

## Chapitre 1

# Notions préliminaires

---

### 1- Introduction

Inspirés par la structure du cerveau humain, les réseaux de neurones artificiels ont été beaucoup appliqués dans des domaines tels que la reconnaissance des formes, l'optimisation, le codage, le contrôle ... etc., principalement à cause de leur aptitude à résoudre des problèmes classés difficiles. Cette aptitude leur est prodiguée par le fait qu'ils apprennent directement et automatiquement des données, sans besoin de formulation a priori.

Un réseau de neurone est généralement constitué d'un grand nombre de processeurs simples, les neurones, qui sont reliés les uns aux autres par des connections. Il apprend à résoudre des problèmes en ajustant les poids des interconnections d'une manière adéquate et en conformité avec les données. Plus que cela, le réseau de neurone, en apprenant, s'adapte facilement aux environnements nouveaux et peut traiter l'information bruitée, inconsistante, vague ou sujette à aléa.

L'hypothèse de base qui a présidé à l'émergence des réseaux de neurones, à l'image du cerveau, est que le comportement intelligent vient de la structure d'ensemble d'une part et, d'autre part du comportement des éléments de base. L'intérêt des neurones réside dans les propriétés qui résultent de leur association en réseaux. C'est l'architecture même du réseau, combinant le comportement des processeurs de base, qui lui octroie la capacité d'exécuter des fonctions jusqu'alors inédites telles que l'adaptation ou l'apprentissage.

Mais vus sous la perspective de la science statistique, les réseaux de neurones peuvent être perçus comme des extensions des techniques conventionnelles qui ont été développées depuis des décennies.

Aujourd'hui, avec la maturité grandissante du domaine, les qualités des neurones artificiels qui, au début s'apparentaient vaguement à celles des neurones biologiques, bénéficient de fondements statistiques solides.

C'est sous cet éclairage que nous allons d'abord décrire ce qu'est un neurone artificiel et ce que peut être son comportement. Nous étudierons ensuite différents types de réseaux, leurs architectures et leurs propriétés.

## 2- Différents neurones :

### 2.1 Neurone biologique

Le neurone biologique est une cellule vivante spécialisée dans le traitement des signaux électriques, c'est une cellule composée d'un corps cellulaire et d'un noyau.

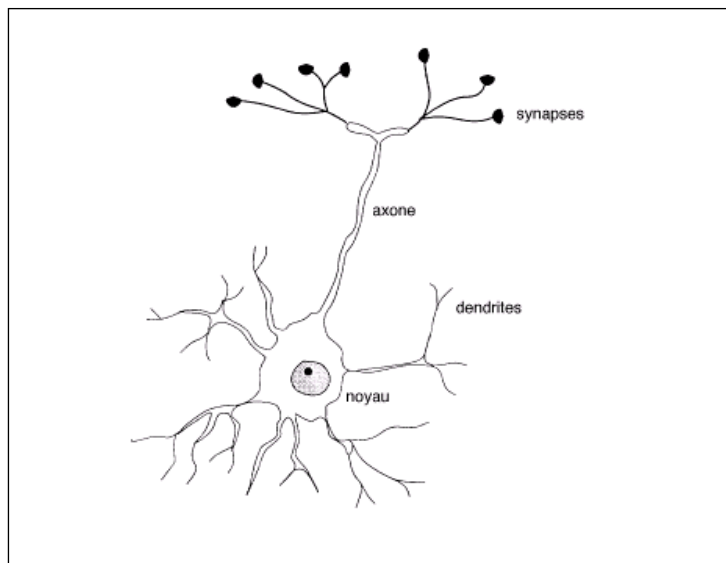
La structure d'un neurone se compose de trois parties :

- **La somma** : ou cellule d'activité nerveuse, au centre du neurone.
- **L'axone** : attaché au somma qui est électriquement actif, ce dernier conduit l'impulsion produite par le neurone.
- **Dendrites** : électriquement passives, elles reçoivent les impulsions d'autres neurones.

C'est par les dendrites que l'information pénètre de l'extérieur vers le soma, corps du neurone. L'information traitée par le neurone chemine ensuite le long de l'axone (unique) pour être transmise aux autres neurones.

La transmission entre deux neurones n'est pas directe. En fait, il existe un espace intercellulaire entre l'axone du neurone qui émet et les dendrites du neurone qui reçoit. La jonction entre deux neurones est appelée la synapse (fig. 1.1).

Les neurones font une sommation des signaux reçus en entrée et en fonction du résultat obtenu vont fournir un signal en sortie.



*Figure 1.1 : le neurone biologique*

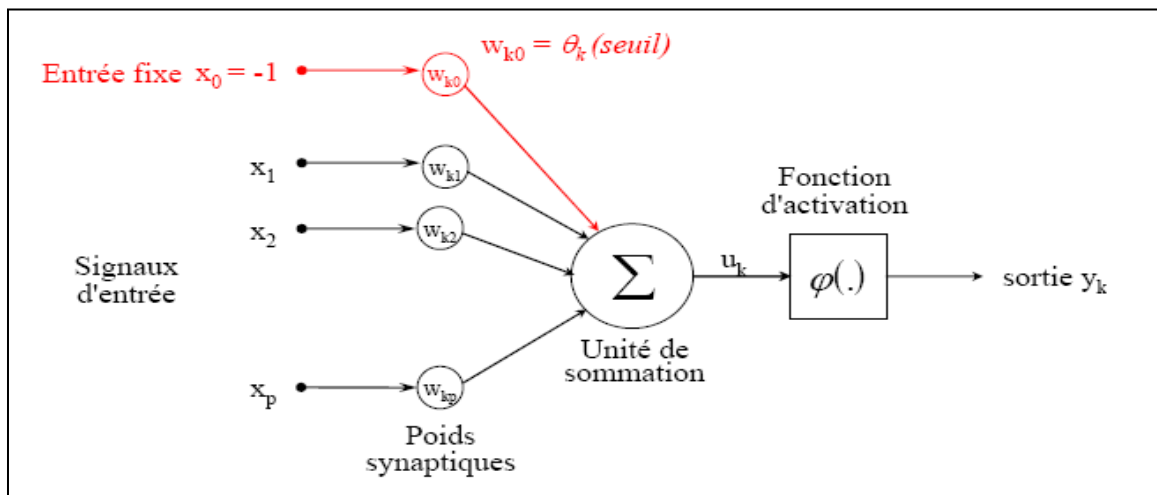
## 2.2 Neurone artificiel

La correspondance entre le neurone biologique et le neurone artificiel est modélisée dans le tableau 1 ci après

