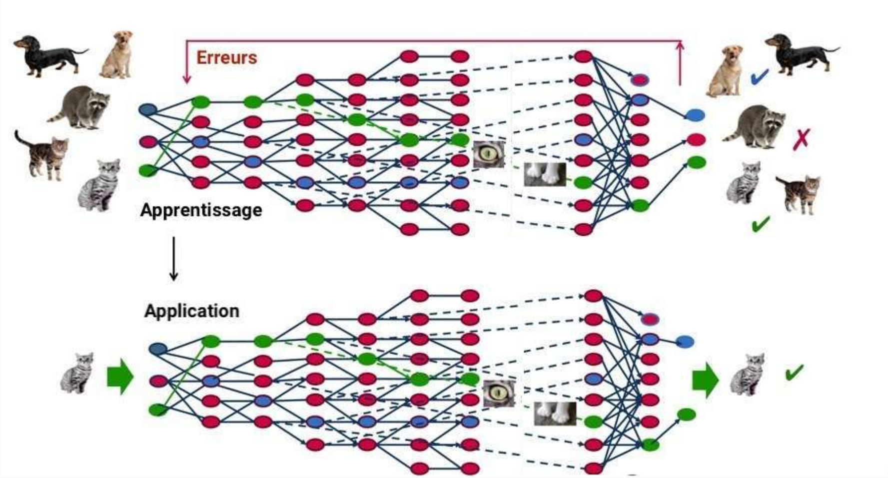
Convolutional Layers, transfert learning, LightLayers: comparaison des performances pour la classification d’images

Xavier Barbier - Mars 2021

Sommaire

1. Contexte
2. Projet
3. Données
4. Protocole
5. Résultats
6. Conclusion

Contexte:



Resource

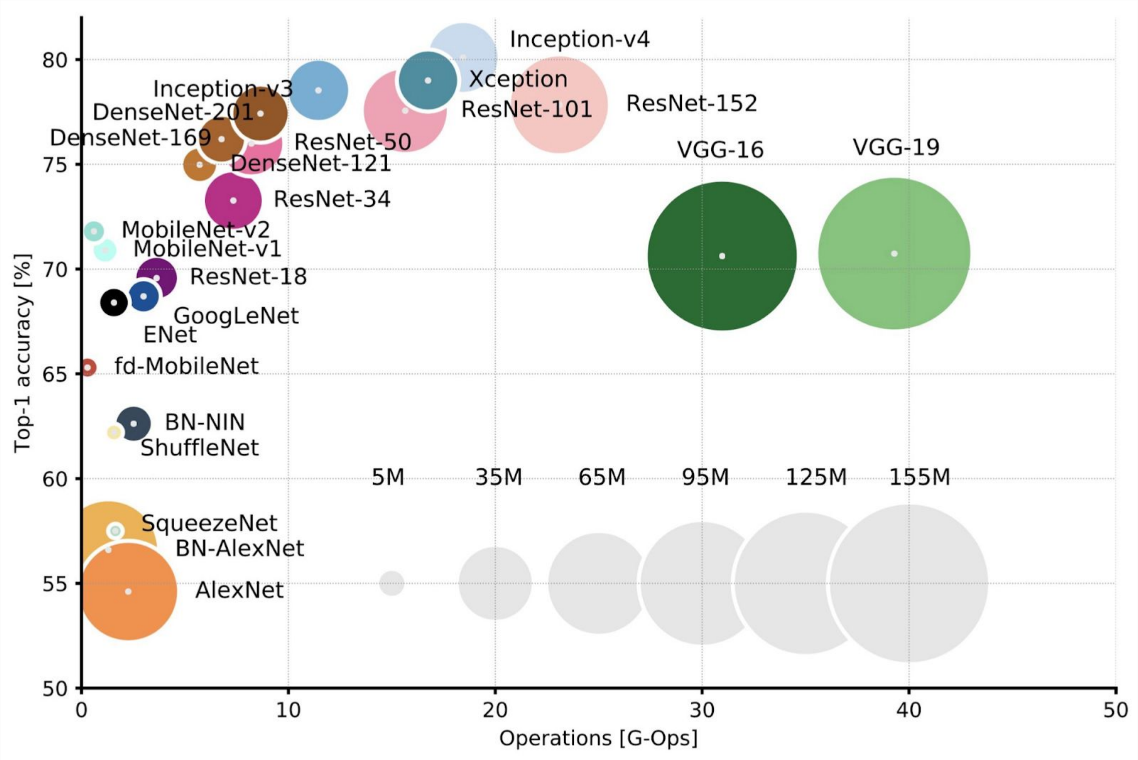
Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning

Daniel S. Kermany [1, 2, 14], Michael Goldbaum [2, 14], Wenjia Cai [2, 14], Carolina C.S. Valentim [2, 14], Huiying Liang [1, 14], Sally L. Baxter [2, 14], Alex McKeown [3], Ge Yang [2], Xiaokang Wu [4], Fangbing Yan [4], Justin Dong [1], Made K. Prasadha [2], Jacqueline Pei [1, 2], Magdalene Y.L. Ting [2], Jie Zhu [1, 5], Christina Li [2], Sierra Hewett [1, 2], Jason Dong [1] ... Kang Zhang [1, 2, 4, 12, 13, 15]

