haute école ingénierie
neuchâtel berne jura
www.he-arc.ch

Jonatan Baumgartner, Massimo De Santis, Florian Feuillade

1)
La descente de gradient, les paramètres : le batch size, le nombre d'epoch, la loss function, quel metrique utiliser.

Categorical cross entropy $$\text{Loss} = -\sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i$$

2)
MLP from raw data : Input -> Dense1 -> Dense2(output)
Size :

- Input = 784
- Dense1 = 300
- Dense2 = 10

Parameters :

- Input -> Dense1 = 235200 weight + 300 bias = 235500 parameters
- Dense1 -> Dense2 = 3000 weight + 10 bias = 3010 parameters
- Total = 238510 parameters

MLP from HOG : Input -> Dense1 -> Dense2(output)
Size :

- Input = 392
- Dense1 = 200
- Dense2 = 10

Parameters :

- Input -> Dense1 = 78400 weight + 200 bias = 78600 parameters
- Dense1 -> Dense2 = 2000 weight + 10 bias = 2010 parameters
- Total = 80610 parameters

CNN : Input -> Conv2D1 -> Maxpooling2D1 -> Conv2D2 -> Maxpooling2D2 -> Conv2D3 -> Maxpooling2D3 -> Flat -> Dense1 -> Dense2(output)
Size :

- Input = (28, 28, 1)
- Conv2D1 = (28, 28, 9), kernel = (5, 5) x 9 times
- Maxpooling2D1 = (14, 14, 9)
- Conv2D1 = (14, 14, 9), kernel = (5, 5) x 9 times
- Maxpooling2D1 = (7, 7, 9)
- Conv2D1 = (7, 7, 16), kernel = (3, 3) x 16 times
- Maxpooling2D1 = (3, 3, 16)
- Flat = 144
- Dense1 = 25
- Dense2 = 10

Parameters :

- Input -> Conv2D1 = 225 weight + 9 bias = 234 parameters
- Conv2D1 -> Maxpooling2D1 = 0 parameters
- Maxpooling2D1 -> Conv2D2 = 2025 weight + 9 bias = 2035 parameters
- Conv2D2 -> Maxpooling2D2 = 0 parameters
- Maxpooling2D2 -> Conv2D3 = 1296 weight + 16 bias = 1312 parameters

Jonatan Baumgartner, Massimo De Santis,
Florian Feuillade

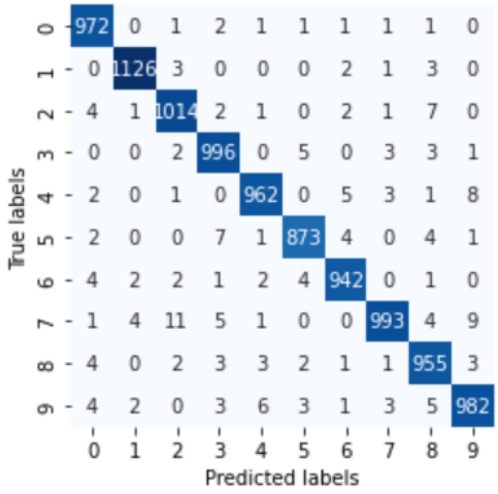
- Conv2D3 -> Maxpooling2D3 = 0 parameters
- Maxpooling2D3 -> Flat = 0 parameters
- Flat -> Dense1 = 3600 weight + 25 bias = 3625 parameters
- Dense1 -> Dense2 = 250 weight + 10 bias = 260 parameters
- Total = 7465 parameters

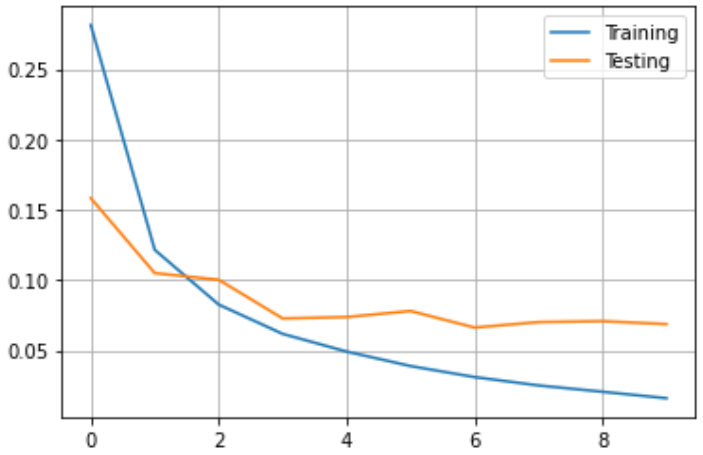
3)
Pas forcément, car un réseau peu profond peut être aussi très “large” et avoir beaucoup de paramètres,
Par contre les réseaux profonds de manière générale peuvent effectuer des tâches plus complexes à nombre de paramètres égal. Donc les mêmes tâches avec moins de paramètres.

