RAPPORT PROJET AI & BIG DATA

Groupe 11 - T-DEV-810 - Epitech promo 2021 - Msc pro

05/05/2020

SOMMAIRE

INTRODUCTION

A l'aide d'un jeu de données, nous devons réaliser la création et l'entrainement d'un ou plusieurs modèles de machine learning afin de déterminer si les radiologies de patients démontrent ou non une pneumonie. Dans le cas là d'une pneumonie, nous devons pouvoir détecter si elle provient d'un virus ou d'une bactérie.

ÉQUIPE

L'équipe est composé de David Vera, Léo Hamon, Antoine Falais, Said Ali Hamadou et Mathieu Dufour, tous étudiants à Epitech en Master Professionnel.

OUTILS UTILISÉS

Afin de mener à bien ce premier projet IA et Big Data, nous avons décidé d'utiliser des technologies très présentes dans ce pan de l'informatique.

TensorFlow (https://www.tensorflow.org/).

Scikit-learn (https://scikit-learn.org/stable/index.html)

Pandas (https://pandas.pydata.org/)

OpenCV (https://opencv.org/)

NumPy (https://numpy.org/)

PRÉPARATION

Le dataset fourni contient 5856 images.

Le dataset en l'état ne peut pas être utilisé pour une stratégie de train-validation-test car il est déséquilibré. En effet, il est généralement préférable d'avoir une répartition de

60%/20%/20% (ou 70%/15%/15%) pour le dataset de train/validation/test.

Nous effectuons de plus une augmentation de données faite aléatoirement sur les images. Nous appliquons une rotation, un agrandissement, une réduction ou une transposition sur celles-ci afin d'avoir des résultats plus cohérent sur la phase de test.

Sources pour l'équilibrage du dataset :
https://stackoverflow.com/questions/13610074/is-there-a-rule-of-thumb-for-how-to-divide-a-dataset-into-training-and-validatio

https://visualstudiomagazine.com/articles/2015/05/01/train-validate-test-stopping.aspx

https://www.researchgate.net/post/Is_there_an_ideal_ratio_between_a_training_set_and_validation_set_Which_trade-off_would_you_suggest

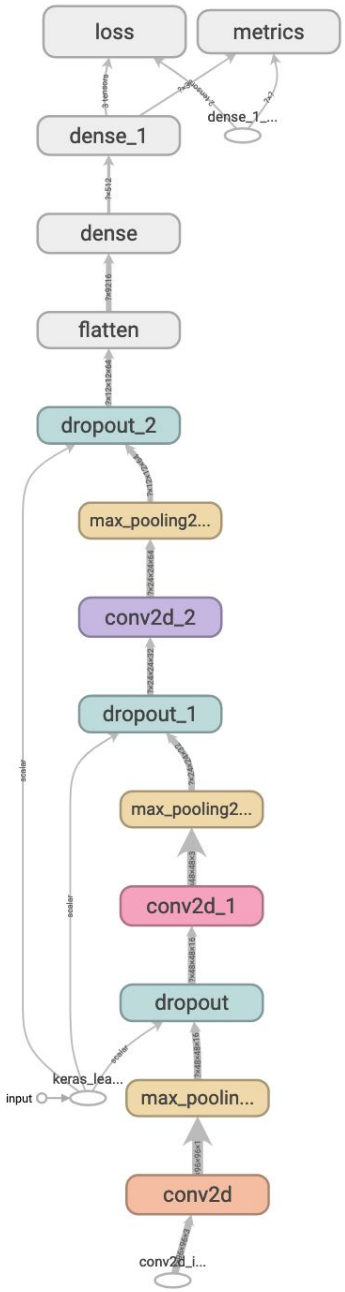
MODÈLES UTILISÉS

VGG16

Le VGG16 est un modèle de réseau neuronal convolutif proposé par K. Simonyan et A. Zisserman de l'université d'Oxford dans l'article "Very Deep Convolutional Networks for Large-Scale Image Recognition".

Il utilise 3 couches convolutionnelles classiques complètements connectées chacunes suivies d'une couche de max pooling permettant une mise en commun de l'espace des pixels de sortie pour faire une réduction de 50% des pixels utilisés pour la couche convolutionnel suivante.

Structure du modèle :



