CentraleSupélec

&

OPENCLASSROOMS

Convolutional Layers, transfert learning, LightLayers: comparaison des performances pour la classification d'images

Diplôme : Data scientist
« Ingénieur Machine Learning »

Présenté par : Xavier Barbier

Résumé

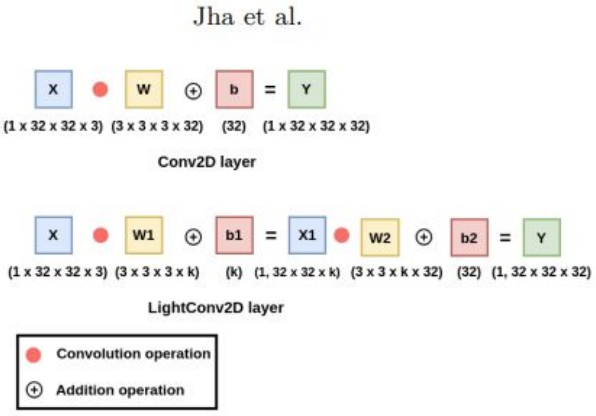
Les techniques d'apprentissage profond ont révolutionné le domaine du Machine Learning (ML). Notamment les réseaux de neurones convolutifs (CNN) pour le traitement d'images. Cependant, les inconvénients des CNN sont qu'ils nécessitent énormément de données, de temps d'entraînement et de stockage. Le but de cette étude était de comparer les performances et les ressources nécessaires à l'entraînement, l'exportation et l'utilisation, de différentes solutions (réseau vierge, apprentissage par transfert, LightLayers) de réseau de neurones convolutifs (CNN). Pour cela un jeu de données d'images médicales a été utilisé pour comparer différents indicateurs. Il en résulte qu'il semblerait possible d'obtenir avec les LightLayers des résultats au moins comparables à un modèle d'apprentissage par transfert, avec une taille représentant seulement 0,5% de ce dernier.

Introduction

Les techniques d'apprentissage profond ont révolutionné le domaine du Machine Learning (ML) et ont attiré une immense attention de la recherche au cours de la dernière décennie.
Une des techniques les plus populaires au sein de l'apprentissage en profondeur est le réseau de neurones convolutifs (CNN), qui possède une structure parfaitement adaptée au traitement d'images et de vidéos [1,2].
Celles-ci ont provoqué une effervescence dans le diagnostic et la gestion des maladies en effectuant des classifications difficiles pour les experts humains et en examinant rapidement d'immenses quantités d'images [3].

Les principaux inconvénients des CNN sont qu'ils nécessitent énormément de données, de temps d'entraînement, et de stockage.
Une méthode pour remédier à ces inconvénients consiste à exploiter les données d'un domaine similaire, une technique connue sous le nom d'apprentissage par transfert. L'apprentissage par transfert s'est avéré être une technique très efficace, en particulier face à des domaines avec des données limitées [4,5,6].
Plutôt que de former un réseau complètement vierge, l'apprentissage par transfert permet d'utiliser un réseau déjà pré-entraîné sur un grand nombre d'images, et déjà optimisé pour reconnaître les structures.
De plus, un long entraînement du modèle est exigeant en termes de ressources car une grande quantité de mémoire est requise, ce qui le rend difficile pour le déployer dans des appareils basse consommation [7].

Dans le cadre de la réduction du coût de l'entraînement du modèle, les réseaux légers ont attiré l'attention [8,9,10,11,12]. Certaines études s'appuyant notamment sur des méthodes de décomposition [13,14,15,16,17,18,19,20].
Dans ce contexte d'études indiquant qu'il existe un grand potentiel d'utilisation de réseaux légers pour les tâches de vision par ordinateur, Jha & col [21] ont introduit les LightLayers (Couches légères). Les auteurs proposent de décomposer la matrice de poids W dans W1 et W2 sur la base de l'hyperparamètre k. Ce qui conduit à une réduction du nombre total de paramètres entraînables dans le réseau en comparaison avec des couches de convolution classiques [25]. Ceci sans dégrader la performance.



Le code des différents LightLayers est disponible sur l'adresse suivante : https://github.com/DebeshJha/LightLayers

Disposant de plusieurs solutions (réseau vierge, apprentissage par transfert, LightLayers) dans l'optique de réaliser des tâches de classification d'images, il est apparu judicieux de comparer les performances de celles-ci.

Méthode

1. Données

Le jeu de données utilisé est disponible sur Kaggle et est constitué de radiographies du thorax. Elles sont issues d'un article [3] explorant l'utilisation de l'apprentissage profond par transfert pour l'imagerie médicale. Les données utilisées sont également disponibles dans un ensemble plus important d'imageries [22].
Le jeu de données est composé de 5856 images de patients sains ou présentant des pneumonies.

Les images se décomposent en 3 parties : Entrainement (5216 images), Validation (624) et Test (16).

2. Protocole

Le protocole se scinde en 3 parties :

- reproduction d'une étude [3] utilisant l'apprentissage par transfert (Inception V3)
- utilisation des LightLayers (LightConv2D pour lesquels différentes valeurs de k seront testées (1-5)
- construction d'une architecture similaire mais à partir de couches de convolution classiques (voir annexe)

Nous avons implémenté des couches proposées en utilisant le framework Keras [23] et TensorFlow 2.4 [24]. Nous avons effectué toutes les expériences sur un notebook Kaggle pour lequel le GPU avait été activé. Afin de nous rapprocher au plus possible des conditions de l'étude introduisant les LightLayes [21], une taille des lots de 64 a été utilisée et les modèles ont été entraînés sur 20 époques en utilisant un learning rate de $1e-3$. Une normalisation par lots est utilisée après chaque couche de convolution, ainsi qu'une activation par unité linéaire rectifiée (ReLU). Enfin, le meilleur modèle pour chaque architecture a été sauvegardé grâce à un rappel automatique.

L'évaluation des modèles a été faite à partir des critères suivants:

- Nombre de paramètres du modèle

- Taille du modèle
- Accuracy (test set)
- Temps total de calcul
- Temps de calcul par époques

Résultats

Les courbes d'évolution de accuracy et loss pour les jeux d'entraînement et de validation sont disponibles en annexe (fig.1-7). Les résultats sont disponibles dans le tableau 1.

