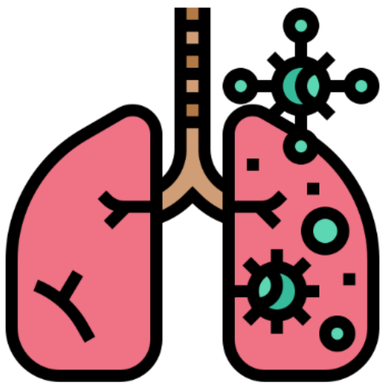


Rouibah Hanine

2025-2026

Report on Pneumonia vs. Normal Classification

1. **Part 1 : Training a CNN Model Directly with Images**

1.1. Data Preparation

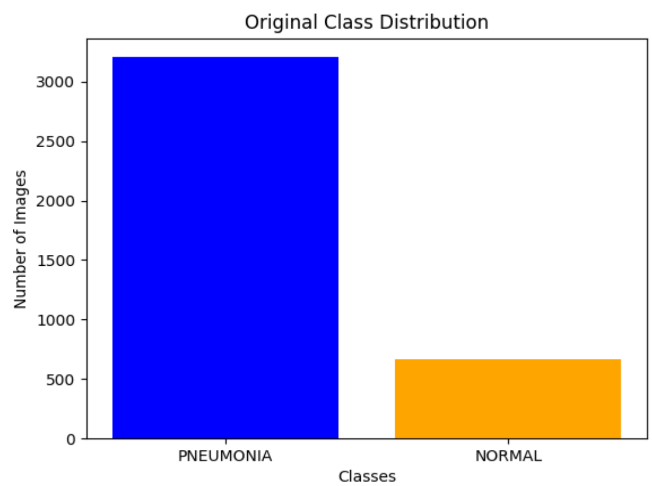
Dataset : Images from the two classes (**Pneumonia and Normal**):

- **Training : 3200** images for the **Pneumonia** class and **708** images for the **Normal** class
- **Validation : 651** images for the **Pneumonia** and **621** images for the **Normal** class

Class imbalance issue : The Pneumonia and Normal classes were initially imbalanced.

Data augmentation techniques were applied to artificially generate more examples in both classes to balance them.

Before Data Augmentation (training data only)

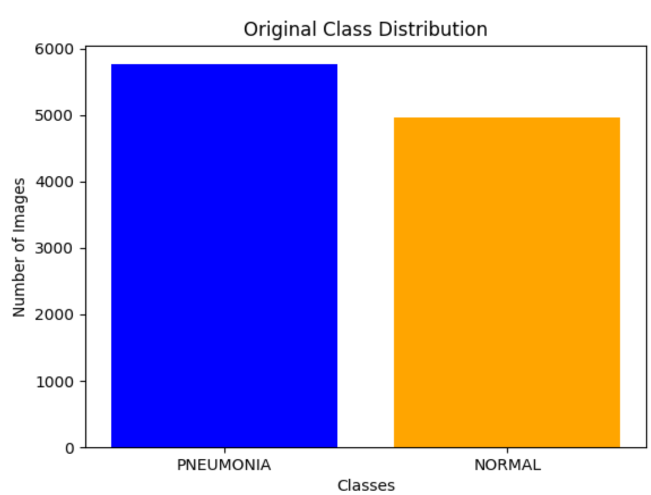


Data Augmentation :

- **Rotation ±15°**
- **Vertical shift 5%**
- **Horizontal shift 5%**
- **Zoom 10%**
- **Horizontal flip**

We augmented 20% of the data for the Pneumonia class and 100% for the Normal class.