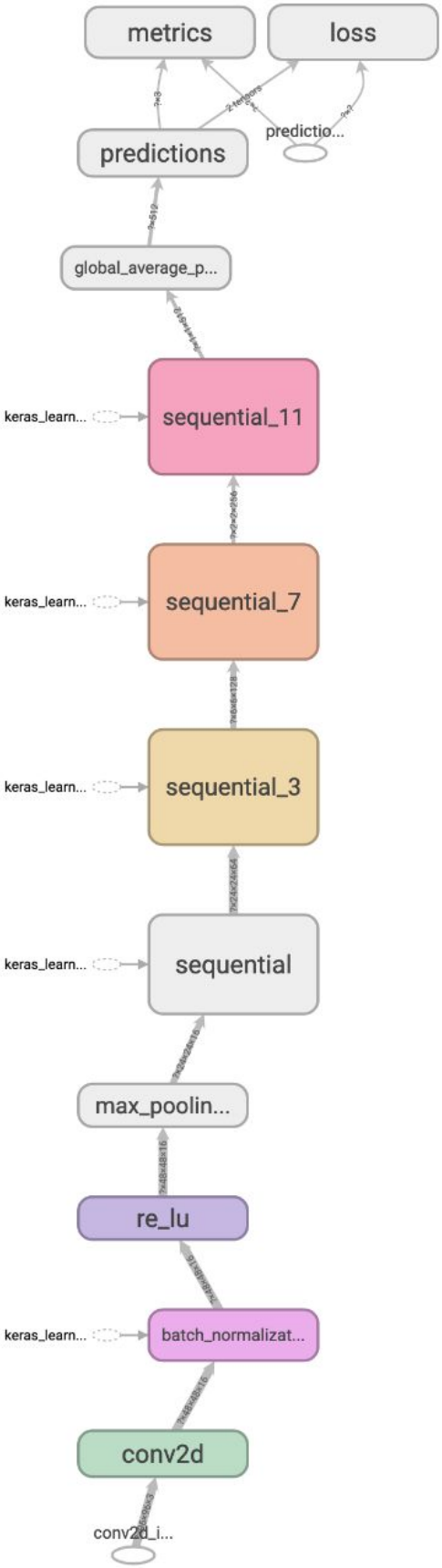
RESNET18 et RESNET34

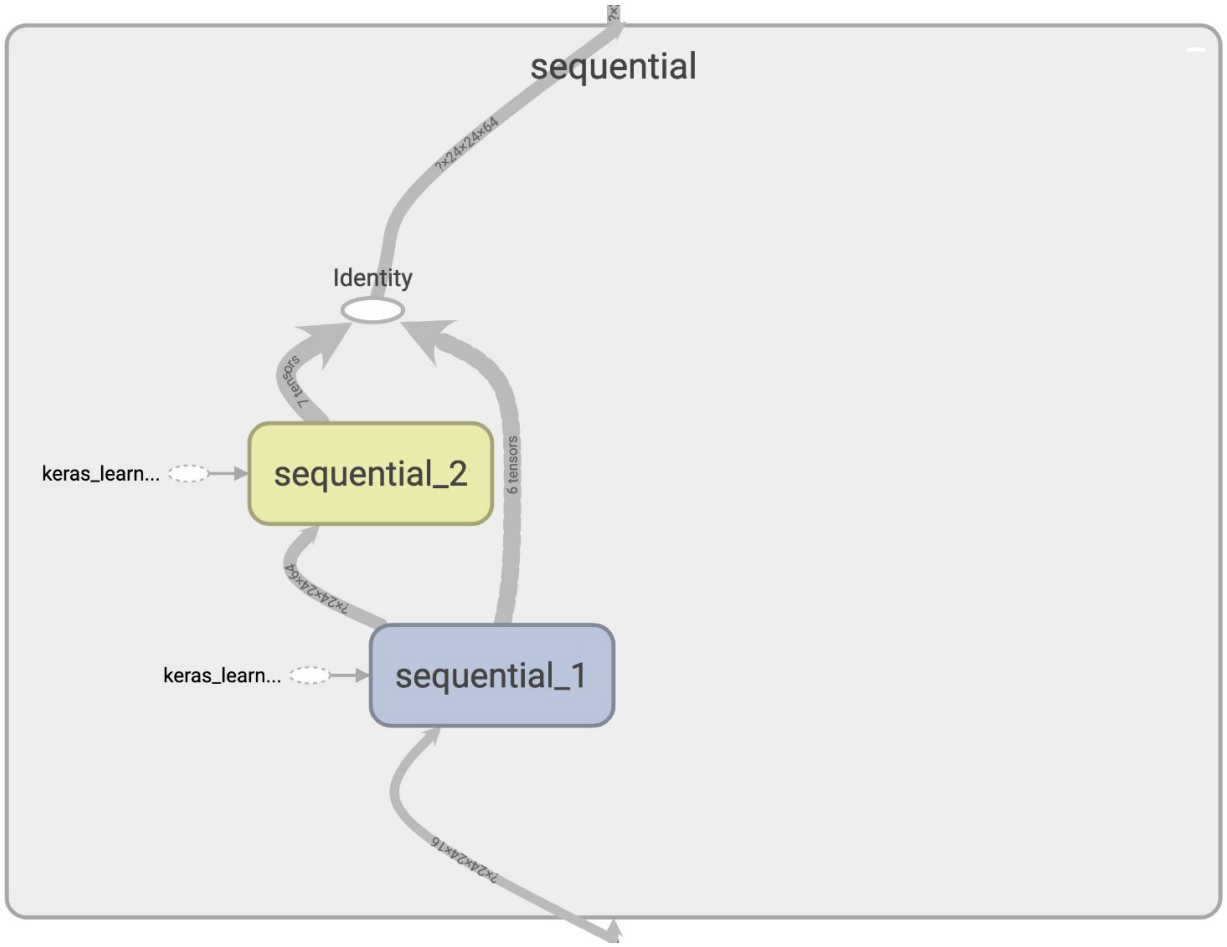
Un réseau neuronal résiduel (ResNet) est un réseau qui s'appuie sur des constructions connues des cellules pyramidales du cortex cérébral. Les réseaux neuronaux résiduels le font en utilisant des connexions de saut, ou des raccourcis pour sauter par-dessus certaines couches. Les modèles ResNet typiques sont mis en oeuvre avec des sauts de couche doubles ou triples qui contiennent des non-linéarités (ReLU) et une normalisation par lots entre les deux.

Une des motivations pour sauter des couches est d'éviter le problème de la disparition des gradients, en réutilisant les activations d'une couche précédente jusqu'à ce que la couche adjacente apprenne ses poids.

Nous utilisons un Resnet18 et un Resnet34, qui va juste changer le nombre de couches pour l'entraînement.

Structure du modèle Resnet:





MÉTRIQUES UTILISÉES

Voici la liste des métriques retournées pour chaque entrainement des modèles :

- BinaryAccuracy/CategoricalAccuracy: Calcule la fréquence à laquelle les prédictions correspondent aux étiquettes.
- FalseNégatives: Calcule le nombre de faux négatifs.
- FalsePositives: Calcule le nombre de faux positifs.
- TrueNégatives: Calcule le nombre de vrais négatifs.
- TruePositives: calcule le nombre de vrais positifs.
- Precision : Calcule la précision des prévisions en ce qui concerne les étiquettes.
- Recall : Calcule le rappel des prédictions en ce qui concerne les étiquettes.
- AUC : Calcule la CUA (surface sous la courbe) approximative au moyen d'une somme de Riemann. Permet de savoir si le modèle va bien assigner un type d'étiquette à l'étiquette correspondante.

RÉSULTATS SUR DE LA CLASSIFICATION 2 LABELS (NORMAL, PNEUMONIE)

VGG16

1)

Optimizer : Adam

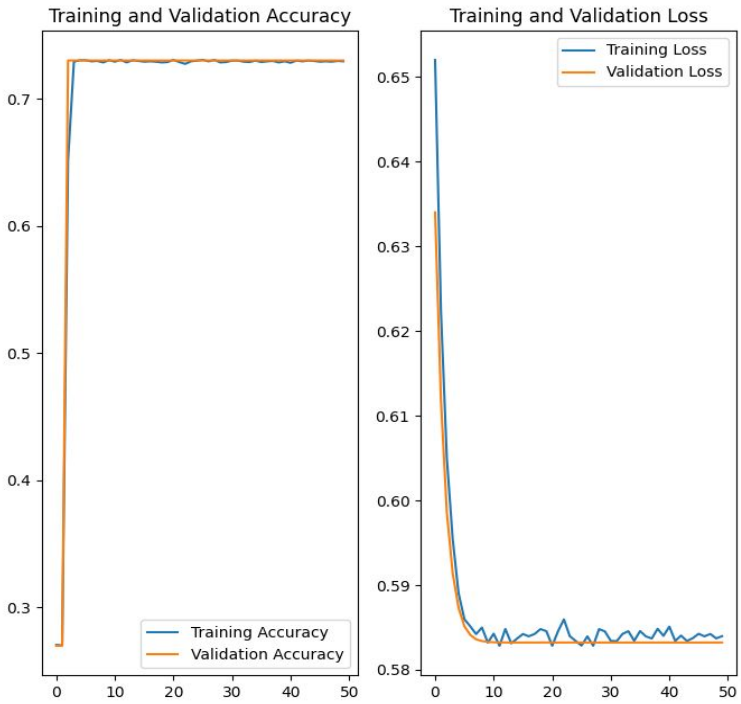
Activation des couches cachées : ReLu

Activation de la couche de sortie : Softmax

Résultat N°1:

loss : 0.5840

accuracy : 0.7293



Ce modèle est un modèle de classification d'image assez simple, d'où la monté très rapide de l'accuracy. L'utilisation d'un early-stopping le fait donc s'arrêter à l'epoch numéro 50 car il n'apprend plus.

