RESEAU DE NEURONES

Notre objectif dans ce chapitre sera de développer votre intuition sur ces notions sans trop rentrer dans les détails techniques.

Avant de parler des différents composants d'un réseau de neurones, on s'intéressera à la forme des données fournies à ce réseau.

TENSEURS:

Un tenseur est un conteneur de données, généralement des données numériques. C'est donc un conteneur de nombres. Les tenseurs sont une généralisation des matrices à un nombre arbitraire de dimensions ou d'axes.

Tenseurs de rang 0:

Les tenseurs de rang 0 sont des scalaires. Dans Numpy, un nombre float32 ou flaot64 est un tenseur scalaire. Vous pouvez afficher le nombre d'axes d'un tenseur Numpy via l'attribut ndim; un tenseur scalaire a 0 axe (ndim == 0).

```
>>> import numpy as np
>>> x = np.array(12)
>>> x
array(12)
>>> x.ndim
0
```

Tenseurs de rang 1:

Les tenseurs de rang 1 sont des vecteurs. Un tenseur de rang 1 est censé avoir exactement un axe.

```
>>> x = np.array([12, 3, 6, 14, 7])
>>> x
array([12, 3, 6, 14, 7])
>>> x.ndim
1
```

Tenseurs de rang 2:

Les tenseurs de rang 2 sont des matrices. Une matrice a deux axes.

```
>>> x = np.array([[5, 78, 2, 34, 0],
                  [6, 79, 3, 35, 1],
                  [7, 80, 4, 36, 2]])
>>> x.ndim
2
```

Tenseurs de rang 3 et de rang supérieur:

Si vous regroupez ces matrices dans un nouveau tableau, vous obtenez un tenseur de rang 3, que vous pouvez interpréter visuellement comme un cube de nombres.

```
>>> x = np.array([[[5, 78, 2, 34, 0],
                   [6, 79, 3, 35, 1],
                   [7, 80, 4, 36, 2]],
                  [[5, 78, 2, 34, 0],
                   [6, 79, 3, 35, 1],
                   [7, 80, 4, 36, 2]],
                  [[5, 78, 2, 34, 0],
                   [6, 79, 3, 35, 1],
                   [7, 80, 4, 36, 2]]])
>>> x.ndim
3
```

En regroupant les tenseurs de rang 3 dans un tableau, vous pouvez créer un tenseur de rang 4, et ainsi de suite. En général on manipulera des tenseurs de rang compris entre 0 et 4.

Exemples concrets de tenseurs de données

Les données qu'on manipulera appartiendront presque toujours à l'une de ces catégories.

Données vectorielles:

Tenseur de rang 2 où on a la forme suivante: (samples,features). Chaque échantillon a un vecteur d'attributs 'features'.

	X1	X2	X3	X4
Y1	0	4	1	7
Y2	2	3	5	6
Y3	8	4	9	10

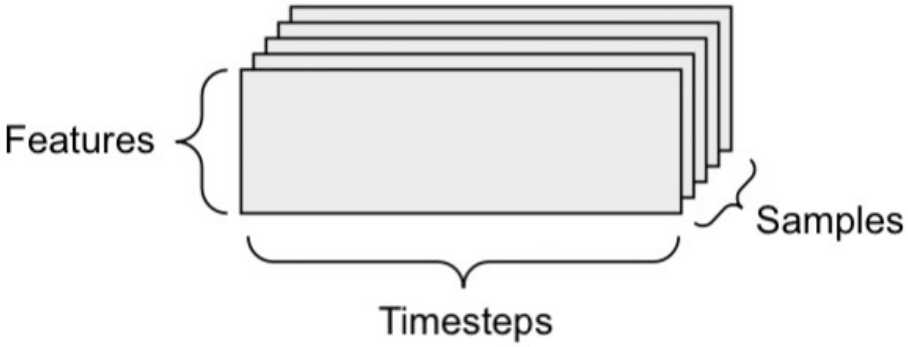
Le tenseur représentant ce jeu de données est le suivant:

```
[[0, 4, 1, 7],
 [2, 3, 5, 6],
 [8, 4, 9, 10]]
```

C'est un tenseur de rang 2 de taille: (3, 4)

Données temporelles ou séquentielles:

Lorsque la notion de temps ou de séquences est importante dans notre jeu de données , il est judicieux de les stocker dans un tenseur de rang 3 avec un axe temporel explicite.