After Data Augmentation



Visualization of augmented data

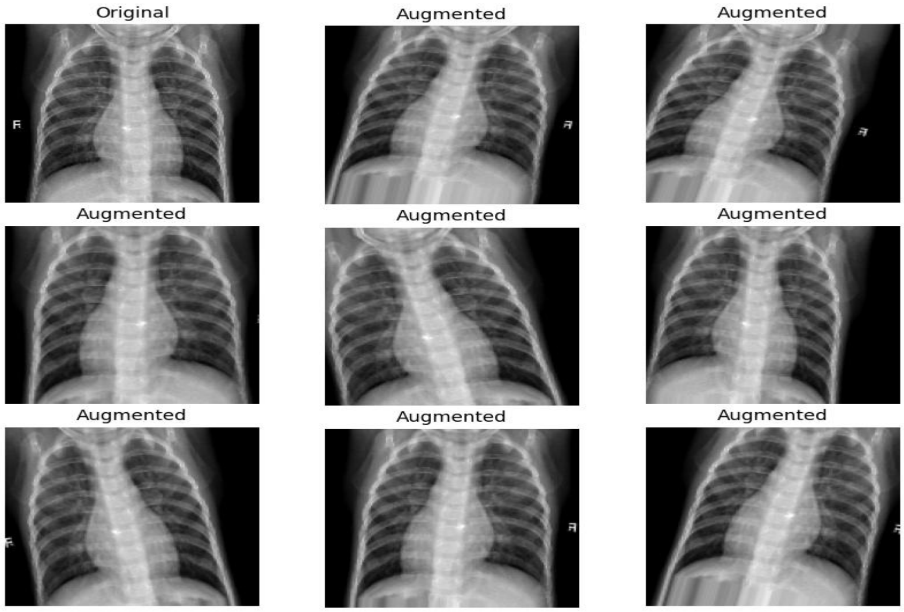


Image Resizing : All images (original and generated) were resized to **150×150** pixels and saved for future use.

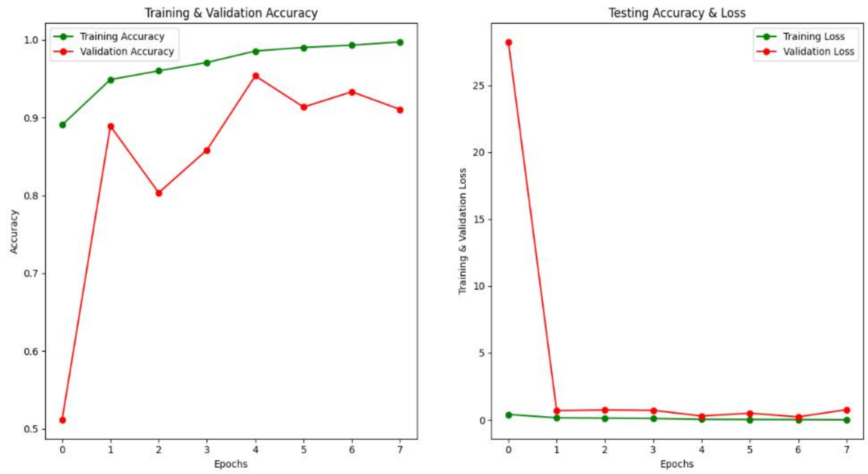
Data Normalization : Pixel values were normalized between [**0, 1**].

