pretraitement

December 9, 2021

1 Pretraitement des données

[1]:
```
import tensorflow as tf
import tensorflow.keras
import numpy as np
from matplotlib import pyplot as plt

import random
```

```
2021-12-03 10:34:59.997657: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcudart.so.11.0'; dlerror: libcudart.so.11.0: cannot open
shared object file: No such file or directory
2021-12-03 10:34:59.997881: I tensorflow/stream_executor/cuda/cudart_stub.cc:29]
Ignore above cudart dlerror if you do not have a GPU set up on your machine.
```

[2]:
```
# Chargement du jeu de données
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.cifar10.load_data()
assert x_train.shape == (50000, 32, 32, 3)
assert x_test.shape == (10000, 32, 32, 3)
assert y_train.shape == (50000, 1)
assert y_test.shape == (10000, 1)
```

[3]:
```
image = random.choice(x_train)

print(f"Nombre d'images : {len(x_train)}")
print(f"Forme d'une image : {image.shape}")
print(f"Un pixel : {image[0][0]}")
print(f"Frome d'un pixel : {image[0][0].shape}")
```

```
Nombre d'images : 50000
Forme d'une image : (32, 32, 3)
Un pixel : [213 216 212]
Frome d'un pixel : (3,)
```

[4]:
```
# Normalisation
tmp = x_train.astype("float32") / np.amax(x_train)
```

[5]:
```
image = random.choice(tmp)

print(f"Nombre d'images : {len(x_train)}")
print(f"Forme d'une image : {image.shape}")
print(f"Un pixel : {image[0][0]}")
print(f"Frome d'un pixel : {image[0][0].shape}")
```

```
Nombre d'images : 50000
Forme d'une image : (32, 32, 3)
Un pixel : [0.75686276 0.8352941  0.42352942]
Frome d'un pixel : (3,)
```

[6]:
```
label = random.choice(y_train)

print("Label: ", label)
print(f"Nombre de label: {len(y_train)}")
print(f"Forme des labels : {y_train.shape}")
```

```
Label:  [0]
Nombre de label: 50000
Forme des labels : (50000, 1)
```

[7]:
```
# one hot encoding
unique_y = np.sort(np.unique(y_train))
y_tmp = np.array(list(map(lambda x: [1 if x == k else 0 for k in unique_y],␣
 ↪y_train)))
```

[8]:
```
label = random.choice(y_tmp)

print("Label: ", label)
print(f"Nombre de label: {len(y_train)}")
print(f"Forme des labels : {y_train.shape}")
```

```
Label:  [0 0 0 0 0 0 0 0 0 1]
Nombre de label: 50000
Forme des labels : (50000, 1)
```

[9]:
```
def normalize_dataset(x_train, y_train):
    # Scale images to the [0, 1] range
    x_train = x_train.astype("float32") / np.amax(x_train)

    # One hot encoding
    unique_y = np.sort(np.unique(y_train))
    y_train = np.array(list(map(lambda x:  [1 if x == k else 0 for k in␣
 ↪unique_y], y_train)))

    return x_train , y_train
```