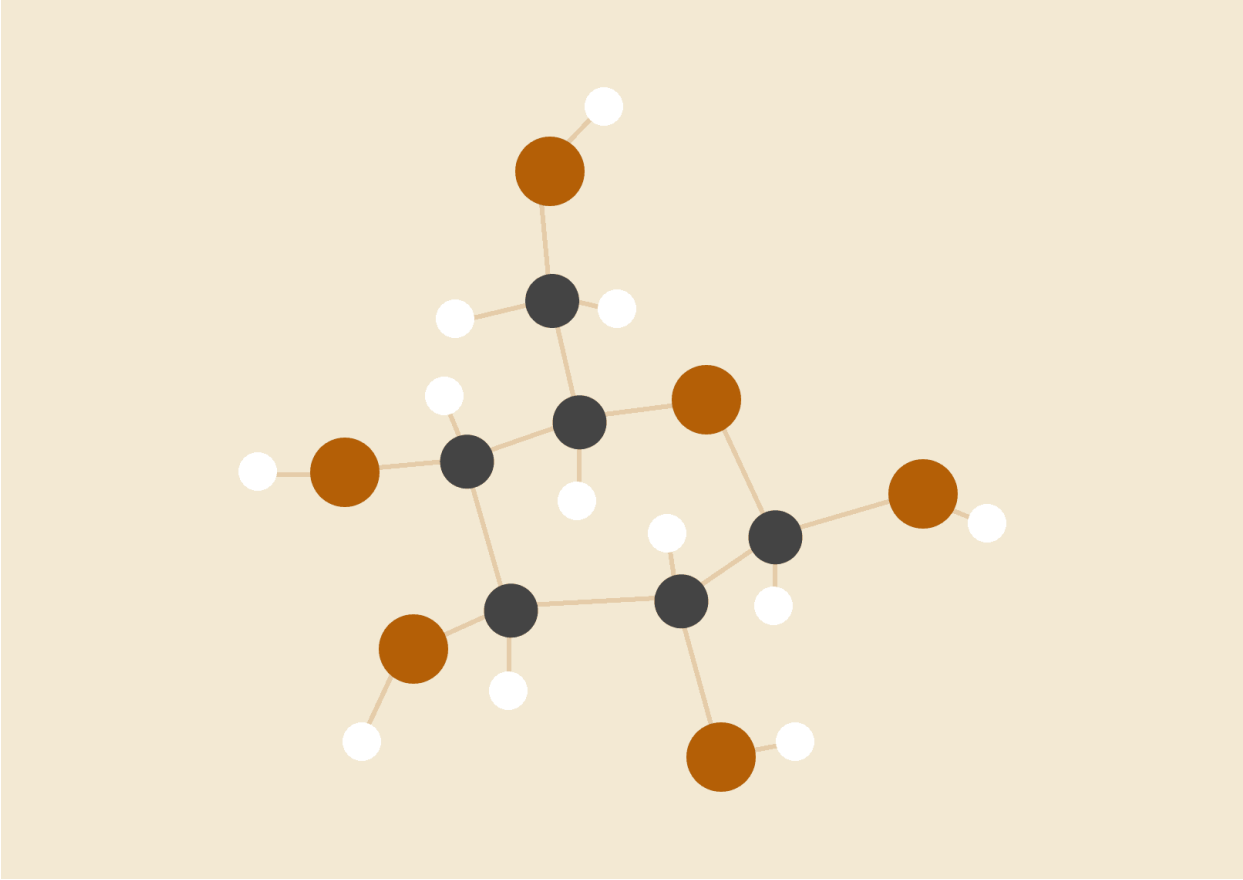
ZOIDBERG2.0 SYNTHESIS

Machine learning to detect pneumonia



Johan-David Dehi
Pierre Bobard
Gregory Koenig
Richard La
Maxime Lardier

05/09/2021

IA – MSc Pro 1

TABLE OF CONTENTS

INTRODUCTION

The aim of this project is to help doctors detect pneumonia by using machine learning.

DATASET

A dataset is a collection of data.

To achieve this task, doctors granted us access to three datasets of chest X-ray images. Each dataset is composed of two folders : one for normal patients, one for pneumonia patients.

We have around 5,500 chest X-rays images composing our dataset. It consists of three subcategories (test, train and val).