1.2. CNN Architecture

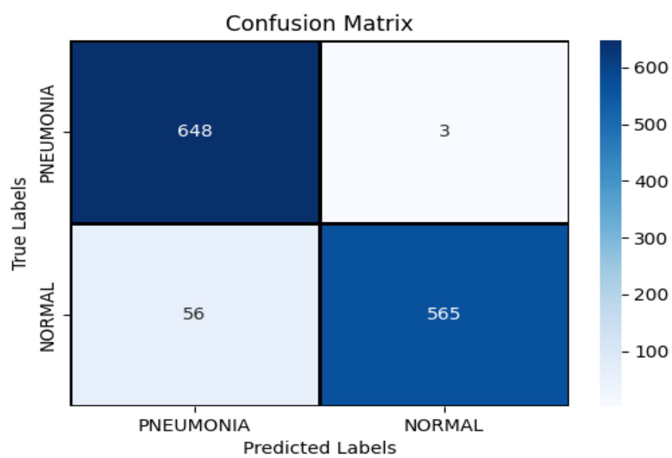
- **Layers** :
  - **Convolutional Layers** (**Conv2D**) **:** Apply filters to the input image to extract spatial features such as edges, textures, and patterns.
  - **MaxPooling Layers :** Reduce the spatial dimensions of the feature maps, retaining the most important information while reducing computational complexity.
  - **Batch Normalization :** Normalizes activations to stabilize and accelerate training.
  - **Dropout :** Randomly deactivates a fraction of input units during training to prevent overfitting.
  - **Flatten :** Converts **2D** feature maps into a **1D** vector for input into fully connected layers.
  - **Fully Connected Dense Layers with Activation Functions : ReLU** is used in hidden layers to introduce non-linearity, while **Sigmoid** is used in the output layer for binary classification.

- **Training Parameters** :
  - **Epochs : 15**
  - **Optimizer : RMSprop** (**Root Mean Square Propagation**) was used to adjust the network weights, due to its ability to maintain an adaptive and stable learning rate.
  - **Learning Rate : 0.001**
  - **Early Stopping :** Monitors validation accuracy and stops training if no improvement is observed after **3 epochs**.
  - **Reduction of learning rate** (**ReduceLROnPlateau**) : Monitors validation accuracy and reduces the **learning rate** by a factor of **0.3** if no improvement is seen after **2 epochs**.

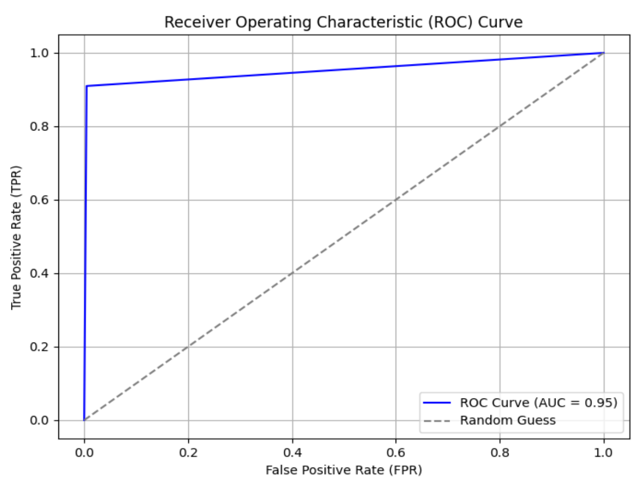
1.3. Training Results



The Confusion Matrix :



Receiver operating characteristic (ROC Curve):



