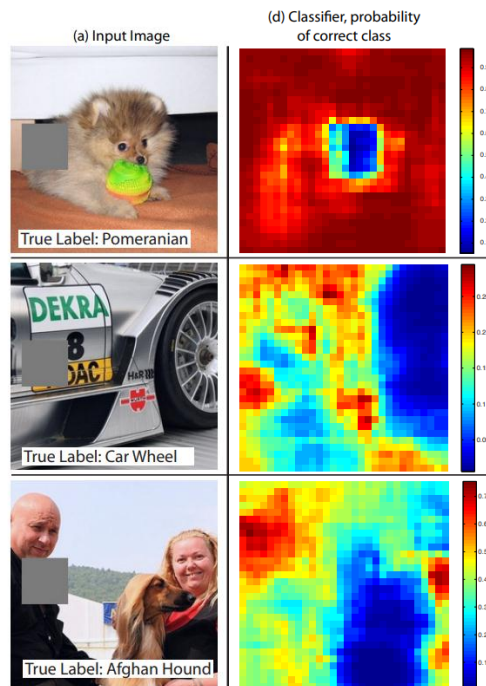


Figure II- 18: example of saliency via occlusion method [69]

### II.9.4. Gradient based methods

As its name indicates, these methods rely on gradient calculation. The simplest method is visualizing the data gradient where since the gradient of an image has three channels the gradient is obtained by taking the absolute value of it in each channel then picking the max gradient in the three channels. This gradient gives an indication of which part (pixels) of the image is the responsible of maximize a particular probability output.

## CHAPTRE. II: DEEP LEARNING

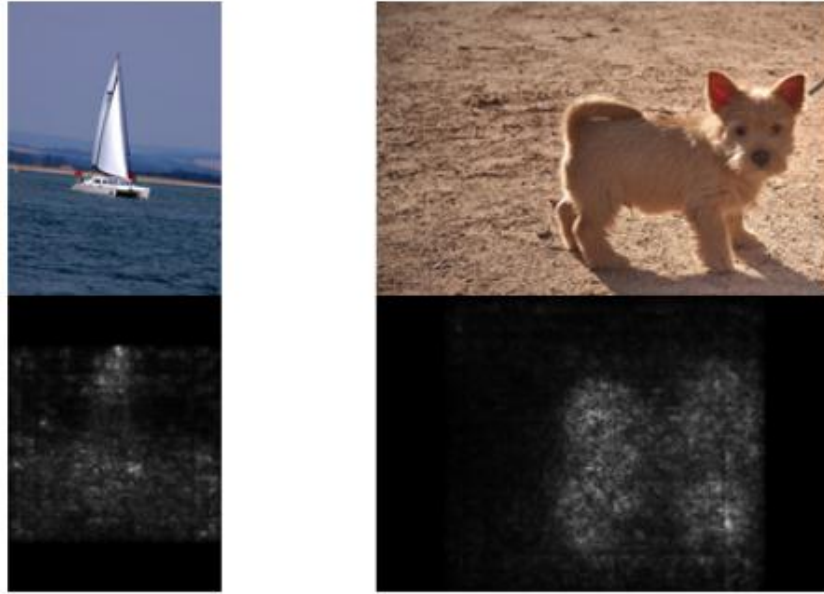


Figure II- 19: example of saliency maps [70]

Another method called guided backpropagation is used to get cleaner visualizations by making some changing in the backpropagation operation. This method is a combination of traditional gradient base visualization and the “decovent” [71] [70].

The difference between “decovent” and the normal backpropagation is in terms of how ReLUs are treated where instead of passing on gradient through ReLU when input is positive like in normal backpropagation, “decovent” passes on gradient through ReLU when backpropagated gradient is positive.

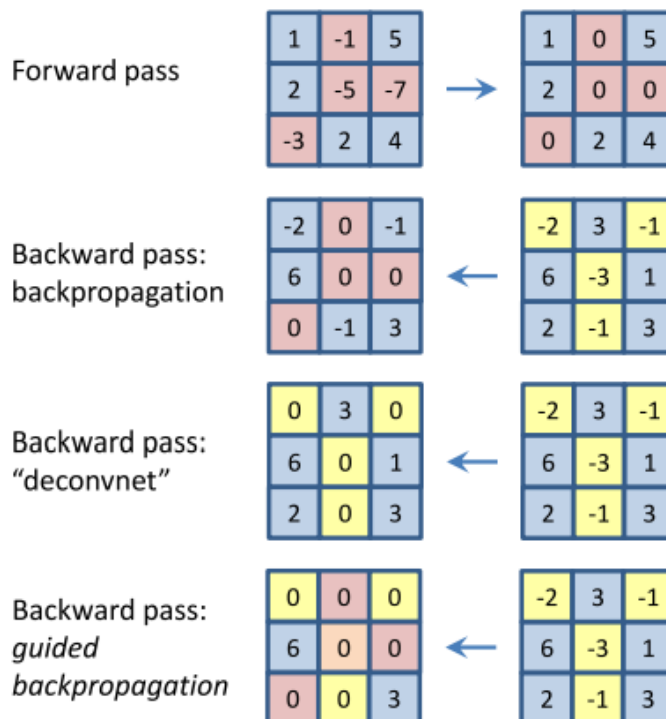


Figure II- 20: Different methods of propagating back through a ReLU nonlinearity [72]

Guided backpropagation takes advantage of both results by passing on gradient through ReLU when both input and backpropagated gradient are positive (figure II-19). This method works by feeding an image into the network then picking a layer and set the gradient there to zero except for the neuron of interest. After the backpropagation it is possible to visualize the data gradient (the figure below).

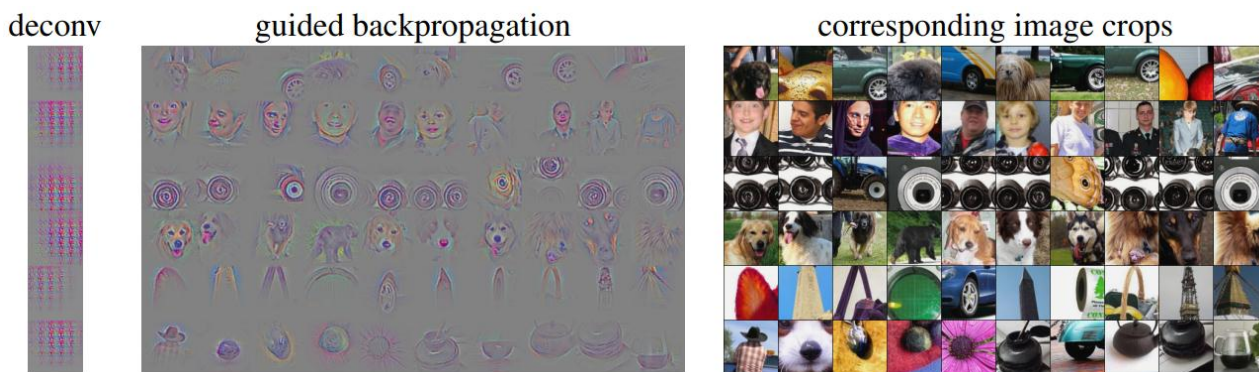


Figure II- 21: Visualization of the data gradient [72]

This method helps to make clear which part of an input image is responsible for a particular prediction with high resolution and it highlights fine grained details in the image however, it is not class discriminative where in figure II-21 [2] the guided

## CHAPTRE. II: DEEP LEARNING

backpropagation is applied on the image and the class of interest is ‘Cat’ but both the dog and the cat are highlighted.

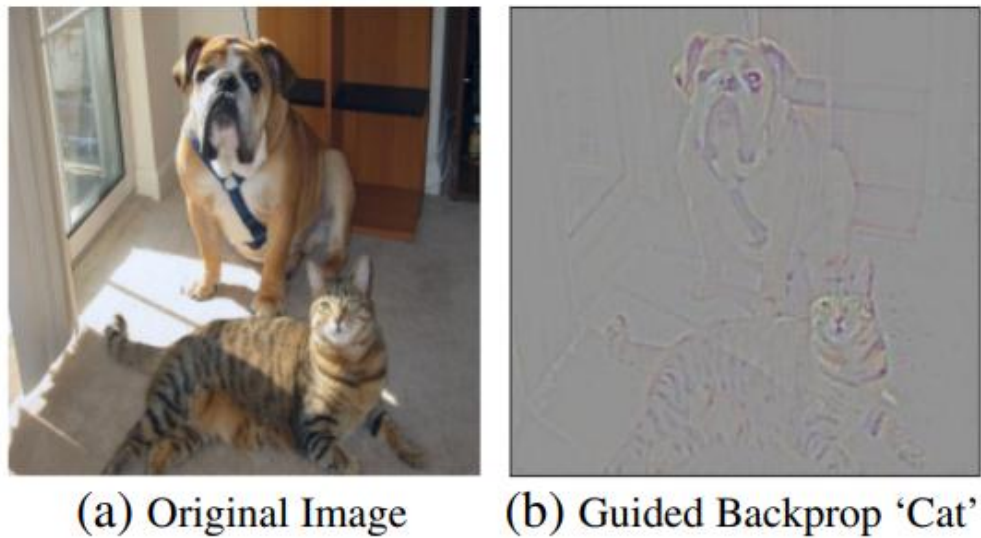


Figure II- 22: guided backpropagation example [73]

One of the earliest class discriminative methods is called Class Activation Maps or CAM [74]. This method uses global average pooling to generate activation map that visualize the discriminative parts of the image used by the network to identify a particular class [74]. It works by taking the last convolutional layer and for each feature map  $f_k(x, y)$  in this layer where  $k$  represent the index of the feature map it applies global average pooling (GAP) to convert each feature map to a scalar  $F^k$  by averaging all its intensity values which means  $F^k = \sum_{x,y} f_k(x, y)$ , and each scalar has a learned weight  $w_k^c$  corresponding to class  $c$  for the  $k$ th feature map that indicates the importance of  $F^k$  for a class  $c$ . Then a weighted sum is performed between the feature maps and their correspondent weights to generate the final class activation map  $M_c(x, y)$  that indicates the important parts of the image for the class  $c$ . Where:

$$M_c(x, y) = \sum_k w_k^c f_k(x, y) \quad \dots\dots\dots\text{Eq II- 27}$$

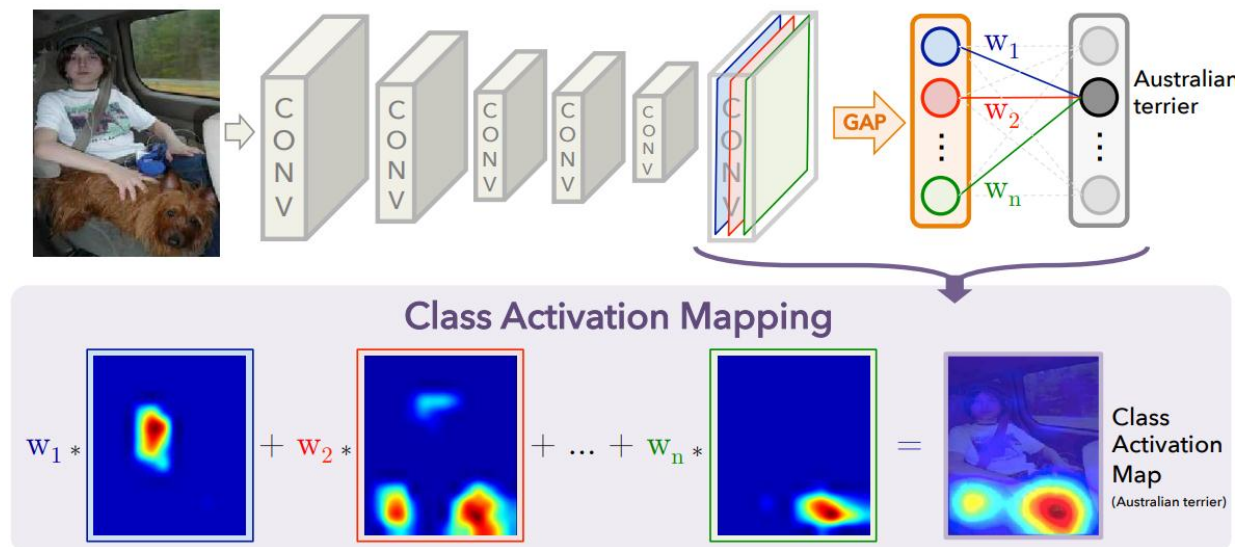


Figure II- 23: CAM overview [74]

From the advantages of CAM is that it is class discriminative and doesn't require a backward pass unlike the previous methods, however it needs to retrain the model to explain it, it imposes on the architecture to be involving a GAP layer instead of fully connected layers to be able to explain models, and if the GAP layers are used alone for classification the absence of the fully connected layers poses a risk of achieving lower accuracy to get a better explainability.