The subdirectories have been separated into three subdirectories each, all named “BACTERIA”, “NORMAL” and “VIRUS”, so that the three subcategories (test, train and val) would all have samples of the dataset.

MACHINE LEARNING

The process of learning is how AI (Artificial Intelligence) systems can make decisions. They can adapt their behaviour based on their experiences.

There are three types of learning:

- **Reinforcement learning:** It is the process of learning in an environment through feedback from an AI’s behaviour. It is how most human beings learn: no one tells them how, they just practice, stumble and get better until they manage to do a specific task.
- **Unsupervised learning:** It is the process of learning without training labels. It could also be called clustering or grouping. For instance, it can be used to find patterns in the frames of a video and compress those frames so that videos can be streamed to viewers quickly.
- **Supervised learning:** It is the process of learning with training labels. It is the most widely used kind of learning when it comes to AI. Supervised learning is when someone who knows the right answers, called a supervisor, points out mistakes during the learning process.

Here we use supervised learning and a CNN (Convolutional Neural Network), which is a class of deep learning. Deep learning represents the use of multiple layers in a network.

A CNN is inspired by the animal visual cortex, where neurons are connected to one another through layers. CNNs analyze data and assemble patterns, optimizing kernels through automated learning.

PYTHON

Python is an interpreted high-level general-purpose programming language.

Python is the most popular programming language used for AI. It's easy to understand, which makes data validation quick and practically error-free. It’s also more intuitive than other languages.

TENSORFLOW

TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

We decided to use TensorFlow because:

- It is very well documented
- It scales up into production
- It is being able to use many PUs (Processing Unit) or Google TPUs (Tensor Processing Unit)
- It allows flexible creation of deep learning architecture using basic building blocks
- It is supported by Google
- It has a very large following and a strong community

KERAS

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. It’s a high-level neural networks library that is running on top of it.

We turned to Keras because it's:

- User-friendly
- Modular and composable
- Easy to extend
- Easy to use

SCIKIT-LEARN

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

We have chosen Scikit-learn because it fits perfectly for supervised learning and it has various methods to check the accuracy for cross-validation. Cross-validation is used to estimate the accuracy of machine learning models and to protect them against overfitting.

OTHER LIBRARIES

Here other specific libraries were used with the Python programming language:

- Matplotlib: generate plots and graphs from data
- NumPy: support for multidimensional arrays and matrices
- OpenCV: computer vision and image processing
- PIL: Python image library
- Seaborn: statistical data visualization
- Argparse: parser for command-line arguments
- Random: generating and handling random values
- Pickle: serialization of objects
- Sys: Python parameters and system methods
- OS: Operation system library

PROCEDURE

To work collaboratively, we mainly use Google Colab. Google Colab is a cloud service for training an AI model without having to install anything on a local machine. It allows us to use Jupyter Notebook which is a web application containing texts and executable blocks of Python code. Google Colab also has the great advantage of running on GPUs and TPUs. TPUs stands for Tensor Processing Units, developed by Google specifically for neural network machine learning, particularly using Google's own TensorFlow software. It's much faster for training our AI model than CPUs or even GPUs.

Designing the model

Designing the model is one of the first development steps.

Deep learning and artificial neural networks work with layers: input, hidden and output. The input layer in our case is the dimension of our images. We use resizing, so 64x64 would mean 4096 input neurons. We use the Conv2D layer in our sequential model. The output layer represents the number of predictions we need, in our case three. Since we have a multi-class classification, we also use the softmax activation function.

Our model has 8 hidden layers, between the Conv2D input layer and the Dense output layer with a softmax activation function.

Training the model

To train our model, we feed it our dataset. Not the complete dataset, but parts of it.

We parse our subdirectories and get the path and the image which we append to different arrays. We resize and reshape our arrays to create multidimensional arrays. The train_test_split method is then used to split our dataset into two categories, one used for the training and the other for the test. To sharpen our model, we use the ImageDataGenerator method, which allows us to flip, shift or even change the brightness of every image.

We finally compile our model with categorical cross entropy, to measure the loss. We use the model.fit method with a specific number of iterations, also called epochs, and a specific quantity of samples from the dataset, also called batch size. We also split a part of the data to use it for validation.

