



Software Engineering Department

Master Thesis

Author Name: Mamadou Lamarana Diallo

Directeur de These : Prof. Dr. AYTUĞ ONAN

FEBRUARY 2024

This study titled "**Convolutional Neural Networks in Early Diagnosis of Diabetic Retinopathy**", prepared by Mamadou Lamarana Diallo, a student of Izmir Kâtip Çelebi University Graduate School of Natural and Applied Sciences, has been read by us, found sufficient in terms of scope and quality as a result of the defence exam and accepted as a MASTER'S THESIS by our jury.

**APPROVED BY:**

**Thesis Supervisor: Prof. Dr. AYTUĞ ONAN**  
İzmir Kâtip Çelebi Üniversitesi

**Defense date: 01.02.2024**

# Declaration of Authorship

I, **Mamadou Lamarana Diallo**, declare that this thesis, entitled **Convolutional Neural Networks for Early Detection of Diabetic Retinopathy**, and the information presented herein are my own. I also declare that:

- This work has been carried out wholly or mainly during the time I have been studying for a Master's/PhD degree at this university.
- If any part of this thesis has previously been submitted to this or any other institution for another degree or qualification, this has been clearly stated.
- Where I have referred to the published work of others, I have explicitly cited those works.
- Where I have quoted the work of others, I have always acknowledged the source. The remainder of the thesis, apart from these quotations, is entirely my own work.
- I thanked all the sources from which I received significant help.
- If there is any work carried out with others in the thesis, I have fully explained their contribution and my own work..

Date: 01.02.2024

---

# Tezin Türkçe Başlığı

## Diyabetik Retinopatinin Erken Tespitine Dayanan

## Evrişimli Sinir Ağları

### ÖZ

Son zamanlarda yapay zeka (AI), sunduğu çözümler nedeniyle bilimsel araştırmaların tüm alanlarını işgal etti. Sağlık da bir istisna değil. Diyabet dünyadaki en yaygın hastalıklardan biridir.

Komplikasyonlarından biri, hastanın görüşünü bulanıklaştırabilen veya bozabilen ve körlüğün ana nedenlerinden biri olan diyabetik retinopatidir.

Diyabetik retinopatinin erken teşhisi tedaviye büyük ölçüde yardımcı olabilir. Yapay Zeka ve özellikle derin öğrenme alanındaki son gelişmeler, birçok hastalığı erken evrelerinde tahmin etmek, öngörmek ve teşhis etmek için kullanılacak iddialı çözümler sunmaktadır.

Son yıl projemizde, retina görüntülerini analiz etmek için derin öğrenmenin potansiyelini araştırdık. Diyabetik retinopati seviyelerini otomatik olarak tespit etmemizi ve sınıflandırmamızı sağlayacak bir model oluşturmak için Derin Öğrenme (DL) kavramlarını bir konvolüsyonel sinir ağı (CNN) algoritması ile inceleyeceğiz. Göz ve diyabetik retinopati, ardından farklı diyabetik retinopati türleri, diyabetik retinopatinin nedenleri, önlenmesi, teşhisi ve uygun tedavisi hakkında bir sunum yapacağız. Modellerimizi eğitmek için herkesin erişebileceği bir platform olan Google Colab'ı kullanacağız.

Projemizde Resnet ve InceptionResnet olmak üzere iki mimariyi (CNN) farklı yapılarla sahip diyabetik retinopati veri setine uygulayacağız. Projemizin sonunda elde edilen sonuçları sunacağız..

**Keywords:** Health, Diabetic Retinopathy, Deep Learning, Artificial Intelligence

# Title of the Thesis in English

## Convolutional Neural Networks Based on Early Detection of Diabetic Retinopathy

### Abstract

Recently, artificial intelligence (AI) has invaded all areas of scientific research, because of the solutions it offers. Health is no exception. Diabetes is one of the most common diseases in the world.

One of its complications is diabetic retinopathy, which can blur or distort a patient's vision, and is one of the main causes of blindness.

Early detection of diabetic retinopathy can greatly aid treatment. Recent developments in the field of Artificial Intelligence and especially deep learning are providing ambitious solutions that can be used to predict, forecast and diagnose many diseases in their early stages.

In our final year project, we explored the potential of deep learning for analysing retinal images. We will study the concepts of Deep Learning (DL) with a convolutional neural network (CNN) algorithm to build a model that will enable us to automatically detect and classify the levels of diabetic retinopathy. We will give a presentation on the eye and diabetic retinopathy, then the different types of diabetic retinopathy, the causes, prevention, diagnosis and appropriate treatment of diabetic retinopathy. We are going to use Google Colab, which is a platform accessible to everyone, to train our models.

In our project, we will apply two architectures (CNN), Resnet and InceptionResnet, to a dataset of diabetic retinopathy with different structures. At the end of our project, we will present the results obtained.

**Keywords:** Health, Diabetic Retinopathy, Deep Learning, Artificial Intelligence

*To you, my dear parents...*

*For everything you have done for me, all your prayers, all your efforts, to see me  
happy.*

*To you, my dear father, I would like to thank you from the bottom of my heart for  
everything you have done for me from the beginning to the end of this adventure,  
may Allah give you long life.*

*To you, my dear brothers and sisters...*

*For all your moral support and encouragement.*

*I'd like to pay tribute to my big sister who has passed away. I'd like to thank you for  
everything you've done for me and know that I love you very much, may your soul  
rest in peace.*

*To all those who participated in any way in the realisation of this project.*

*I respect you all, you gave me the time of my life.*

*To all my family, friends and all those I respect, I thank you from the bottom of my  
heart....*

# Acknowledgements

I thank Almighty Allah for giving me the strength and health to complete this project.

I would like to warmly thank my advisor Dr. AYTUĞ ONAN, Head of the Software Engineering Department, for his monitoring, encouragement, moral support, availability and the exchanges we had around this project.

I would like to take this opportunity to express my sincere thanks to all the professors of Izmir Katip Çelebi İKÇÜ University. Please accept the expression of my deepest dedication and gratitude..

# Contents

Declaration of Authorship .....	ii
Öz .....	iii
Abstract .....	iv
Acknowledgements .....	vi
List of Figures .....	xii
List of Tables .....	xiii
List of Abbreviations .....	xiv
Symbols List .....	xv
<b>1 Introduction .....</b>	<b>1</b>
<b>1.2 DEEP LEARNING .....</b>	<b>1</b>
1.3 Definition and History .....	1
1.4 Why Deep Learning .....	1
2. Artificial Neuron Network .....	2
2.1. Biological Neuron .....	2
2.2. Perceptron .....	2
2.3. Artificial neuron .....	2
2.4. Activation function .....	4
2.5. Topology of Artificial Neural Networks .....	5
2.5.1 Feed Forward neural network .....	5
2.5.2. Simple Perceptron .....	6
2.5.3. Multilayer Perceptron .....	6
2.5.4. Deep neural network .....	6



2.5.5. Recurrent Neural Networks .....	6
2.5.6. Hopfield model .....	6
2.5.7. Resonance Neural Networks.....	7
2.5.8. Boltzmann machine .....	7
2.5.9. Self-organizing network.....	8
3. Loss function .....	9
3.1. Mean Squar Error .....	9
3.2. Log Loss: .....	9
3.3. Backpropagation.....	9
3.4. Gradient descent .....	9
3.5. Neural Network Layer.....	10
3.6. The perceptron consists of three layers .....	10
3.7 Input layer.....	10
3.8 Hidden layer .....	10
3.9 Output layer.....	11
4. Convolutional Neural Networks .....	11
4.1. History of the CNN.....	11
4.2. Principle of Convolutional Neural Networks .....	11
4.3. Architecture of the Convolutional Neural Network (CNN).....	12
4.4. Impact of AI on Medical Imaging .....	14
5. Conclusion.....	15
<b>1 Introduction .....</b>	<b>16</b>
1.2 The Anatomy of the Eye .....	16
1.2.1. Description of the eye .....	17
2. The Retina.....	18
2.1. The central retina .....	18
2.2. The peripheral retina .....	18

3.	RETINOPATHIES .....	18
3.1.	Diabetic Retinopathy .....	18
3.2.	Hypertensive Retinopathy .....	19
3.3.	Symptom and Complication .....	19
3.4.	Diagnostic .....	20
3.5.	Treatment and Prevention.....	21
3.6.	To treat retinopathy we have two methods.....	22
4.	Conclusion .....	22
<b>1</b>	<b>Introduction .....</b>	<b>23</b>
2.	Problem .....	23
3.	The Architecture of the convoulutional Neural Networks used in the project .....	23
3.1.	Resnet.....	23
3.1.2.	Architecture overview .....	24
3.1.3.	InceptionResnet.....	27
3.1.4.	Inception Resnet Module .....	27
3.1.5.	Inception A.....	28
3.1.6.	Inception B .....	28
3.1.7.	Inception C .....	28
3.1.8.	Residual initiation block .....	29
3.1.9.	Residual Scaling.....	29
3.1.10.	Reduction Block.....	30
3.1.11.	Block A .....	30
3.1.12.	Block B .....	30
3.1.13.	Stem .....	31
3.1.14.	Global Architecture of InceptionResnetV2.....	31
3.2.	DESCRIPTION OF THE DATASET.....	32
3.2.1.	Dataset:.....	32

3.2.2. Representation of our system .....	33
3.2.3. Preprocessing .....	33
3.2.4. Transfer Learning .....	33
3.2.5 Data increase .....	34
3.3. Realization and implementation.....	34
3.3.1. Python .....	34
3.3.2. Google Colab .....	34
3.3.3. Matplotlib.....	35
3.3.4. Keras .....	35
3.3.5. Tensorflow .....	36
3.3.6. Numpy.....	36
3.3.7. Kaggle .....	36
3.3.8. Pandas .....	37
3.3.9. Code for programming our system: .....	37
4. Conclusion .....	42
<b>1. Introduction .....</b>	<b>43</b>
2. Choice of evaluation criteria .....	43
3. Comparative study between Resnet50 and Inception .....	44
3.1. Configuration of our model parameter.....	44
3.2. Evaluation of Resnet and Inception .....	44
4. Conclusion .....	47
<b>Resources .....</b>	<b>48</b>
<b>Özgeçmiş .....</b>	<b>49</b>

# List of Figures

Figure 2.3. Biological Neuron/Artificial Neuron .....	3
Figure 2.3. Structure of an Artificial .....	4
Figure 2.4. Activation functions .....	5
Figure 2.5.6. Hopfield model.....	7
Figure 2.5.8. Boltzmann machine .....	8
Figure 2.5.9. Self-organising network.....	8
Figure 3.4 Gradient descent.....	10
Figure 3.6 Perceptrons Multicouche.....	11
Figure 4 Architecture of the CNN .....	12
Figure 4.3 Fonction d'activation RELU .....	13
Figure 1.2 Anatomy of the eye .....	16
Figure 3.4 The normal eye and the eye affected by diabetic retinopathy .....	20
Figure 3.4 Photograph of the back of the affected eye DR .....	21
Figure 3.4 Photograph of a retina affected by RH.....	21
Figure3.1.2Block Resnet with and without 1*1 .....	26
Figure3.1.2 VGG19 and RESNET architecture .....	26
Figure3.1.7 Residual learning building block .....	27
Figure3.1.4Block Inception A, B, C .....	29
Figure 3.1.9. Scaling of residues .....	30
Figure3.1.10. Block A and Block B reduction .....	31
Figure 3.1.13 Block stem .....	31
Figure 3.2.14. InceptionResnet .....	32
Figure3.2.1. Data fragmentation .....	32
Figure 3.2.2. Diagram of the detection system .....	33
Figure 3.3.2. Google Colab .....	35

Figure 3.3.3. Matplotlib .....	35
Figure 3.3.4. Keras .....	36
Figure 3.3.5. Tensorflow .....	36
Figure 3.3.6. NumPy .....	36
Figure 3.2.7. kaggle.....	37
Figure 3.3.8. Pandas .....	37
Figure 3.2 Receiver Operating Characteristic .....	46
Figure 3.2 Sensitivity and Specificity.....	46

# List of Tables

Table 3.3.2 Google Colab Resources .....	35
Table 3.2 Configuration of Resnet and Inception models .....	44
Table 3.2 Resnet50 metrics .....	47
Table 3.2 Inception metrics .....	47
Table 3.2 Resnet50 Confusion Matrix .....	45
Table 3.2 Inception Confusion Matrix .....	45

# List of Abbreviations

AI	Artificial Intelligence
CNN	Convolutional Neural Network
DR	Diabetic Retinopathy
DL	Deep Learning
İKÇÜ	İzmir Kâtip Çelebi University
MLP	Multilayer Perceptron
RH	Hypertensive retinopathy
RNN	Recurrent Neural Networks

# Symbols List

$W$	Weight
$X$	Input
$Z$	Weighted sum
$RELU(x)$	Rectified Linear Units



# Chapter 1

## 1. Introduction

### 1.2. DEEP LEARNING :

### 1.3. Definition and History :

In this chapter, we will introduce Deep Learning and explain why it has emerged and its impact. We will then look at an example of a neural network and illustrate how it works internally, before moving on to convolutional neural networks for medical imaging.