	Model	Total params	Trainable params	Size	Accuracy	Total time	Time per epoch
0	LightLayers (K = 1)	2753	2417	0.251328	0.717949	1268.383362	63.419168
1	LightLayers (K = 2)	4600	4264	0.401146	0.839744	1291.755693	64.587785
2	LightLayers (K = 3)	6447	6111	0.562813	0.834936	1309.051035	65.452552
3	LightLayers (K = 4)	8294	7958	0.739403	0.849359	1319.003186	65.950159
4	LightLayers (K = 5)	10141	9805	0.930847	0.834936	1333.586277	66.679314
5	Conv2D	31074	30738	1.300407	0.621795	1076.503830	53.825191
6	transfert_learning	21806882	4098	84.690468	0.799679	1075.007121	53.750356

Tableau 1: résultats des différents modèles entraînés.

Les graphiques par critères d'évaluation sont disponibles ci-dessous (fig. 8-13)

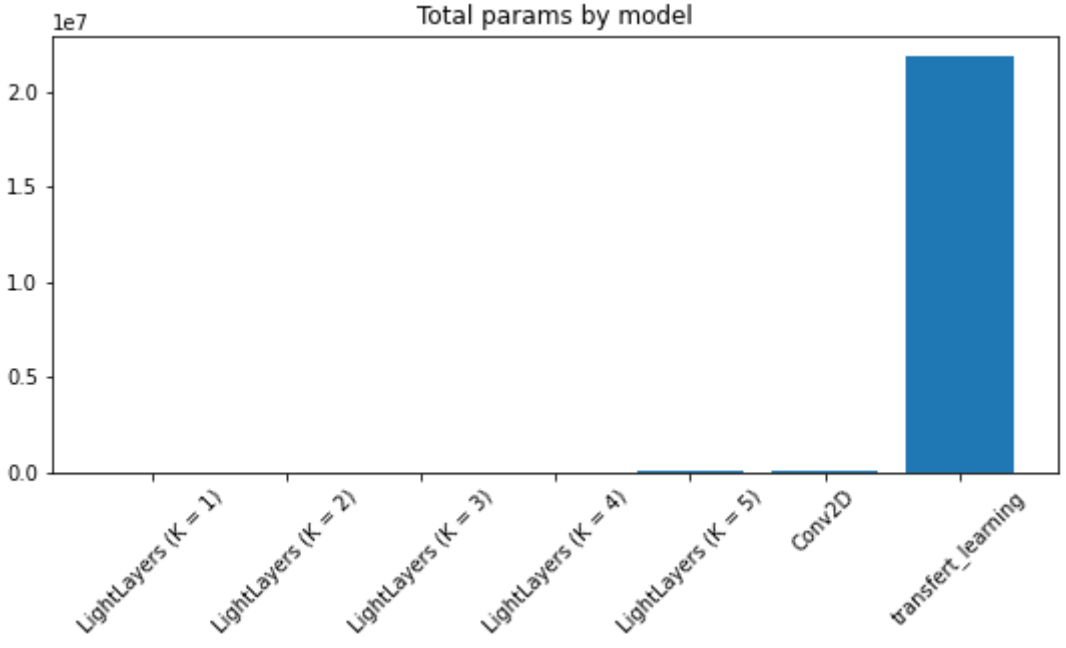


Fig.8: Nombre de paramètres par modèles

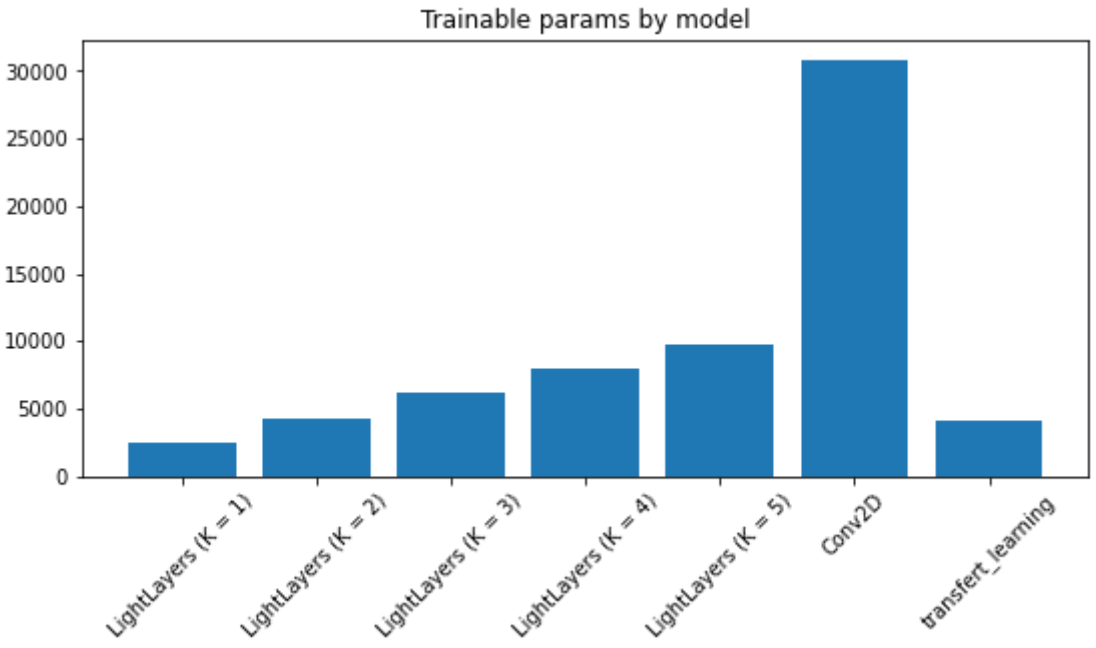


Fig.9: Nombre de paramètres entraînables par modèles.