Interpreting results

We use Scikit-learn metrics to measure and interpret the results.

The classification report gives us accuracy measurements to classify images, from a scale of 0 to 1.

Presenting useful reports to analyze results

Thanks to Scikit-learn and Matplotlib, we generate graphs, both showing the training and validation accuracy and loss.

We also generate a confusion matrix, which tells us how accurate the model predictions were.

Updating the model based on previous results

The goal is to have an accuracy as close to 100% as possible and loss as close to 0% as possible.

After analyzing the previous results, we can update our model (different layers, different activations), our training (image resizing, ImageDataGenerator function) or our parameters (epochs, batch size, optimizer).

Updating the model enables us to learn about overfitting and other limits of our settings. By trying and testing, we can increase our accuracy and reduce our loss.

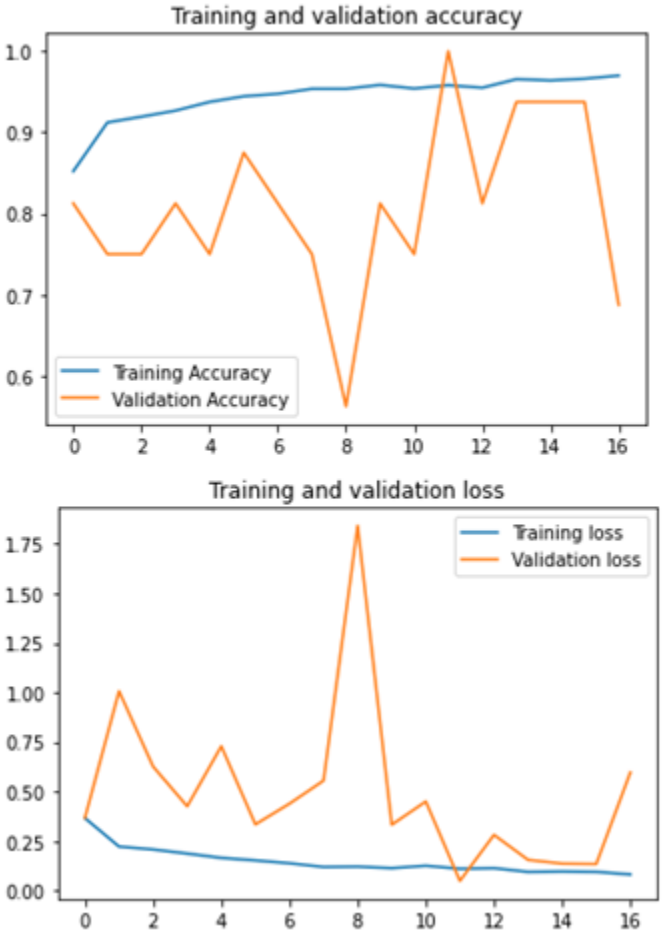
Using the trained model to make predictions

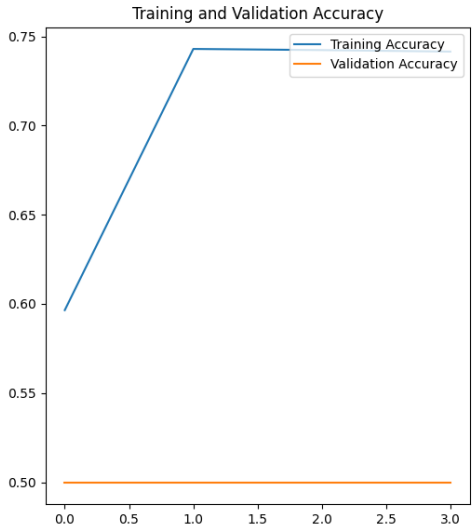
The real aim is here: predicting with the highest precision the category of an image. Here we must classify a generic chest X-ray between three different categories.