CNN

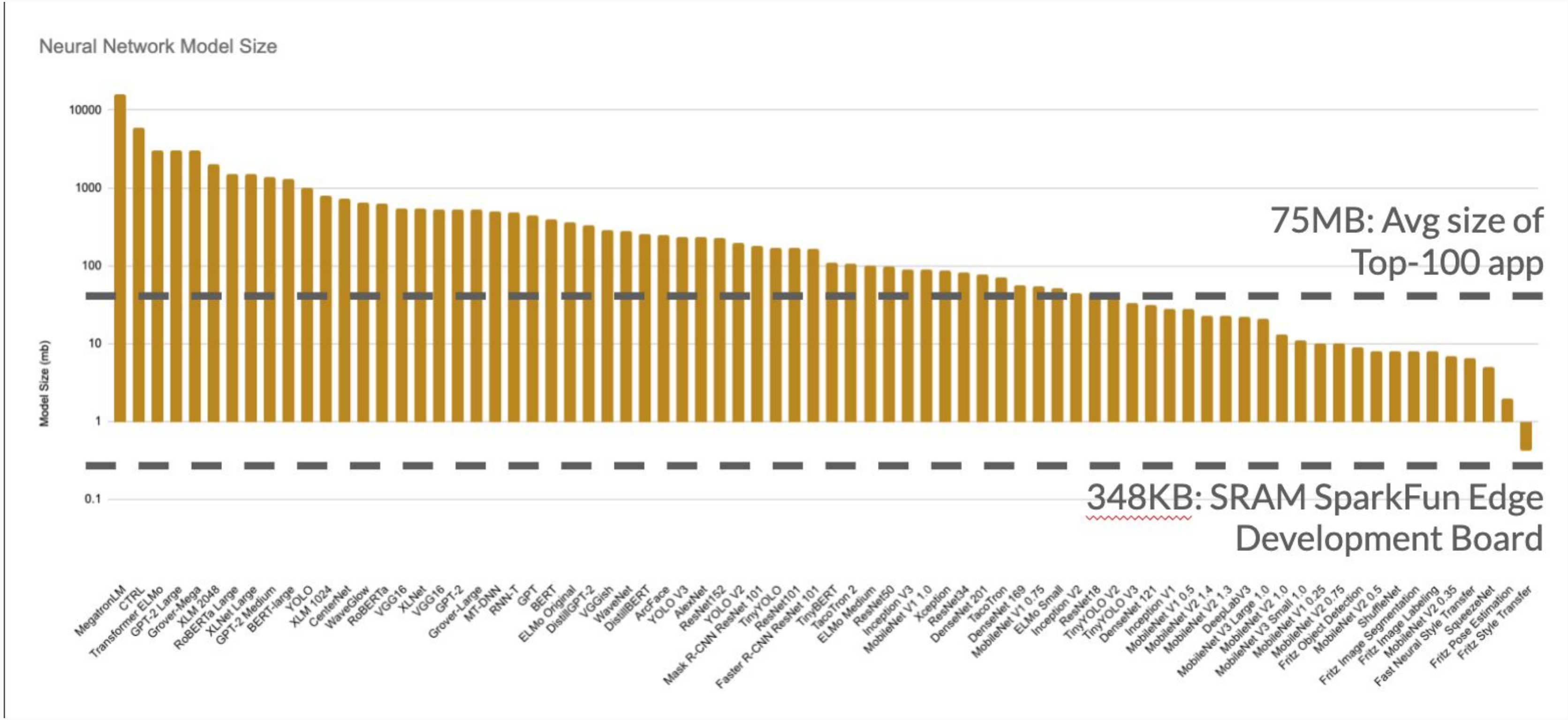
Transfert learning

Contexte:



Source : https://heartbeat.fritz.ai/deep-learning-has-a-size-problem-ea601304cd8

Contexte:



Source : https://heartbeat.fritz.ai/deep-learning-has-a-size-problem-ea601304cd8

Contexte:

[Submitted on 6 Jan 2021]

LightLayers: Parameter Efficient Dense and Convolutional Layers for Image Classification

Debesh Jha, Anis Yazidi, Michael A. Riegler, Dag Johansen, Håvard D. Johansen, Pål Halvorsen

Deep Neural Networks (DNNs) have become the de-facto standard in computer vision, as well as in many other pattern recognition tasks. A key drawback of DNNs is that the training phase can be very computationally expensive. Organizations or individuals that cannot afford purchasing state-of-the-art hardware or tapping into cloud-hosted infrastructures may face a long waiting time before the training completes or might not be able to train a model at all. Investigating novel ways to reduce the training time could be a potential solution to alleviate this drawback, and thus enabling more rapid development of new algorithms and models. In this paper, we propose LightLayers, a method for reducing the number of trainable parameters in deep neural networks (DNN). The proposed LightLayers consists of LightDense andLightConv2D layer that are as efficient as regular Conv2D and Dense layers, but uses less parameters. We resort to Matrix Factorization to reduce the complexity of the DNN models resulting into lightweight DNNmodels that require less computational power, without much loss in the accuracy. We have tested LightLayers on MNIST, Fashion MNIST, CI-FAR 10, and CIFAR 100 datasets. Promising results are obtained for MNIST, Fashion MNIST, CIFAR-10 datasets whereas CIFAR 100 shows acceptable performance by using fewer parameters.

Table 1: Results on **MNIST** test dataset (Number of epochs = 10, Batch size = 64, Learning rate = $1e-3$, Number of filters = $[8, 16, 32]$).