Prenons comme exemple une action boursière. Cette action a pour attributs 'open', 'close', 'high', 'low', 'volume'.

Supposons qu'on veut étudier l'action sur une durée de 20 jours et Pour chaque jour on a la valeur des attributs minute par minute (sachant que dans une journée de négociation on a 390 minutes) . Dans ce cas notre tenseur sera de taille (20, 390, 5).

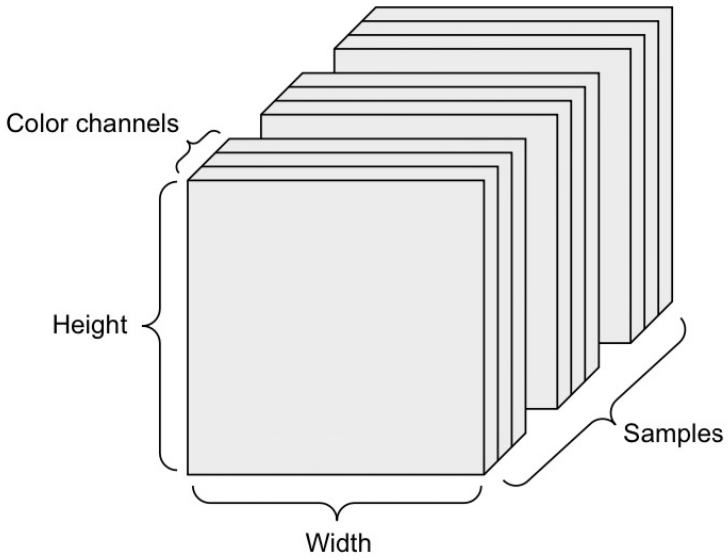
Tenseur représentatif:

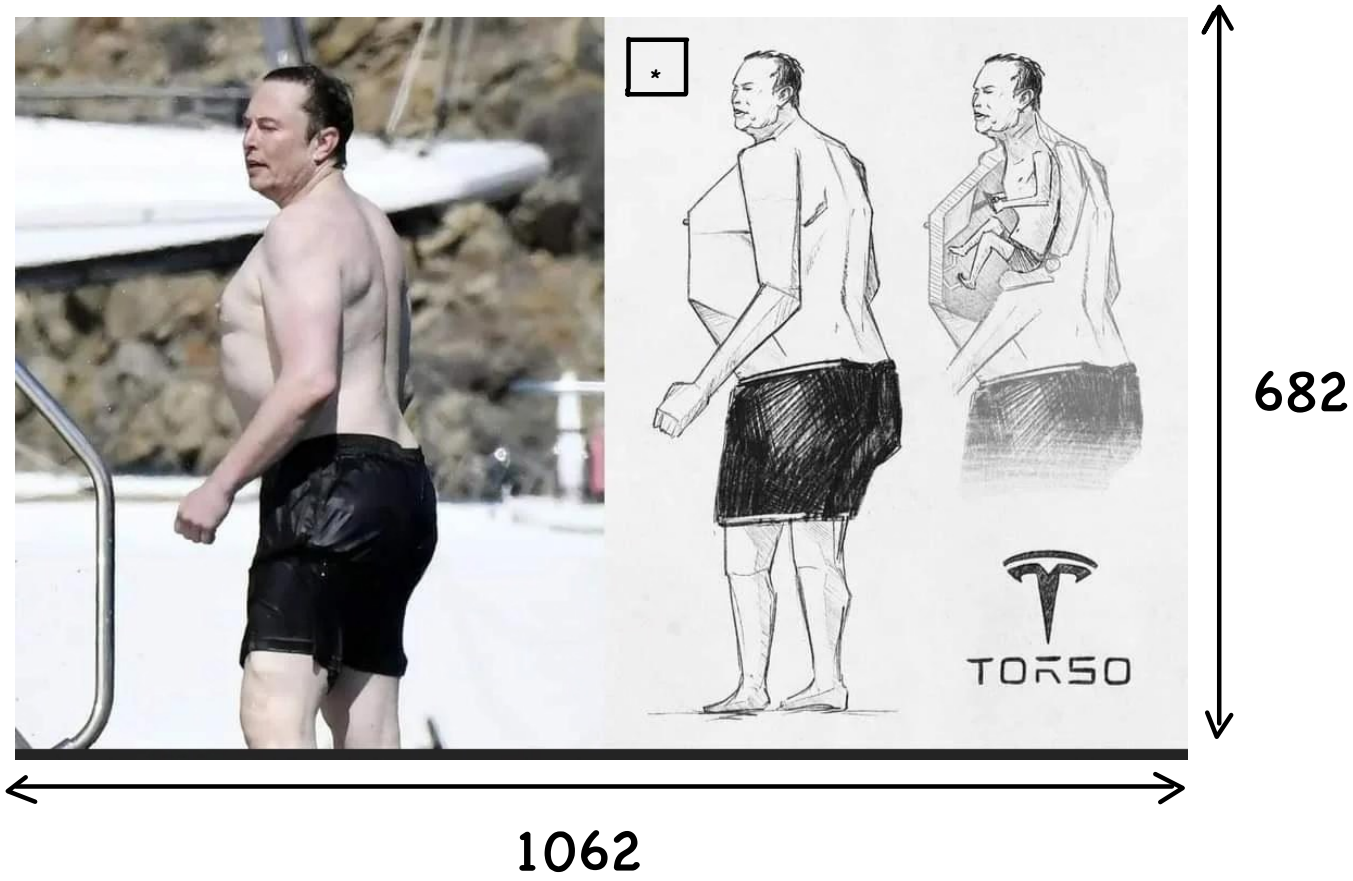
```
[ [ [1200, 1205, 1203, 1200, 20],
    ……….
    [1205, 1203, 1207, 1199, 15]
  ],
  ……..
  [ [1210, 1205, 1211, 1204, 32],
    ……….
    [1209, 1203, 1210, 1202, 23] ] ]
```

Images:

Les images ont généralement trois dimensions : la hauteur, la largeur et la profondeur de couleur. Notre base de données sera de la forme suivante :

(samples, height, width, color_depth)

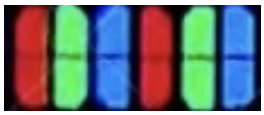


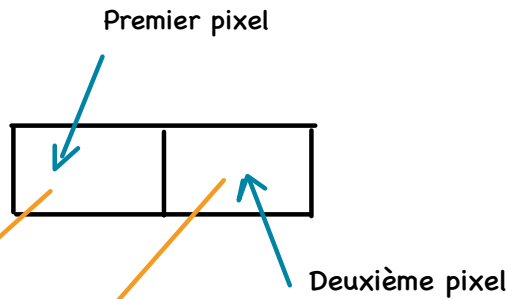


Afin de représenter cette seule image on aura besoin d'un tenseur de taille (1062, 682, 3)

https://www.colorspire.com/rgb-color-wheel/

Tenseur représentatif de * :





[[[255, 255, 255],

[255, 255, 255]]]

Vidéos:

Pour une base de données de vidéos on aura un tenseur de rang 5 de la forme:

(samples, frames, height, width, color_depth)

https://www.youtube.com/watch?v=VN64iCNlrdY