



# DEEP LEARNING



*La réussite comme produit d'essais et d'erreurs.*

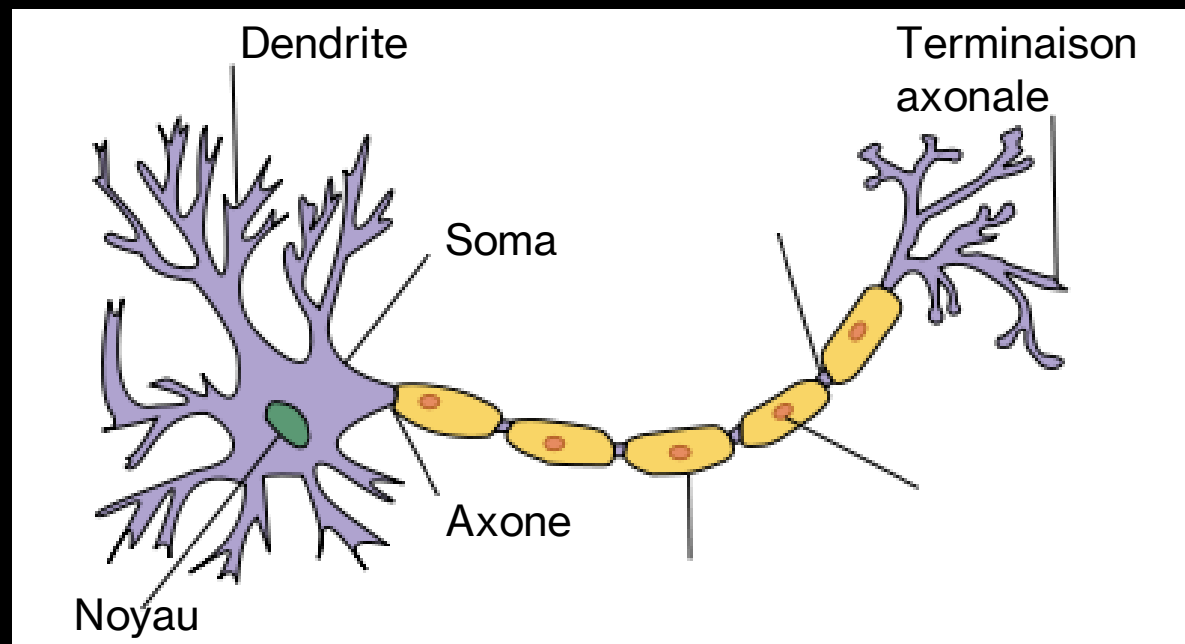
# Deep Learning ou Apprentissage Profond

## Qu'est-ce que c'est?

Le Deep learning, ou Apprentissage Profond, est l'une des technologies principales du Machine Learning. Avec le Deep Learning, nous parlons d'algorithmes capables de mimer les actions du cerveau humain grâce à des réseaux de neurones artificielles. Les réseaux sont composés de dizaines voire de centaines de «couches» de neurones, chacune recevant et interprétant les informations de la couche précédente.

# Neurone biologique

Un **neurone** est une cellule constituant l'unité fonctionnelle de la base du système nerveux. Il assure la transmission de signaux bioélectriques.