- **Accuracy : 95.36 %**
- **Training Time : 2032.55 seconds**

2. Part 2 : Training a Machine Learning Model with Feature Extraction

2.1. Data Preparation

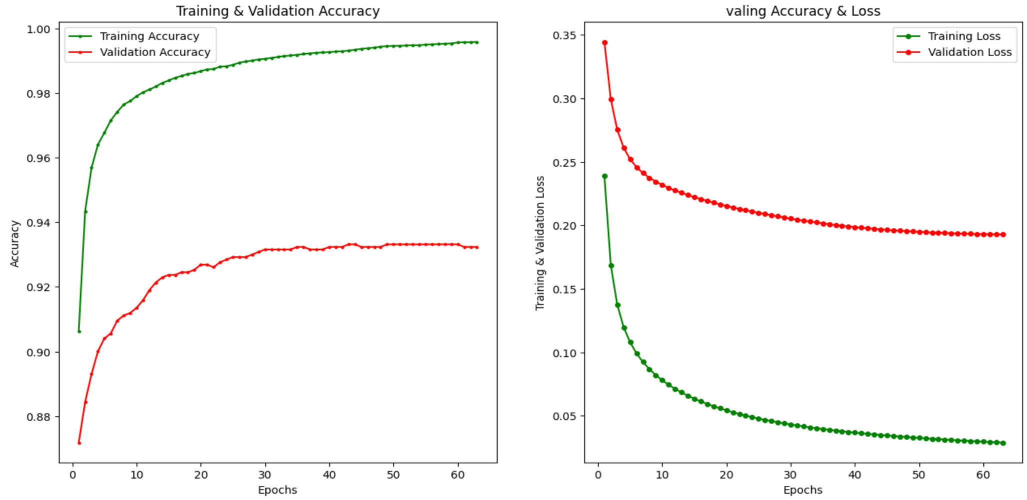
- **Dataset :** The same images used in training the **CNN** model were used here and for the next part, to ensure a consistent basis for comparison.
- **Extracted Features** :
    - **Pyramidal Histogram of Oriented Gradients (PHOG) :** Encodes spatial and orientation information of gradients, capturing the shape and structure of objects.
    - **Gabor Filters :** Capture texture features by analyzing frequency and orientation of the image at multiple scales.
    - **Discrete Cosine Transform (DCT) :** Transforms spatial data into the frequency domain, highlighting low-frequency components often critical for classification.
    - **Fourier Transform :** Converts spatial information into frequency components, useful for analyzing repetitive patterns or periodic structures.
- **Feature Vector**
    - Concatenation of the four methods : **[PHOG, Gabor, Fourier, DCT]**
    - This combination captures complementary features such as texture, frequency, and shape, allowing a comprehensive representation of the images.
- **Data Normalization :** Feature values were normalized between [**0, 1**] using the **Min-Max** method.

2.2. Model

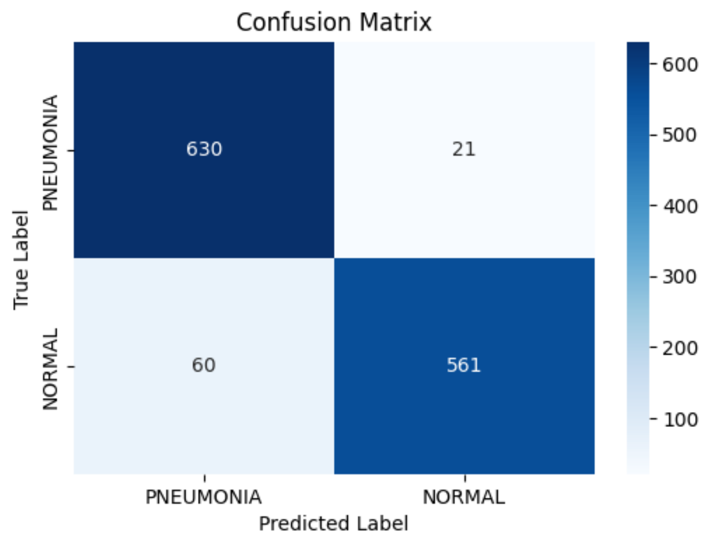
- **Algorithm : SGDClassifier** with **log_loss** loss function (logistic regression).
- **Training Parameters :**
    - **Epochs : 1000**
    - **Learning Rate : 0.0001** adaptive, which decreases based on model performance.
    - **Alpha = 0.0001 :** Regularization strength to prevent overfitting.
    - **Penalty = 'elasticnet' :** Elastic Net regularization combining L1 (lasso) and L2 (ridge) penalties.
    - **L1_ratio = 0.5 : 50 % L1** and **50 % L2** to balance the two types of regularization.
    - **Shuffle :** Shuffles the data at each iteration to improve convergence.
    - **Early stopping :** Monitors validation accuracy and stops training if no improvement is observed after **20 epochs**.