To address these disadvantages another method called Gradient weighted CAM or Grad CAM is used, it is published in 2017 [73]. This method doesn't need any modification on the networks architecture, where it can be applied directly to CNN based architecture and without retraining. It uses the last convolutional layer because of its retention of spatial information, which is lost in the fully connected layers and after the input image is forward passed through the network it calculates the gradient for a class  $c$  with respect to each  $k$  feature map  $A^k$  in the last convolutional. These gradients are global average pooled to obtain the weights  $\alpha_k^c$  where:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial Y^c}{\partial A_{ij}^k} \dots\dots\dots \text{Eq II- 28}$$



## CHAPTRE. II: DEEP LEARNING

Then a weighted sum is performed on the feature maps and its weights and the result is followed by ReLU to show only the positive correlations in the final saliency map.

$$L_{Grad-CAM}^c = ReLU\left(\sum_k \alpha_k^c A^k\right) \quad \dots\dots\dots Eq II- 29$$

Grad CAM is considered as generalization of CAM where the publishers of the Grad CAM paper found that  $w_k^c$  learned in CAM are simply the sum of all the gradients of the class  $c$  score  $Y^c$  with respect to each pixel in the feature map  $A_{ij}^k$  which makes Grad CAM doesn't need to retrain the model where:

$$w_c^k = \sum_i \sum_j \frac{\partial Y^c}{\partial A_{ij}^k} \quad \dots\dots\dots Eq II- 30$$

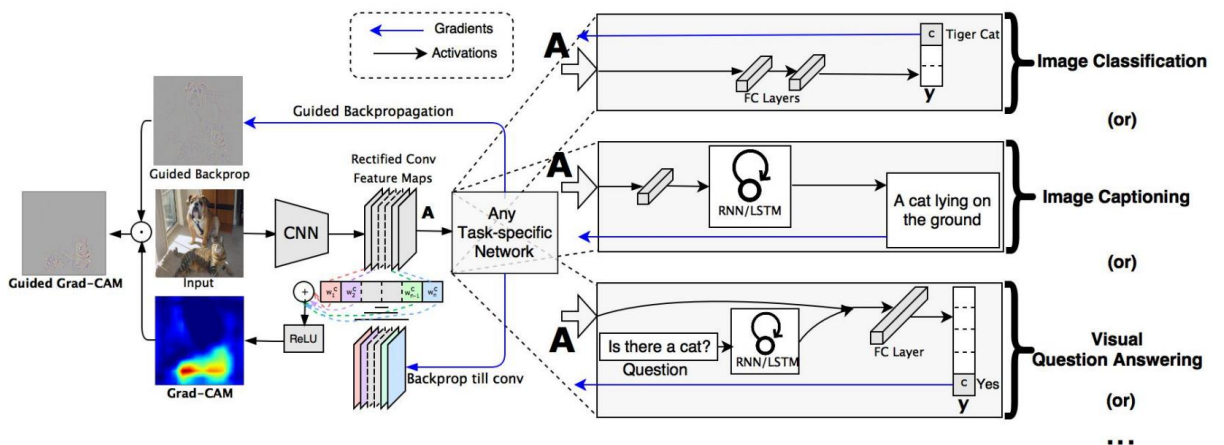


Figure II- 24: Grad CAM overview [73]

Because Grad-CAM lacks the ability to highlight fine detail, it can be combined with guided backpropagation via pixel wise multiplication to produce a high resolution, less noisy and class discriminative visualization. This technic is called guided Grad-CAM [73]

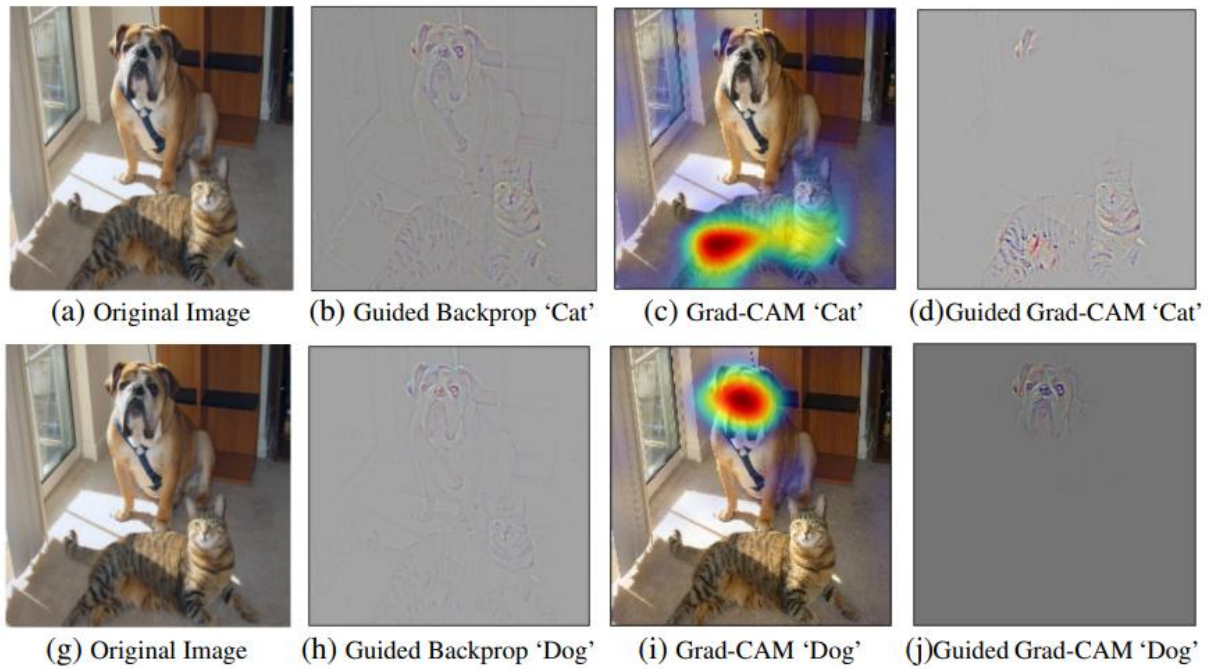


Figure II- 25: example of guided backpropagation, Grad-CAM, and guided Grad-CAM for two classes 'Cat' and 'Dog' [73]

## II.10. Conclusion

In this chapter, we introduced the concepts of deep learning in general, and then we devoted a part to talk about convolutional neural networks, one of the most famous deep learning algorithms, on the basis of which the model that we will talk about in the next chapter was designed. Finally, we concluded the chapter by mentioning the various methods used to construct a visual explanation of models based on convolutional neural networks.

**CHAPTRE. III:  
MODEL DESIGN  
AND  
IMPLEMENTATION**

### III.1. Introduction

During the previous chapters, we discussed the concepts and importance of deep learning in various fields, in addition to its prominent role in computer vision and its applications. This work aims to create a deep learning model based on convolutional neural networks capable of detecting people infected with the Corona virus through chest x-rays and classifying them into four classes: the class of people infected with the virus, those who have viral pneumonia, those who have lung opacity, and the class of normal people.

This chapter presents the tools used in our work and the various stages of this work. It also presents the models used in the work, where four different models of convolutional neural networks were used and many experiments were done on them, and then compared them to choose the best model that provides an acceptable accuracy rate besides the ability to provide a visual explanation that is as relevant as possible for this model to ensure that it bases its decisions on a rationale.

### III.2. Related works

Since the spread of COVID-19 around the world at the beginning of the year 2020, many works have been proposed to detect this virus by classifying X-ray images. Among these works is the work of Panwar, Harsh, et al [75] in 2020, where they proposed a VGG19-based model that provides 95.61% accuracy in detecting COVID-19 cases after studying chest X-Ray and computed tomography Scan images of the chest and performing binary image classification experiments to detect patients those infected and not infected (Normal, Pneumonia, NON-COVID-19 patients) with COVID-19. To make the model more interpretable and interpretable, they applied the color visualization approach using the Grad-CAM method. Another work published by Umair, Muhammad, et al [76] in 2021 where they applied transfer learning technique with fine-tuning to the four pre-trained models VGG16, ResNet-50, MobileNet, and DenseNet-121 and trained them using the dataset (available on Kaggle) of 7232 (COVID-19 and normal image) chest X-ray images in order to detect COVID-19 using these images. They also collected an original dataset of 450 chest X-rays to be used for testing and prediction purposes. The achieved accuracy for VGG16, ResNet-50, MobileNet, and DenseNet-121 was 83.27%, 92.48%, 96.48% and 96.49%, respectively. To generate class- discriminative heatmap images in order to highlight the region where

## CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

the model is given the most attention during feature extraction, they applied the Grad-CAM method. Among the recent works on the subject is the research of Haghanifar, Arman, et al [77] published on April 7, 2022, in which they collected a data set of chest X-ray images from normal lungs and patients with COVID-19. Then they designed and trained a DenseNet based model called COVID-CXNet using weights initially set from the CheXNet [78] model, where it achieved 87.88% as a total accuracy. And to validate the model, they used the GradCAM method. A less recent paper was published on 25 March 2022 by Hemied, Omar S., et al [79] in which they proposed a model to identify and distinguish between pneumonia and COVID-19 from X-ray images using transfer learning based on the pre-trained DenseNet-169 model which it achieved a detection accuracy of 98.824% besides using the GradCAM method to provide a visual check and explanation of model predictions.

### III.3. Dataset used in this work

With the situation deteriorating at the beginning of the year 2020, when the virus called Coronavirus or COVID-19, which first appeared in Wuhan Province, became a global pandemic, it became necessary to find ways to combat this epidemic. Among the proposed solutions was to exploit the superiority of deep learning in the field of computer vision to build a model for classifying chest x-ray images based on whether a person was injured or not. Using deep learning models, examination results can be obtained in a short time compared to traditional detection methods. To build this model, it is necessary to provide enough data to train it.