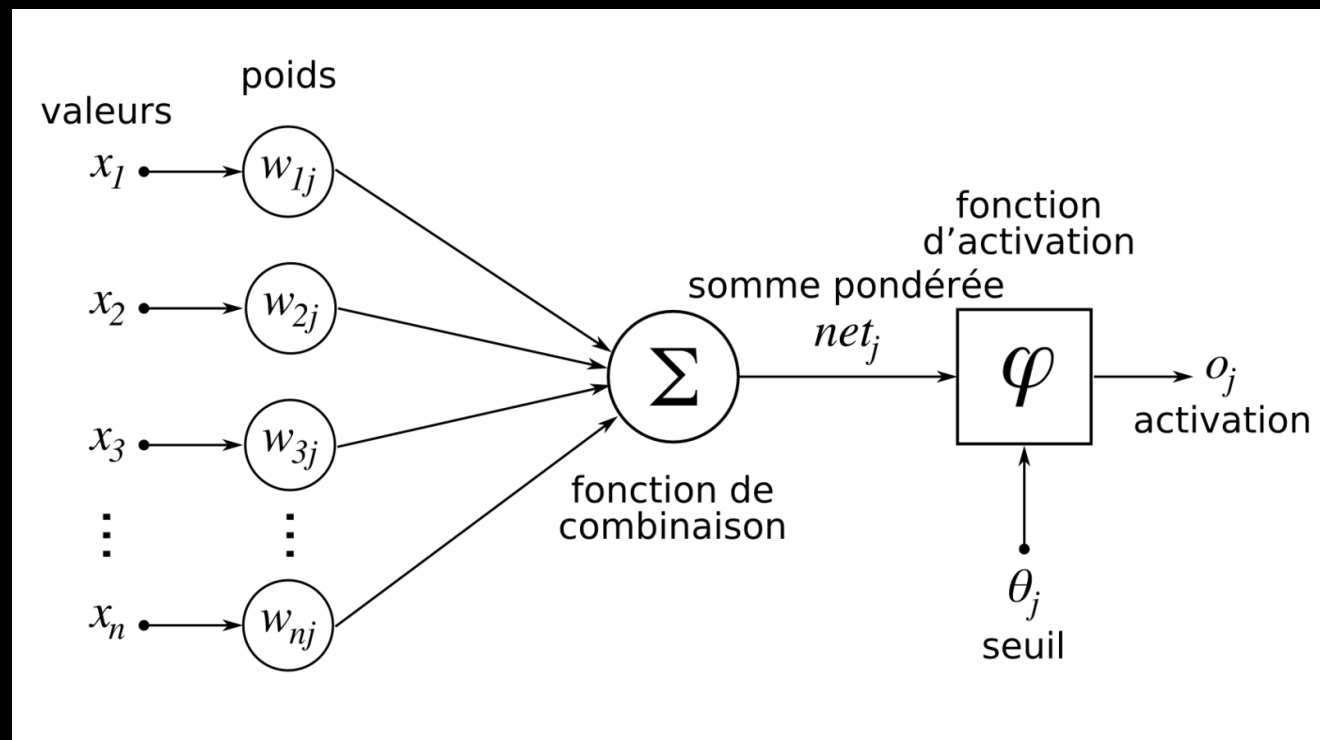
Le neurone a deux propriétés physiologiques :

- l'**excitabilité**, c'est-à-dire la capacité de répondre aux stimulations et de convertir celles-ci en impulsions nerveuses;
- la **conductivité**, c'est-à-dire la capacité de transmettre les impulsions.

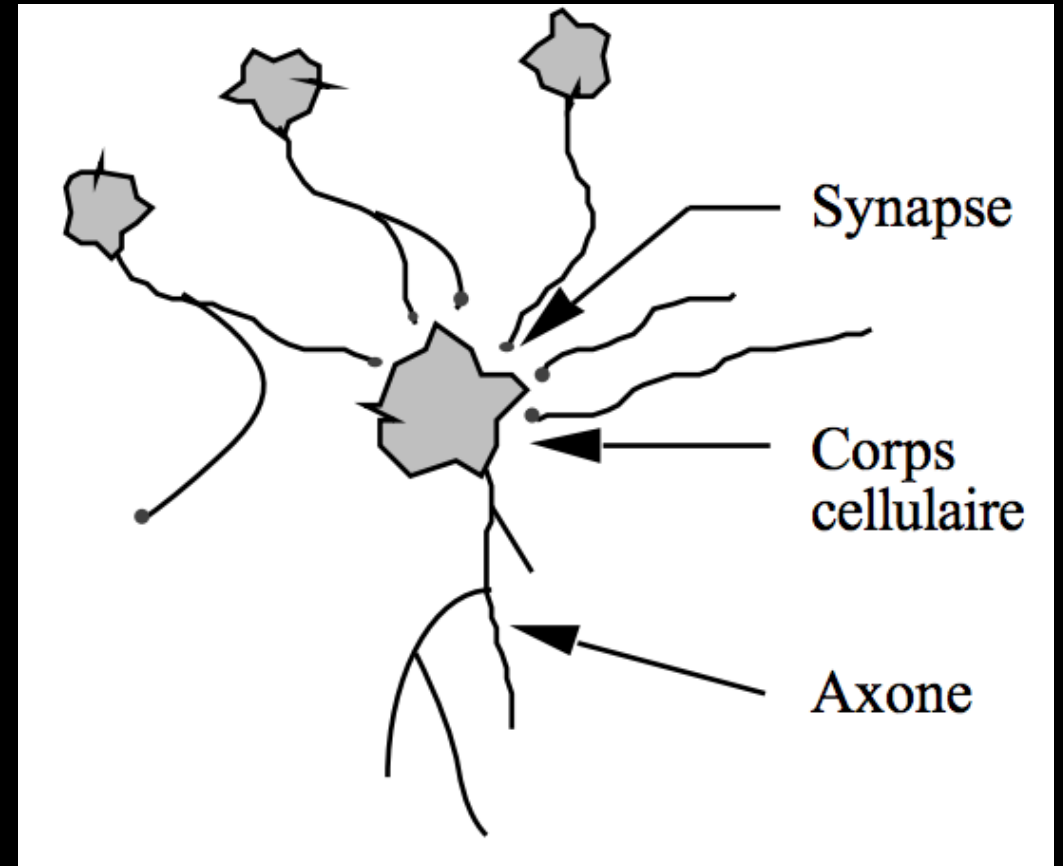
# Neurone artificiel

Un **neurone formel**, ou **neurone artificiel**, est une représentation mathématique et informatique d'un neurone biologique.