Method	Parameters	Test Accuracy	Test Loss
Conv2D	18,818	0.9887	0.018
SeparableConv2D	3,611	0.9338	0.2433
LightLayers ($K = 1$)	2,649	0.9418	0.1327
LightLayers ($K = 2$)	4,392	0.9749	0.0554
LightLayers ($K = 3$)	**6,135**	**0.9775**	**0.0513**
LightLayers ($K = 4$)	7,878	0.9720	0.0704

Table 2: Results on **Fashion MNIST** test dataset (Number of epochs = 10, Batch size = 64, Learning rate = $1e-3$, Number of filters = $[8, 16, 32]$).

Method	Parameters	Test Accuracy	Test Loss
Conv2D	18,818	0.9147	0.1468
SeparableConv2D	3,611	0.8725	0.3175
LightLayers ($K = 1$)	2,649	0.789	0.6752
LightLayers ($K = 2$)	4,392	0.8452	0.4247
LightLayers ($K = 3$)	6,135	0.8695	0.3708
LightLayers ($K = 4$)	7,878	0.8623	0.6184
LightLayers ($K = 5$)	**9,621**	**0.8820**	**0.2810**
LightLayers ($K = 6$)	11,364	0.8733	0.3986

Table 4: Evaluation on **CIFAR100** test set (Number of epochs = 20, Batch size = 64, Learning rate = $1e-4$, Number of filters = $[8, 16, 32, 64]$).

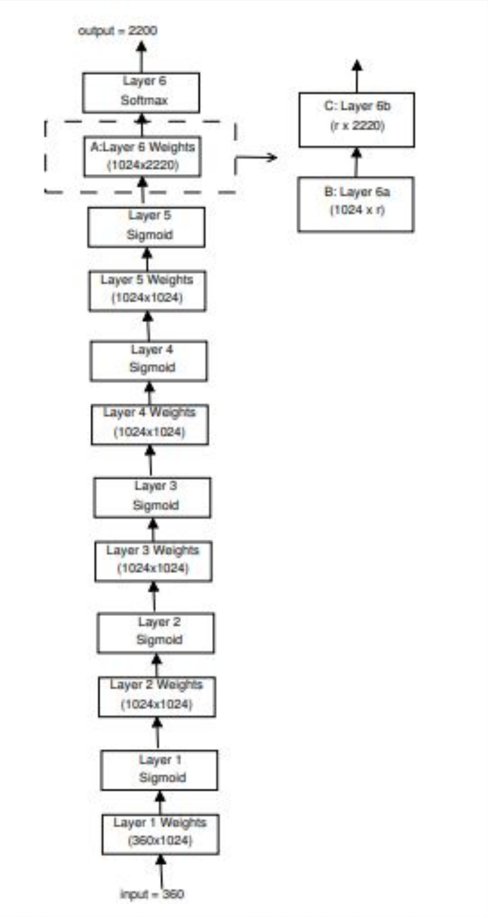
Method	Parameters	Test Accuracy	Test Loss
Conv2D	82,644	0.3262	2.6576
SeparableConv2D	20,290	0.2207	3.2108
LightLayers ($K = 1$)	6,747	0.0275	4.2391
LightLayers ($K = 2$)	10,402	0.0398	4.1836
LightLayers ($K = 3$)	14,057	0.0559	4.0304
LightLayers ($K = 4$)	17,712	0.0551	3.9978
LightLayers ($K = 5$)	**21,367**	**0.0589**	**4.0009**

Contexte:

LOW-RANK MATRIX FACTORIZATION FOR DEEP NEURAL NETWORK TRAINING WITH HIGH-DIMENSIONAL OUTPUT TARGETS

Tara N. Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, Bhuvana Ramabhadran

IBM T. J. Watson Research Center, Yorktown Heights, NY 10598
{tsainath, bedk, vsindhw, earisoy, bhuvana}@us.ibm.com



Projet :

BDD images → Classification

1. Transfert learning (Inception V3) (Kermany, 2018)
2. Conv2D
3. LightLayer (LightConv2D) avec k = 1-5 (Jha, 2021 - submitted)

Python / Notebook Kaggle / CloudReady

DONNÉES

DONNÉES

https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia

Entrainement (5216 images), Validation (624) et test (16).

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


Resource

Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning

Daniel S. Kermany [1, 2, 14], Michael Goldbaum [2, 14], Wenjia Cai [2, 14], Carolina C.S. Valentim [2, 14], Huiying Liang [1, 14], Sally L. Baxter [2, 14], Alex McKeown [3], Ge Yang [2], Xiaokang Wu [4], Fangbing Yan [4], Justin Dong [1], Made K. Prasadha [2], Jacqueline Pei [1, 2], Magdalene Y.L. Ting [2], Jie Zhu [1, 5], Christina Li [2], Sierra Hewett [1, 2], Jason Dong [1] ... Kang Zhang [1, 2, 4, 12, 13, 15]

PNEUMONIA

PNEUMONIA

NORMAL

PNEUMONIA

PNEUMONIA

PROTOCOLE

Protocole

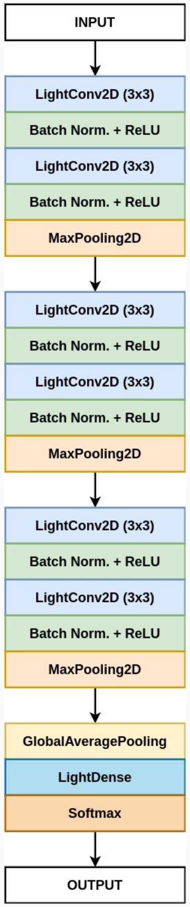
1. **Transfert learning (Inception V3) (Kermany, 2018)**

2. **Conv2D**

3. **LightLayer (LightConv2D) avec k = 1-5 (Jha, 2021 - submitted)**

Paramètres:

- loss = Categorical_Crossentropy
- 20 epochs
- Batch size 64
- lr = 0.001
- Image size = (299,299)
- GPU



Protocole

Nb paramètres

Nb paramètres entrainables

Taille modèle

Accuracy

Temps total d'entrainement

Temps d'entrainement par époque

RÉSULTATS

Protocole

Nb paramètres

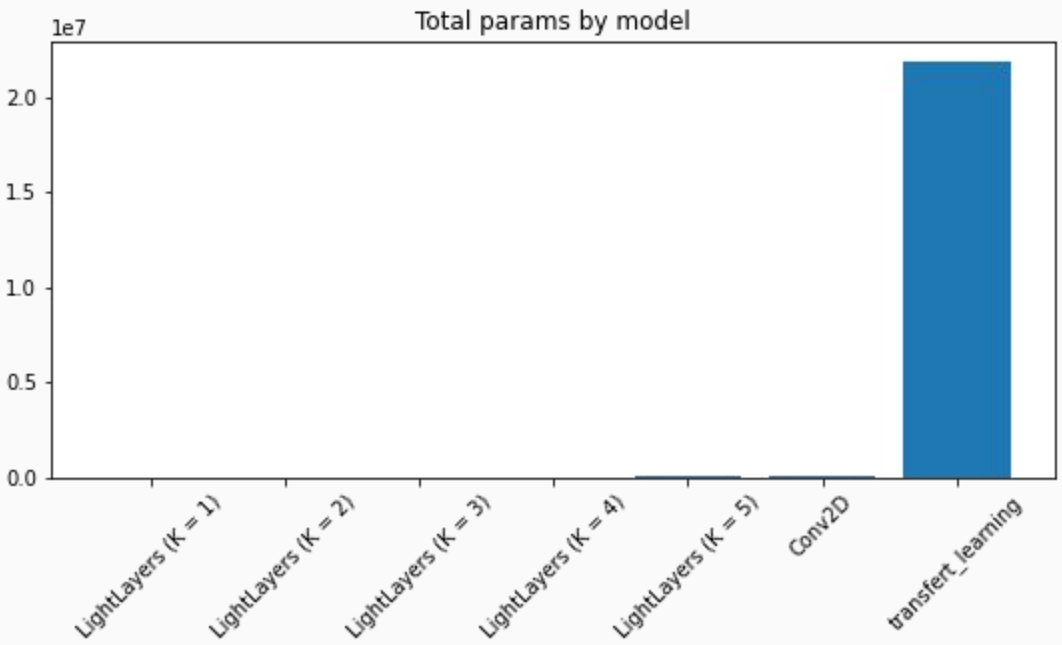
Nb paramètres entrainables

Taille

Accuracy

Temps total d'entraînement

Temps d'entraînement par époque



Protocole

Nb paramètres

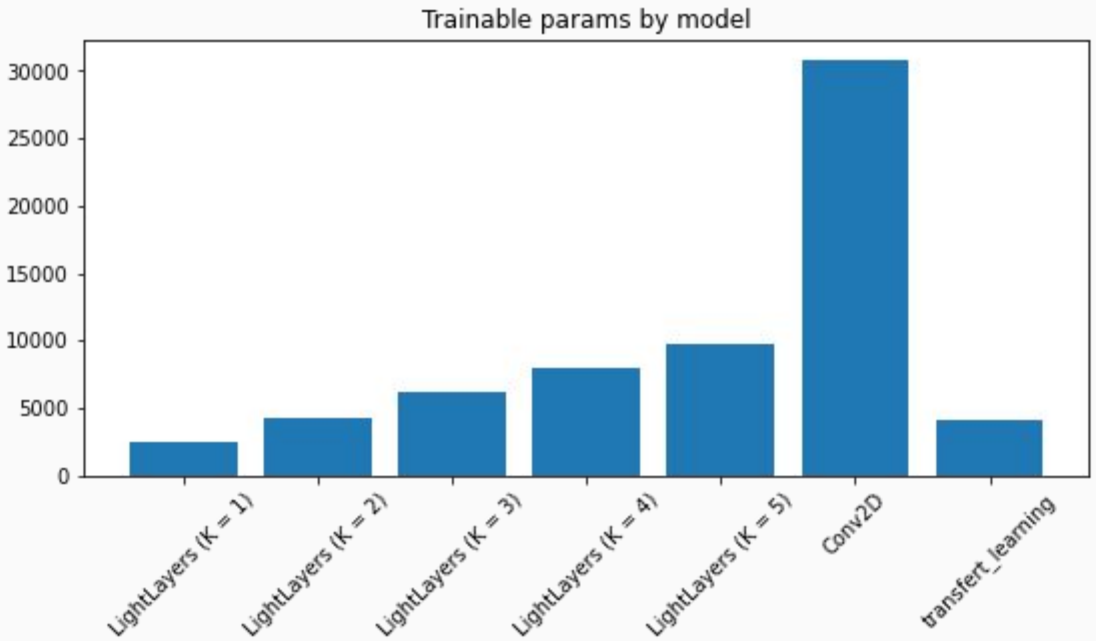
Nb paramètres entrainables

Taille

Accuracy

Temps total d'entraînement

Temps d'entraînement par époque



Protocole

Nb paramètres