Deep Learning is a subset of Artificial Intelligence (AI) that focuses on training artificial neural networks to learn and make predictions. It is inspired by the structure and functions of the human brain, where a network of interconnected neurons processes information.

The concept of Deep Learning (DL) is not entirely new, having first appeared in the early 1940s, when **Warren McCulloch** and **Walter Pitts** proposed a mathematical model of artificial neurons. However, it was not until the 1980s that the first significant breakthrough in the field of Deep Learning (DL) took place. In the 1980s, scientists began to explore **retro-propagation**, a technique for training artificial neural networks. However, this technique was quickly abandoned due to a lack of computing power and the inability to process large quantities of data.

### 1.4. Why Deep Learning

Deep learning is important because it can automatically learn and extract meaningful patterns and representations from raw data. Unlike traditional machine learning algorithms, deep learning algorithms can learn hierarchical features directly from data without the need for manual feature engineering. This makes deep learning models highly adaptable and capable of handling large and complex datasets.

## 2. Artificial Neuron Network :

### 2.1 Biological Neuron :

A biological neuron, a fundamental component of the nervous system, acts as an information transmission unit, using electrical and chemical signals to communicate within the brain. These cells receive signals from other neurons via complex interactions between their dendrites and synapses.

### 2.2 Perceptron :

In the 1940s, researchers Warren McCulloch and Walter Pitts designed the first artificial neuron, named Perceptron. Their inspiration came from the functioning of the visual cortex of mammals. Although capable of solving certain problems, the Perceptron had its limitations. To remedy this, they came up with the idea of combining several Perceptrons, giving rise to the artificial neuron network, also known as a multi-perceptron.

### 2.3 Artificial neuron :

In the field of artificial neural networks, there are similarities with the components and functions of biological neurons. For example, the weights of the connections between neurons correspond to biological synapses, the artificial cell body is comparable to the cell body of biological neurons, and the output element in neural networks corresponds to the axon in biological neurons.

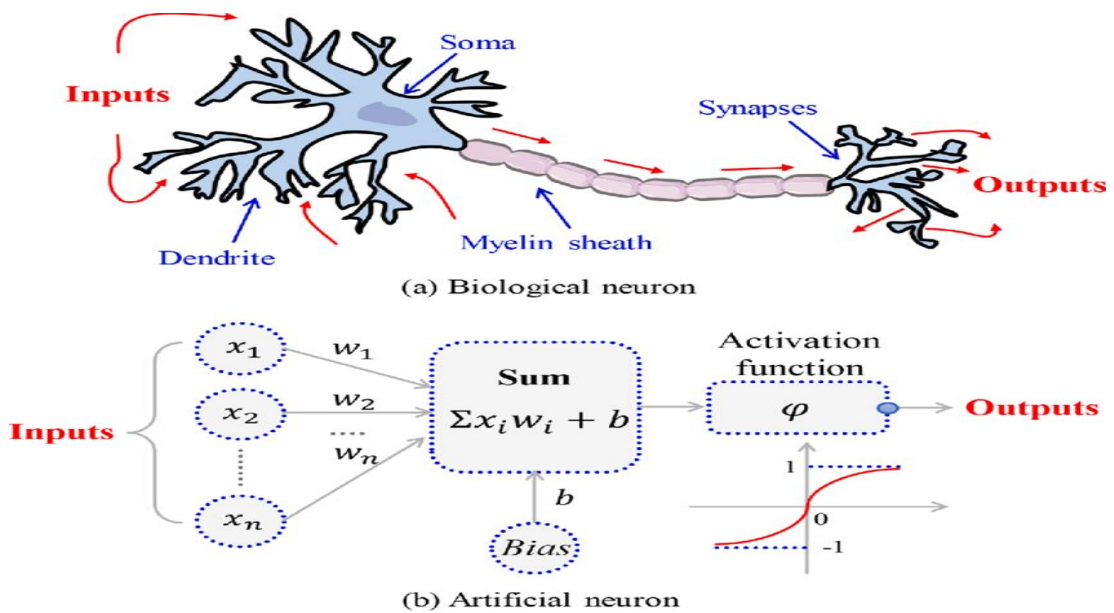


Figure 2.3 : Biological Neuron/Artificial Neuron

The structure of an artificial neuron comprises several components, such as the input, weight, weighted sum, bias, activation function and output.

**Input:** enables each artificial neuron to receive input from other neurons.

**Weight:** adjusts the impact of each input on the neuron's output during the learning process.

**Weighted sum :** is often noted as  $z = w_1 * x_1 + w_2 * x_2 + w_n * x_n$  where  $w$  are weights and  $x$  are inputs.

**Bias:** Allows us to move the output of the neuron and introduces some flexibility into the model.

**Activation function :** introduces non-linearity into the model and allows the neuron to learn complex patterns.

**Output:** is the result of applying the activation function to the weighted sum and the bias.

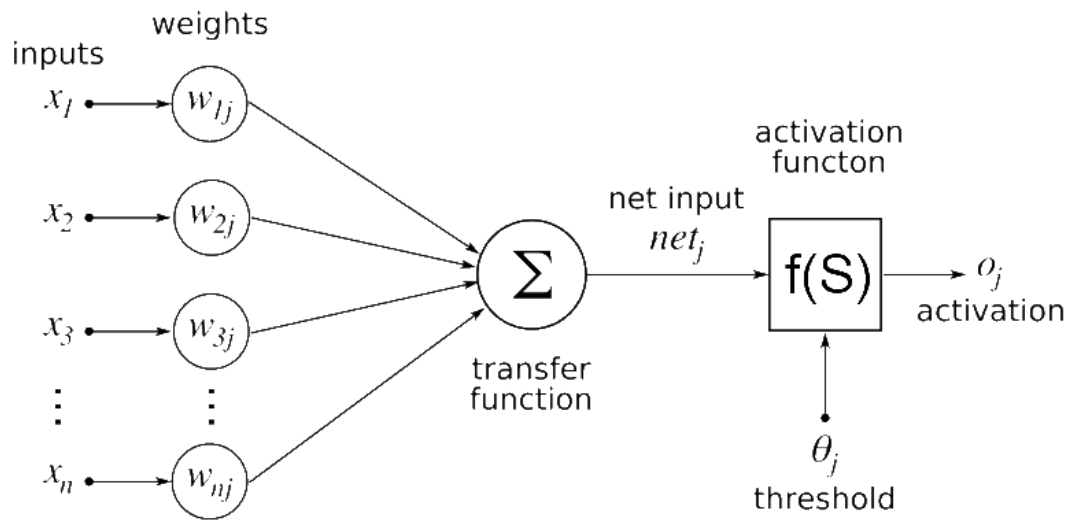


Figure 2.3: Structure of an Artificial

## 2.4 Activation function :

There are several activation functions for the artificial neuron, the choice of which depends on the function of the model we wish to model.









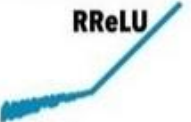
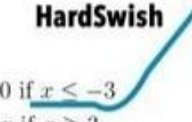
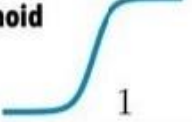
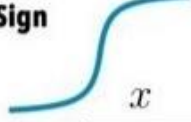

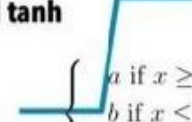
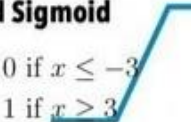
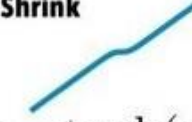
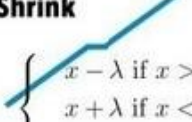
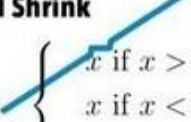
<p><b>ReLU</b></p>  <p><math>\max(0, x)</math></p>	<p><b>GELU</b></p>  <p><math>\frac{x}{2} \left( 1 + \tanh \left( \sqrt{\frac{2}{\pi}} (x + ax^3) \right) \right)</math></p>	<p><b>PReLU</b></p>  <p><math>\max(0, x)</math></p>
<p><b>ELU</b></p>  <p><math>\begin{cases} x &amp; \text{if } x &gt; 0 \\ \alpha(x \exp x - 1) &amp; \text{if } x &lt; 0 \end{cases}</math></p>	<p><b>Swish</b></p>  <p><math>\frac{x}{1 + \exp -x}</math></p>	<p><b>SELU</b></p>  <p><math>\alpha(\max(0, x) + \min(0, \beta(\exp x - 1)))</math></p>
<p><b>SoftPlus</b></p>  <p><math>\frac{1}{\beta} \log(1 + \exp(\beta x))</math></p>	<p><b>Mish</b></p>  <p><math>x \tanh \left( \frac{1}{\beta} \log(1 + \exp(\beta x)) \right)</math></p>	<p><b>RReLU</b></p>  <p><math>\begin{cases} x &amp; \text{if } x \geq 0 \\ ax &amp; \text{if } x &lt; 0 \text{ with } a \sim \mathcal{R}(l, u) \end{cases}</math></p>
<p><b>HardSwish</b></p>  <p><math>\begin{cases} 0 &amp; \text{if } x \leq -3 \\ x &amp; \text{if } x \geq 3 \\ x(x+3)/6 &amp; \text{otherwise} \end{cases}</math></p>	<p><b>Sigmoid</b></p>  <p><math>\frac{1}{1 + \exp(-x)}</math></p>	<p><b>SoftSign</b></p>  <p><math>\frac{x}{1 +  x }</math></p>
<p><b>Tanh</b></p>  <p><math>\tanh(x)</math></p>	<p><b>Hard tanh</b></p>  <p><math>\begin{cases} a &amp; \text{if } x \geq a \\ b &amp; \text{if } x \leq b \\ x &amp; \text{otherwise} \end{cases}</math></p>	<p><b>Hard Sigmoid</b></p>  <p><math>\begin{cases} 0 &amp; \text{if } x \leq -3 \\ 1 &amp; \text{if } x \geq 3 \\ x/6 + 1/2 &amp; \text{otherwise} \end{cases}</math></p>
<p><b>Tanh Shrink</b></p>  <p><math>x - \tanh(x)</math></p>	<p><b>Soft Shrink</b></p>  <p><math>\begin{cases} x - \lambda &amp; \text{if } x &gt; \lambda \\ x + \lambda &amp; \text{if } x &lt; -\lambda \\ 0 &amp; \text{otherwise} \end{cases}</math></p>	<p><b>Hard Shrink</b></p>  <p><math>\begin{cases} x &amp; \text{if } x &gt; \lambda \\ x &amp; \text{if } x &lt; -\lambda \\ 0 &amp; \text{otherwise} \end{cases}</math></p>

Figure 2.4: Activation functions

## 2.5 Topology of Artificial Neural Networks :

A neural network is made up of several interconnected layers.

### 2.5.1. Feed Foward neural networks :

This is the simplest type of neural network architecture, in which information flows in one direction from input to output.

### 2.5.2. Simple Perceptron:

This is a neural architecture characterised by a single layer of interconnected neurons. Each neuron receives input data, processes it and generates an output by applying a mathematical function to the weighted sum of the inputs. This model, known as a simple perceptron, is often used for binary classification tasks, where the network learns to categorise the input data into two distinct classes.

### 2.5.3. Multilayer Perceptron:

This is a feedforward neural network architecture comprising several layers of interconnected neurons. Data flows from the input layer to the output layer without feedback loops. The multilayer perceptron is frequently used for tasks such as classification and regression, where data is processed to predict categories or numerical values.

### 2.5.4. Deep neural network:

This is a type of forward propagation neural network (or MLP) that incorporates several hidden layers. These networks are deployed in complex areas such as image and speech recognition, where the data is highly structured and requires high levels of processing for accurate analysis.

### 2.5.5. Recurrent Neural Networks:

Recurrent neural networks (RNNs) are deep neural network architectures specifically designed for natural language processing. Unlike traditional neural networks, RNNs are suitable for processing sequential data such as sentences or sequences of words. Their key feature is their ability to use recurrent loops, enabling them to maintain and exploit contextual information from previous inputs.

### 2.5.6. Hopfield model:

The Hopfield model is a form of recurrent neural network (RNN) architecture where neurons are densely interconnected. All the neurons are organised in a single layer and each neuron is connected to all the other neurons. This configuration allows the network to store and retrieve information using asynchronous or synchronous update processes.