- Why Use **SGDClassifier** ?
    - **Performance on large Datasets : SGDClassifier** is better suited than **LogisticRegression** for large datasets, as it uses an incremental approach to adjust weights, making it faster and less memory-intensive.
    - **Control Over Regularization** : **SGDClassifier** allows direct specification of regularization parameters such as **l1** or **l2**, which is useful for preventing overfitting.

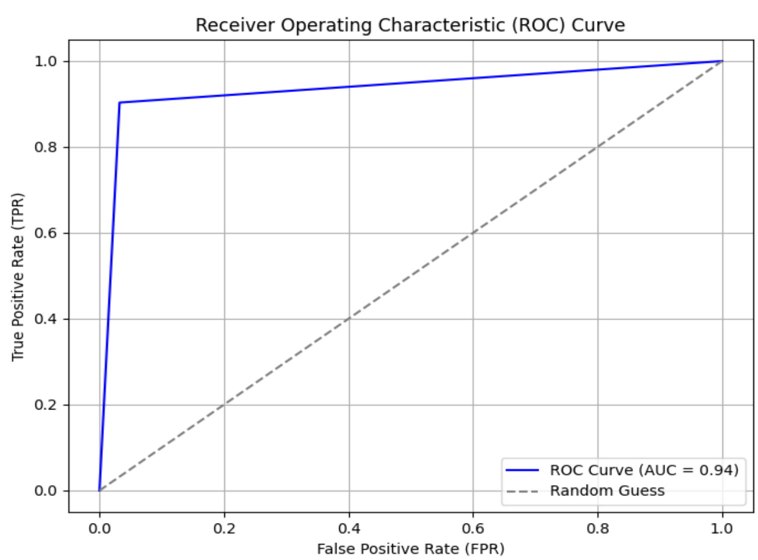
2.3. Training Results :



The Confusion Matrix :



Receiver operating characteristic (ROC Curve):



- **Accuracy : 93.63 %**
- **Training Time : 629.05 seconds**

3. Part 3 : Sequential Feature Processing for Machine Learning Models

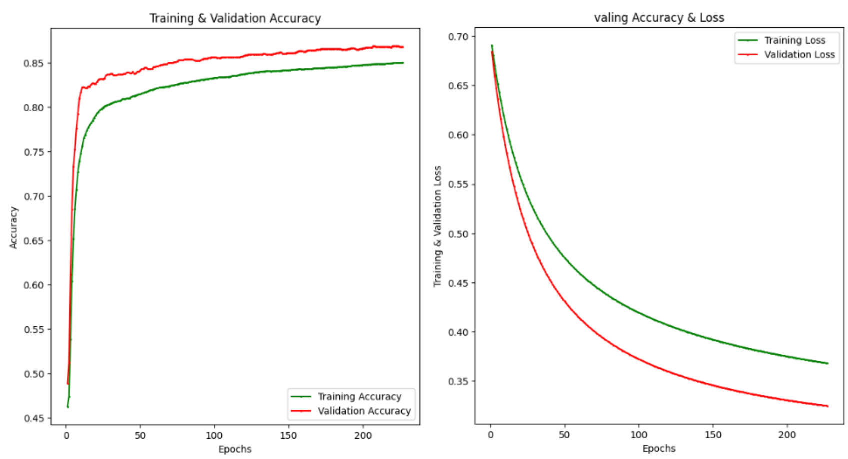
3.1. Data Preparation

- **Feature Extraction Pipeline :**
  - **Sequence** : **Gabor → Fourier → DCT → PHOG**
    - **Reasoning :**
      - **Gabor Filters :** Extract initial texture and orientation features as a base.
      - **Fourier Transform :** Analyzes periodic structures derived from **Gabor** features.
      - **DCT** : Focuses on low-frequency components of the data transformed by **Fourier**, compressing relevant information.
      - **PHOG** : Encodes spatial and gradient information from the processed data, providing shape-based features to complete the analysis.