In this work, a dataset of chest x-ray images of positive cases of COVID-19 in addition to images of normal, lung opacity, and viral pneumonia was used, created by a team consisting of researchers at Qatar University, Doha, Qatar, and Dhaka University, Bangladesh, in addition to their collaborators from Pakistan and Malaysia, in cooperation with doctors [80] [81].

This dataset consists of 3616 images for positive cases of COVID-19 along with 10192 images for normal cases, 6012 images for lung opacity and 1345 images for viral pneumonia. This dataset is available to download on kaggle [82] [83].

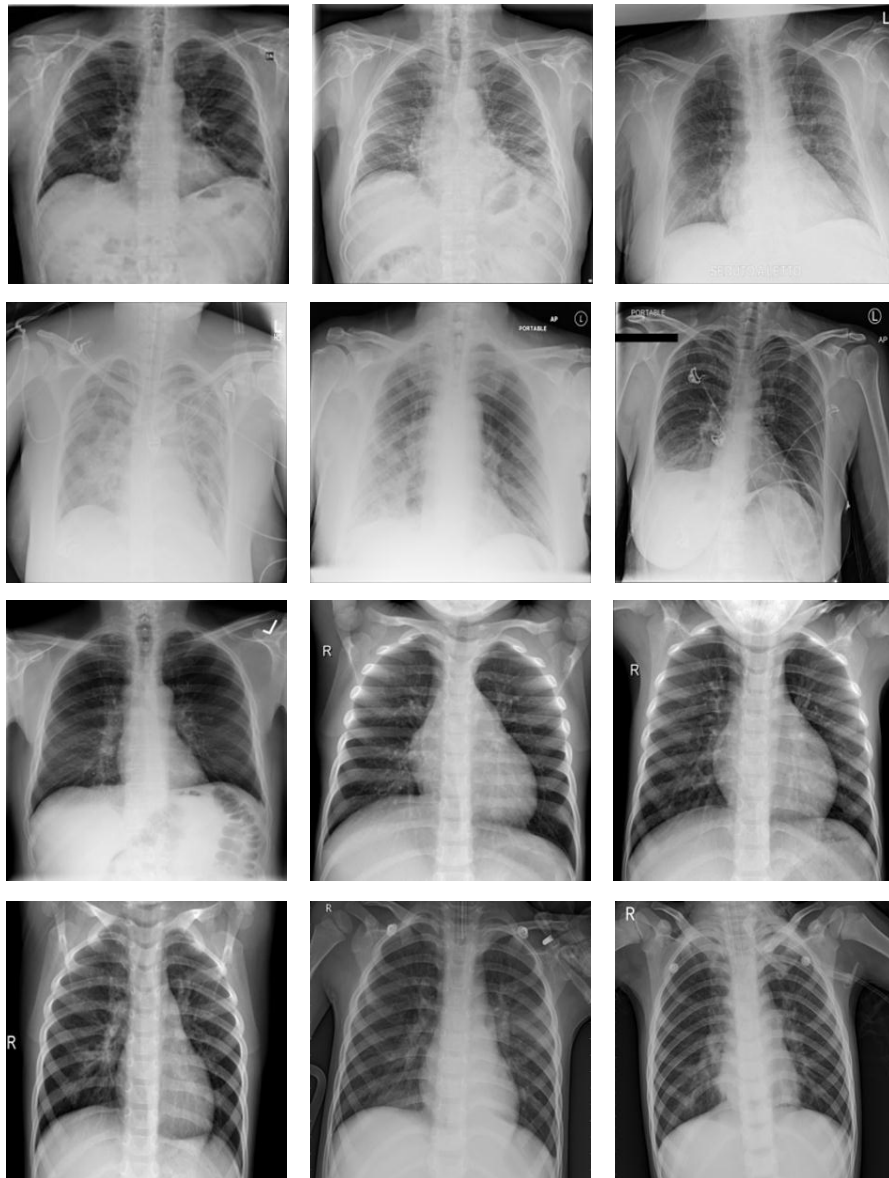


Figure III- 1: some samples from the dataset (the first row is for covid-19 samples the second is for lung opacity samples, the third for normal samples and the fourth is for viral Pneumonia samples)

### III.4. Development tools

All the experiments we performed in this work were done using python language and the tensorflow deep learning framework, which includes the keras deep learning library that was also used. As for the hardware used, these experiments were performed using a computer with a 9th generation Intel Core I7 processor and 16 GB of RAM.

For the models used, four different pre-trained models available in keras were used, which are VGG16 [59], ResNet101V2 [84], MobileNetV2 [85], EfficientNetV2L [86], and then they were compared in terms of the accuracy of performance and the accuracy of the visual explanation.

## CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

In order to build the web application that includes the final form for doing the classification, both django and react frameworks are used.

### III.5. Implementation workflow

After getting the necessary data and dividing it into three subsets: the training set containing 95 % of the total images number, and the remaining 5 % divided equally into a validation set and another for the test, we applied a transfer learning technic to the four models, where for each model we removed the fully connected layers and replace it with GAP layer and five fully connected layers with 256, 128, 64, 32, 4 neurons respectively for each layer. While training each model we freeze the convolutional layers to preserve the previous parameters of the network and only train the layers we added for 25 epochs using mini batch gradient descent approach with 32 for batch size and Adam as optimizer. The training results for each model are presented in the following table:

Model \ Accuracy	Train dataset	Validation dataset	Test dataset
VGG16	0.9186	0.9257	0.9028
ResNet101V2	0.9756	0.9805	0.9844
MobileNetV2	0.9729	0.9681	0.9878
EfficientNetV2L	0.7567	0.742	0.7655

Table III- 1: training results

# CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

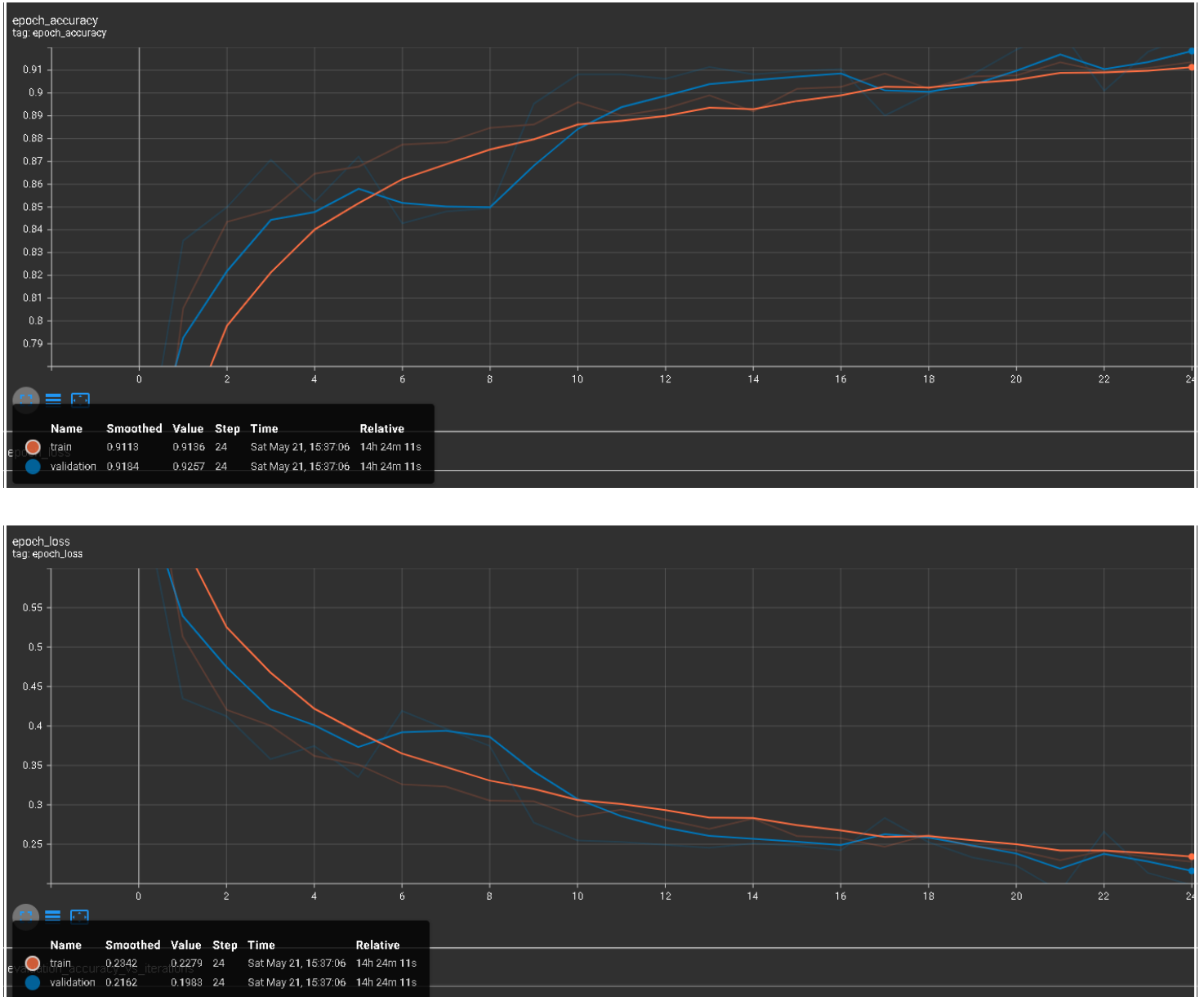


Figure III- 2: VGG16 training accuracy (top) and loss (bottom) for train and validation dataset



### CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

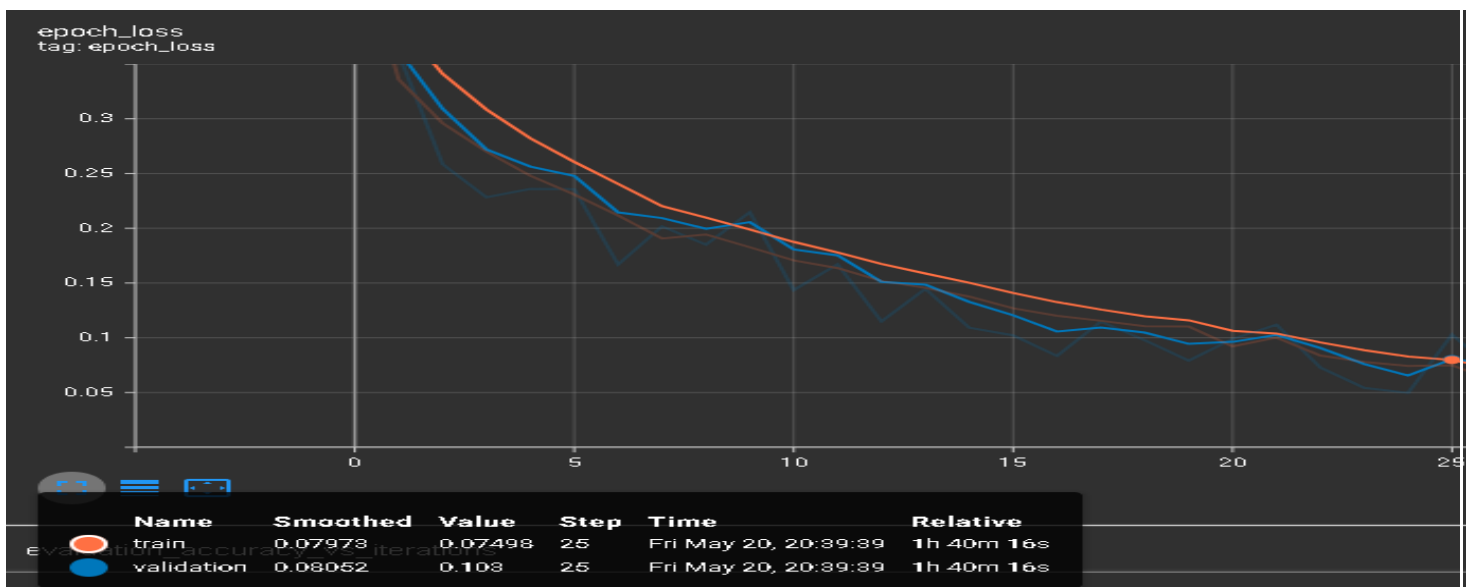
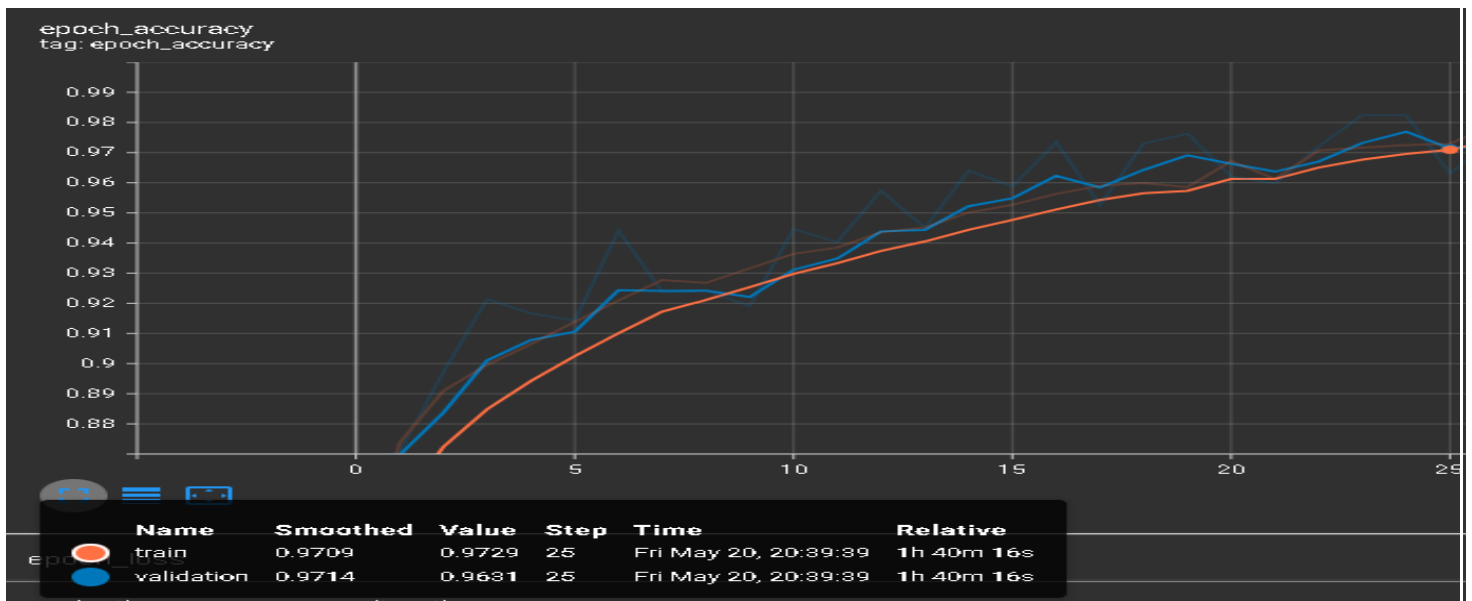


Figure III- 3: MobileNetV2 training accuracy (top) and loss (bottom) for train and validation dataset

### CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

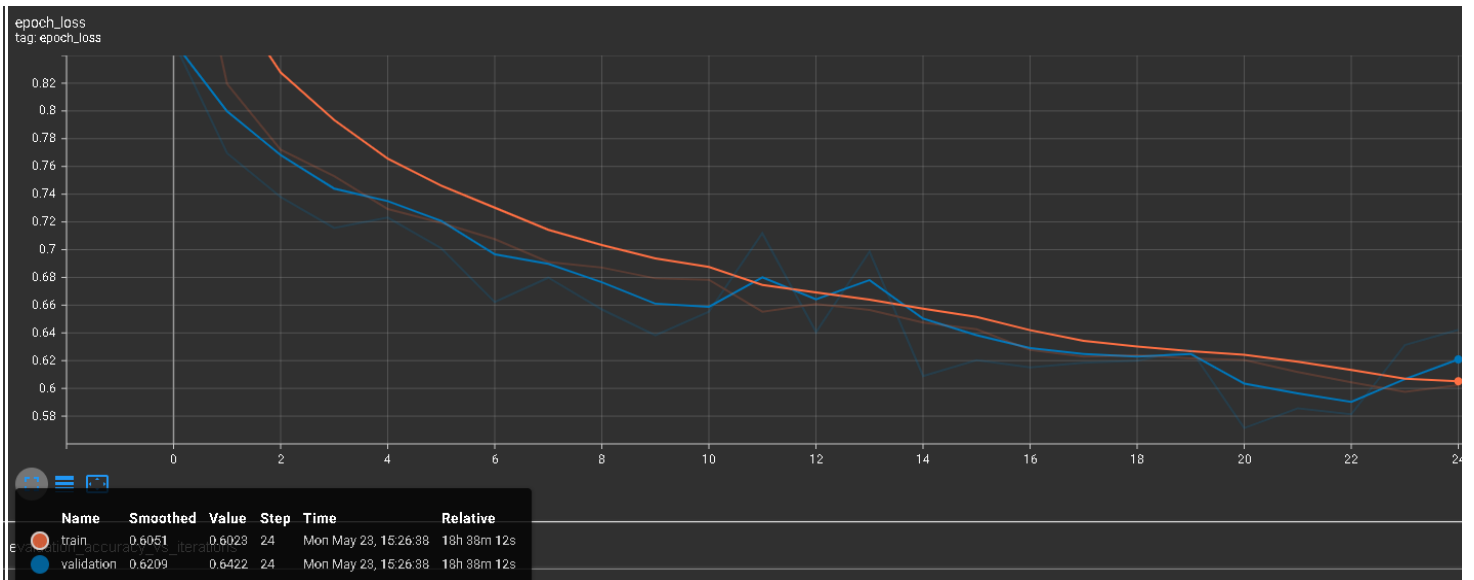
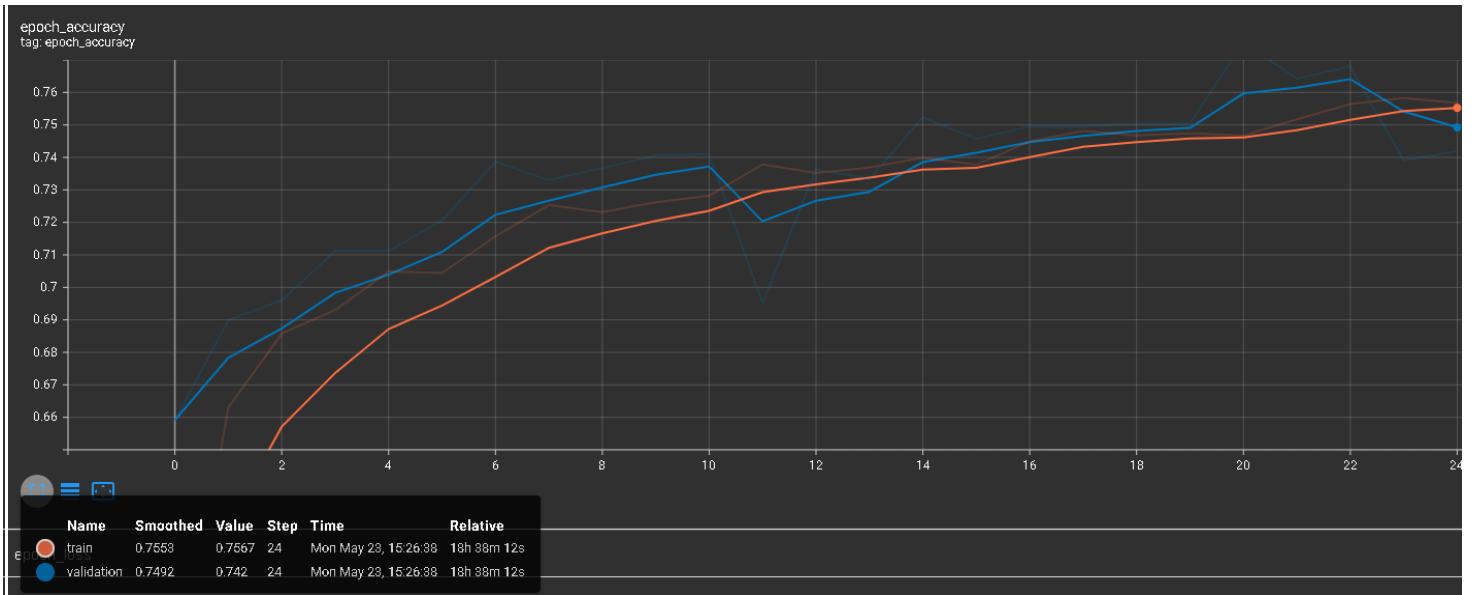


Figure III- 4: EfficientNetV2L training accuracy (top) and loss (bottom) for train and validation dataset

## CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

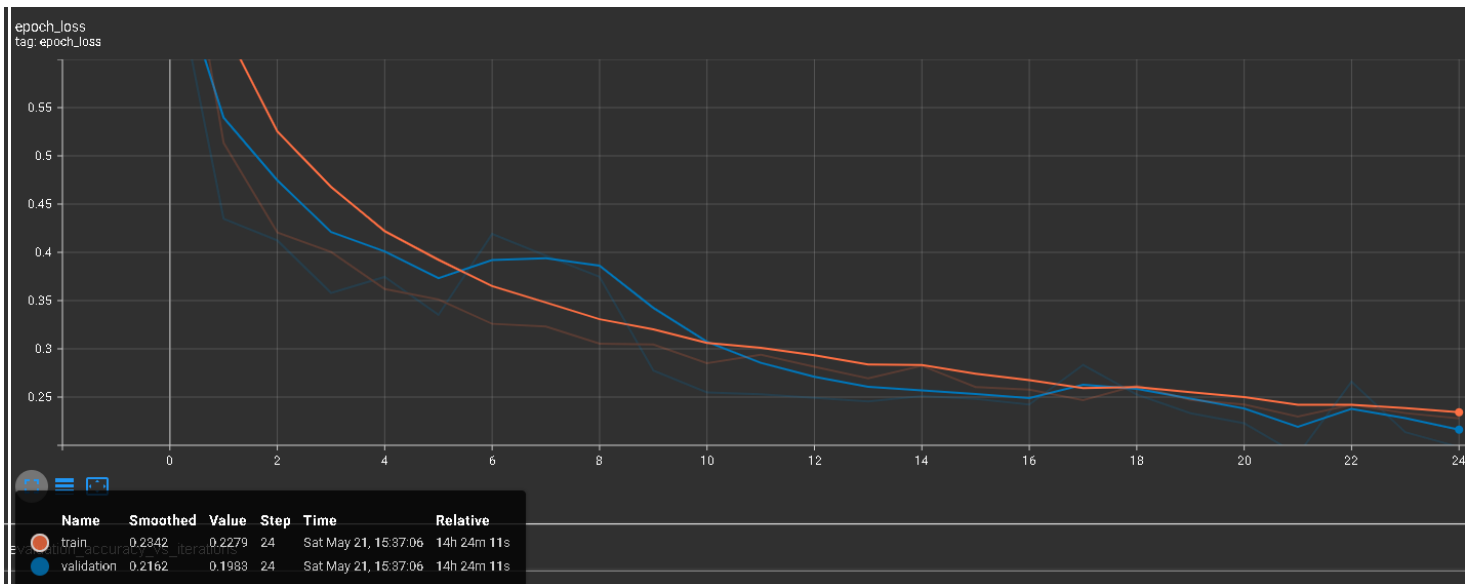
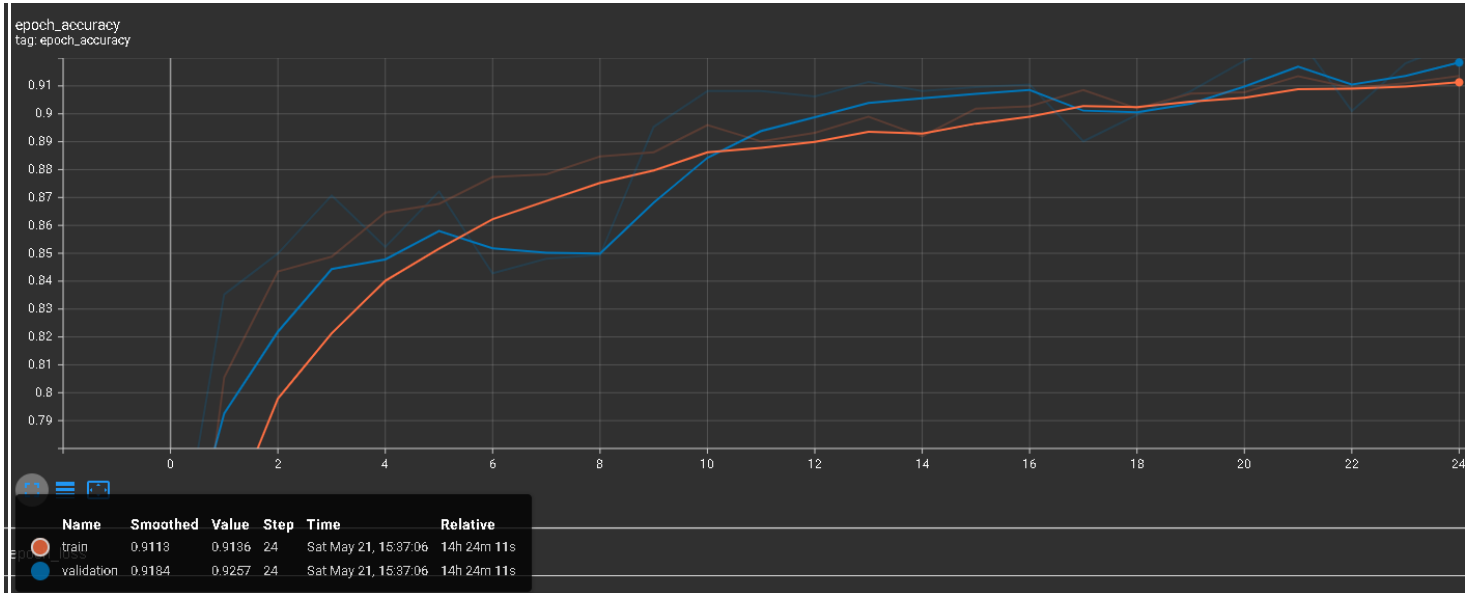


Figure III- 5: ResNet101V2 training accuracy (top) and loss (bottom) for train and validation dataset

### CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

After training the models from the results, we can note that the accuracy in both models ResNet101V2 and MobileNetV2 is high. We applied the GRAD CAM method on them in order to obtain a visual explanation of the obtained results. During the experiments, we noticed that the model based on ResNet101V2 gives good performance in addition to providing relevant visual explanation where it is not biased as much by the noise in the data as MobileNetV2 did.

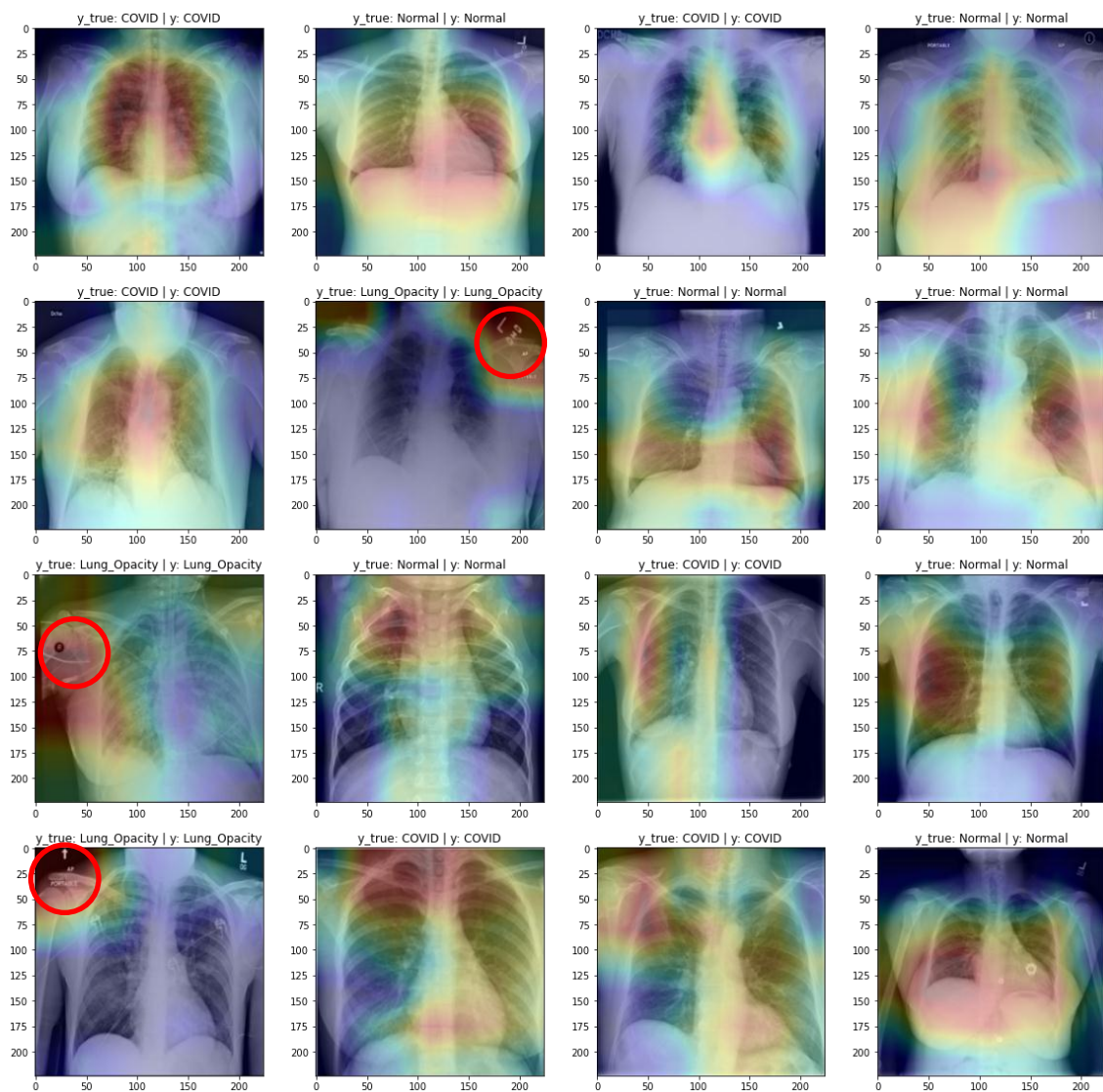


Figure III- 6: examples of Grad-CAM results for MobileNetV2

## CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

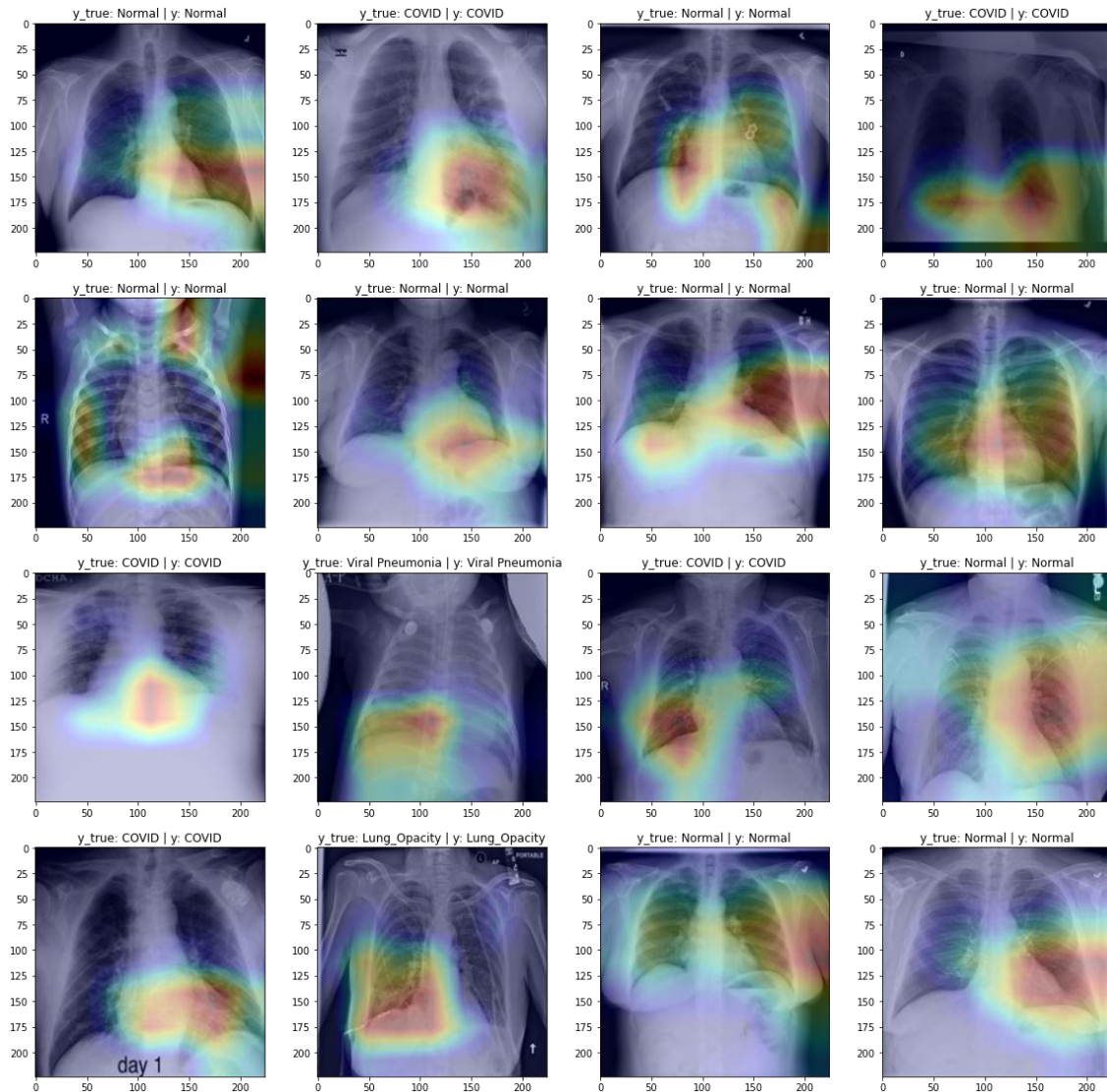


Figure III- 7: examples of Grad-CAM results for ResNet101V2

Based on these results we decide to retry the experiment with ResNet101V2 and this time we train it for 30 epochs, after the training we get the following results:

Model \ Accuracy	Train dataset	Validation dataset	Test dataset
ResNet101V2 for 30 epochs	0.9864	0.9883	0.9913

Table III- 2: training results

### CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

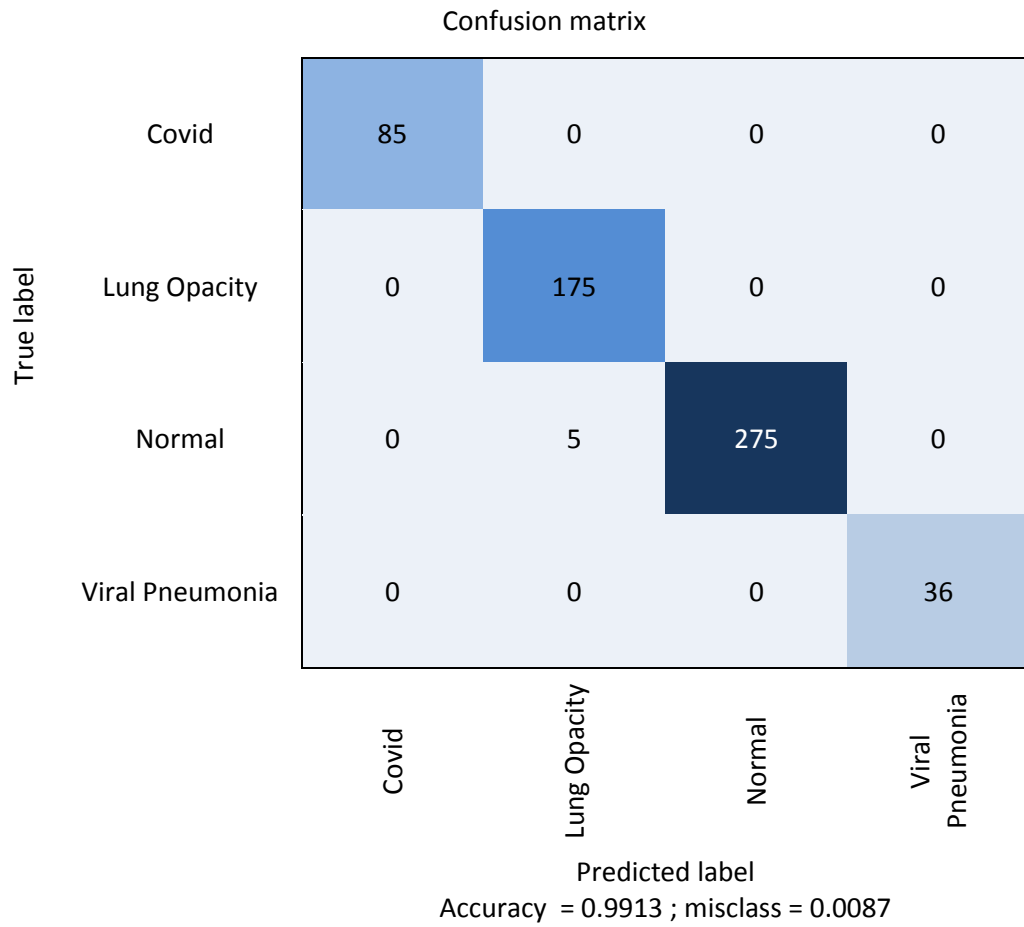


Table III- 3: the model's confusion matrix

	Precision	recall	F1-score	Support
0	1.00	1.00	1.00	85
1	0.97	1.00	0.99	175
2	1.00	0.98	0.99	280
3	1.00	1.00	1.00	36
<b>Accuracy</b>			0.99	576
<b>Macro avg</b>	0.99	1.00	0.99	576
<b>Weighted avg</b>	0.99	0.99	0.99	576

Table III- 4: image classification metrics for the model

### CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

Then we applied the GRAD CAM on the new model. The results are the following

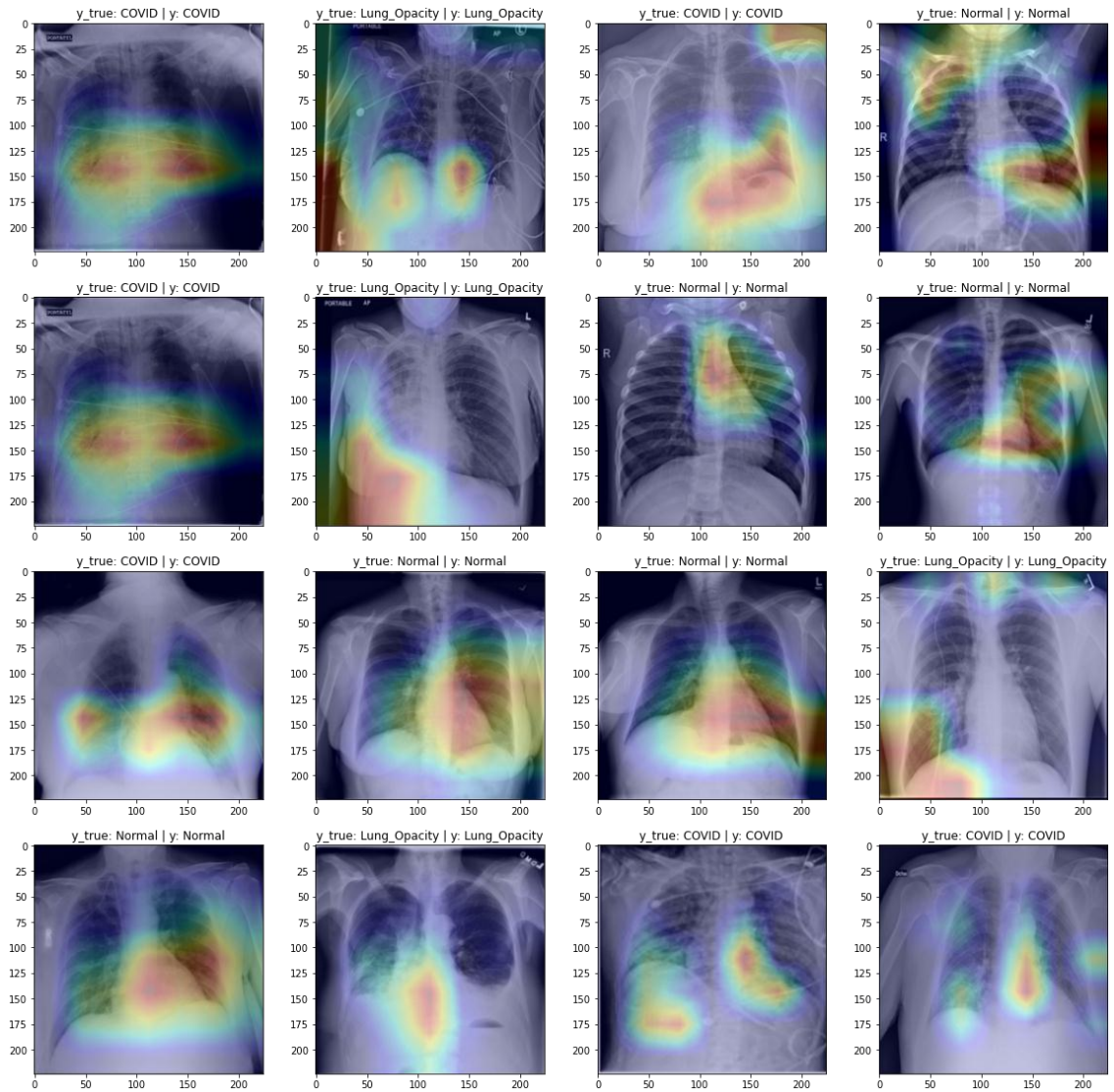


Figure III- 8: examples of Grad-CAM results for ResNet101V2 with 30 epochs

### CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

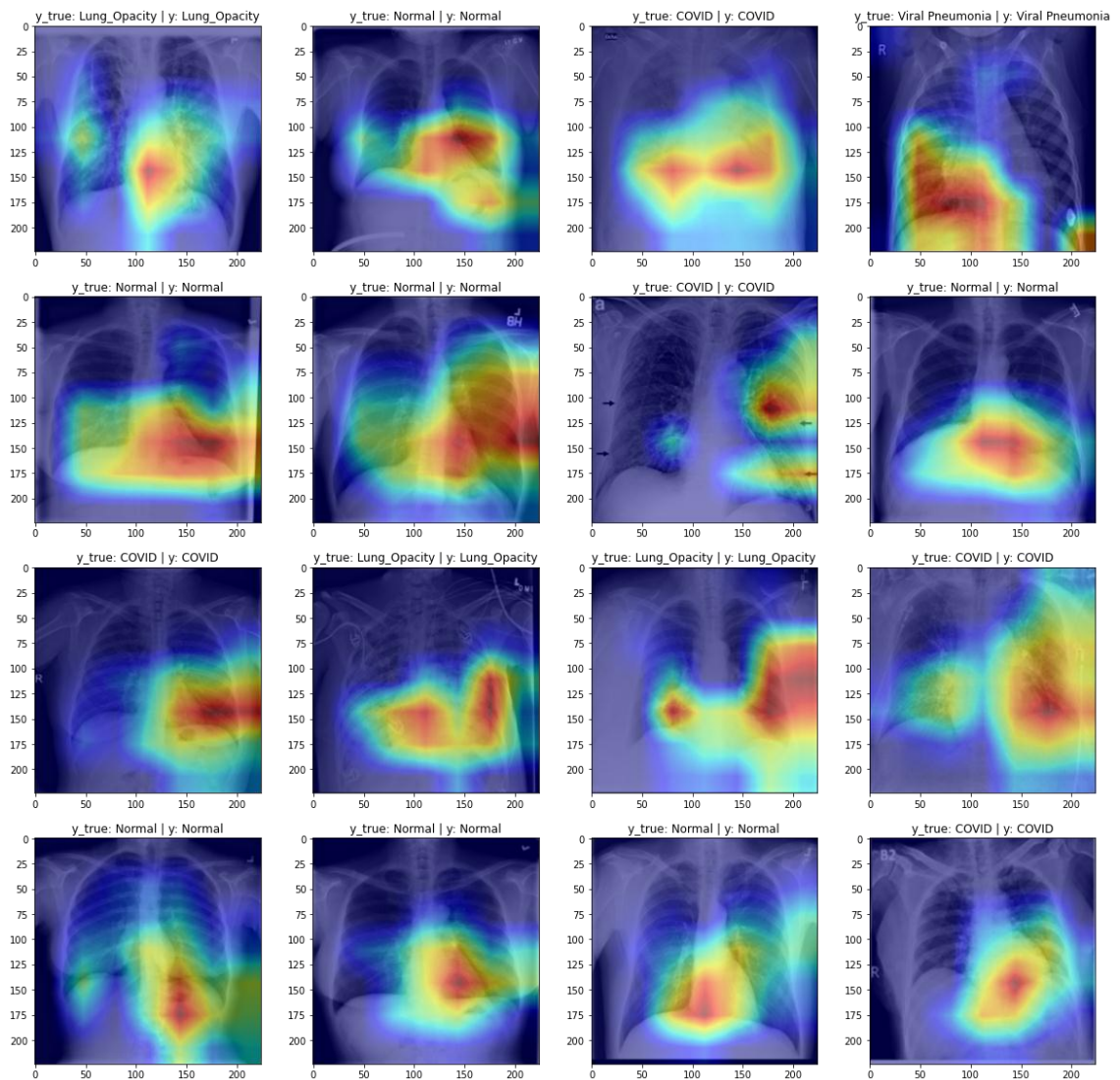


Figure III- 9: examples of Grad-CAM results for ResNet101V2 with 30 epochs and alpha=0.5

