RAPPORT IA : Dog vs Cat Classification CNN

I- Mes premiers paramètres choisis pour la classification Dog and Cat

a. Les paramètres choisis pour commencer

Pour commencer , voici les différents paramètres du premier modèle de classification dog and cat cnn :

- Le nombre d'epochs : 10
- Le nombre de couche (en comptant celle de l'input et de l'output) : 17
    - Parmi ces 17 couches on a notamment :
        - Conv2D avec méthode d'activation relu et kernel_size (3,3) et 64 neurones et aussi avec 128 neurones
        - Maxpool2D
        - Dropout a 0.4
        - Flatten
        - Dense avec méthode d'activation relu (256 neurones) et méthode d'activation sigmoid pour l'output layer
        - Un learning rate qui est égale à 0.0005 et un decay_rate : 1e-5
        - Optimisateur = adam

Pour le nombre d'images d'entrainements , de validation et de teste ,on a environ une répartition des donnés qui correspond à :

- Training (60%) (15000)
- Validation(20%) (5000)
- Test (20%)(5000)

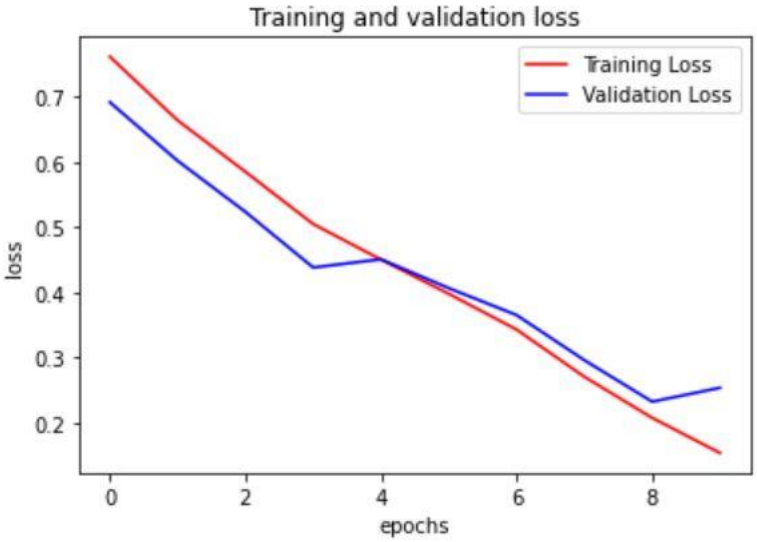
b. Les résultats obtenus

A la fin de la 10 ème epoch , on obtient comme résultats :

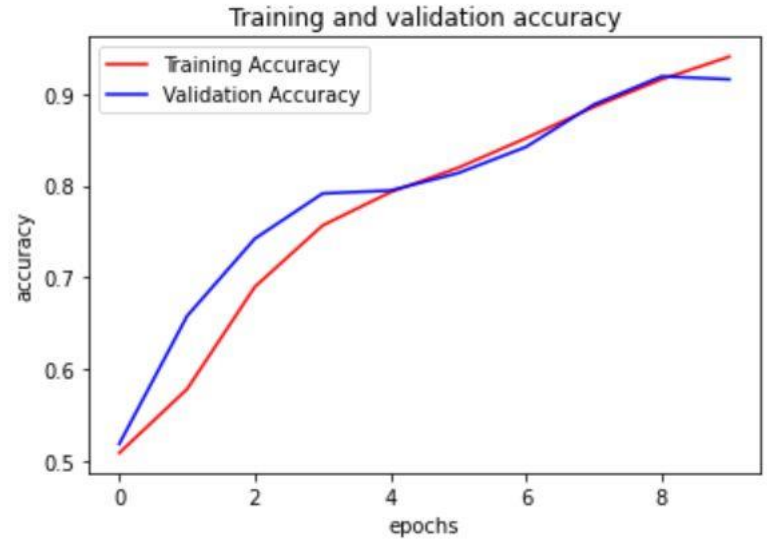
- Training_loss: 0.1447
- Training_acc: 0.9441
- Validation_loss: 0.2534
- Validation_acc: 0.9160

On peut observer qu'on a obtenu de très bon résultats avec ces premiers paramètres

i. Le graphe de coût



ii. Le graphe de précision



c. Quels paramètres modifier ?