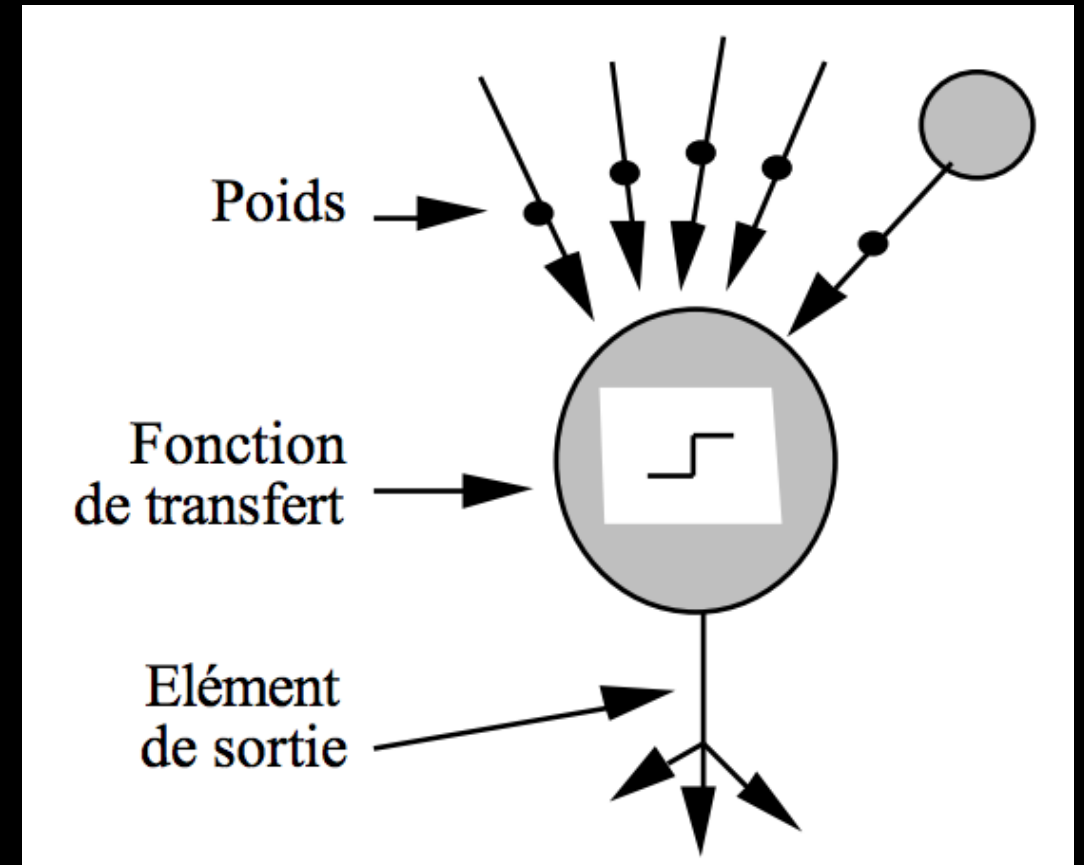
Il possède généralement **plusieurs entrées** et **une seule sortie**, qui correspondent respectivement aux **dendrites** et au cône d'émergence (**axone**) du neurone biologique.