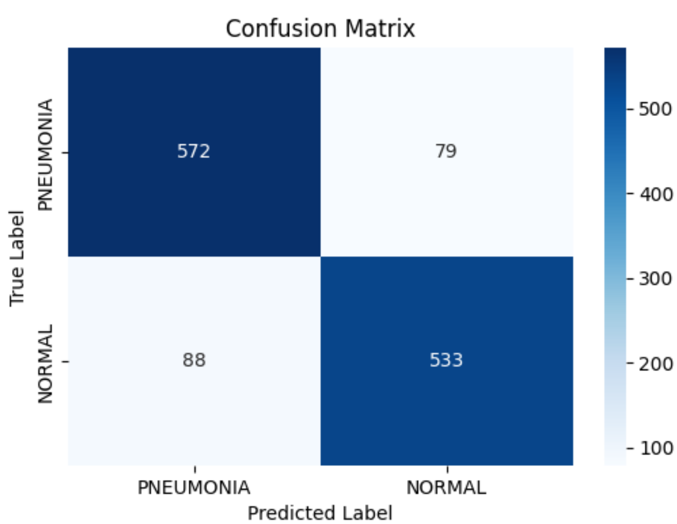
3.2. Model

- **Algorithm :** Identical to Part 2 (**SGDClassifier** with **log_loss**).
- **Model Parameters :** The same parameters used in the previous part.

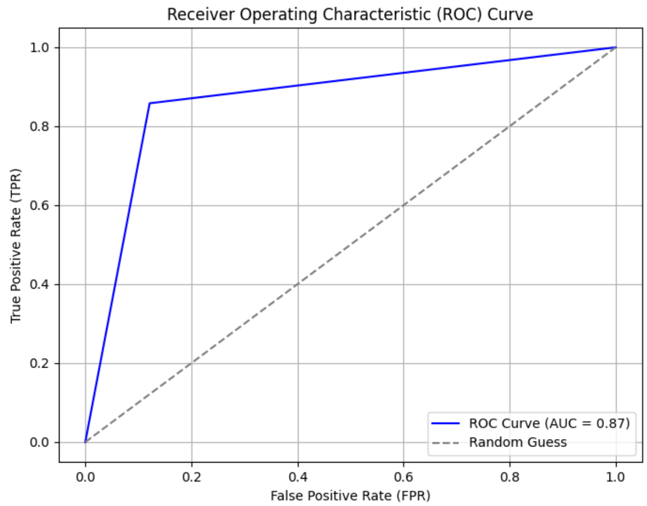
3.3. Training Results :



The Confusion Matrix :



Receiver operating characteristic (ROC Curve):



- **Accuracy : 86.87 %**
- **Training Time : 72.62 seconds**

---

4. Comparisons

4.1. Machine Learning Models

1. **Concatenation-Based Features**
   - **Accuracy : 93.63 %**
   - **Execution Time : 629.05 seconds**
2. **Sequential Features** :
   - **Accuracy : 86.87 %**
   - **Execution Time : 72.62 seconds**

Observation : The concatenation-based feature approach outperformed the sequential feature processing in terms of accuracy, at the cost of increased computation time.

4.2. Machine Learning vs. CNN

- **CNN Model**
  - **Accuracy : 95.36 %**
  - **Execution Time : 2032.55 seconds**
- **Best ML Model (Concatenation-Based Features)**
  - **Accuracy : 93.63 %**
  - **Execution Time : 629.05 seconds**

Observation : The **CNN** model provided the highest accuracy, demonstrating its superior ability to learn directly from images. However, it required significantly more computation time compared to the machine learning models.