2)

Optimizer : Adam

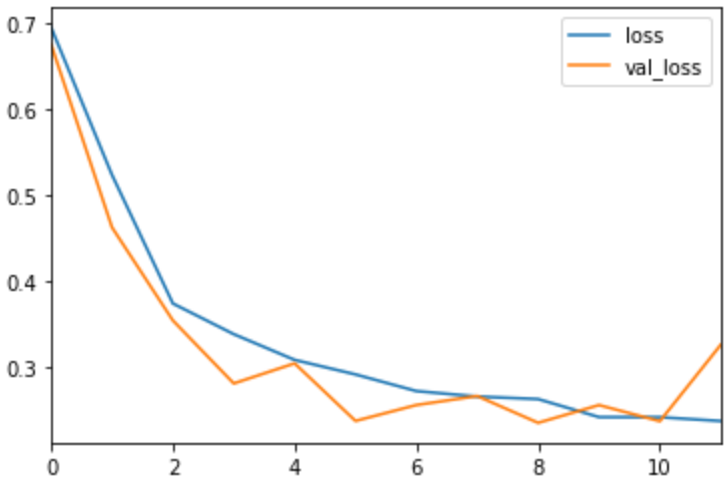
Activation des couches cachées : ReLu

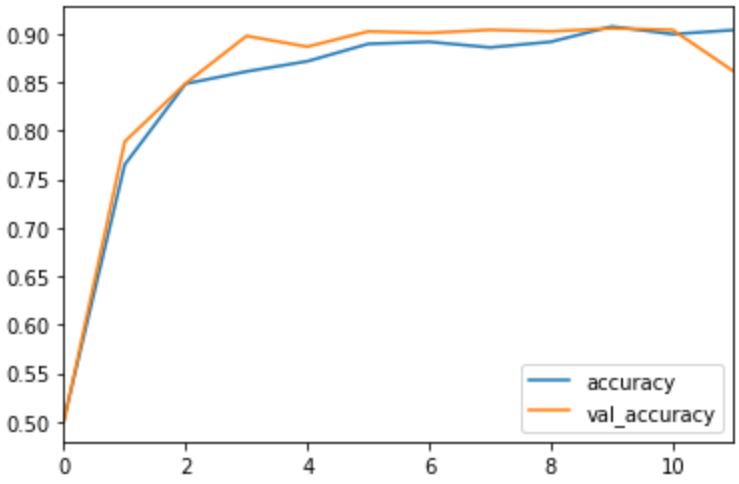
Activation de la couche de sortie : Sigmoid

Résultat N°1 :

loss : 0.1873811746481806

accuracy : 0.9227129

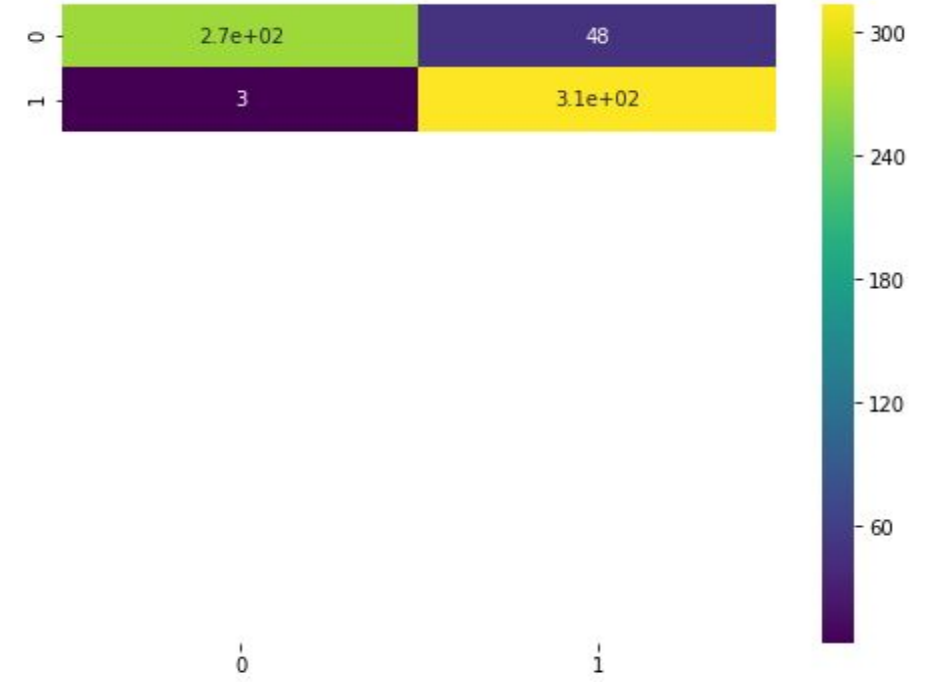




	precision	recall	f1-score	support
0	0.96	0.77	0.86	317
1	0.81	0.97	0.88	317
accuracy			0.87	634
macro avg	0.88	0.87	0.87	634
weighted avg	0.88	0.87	0.87	634

Sur la phase d'apprentissage, la matrice de confusion donne les résultats suivants

```
[[269  48]
 [  3 314]]
Out[85]: (10, 0)
```



Cela correspond à 583 prévisions correctes sur 634, soit 91%. Ici les faux négatif correspondent aux cas de radios de personnes non atteintes déclarées malades. Le nombre de personnes atteintes déclarées comme saines est très faible.

Sur la phase de validation de l'apprentissage, un set de 634 images différentes donnent le résultat suivant :

	precision	recall	f1-score	support
0	0.99	0.85	0.91	317
1	0.87	0.99	0.92	317
accuracy			0.92	634
macro avg	0.93	0.92	0.92	634
weighted avg	0.93	0.92	0.92	634

```
[[269  48]
 [  3 314]]
```

RÉSULTATS SUR DE LA CLASSIFICATION 3 LABELS (NORMAL, VIRUS, BACTÉRIE)

VGG16

1)