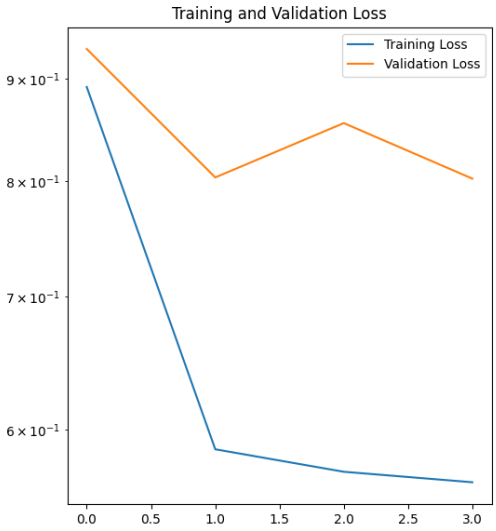
We use our trained model with specific arguments to make predictions on new data, and then return the data with a prediction and its accuracy as a percentage.

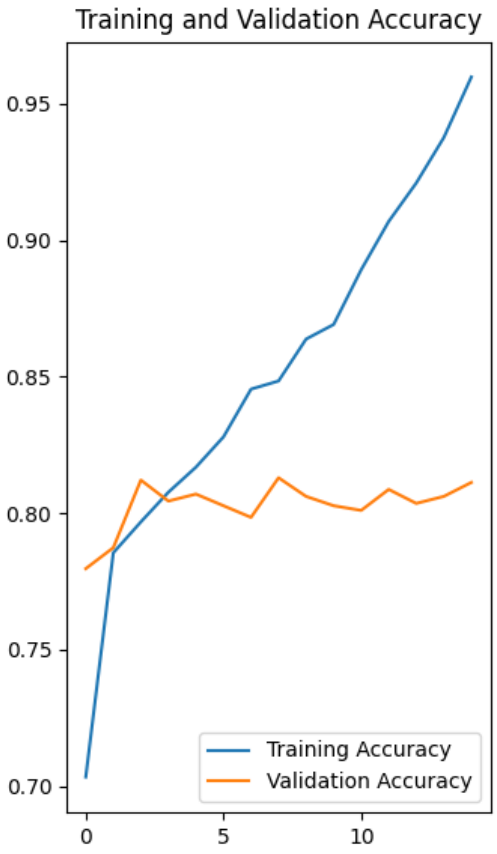
RESULTS

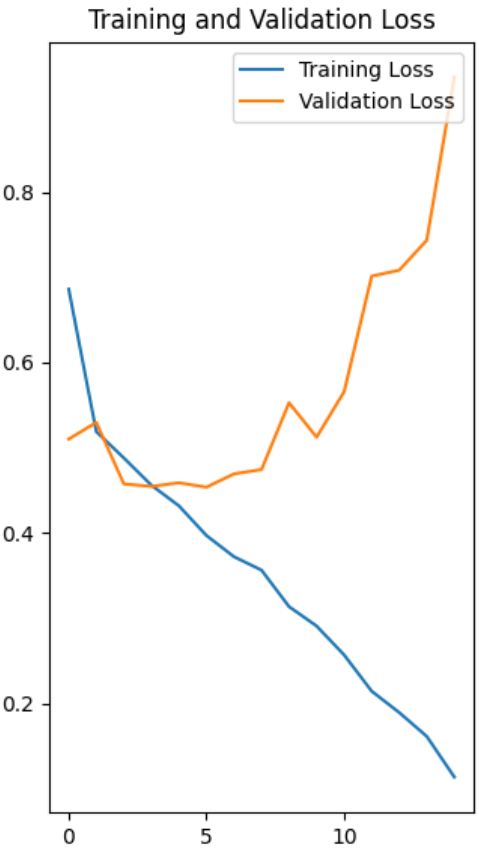
During the training process of our AI model, we kept improving again and again. We fumbled by changing the different parameters at random. Thanks to this, we began to intuitively understand how to orient our model in order to improve its prediction accuracy.