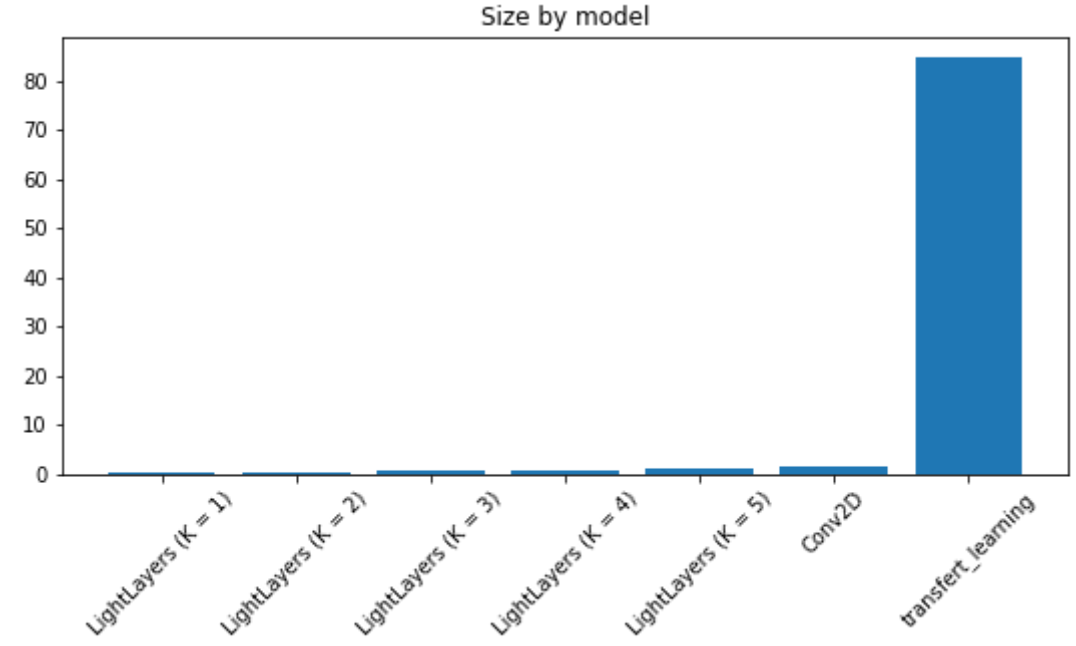


Fig.10: Taille par modèle (Mb).