Optimizer : Adam

Activation des couches cachées : ReLu

Activation de la couche de sortie : Sigmoid

Résultat N°1 :

loss : 0.7128407756487528

accuracy : 0.7757575511932373

true_positives : 127.0

false_positives : 35.0

true_negatives : 295.0

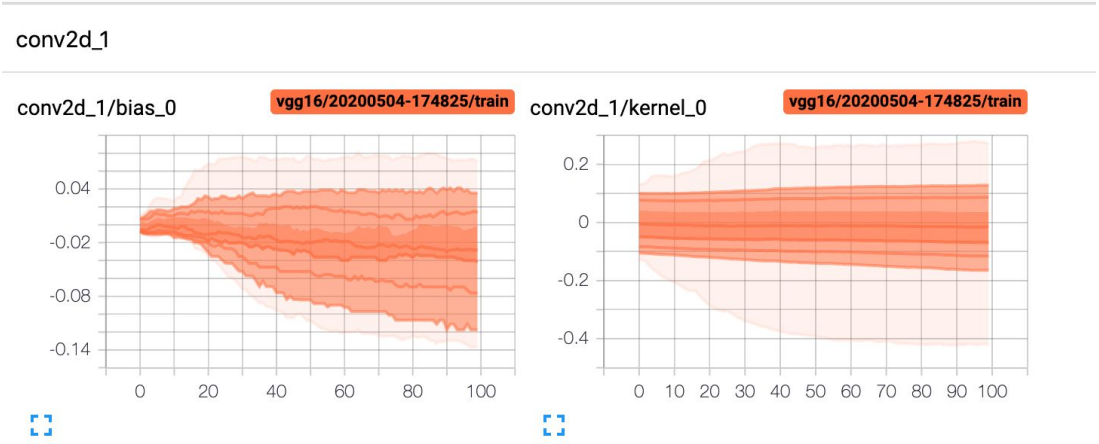
false_negatives : 38.0

precision : 0.7839506268501282

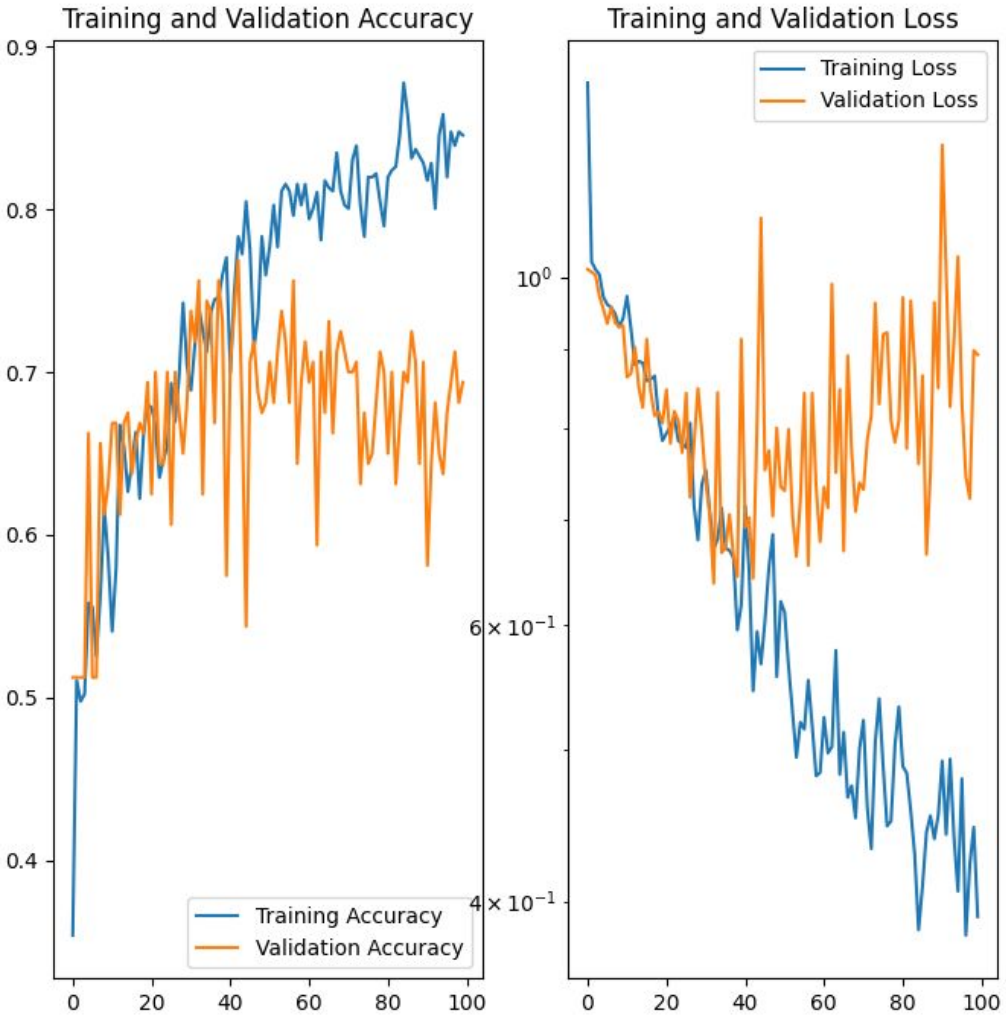
recall : 0.7696969509124756

auc : 0.8929477334022522

Distribution des poids sur la première couche de convolution :



Accuracy et loss pour l’entrainement et la validation :



Résultat N°2 :

