Présentation Cifar10

Deep Learning

Bilal MAHJOUBI
Fabien BARRIOS

Lien du Drive :
https://drive.google.com/drive/folders/18mDDic1gVUpeB86Hk3BwYF-IACehgyfh?usp=sharing

Lien du colab :
https://colab.research.google.com/drive/17w_RXNvB_oqQkYScpdzRRdk1aIOMdlYO?usp=sharing

MLP

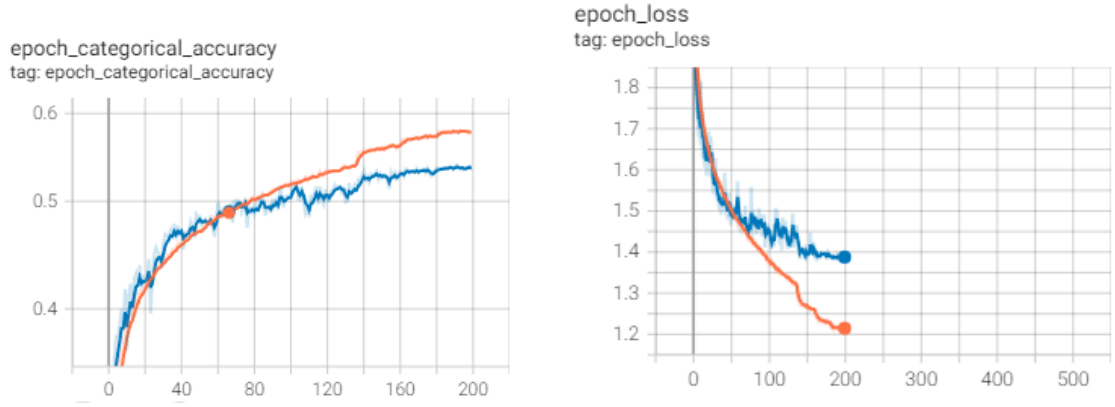
Nous avons décidé de commencer notre étude de cifar10 par le MLP. Voici les paramètres de départ :