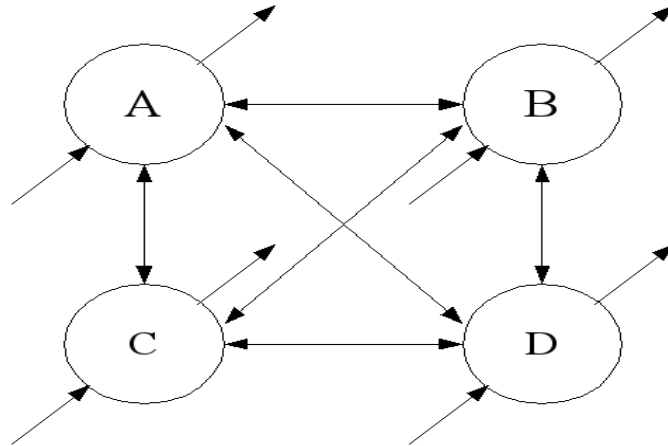


Figure 2.5.6: HopField model

### 2.5.7. Resonance Neural Networks:

### 2.5.8. Boltzmann machine:

Boltzmann machines, a type of neural network, belong to a class of probabilistic generative models where multiple neurons are connected together. They are often used for tasks such as feature extraction, data clustering and unsupervised learning.

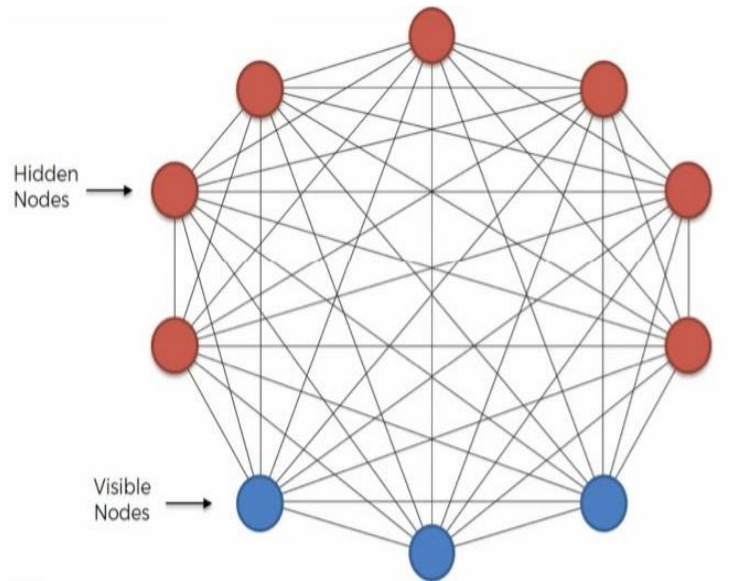


Figure2.5.8.: Boltzmann machine

### 2.5.9. Self-organizing network:

It is a neural network architecture used for unsupervised learning, capable of discovering and internalising structures and patterns from input data without relying on pre-established labels or categories. This enables it to identify relevant representations and features in the data, which is beneficial for various tasks such as pattern recognition, classification and data compression. This type of neural network is particularly suitable for analysing unstructured data such as images, text or signals.

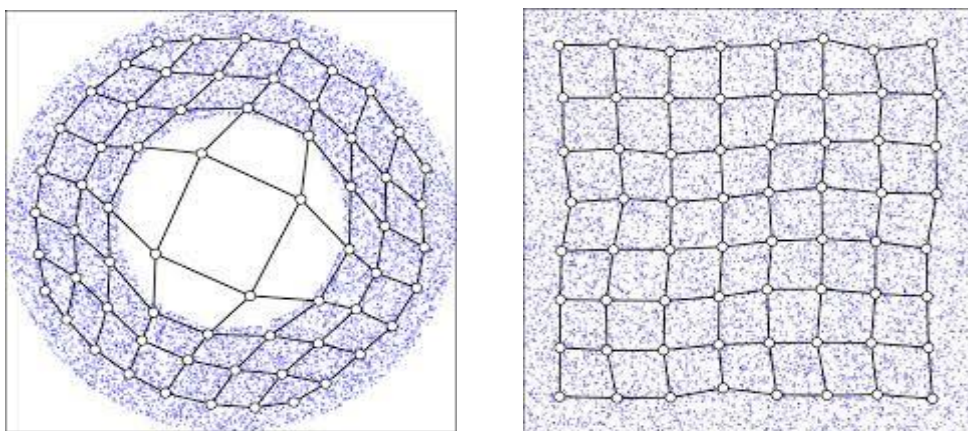


Figure 2.5.9 : Self-organising network



### 3. Loss function:

The loss function is a mathematical function that evaluates the disparity between the prediction of a machine learning model and the actual or expected result. It plays a crucial role in the training of neural networks, as it makes it possible to quantify errors and improve their accuracy by minimising them.

#### 3.1. Mean Squar Error:

The mean square error is a common measure used to evaluate the root mean square difference between predicted and actual values.

#### 3.2. Log Loss:

Log loss is a function commonly used in deep learning to evaluate the performance of a classification model producing probabilities.

#### 3.3. Backpropagation:

Backpropagation is a learning algorithm that allows neural networks to adjust their weights and biases in order to reduce the difference between the actual output and the desired output. This method is essential for training neural networks to perform complex tasks such as image recognition, natural language processing and speech recognition. In backpropagation, another algorithm called gradient descent is also involved, playing a central role in the backpropagation process.

#### 3.4. Gradient descent :

Gradient descent is an optimisation algorithm used to converge to the minimum of a convex function. This first-order iterative algorithm is frequently used in many applications such as linear regression, logistic regression and neural networks. Its objective is to find parameter values that minimise the cost function, representing the difference between predicted and actual values.

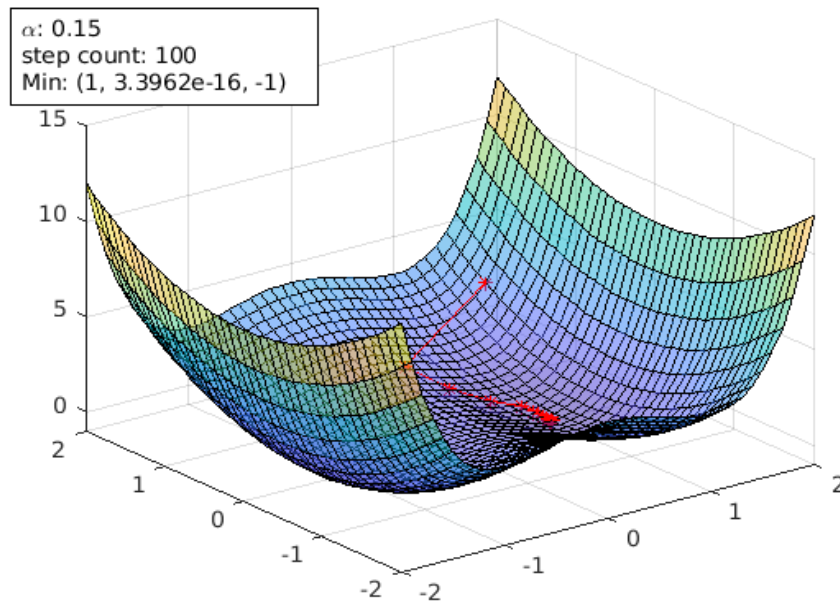


Figure 3.4 : Gradient descent

### 3.5. Neural Network Layer:

Explorons la structure d'un perceptron multicouche, un exemple de réseau de neurones.

### 3.6. The perceptron consists of three layers:

#### 3.7. Input layer:

The input layer of the neural network receives the initial data. Each piece of data is then transmitted to each neuron in this layer with specific weights. This weighted data is then propagated to the neurons in the next layer.

#### 3.8. Hidden layer:

The hidden layer of the neural network is responsible for learning the non-linear relationships between inputs and outputs. It performs complex transformations on the input data, enabling the network to recognise more sophisticated patterns and solve more complex problems.

### 3.9. Output layer:

The output layer of the perceptron is used to analyse the results and make decisions based on the input data and associated weights. It is also connected to an activation function that determines whether the output will be positive or negative depending on certain conditions.

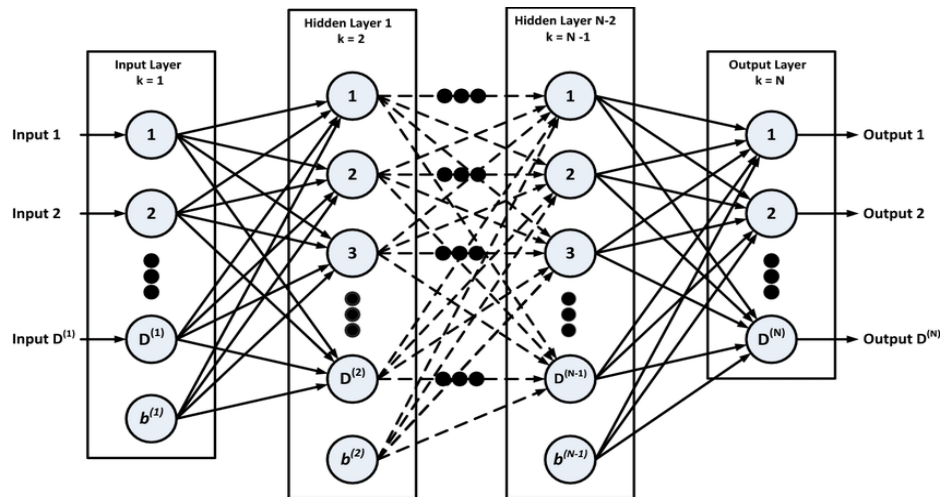


Figure 3.6: Multilayer perceptrons

## 4. =CONVOLUTIONAL NEURAL NETWORKS:

### 4.1. HISTORY OF THE CNN:

The convolutional neural network (CNN) was introduced in the 1980s by researcher Yann Le Cun. At the time, it was used to recognise handwritten characters. However, due to technological limitations at the time, CNN was not widely adopted. It wasn't until 2010, with the advent of multi-core processors, that deep learning and CNN really came into their own. These technological advances made it possible to solve the calculations required to train deep neural networks such as CNN.

### 4.2 PRINCIPLE OF CONVOLUTIONAL NEURAL NETWORKS:

The principle of convolutional neural networks (CNNs) is based on the use of convolution layers to extract relevant features from input data, such as images. These layers are designed to mimic the behaviour of neurons in the visual cortex of the human brain. They use filters or weight matrices that are applied sequentially to the input data, performing multiplication operations. This process allows convolution layers to learn and extract different features such as edges, textures and more complex patterns. The main advantage of convolution layers is their ability to capture local spatial dependencies in the input data.

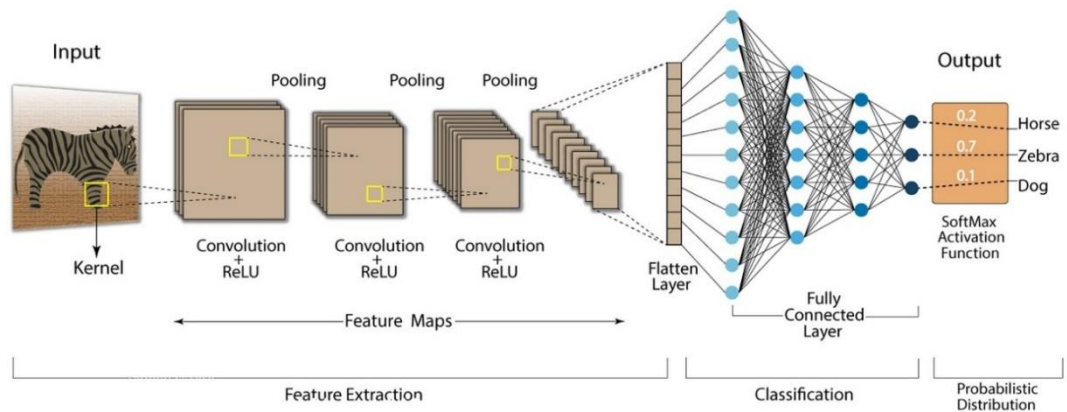


Figure 4 : Architecture of the CNN

### 4.3. ARCHITECTURE OF THE CONVOLUTIONAL NEURAL NETWORK (CNN):

A convolutional neural network generally consists of 4 layers:

- A Convolution layer
- A Pooling layer
- A RELU Correction layer
- A Fully-Connected layer.

#### a) Convolution layer :

Detects features using convolution filtering. The features searched for act as filters. Each is therefore defined by a convolution kernel. The image is filtered by each of the kernels. The convolution between the image and the filter produces a feature

map. The very bright regions of this map indicate the places in the image where the corresponding feature has been found. The first layer of a neural network is always a convolution layer. This type of layer is what makes convolutional neural networks so special.

### b) Pooling layer :

Used to apply a local maximum operation to feature maps. To do this, it divides a feature map into small windows of the same size, which can overlap. It retains only the maximum value in each window. The pooling layer therefore produces a smaller output feature map. This reduces the number of network parameters, and therefore avoids overlearning.