Jonatan Baumgartner, Massimo De Santis,
Florian Feuillade

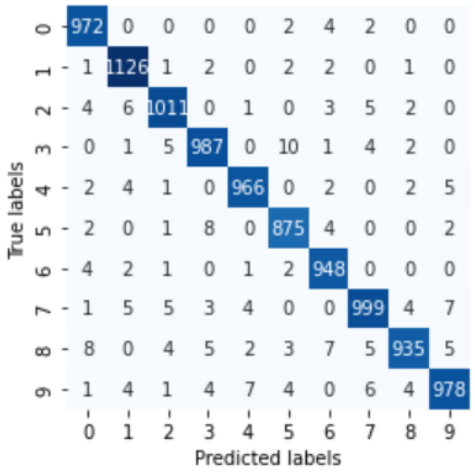
4)

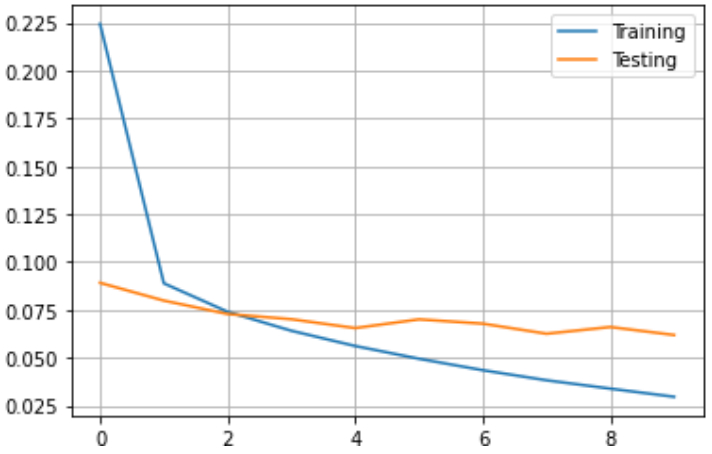
MLP Raw data : 98.1%



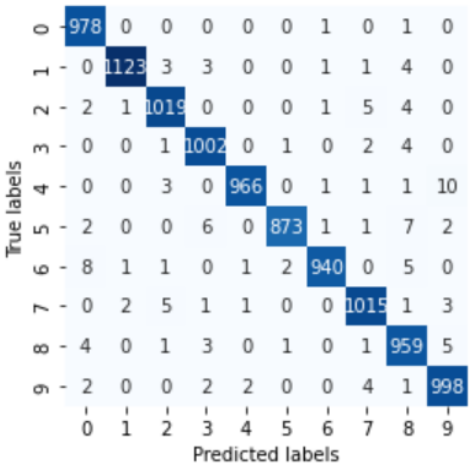


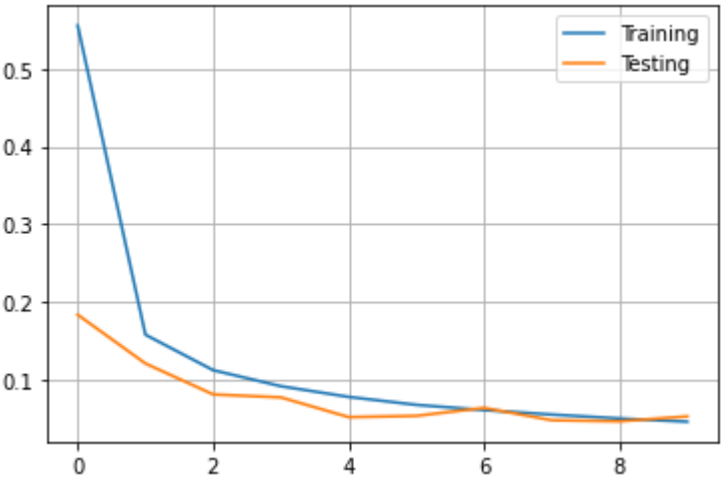
MLP HOG : 97.9%





CNN : 98.7%

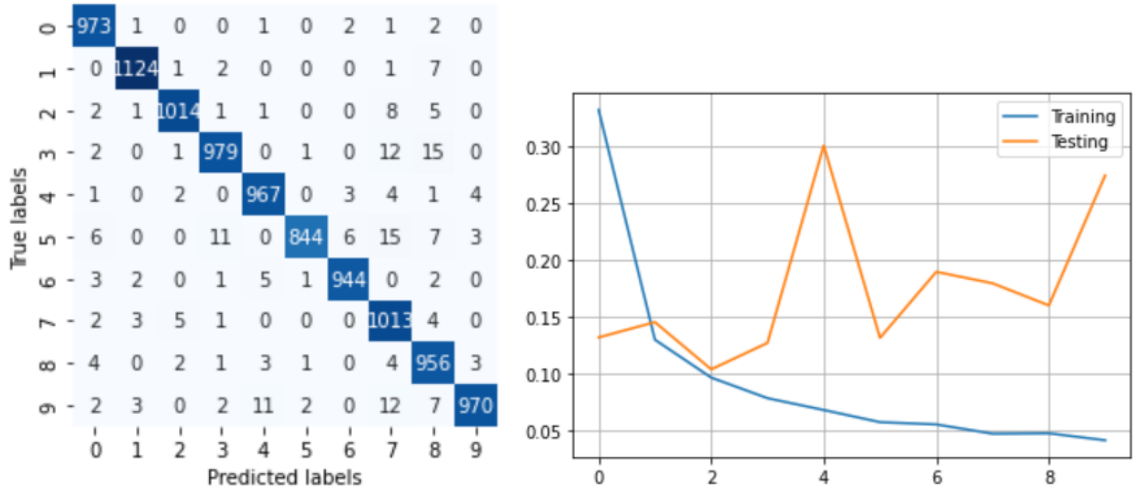




Jonatan Baumgartner, Massimo De Santis, Florian Feuillade

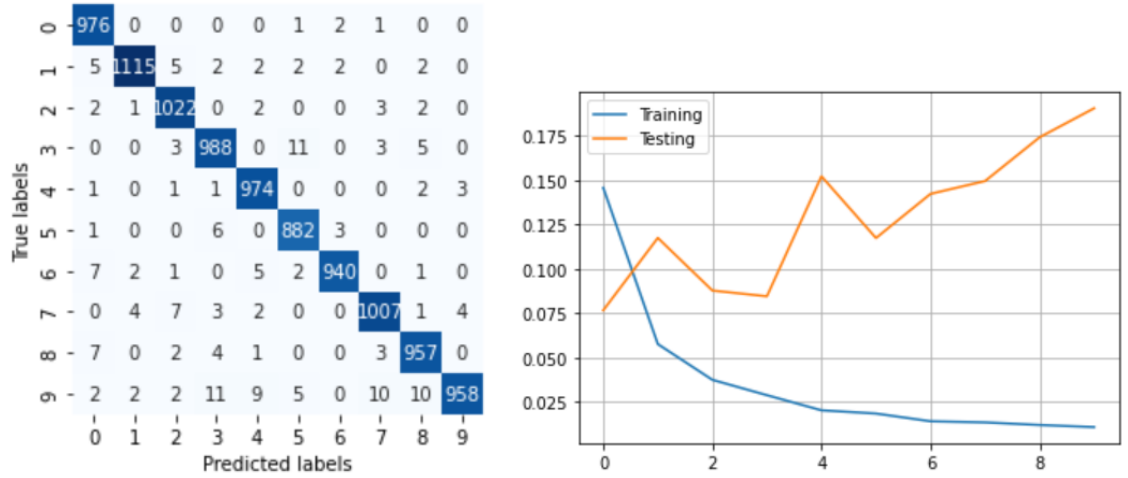
Les performances sont très bonnes pour les 3 modèles, on ne voit pas de nette différence entre eux.
De plus, les matrices de confusions indiquent qu'il ne se trompent pas particulièrement beaucoup dans un cas spécifique.

fat MLP (3 hidden layers 3000 neurons) 97.8%