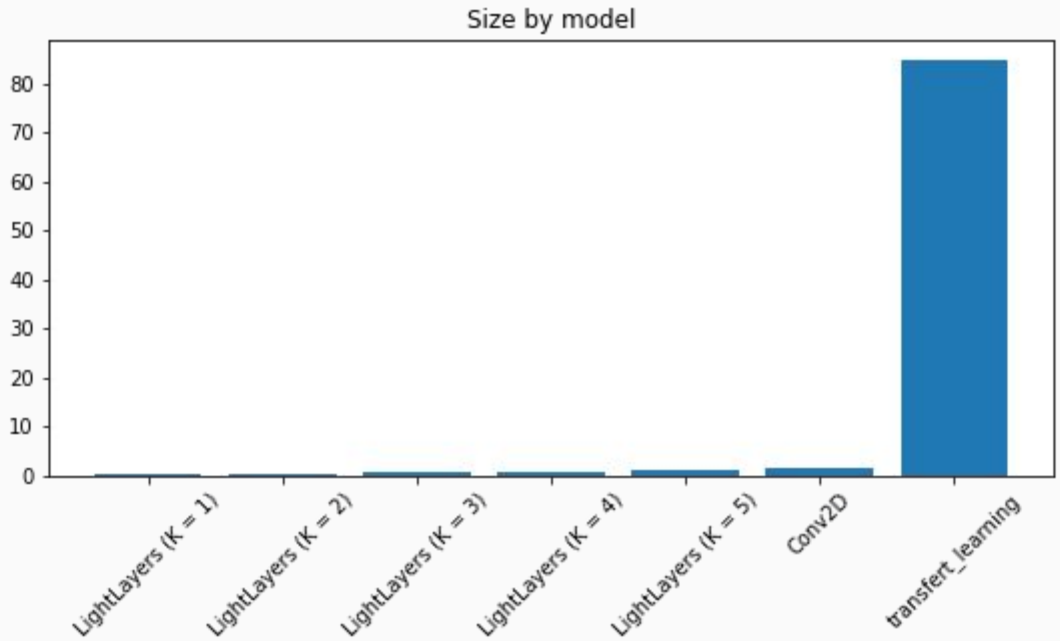
Nb paramètres entrainables

Taille

Accuracy

Temps total d'entraînement

Temps d'entraînement par époque



Protocole

Nb paramètres

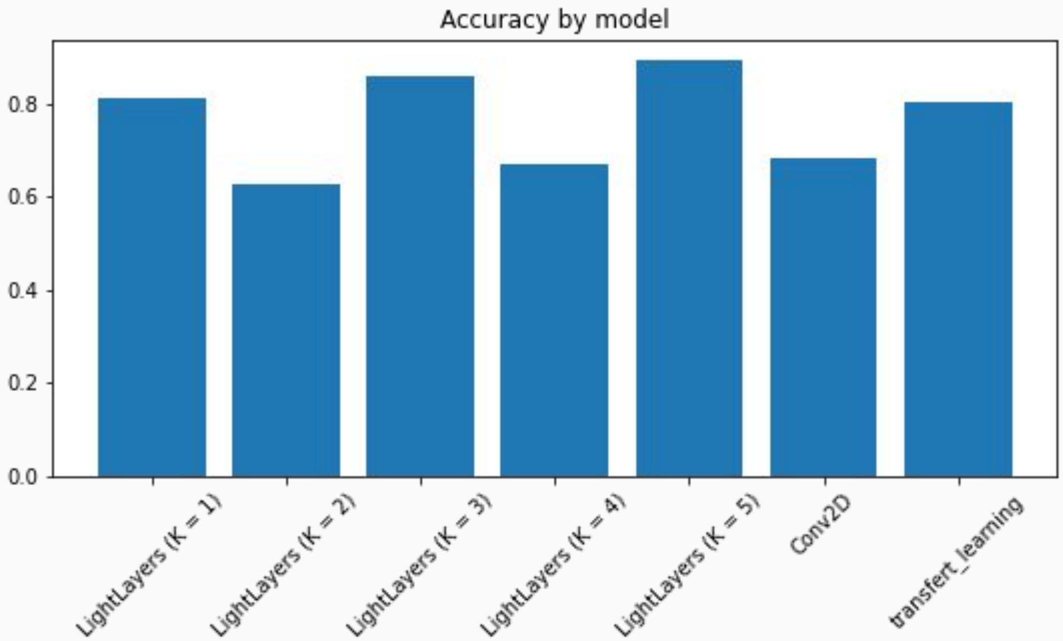
Nb paramètres entrainables

Taille

Accuracy

Temps total d'entraînement

Temps d'entraînement par époque



Protocole

Nb paramètres

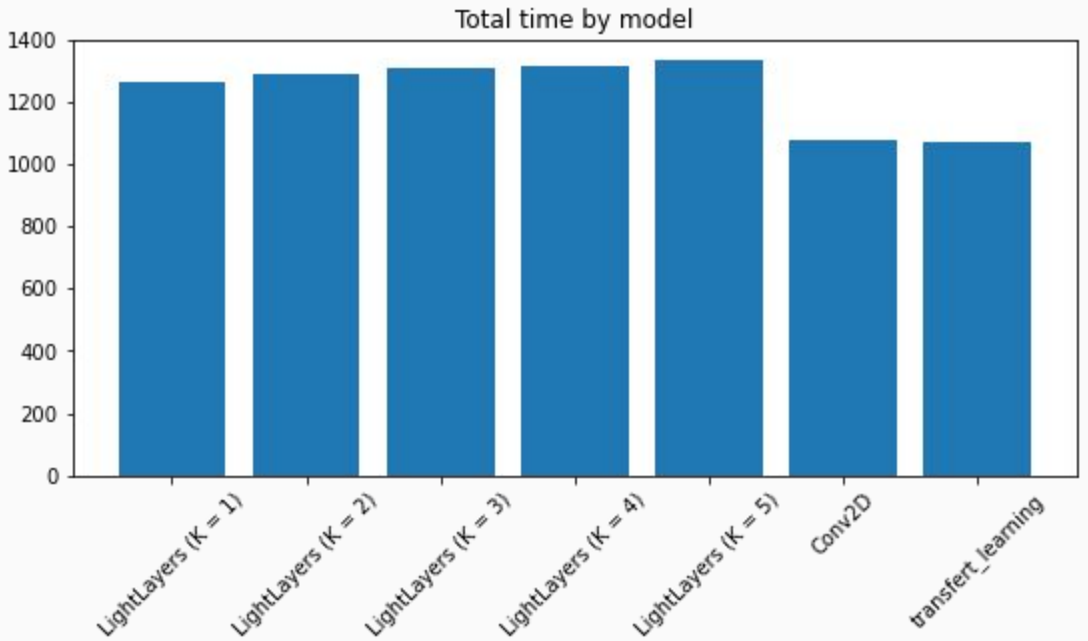
Nb paramètres entrainables

Taille

Accuracy

Temps total d'entraînement

Temps d'entraînement par époque



Protocole

Nb paramètres

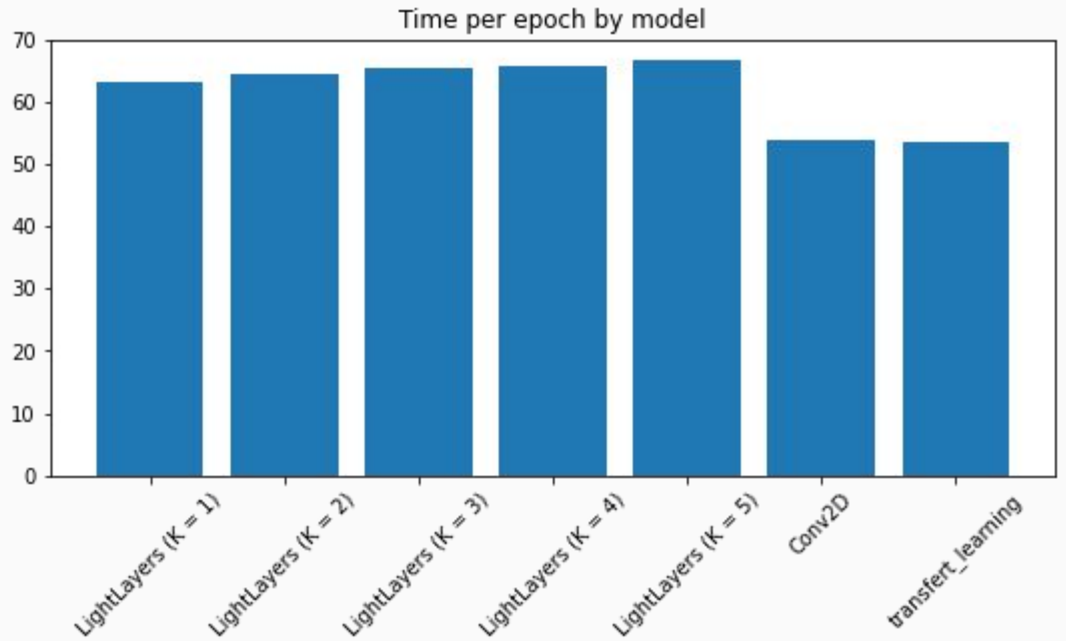
Nb paramètres entrainables

Taille

Accuracy

Temps total d'entraînement