### c) Correction layer RELU :

Transforms negative values into 0, and does not modify positive values. It acts like an activation function.

$$\text{RELU}(x) = \max(0, x).$$

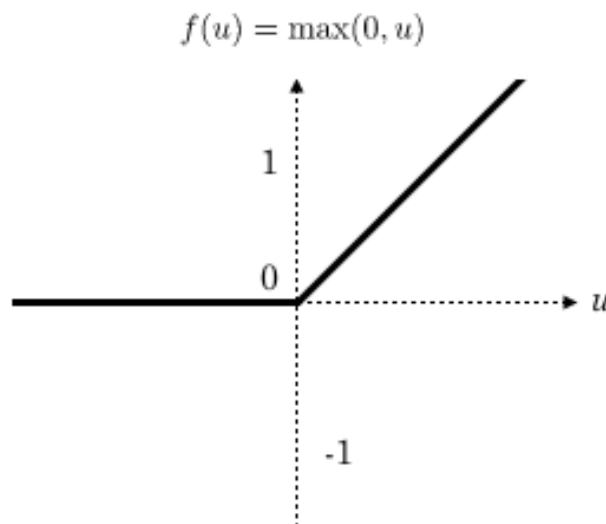


Figure 4.3: activation function

#### d. Fully-Connected layers :

Receives a vector as input, and modifies its values to produce a new one. Each node in the layer is connected to all the neurons in the previous layer, hence the name fully-connected. The last layer of a convolutional neural network is always of the fully-connected type: it receives a vector whose  $n$ th element indicates the probability that the input image belongs to the  $n$ th class. These probabilities are calculated using an activation function. In general, the logistic function is used for binary classification and the softmax function for multi-class classification. The network parameters are the features of the convolution layers and the weights of the fully-connected layers. These are determined during the network training phase using the gradient back-propagation method.

### 4.4. IMPACT OF AI ON MEDICAL IMAGING :

Artificial intelligence is having a significant impact on medical imaging. Convolutional neural networks (CNN) are widely used in this field for the detection and classification of different pathologies. They can be trained to recognize specific patterns in medical images, which helps doctors diagnose diseases such as cancer, cardiovascular disease and bone fractures. The use of artificial intelligence in medical imaging also speeds up the image analysis process and improves diagnostic accuracy. In summary, artificial intelligence has a major impact on medical imaging by improving the accuracy of diagnoses, enabling early detection of diseases and helping doctors in their decision-making.

### 5. CONCLUSION:

In conclusion, this chapter was devoted to presenting the concepts of Deep Learning (DL) with a focus on convolutional neural networks (CNN). We approached the general concept of Deep Learning by highlighting the emergence of CNNs. Furthermore, we examined the crucial role of CNNs in image processing, especially

in the field of medical imaging, and highlighted the significant impact of this method in the medical field.

# Chapter 2

## PATHOLOGY AND ORGAN CONCERNED

### 1. Introduction

In this chapter we will talk about the ocular organ in which we will see its function as well as its anatomy, then we will see diabetes and a complication linked to the latter Retinopathy which will be the subject of a diploma project.

#### 1.2. The Anatomy of the Eye:

The eye is the main sensory organ involved in vision, light waves are transmitted through the cornea and enter the eye through the pupil.

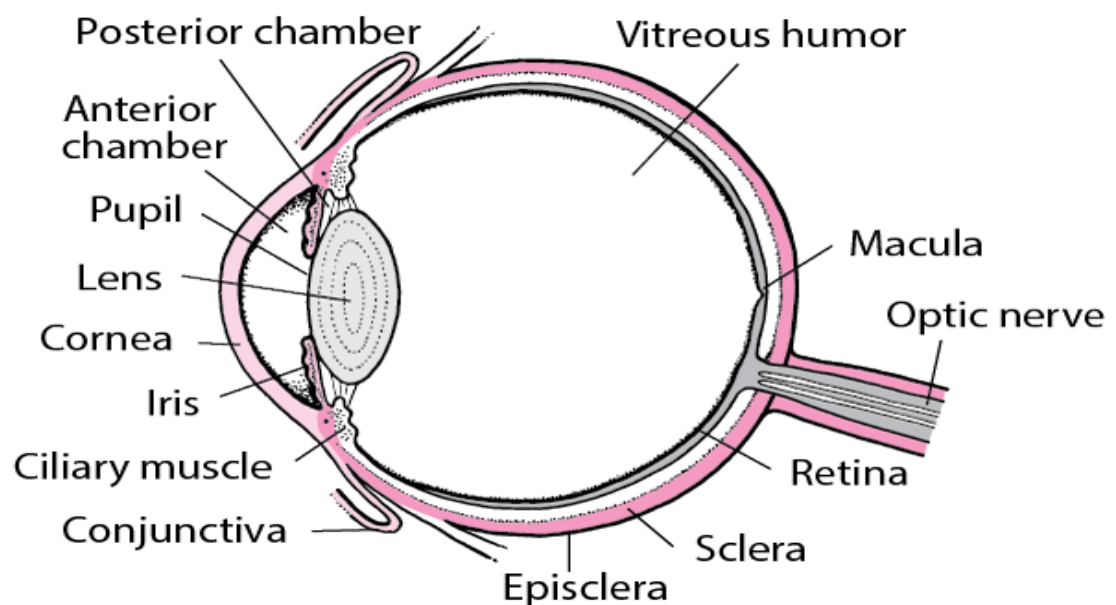


Figure1.2: Anatomy of the eye



## 1.2.1. Description of the eye:

According to the previous figure we can divide the eye into three parts:

### a) The external parts of the eye:

**The cornea:** is the transparent part located at the front of the eye which aims to protect the eye by helping to focus light entering the retina.

**The sclera:** is the white layer and resistance of the eye. Which aims to offer more protection and support to the internal structure of the eye.

**The conjunctiva:** is a thin layer of transparent tissue that covers the front surface of the eye and lines the inside of the eyelids. Its purpose is to keep the eyes moist and protect them from foreign particles.

### b) The internal visible parts:

**The iris:** Is the colored part of the eye. It helps control the size of the pupil, which is the opening that allows light to enter the eye. This allows the eye to adapt to different lighting conditions.

**The pupil :** Is the opening in the iris of the eye that allows light to enter. Its purpose is to control the amount of light that enters the eye by adjusting its size.

### c) Internal non-visible parts:

**The retina :** Is an innermost layer of the eye. It contains specialized cells called photoreceptors that convert light into electrical signals, which are then transmitted to the brain through the optic nerve. This is where the actual process of vision begins, as the brain interprets the electrical signals received from the retina and forms a visual image.

**The macula:** Is a small specialized area located in the center of the retina and responsible for detailed central vision. It allows you to see details, colors and participate in tasks such as reading, driving and recognizing faces.

**The optic nerve:** Is the link between the eye and the brain. It is through this optic nerve that the electrical signals are converted by the retina and transmitted to the brain to be interpreted as a visual image.

## 2. The Retina:

The retina is made up of 10 layers: 4 layers of visual cells which allow the reception of light, then 6 layers allowing its transmission to the brain by the optic nerve. The retina is made up of two parts: the central retina and the peripheral retina:

### 2.1 The central retina:

It is located at the center of vision, it includes the macula and the fovea of 5 to 6 millimeters in diameter, it is composed of photoreceptor cells called cones, which are specialized in the interpretation of shapes and colors in daytime vision. There are around 7 million.

### 2.3. The peripheral retina:

It extends over approximately 17 millimeters, it is more particularly made up of Rods, the cells are responsible for analyzing light. The peripheral retina is very essential in night vision.

## 3. RETINOPATHIES:

Retinopathy is a disease that affects the retina of the eyes and can lead to visual impairment. This part is the light-sensitive layer at the back of the eye. It is due to a complication of various diseases such as diabetes and hypertension. If not treated in time it can lead to loss of vision. This disease is caused by damage to the blood vessels of the retina which can lead to bleeding in the retina.

### 3.1 Diabetic Retinopathy:

Diabetic retinopathy is a disease that is linked to the common complication of diabetes that affects the eyes. Diabetic retinopathy is caused when high blood sugar levels damage the blood vessels of the retina. The retina plays a very important role in vision by converting light into electrical signals sent to the brain.

In a healthy eye the blood vessels of the retina function perfectly and provide the necessary nutrients and oxygen to the retinal cells. Unlike people with diabetes, excess sugar in the blood causes blood vessels to weaken and even close.

As these blood vessels weaken, they can cause swelling, bleeding from abnormal blood vessels. These signs of change can lead to loss of vision, or even a complete absence of vision if they are not treated. Diabetic retinopathy is the primary cause of blindness in adults with diabetes.

### 3.2 Hypertensive Retinopathy:

Hypertensive retinopathy is a disease that affects the blood vessels in the retina of the eye, it is caused by high blood pressure which puts excessive pressure on the blood vessels in the retina. It is often seen in people with uncontrolled or long-standing high blood pressure and can also occur in people with severe hypertension.

### 3.3 Symptom and Complication:

At the beginning of retinopathy we do not observe any symptoms at all. But as the disease progresses, symptoms appear. The symptoms of retinopathy are generally not painful and are similar in all cases. We can differentiate between the symptoms of diabetic retinopathy and hypertensive retinopathy.

#### a) symptoms of diabetic retinopathy

- Blurred vision
- Floating spot vision
- Suddenly green from vision
- Eye pain for patients whose cases are advanced
- See Floating Spots.

## b) Symptoms of Hypertensive Retinopathy

- Headache
- Change of Vision
- Double Vision

### 3.4. Diagnostic :

The diagnosis of diabetic retinopathy is based on examinations of the back of the eye. Color photography of the fundus helps categorize retinopathy. Fluorescein angiography will allow us to visualize poorly perfused areas. The earliest signs of retinopathy that ophthalmologists can detect is the formation of microaneurysms.

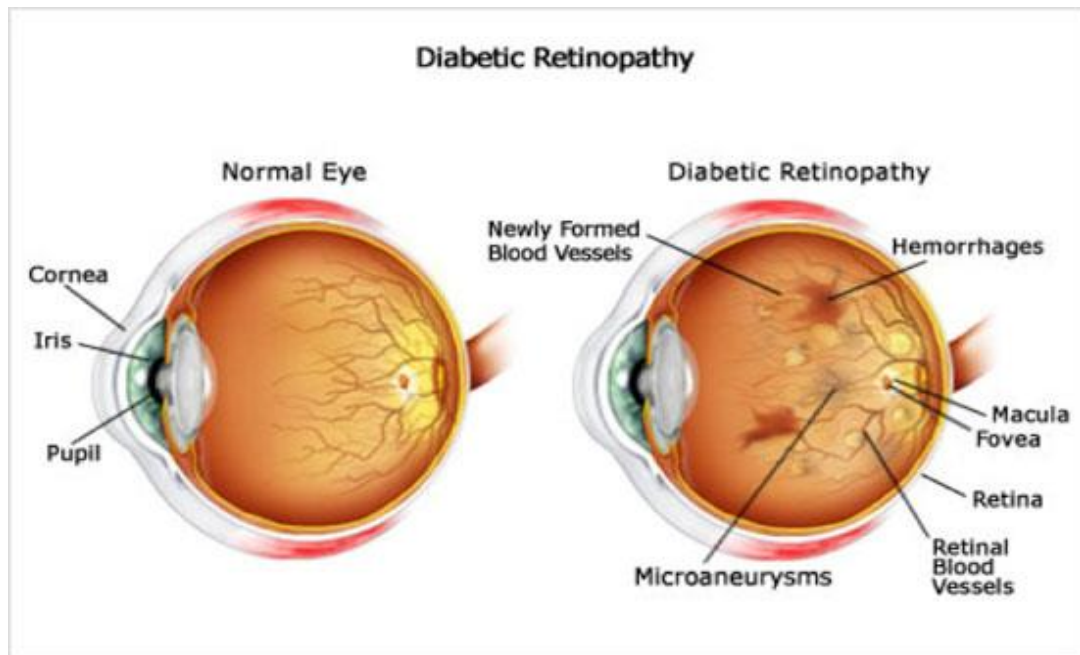


Figure3.4: the normal eye and the eye affected by diabetic retinopathy



Figure3.4: Photograph of the fundus of the eye affected by diabetic retinopathy

The diagnosis of hypertensive retinopathy is made like that of diabetic retinopathy. The doctor looks for tiny plaques of cholesterol in the blood vessels or by examining the retina using an ophthalmoscope from the back of the eye.



Figure3.4: Photograph of a retina affected by hypertensive retinopathy

### 3.5 Treatment and Prevention:

To prevent retinopathy we need to control blood sugar levels in people with diabetes. This will allow us to maintain blood sugar balance and reduce the risk of developing retinopathy. Maintaining balanced blood sugar levels requires eating a healthy diet, exercising regularly, and taking medications prescribed by a doctor. It is very

important to monitor blood pressure regularly because high blood pressure increases the risk of retinopathy.