Learning rate = 0.06

Optimiser = SGD

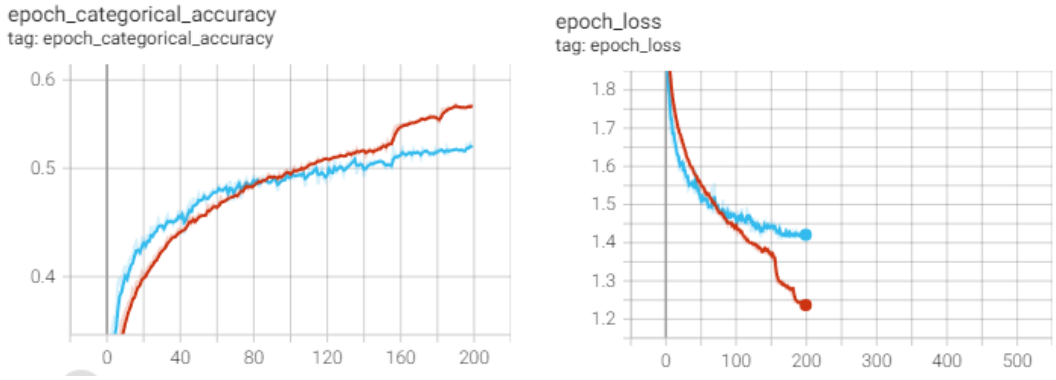
Seed = 42

Batch size = 256



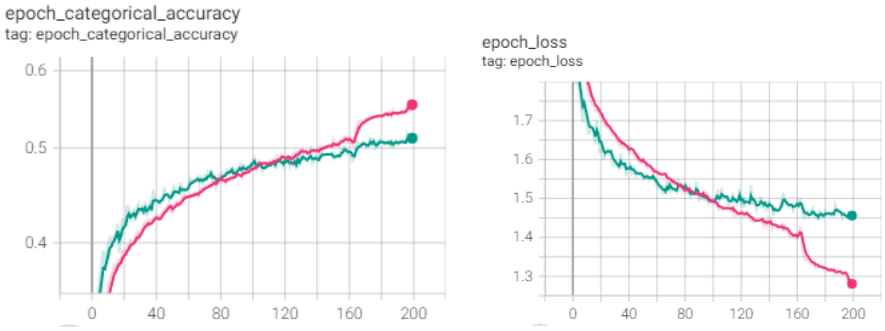
MLP_0.06_SGD_256_Seed_42_2022-01-03_09_07_02.555849

Nous pouvons voir que l'accuracy commence à stagner à 0.58. Sachant que l'exécution prend 20 minutes, nous décidons d'augmenter le batch size (512) et d'ajouter un momentum (0.2). Nous décidons d'ajouter un dropout de 0.2.



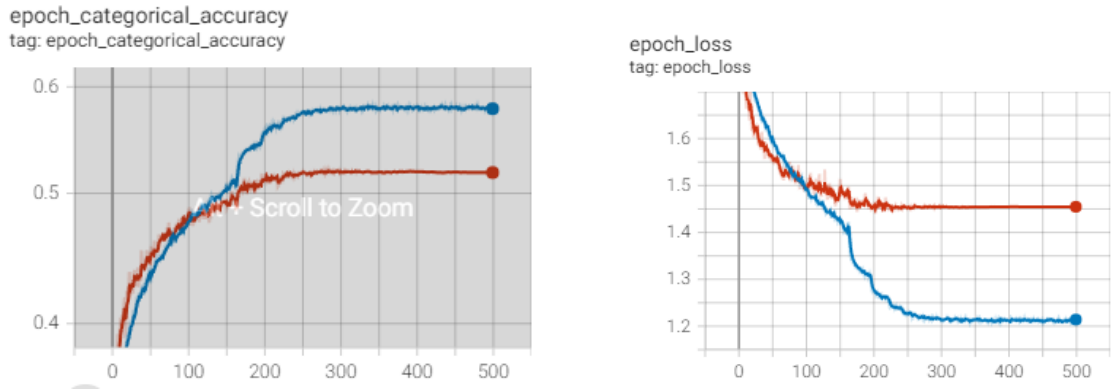
MLP_0.06_SGD_512_Seed_42_2022-01-03_09_29_20.012434

Le résultat est presque pareil mais le temps d'exécution a diminué. Nous avons décidé par la suite d'augmenter le learning rate à 0.08 pour améliorer accuracy.



MLP_0.08_SGD_512_Seed_42_2022-01-03_10_37_45.261591

L'accuracy n'augmente pas malgré l'augmentation du learning rate. On décide d'augmenter le nombre de couche(+5 couches) mais nous voyons sur la courbe que modèle overfit à cause de l'augmentation du nombre de paramètres.



Nous décidons de changer de modèle et de passer à un MLP avec des skips connection.

MLP WITH SKIP CONNECTION

Nous commençons avec les paramètres suivants :

