Formation Machine Learning

Atelier Pratique AP-ML6

ALIASE

Atelier Pratique AP-ML6 : Exercice ML6.2

ALIASE
IA - SERVICE & FORMATION

Objectif :

- Modifier le code de l'exercice **6.1** pour utiliser la fonction **sigmoïd** au lieu de la fonction **softmax**

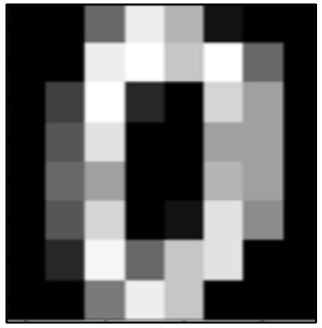
Solution → Voir MLExo6.2.py

Atelier Pratique AP-ML6 : Exercice ML6.3.1

ALIASE
IA - SERVICE & FORMATION

Tester le CNN avec le dataset des digits 8x8 :

Objectif : Classer les images en 10 classes



```
inputs = layers.Input(shape=(8, 8, 1))
Layer0 = layers.Conv2D(32, (3,3),   activation='relu')    # 32 = nb filtres
Layer1 = layers.MaxPooling2D((2,2))                       # (2,2) dimension de la matrice de pooling

model = models.Sequential( [ inputs, Layer0, Layer1 ] )

LayerA = layers.Flatten()
LayerB = layers.Dense(512, activation='relu')
LayerC = layers.Dense(10, activation='softmax')

model.add( LayerA )
model.add( LayerB )
model.add( LayerC )
```


Solution ➡ Voir MLExo6.3.1.py

Atelier Pratique AP-ML6 : Exercice ML6.3.2

ALIASE
IA - SERVICE & FORMATION

Tester le CNN avec le dataset des digits 28x28 :

Objectif : Classer les images en 10 classes

```
# Load du DATASET d'images : 70000 images de chiffres manuscrits, classés en 10 categories

(x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()

print("x_train.shape : ", x_train.shape)      # (60000, 28, 28)
print("y_train.shape : ", y_train.shape)      # (60000,)
print("x_test.shape  : ", x_test.shape)       # (10000, 28, 28)
print("y_test.shape  : ", y_test.shape)       # (10000,)

reso = 28      # résolution de l'image : 28 x 28
```


Solution ➡ Voir MLExo6.3.2.py

Atelier Pratique AP-ML6 : Exercice ML6.4

ALIASE
IA - SERVICE & FORMATION

Tester le CNN avec le dataset fashion_mnist :

Objectif : Classer les images en 10 classes

Créer un modèle avec 3 niveaux de convolution :

- **Niveau 1 : 32 filtres**
- **Niveau 2 : 64 filtres**
- **Niveau 3 : 64 filtres**