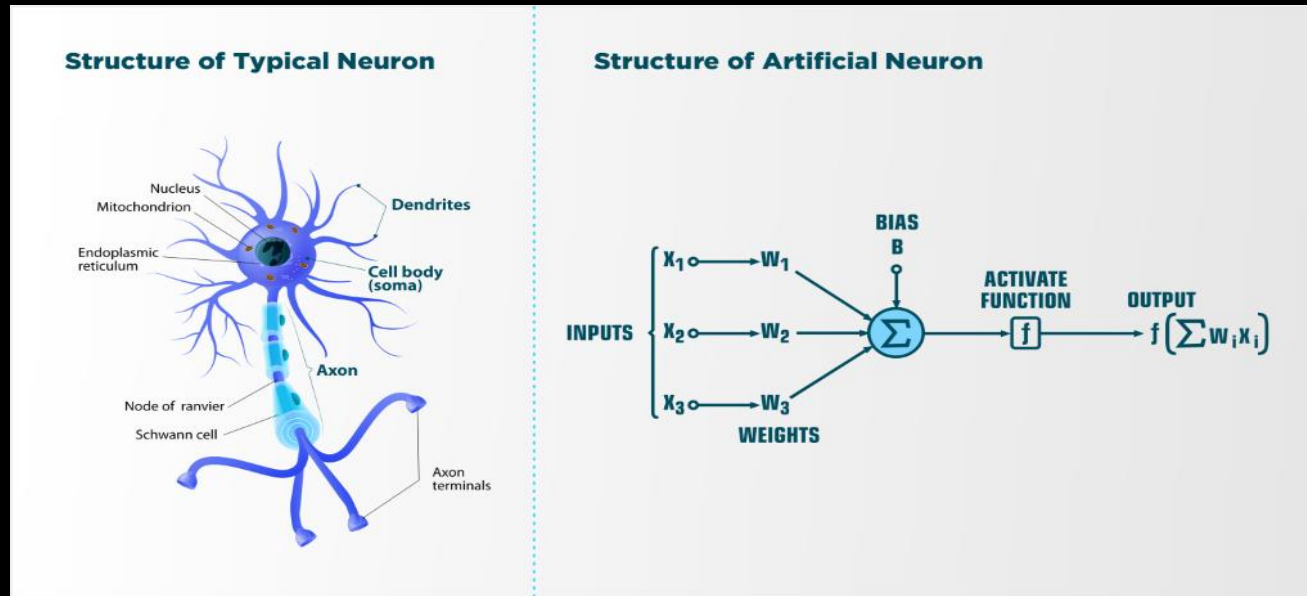
Au niveau d'une zone fonctionnelle de contact, appelée **synapse**, le neurone biologique reçoit des signaux transmis par d'autres neurones. Au niveau du corps (**soma**), le neurone analyse et traite ces signaux en les sommant. Si le résultat obtenu atteint ou est supérieur au **seuil d'activation** (ou d'excitabilité), il envoie une décharge -alors nommée potentiel d'action- le long de son **axone** vers d'autres neurones biologiques.



Un neurone formel, au même titre qu'un neurone biologique, reçoit plusieurs stimuli. Les actions excitatrices et inhibitrices des synapses du neurone biologique sont ici représentées, la plupart du temps, par des coefficients numériques: les **poids synaptiques**. Chaque poids est **associé à une entrée**. Dans sa version la plus simple, un neurone formel calcule la **somme pondérée** des entrées reçues, puis applique à cette valeur une **fonction d'activation**, généralement non linéaire. La valeur finale obtenue est la **sortie** du neurone.