Neurone biologique	Neurone artificiel
Synapses	Poids de connexions
Axones	Signal de sortie
Dendrite	Signal d'entrée
Somma	Fonction d'activation

Tableau 1.1

Le fonctionnement d'un neurone artificiel peut être modélisé par le schéma suivant :



Ce modèle est mathématiquement décrit par deux équations :

$$u_k = \sum_{j=1}^p w_{kj} x_j \quad \text{et} \quad y_k = \varphi(u_k - \theta_k)$$

Où :

$w_{k1}, w_{k2}, \dots, w_{kp}$  sont les poids synaptiques du neurone  $k$ ,

$u_k$  est la sortie de l'unité de sommation,

$\theta_k$  est le seuil,

$\varphi(.)$  est la fonction d'activation,

$y_k$  est le signal de sortie du neurone  $k$ .

Sur ce schéma, le neurone a plusieurs connexions en entrée le reliant à autant de neurones (ou entrées). Il reçoit de l'information provenant de chacun de ces neurones (ou entrées).

Les valeurs qu'il reçoit ainsi en entrée par chacune de ses connexions sont respectivement notées  $x_1, x_2, \dots, x_n$  : ce sont les **entrées** du neurone. (-1 est le signal constant réservé au biais).

Toutes les connexions n'ont pas une importance égale pour le neurone. Certaines sont plus importantes que d'autres. Le **poids**  $w$  affecté à chaque connexion mesure cette importance relative : le poids est proportionnel à l'importance de la connexion. La somme pondérée, appelée activation est calculée :

$$a = w_1x_1 + w_2x_2 + \dots + w_nx_n$$

Le neurone effectue ensuite une opération qui dépend de **a**. Cela revient à dire qu'il applique une fonction  $f$  à la valeur **a**. Cette fonction  $f$  est appelée **fonction d'activation**.

Le choix de cette fonction  $f$  se révèle être un élément constitutif important des réseaux de neurones.

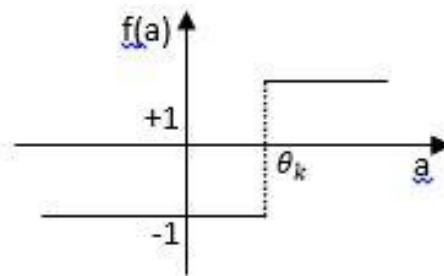
La valeur  $f(a)$  calculée par le neurone constitue la **sortie**  $S$ . Cette valeur en sortie devient ensuite une valeur d'entrée pour tous les neurones avec lesquels notre neurone de référence est connecté.

### 3. Différentes fonctions d'activation

Il existe de nombreuses formes possibles pour la fonction d'activation. Les plus courantes sont celles qui suivent

#### a) Fonction seuil

$$f(a) = \begin{cases} 1 & \text{si } (a - \theta_k) \geq 0 \\ -1 & \text{sinon} \end{cases}$$



Le seuil introduit une non-linéarité dans le comportement du neurone, cependant il limite la gamme des réponses possibles à deux valeurs.

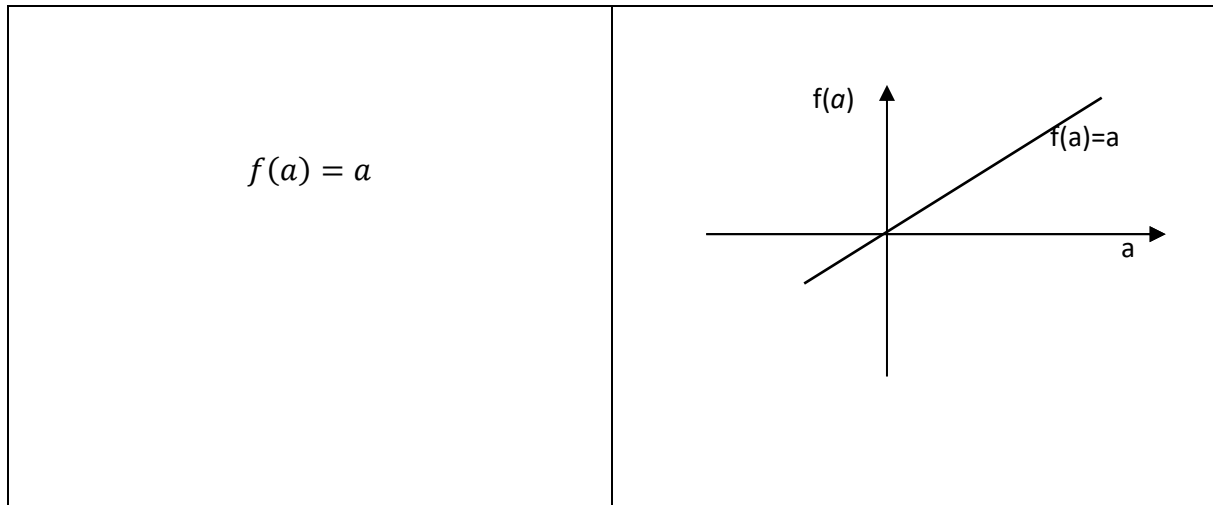
La fonction seuil transforme les signaux d'activation positifs  $a$  en signaux unité et les signaux d'activation négatifs en signaux nuls. La discontinuité se produit à la valeur d'activation nulle (qui est égal au "seuil" de la fonction signal).

$$f(a) = \begin{cases} 1 & \text{si } a \geq 0 \\ 0 & \text{sinon} \end{cases}$$

Les fonctions seuils ont été utilisées dans les premiers développements des réseaux de neurones, tels que le perceptron, cependant du fait qu'elles n'étaient pas différentiables, elles ont représentées un obstacle dans le développement des réseaux de neurones jusqu'à l'adoption des fonctions sigmoïdales et le développement des techniques de la descente du gradient (pour l'adaptation des poids).

