Rapport Intelligence Artificielle

VO Frédéric

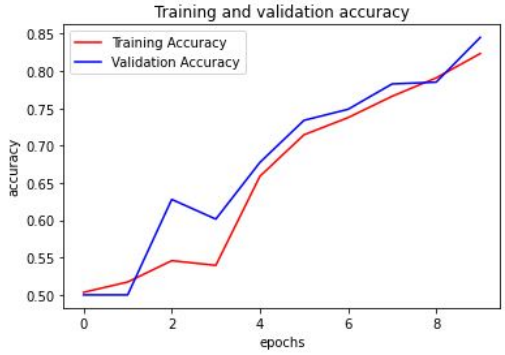
22 octobre 2022

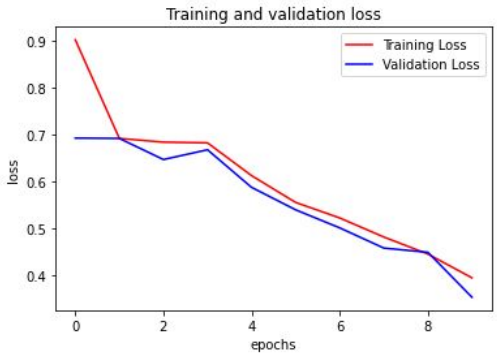
1 Modification des paramètres

Dans ce tp, nous allons prendre un code sur https://www.kaggle.com/ permettant de détecter les images de chiens et chats afin d'y changer les paramètres. Le code que j'ai choisi vient de https://www.kaggle.com/code/benyaminghahremani/dog-vs-cat-classification-cnn. Le but est de trouver les meilleurs paramètres afin d'avoir le meilleur pourcentage de validation.

1.1 Optimizer

Tout d'abord, j'ai décidé de changer le paramètre **optimizer**. L'optimizer est un algorithme qui va permettre de diminuer le loss et donc, d'avoir de meilleurs prédictions. Nous allons tester avec **Adam**, un optimizer très souvent utilisé.





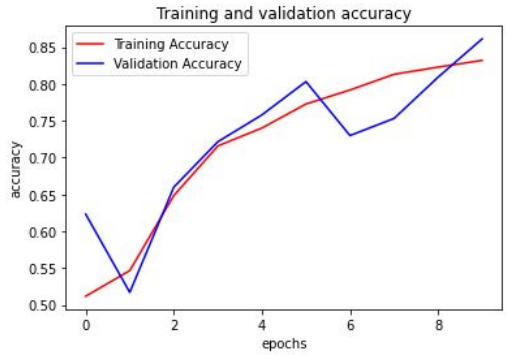
Les différents paramètres sont :

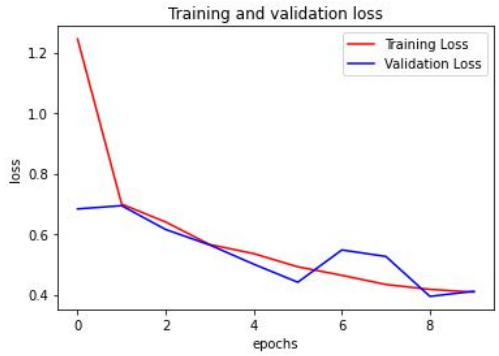
1. Epoch(s) : **10** ;
2. Optimizer : **Adam**
3. Learning rate : **0.0005** et decay rate : **1e-5**
4. Nombre de layers : **31** dont 11 Conv2D (**64(x2)/128(x3)/256(x3)/512(x3)**) filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

La validation et le training accuracy commence tous deux à **0.5**. Ensuite, nous pouvons voir une évolution allant jusqu'à **0.8183** pour le training et **0.8452** pour la validation.

Concernant le **loss**, on peut voir qu'il diminue aussi au fil des epochs. Comparé à l'accuracy, le loss n'est pas représenté par un pourcentage.

Nous allons comparer ces graphes avec d'autres qui ont utilisé un différent optimizer, **RMSprop**.