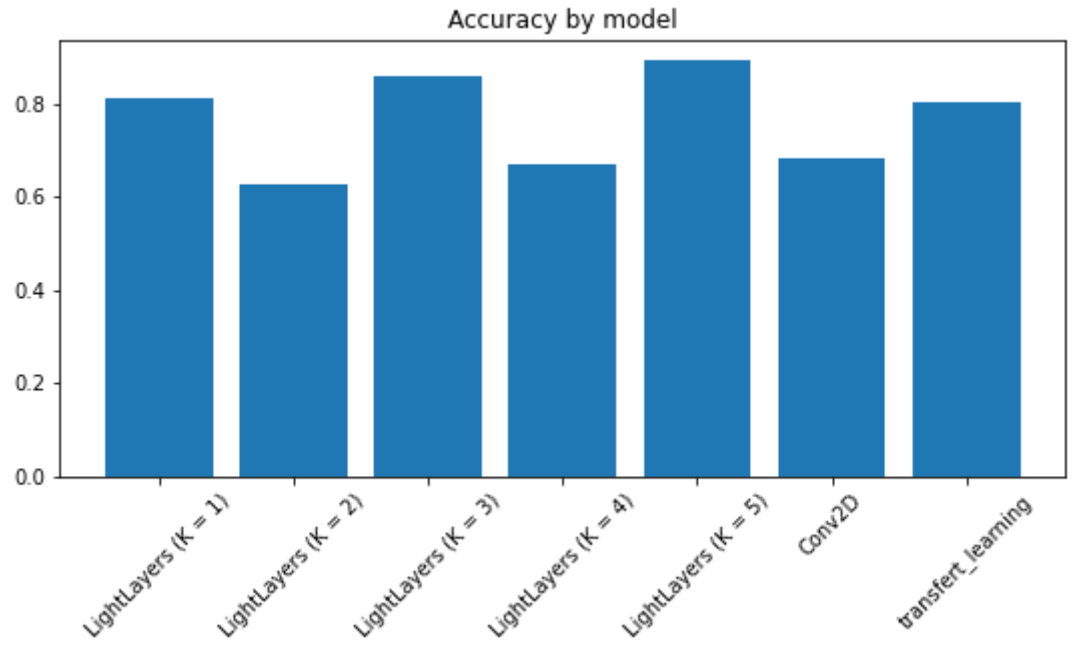


Fig.11: Accuracy score par modèles

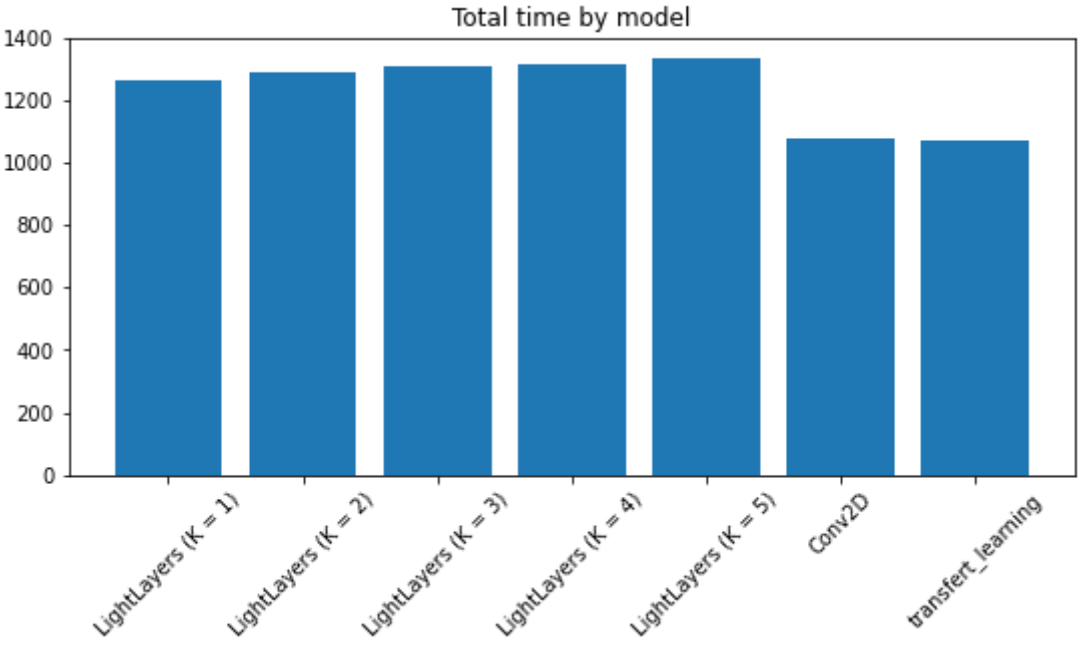


Fig.12: Temps d'entraînement total par modèles

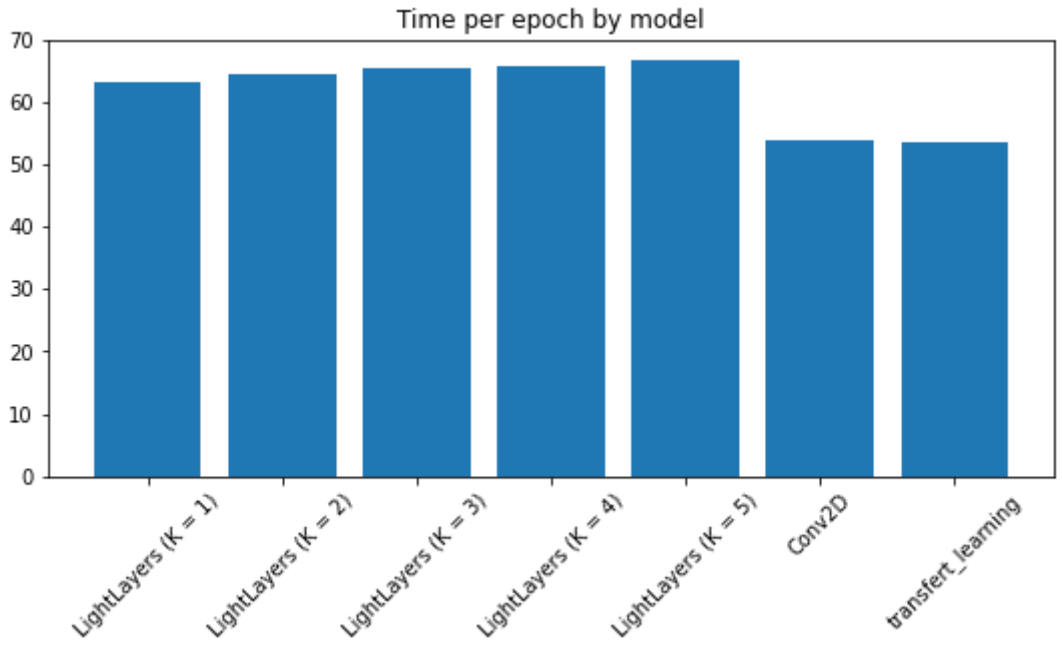


Fig.13: Temps d'entraînement par époques par modèle

Le modèle utilisant l'apprentissage par transfert est celui nécessitant le plus de paramètres (+21 millions). Très loin devant le second, Conv2D (31 000). Suivi par les LightLayers (5,4,3,2,1).
Le modèle Conv2D est celui utilisant le plus de paramètres entraînables (30 000). Suivi par les modèles LightLayers (k = 5,4,3,2). Les modèles de transfert d'apprentissage et LightLayers (k = 1) complètent dans cet ordre le classement.
Tous les modèles LightLayers présentent des durées par époques et des durées totales d'entraînement supérieures aux modèles Conv2D et d'apprentissage par transfert.
Le meilleur score de justesse (accuracy) est obtenu pour le LightLayers (k=5) avec 0,89. Suivi par les LightLayers (k = 3 et 1) avec 0,86 et 0,81. Le modèle d'apprentissage par transfert arrive en quatrième position avec un score de 0,80. Viennent enfin le modèle Conv2D avec 0,68 , le modèle LightLayers (k=4) avec 0,66, puis enfin le modèle LightLayers (k=2).
Concernant la taille du modèle, le modèle d'apprentissage par transfert présente une taille de 84 Mb. Le modèle Conv2D présente une taille de 1,3 Mb. Tous les modèles de LightLayers (k = 1,2,3,4,5) présentent des tailles inférieures à 1Mb (0,25 - 0,40 - 0,56 - 0,73 - 0,93).

Discussion

Le but de cette étude était de comparer les performances et les ressources nécessaires à l'entraînement, l'exportation et l'utilisation de différentes solutions (réseau vierge, apprentissage par transfert, LightLayers) de réseau de neurones convolutifs (CNN) . Architectures particulièrement adaptées et performantes pour le traitement d'images et de vidéos.

Le principal aspect est qu'il semblerait être possible d'obtenir avec des modèles LightLayers des performances supérieures ou similaires au transfert d'apprentissage. Les modèles LightLayers ayant pour avantage de représenter 0,5% de la taille d'une modèle d'apprentissage par transfert (< 1 Mb).
Ces résultats sont proches de ceux obtenus par Jha & col. [21].
Il serait intéressant d'utiliser la même approche avec des jeux de données présentant plus que 2 classes. En effet, Jah & col. [21] obtenant des performances sensiblement égales pour les jeux MNIST (10 classes) et Fashion MNIST (10 classes), mais un très importante dégradation des performances pour le jeu CIFAR100 (100 classes).

Enfin, dans le cadre du jeu de données utilisé se pose la problématique de la séparation entraînement, validation et test. Le jeu de test étant constitué d'un faible nombre d'observations (n = 16), il est possible de douter de sa pertinence pour évaluer le modèle.
De plus, il est curieux de constater un décalage entre les performances des modèles sur les jeux de validation et de test. Le lightConv2D (k = 3) obtient un score de 1,0 sur le jeu de validation mais 0,86 sur le jeu de test (fig.3 en annexe). A l'inverse, le lightConv2D (k = 5) obtient un score de 0,86 sur le jeu de validation mais 0,89 sur le jeu de test (fig.5 en annexe).
Il serait intéressant d'utiliser la même approche avec une séparation plus classique du jeu de données (70/30 par exemple).
Pour finir, face à un sur-apprentissage généralisé sur l'ensemble de modèles testés, il serait pertinent d'utiliser des techniques connues pour limiter celui-ci (augmentation d'image, dropout).