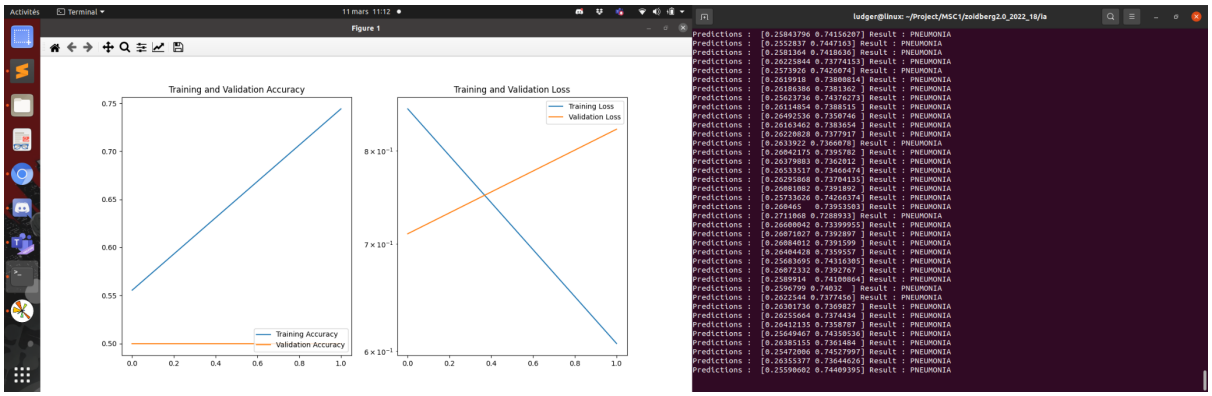

Training and Validation Accuracy
Training Accuracy
Validation Accuracy
0.75
0.70
0.65
0.60
0.55
0.50
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Training and Validation Loss
Training Loss
Validation Loss
9×10^{-1}
8×10^{-1}
7×10^{-1}
6×10^{-1}
0.0
0.5
1.0
1.5
2.0
2.5
3.0

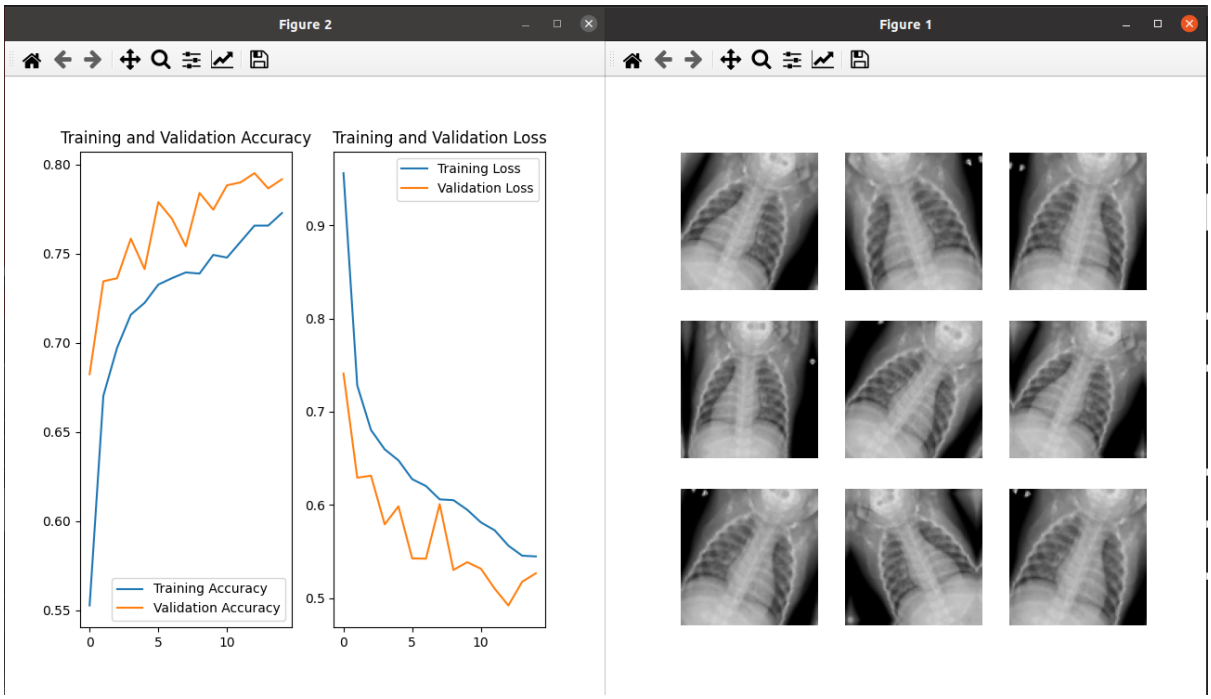

Training and Validation Accuracy
0.95
0.90
0.85
0.80
0.75
0.70
0
5
10
Training Accuracy
Validation Accuracy