Les différents paramètres sont :

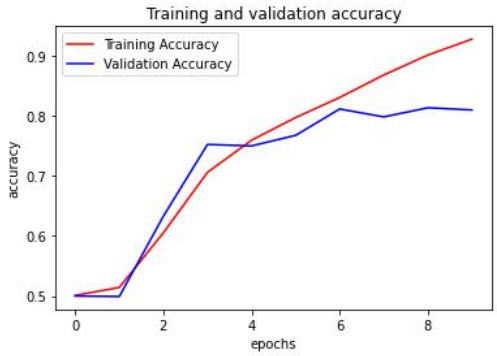
1. Epoch(s) : **10**
2. Optimizer : **RMSprop**
3. Learning rate : **0.0005**
4. Nombre de layers : **31** dont 11 Conv2D (**64(x2)/128(x3)/256(x3)/512(x3)**) filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

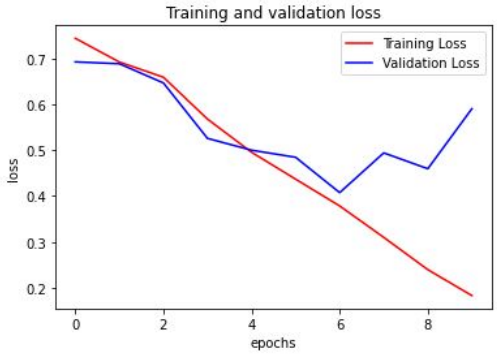
La **validation accuracy** au bout de **10** epochs est supérieure à celle avec **Adam**. De plus, nous pouvons remarquer qu'au bout de **5 epochs**, le graphe avec RMSprop atteint **0.8032** contre **0.7340** pour Adam.

On constate donc que RMSProp obtient de meilleurs résultats que Adam sur le court terme.

1.2 Layers et Optimizers

Par rapport au code de base, j'ai décidé de réduire le nombre de layers pour voir la différence avec le nombre de base :



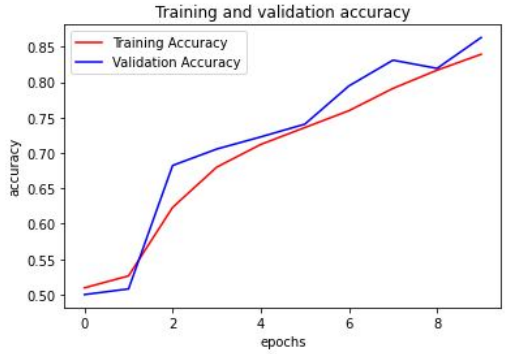


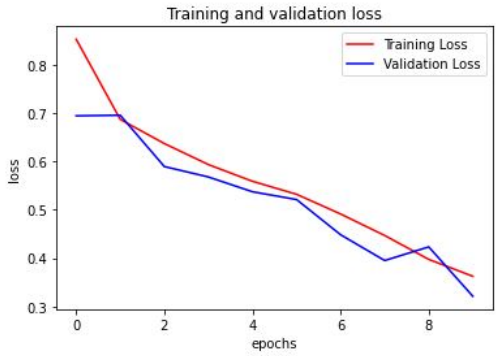
Les différents paramètres sont :

1. Epoch(s) : **10**
2. Optimizer : **Adam**
3. Learning rate : **0.0005**
4. Nombre de layers : **17** dont 5 Conv2D (**64(x2)/128(x3)** filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

Nous remarquons ici que la training accuracy augmente au long des epochs, alors que la validation accuracy augmente rapidement à **0.80** avant de stagner.

Pareillement avec le graphe de loss, le training loss baisse tandis que la validation loss stagne et augmente vers la fin. Réduire considérablement le nombre de layers rend les résultats moins précis. Essayons avec plus de layers.



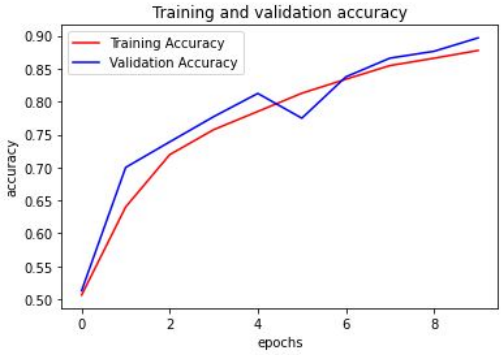


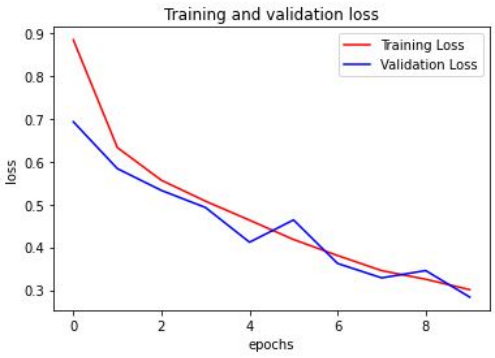
Les différents paramètres sont :