Je pense que les paramètres qu'il pourrait avoir un impact important sur l'augmentation de l'efficacité du modèle de classification dog and cat sont notamment :

- Le nombre d'epoch
  - Car logiquement plus , le modèle est entrainer plus il est performant (en théorie)
- Le nombre de couches (Layers)
  - En effet , il est possible d'affecter des neurones à ces différents couches et notamment aussi des méthodes d'activations qui je pense peuvent avoir un impact important
- Learning rate
  - Ce paramètre me semble important au vu de sa traduction directe : « vitesse d'apprentissage »
- Régularisation
  - Il existe dans les différents couches , des couches telles que Dropout qui permettrait de contrer l'overfitting et optimiser le modèle d'apprentissage , je pense donc qu'il aura son impact.
- Optimisateur
  - Je ne sais pas à quoi sert ce paramètre mais au vu de son nom , il aura sûrement son importance.

II- Modification de différents paramètres et observation du comportement

a. Nombre d'epochs

10 epochs :

- Training_loss: 0.1447
- Training_acc: 0.9441
- Validation_loss: 0.2534
- Validation_acc: 0.9160

15 epochs :

- Training_loss: 0.0622
- Training_acc: 0.9771
- Validation_loss: 0.2328
- Validation_acc: 0.9255

L'augmentation du nombre d'epochs permet d'obtenir un training_loss faible et une valeur pour le training_accuracy et le validation_accuracy plus importante.

b. Vitesse d'apprentissage (learning rate et decay_rate) , optimisateur

Learning rate :0.0005 , decay_rate : 1e-5 , optimisateur adam :

- Training_loss: 0.1447
- Training_acc: 0.9441
- Validation_loss: 0.2534
- Validation_acc: 0.9160

Learning rate :0.001 , decay_rate : 1e-6 , optimisateur adam :

- Training_loss: 0.6924
- Training_acc: 0.5079
- Validation_loss: 0.6930
- Validation_acc: 0.5000

Un augmentation du learning rate ne semble permettre d'obtenir de meilleur résultats , le modèle ne semble réussir à bien apprendre.

Learning rate :0.0005 , decay_rate : 1e-5 , optimisateur RMSprop :

- Training_loss: 0.3043
- Training_acc: 0.8764
- Validation_loss: 0 .4714
- Validation_acc: 0.8445

Learning rate :0.0001 , decay_rate : 1e-6 , optimisateur RMSprop :

- Training_loss: 0. 3402
- Training_acc: 0.8570
- Validation_loss: 0.3643
- Validation_acc: 0.8597

Un augmentation du learning rate ne semble permettre d'obtenir de meilleur résultats mais le modèle semble mieux réussir son apprentissage que pour optimisateur « Adam».

L'optimisateur Adam , ainsi que un petit learning rate semble être les paramètres les plus intéressant à utiliser.

c. Nombre de layer , nombres de neurones et fonction d'activation

Dans cette partie , toutes les fonctions utilisées ne sont pas évoqués telles que : Dense , Flatten,Conv2D ou Maxpool2D. Seulement quelques unes seront évoquées.

17 couches , nombre de neurones totaux (64 x 2 , 128 x 3 ,256 x 3) , utilisation de la fonction d'activation relu et régularisation à l'aide de dropout(0.4) , Conv2D avec kernel_size (3,3) ,… :

- Training_loss: 0.1447
- Training_acc: 0.9441
- Validation_loss: 0.2534
- Validation_acc: 0.9160

17 couches , nombre de neurones totaux (64 x 2 , 128 x 3 ,256 x 3) , utilisation de la fonction d'activation relu et régularisation à l'aide de dropout(0.8) , Conv2D avec kernel_size (3,3) ,… :

- Training_loss: 0.6360
- Training_acc: 0.6486
- Validation_loss: 0.6335
- Validation_acc: 0.6252

17 couches , nombre de neurones totaux (64 x 2 , 128 x 3 ,256 x 3) , utilisation de la fonction d'activation relu et régularisation à l'aide de dropout(0.2) , Conv2D avec kernel_size (3,3) ,… :

- Training_loss: 0.0323
- Training_acc: 0.9892
- Validation_loss: 0.3222
- Validation_acc: 0.9242

L'Utilisation de la couche de régularisation dropout à 0.2 semble donner les meilleurs résultats , cela devrait donc dire qu'il n'y a pas besoin de mettre en silence beaucoup de neurones pour avoir notre modèle parfait .

12 couches , nombre de neurones totaux (64 x 2 , 128 x 3) , utilisation de la fonction d'activation relu et régularisation à l'aide de dropout(0.2) , Conv2D avec kernel_size (3,3) ,… :

- Training_loss: 0.0443
- Training_acc: 0.9856
- Validation_loss: 0.4297
- Validation_acc: 0.9065

31 couches , nombre de neurones totaux : (64 x 2, 128 x 3 , 256 x 3 , 512 x 3 , 1024 x 3) , utilisation de la fonction d'activation relu et régularisation à l'aide de dropout(0.2) , Conv2D avec kernel_size (2,2) :

- Training_loss: 0.2300
- Training_acc: 0.9022
- Validation_loss: 0.1894
- Validation_acc: 0.9208

L'augmentation du nombres de layers et du nombre de neurones ne semble pas rendre le modèle plus performant que ça.

