Report TP-4DL

Transfer learning

binome : ATEK , IGUERNAISSI

- The importance of data loading and pre-processing for transfer learning with chest X-ray images. the use of ImageDataGenerator for image loading, normalization, and the creation of separate generators for training, validation, and testing. Implementing these practices lays the foundation for effective transfer learning with chest X-ray datasets.

- **Mobilnet :**

MobileNet used for Transfer Learning :

MobileNet is a pre-trained convolutional neural network (CNN) architecture known for its efficiency and accuracy, making it a strong candidate for transfer learning, especially when dealing with limited computational resources or large datasets like chest X-rays.

Transfer Learning Approach

1. **Load the Pre-trained MobileNet Model.**
2. **Freeze Base Layers:** Transfer learning leverages the knowledge learned by the pre-trained model. The initial layers (often called the base layers) typically extract low-level features that are generally applicable across different tasks. Freezing these layers prevents them from being re-trained on your specific chest X-ray dataset, maintaining their generalizability**.**
3. **Add New Classification Layers(Relu, softmax):** Following the frozen base layers, you'll add new layers specific to your classification task, these layers would be responsible for classifying chest X-rays.
4. **Compile the Model.**
5. **Train the Model:** Train the model using the pre-processed training data generated earlier. Monitor the training process and adjust hyperparameters as needed.
6. **Evaluate the Model:** Evaluate the model's performance on unseen validation data using the chosen metrics. This helps assess the model's ability to generalize to new data.

Layres of mobilnet before freezing :

Model: "mobilenet_1.00_224"

Layer (type)	Output Shape	Param #
input_layer_2 (InputLayer)	(None, 224, 224, 3)	0
conv1 (Conv2D)	(None, 112, 112, 32)	864
conv1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv1_relu (ReLU)	(None, 112, 112, 32)	0
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288
conv_dw_1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv_dw_1_relu (ReLU)	(None, 112, 112, 32)	0

Total params: 4,253,864 (16.23 MB)

Trainable params: 4,231,976 (16.14 MB)

Non-trainable params: 21,888 (85.50 KB)

- **Remove and replace the last fully connected layers by three fully connected layers composed of 1024, 512, and c (Number of classes) neurons.**

Total params: 5,805,740 (22.15 MB)

Trainable params: 5,783,852 (22.06 MB)

Non-trainable params: 21,888 (85.50 KB)

Comparison of MobileNet and MobileNet with Modified Fully-Connected Layers.

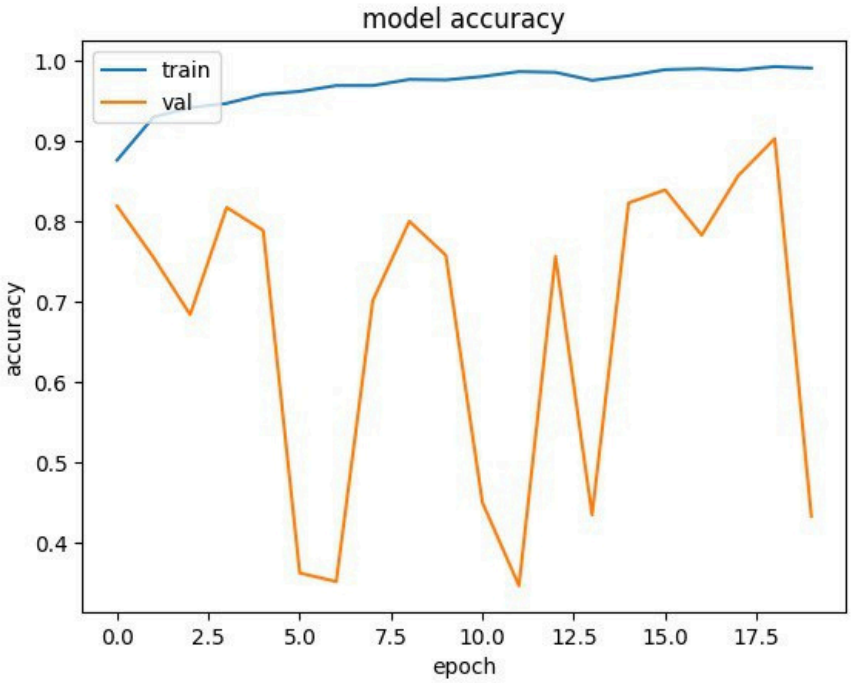
Here's a table comparing MobileNet and MobileNet with modified fully-connected layers:

Feature	MobileNet	MobileNet with Modified Fully-Connected Layers
Architecture	Pre-trained convolutional neural network (CNN)	Pre-trained MobileNet with modified top layers
Base Layers	Frozen (not trainable)	Frozen (not trainable)
New Fully-Connected Layers	None	Three fully-connected layers with 1024, 512, and C neurons
Number of Trainable Parameters	Lower (fewer parameters to train)	Higher (more parameters to train)
Training Time	Faster	Slower (due to additional trainable parameters)
Generalization Ability	Potentially better (leveraging pre-trained features)	May vary depending on dataset and task complexity
Classification Performance	Good for various tasks	Potentially better for specific tasks

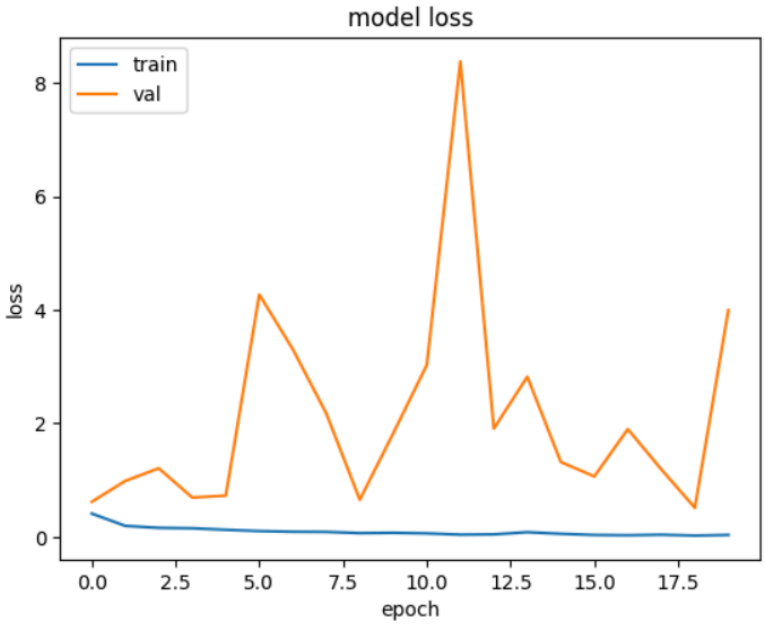
one convolutional layers + the last three fully connected :

```
dropout (Dropout)           (None, 1, 1, 1024)        0

conv_preds (Conv2D)         (None, 1, 1, 1000)        1025000

reshape_2 (Reshape)         (None, 1000)              0

dense (Dense)               (None, 1024)              1025024

dense_1 (Dense)             (None, 512)               524800

dense_2 (Dense)             (None, 4)                 2052

=================================================================
Total params: 5805740 (22.15 MB)
Trainable params: 2576876 (9.83 MB)
Non-trainable params: 3228864 (12.32 MB)
```