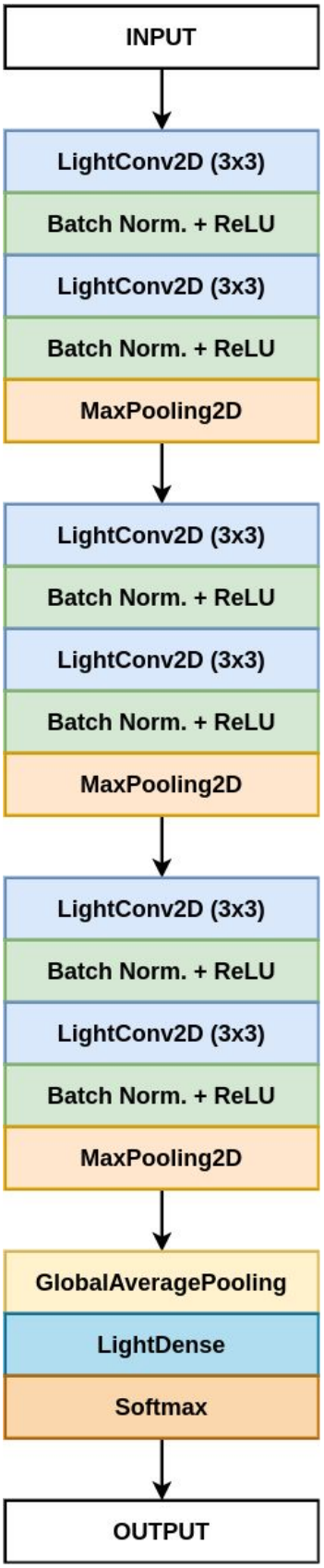
Références

1. Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks." Communications of the ACM 60.6 (2017): 84-90.
2. Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." European conference on computer vision. Springer, Cham, 2014.
3. Kermany, D. S., Goldbaum, M., Cai, W., Valentim, C. C., Liang, H., Baxter, S. L., ... & Zhang, K. (2018). Identifying medical diagnoses and treatable diseases by image-based deep learning. Cell, 172(5), 1122-1131.
4. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2013). DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. Proceedings of the 31st International Conference on Machine Learning 32, 647–655.
5. Razavian, A.S., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. In 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 512–519.
6. Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems 2, 3320–3328.

7. Kim, Y.D., Park, E., Yoo, S., Choi, T., Yang, L., Shin, D.: Compression of deep convolutional neural networks for fast and low power mobile applications. arXiv preprint arXiv:1511.06530 (2015)
8. Xue, J., Li, J., Gong, Y.: Restructuring of deep neural network acoustic models with singular value decomposition. In: Interspeech. pp. 2365–2369 (2013)
9. Li, J., Zhao, R., Huang, J.T., Gong, Y.: Learning small-size dnn with output-distribution-based criteria. In: Proceedings of the conference of the international speech communication association (2014)
10. Xue, J., Li, J., Yu, D., Seltzer, M., Gong, Y.: Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network. In: Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 6359–6363 (2014)
11. Garipov, T., Podoprikhin, D., Novikov, A., Vetrov, D.: Ultimate tensorization: compressing convolutional and fc layers alike. arXiv preprint arXiv:1611.03214 (2016)
12. Kim, H., Sim, J., Choi, Y., Kim, L.S.: A kernel decomposition architecture for binary-weight convolutional neural networks. In: Proceedings of the Annual Design Automation Conference. pp. 1–6 (2017)
13. Wu, C.W.: Prodsumnet: reducing model parameters in deep neural networks via product-of-sums matrix decompositions. arXiv preprint arXiv:1809.02209 (2018)
14. Agarwal, P., Alam, M.: A lightweight deep learning model for human activity recognition on edge devices. arXiv preprint arXiv:1909.12917 (2019)
15. Denton, E.L., Zaremba, W., Bruna, J., LeCun, Y., Fergus, R.: Exploiting linear structure within convolutional networks for efficient evaluation. In: Advances in neural information processing systems. pp. 1269–1277 (2014)
16. Lebedev, V., Ganin, Y., Rakhuba, M., Oseledets, I., Lempitsky, V.: Speeding-up convolutional neural networks using fine-tuned cpdecomposition. arXiv preprint arXiv:1412.6553 (2014)
17. Kim, Y.D., Park, E., Yoo, S., Choi, T., Yang, L., Shin, D.: Compression of deep convolutional neural networks for fast and low power mobile applications. arXiv preprint arXiv:1511.06530 (2015)
18. Mariet, Z., Sra, S.: Diversity networks: Neural network compression using determinantal point processes. arXiv preprint arXiv:1511.05077 (2015)
19. Novikov, A., Podoprikhin, D., Osokin, A., Vetrov, D.P.: Tensorizing neural networks. In: Advances in neural information processing systems. pp. 442– 450 (2015)
20. Oseledets, I.V.: Tensor-train decomposition. SIAM Journal on Scientific Computing 33(5), 2295–2317 (2011)
21. Jha, D., Yazidi, A., Riegler, M. A., Johansen, D., Johansen, H. D., & Halvorsen, P. (2021). LightLayers: Parameter Efficient Dense and Convolutional Layers for Image Classification. arXiv preprint arXiv:2101.02268.
22. Kermany, Daniel; Zhang, Kang; Goldbaum, Michael (2018), “Large Dataset of Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images”, Mendeley Data, V3, doi: 10.17632/rscbjbr9sj.3
23. Chollet, F., et al.: Keras (2015), https://keras.io
24. Abadi, M., Barham, P., et al.: Tensorflow: A system for large-scale machine learning. In: Proceedings of the Symposium on Operating Systems Design and Implementation (OSDI). pp. 265–283 (2016)

25. Sainath, Tara & Kingsbury, Brian & Sindhwani, Vikas & Arisoy, Ebru & Ramabhadran, Bhuvana. (2013). Low-rank matrix factorization for Deep Neural Network training with high-dimensional output targets. Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on. 6655-6659. 10.1109/ICASSP.2013.6638949.