loss : 0.8620439816165615

accuracy : 0.7639007568359375

true_positives : 885.0

false_positives : 263.0

true_negatives : 2075.0

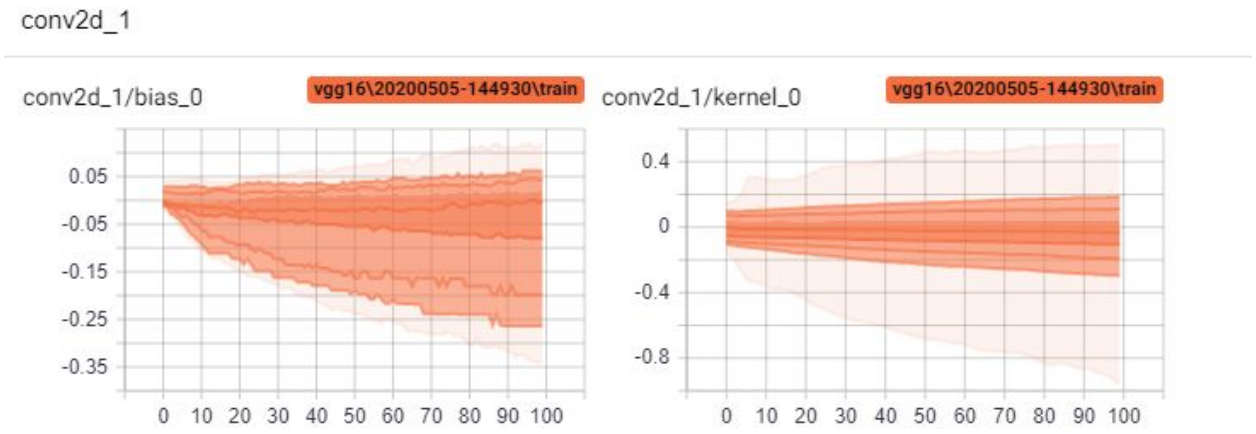
false_negatives : 284.0

precision : 0.7709059119224548

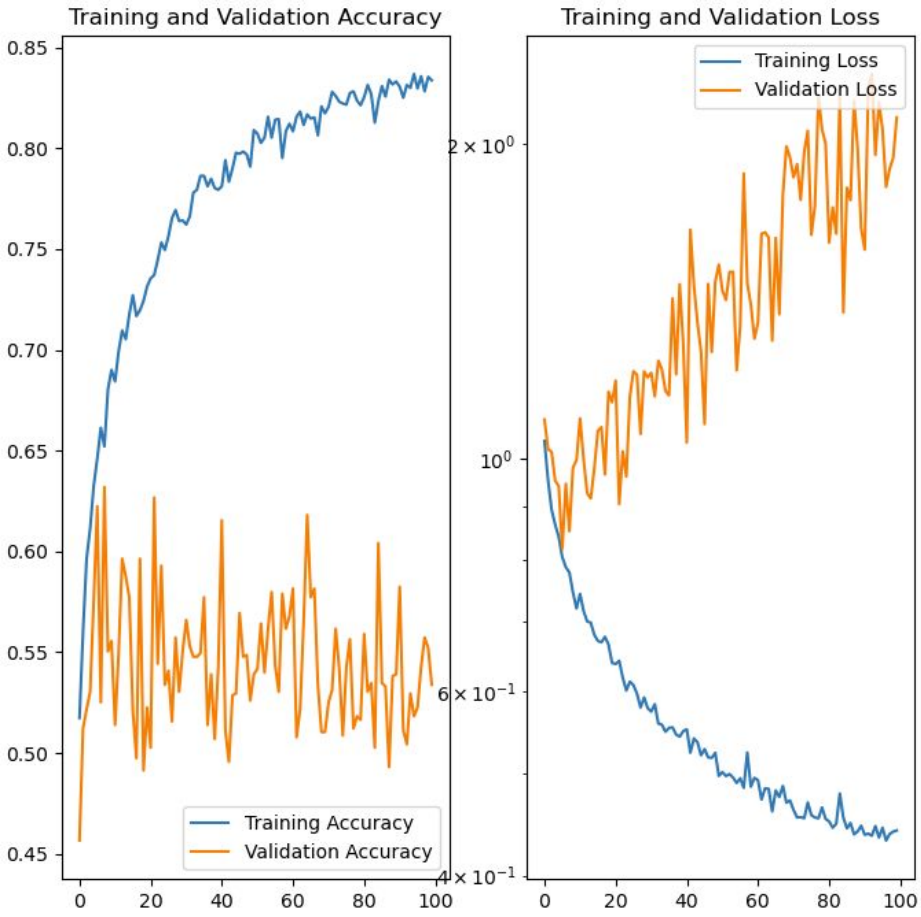
recall : 0.7570573091506958

auc : 0.8843570351600647

Distribution des poids sur la première couche de convolution :



Accuracy et loss pour l’entrainement et la validation :



2)

Optimizer : RMSProp

Activation des couches cachées : ReLu

Activation de la couche de sortie : Sigmoid

Résultat N°1 :