## b) Fonction linéaire :

C'est l'une des fonctions d'activations les plus simples, sa fonction est définie par :

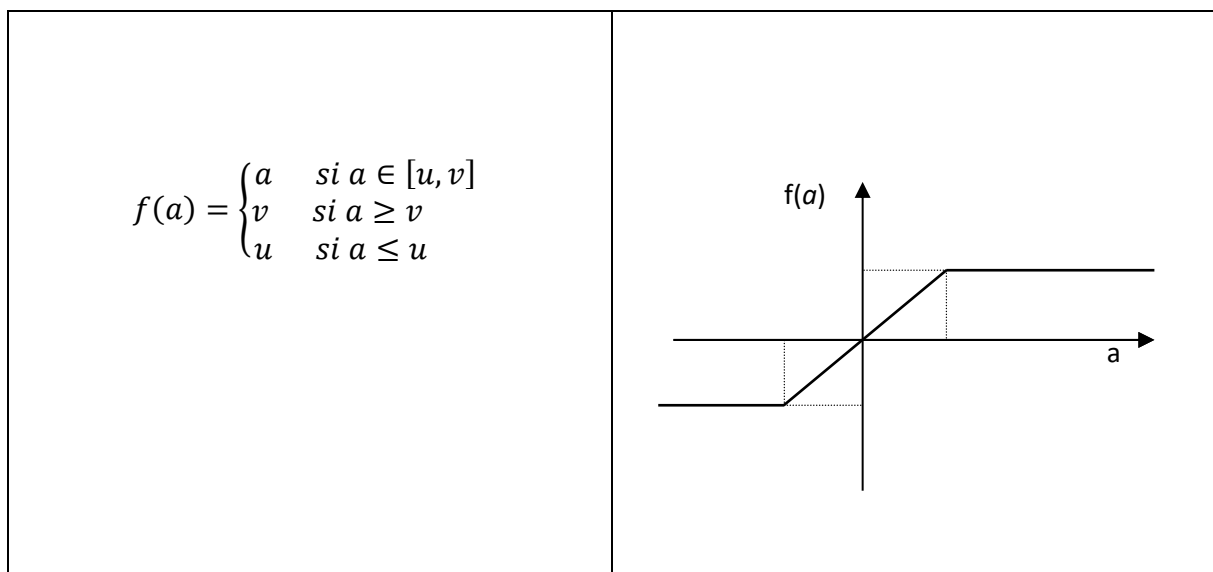


L'entrée passe à la sortie sans aucune modification.

On reste ici dans une situation de proportionnalité.

## c) Fonction linéaire à seuil ou Fonction rampe :

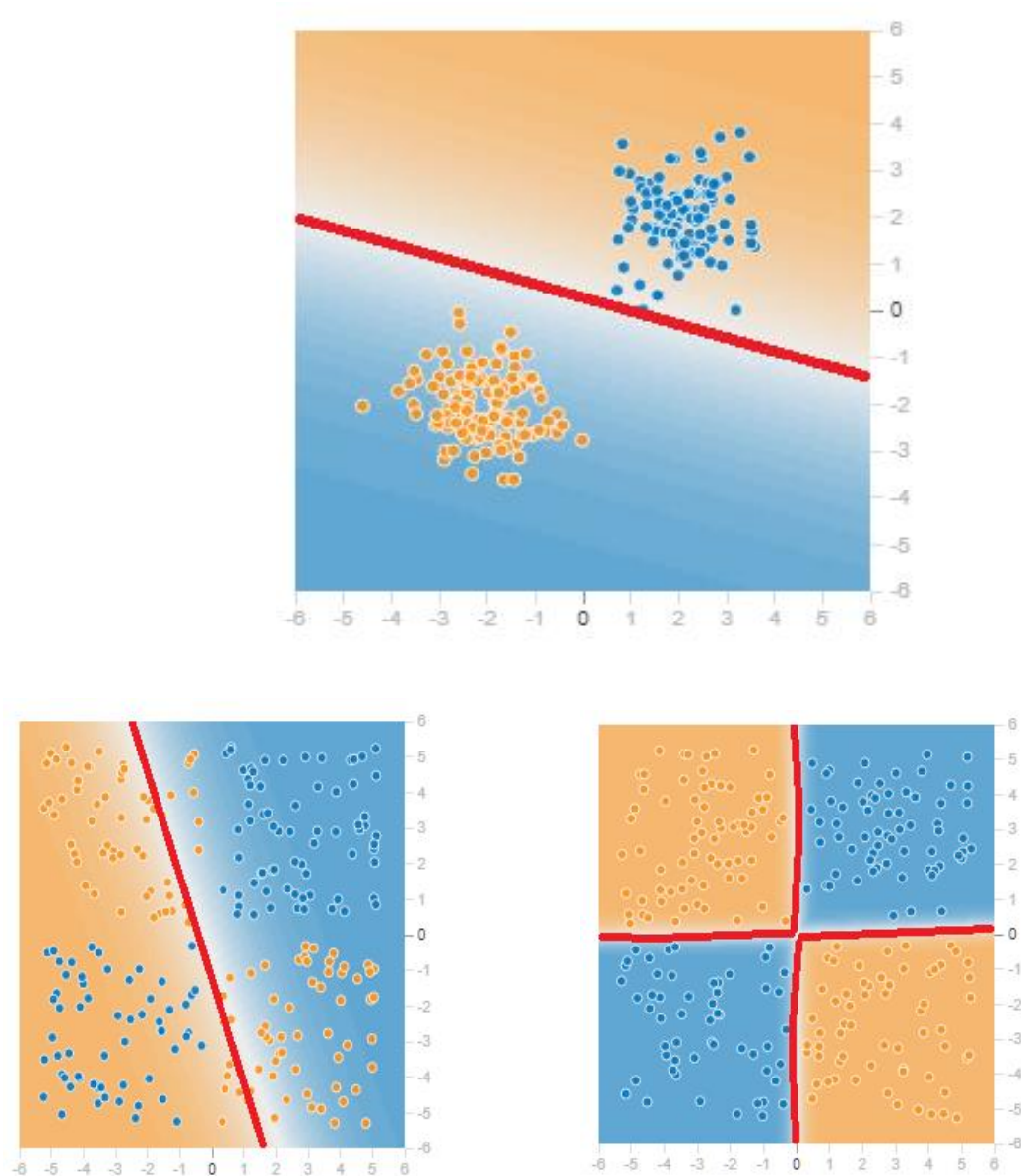
Elle est définie comme suit :



Cette fonction représente un compromis entre la fonction linéaire et la fonction seuil : entre ses deux barres de saturation, elle confère au neurone une gamme de réponses possibles. En modulant la pente de la linéarité, on affecte la plage de réponse du neurone.