Solution → Voir MLExo6.4.py

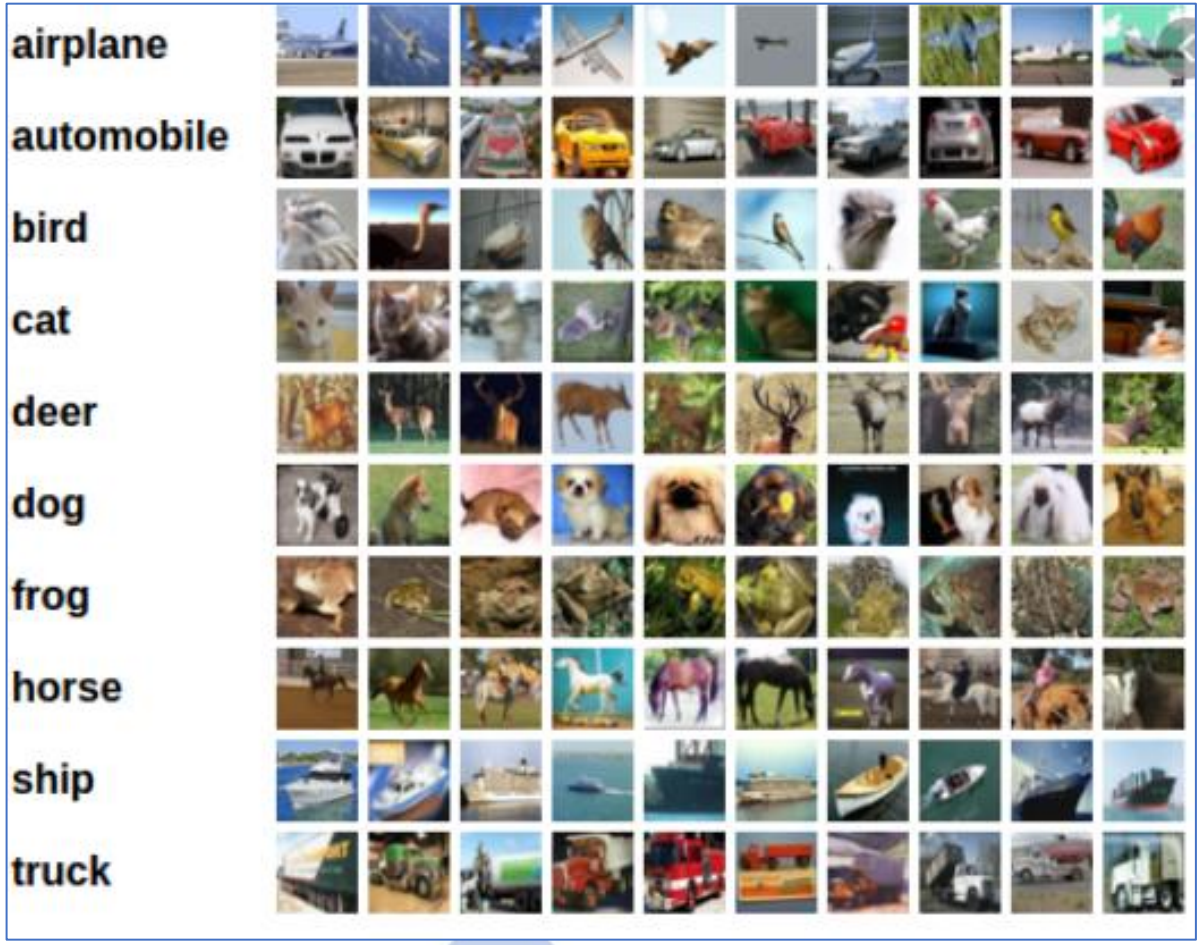
Atelier Pratique AP-ML6 : Exercice ML6.5

ALIASE
IA - SERVICE & FORMATION

Créer un programme de classification d'images : source = dataset *CIFAR10* , *nb classes : 10*

```
from tensorflow.keras.datasets import cifar10

(x_train, y_train), (x_test, y_test) = cifar10.load_data()
```



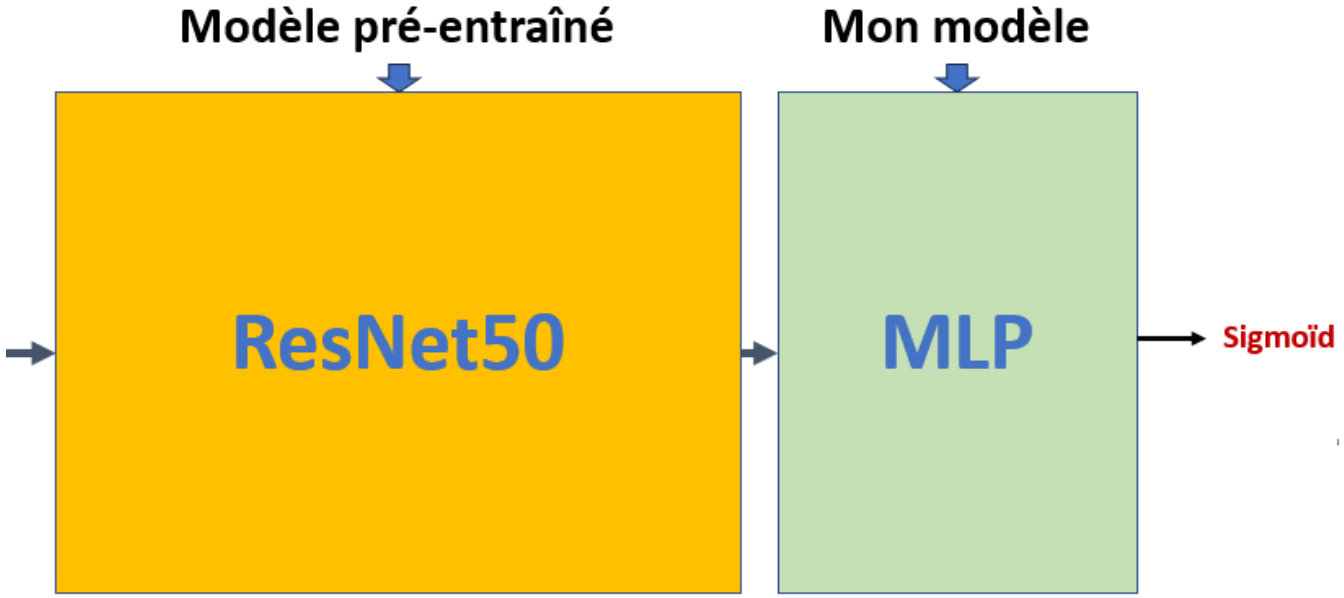
Solution → Voir MLExo6.5.py

Atelier Pratique AP-ML6 : Exercice ML6.6.4

ALIASE
IA - SERVICE & FORMATION

Objectif :