17 couches , nombre de neurones totaux (64 x 2 , 128 x 3 ,256 x 3) , utilisation de la fonction d'activation elu et régularisation à l'aide de dropout(0.2) , Conv2D avec kernel_size (3,3) ,… :

- Training_loss: 0.7083
- Training_acc: 0.5021
- Validation_loss: 0.6940
- Validation_acc: 0.5000

La fonction d'activation élu ne permet pas d'obtenir un apprentissage intéressant pour notre modèle .

III- Choix du meilleur modèle : des meilleurs paramètres

a. Paramètres

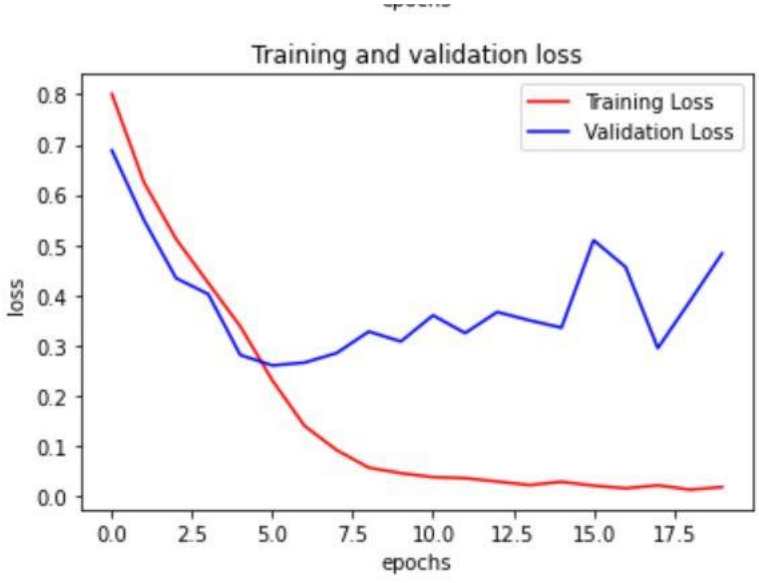
Pour finir , voici les différents paramètres de mon modèle de classification dog and cat cnn :

- Le nombre d'epochs : 20
- Le nombre de couche (en comptant celle de l'input et de l'output) : 17
    - Parmi ces 17 couches on a notamment :
        - Conv2D avec méthode d'activation relu et kernel_size(3,3), avec 64 neurones et aussi avec 128 neurones
        - Maxpool2D
        - Dropout(0.2)
        - Flatten
        - Dense avec méthode d'activation relu (256 neurones) et méthode d'activation sigmoid pour l'output layer
        - Un learning rate qui est égale à 0.0005 et decay_rate = 1e-5
        - Optimisateur = adam

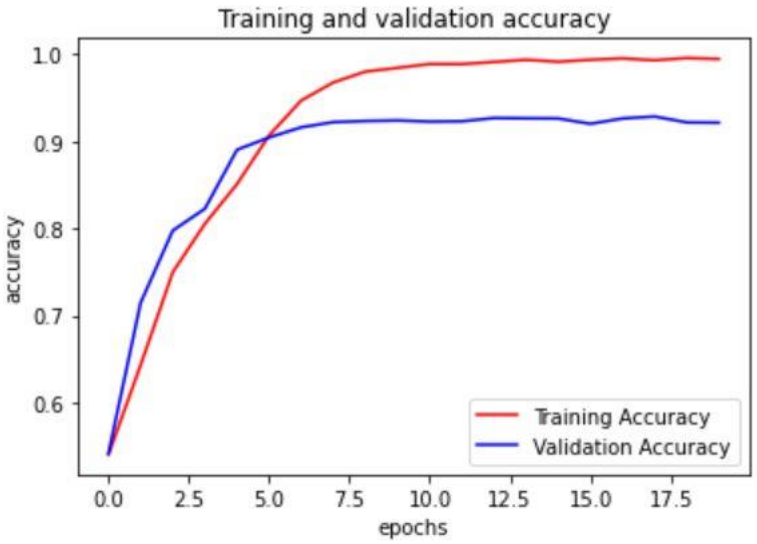
A la fin de la 20 ème epoch , on obtient comme résultats :

- Training_loss: 0.0146
- Training_acc: 0.9950
- Validation_loss: 0.4832
- Validation_acc: 0.9211

b. Graphe de coût



c. Graphe de précision



d. Matrice de confusion

```
[[2370  130]
 [  97 2403]]
```