1. Epoch(s) : **10**
2. Optimizer : **Adam**
3. Learning rate : **0.0005**
4. Nombre de layers : **24** dont 8 Conv2D (**64(x2)/128(x3)/256(x3)** filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

La courbe de validation augmente de manière significative et atteint **0.8632**, soit plus que lors du premier test avec **31 couches** et avec **Adam**

Pour le moment, on constate que un grand nombre de layers n'a pas forcément une influence sur l'accuracy.





Les différents paramètres sont :

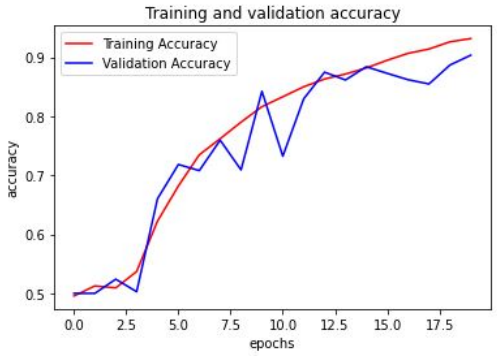
1. Epoch(s) : **10**
2. Optimizer : **RMSprop**
3. Learning rate : **0.0005**
4. Nombre de layers : **24** dont 8 Conv2D (**64(x2)/128(x3)/256(x3)** filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

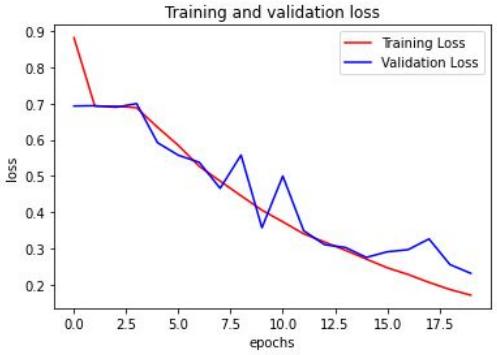
On obtient ici notre meilleur résultat en validation accuracy. En effet, on obtient **0.8968** avec RMSprop, donc plus qu'avec Adam.

Essayons désormais d'augmenter le nombre d'epochs pour voir les différentes évolutions.

1.3 Epochs

Enfin, j'ai repris les derniers models et les ai testé avec un nombre d'epochs à **20**.

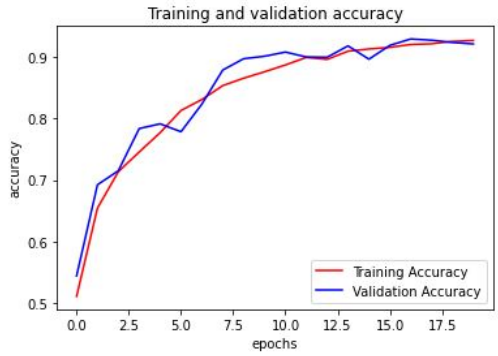


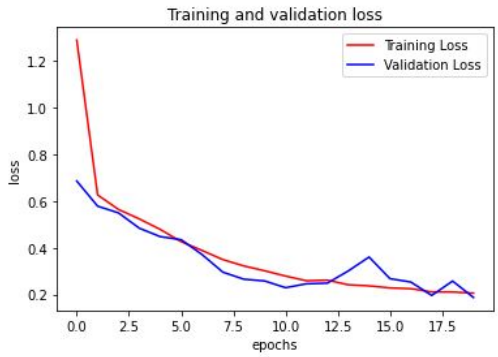


Les différents paramètres sont :

1. Epoch(s) : **20**
2. Optimizer : **Adam**
3. Learning rate : **0.0005**
4. Nombre de layers : **24** dont 8 Conv2D (**64(x2)/128(x3)/256(x3)** filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

On voit très bien que les courbes de training et validation accuracy fluctuent énormément. Cependant, on atteint pour la première la barre des 0.90 avec une validation accuracy de **0.9036**.