Training and Validation Loss
Training Loss
Validation Loss
0.8
0.6
0.4
0.2
0
5
10

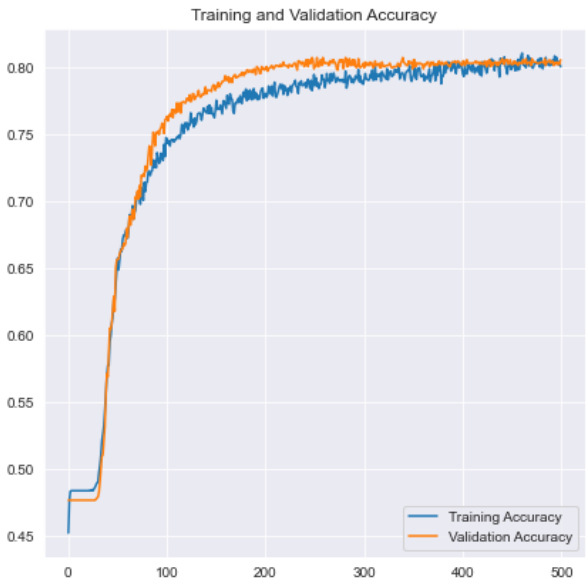
Unfortunately, at some point we were unable to avoid overfitting.

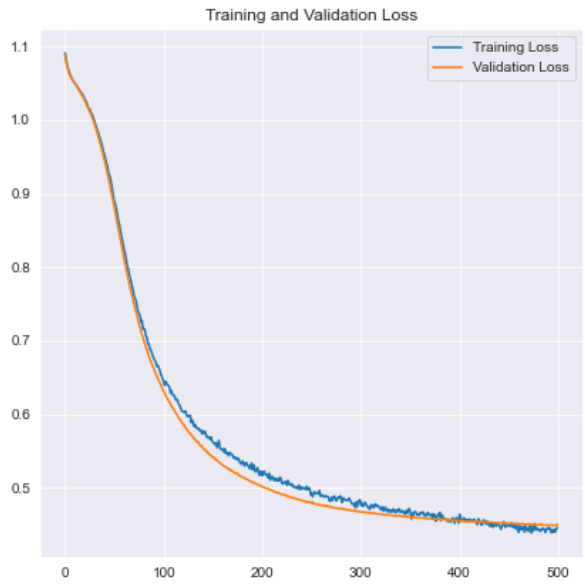


We also created several versions of each image by performing a simple rotation in order to increase the size of our dataset for training our model.