**Remarque :**

Le type de neurone qui résulte de tous ces genres cités (à seuil, linéaire, à rampe ou sigmoïde) s'appelle neurone sommateur.



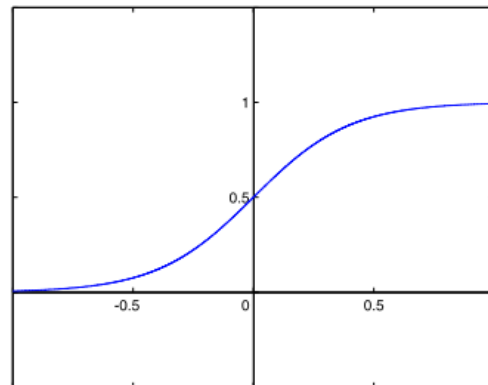
## d) Fonctions non linéaires

Il existe plusieurs types de fonctions non linéaires couramment utilisées comme fonctions d'activation.

### .La fonction sigmoïde :

Elle introduit de la non-linéarité, mais c'est aussi une fonction continue, différentiable. Elle est définie par :

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



La fonction Sigmoïde a ses défauts :

- Elle n'est pas centrée sur zéro, c'est à dire que des entrées négatives peuvent engendrer des sorties positives.
- Etant assez plate, elle influe assez faiblement sur les neurones par rapport à d'autres fonctions d'activations. Le résultat est souvent très proche de 0 ou de 1 causant la saturation de certains neurones.
- Elle est coûteuse en termes de calcul car elle comprend la fonction exponentielle.



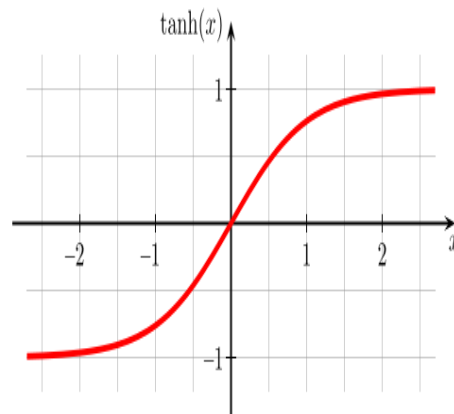
### .La fonction tangente hyperbolique :

Cette fonction ressemble à la fonction Sigmoidé. La différence avec la fonction Sigmoidé est que la fonction Tanh produit un résultat compris entre -1 et 1.

La fonction Tanh est en terme général préférable à la fonction Sigmoidé car elle est centrée sur zéro.

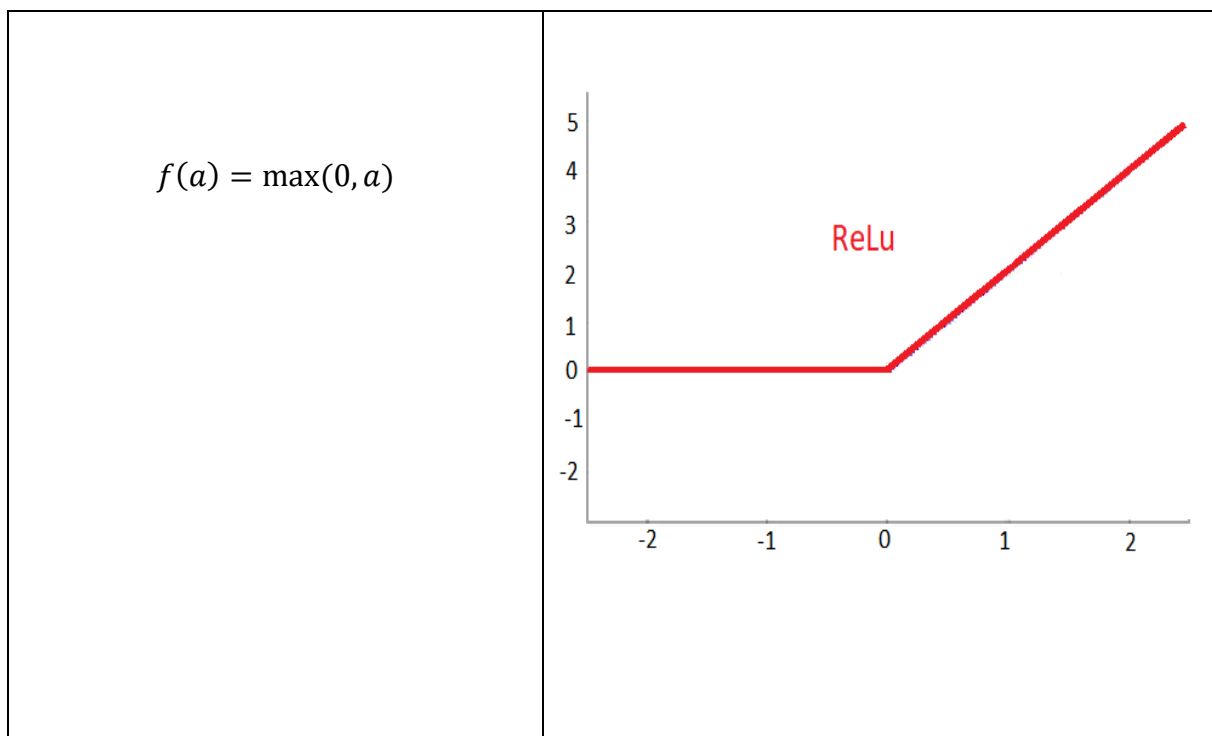
La fonction Tanh possède les mêmes autres inconvénients que la fonction Sigmoidé.