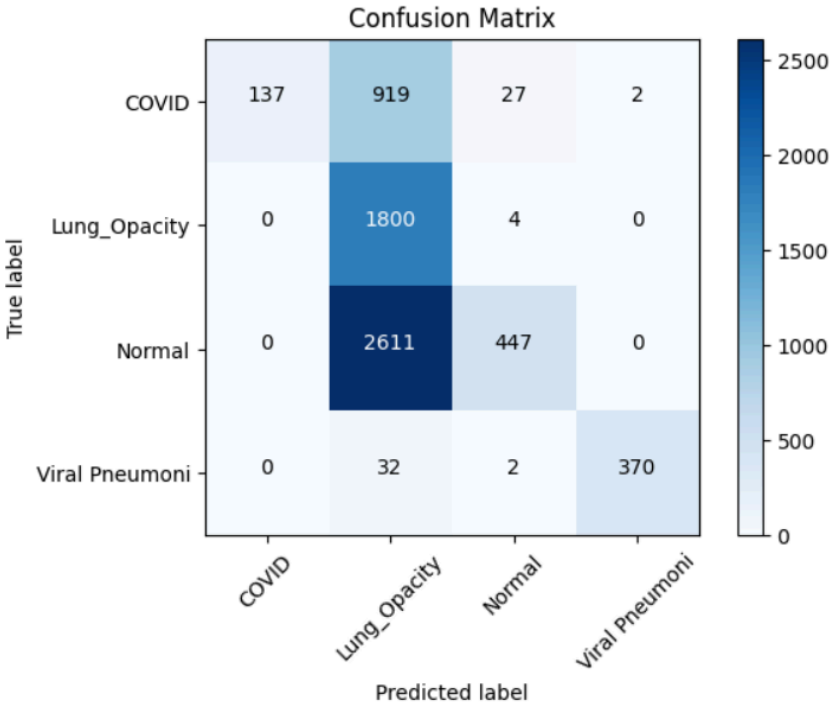
Accuracy :



Loss :



Confusion matrix :



test :

Accuarcy 0.4336324988190836

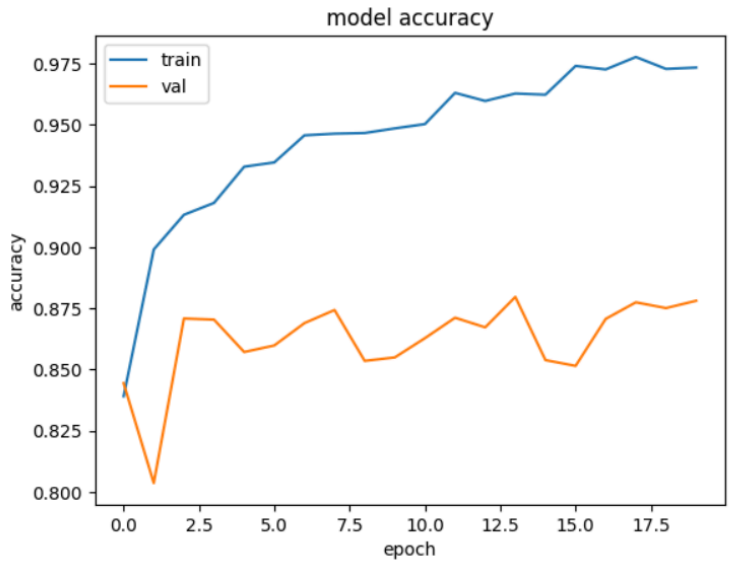
f1_score 0.4832220699736022

Precision 0.8153923229676777

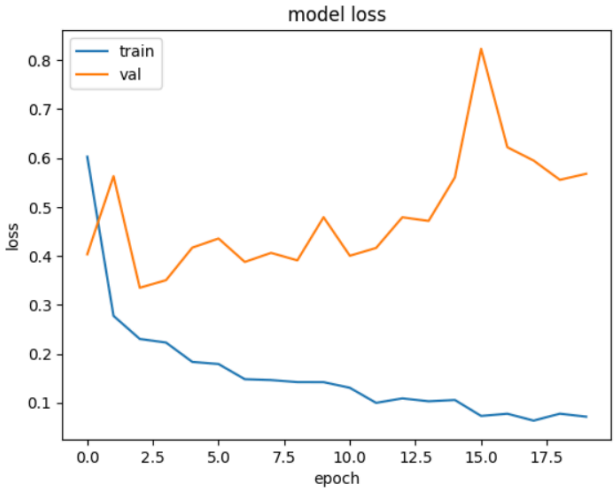
Recall 0.5465163850697905

Two convolutional layers + the last three fully connected:

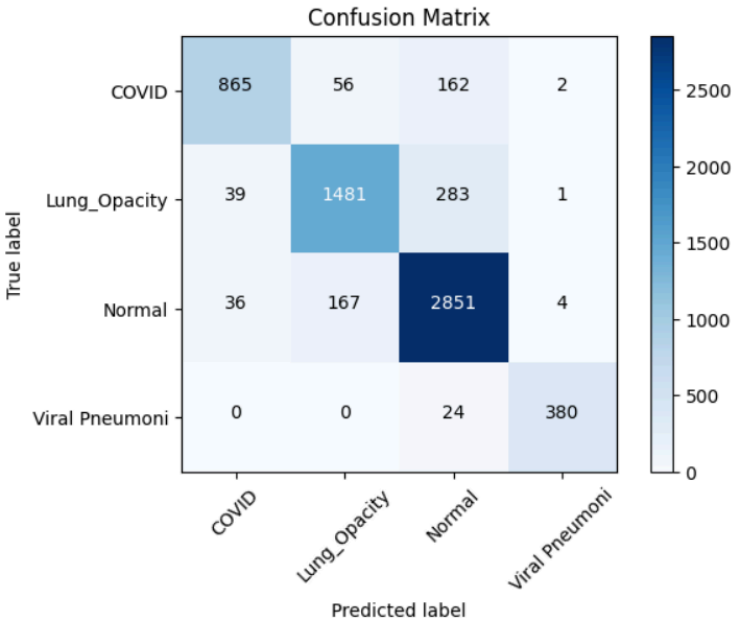
Accuracy :



Loss :



Confusion Matrix :



Test :