loss : 0.7761348982652029

accuracy : 0.7696969509124756

true_positives : 124.0

false_positives : 32.0

true_negatives : 298.0

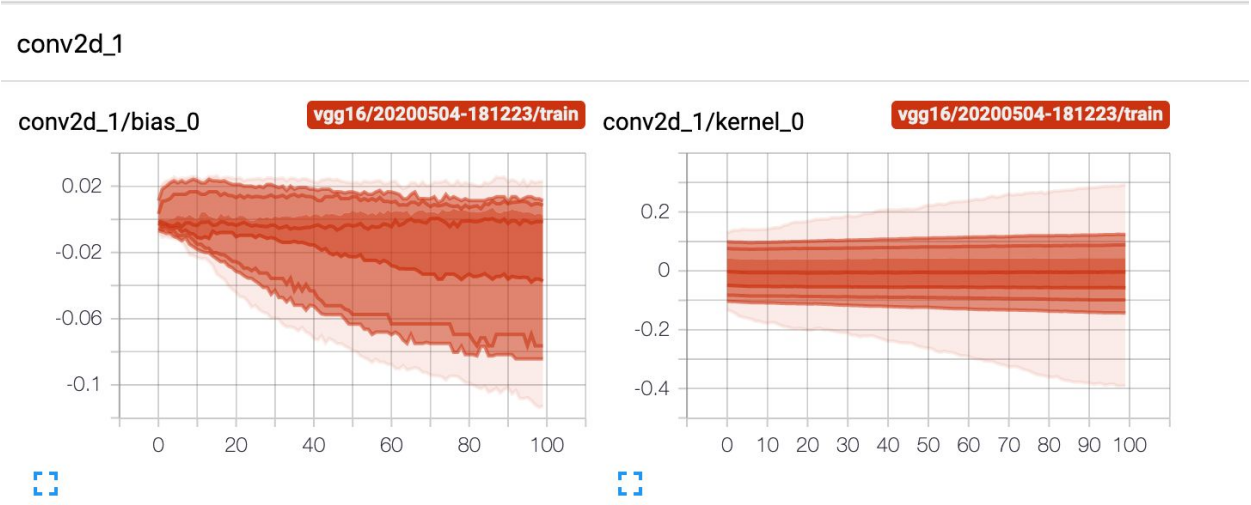
false_negatives : 41.0

precision : 0.7948718070983887

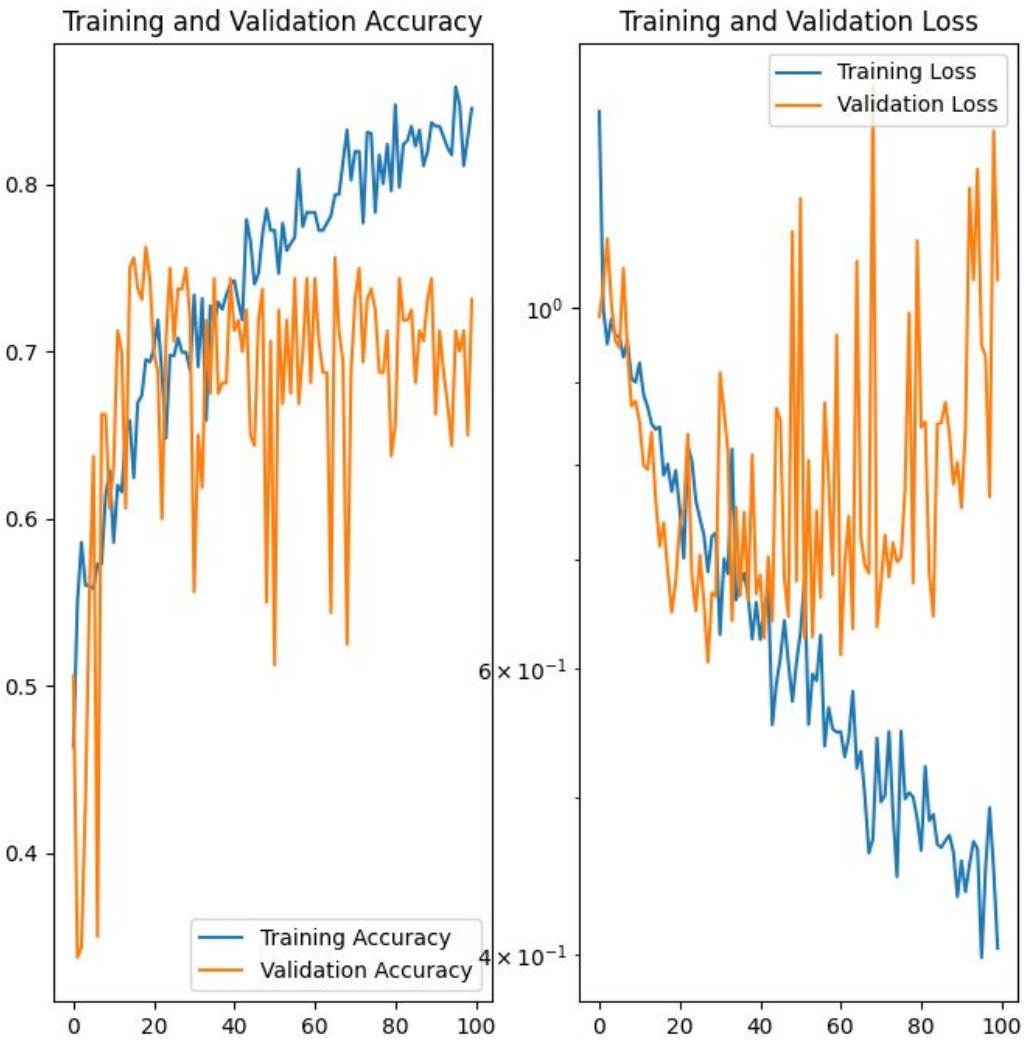
recall : 0.7515151500701904

auc : 0.8879890441894531

Distribution des poids sur la première couche de convolution :



Accuracy et loss pour l’entrainement et la validation :



RESNET18

1)

Optimizer : Adam

Activation des couches cachées : ReLu

Activation de la couche de sortie : Sigmoid

loss : 0.8718771437803904

accuracy : 0.6121212244033813

true_positives : 94.0

false_positives : 55.0

true_negatives : 275.0

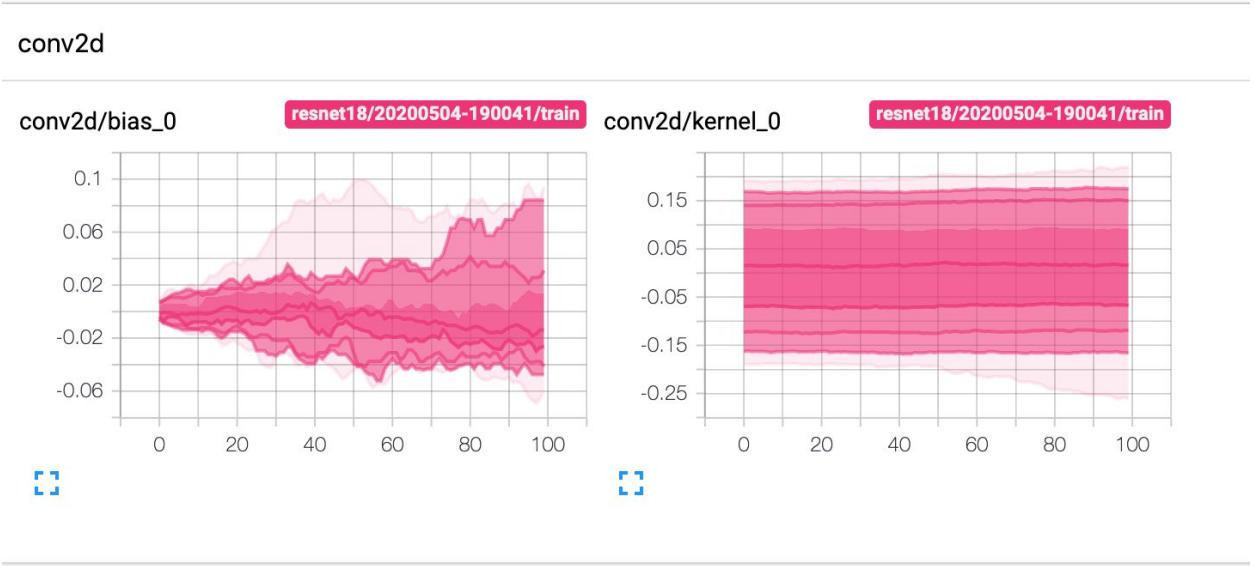
false_negatives : 71.0

precision : 0.6308724880218506

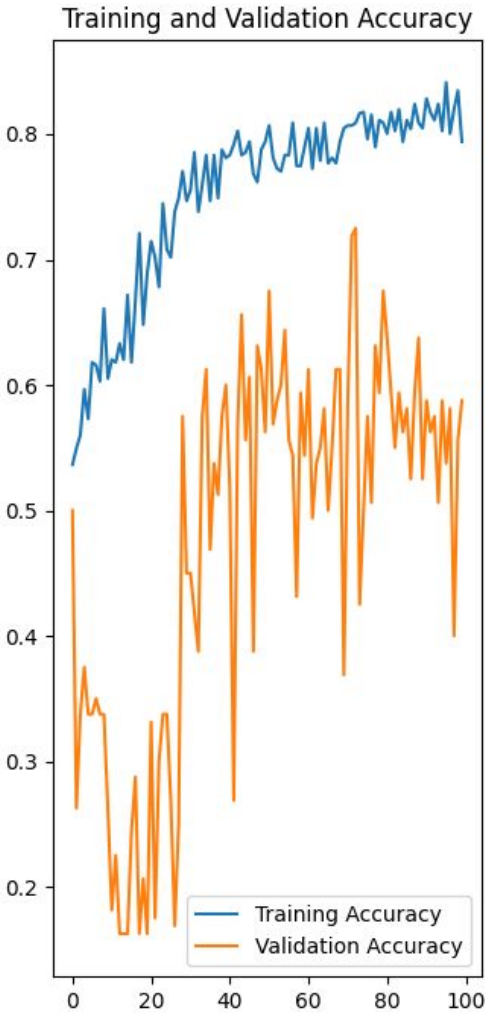
recall : 0.5696969628334045

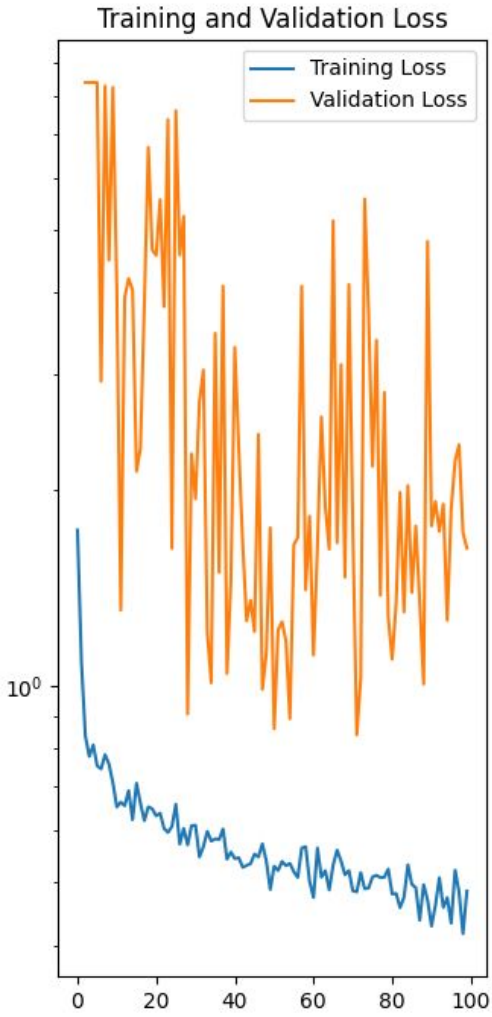
auc : 0.7858034372329712

Distribution des poids sur la première couche de convolution :