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



### .ReLU

Pour résoudre le problème de saturation des deux fonctions précédentes (Sigmoidé et Tanh) il existe la fonction ReLU (Unité de Rectification Linéaire). Cette fonction est la plus utilisée.



Si l'entrée est négative la sortie est 0 et si elle est positive alors la sortie est a. Cette fonction d'activation augmente considérablement la convergence du réseau et ne sature pas.

Mais la fonction ReLU n'est pas parfaite. Si la valeur d'entrée est négative, le neurone reste inactif, ainsi les poids ne sont pas mis à jour et le réseau n'apprend pas.

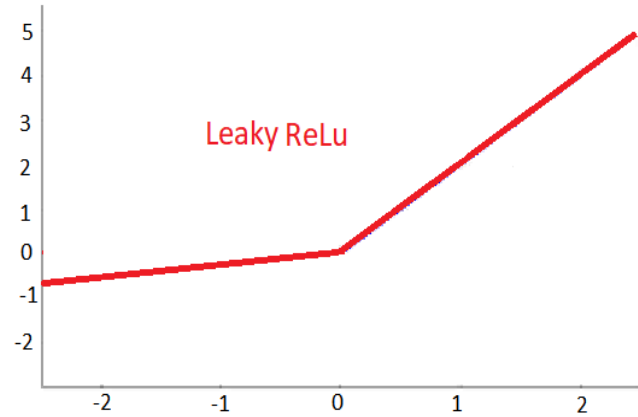
### .La fonction Leaky Relu :

Essaye de corriger la fonction ReLU lorsque l'entrée est négative. Le concept de Leaky ReLU est lorsque l'entrée est négative, il aura une petite pente positive de 0,1.

Cette fonction élimine quelque peu le problème d'inactivité de la fonction ReLU pour les valeurs négatives, mais les résultats obtenus avec elle ne sont pas cohérents.

Elle conserve tout de même les caractéristiques d'une fonction d'activation ReLU, c'est-à-dire efficace sur le plan des calculs, elle converge beaucoup plus rapidement et ne sature pas dans les régions positives.

$$f(a) = \max(0.1a, a)$$



## Conclusion

Diverses fonctions ont été utilisées. Seules les principales ont été citées. A l'heure actuelle il reste encore beaucoup de recherche à effectuer pour trouver une fonction d'activation appropriée permettant au réseau neuronal d'apprendre plus efficacement et plus rapidement.

## 4. Différentes architectures

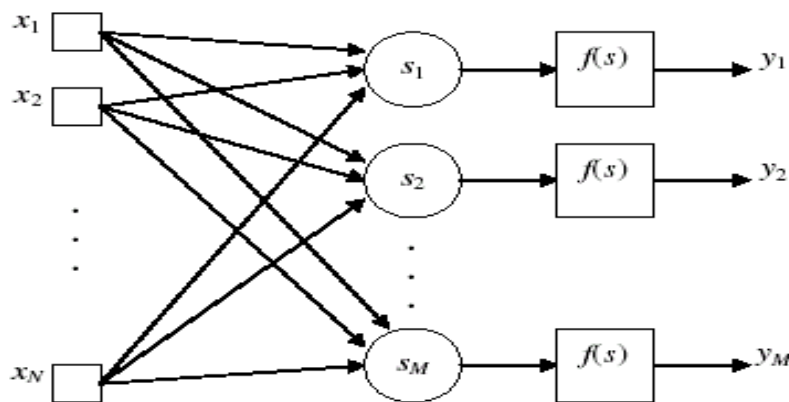
La façon dont les neurones d'un réseau sont structurés est intimement liée avec l'algorithme d'apprentissage employé pour entraîner le réseau.

En général, nous pouvons identifier trois classes fondamentalement différentes d'architectures de réseaux :

### 4.1 Réseau à propagation avant à une seule couche

Dans un réseau de neurones à couches les neurones sont organisés sous forme de couches. Dans la forme la plus simple d'un réseau à couches nous avons une couche d'entrée de nœuds qui se projette sur une couche de sortie (nœuds de calculs), mais pas dans le sens inverse. En d'autres termes, ce réseau est strictement à propagation avant ou de type acyclique.

Un tel réseau est appelé réseau à une seule couche, où la désignation '*une seule couche*' se réfère à la couche de sortie composée des nœuds de calcul (neurones). Nous ne comptons pas la couche d'entrée des nœuds source du fait qu'aucun calcul n'y est fait.



**FIGURE 1.2 :** Réseau à propagation avant (ou réseau acyclique) avec une seule couche de neurones.

### 4.2 Réseau multicouches à propagation avant

La seconde classe de réseau à propagation avant se distingue par la présence d'une ou de plusieurs couches cachées, dont les nœuds de calcul sont appelés neurones cachés ou unités cachées.

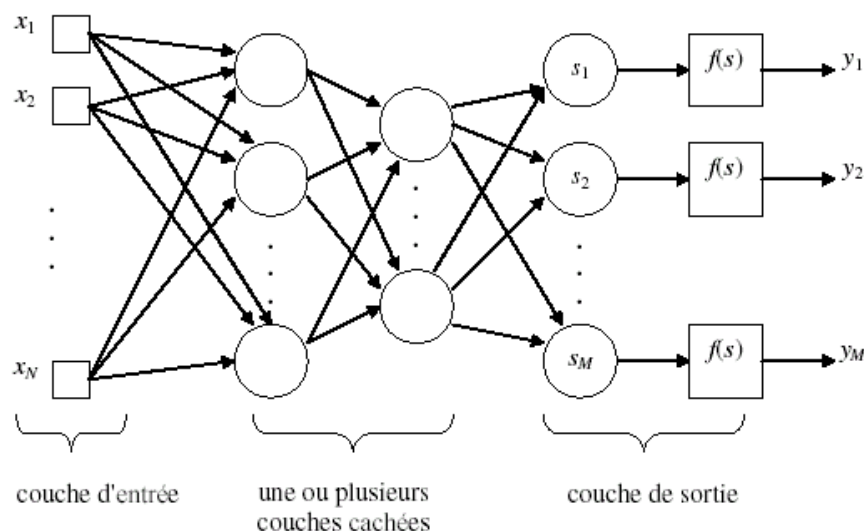
La fonction des neurones cachés est d'intervenir entre l'entrée et la sortie du réseau d'une manière utile. En ajoutant une ou plusieurs couches cachées, le réseau devient capable d'extraire des statistiques d'ordre supérieur.