### 3.6. To treat retinopathy we have two methods:

**The first:** consists of knowing the causes of retinopathy. If the retinopathy is diabetic then it is essential to do blood sugar tests to slow the progression of the disease. Or blood pressure control is very essential to prevent complications associated with retinopathy.

**The second:** consists of having a laser photocoagulation examination. This method aims to reduce blood leakage as well as prevent the formation of abnormal blood vessels. Photocoagulation can be used for the treatment of diabetic retinopathy and hypertensive retinopathy.

## 4. Conclusion:

In this chapter we have illustrated the anatomy of the eye to better understand its different parts then to better understand where the retina is located, we have further illustrated diabetic retinopathies its definition, causes of retinopathies, the different types of retinopathies, the symptoms, diagnosis, as well as treatment and prevention of diabetic retinopathy.

# Chapter 3

## DESIGN AND PRODUCTION

### 1. Introduction

In this chapter we will present the solution that we propose to respond to the problem linked to the early detection of diabetic retinopathy. Finally we will finish the theoretical part, then we will approach the technical part in which we will introduce the environment, the language as well as the platform used for detection. We will end with the presentation of the summary results.

### 2. Problem:

Our work will consist of implementing a medical diagnostic aid system which will detect the symptoms linked to diabetic retinopathy and classify them into two stages, starting from the normal eye stage to the I stage. terminal eye.

### 3. THE ARCHITECTURE OF THE CONVOLUTIONAL NEURAL NETWORK USED IN OUR PROJECT:

#### 3.1. RESNET :

Resnet or residual neural network is a deep neural network architecture that was introduced in 2015 by Microsoft researchers. Resnet is considered one of the most

significant advances in the field of deep learning. The main idea of Resnet is to solve the problem of gradient disappearance that occurs when training deep neural networks. The gradient disappearance problem leads neural networks to be unable to learn and improve its performance. This architecture relies mainly on residual learning, which consists of adding connections between layers of the network. These different shortcut connections between layers allow the network to learn residual functions, which make the differences between the input and output of the layers. By adding these residual functions, the network is able to learn more efficiently. We have been shown that Resnet produces cutting-edge results on various computer vision tasks, more specifically image classification, or object detection. The main advantage of Resnet is that it has the capacity to train very deep neural networks. For example, in traditional neural networks, performance decreases as the number of layers increases due to the problem of gradient disappearance. Unlike Resnet, it is capable of training neural networks with hundreds of layers without any reduction in performance and this has led to significant improvements in the accuracy of deep learning models. Resnet can be easily adapted to new spots in a dataset by adding precision to the weighting of the pre-trained network. In conclusion Resnet has become one of the fundamental architectures in the field of learning, inspiring new advances and extensions in the design of neural networks.

### 3.1.2. Architecture overview:

This ResNet architecture has an input layer with identity shortcuts. The left side shows us that the original block with batch normalization layers is pre-trained in two dimensions in light blue, the right side shows us that the block is modified where the batch normalization layers are replaced by identity layers in dark blue.



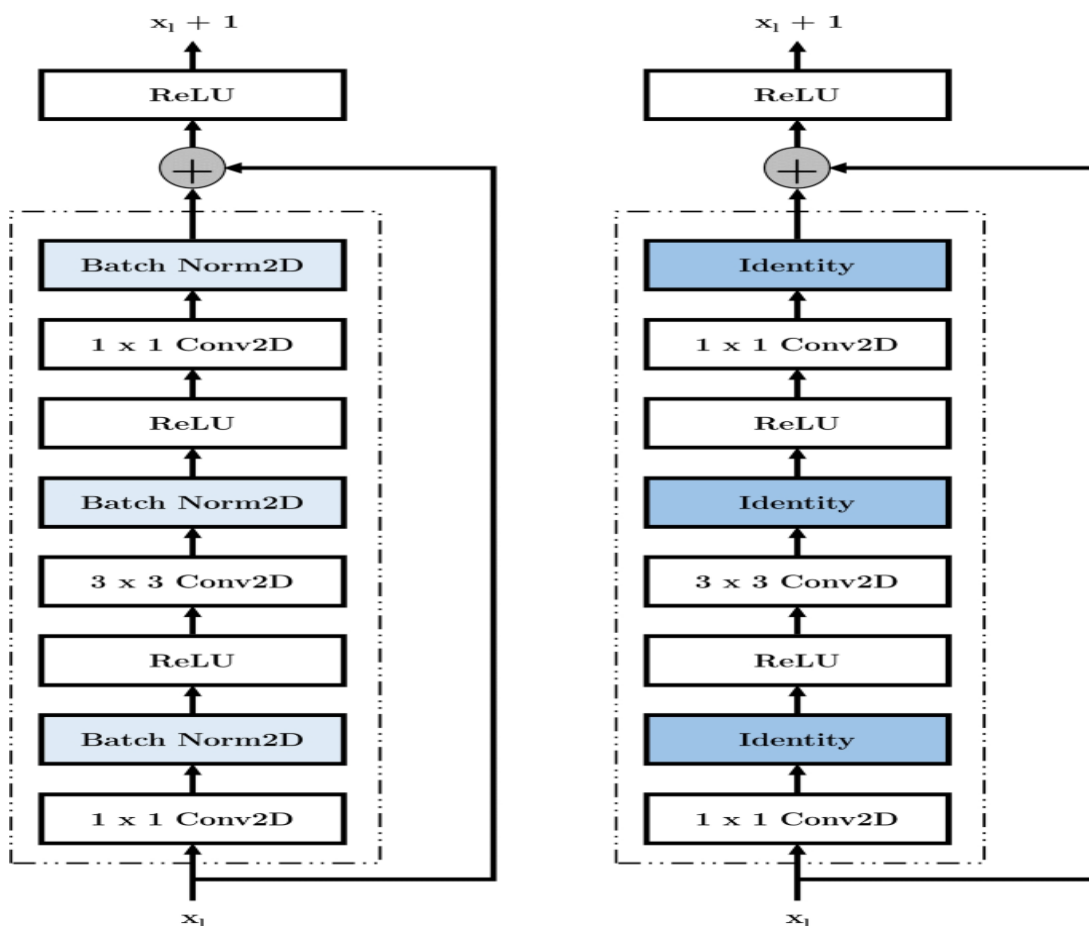


Figure3.1.2: Block Resnet with and without 1\*1

The network of this architecture is 34 layers and is inspired by the VGG-19 model which is used by Resnet which is followed by the addition of the shortened connection. This architecture is then transformed into a residual connection by these shortened connections. As shown in the following figure. In the figure below we see the VGG 19 model which is taken as a reference on the left, a flat network with 34 parameter layers in the middle and the residual network model with 34 parameter layers on the right. We see that the model has fewer filters and lower complexity than VGG networks. The 34 times model has 3.6 billion FLOPs (multiplication-addition), which is only 18% of VGG-19 (19.6 billion FLOPs).

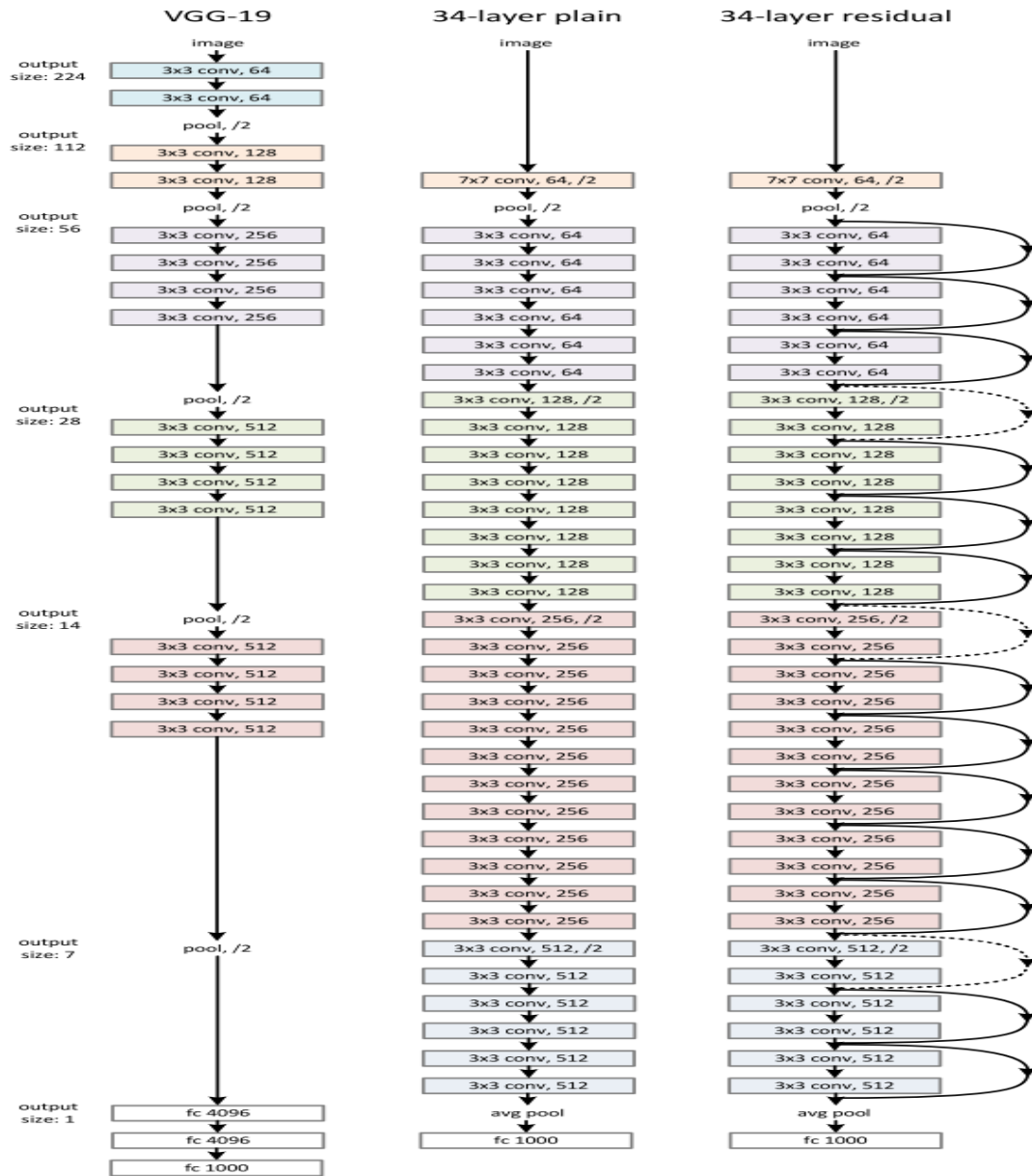


Figure 3.1.2: VGG19 and RESNET architecture

The straight line that transposes the input from layer  $x$  to the aggregation processes is called the residual link or shortcut link. Shortcut links are those that bypass one or more layers with blocks, inputs can now propagate more quickly via the connection between layers. Before we use a shortened connection, our input  $x$  must be multiplied by the weights of the layer and a term be added. Then we will go through the activation function  $f()$  and we obtain the result under  $H(x)$ .

$$H(\mathbf{x}) = f(\mathbf{w}\mathbf{x} + \mathbf{b}) \text{ or } H(\mathbf{x}) = f(\mathbf{x})$$

So in the end we will come out with a shortcut link like these:

$$H(\mathbf{x}) = f(\mathbf{x}) + \mathbf{x}$$

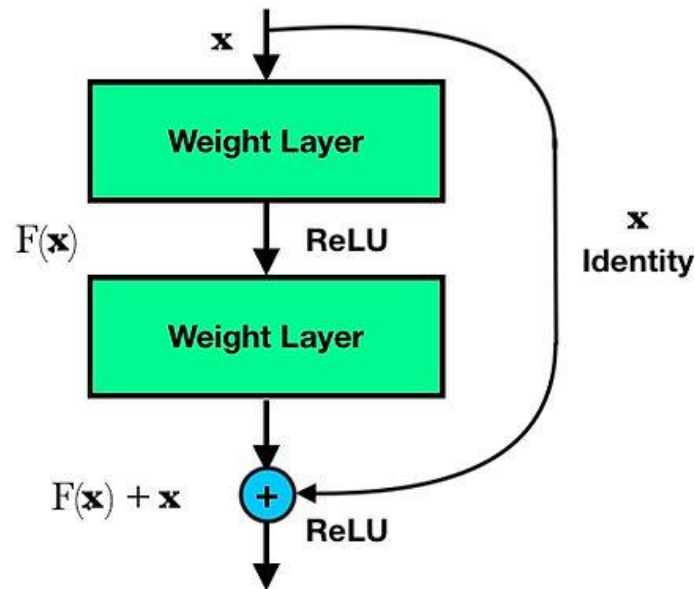


Figure31.2 : Residual Learning: a building block

### 3.1.3 InceptionResnet:

It is a deep architecture that combines the strengths of two popular convolutional neural network (CNN) models, Inception and Resnet. This architecture was introduced in 2015 by Google researchers. Its objective was to achieve both high precision and high calculation efficiency. Inception aims to capture features of different scales using parallel convolution layers with different filter sizes. This will allow the model to efficiently learn local and global features. By combining these two models Inception and Resnet are able to achieve cutting-edge performance in object classification and detection.