After these results we decided to choose this template to use in our web application to make it available to everyone. Through this application it is possible to predict whether a person is infected or not with a visual explanation of the results using Grad-CAM in addition to the availability of the Guided Grad-CAM method as well.



# CHAPTRE. III: MODEL DESIGN AND IMPLEMENTATION

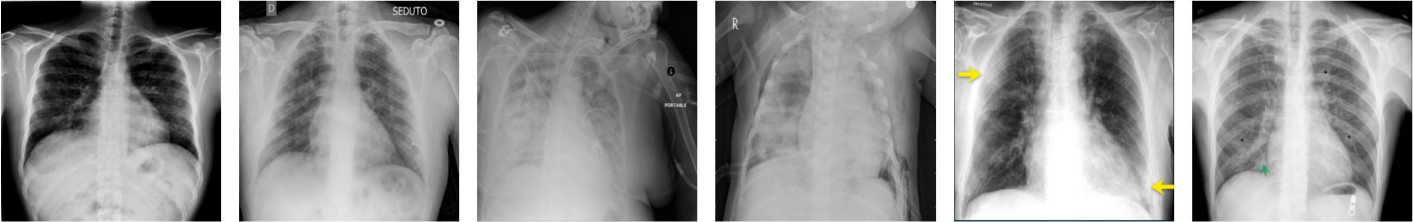
COVID-19 DTECTOR [Home](#) [Download](#) [About](#)

## COVID-19 DTECTOR A simple web application to detect COVID-19

**Hello, world!**

This is a simple website for the purpose of detecting the Corona virus. It includes a means of visual explanation of what the model sees using Grad-CAM method. I hope it is a start to create something unique

You can try the model by choosing one of this image samples:



Or you can upload your own images:


Image input

Choisir un fichier | Aucun fichier choisi

Scan

the prediction class is Lung\_Opacity

why it is ?



COVID-19: 0.00%
Lung_Opacity: 100.00%
Normal: 0.00%
Viral Pneumonia: 0.00%




Figure III- 10: our web application home page

### III.6. Conclusion

In this chapter we presented our contribution to explainability of the convolutional neural network and for covid-19 chest X-ray classification. During this work we arrived at that the choice of the dataset is very important, for example in our work the biggest obstacle was the bias of the models to patterns that are not related to the diagnosis of these diseases, as we noticed that these models provide a correct prediction of the disease, but when we refer to the results of the visual explanation, we find that they focus on the numbers and letters on the images or take a decision based on the presence of the shoulder bones for example.

Therefore, the availability of noise-free data is desirable. Also, the use of other methods to identify important areas in the diagnosis can help a lot, for example, it can help a lot in the field of medical diagnosis by creating masks to identify the important parts only using semantic segmentation for example.

# **GENERAL CONCLUSION**

## GENERAL CONCLUSION

### General conclusion

Our work aims to build a deep learning model based on convolutional neural networks, which is one of the most important deep learning algorithms and a standard for computer vision systems. This model is able to classify the input chest x-ray images and predict whether the input image is an image of a person infected with the Corona virus or not.

Given that convolutional neural networks are difficult to explain, as their complex structure makes knowing what the model relied on in building its decisions very difficult, so the second goal of this work is to provide the user with a visual explanation of the results of the model in order to increase the confidence of the user in the results of the model as it helped us to check if the model is performing correctly or not.

During this work, we passed by three chapters, where the first chapter was an introduction to the field of computer vision, where we started with a definition of this field and its various applications. We also focused on the image classification, one of the most important applications of computer vision. As we have seen, its types, dataset and metrics. The second chapter is an introduction to deep learning, where we have seen the concept of deep learning and its applications, a brief history of it, and also the artificial neural networks, the basis of deep learning such as convolutional neural networks, and in the end we saw some methods used to visually explain these networks. Finally, we dedicated the third chapter to talk about the various stages and means used to achieve the objectives of this work.

# **REFERENCES AND BIBLIOGRAPHY**

## REFERENCES AND BIBLIOGRAPHY

### Bibliography

- [2] Ramprasaath R Selvaraju et al., "Grad-cam: Visual explanations from deep networks via gradient-based localization," *Proceedings of the IEEE international conference on computer vision*, pp. 618-626, 2017.
- [4] Jason Brownlee, *Deep learning for computer vision: image classification, object detection, and face recognition in python.*: Machine Learning Mastery, 2019.
- [5] Chollet François, *Deep learning with Python.*: Manning Shelter Island, 2017.
- [11] Alex Krizhevsky and Geoff Hinton, "Convolutional Deep Belief Networks on CIFAR-10," *Unpublished manuscript*, pp. 1-9, 2010.
- [13] Li Fei-Fei et al., "ImageNet: A large-scale hierarchical image database," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248-255, 2009.
- [15] Rosebrock Adrian, *Deep Learning for Computer Vision with Python.*: PyimageSearch, 2017, vol. 1.
- [20] Warren. S McCulloch and Walter Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115-133, 1943.
- [21] Frank Rosenblatt, "The perceptron—a perceiving and recognizing automation," *Cornell Aeronautical Laboratory*, 1957.
- [22] "NEW NAVY DEVICE LEARNS BY DOING," *New York Times*, p. 25, 1958.
- [23] Jan Mycielski, minsky Marvin, and papert Seymour, "perceptrons, an introduction to computational geometry," *Bulletin of the American Mathematical Society*, vol. 78, no. 1, pp. 12-15, 1972.
- [25] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533-536, 1986.
- [28] Mikel Olazaran, "A sociological study of the official history of the perceptrons controversy," *Social Studies of Science*, vol. 26, no. 3, pp. 611-659, 1996.
- [30] Murilo Gustineli, "A survey on recently proposed activation functions for Deep Learning," *arXiv preprint arXiv:2204.02921*, 2022.
- [31] Leonid Datta, "A survey on activation functions and their relation with xavier and he normal initialization," *arXiv preprint arXiv:2004.06632*, 2020.
- [34] Yoshua Bengio , Aaron Courville Ian Goodfellow, *Deep Learning*. USA: MIT Press, 2016.
- [38] Ning Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145-151, 1999.

## REFERENCES AND BIBLIOGRAPHY

- [39] Sebastian Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.
- [40] Yurii Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence  $O(1/k^2)$ ," *Doklady an ussr*, vol. 269, pp. 543-547, 1983.
- [41] John Duchi, Elad Hazan, and Yoram Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of machine learning research*, vol. 12, no. 7, 2011.
- [42] Matthew, D Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.
- [43] Diederik, P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [44] Y LeCun et al., "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541-551, 1989.
- [45] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep learning.*: MIT press, 2016.
- [47] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 84–90, 2012.
- [54] Yann LeCun, Léon Bottou, Yoshua Bengio, and P Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, pp. 2278-2324, 1998.
- [55] Francois Chollet, *Deep learning with Python*. USA: Simon and Schuster, 2021.
- [56] David H Hubel and Torsten N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of physiology*, vol. 148, no. 3, pp. 574-591, 1959.
- [57] Kunihiko Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybernetics*, vol. 36, pp. Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern193–202, 1980.
- [58] Szegedy Christian et al., "Going Deeper with Convolutions," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1-9, 2015.
- [59] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [60] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778, 2016.

## REFERENCES AND BIBLIOGRAPHY

- [61] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson, "How transferable are features in deep neural networks?," *Advances in neural information processing systems*, vol. 27, 2014.
- [62] Razavian, Ali Sharif, Hossein Azizpour, Josephine Sullivan, and S Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 806-813, 2014.
- [64] Yudkowsky Eliezer, "Artificial Intelligence as a Positive and Negative Factor in Global Risk," *Global Catastrophic Risk*, pp. 308–345, 2008.
- [68] Laurens Van der Maaten and Geoffrey Hinton, "Visualizing data using t-SNE," *Journal of machine learning research*, vol. 9, no. 11, 2008.
- [69] Matthew D Zeiler and Rob Fergus, "Visualizing and understanding convolutional networks," *European conference on computer vision. Springer*, pp. 818-833, 2014.
- [70] Simonyan Karen, Vedaldi Andrea, and Zisserman Andrew, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," *arXiv preprint arXiv:1312.6034*, 2013.
- [71] D, Zeiler Matthew, W, Taylor Graham, and Fergus Rob, "Adaptive deconvolutional networks for mid and high level feature learning," *International Conference on Computer Vision*, pp. 2018-2025, 2011.
- [72] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Riedmiller Martin, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.
- [73] R. Selvaraju Ramprasaath et al., "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization," *Proceedings of the IEEE international conference on computer vision*, pp. 618-626, 2017.
- [74] Zhou Bolei, Khosla Aditya, Lapedriza Agata, Oliva Aude, and Torralba Antonio, "Learning Deep Features for Discriminative Localization," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921-2929, 2016.
- [75] Harsh Panwar et al., "A deep learning and grad-CAM based color visualization approach for fast detection of COVID-19 cases using chest X-ray and CT-Scan images," *Chaos, Solitons & Fractals*, vol. 140, p. 110190, 2020.
- [76] Muhammad Umair et al., "Detection of COVID-19 Using Transfer Learning and Grad-CAM Visualization on Indigenously Collected X-ray Dataset," *Sensors*, vol. 21, no. 17, p. 5813, 2021.
- [77] Arman Haghanifar, Mahdiyar Molahasani Majdabadi, Younhee Choi, S Deivalakshmi, and Seokbum Ko, "COVID-CXNet: Detecting COVID-19 in frontal chest X-ray images using deep learning," *Multimedia Tools and Applications*, pp. 1-31, 2022.