Les nœuds source dans la couche d'entrée du réseau fournissent les éléments de la forme d'activation (le vecteur d'entrée), qui constituent les signaux d'entrée appliqués aux neurones dans la deuxième couche.

Les signaux de sortie de la deuxième couche sont utilisés comme entrée pour la troisième couche, et ainsi de suite pour le reste du réseau. Typiquement, les neurones de chaque couche du réseau ont pour entrées seulement les sorties de la couche précédente.

L'ensemble de signaux de sortie des neurones de la dernière couche (finale) du réseau constitue la réponse finale du réseau à la forme d'activation fournie par les nœuds source dans la couche d'entrée (première couche).

Le réseau de la figure 1.3 est dit complètement connecté dans le sens où chaque nœud dans chaque couche est connecté à chaque nœud de la couche suivante. Si cependant, quelques liens de communication (connexions synaptiques) manquent au réseau, nous dirons que ce réseau est partiellement connecté.



*Figure 1.3 : réseau à propagation avant (ou acyclique) complètement connecté avec plusieurs couches cachées et une couche de sortie.*

## 4.3 Quelques réseaux multicouches

### a) Les perceptrons multicouches

Le réseau consiste en un ensemble d'unités sensorielles qui constituent la couche d'entrée, une ou plusieurs couches cachées, et une couche de sortie.

Le signal d'entrée se propage vers l'avant, de couche en couche.

Les perceptrons multicouches ont été utilisés pour résoudre avec succès des problèmes difficiles et divers, en les entraînant d'une manière supervisée avec l'algorithme de rétro propagation.

Le PMC possède trois caractéristiques distinctives :

1. Chaque neurone caché est doté d'une fonction d'activation non linéaire.
2. Le réseau contient une ou plusieurs couches cachées, ce qui permet au réseau d'apprendre des tâches complexes en extrayant plus de caractéristiques significatives à partir des données.
3. Le réseau présente de hauts degrés de connectivité.

*Le perceptron multicouche, entraîné avec l'algorithme de rétro propagation, repose sur une forme d'optimisation globale pour sa conception et c'est un approximateur universel.*

### PMC : Formules et propriétés

• Passage C. Entrée C. Cachée

$$v_1 = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$$
$$v_2 = w'_0 + w'_1 x_1 + w'_2 x_2 + w'_3 x_3$$

• Sortie de la C. Cachée

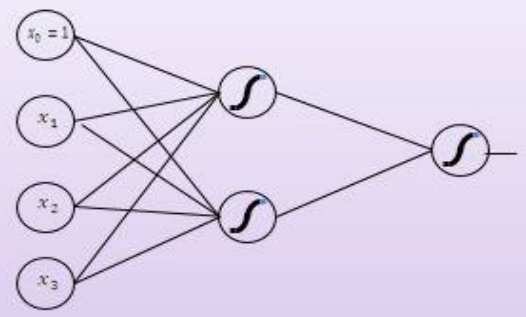
$$u_1 = g(v_1) = \frac{1}{1 + e^{-v_1}}$$
$$u_2 = g(v_2) = \frac{1}{1 + e^{-v_2}}$$

• Passage C. Cachée → C. Sortie →

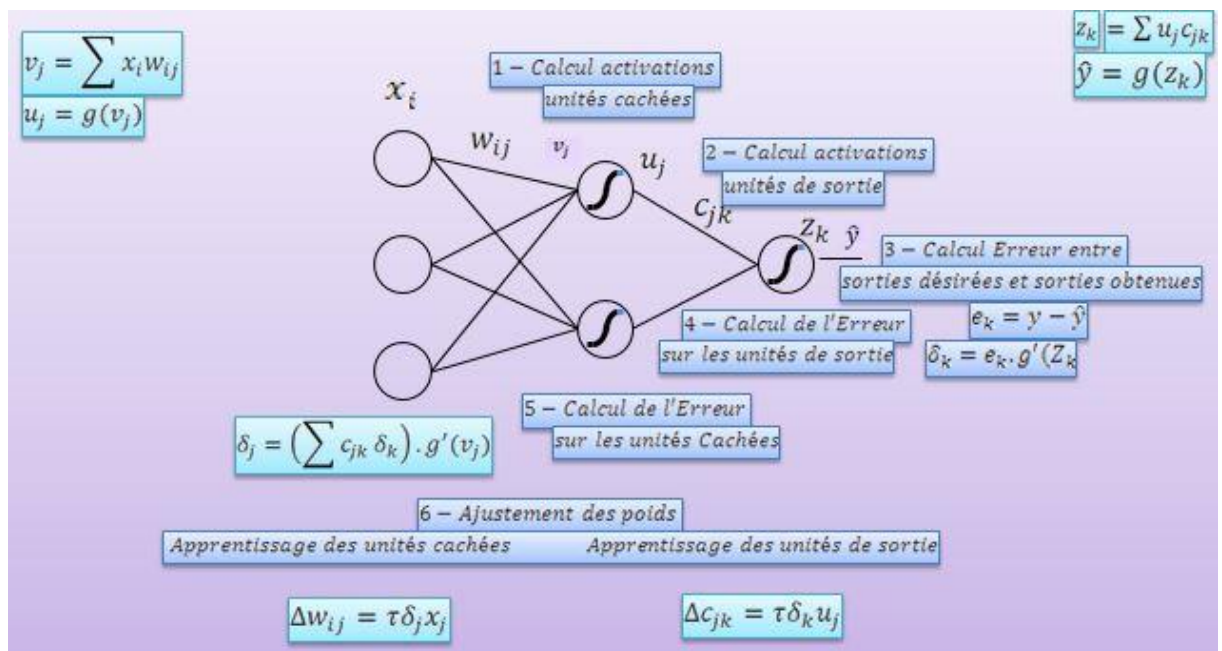
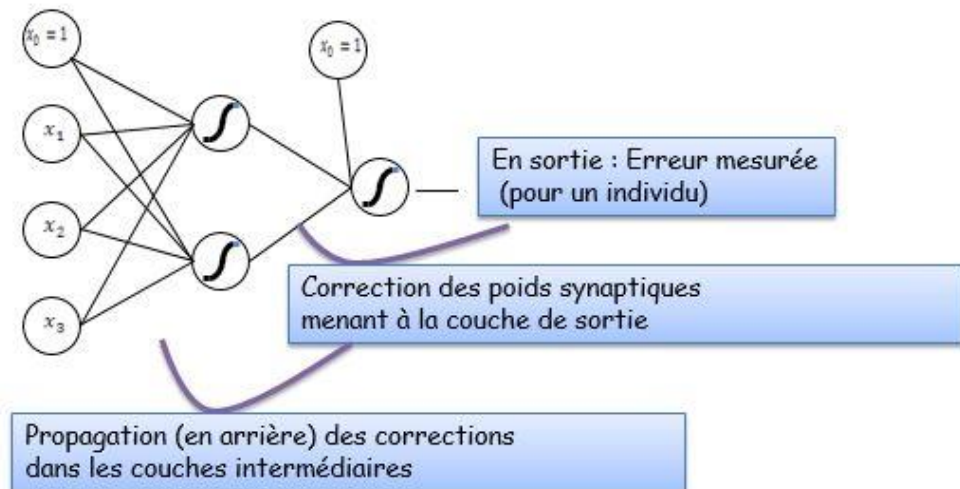
$$z = c_0 + c_1 u_1 + c_2 u_2$$