Temps d'entraînement par époque



Conclusion

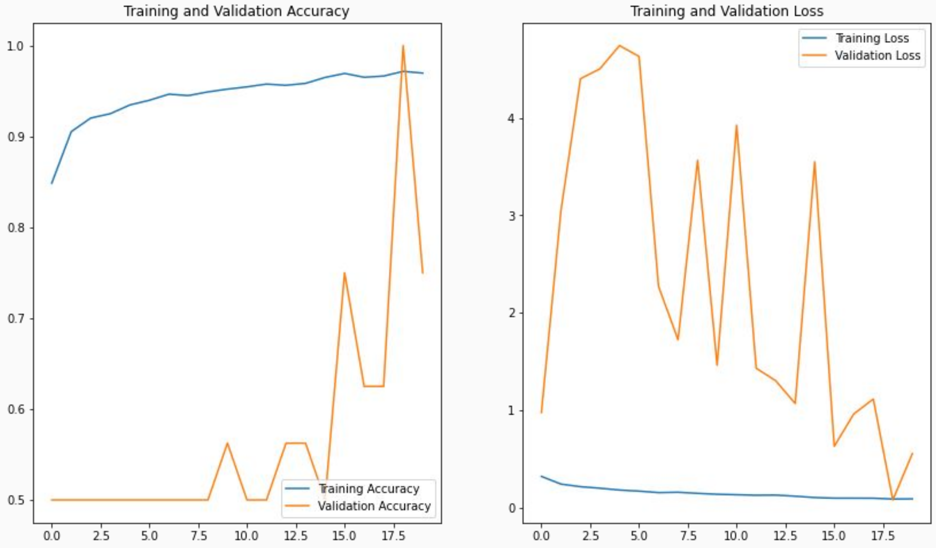
Model	Total params	Trainable params	Size	Accuracy	Total time	Time per epoch
LightLayers (K = 1)	2753	2417	0.251328	0.812500	1395.305205	69.765260
LightLayers (K = 2)	4600	4264	0.401146	0.625000	1425.277316	71.263866
LightLayers (K = 3)	6447	6111	0.562813	0.860577	1408.519347	70.425967
LightLayers (K = 4)	8294	7958	0.739403	0.669872	1450.185916	72.509296
LightLayers (K = 5)	10141	9805	0.930847	0.891026	1466.354471	73.317724
Conv2D	31074	30738	1.300407	0.684295	1157.990143	57.899507
transfert_learning	21806882	4098	84.690468	0.802885	1172.969305	58.648465