### 3.1.4 Inception Resnet Module:

This architecture introduces modules called Inception A, Inception B and Inception C which are designed to extract features at different spatial scales using various parallel operations.

### 3.1.5 Inception A:

In this set of convolution is  $1 \times 1$  followed by a  $1 \times 1$  convolution then another  $3 \times 3$ , then followed by a  $1 \times 1$  convolution, then followed by a  $3 \times 3$  convolution, at the end followed by a  $3 \times 3$  convolution, then at the output we will have parallel branches which are concatenated along the channel axes.

### 3.1.6 Inception B:

In this branch the convolution is  $1 \times 1$ , followed by a  $1 \times 7$  convolution, followed by a  $7 \times 1$  convolution, then another  $1 \times 7$  convolution and a  $7 \times 1$  convolution, then another  $1 \times 7$ , at the end of a convolution of  $7 \times 1$  then at the output we will have parallel and concatenated branches.

### 3.1.7 Inception C:

In this branch the convolution is  $1 \times 1$ , followed by a  $1 \times 3$  convolution, another layer of  $3 \times 1$ , then a  $1 \times 3$  convolution followed by another  $3 \times 1$ , then another  $1 \times 3$  convolution at the end of a  $3 \times 1$  convolution then at the output we will have parallel and concatenated branches.

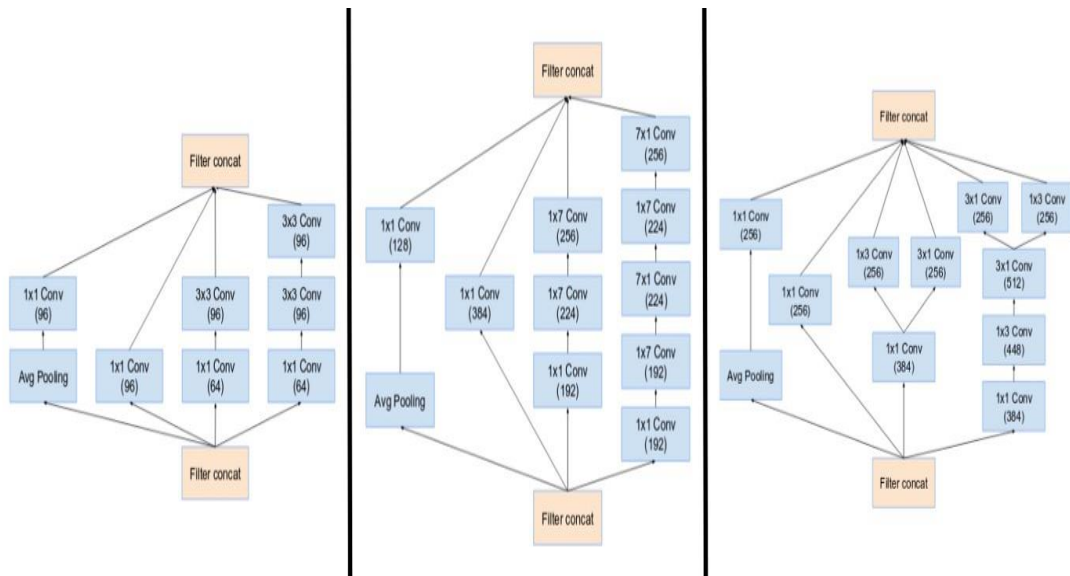


Figure 3.1.4: Block Inception A, B, C

### 3.1.8 Residual initiation block:

In each block Inception is followed by a 1x1 convolution without activation which is called filter expansion. This is done to increase the dimensionality of the filter bank, to match the input depth to the next layer. For the case of InceptionResnet, batch normalization is not used after summations. This is done to reduce the model size to make it trainable on a single GPU.

### 3.1.9 Residual Scaling:

In the case where our filter counts exceeded 1000, the residual variants started to exhibit instabilities, which would mean that the last layer before average pooling started producing only zeros after about ten thousand iterations. This cannot be avoided either by reducing the learning rate or by adding additional batch normalization to this layer. Scaling the residuals before adding them to the previous layer activation would seem to stabilize the training.

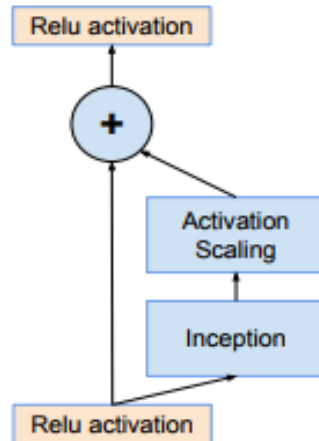


Figure3.1.9: Residual scaling

### 3.1.10 Reduction Block:

In this architecture our blocks A and B are used to reduce the spatial dimension of the characteristics between the different stages of the architecture

### 3.1.11 Block A:

In this block we have a 3x3 convolution with a larger stride to reduce the spatial size, then a 1x1 convolution which will allow us to adjust the number of feature channels, another 3x3 convolution which allows us to extract d The advantage of characteristics at the end is a max pooling operator whose role will be to extract dominant characteristics. The different combinations of operations help us to reduce the dimensions of the features.

### 3.1.12 Block B:

In this block we have a 1x1 convolution which will reduce the number of channels, then a 3x3 convolution with a larger stride to reduce the spatial size, then a second 3x3 convolution with a larger stride and the end we will have a last convolution of 1x1 to adjust dimensionality

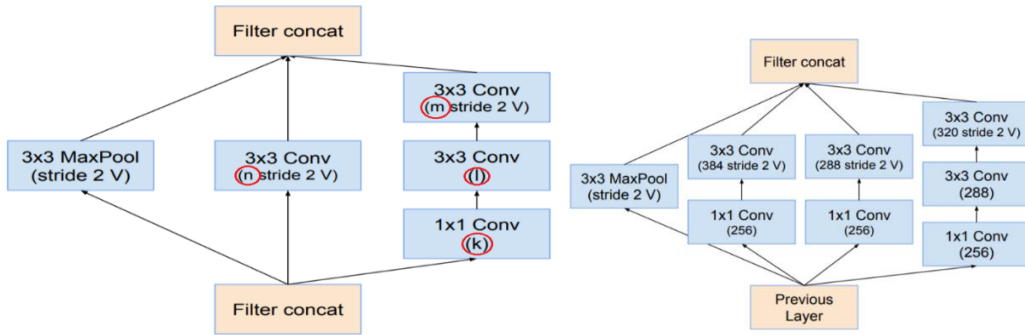


Figure 3.1.10 : Reduction Block A and Block B

### 3.1.13 Stem:

The stem is the first block of the inceptionResnetV2 architecture, it is composed of a series of convolution operations of different filter formats and thus sequential pooling. In addition, the stem plays a very important role in preprocessing the input to extract initial characteristics before passing them successively through the other inceptionResnet blocks.

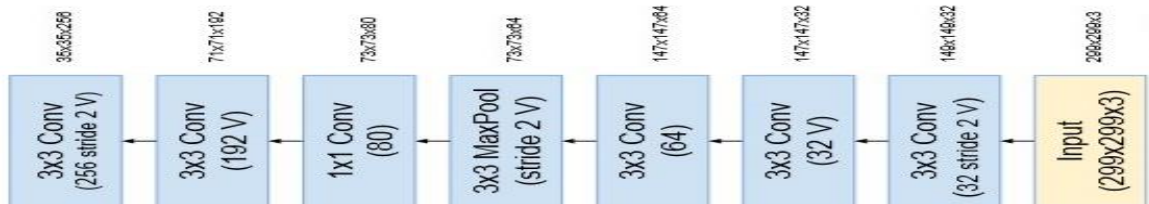


Figure3.1.13: block stem

### 3.1.14 Global Architecture of InceptionResnetV2:

We see in this figure the simplified structure of the InceptionResnetV2 neural network and its following components:

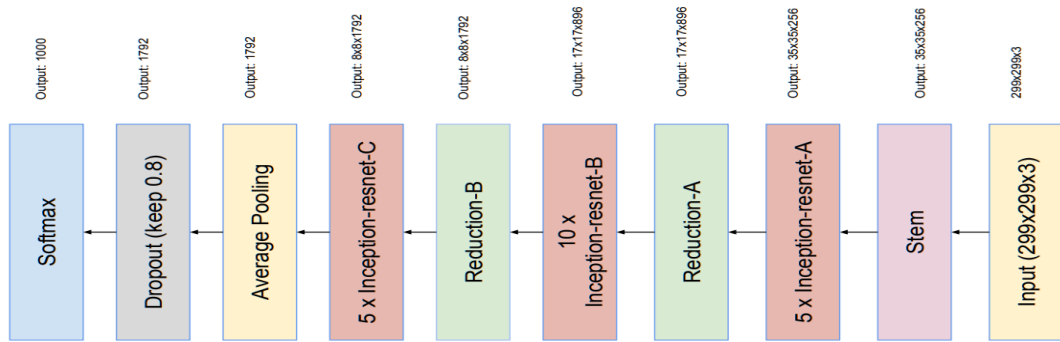


Figure3.1.14 : InceptionResnetV2

## 3.2. DESCRIPTION OF THE DATASET:

### 3.2.1. Dataset:

The dataset we used in this project was provided to us by kaggle. Kaggle is a community bringing together data scientists of all levels, which allows them to publish their work. Kaggle also organizes competitions offered by large companies where the winners will be paid. Our dataset includes 35251 images of average quality, it has been divided into 15 zip files due to its large size and it is accompanied by 2 csv files containing details of the data. Its fundus images were checked by an ophthalmologist specializing in the subject and then classified into 5 categories:

- a. No\_diabetic\_retinopathy
- b. Mild\_retinopathy
- c. Moderate\_retinopathy
- d. Severe\_retinopathy
- e. Proliferate\_retinopathy

In our case study we were able to use 5590 samples which we divided into 1 class. Our dataset was fragmented proportionally to 80% train set and 20% test set with random arrangements of classes within the train set and the test set.

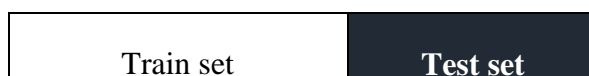


FIGURE 3.2.1: Data Fragmentation



### 3.2.2. Representation of our system:

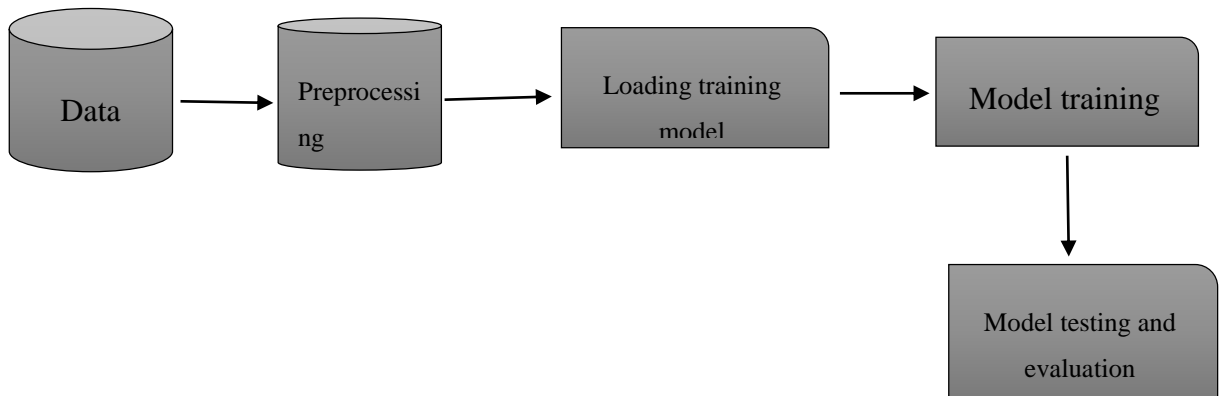


Figure3.2.2: Detection system diagram

### 3.2.3. Preprocessing:

This represents a crucial stage wherein raw data undergoes processing to render it usable for modeling purposes. In our specific context, our dataset comprises images capturing the fundus of the eye, and the processing steps applied include:

1. Converting image format from JPEG to PNG.
2. Aggregating all data into a single class.
3. Normalizing images by subtracting the minimum pixel intensity of each channel and dividing it by the average pixel intensity, thereby ensuring pixel values are represented within a range of 0 to 1.

### 3.2.4. Transfer Learning:

It is a technique that is used in machine learning in which a model that is trained on one task can be used for another task. In a more understandable term, it is a process of using a pre-trained model as a starting point for a new task that has less training data.