# Réseaux de neurones biologiques VS Réseaux de neurones artificiels

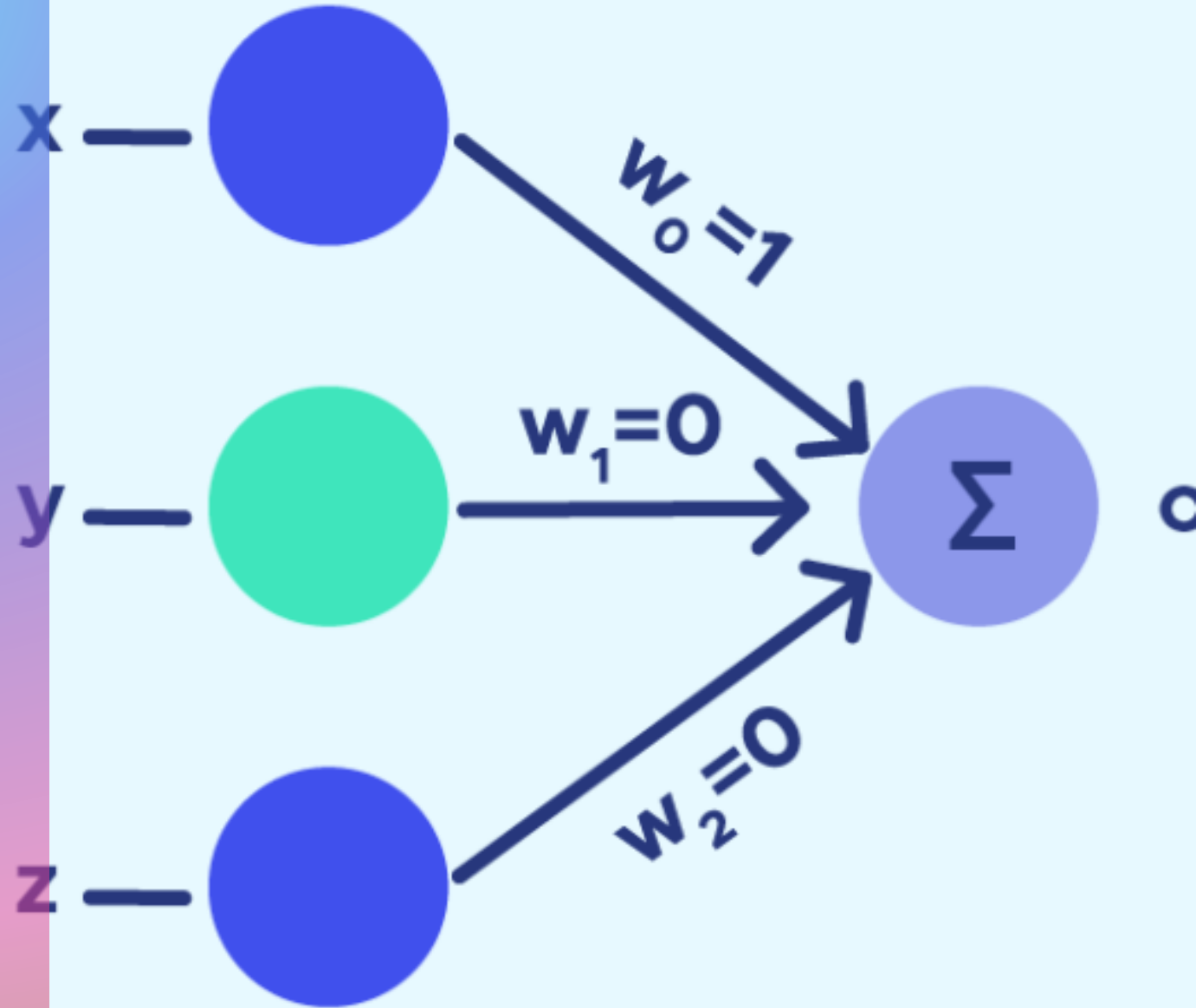


- Une réelle différence du nombre de neurones ; Chez l'humain des milliards de neurones et seulement des dizaines de milliers en artificiels;
- BNNs ont des trillions de paramètres ajustables alors que qu'en ANNs elles n'ont que quelques millions;
- ANNs utilise pour apprendre la descente de gradient alors qu'en