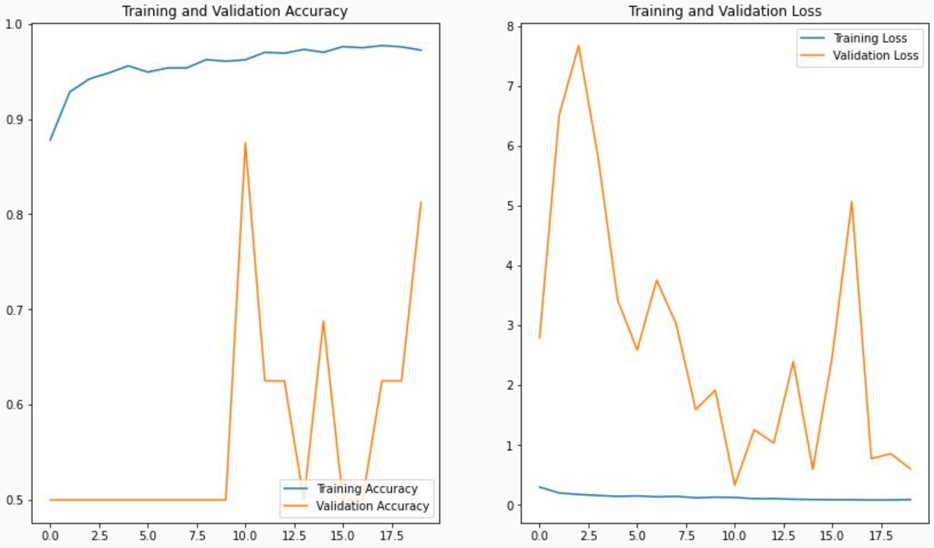
DISCUSSION

Discussion

- En raison de moins de paramètres, l'espace occupé par le fichier du modèle est plus petit, ce qui le rend plus adapté aux appareils où l'espace de stockage est limité.
- Précision sur la décomposition matricielle utilisée.
- Autres jeux de données, avec +2 classes notamment.
- Split train, val, test (Entraînement (5216 images), Validation (624) et test (16))
- Différence performance val et test sets

Discussion





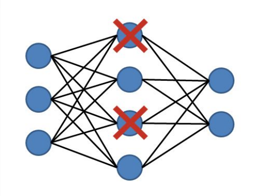
k = 3, accuracy = 0.86

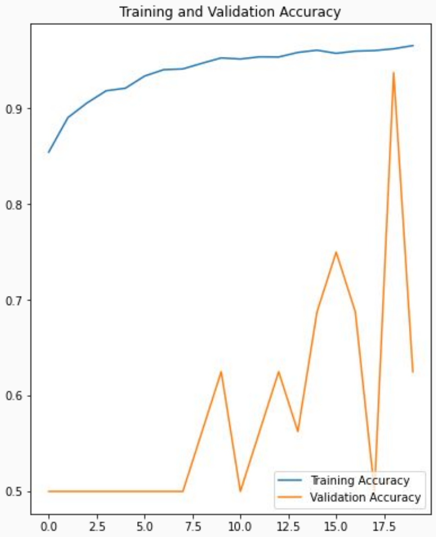
k = 5, accuracy = 0.89

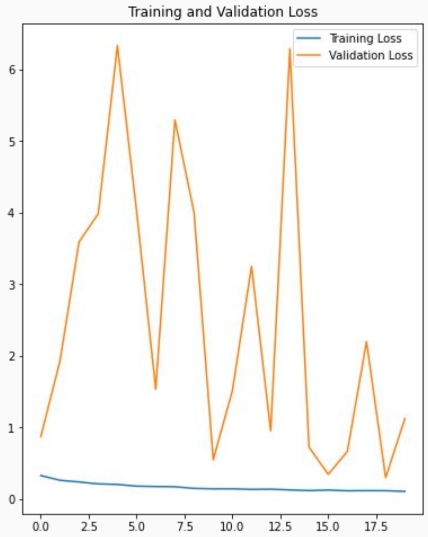
Discussion

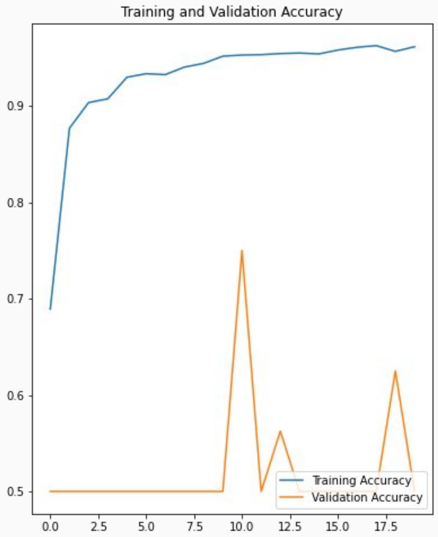
- Mise en place stratégie de lutte contre le sur-apprentissage (Augmentation, drop out)
- Learning rate