Accuarcy 0.8781294284364667

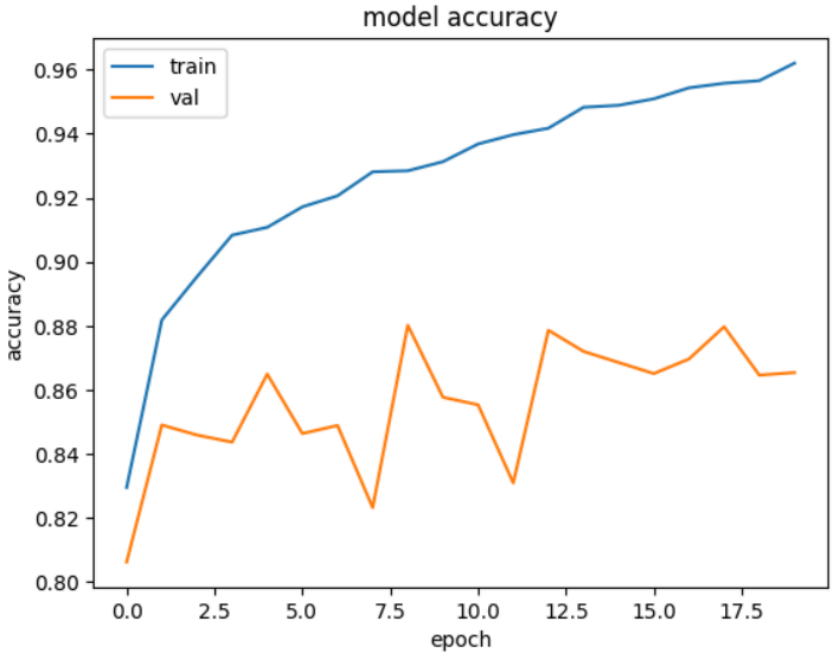
f1_score 0.8883741274931456

Precision 0.9074978264545966

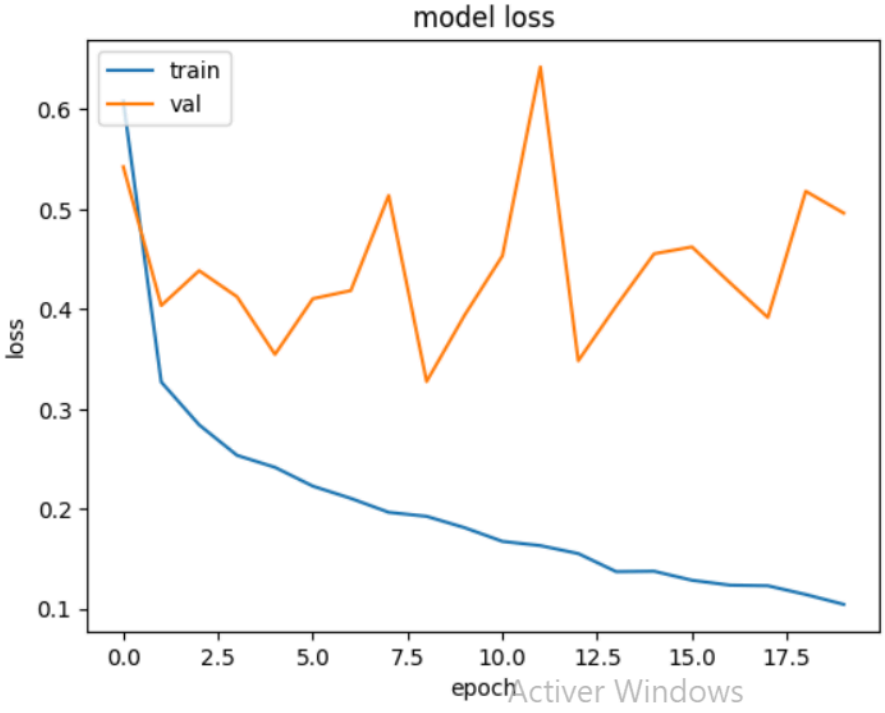
Recall 0.8727728044375648

The last three fully connected :

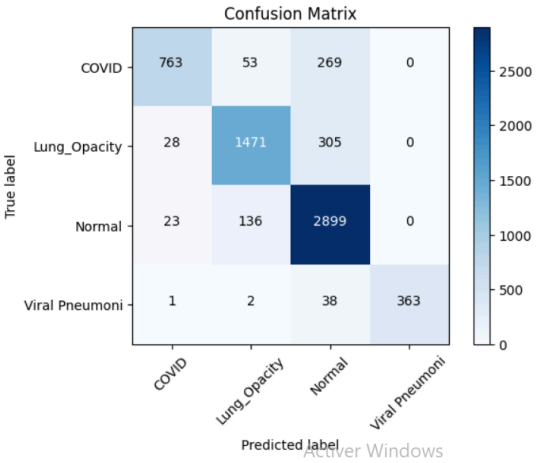
Accuracy :



Loss :



Confusion Matrix :



Test :

Accuarcy 0.865375531412376

f1_score 0.8702876231900918

Precision 0.9117413061114422

Recall 0.8412890224177991

Performance Comparison of MobileNet Architectures for Chest X-ray Classification :

Configuration	Accuracy	F1-Score	Precision	Recall
MobileNet with Last 3 Fully-Connected Layers	0.8654	0.8703	0.9117	0.8413
MobileNet with 2 Additional Conv Layers + Last 3 FC	**0.8781**	**0.8884**	0.9075	0.8728
MobileNet with 1 Additional Conv Layer + Last 3 FC	0.4336	0.4832	0.8154	0.5465

Key Observations:

- **Adding Convolutional Layers Improves Performance:** The configuration with two additional convolutional layers achieved the highest accuracy (0.8781) and F1-score (0.8884)

compared to the baseline with only fully-connected layers. This suggests that the additional convolutional layers were able to extract more relevant features from the chest X-ray images, leading to better classification.

- **Impact of Fully-Connected Layers:** All configurations utilizing the last three fully-connected layers achieved reasonable accuracy, indicating their importance in learning the classification boundaries for the specific task.
- **Single Convolutional Layer Insufficient:** The configuration with just one additional convolutional layer resulted in significantly lower performance (0.4336 accuracy), suggesting that a single layer might not be enough to learn effective feature representations for this task

The optimal configuration depends on several factors, including:

- **Dataset Size:** Larger datasets can support more complex models with additional convolutional layers.
- **Computational Resources:** Training models with more convolutional layers requires more processing power and memory.
- **Performance Requirements**
- **Real-Time Constraints:** If the model needs to run in real-time, faster training and inference speeds might be a priority.