BNNs les chercheurs n'ont pas encore une connaissance certaine du fonctionnement;
- La vitesse de traitement est beaucoup plus important en BNNs;
- BNNs consomme beaucoup moins d'énergie que ANNs



# Perceptron monocouche

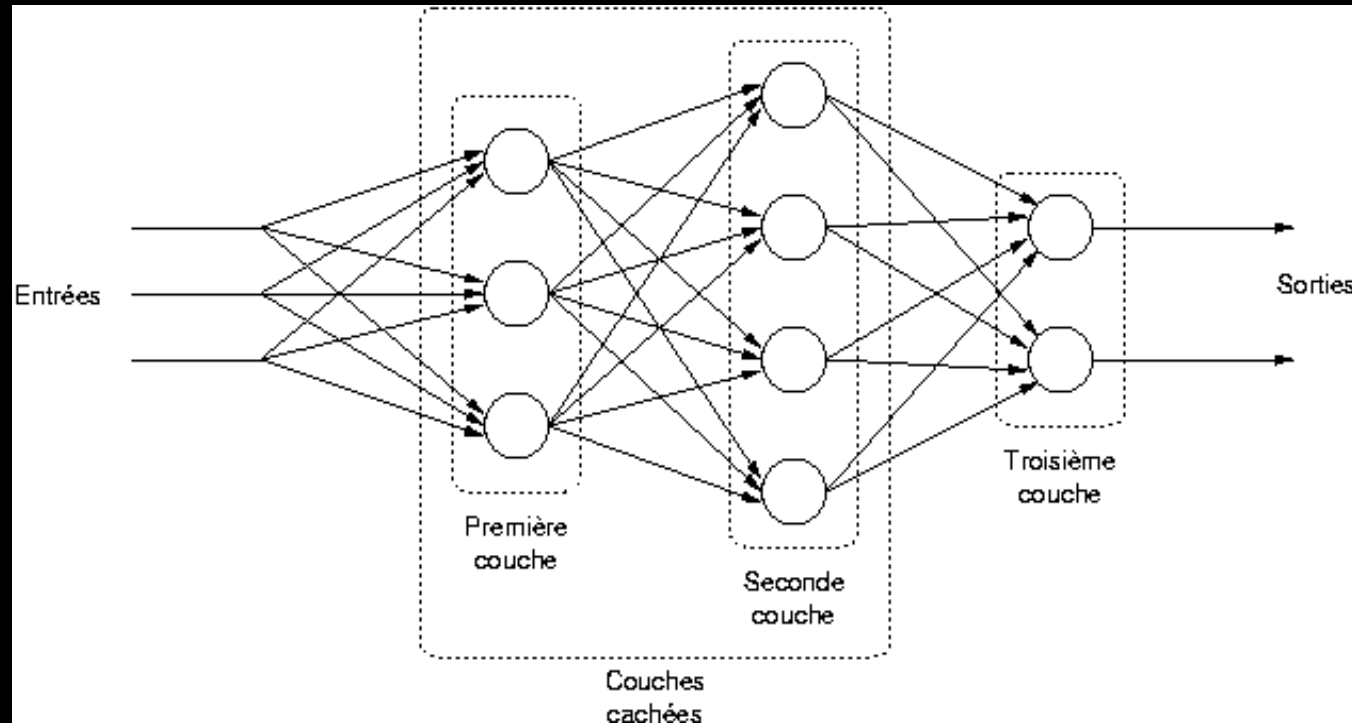
- Le perceptron est un algorithme d'apprentissage supervisé de classifieurs binaires
- Le perceptron à une seule couche séparant les classes de façon linéaire.






