

## Mon premier réseau de neurones from scratch

Prérequis : Base en Programmation Orientée Objets (POO) en Python

### Qu'est ce qu'un réseau de neurones ?

La plupart des articles compare ce concept à celui du cerveau humain. Cette image, bien que parlante, est assez éloignée de la réalité.

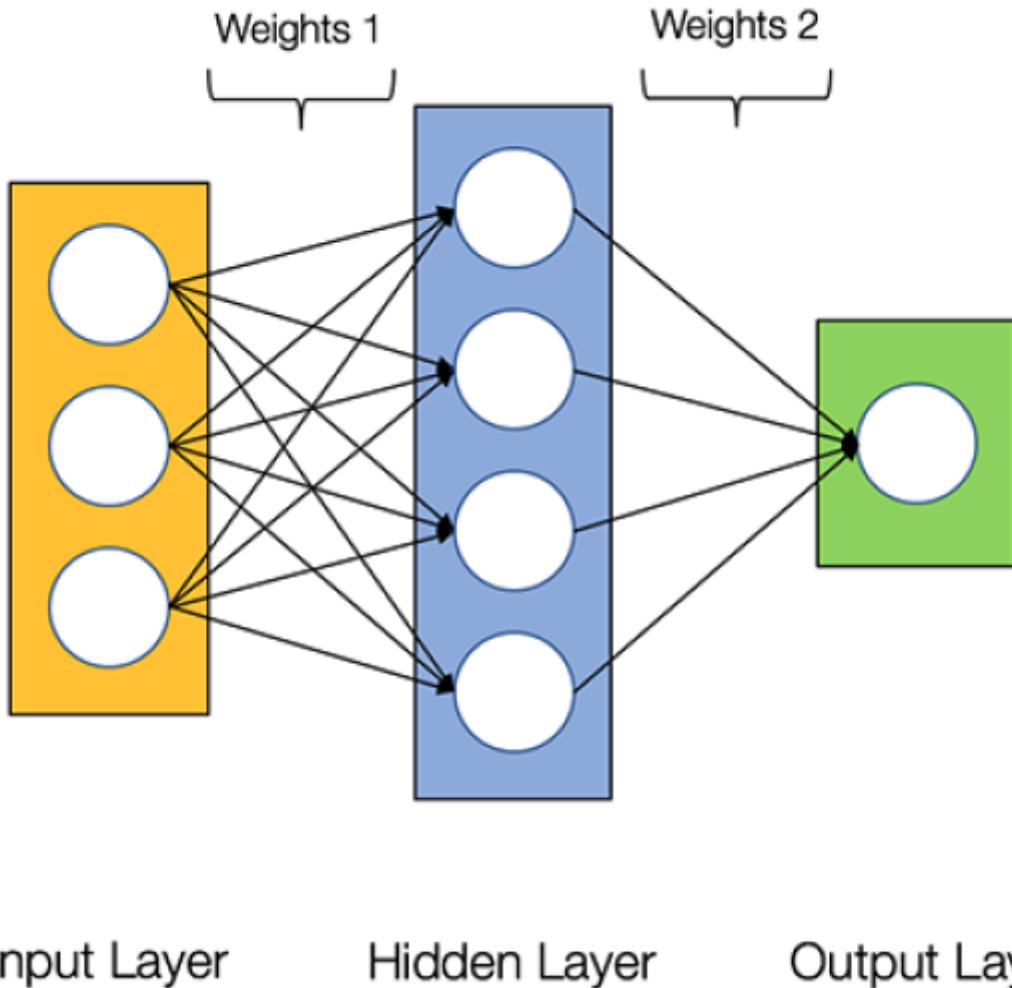
Il est plus facile de décrire un réseau de neurones comme une fonction mathématique qui à partir d'un élément initialement donné (input) renvoie un élément (output) désiré.

Ainsi un réseau de neurones est constitués des composants suivants :

- Une couche d'entrée (**input layer**),  $x$
- Un nombre arbitraire de couches cachées (**hidden layers**)
- Une couche de sortie (**output layer**),  $\hat{y}$
- Un set de poids (**weights**) et de préjugés (**biases**) entre chaque couche,  $W$  et  $b$
- Le choix d'une fonction d'activation (**activation function**) pour chaque couche cachée,  $\sigma$

*Nous choisirons ici la fonction **Sigmoïde** en tant que fonction d'activation.*

Le graphique suivant nous montre l'architecture d'un 2-layer Neural Network (N.B: on exclut l'input layer lorsque l'on compte le nombre de couche dans un réseau de neurones).



Input Layer

Hidden Layer

Output Layer

## Création de l'objet réseau de neurones

Commençons par importer les librairies dont nous aurons besoin :

```
import numpy as np
```

Puis créons la classe selon les composants dont nous avons parlé plus tôt :

```
class NeuralNetwork:  
    def __init__(self,x,y):  
        self.input      = x  
        self.weights1  = np.random.rand(self.input.shape[1],4)  
        self.weights2  = np.random.rand(4,1)  
        self.y         = y  
        self.output    = np.zeros(y.shape)
```

## Entraînement de notre réseau de neurones

L'output  $\hat{y}$  d'un 1-layer Neural Network est  $\hat{y} = \sigma(W_1x + b_1)$

L'output  $\hat{y}$  d'un 2-layer Neural Network est donc  $\hat{y} = \sigma(W_2\sigma(W_1x + b_1) + b_2)$

On remarque dans les équations ci-dessus que les poids **W** et les préjugés **b** sont les seules variables dont dépend  $\hat{y}$ .

Naturellement, trouver les bonnes valeurs pour les poids et les préjugés feront la force de prédiction de votre réseau. Le procédé d'ajustement de ces valeurs à partir des données de départ est appelé **entraînement du réseau de neurones**.

From:

<https://wiki.centrale-med.fr/ginfo/> - Wiki GInfo



Permanent link:

[https://wiki.centrale-med.fr/ginfo/projets:creer\\_son\\_premier\\_reseau\\_de\\_neurone\\_from\\_scratch](https://wiki.centrale-med.fr/ginfo/projets:creer_son_premier_reseau_de_neurone_from_scratch)

Last update: **12/06/2019 14:44**