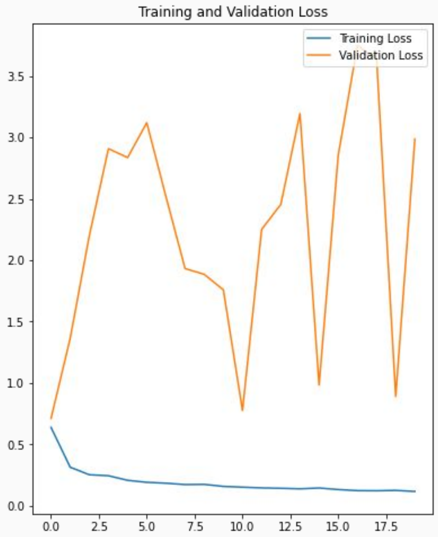
Discussion







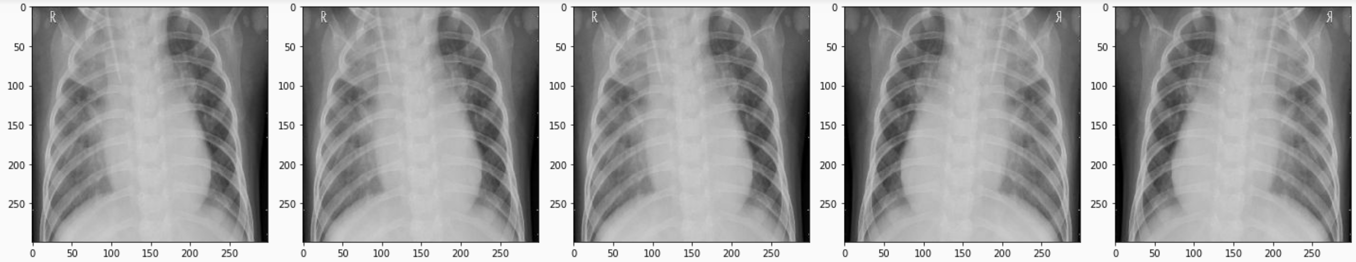


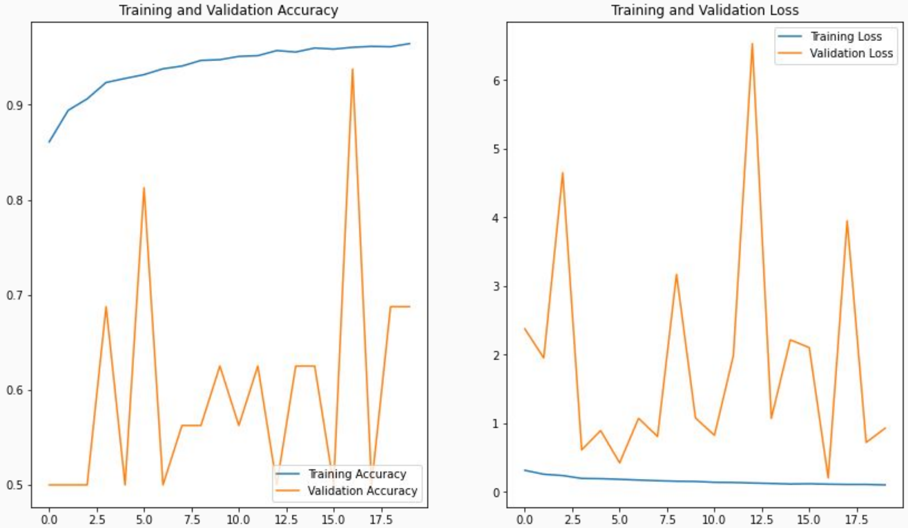


k = 3, accuracy = 0.81

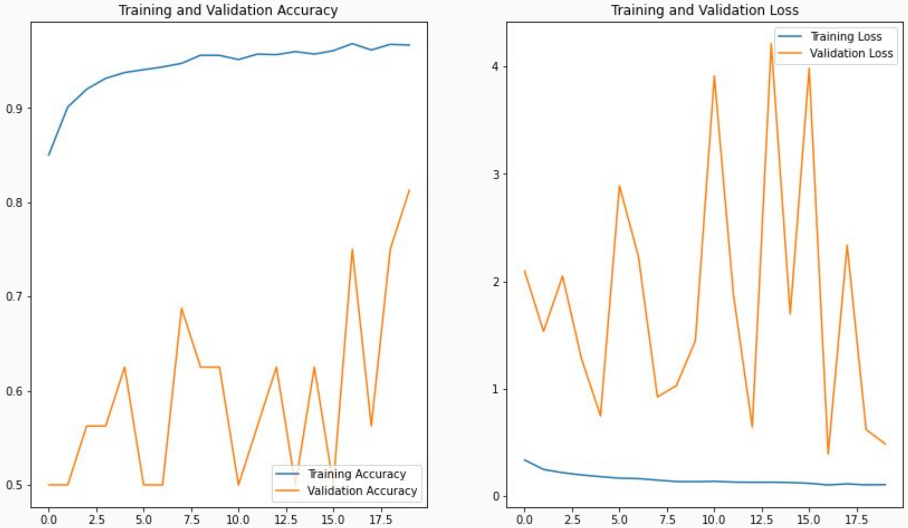
k = 5, accuracy = 0.62

Discussion





k = 3, accuracy = 0.84



k = 5, accuracy = 0.67

MERCI !

@xavbarbier

https://www.linkedin.com/in/barbierxavier/

https://github.com/xavierbarbier/

contact@xavierbarbier.com