# Perceptron multi-couche

- Un perceptron **multicouche** permet de classier des groupes qui ne sont **pas séparables de façon linéaire**.
- On l'utilise pour résoudre des problèmes que les algorithmes à une seule couche ne peuvent résoudre.
- La première couche est reliée aux entrées, puis ensuite chaque couche est reliée à la couche précédente. C'est la dernière couche qui produit les sorties du PMC.

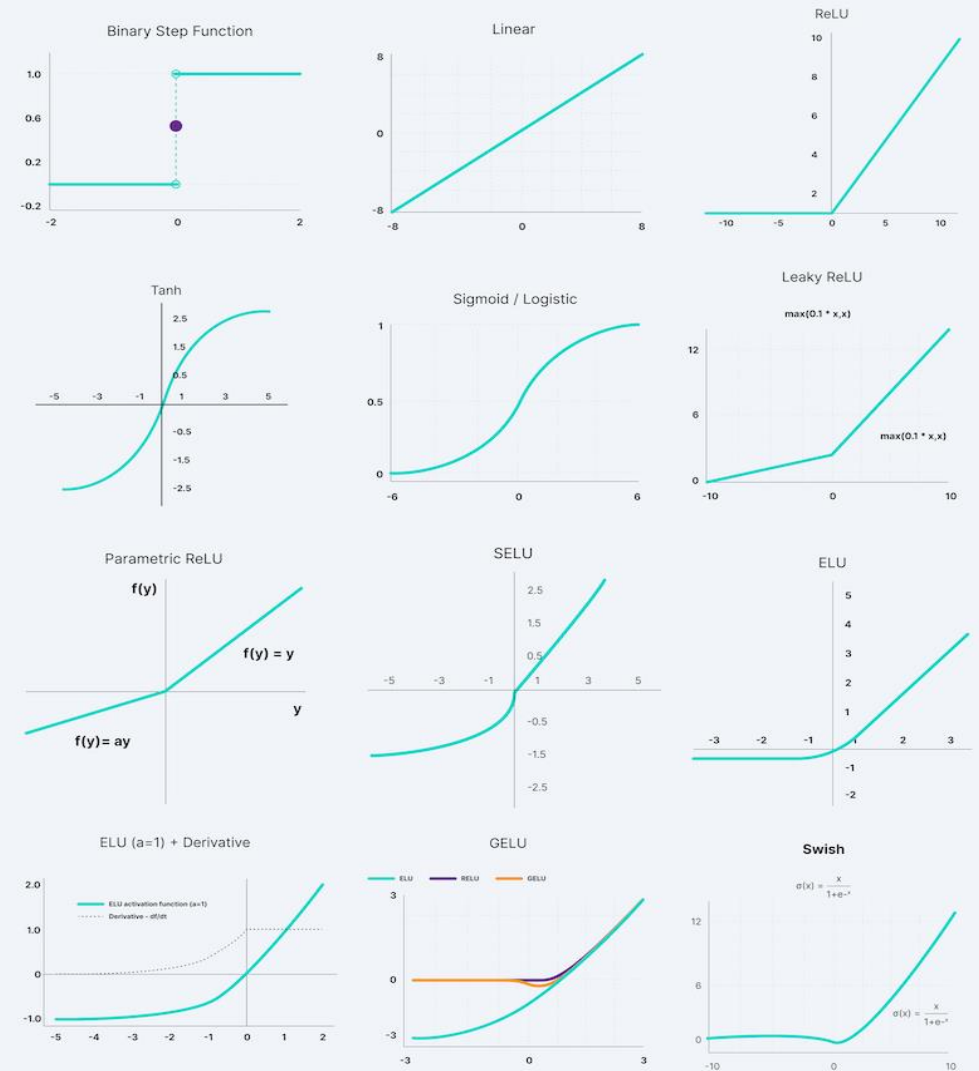


- L'ensemble des poids détermine le fonctionnement du réseau de neurones. Les motifs sont présentés à un sous-ensemble du réseau de neurones : la **couche d'entrée**. Lorsqu'un motif est appliqué à un réseau, celui-ci cherche à atteindre un état stable. Lorsqu'il est atteint, les valeurs d'activation des neurones dans la **couche de sortie** constituent le résultat. Les neurones qui ne font ni partie de la couche d'entrée ni de la couche de sortie sont dits **neurones cachés**.
- Un neurone a une seule sortie, une valeur unique qui est envoyé à plusieurs neurones de la couche suivante.
- Il n'y a pas de connexion entre les neurones d'une même couche. Les connexions ne se font qu'avec les couches amont ou aval.

# Fonctions d'activation

- I.  **Les fonctions d'activations** aident le réseau de neurones à utiliser les informations importantes tout en supprimant les informations sans intérêt.
- II. Elles apportent une approche **NON** linéaire.
- III. Elles décident si un neurone doit être activé ou pas.

## Neural Network Activation Functions



# Les 3 les plus connues

## ➤ Binary Step

- Si la somme est supérieure à 0 alors la fonction transforme en 1 et si elle est inférieure à 0 elle la transforme en 0

*Binary step*

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$$

## ➤ Rectified Linear Unit ReLU

- Retourne 0 si l'entrée est négative sinon retourne la valeur d'entrée.

*ReLU*

$$f(x) = \max(0, x)$$

## ➤ Sigmoid Logistic

- Elle prend n'importe quelle valeur en entrée et renvoie une valeur comprise entre 0 et 1

*Sigmoid / Logistic*

$$f(x) = \frac{1}{1 + e^{-x}}$$

# Régression avec perceptron multicouche

# Les différentes étapes du process

Préparation des données

Partitionnement des données

Features Scaling

Entrainement du modèle

Evaluation des performances du modèle sur le jeu d'apprentissage

- Erreur quadratique moyenne
- Racine carrée de l'erreur quadratique moyenne
- Coefficient de détermination
- Droite de régression

Synthèse des performances du modèle sur le jeu d'apprentissage

Evaluation des performances du modèle sur le jeu de test

- Erreur quadratique moyenne
- Racine carrée de l'erreur quadratique moyenne
- Coefficient de détermination
- Droite de régression

Synthèse des performances du modèle sur le jeu d'apprentissage





# ***Classification binaire et multiclasse***



# Classificateur binaire

On crée notre réseau de neurones qui aura autant d'entrées que de features ;

On détermine le nombre hidden layers (souvent 2, pour éviter un temps trop important de calcul) et son nombre de neurones contenu à l'intérieur (souvent 512)

INCLURE une fonction d'activation de type sigmoïde

En output, on crée le neurone de sortie qui sera notre prédiction

On utilise, compile() sur le modèle afin d'optimiser le mean squared error, optimizer, et metrics.

On paramètre la loss function en : binary\_crossentropy qui est construite exprès pour les problèmes binaires

On fit le train dans le network et on laisse opérer la magie :)

# Classificateur multiclasse

On crée notre réseau de neurones qui aura autant d'entrées que de features (pour une photo exemple (810x606 pixels))

On détermine le nombre hidden layers (on double le nombre de features)

En output, on crée les neurones de sortie qui seront au nombre de features à prédire

On utilise la fonction d'activation Softmax pour nous prédire une probabilité.