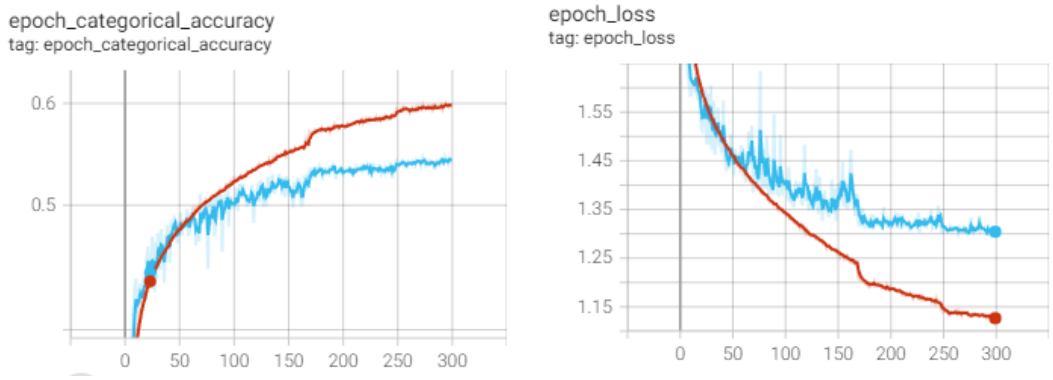
Learning rate = 0.01

Optimiser = SGD

Seed = 42

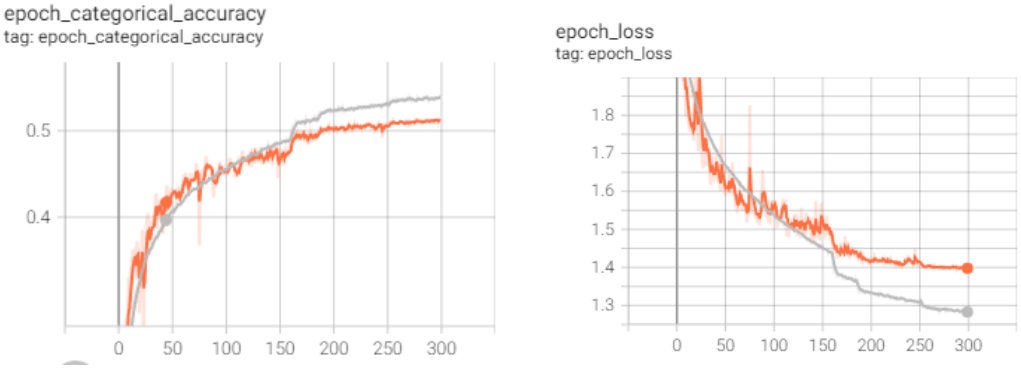
Couche = 10

Batch size = 64



MLPS_0.01_SGD_64_Seed_42_2022-01-03_12_09_39.222688

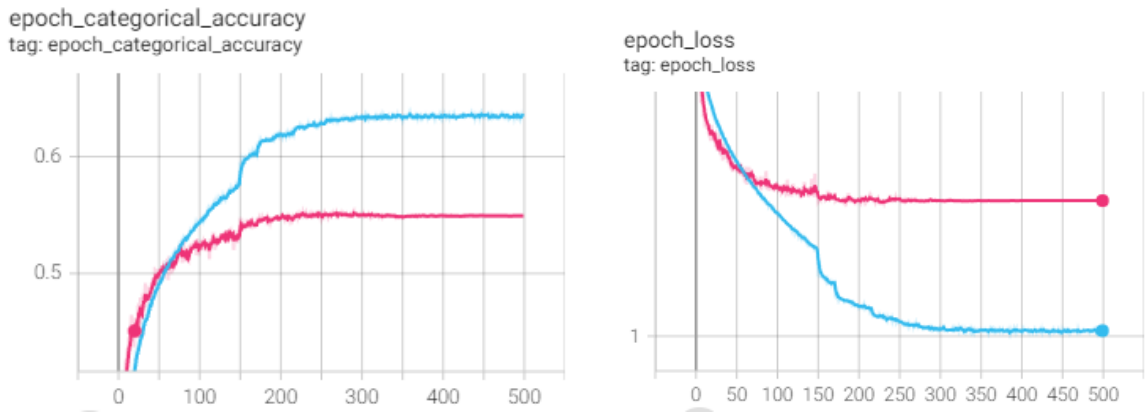
Premier constat, l'exécution prend beaucoup trop de temps (35 minutes) et l'accuracy atteint les 60% nous augmentons le learning rate (0.06) et le batch size (256) :



MLPS_0.06_SGD_256_Seed_42_2022-01-03_13_47_31.194929

Plus nous augmentons le lr et le batch size et plus le score d'accuracy diminue.

Nous diminuions le batch size (32) et le learning rate (0.01). Nous augmentons aussi le nombre d'époque.