• Sortie du réseau →

$$\hat{y} = g(z) = \frac{1}{1 + e^{-z}}$$



## PMC : La rétro propagation du gradient



Ajuster les poids

$$w_{new} = w_{old} + \Delta w_{ij}$$

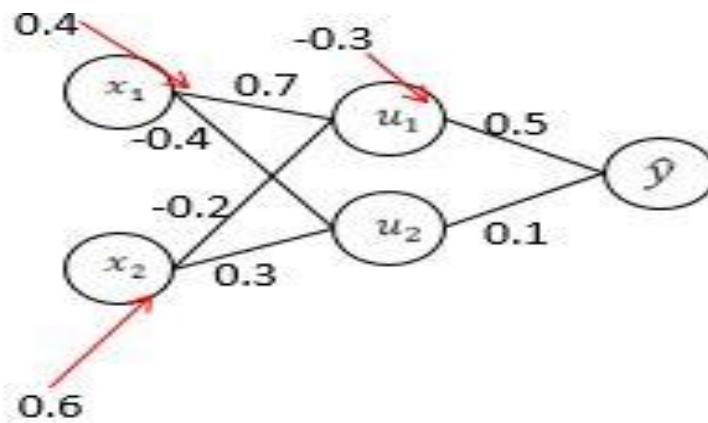
$$c_{new} = c_{old} + \Delta c_{jk}$$

## *Devoir à la maison*

Entraînez avec l'algorithme de rétro-propagation le réseau suivant pour :

Une sortie désirée :  $y=0.3$

Une sigmoïde comme fonction d'activation.



A rendre le Mercredi 14 Novembre 2018



## b) Réseaux à fonctions radiales de base (RBF) :

Un réseau de neurone de type RBF est un PMC spéciale, son architecture est identique à celle d'un PMC à une seule couche cachée

Donc on peut dire qu'il prend toutes les caractéristiques d'un PMC simple sauf qu'il diffère en quelques points nous citons quelques-uns :

- Un réseau RBF ne peut contenir qu'une seule couche cachée, son architecture est fixée pour tous les problèmes à étudier.
- Le réseau RBF utilise toujours une fonction dite à base radiale centrée.

En ce qui concerne les ressemblances entre un réseau RBF et un PMC, on peut mentionner quelques points :

- Généralement une simple fonction linéaire qui renvoie une sommation pondérée des valeurs calculées par les neurones de la couche cachée. Bien sûr, ce n'est pas toujours le cas, parfois l'utilisation d'autres fonctions pourrait être plus adéquate dans un problème donné.
- Les connexions entre les couches suivent le même sens, on peut dire qu'elles ne sont pas récurrentes, et chaque neurone est entièrement connecté vers les neurones de la couche suivante.
- Pour calculer les poids de la couche de sortie, on utilise un apprentissage supervisé pour les deux types de réseaux.