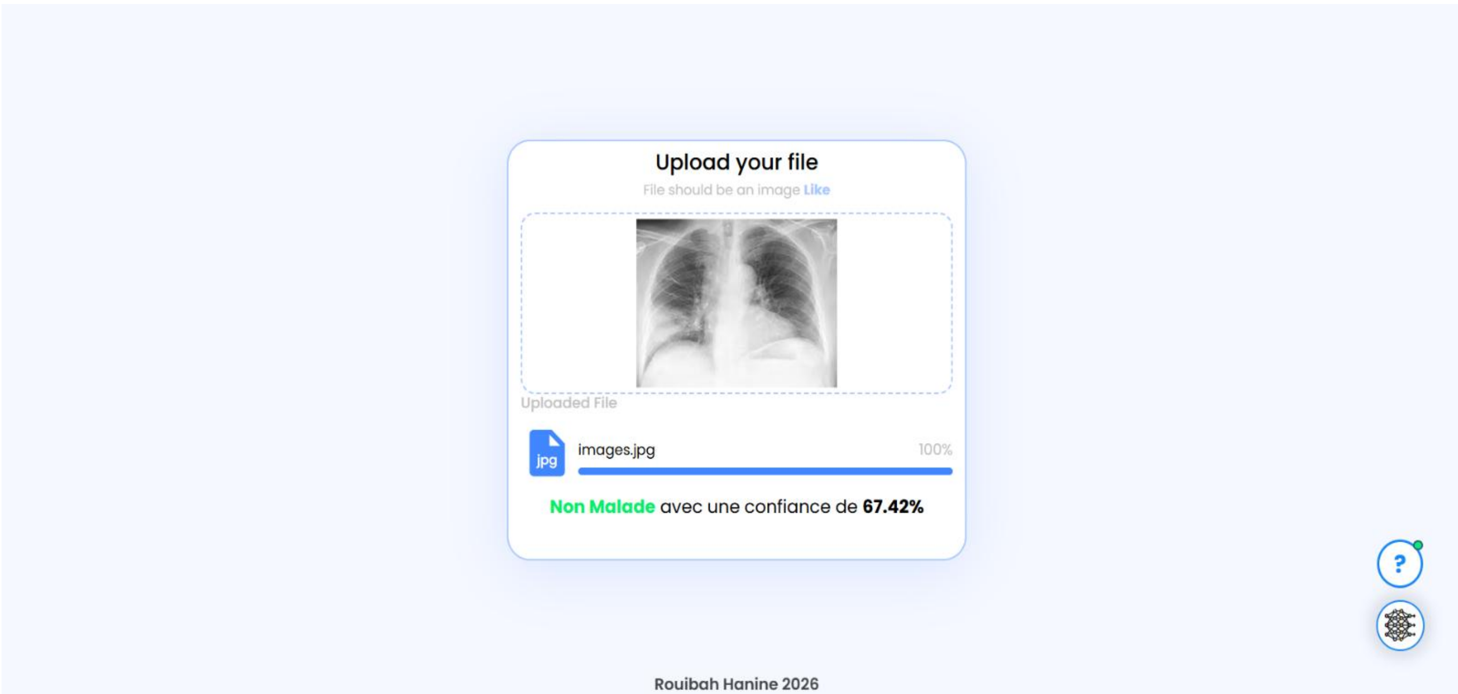
---

5. Presentation of the Prediction Interface

In addition to the developed models, a user interface was implemented to enable real-time prediction.

This interface allows:

- The user to choose the model : **CNN** or **ML**.
- Upload an image.
- Directly display the classification result : **Sick** or **Not Sick**, accompanied by a confidence percentage.
- Save the analyzed image in dedicated folders, classified according to the prediction result.



---

6. Conclusion

The **CNN** model is highly effective for binary classification (**Pneumonia vs. Normal**), achieving the best accuracy **(95.36%).**

Among the **machine learning models**, the concatenation-based feature approach (**PHOG, Gabor, Fourier, DCT**) obtained better results than the **sequential approach**, with a notable accuracy of **93.63%.**

For scenarios where **execution time** is a critical factor, the sequential feature processing model (**86.87%** accuracy in **72.62 seconds**) can be a **viable alternative**.

END