MLPS_0.01_SGD_32_Seed_42_2022-01-03_15_21_11.797646

Nous pouvons voir que qu'au bout de la 200ème le modèle commence à overfit.

Le premier modèle de MLP avec des skips de connection reste pour l'instant le meilleur.

Nous décidons de passer au CNN.

CNN

Pour le CNN voici les paramètres que nous avons utilisé pour commencer :

Learning rate = 0.01

Couche = 3

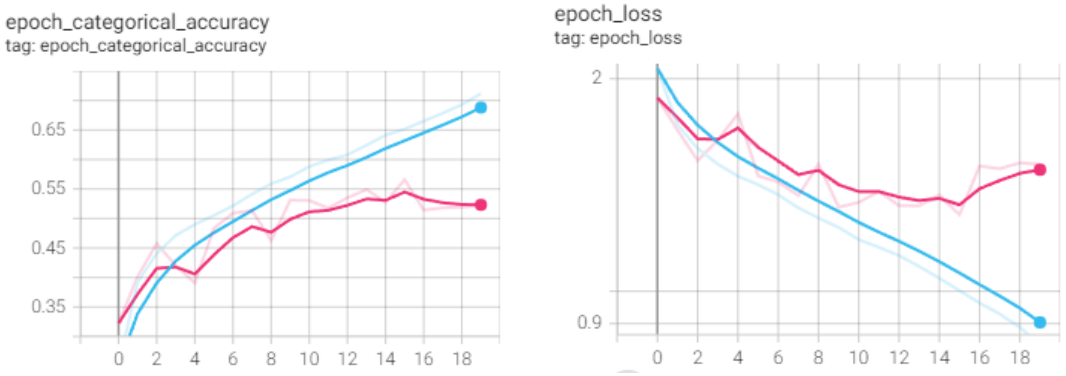
Optimiser = SGD

Seed = 42

Batch size = 128

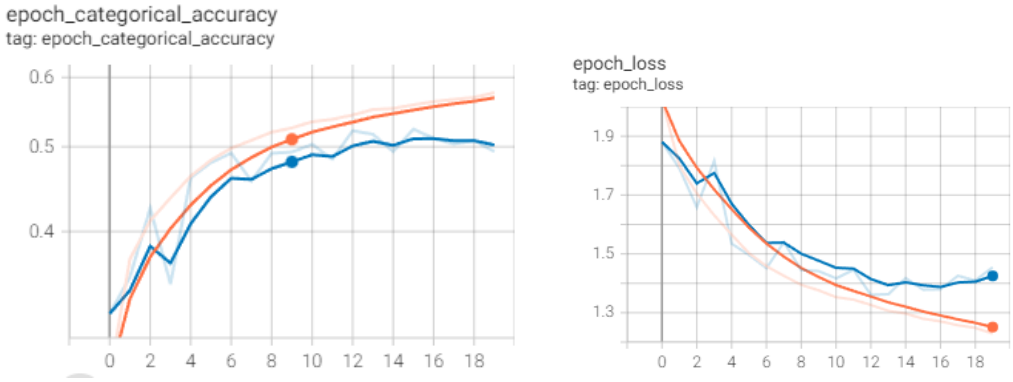
Pour ce modèle nous n'avons pas de graphique à proposer car l'exécution était beaucoup trop longue, un peu près 3 minutes par epoch. On augmente donc le learning rate (0.08) et le batch size (1024). Encore une fois, nous n'avons pas de graphique à proposer car ce modèle overfit avec ces paramètres.

Nous diminuions le nombre d'itération et nous enlevons une couche (2) pour éviter un overfitting éventuel.