Annexes



Architecture CNN

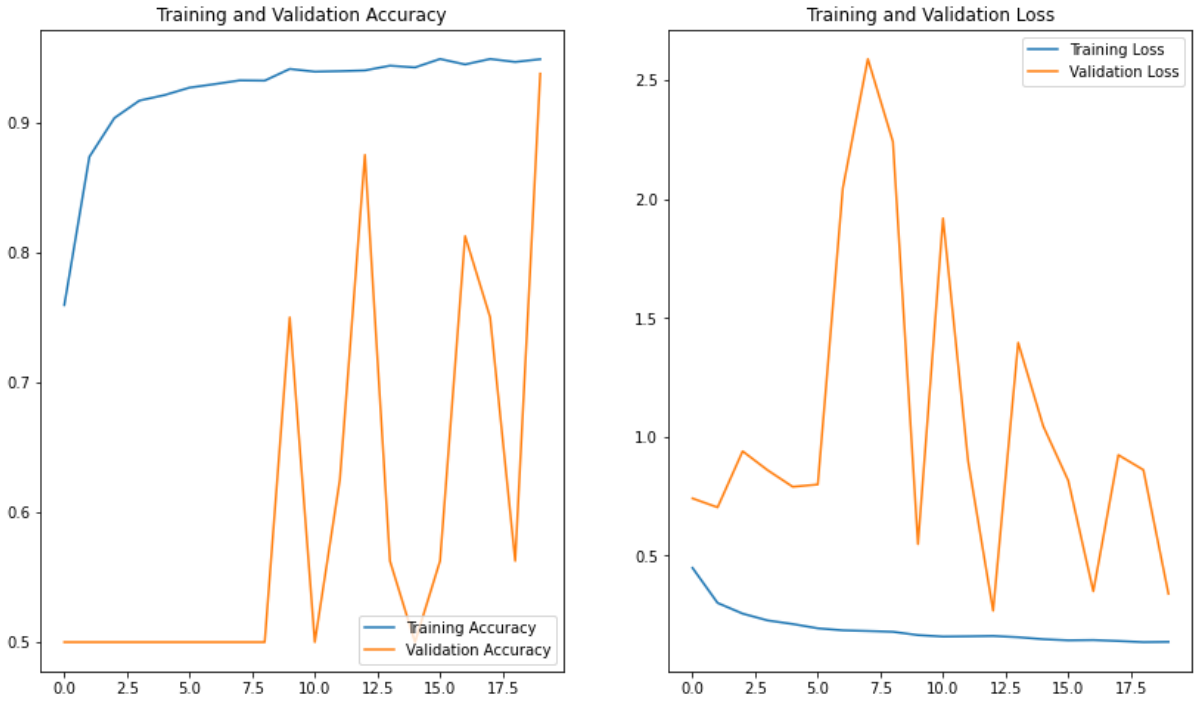


Fig.1: History LightLayers (k=1)

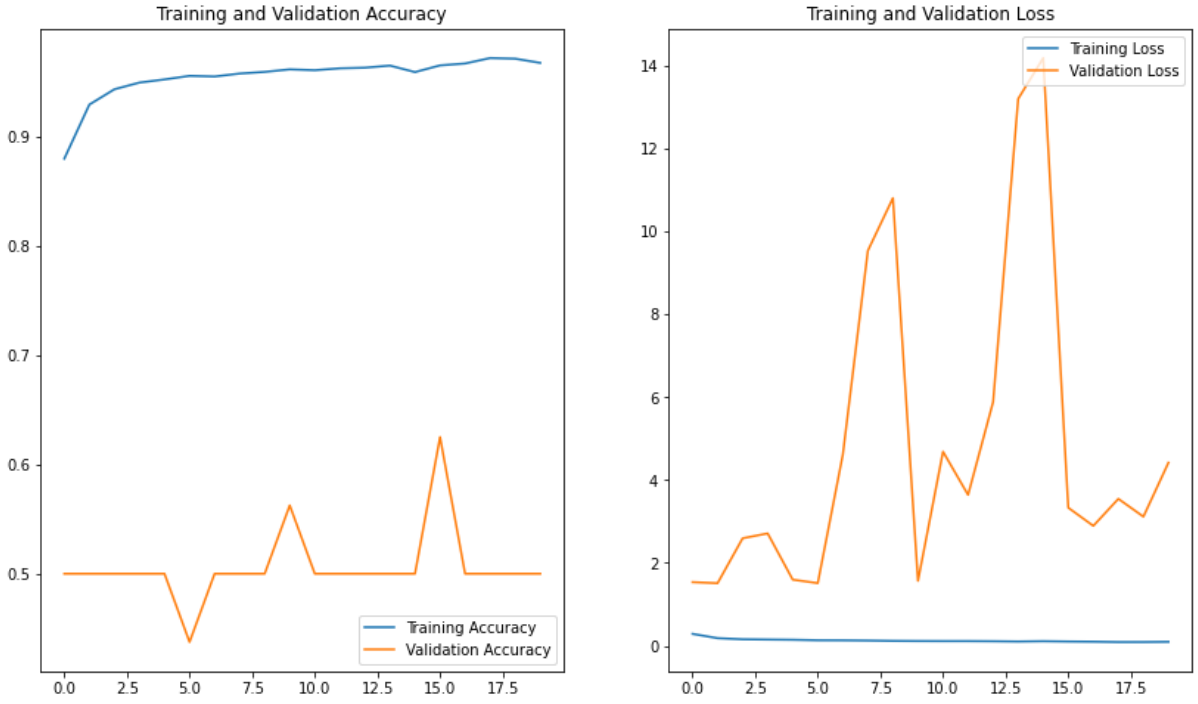


Fig.2: History LightLayers (k=2)