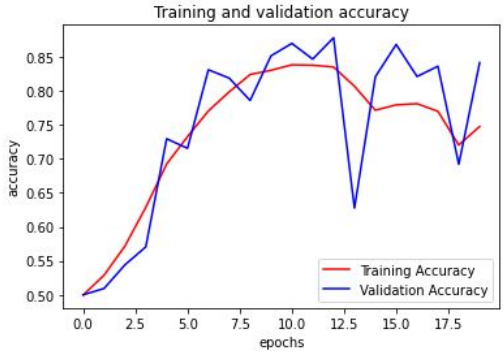


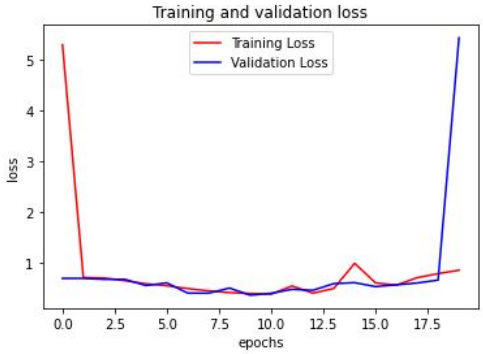


Les différents paramètres sont :

1. Epoch(s) : **20**
2. Optimizer : **RMSprop**
3. Learning rate : **0.0005**
4. Nombre de layers : **24** dont 8 Conv2D (**64(x2)/128(x3)/256(x3)** filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

Cette fois-çi, la validation accuracy atteint rapidement la barre des **0.90** au bout de **10 epochs**. A la fin, on obtient une accuracy de **0.9208**, alors que la courbe a une tendance stagnante après 10 epochs. Le pourcentage de validation accuracy semble satisfaisant.



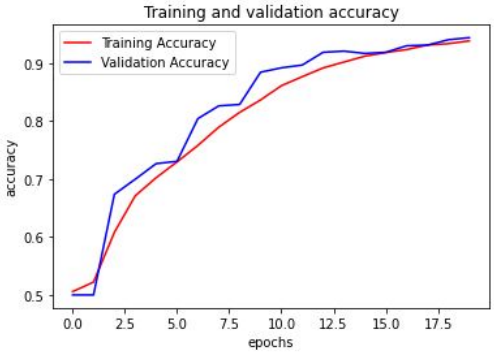


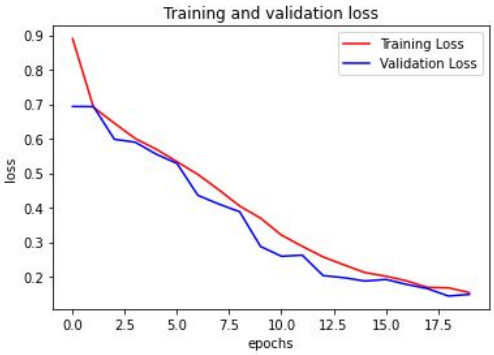
Les différents paramètres sont :

1. Epoch(s) : **20**
2. Optimizer : **RMSprop**
3. Learning rate : **0.0005**
4. Nombre de layers : **31** dont 11 Conv2D (**64(x2)/128(x3)/256(x3)/512(x3)**) filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

La courbe de validation varient beaucoup et les résultats ne semblent pas exploitables. On remarque même que le training accuracy commence à baisser au bout de 12 epochs.

La validation et training loss semblent stagner jusqu'à la fin où la validation loss augmentent énormément.





Les différents paramètres sont :

1. Epoch(s) : **20**
2. Optimizer : **Adam**
3. Learning rate : **0.0005**
4. Nombre de layers : **31** dont 11 Conv2D (**64(x2)/128(x3)/256(x3)/512(x3)**) filters, activation relu), des MaxPool2D (2,2), Dropout à 0.4, Flatten, et dense (1024 units).

Cette fois-ci, nous remarquons que la courbe de validation accuracy augmente jusqu'à **0.9436** et stagne. Ce model est sans doute le meilleur qu'on puisse avoir.

On a vu avec les différents tests que les paramètres ont un impact important sur les résultats. Par exemple, on a vu que les optimizers se comportaient différemment selon le nombre de couches et d'epochs. Les différents jeux de paramètres nous ont donc permis d'obtenir le meilleur résultat possible. Pour conclure ce TP, on voit donc que l'optimizers **Adam**, ainsi qu'un nombre d'epochs d'environ **20**, permettent d'obtenir un accuracy dans les alentours de **94%** et des loss inférieurs à **0.1**.