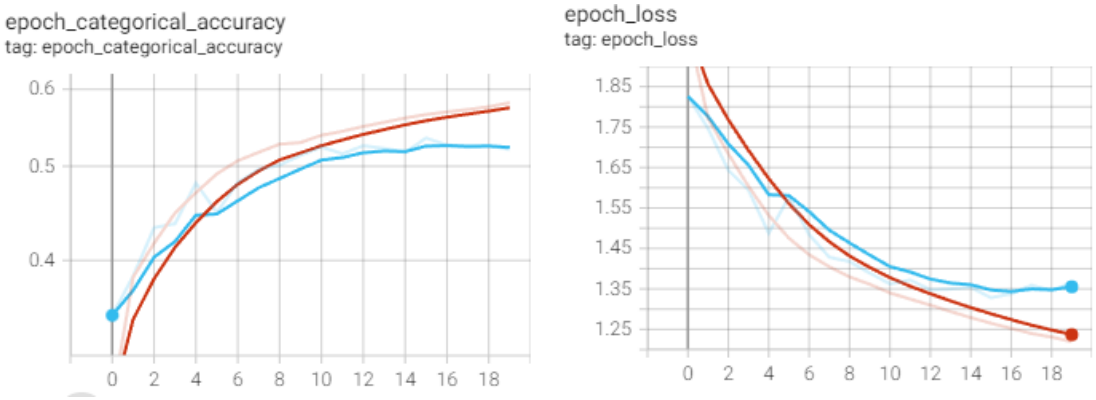
CNN_0.04_SGD_256_Seed_42_2022-01-05_17_46_17.150958

On remarque sur la fin un début d'overfitting. On baisse un petit peu le learning rate (0.02)



CNN_0.02_SGD_256_Seed_42_2022-01-05_18_24_13.747746

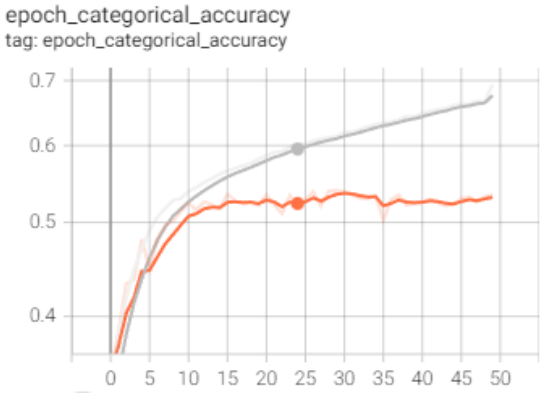
On remarque que l'overfitting est moins marqué. Nous passons sur le GPU pour essayer de réduire le temps d'exécution.

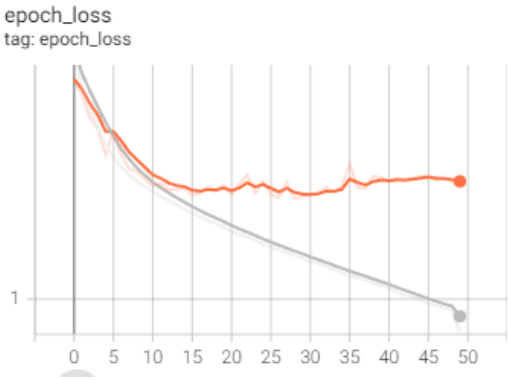


CNN_0.01_SGD_128_Seed_42_2022-01-05_20_04_18.696393

On peut apercevoir que nous overfittons plus et que le temps à été divisé par 10. Nous sommes passé d'un temps d'exécution de 30 minutes à 3 minutes.

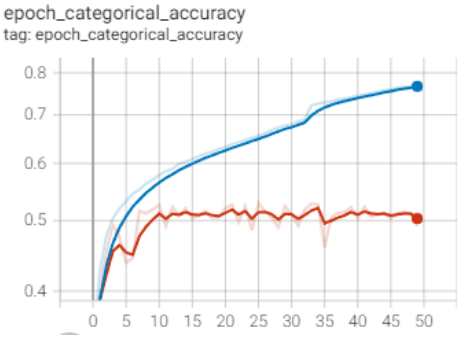
Nous rajoutons 3 couches et 30 itérations (5 couches au total) pour améliorer le score d'accuracy.