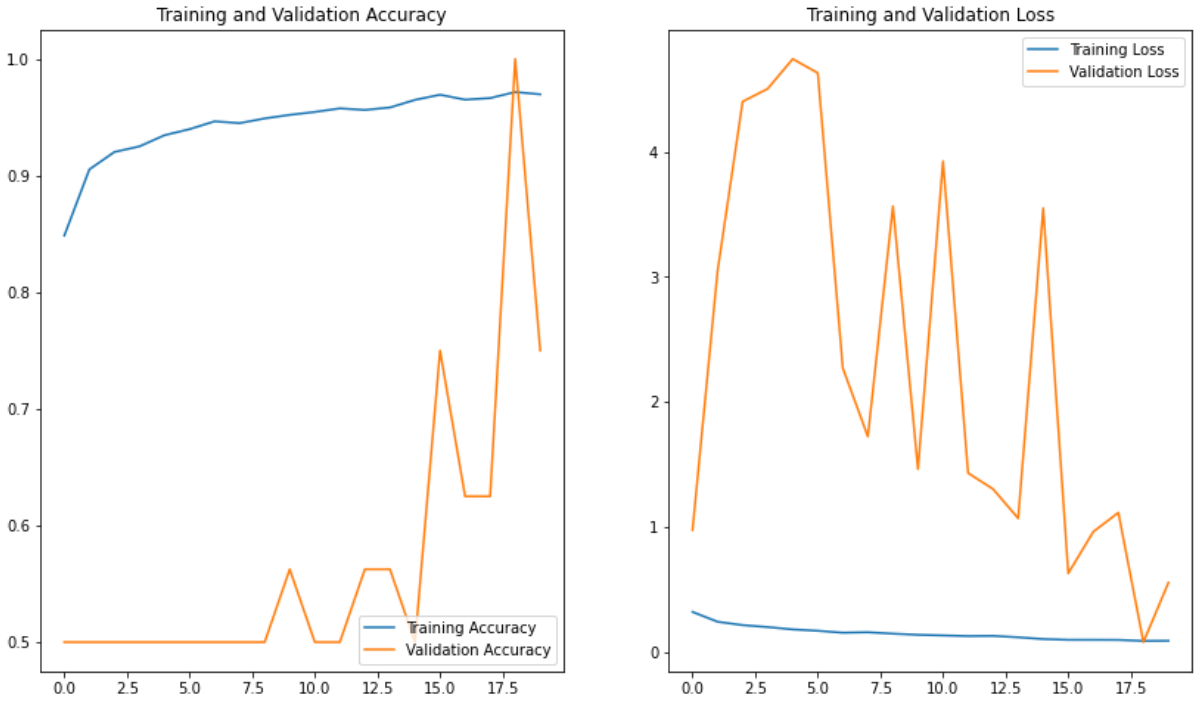


Fig.3: History LightLayers (k=3)

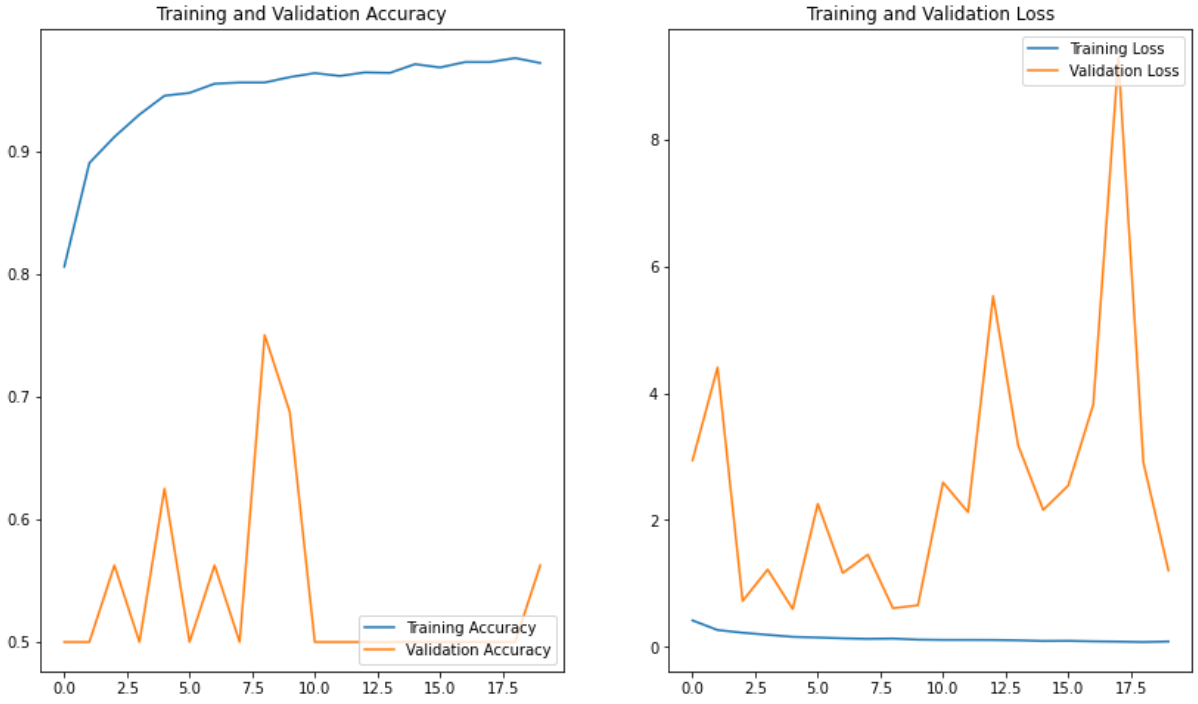


Fig.4: History LightLayers (k=4)

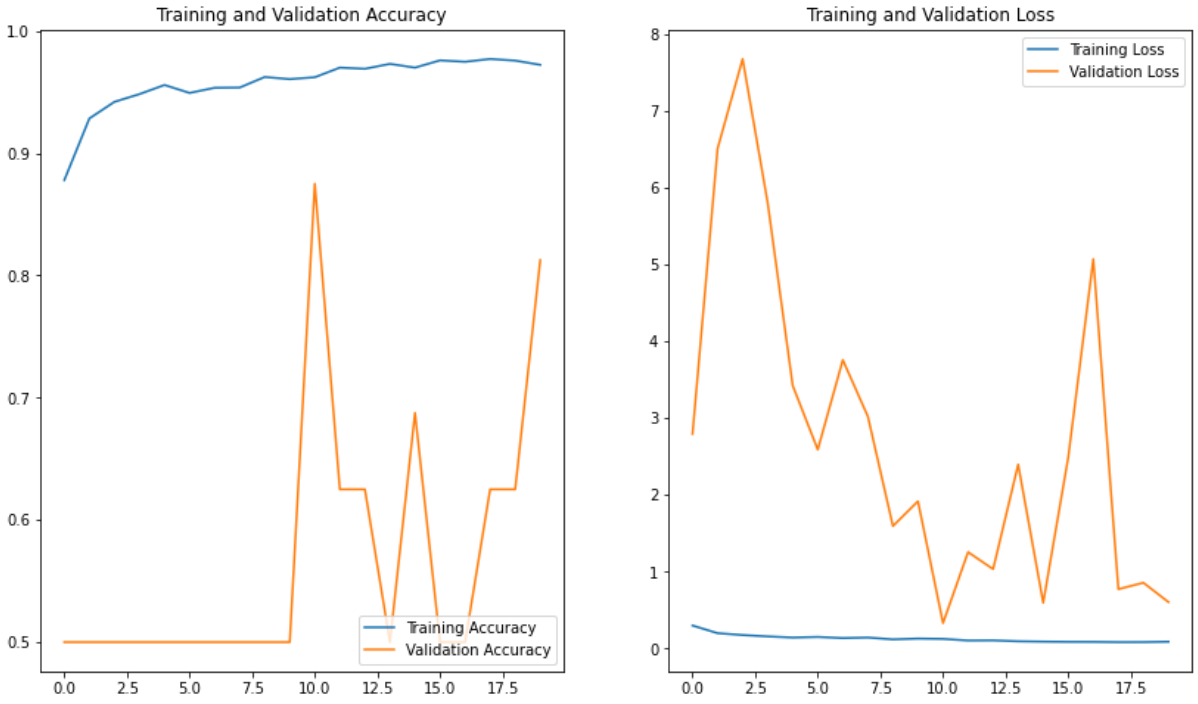


Fig.5: History LightLayers (k=5)