Ayant beaucoup plus de paramètres, ce modèle est fortement vulnérable à l'overfitting, qu'on distingue clairement avec la test loss qui s'envole pendant que la train loss s'améliore. Malgré cela, étonnamment il maintient un bon score final de test.

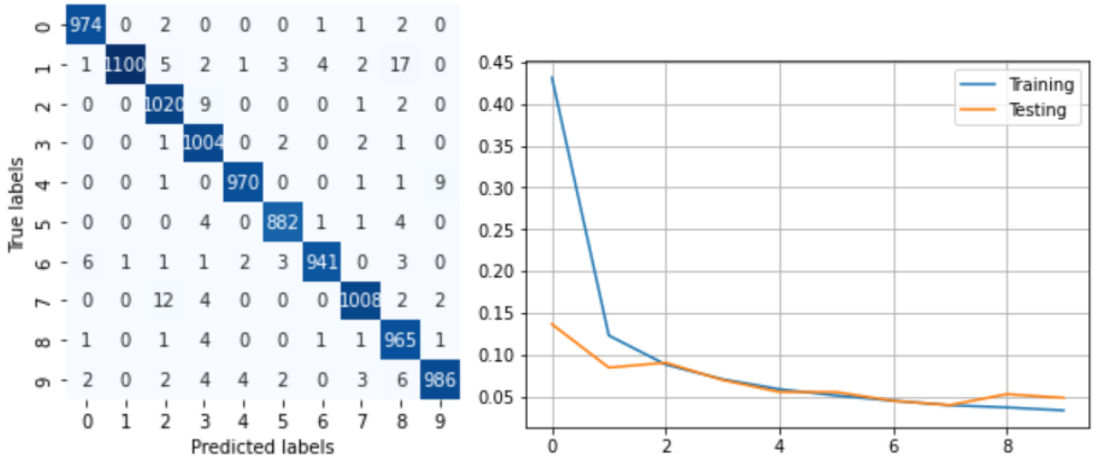
fat HOG 98.1%



De manière similaire avec un grand nombre de weights pour la version HOG, le modèle overfit très vite, mais étonnamment pour celui-ci aussi le score de test final est bon.

Jonatan Baumgartner, Massimo De Santis,
Florian Feuillade

CNN fat 98.5%



Une version du CNN avec plus de paramètres dans ce cas améliore l'entraînement, bien que l'évaluation finale ne change pas vraiment.