### 3.2.5. Data increase:

Data augmentation is a technique widely used in machine learning and computer vision that increases the size and diversity of a data set. The goal of data augmentation is to improve the generalization and robustness of machine learning models by exposing them to a wide range of data variation. For example in image classification data augmentation allows us to rotate an image by a certain degree, flip it horizontally or vertically and add random noise.

## 3.3. Realization and implementation:

Objective of this section is to present the tools as well as the technology that we used in the design of our system and the system code.

### 3.3.1. Python:

Python is a high-level programming language, which was created by Guido van Rossum and first published in 1991 in data analysis, artificial intelligence and scientific computing. Python is very versatile and can be used for various tasks including web development. Python is known for its simplicity and readability. Python is an interpreter language, which means it is usable in data analysis and artificial intelligence, it is gaining popularity among developers due to its libraries and scopes of its frameworks which make coding easier and more efficient.

### 3.3.2. Google Colab:

Is a product of Google Colab allows users to write and run python code in a web browser, eliminating the need to install and configure python on local computers or machines.

	GPU	Runtime	RAM	Disk Capacite
Colab Pro	T4 et P100	24h	25GB	200GB

Table 3.3.2: Google Colab Resources



Figure 3.3.2: Google Colab

### 3.3.3. Matplotlib:

Is a python data visualization library that is widely used for creating statistical and interactive visualizations in various formats. It was developed in 2003 by John D. Hunter to emulate MATLAB plotting.



Figure 3.3.3: matplotlib

### 3.3.4. Keras:

Is an open source deep learning library written in python it is a high-level deep neural network library capable of running on Tensorflow, Microsoft Cognitive and Toolkit. Keras is designed to enable rapid experimentation with deep neural networks, ultimately allowing developers to prototype and quickly develop neural network models.

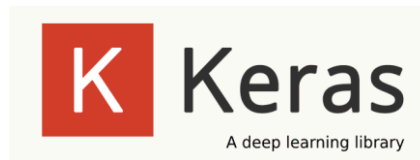


Figure 3.3.4: Keras

### 3.3.5. Tensorflow:

Is an open source software library developed by Google, It is a very powerful tool used to create and train machine learning models. Tensorflow allows developers to create and deploy machine learning models for tasks such as image recognition, natural language processing, and predictive analytics.



Figure 3.3.5: Tensorflow

### 3.3.6. Numpy:

Is a very popular python library widely used in scientific computing. The library supports large multidimensional arrays and matrices, as well as a wide range of mathematical functions for operating on arrays.



Figure 3.3.6: NumPy

### 3.3.7. Kaggle:

Is a popular online platform that hosts data science competitions and provides a space for data scientists and machine learning enthusiasts to collaborate, learn, and showcase

their skills It was founded in 2010 by Anthony Goldbloom and offers a diverse range of datasets and resources for people interested in data analysis and machine learning.



Figure 3.3.7: kaggle

### 3.3.8. Pandas:

Is a data manipulation tool that is widely used in the field of data sciences It is developed on the Python programming language and offers a variety of data and function structures that make manipulation and analysis much simpler and efficient.



Figure 3.3.8: Pandas

### 3.3.9. Code for programming our system:

```
1. from keras.preprocessing.image import ImageDataGenerator
2.
3. # Define the data generators for training and testing
4. train_datagen = ImageDataGenerator(rescale=1./255)
5. test_datagen = ImageDataGenerator(rescale=1./255)
6.
7. # Define the paths to the training and testing datasets
8. train_data_dir =
   '/content/drive/MyDrive/dataset_of_retinopathie/private'
9. test_data_dir =
   '/content/drive/MyDrive/dataset_of_retinopathie/private'
10.
11. # Define the batch size and target size for the images
12. batch_size = 32
13. target_size = (224, 224)
14.
15. # Define the train and test generators
16. train_generator = train_datagen.flow_from_directory(
17.     train_data_dir,
```

```

18.     target_size=target_size,
19.     batch_size=batch_size,
20.     class_mode='binary')
21.
22.     test_generator = test_datagen.flow_from_directory(
23.         test_data_dir,
24.         target_size=target_size,
25.         batch_size=batch_size,
26.         class_mode='binary')
27.     from tensorflow.keras.applications import ResNet50,
        InceptionV3
28.     from keras.layers import GlobalAveragePooling2D, Dense,
        Input
29.     from keras.models import Model
30.
31.     # Define the input shape for the models
32.     input_shape = (224, 224, 3)
33.
34.     # Define the Resnet50 model
35.     resnet50_input = Input(shape=input_shape)
36.     resnet50_model = ResNet50(include_top=False,
        weights='imagenet', input_tensor=resnet50_input)
37.     resnet50_output =
        GlobalAveragePooling2D()(resnet50_model.output)
38.     resnet50_fc = Dense(256,
        activation='relu')(resnet50_output)
39.     resnet50_pred = Dense(1,
        activation='sigmoid')(resnet50_fc)
40.     resnet50_model = Model(inputs=resnet50_input,
        outputs=resnet50_pred)
41.
42.     # Define the InceptionV3 model
43.     inceptionv3_input = Input(shape=input_shape)
44.     inceptionv3_model = InceptionV3(include_top=False,
        weights='imagenet', input_tensor=inceptionv3_input)
45.     inceptionv3_output =
        GlobalAveragePooling2D()(inceptionv3_model.output)
46.     inceptionv3_fc = Dense(256,
        activation='relu')(inceptionv3_output)
47.     inceptionv3_pred = Dense(1,
        activation='sigmoid')(inceptionv3_fc)
48.     inceptionv3_model = Model(inputs=inceptionv3_input,
        outputs=inceptionv3_pred)
49.     # Freeze the layers of the models
50.     for layer in resnet50_model.layers[:-3]:
51.         layer.trainable = False
52.     for layer in inceptionv3_model.layers[:-3]:
53.         layer.trainable = False

```

```

54. # Compile the models
55. resnet50_model.compile(loss='binary_crossentropy',
56. optimizer='adam', metrics=['accuracy'])
57. inceptionv3_model.compile(loss='binary_crossentropy',
58. optimizer='adam', metrics=['accuracy'])
59. # Define the paths to the training and testing datasets
60. train_data_dir =
61.     '/content/drive/MyDrive/dataset_of_retinopathie/private'
62. test_data_dir =
63.     '/content/drive/MyDrive/dataset_of_retinopathie/private'
64.
65. # Define the batch size and number of epochs for
66. training
67. batch_size = 32
68. epochs = 10
69.
70. # Define the train and test generators
71. train_generator = train_datagen.flow_from_directory(
72.     train_data_dir,
73.     target_size=target_size,
74.     batch_size=batch_size,
75.     class_mode='binary')
76.
77. test_generator = test_datagen.flow_from_directory(
78.     test_data_dir,
79.     target_size=target_size,
80.     batch_size=batch_size,
81.     class_mode='binary')
82.
83. # Train the models
84. resnet50_history = resnet50_model.fit(
85.     train_generator,
86.     epochs=epochs,
87.     validation_data=test_generator)
88.
89. inceptionv3_history = inceptionv3_model.fit(
90.     train_generator,
91.     epochs=epochs,
92.     validation_data=test_generator)
93.
94. # Evaluate the models on the testing dataset
95. resnet50_score = resnet50_model.evaluate(test_generator)
96. inceptionv3_score =
97.     inceptionv3_model.evaluate(test_generator)
98.
99. from sklearn.metrics import confusion_matrix, roc_curve,
100. auc
101.
102. # Generate the confusion matrix

```

```

94.     resnet50_y_true, resnet50_y_pred =
        train_generator.classes,
        resnet50_model.predict(test_generator)
95.     resnet50_cm = confusion_matrix(test_generator.classes,
        resnet50_y_pred.round())
96.
97.     inceptionv3_y_true, inceptionv3_y_pred =
        train_generator.classes,
        inceptionv3_model.predict(test_generator)
98.     inceptionv3_cm =
        confusion_matrix(test_generator.classes,
        inceptionv3_y_pred.round())
99.
100.    #Generate the ROC curve
101.    resnet50_fpr, resnet50_tpr, _ =
        roc_curve(test_generator.classes,
        resnet50_y_pred.round()[:, 0])
102.    resnet50_roc_auc = auc(resnet50_fpr, resnet50_tpr)
103.
104.    inceptionv3_fpr, inceptionv3_tpr, _ =
        roc_curve(test_generator.classes,
        inceptionv3_y_pred.round()[:, 0])
105.    inceptionv3_roc_auc = auc(inceptionv3_fpr,
        inceptionv3_tpr)
106.    from sklearn.metrics import confusion_matrix
107.
108.    def specificity(y_true, y_pred):
109.        tn, fp, fn, tp = confusion_matrix(y_true,
        y_pred).ravel()
110.        specificity_val = tn / (tn + fp)
111.        return specificity_val
112.
113.    # Reste du code...
114.
115.    # Generate sensitivity, specificity, precision, and F1-
        score
116.    resnet50_sensitivity =
        recall_score(test_generator.classes,
        resnet50_y_pred.round()[:, 0])
117.    resnet50_specificity =
        specificity(test_generator.classes,
        resnet50_y_pred.round()[:, 0])
118.    resnet50_precision =
        precision_score(test_generator.classes,
        resnet50_y_pred.round()[:, 0])
119.    resnet50_f1 = f1_score(test_generator.classes,
        resnet50_y_pred.round()[:, 0])
120.

```



```

121. inceptionv3_sensitivity =
    recall_score(test_generator.classes,
                 inceptionv3_y_pred.round()[:, 0])
122. inceptionv3_specificity =
    specificity(test_generator.classes,
               inceptionv3_y_pred.round()[:, 0])
123. inceptionv3_precision =
    precision_score(test_generator.classes,
                   inceptionv3_y_pred.round()[:, 0])
124. inceptionv3_f1 = f1_score(test_generator.classes,
                              inceptionv3_y_pred.round()[:, 0])
125.
126. # Display the results
127. print("Resnet50 Metrics:")
128. print(f"Sensitivity: {resnet50_sensitivity:.4f}")
129. print(f"Specificity: {resnet50_specificity:.4f}")
130. print(f"Precision: {resnet50_precision:.4f}")
131. print(f"F1-score: {resnet50_f1:.4f}")
132.
133. print("InceptionV3 Metrics:")
134. print(f"Sensitivity: {inceptionv3_sensitivity:.4f}")
135. print(f"Specificity: {inceptionv3_specificity:.4f}")
136. print(f"Precision: {inceptionv3_precision:.4f}")
137. print(f"F1-score: {inceptionv3_f1:.4f}")
138. import matplotlib.pyplot as plt
139.
140. # Display the confusion matrix
141. plt.figure(figsize=(5,5))
142. plt.imshow(resnet50_cm, cmap='Blues')
143. plt.title('Resnet50 Confusion Matrix')
144. plt.xlabel('Predicted')
145. plt.ylabel('True')
146. plt.colorbar()
147. plt.xticks([0, 1], ['No DR', 'DR'], rotation=45)
148. plt.yticks([0, 1], ['No DR', 'DR'])
149. plt.show()
150.
151. plt.figure(figsize=(5,5))
152. plt.imshow(inceptionv3_cm, cmap='Blues')
153. plt.title('InceptionV3 Confusion Matrix')
154. plt.xlabel('Predicted')
155. plt.ylabel('True')
156. plt.colorbar()
157. plt.xticks([0, 1], ['No DR', 'DR'], rotation=45)
158. plt.yticks([0, 1], ['No DR', 'DR'])
159. plt.show()
160.
161. # Display the ROC curve

```

```

162. plt.figure(figsize=(5,5))
163. plt.plot(resnet50_fpr, resnet50_tpr, label='Resnet50 ROC
      curve (area = {:.2f})'.format(resnet50_roc_auc))
164. plt.plot(inceptionv3_fpr, inceptionv3_tpr,
      label='InceptionV3 ROC curve (area =
      {:.2f})'.format(inceptionv3_roc_auc))
165. plt.plot([0, 1], [0, 1], 'k--')
166. plt.xlim([0.0, 1.0])
167. plt.ylim([0.0, 1.05])
168. plt.xlabel('False Positive Rate')
169. plt.ylabel('True Positive Rate')
170. plt.title('Receiver Operating Characteristic')
171. plt.legend(loc="lower right")
172. plt.show()
173.
174. # Display the sensitivity and specificity graphs
175. plt.figure(figsize=(5,5))
176. plt.plot([0, 1], [resnet50_sensitivity,
      resnet50_sensitivity], label='Resnet50 Sensitivity')
177. plt.plot([0, 1], [inceptionv3_sensitivity,
      inceptionv3_sensitivity], label='InceptionV3 Sensitivity')
178. plt.plot([resnet50_specificity, resnet50_specificity],
      [0, 1], label='Resnet50 Specificity')
179. plt.plot([inceptionv3_specificity,
      inceptionv3_specificity], [0, 1], label='InceptionV3
      Specificity')
180. plt.xlim([0.0, 1.0])
181. plt.ylim([0.0, 1.05])
182. plt.xlabel('False Positive Rate')
183. plt.ylabel('True Positive Rate')
184. plt.title('Sensitivity and Specificity')
185. plt.legend(loc="lower right")
186. plt.show()

```

## 4. Conclusion:

In this chapter we presented the different stages of concepts and realization of our system for early detection of our diabetic retinopathy, then we discussed the concepts of the architectures and the programming codes used in our system.