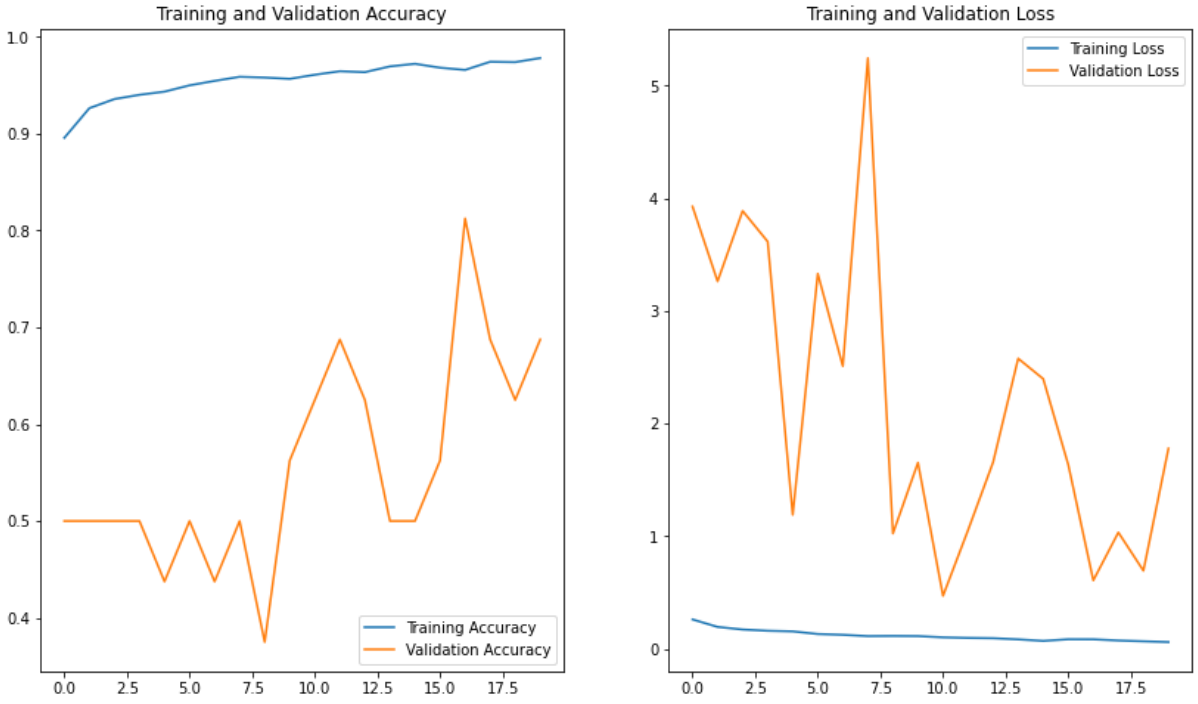


Fig.6: Conv2D

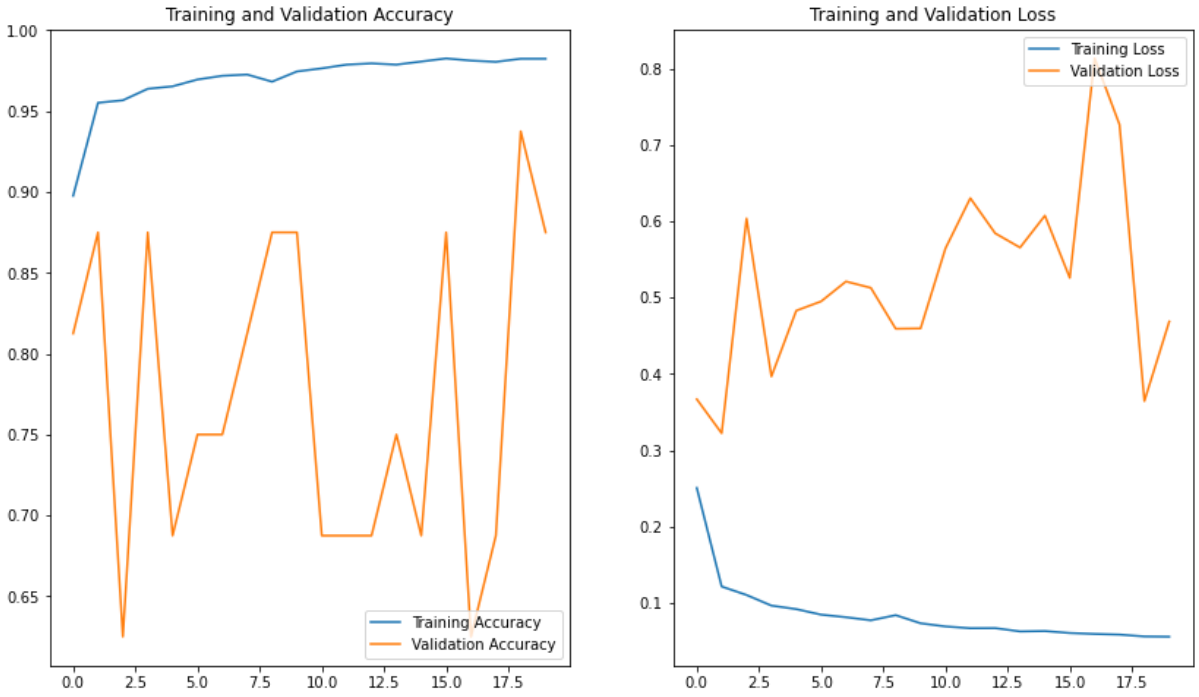


Fig.7: Transfert learning (Inception V3)