Accuracy et loss pour l’entrainement et la validation :





2)

Optimizer : RMSProp

Activation des couches cachées : ReLu

Activation de la couche de sortie : Sigmoid

loss : 0.9877559443314871

accuracy : 0.5636363625526428

true_positives : 86.0

false_positives : 72.0

true_negatives : 258.0

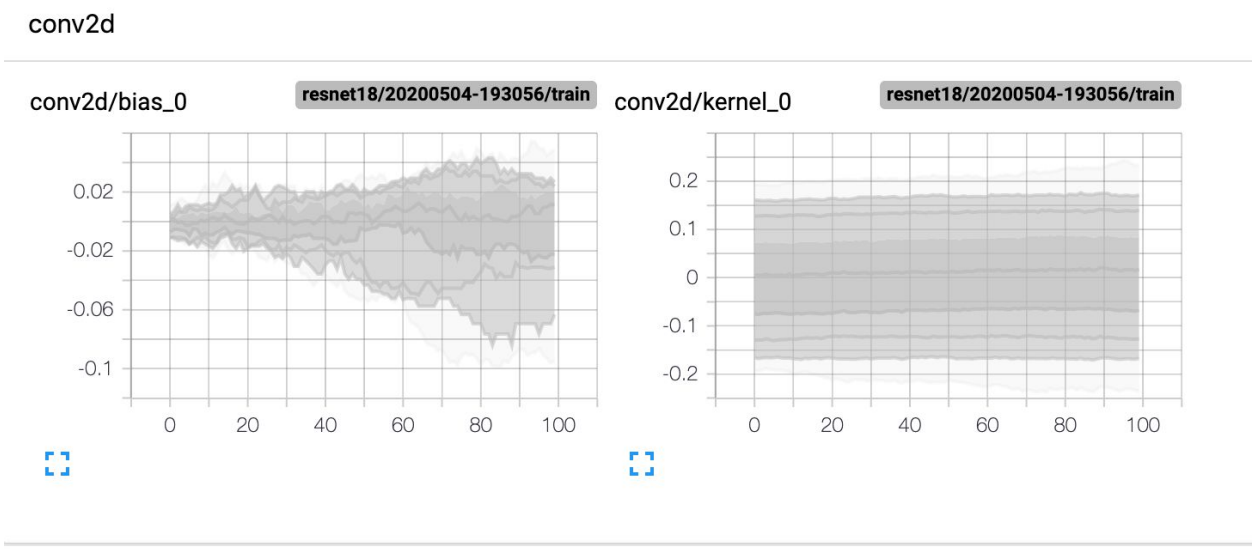
false_negatives : 79.0

precision : 0.5443037748336792

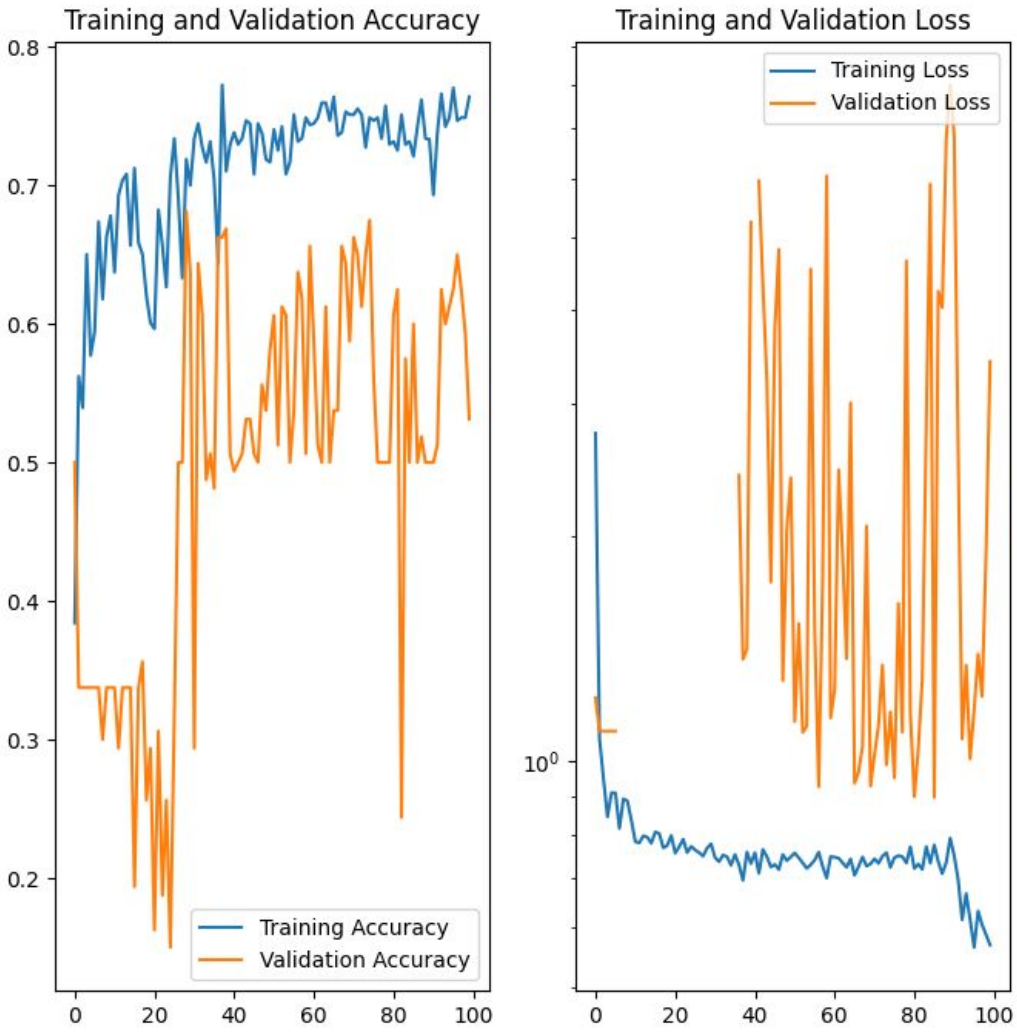
recall : 0.521212100982666

auc : 0.7047658562660217

Distribution des poids sur la première couche de convolution :



Accuracy et loss pour l’entrainement et la validation :



RESNET34

1)