In the next chapter we will present the last part of our end-of-study project which is devoted to the presentation of the results obtained.

# Chapter 4

## TESTING THE RESULTS

### 1. Introduction

In this chapter, we will present the results of our approach based on the use of convolutional neural networks with the architecture proposed by Resnet and Inception with the aim of detecting the presence of diabetic retinopathy (DR).

We will begin to present the results followed by an evaluation of the results according to the criteria linked to the problem.

### 2. Choice of evaluation criteria:

The reasons for our field in which we wish to develop our model, we will take into account metrics which better respect the constraints place in the medical field, we will choose on sensitivity and specificity.

For this we submitted our model to a series of tests in order to evaluate our model then we will compare the results, and for this we will take the following measurements each time.

1) Sensitivity: otherwise called recall, it is a detection model that measures the ability to correctly identify all positive cases of a DR among a group of diabetic patients. This measure is crucial in screening for DR because early detection is essential to prevent potential complications associated with this disease.

2) Specificity: is the ability of a model to correctly identify eyes that do not show signs of DR among all eyes that are actually healthy.

When specificity is high this means that the model minimizes false positives, i.e. it correctly identifies most, or even all, healthy eyes as such.

### 3 Comparative study between Resnet50 and InceptionV3:

In this section, we will make a comparative analysis of classification performance using two different neural networks: Resnet and Inception. Indeed we will study the effect of hyper-parameters on the classification results.

#### 3.1. Configuration of our model parameter:

For the configuration of our Resnet and Inception model we have the following parameters:

Parameters Constants:

- Structure of the neuron: the number of layers and neurons in each layer remains constant throughout the study.
- Data distribution: the data is separated into two parts, the train set (80%) and test set (20%) which remain identical throughout the experiment.

Parameter Variables: Input image resolution, Dropout layer, Optimization algorithm, Number of Iterations and Batch size.

#### 3.2. Evaluation of Resnet and Inception:

Model Name	Learning Rate	Dropout Layer	Model input resolution	Optimization Algorithm
Resnet			(224, 224, 3)	Adam
Inception			(224, 224, 3)	Adam

Table3.2: Resnet and Inception model configuration

The number of iterations was fixed at 10 (epochs = 10), with a Batch\_Size of 32 for the train set and the test set.

Results Obtained:

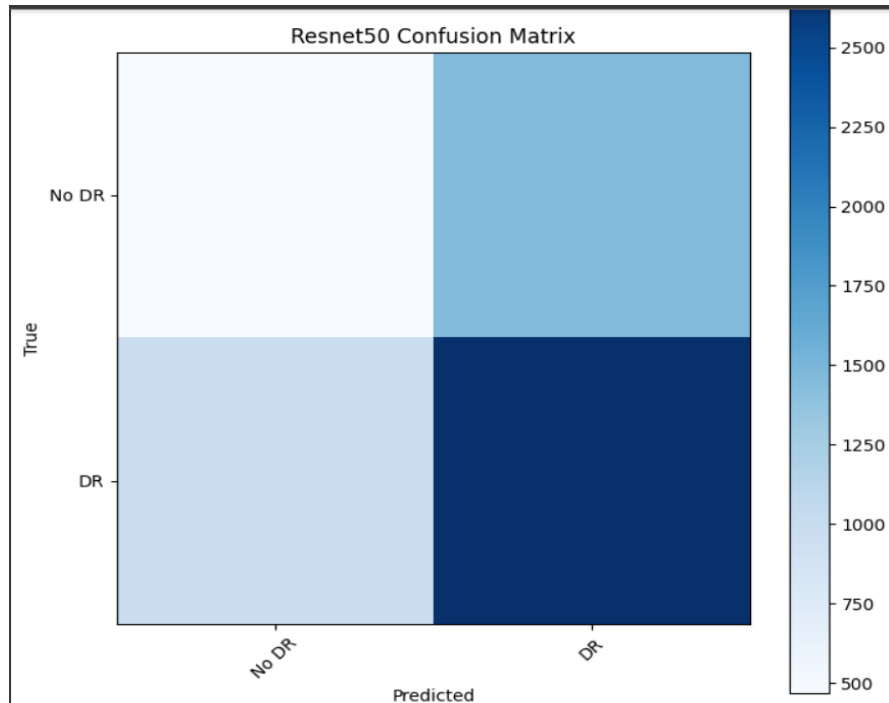


Table: Resnet50 Confusion Matrix

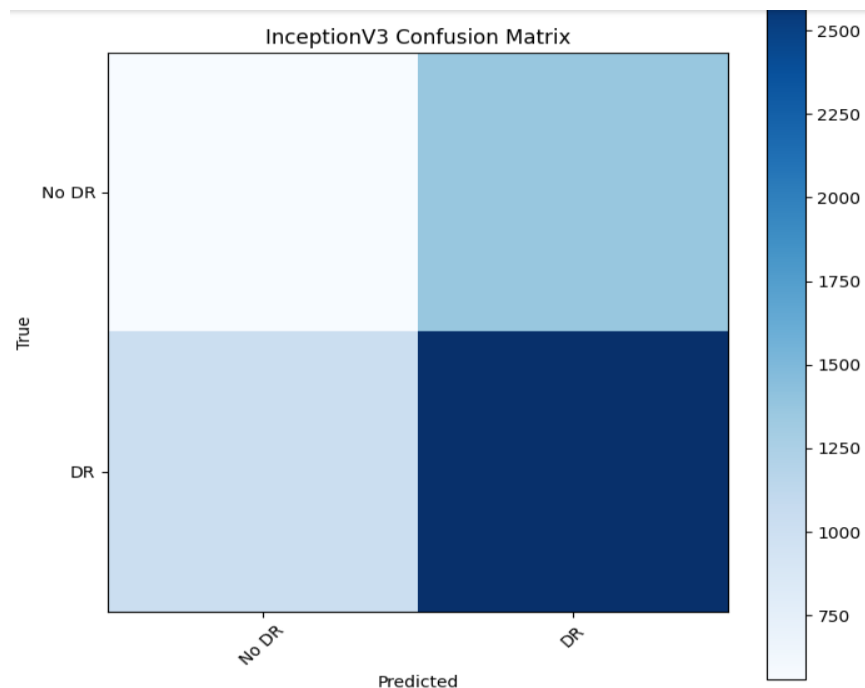


Table: InceptionV3 Confusion Matrix

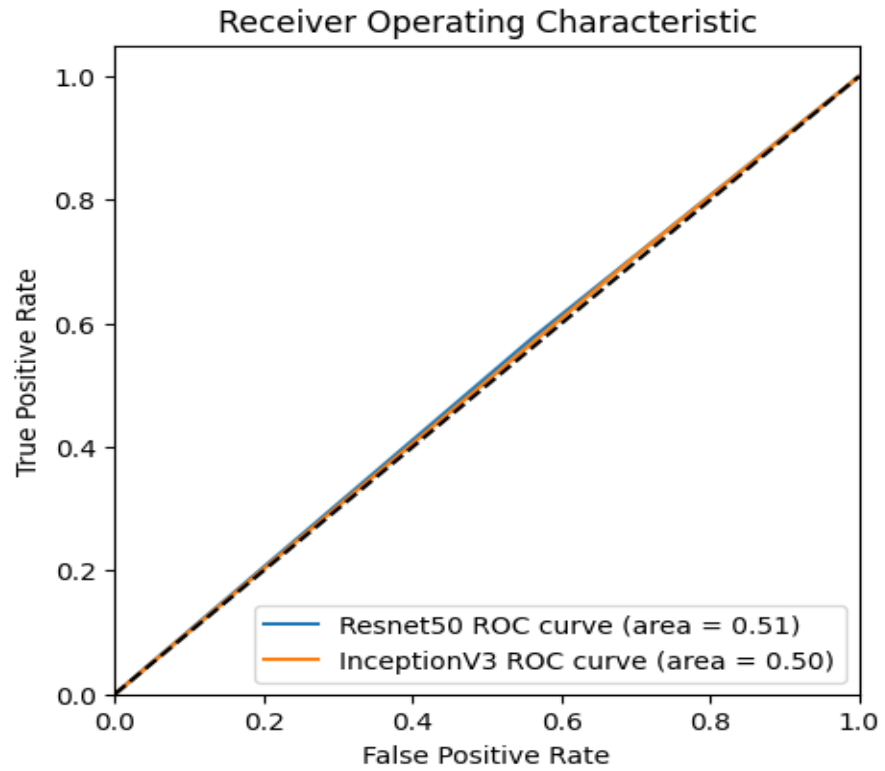


Figure 3.2: Receiver Operating Characteristic

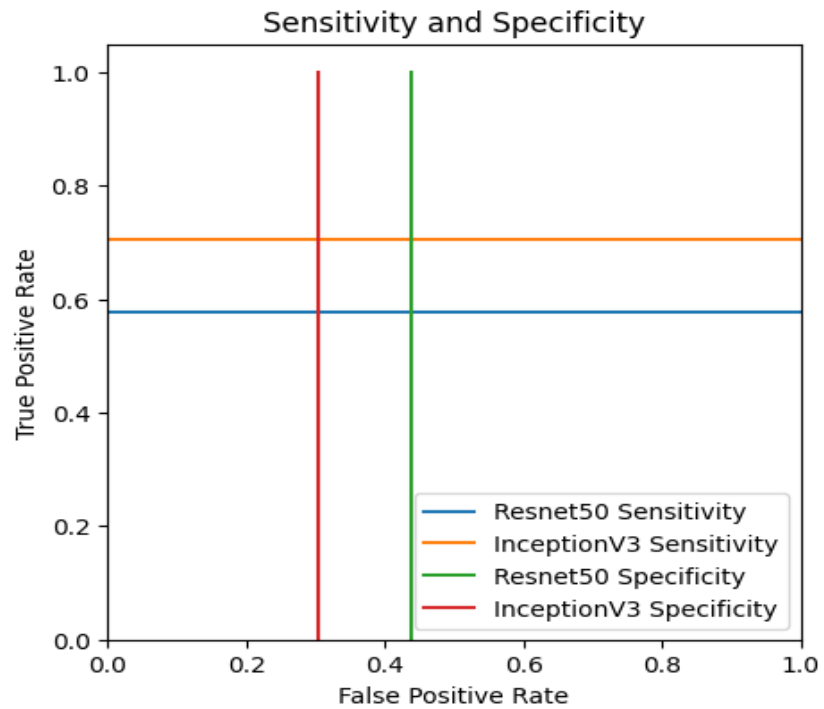


Figure 3.2 : Sensitivity and Specificity

Results obtained on Resnet50 and Inception metrics

Sensitivity	0.57
Specificity	0.43
Precision	0.66
F1-Score	0.61

Table: Resnet50 metrics

Sensitivity	0.57
Specificity	0.43
Precision	0.66
F1-Score	0.61

Table: Inception metrics

#### 4. conclusion:

In this chapter, we introduced the appropriate measures to evaluate the effectiveness of our medical diagnosis support system, in particular sensitivity and specificity. We performed a series of benchmark tests using various fixed parameters for both models, Resnet and Inception. In conclusion we presented the results obtained.

# Resources

## Magazine article examples::

<https://suhedacilek.medium.com/resnet-residual-network-nedir-49105e642566>

<https://www.msmanuals.com/fr/accueil/troubles-oculaires/maladies-de-la-r%C3%A9tine/r%C3%A9tinopathie-diab%C3%A9tique>

<https://www.retinegallien.com/retinopathie-diabetique.php>

<https://www.dinnosante.fr/comprendre-le-diabete/retinopathie-diabetique>

<https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202>

<https://medium.com/swlh/understanding-inception-simplifying-the-network-architecture-54cd31d38949>

<https://www.kaggle.com/competitions/diabetic-retinopathy-detection>

## Book examples:

Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2E  
ISBN 9781492032649

2019 Aurelien Geron.



# Özgeçmiş

Name and surname: Mamadou Lamarana Diallo

## Education::

2023–2024 İzmir Kâtip Çelebi Üniversitesi, yazılım Müh.

2021–2023 İzmir Kâtip Çelebi Üniversitesi, yazılım Müh. Bölümü

## Work Experience:

2019 – 2024 computer engineer Guinean Army