## REFERENCES AND BIBLIOGRAPHY

- [78] Rajpurkar Pranav et al., "CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning," *arXiv preprint arXiv:1711.05225*, 2017.
- [79] Omar S Hemied et al., "A COVID-19 Visual Diagnosis Model Based on Deep Learning and GradCAM," *IEEJ Transactions on Electrical and Electronic Engineering*, 2022.
- [80] Chowdhury Muhammad E. H. et al., "Can AI Help in Screening Viral and COVID-19 Pneumonia?," *IEEE Access*, vol. 8, pp. 132665-132676, 2020.
- [81] Rahman Tawsifur et al., "Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images," *Computers in Biology and Medicine*, vol. 132, 2021.
- [84] He Kaiming, Zhang Xiangyu, Ren Shaoqing, and Sun Jian, "Identity Mappings in Deep Residual Networks," *European conference on computer vision*, pp. 630-645, 2016.
- [85] Sandler Mark, Howard Andrew, Zhu Menglong, Zhmoginov Andrey, and Chen Liang-Chieh, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510-4520, 2018.
- [86] Mingxing Tan and Quoc Le, "Efficientnetv2: Smaller models and faster training," *International Conference on Machine Learning*, pp. 10096-10106, 2021.
- [91] Christian JANIESCH, Patrick ZSCHECH, and Kai HEINRICH, "Machine learning and deep learning," *Electronic Markets*, vol. 31, no. 3, pp. 685-695, 2021.
- [92] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.
- [93] T. N. Y TIMES, "New navy device learns by doing," *Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser*, 1958.
- [94] Yihong Gong and Wei Xu, "Machine learning for multimedia content analysis," *Springer Science & Business Media*, vol. 30, 2007.

## REFERENCES AND BIBLIOGRAPHY

### Webography

- [1] Hannah Ritchie et al. (2020, March) Our World in Data. [Online].  
<https://ourworldindata.org/coronavirus>
- [3] IBM. IBM. [Online]. <https://www.ibm.com/topics/computer-vision>
- [6] Kolungade Shivani. (2020, august) medium. [Online].  
<https://medium.com/@kolungade.s/object-detection-image-classification-and-semantic-segmentation-using-aws-sagemaker-e1f768c8f57d>
- [7] Meel Vidushi. (2022) viso.ai. [Online]. <https://viso.ai/applications/computer-vision-applications/>
- [8] Bandyopadhyay Hmrishav. (2022, May) V7 labs. [Online].  
<https://www.v7labs.com/blog/image-classification-guide>
- [9] LeCun Yann, Cortes Corinna, and J.C. Burges Christopher. THE MNIST DATABASE of handwritten digits. [Online]. <http://yann.lecun.com/exdb/mnist/>
- [10] Krizhevsky Alex, Nair Vinod, and Hinton Geoffrey. CIFAR-10 and CIFAR-100 datasets. ] [Online]. <https://www.cs.toronto.edu/~kriz/cifar.html>
- [12] kaggle. kaggle. [Online]. <https://www.kaggle.com/datasets/msambare/fer2013>  
]
- [14] Sharma Shivam. (2020, May) Towards Data Science. [Online].  
] <https://towardsdatascience.com/how-to-train-cnns-on-imagenet-ab8dd48202a9>
- [16] source. [Online]. <https://cs.stanford.edu/people/karpathy/cnnembed/>  
]
- [17] Lasorsa Caroline. Superb AI. [Online]. <https://www.superb-ai.com/blog/an-introduction-to-image-classification-superb-ai-tutorial>
- [18] IBM Cloud Education. (2020, May) IBM. [Online]. <https://www.ibm.com/cloud/learn/deep-learning#:~:text=Deep%20learning%20is%20a%20subset,from%20large%20amounts%20of%20data>
- [19] Chatterjee Marina. (2022, January) Great Learning. [Online].  
] <https://www.mygreatlearning.com/blog/deep-learning-applications/>
- [24] Wylie Caspar. (2018, April) opendatascience. [Online]. <https://opendatascience.com/the-history-of-neural-networks-and-ai-part-i/>
- [26] Keith D. Foote. (2022, February) dataversity. [Online]. <https://www.dataversity.net/brief-history-deep-learning/>

## REFERENCES AND BIBLIOGRAPHY

- [27 cusabio. cusabio. [Online]. <https://www.cusabio.com/Cell-Marker/Neuron-Cell.html>  
]
- [29 Wood Thomas. (2021, january) deepai. [Online]. <https://deepai.org/machine-learning-glossary-and-terms/activation-function>  
]
- [32 Pere Christophe. (2020, Jun) Towards Data Science. [Online].  
] <https://towardsdatascience.com/what-is-loss-function-1e2605aeb904>
- [33 Parmar Ravindra. (2018, September) Towards Data Science. [Online].  
] <https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23>
- [35 Oppermann Artem. (2022, February) Built in. [Online]. <https://builtin.com/machine-learning/loss-functions>  
]
- [36 Donges Niklas. (2021, August) Built in. [Online]. <https://builtin.com/data-science/gradient-descent>  
]
- [37 geeksforgeeks. (2020, jun) geeksforgeeks. [Online].  
] <https://www.geeksforgeeks.org/gradient-descent-algorithm-and-its-variants/?ref=rp>
- [46 3Blue1Brown. (2017, November) YouTube. [Online].  
] <https://www.youtube.com/watch?v=Ilg3gGewQ5U>
- [48 Madan Piyush and Madhavan Samaya. (2020, novembre) IBM Developer. [Online].  
] <https://developer.ibm.com/learningpaths/get-started-with-deep-learning/an-introduction-to-deep-learning/>
- [49 Madhavan Samaya and Jones M. Tim. (2021, january) IBM Developer. [Online].  
] <https://developer.ibm.com/learningpaths/get-started-with-deep-learning/deep-learning-architectures/>
- [50 S. Batista David. (2018, mars) David S. Batista. [Online].  
] <https://www.davidsbatista.net/blog/2018/03/31/SentenceClassificationConvNets/>
- [51 Kumar Niranjan. (2019, October) Towards Data Science. [Online].  
] <https://towardsdatascience.com/visualizing-convolution-neural-networks-using-pytorch-3dfa8443e74e>
- [52 geeksforgeeks. (2021, october) geeksforgeeks. [Online].  
] <https://www.geeksforgeeks.org/cnn-introduction-to-padding/>
- [53 sebastian. (2021, December) Programmatically. [Online].  
] <https://programmatically.com/understanding-padding-and-stride-in-convolutional-neural-networks/>
- [63 Deshpande Adit. (2016, July) A Beginner's Guide To Understanding Convolutional Neural  
] Networks Part 2. [Online]. <https://adeshpande3.github.io/A-Beginner%27s-Guide-To->

## REFERENCES AND BIBLIOGRAPHY

[Understanding-Convolutional-Neural-Networks-Part-2/](#)

- [65 Kaufman Jeff. (2017, October) jefftk. [Online]. <https://www.jefftk.com/p/detecting-tanks>  
]
- [66 Rosebrock Adrian. (2020, March) Py Image Search. [Online].  
] <https://pyimagesearch.com/2020/03/09/grad-cam-visualize-class-activation-maps-with-keras-tensorflow-and-deep-learning/>
- [67 ConvNetJS CIFAR-10 demo. [Online].  
] <https://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html>
- [82 Rahman Tawsifur, Muhammad Chowdhury, and Khandakar Amith. Kaggle. [Online].  
] [https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database?select=COVID-19\\_Radiography\\_Dataset](https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database?select=COVID-19_Radiography_Dataset)
- [83 Rahman Tawsifur, Chowdhury Muhammad, and Khandakar Amith. Kaggle. [Online].  
] [https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database?select=COVID-19\\_Radiography\\_Dataset](https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database?select=COVID-19_Radiography_Dataset)
- [87 Jason Brownlee. (2019, april) pooling layers for convolutional neural networks. [Online].  
] <https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/>
- [88 Samaya Madhavan and M. Tim Jones. (2021, january) IBM Developer. [Online].  
] <https://developer.ibm.com/learningpaths/get-started-with-deep-learning/deep-learning-architectures/>
- [89 Kamalika Some. (2018, October) analyticsinsight. [Online].  
] <https://www.analyticsinsight.net/the-history-evolution-and-growth-of-deep-learning/#:~:text=The%20history%20of%20deep%20learning,to%20mimic%20the%20thought%20process>
- [90 Long. (2017, jun) Medium. [Online]. <https://medium.com/@Aj.Cheng/convolutional-neural-network-d9f69e473feb>  
]
- [95 [Online]. [http://cs231n.stanford.edu/slides/2019/cs231n\\_2019\\_lecture13.pdf](http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture13.pdf)  
]

## Abstract

With the rapid spread of the COVID-19 virus around the world, it is necessary to develop methods that provide rapid diagnosis of infections in a short time. Since 2012, deep learning, in particular, convolutional neural networks, has achieved impressive results, especially in the field of image classification, which prompted researchers to exploit the superiority of these networks. One of the obstacles facing these networks is the difficulty of explaining their results, which makes them considered as a black box. Fortunately, there are a number of methods that are used to provide a visual explanation of models based on convolutional neural networks in order to increase confidence in their results. One of the most prominent of these methods is the Grad-CAM method. In this work, we combine convolutional neural networks with Gradient-Class-Activation-Maps (Grad-CAM) in diagnosing and interpreting Covid19 on X-ray images.

## Résumé

Avec la propagation rapide du virus COVID-19 dans le monde, il est nécessaire de développer des méthodes permettant un diagnostic rapide des infections en peu de temps. Depuis 2012, le deep learning, en particulier les réseaux de neurones convolutionnels, a obtenu des résultats impressionnants, notamment dans le domaine de la classification d'images, ce qui a poussé les chercheurs à exploiter la supériorité de ces réseaux. L'un des obstacles auxquels sont confrontés ces réseaux est la difficulté d'expliquer leurs résultats, ce qui les fait considérer comme une boîte noire. Heureusement, il existe un certain nombre de méthodes qui sont utilisées pour fournir une explication visuelle des modèles basés sur des réseaux de neurones convolutionnels afin d'augmenter la confiance dans leurs résultats. L'une des plus importantes de ces méthodes est la méthode Grad-CAM. Dans ce travail, nous combinons les réseaux convolutionnels avec les cartes des activations de classes basées sur le gradient (Grad-CAM) pour diagnostiquer et interpréter le Covid19 sur les images radiologiques.

## ملخص

مع الانتشار السريع لفيروس COVID-19 حول العالم صار من الواجب تطوير طرق توفر تشخيص سريع للإصابات و في وقت قصير. منذ سنة 2012 حقق التعلم العميق بالتحديد الشبكات العصبية التلافيفية نتائج مبهرة خاصة في مجال تصنيف الصور ما جعل الباحثين يتوجهون لاستغلال تفوق هذه الشبكات. من العوائق التي تواجه هذه الشبكات هي صعوبة شرح نتائجها ما يجعلها تعتبر كصندوق اسود، لحسن الحظ هناك مجموعة من الطرق التي تستعمل لتقديم شرح بصري للنماذج القائمة على الشبكات العصبية التلافيفية من اجل زيادة الثقة في نتائجها.

من ابرز هذه الطرق طريقة ال Grad-CAM. في هذا العمل قمنا بدمج الشبكات العصبية التلافيفية مع خرائط تنشيط الطبقة المتدرجة (Grad-CAM) لتشخيص وتفسير Covid19 على صور الأشعة السينية.