Optimizer : Adam

Activation des couches cachées : ReLu

Activation de la couche de sortie : Sigmoid

loss : 1.0291675925254822

accuracy : 0.521212100982666

true_positives : 85.0

false_positives : 76.0

true_negatives : 254.0

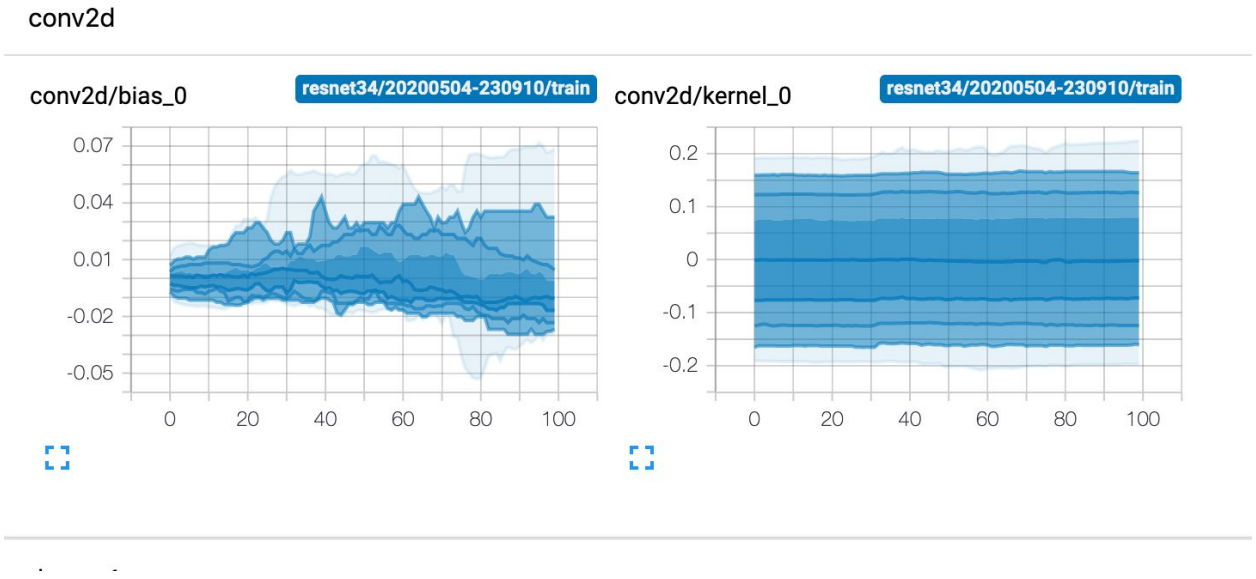
false_negatives : 80.0

precision : 0.5279502868652344

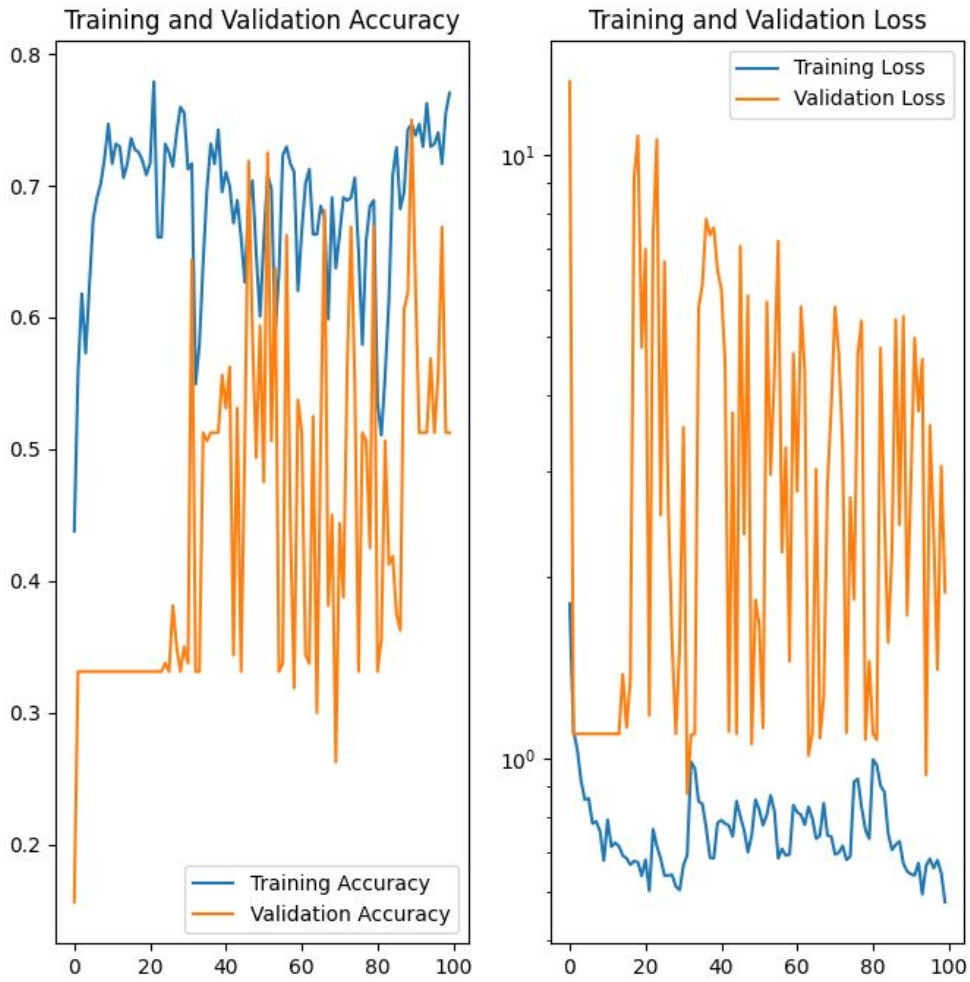
recall : 0.5151515007019043

auc : 0.7336455583572388

Distribution des poids sur la première couche de convolution :



Accuracy et loss pour l’entrainement et la validation :



CONCLUSION

L'apprentissage automatique (en anglais : *machine learning*) est un champ d'étude de l'intelligence artificielle qui se fonde sur des approches mathématiques et statistiques pour donner aux ordinateurs la capacité d'apprendre à partir de données, c'est-à-dire d'améliorer leurs performances à résoudre des tâches sans être explicitement programmés pour chacune. Plus largement, il concerne la conception, l'analyse, l'optimisation, le développement et l'implémentation de telles méthodes.

Ce machine learning nous a permis de diagnostiquer via différents modèles d’apprentissages une maladie de type pneumonie, causée par bactérie ou par virus.

Nous avons pu en apprendre plus sur les technologies utilisées actuellement dans le monde de l’intelligence artificielle et de l’apprentissage profond grâces aux réseaux de neurones. Et malgré le fait que chacun des modèles n’ait été entrainé qu’une seule fois, nous obtenons des résultats convenables, améliorables mais pouvant servir de base à des prédictions en très grande partie correctes sur ce type de maladie.