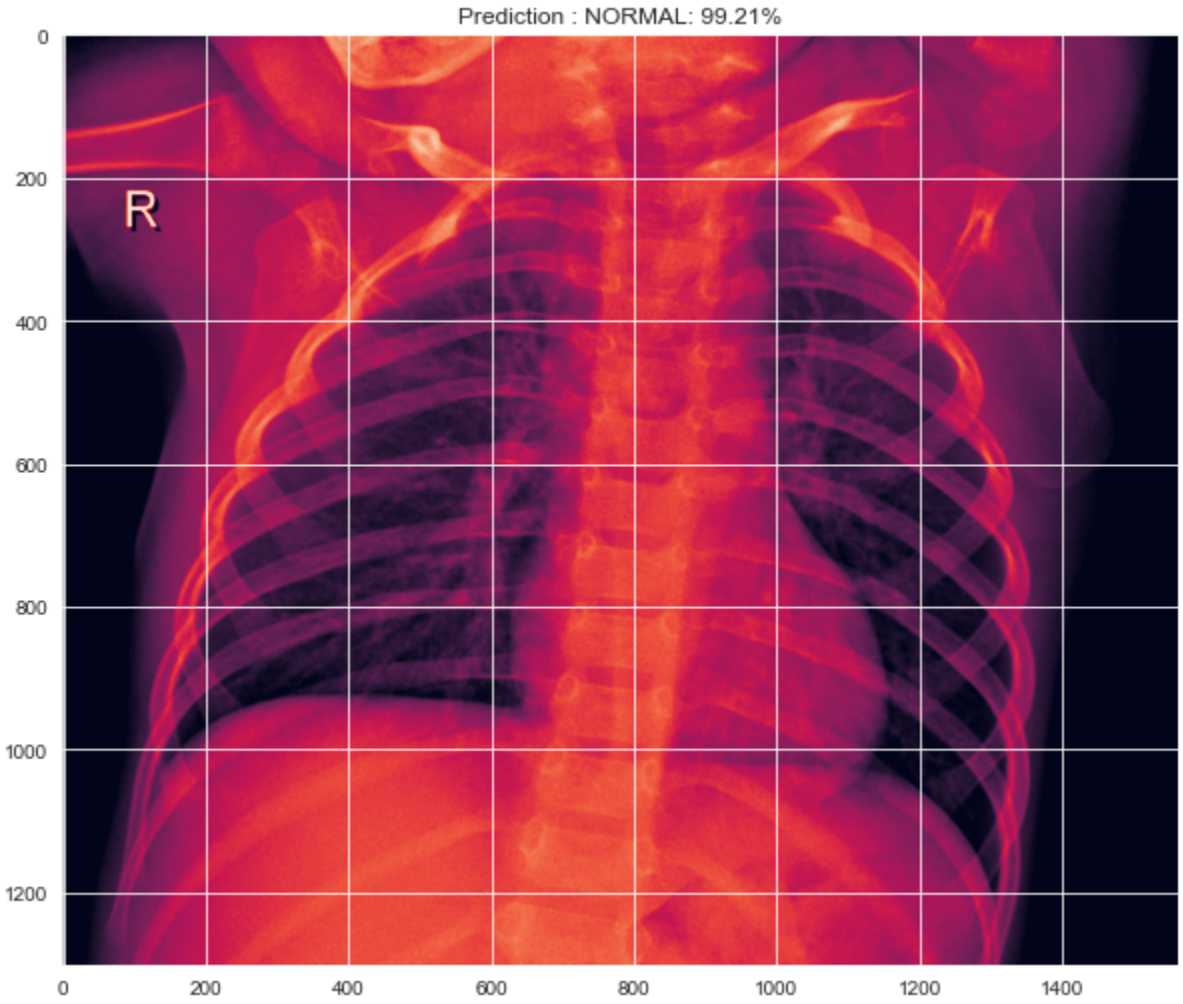


And finally, we managed to achieve results that exceeded our expectations.





Now the moment of truth. We are going to test our model with an image that it does not know.



99,21%, not bad for a first… For our first meeting with AI, we are very satisfied.

CONCLUSION

It was a very challenging and interesting project.

Except one, none of us have worked with AI before. We managed to pool our strengths together to complete this project. Although we have made progress in the understanding and application of artificial intelligence, many remain to be clarified. We were indeed able to familiarize ourselves with building a model and training it with a dataset in a real context with high stakes.

Health is a crucial area where AI can help save lives. It focuses on highly technical and repetitive tasks such as data analysis and allows physicians to focus on the core of their profession, namely their patients.

AI is revolutionizing the world. It is important that everyone understands how it works in general so as not to be afraid of it. The AI in sci-fi movies has nothing to do with the AI we use every day. It doesn't make any decisions on its own, but it helps us by directing us. It is a tool just like a hammer can be for a carpenter. Nothing more.

REFERENCES

1. https://en.wikipedia.org/wiki/Python_(programming_language)
2. https://en.wikipedia.org/wiki/Data_set
3. https://en.wikipedia.org/wiki/TensorFlow
4. https://en.wikipedia.org/wiki/Keras
5. https://en.wikipedia.org/wiki/Scikit-learn
6. https://en.wikipedia.org/wiki/Tensor_Processing_Unit
7. https://gandalf.epitech.eu/pluginfile.php/3370/mod_resource/content/13/cycle_2_KYT-CAT_artificial_intelligence_mindmapV2.pdf
8. https://www.youtube.com/playlist?list=PLH2l6uzC4UEVGUu2--3xBjTMFily1IwP9
9. https://github.com/Samorinho/mnist_digits
10. https://www.tensorflow.org/
11. https://keras.io/
12. https://scikit-learn.org/stable/
13. https://pythonprogramming.net/introduction-deep-learning-python-tensorflow-keras/
14. https://www.tensorflow.org/api_docs/python/tf
15. https://www.tensorflow.org/tutorials/images/classification