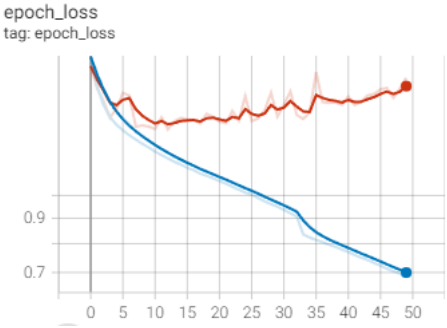




CNN_0.01_SGD_128_Seed_42_2022-01-05_20_23_06.903341

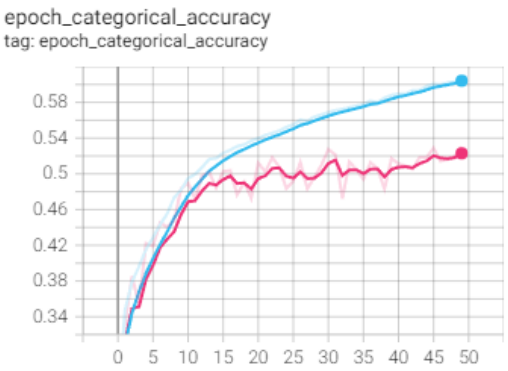
Nous constatons un phénomène d'overfitting au bout de la 25eme epoch. Nous diminuions le batch size.





Nous faisons la constatation suivante : plus nous diminuions le batch size plus nous arrivons à atteindre des scores d'accuracy élevés sur le dataset de train mais plus le phénomène d'overfitting s'accentue. Cela est peut-être du au fait que nous ne diminuions pas le learning rate avec le batch size.

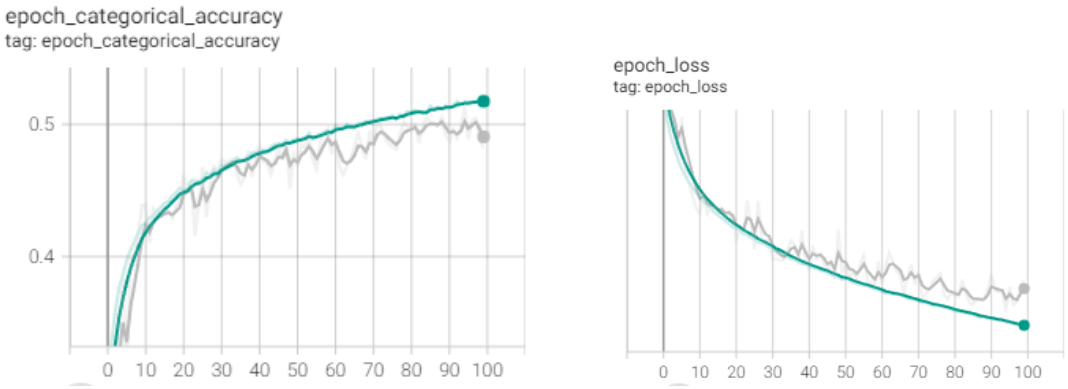
On décide pour finir d'augmenter le batch size et d'ajouter encore 5 couches (10).



L'overfitting est réduit et atteignons un score d'accuracy sur le dataset de train de 0.60.

Modèle linéaire

Curieux du résultat, nous avons testé avec un modèle linéaire avec une seule couche.



On remarque que le score d'accuracy sur 100 epochs sur le dataset de train et de validation dépasse les 50%. Le résultat est assez satisfaisant sachant que le modèle n'a qu'une couche de neurones et que le dataset possède 10 classes.

Conclusion

Pour Conclure, on peut dire que tous les modèle que nous avons essayé avec cifar10 c'est vraiment le CNN qui tire son épingle du jeu. Nous atteignons un très peu d'itérations les 60% d'accuracy sur 10 classes (20 epochs pour le CNN contre 200 epochs pour le MLP). Le loss lui est un peu élevé mais il diminue très rapidement.