- Modifier le code de l'exercice **6.6.2** (Fine-Tunning) pour utiliser la fonction **sigmoïd** au lieu de la fonction **softmax**



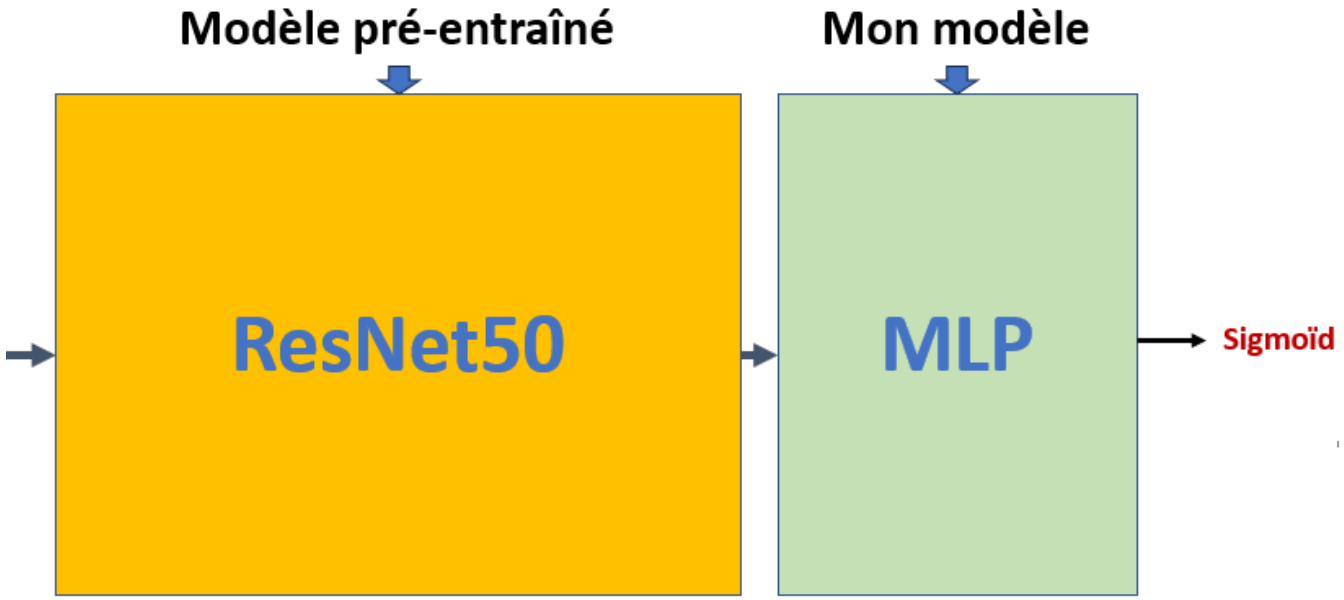
Solution → Voir MLExo6.6.4.py

Atelier Pratique AP-ML6 : Exercice ML6.6.3

ALIASE
IA - SERVICE & FORMATION

Objectif :

- Modifier le code de l'exercice **6.6.2** (Fine-Tunning) pour utiliser la format one-hot des labels



Solution → Voir MLExo6.6.3.py

Atelier Pratique AP-ML6 : Exercice ML6.6.3

ALIASE
IA - SERVICE & FORMATION

Format 1 : entiers

```
model.compile(loss='sparse_categorical_crossentropy)
```

```
model.fit( x_train, y_train))
```

y_train sont des entiers

2
4
0
3
4
1
2
1

Format 2 : one-hot

```
model.compile( loss='categorical_crossentropy'  )
```

```
model.fit( x_train, y_train_encoded))
```

y_train_encoded sont des vecteurs binaires

0	0	1	0	0
0	0	0	0	1
1	0	0	0	0
0	0	0	1	0
0	0	0	0	1
0	1	0	0	0
0	0	1	0	0
0	1	0	0	0