On initialise les poids en utilisant un court intervalle exemple: [ 0:1]

On paramètre la fonction cout en categorical\_crossentropy qui pendant l'entraînement a pour rôle de pénaliser les erreurs en output

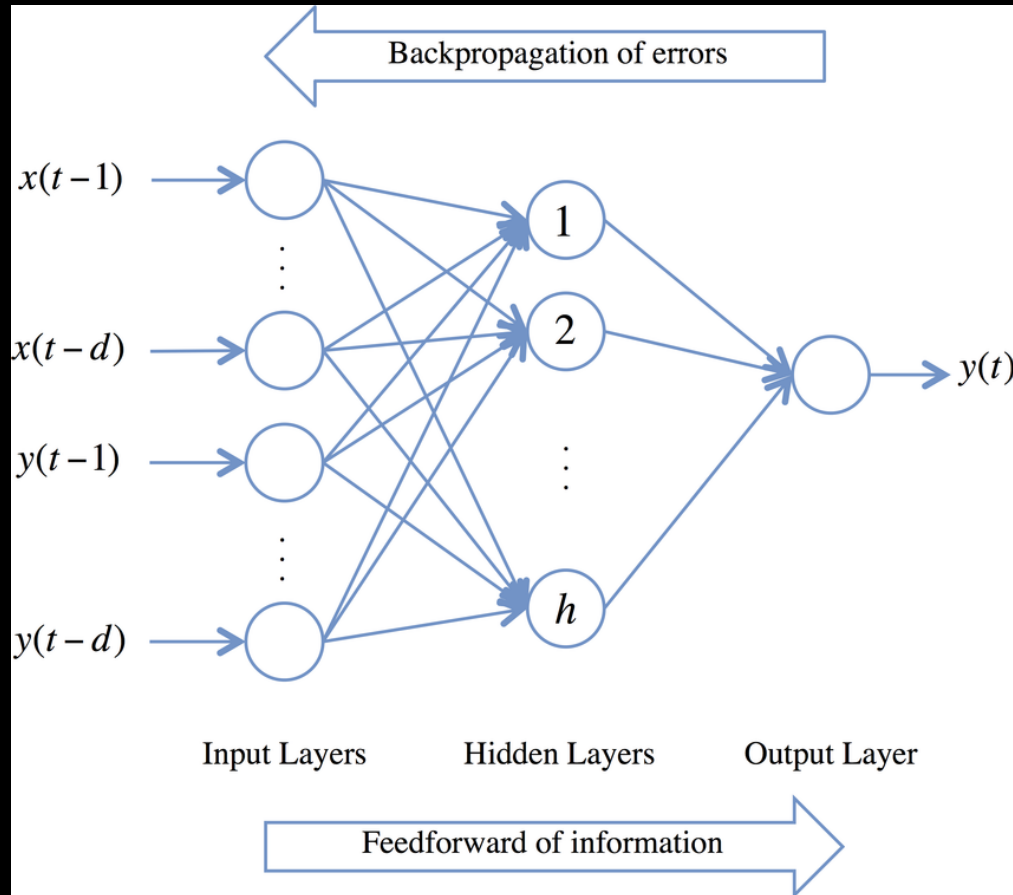
Une bonne pratique est d'utiliser avant l'entraînement un one-hot-encoder

On performe la back propagation

# Feedforward et Backpropagation

- Un réseau neuronal Feed-Forward est un type d'architecture de réseau neuronal dans lequel les connexions sont "alimentées en avant", c'est-à-dire qu'elles ne forment pas de cycles.
- La backpropagation est un algorithme de formation qui se compose de deux étapes : 1) transmettre les valeurs 2) calculer l'erreur et la propager vers les couches précédentes. Donc, pour être précis, le feedforward fait partie de l'algorithme de rétropropagation mais vient avant la propagation en arrière.

# Feedforward Propagation & Backpropagation







# ***Architecture du reseau de neurones***

# REGRESSION

avec perceptron multi-couche

type couche	nombre couches	nombre neurones par couche	fonction d'activation	fonction coût
entrée	1	autant que de features	Identité / il n'y en a pas / $y=x$	MSE ou encore mieux: RMSE
cachées intermédiaires	à définir	à définir	à définir	
sortie	1	1	Identité / il n'y en a pas / $y=x$	

# CLASSIFICATION

avec perceptron multi-couche

Binaire

type couche	nombre couches	nombre neurones par couche	fonction d'activation	fonction coût
entrée	1	autant que de features	Identité / il n'y en a pas / $y=x$	Binary Cross Entropy
cachées intermédiaires	à définir	à définir	à définir	
sortie	1	1	<b>sigmoïde</b> ou (mais autorisé à utiliser <b>softmax</b> )	

Multi-classes

entrée	1	autant que de features	Identité / il n'y en a pas / $y=x$	Categorical Cross Entropy
cachées intermédiaires	à définir	à définir	à définir	
sortie	1	1	<b>softmax</b> (n°classes >2)	