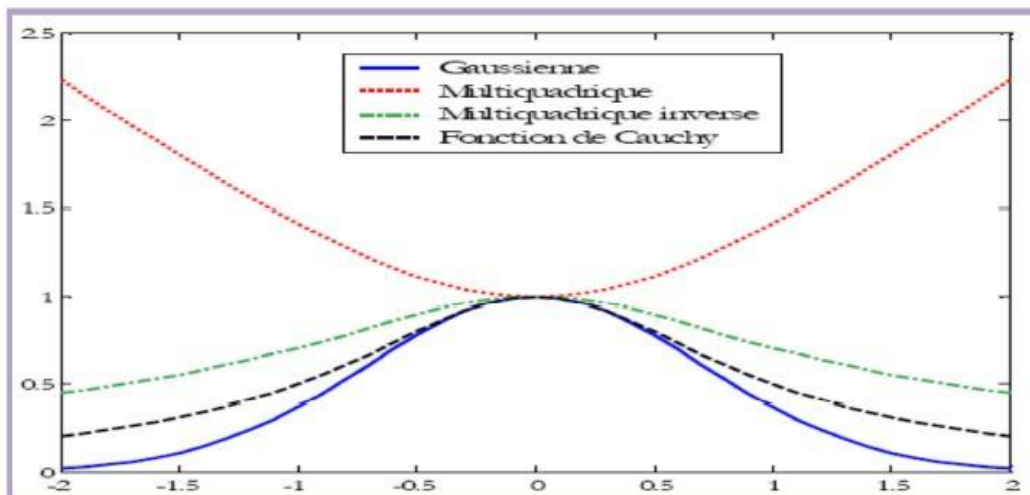


Figure 1.4 : Quelques fonctions radiales

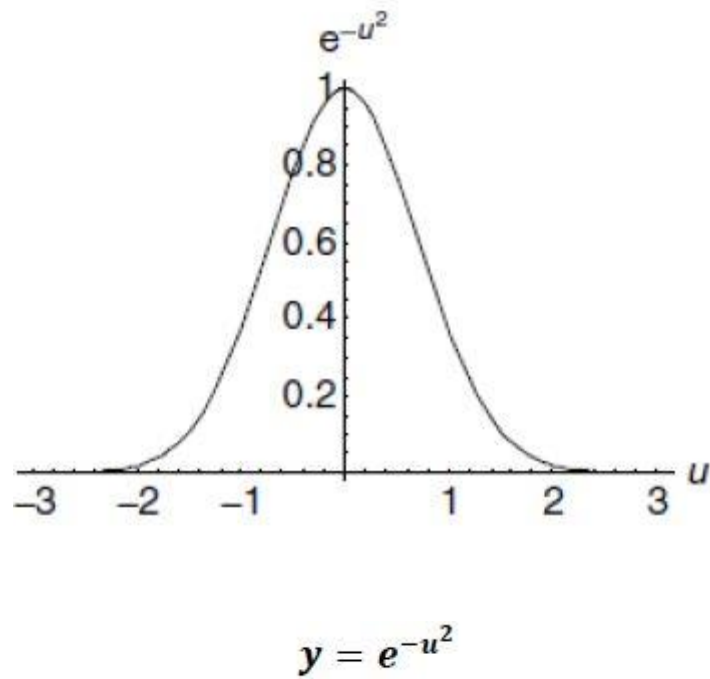


Figure 1.5 : La fonction d'activation gaussienne

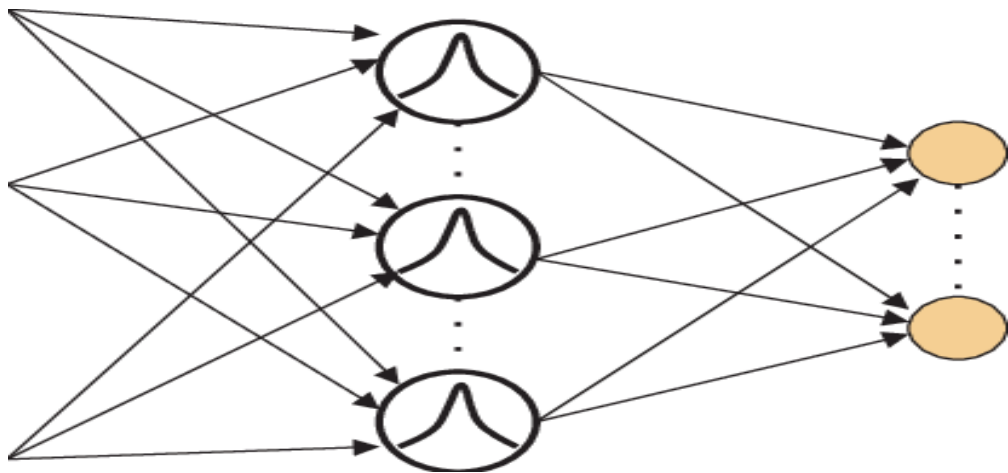


Figure 1.6 : Architecture d'un Réseau de Neurone RBF

Le réseau RBF repose sur une forme d'optimisation locale pour sa conception et c'est un approximateur universel.

### c) Les machines à vecteurs supports (SVM)

Fondamentalement, le SVM est une machine linéaire possédant d'excellentes propriétés.

Elles permettent de trouver des surfaces discriminantes de n'importe quelle forme, avec un algorithme spécifique.

Un des intérêts des SVM est que la fonction de coût que l'on minimise durant l'apprentissage est convexe (présente un seul minimum), alors la fonction de coût des moindres carrés utilisée pour la régression présentent des minima locaux.

## 4.4 Réseaux récurrents

Un réseau récurrent se distingue d'un réseau à propagation avant par le fait qu'il possède au moins une boucle arrière.

De tels réseaux sont plus complexes, mais également beaucoup plus puissants.

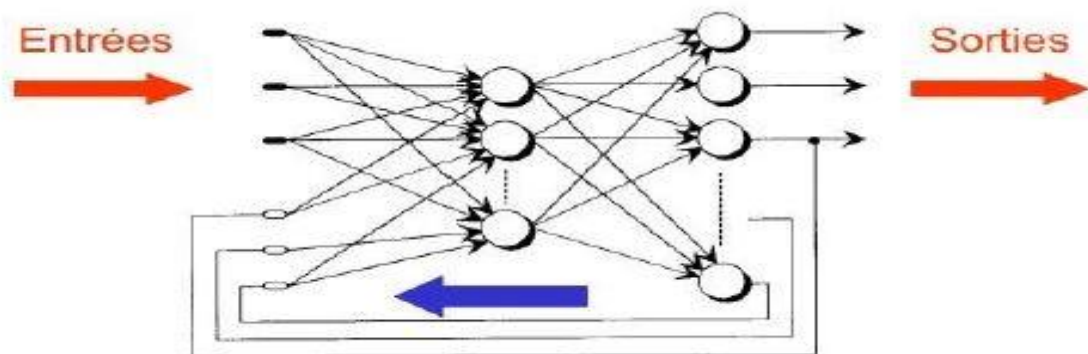


Figure 1.6 : réseau récurrent

## 4.5 Quelques réseaux récurrents

### a) Le réseau de Hopfield

Le modèle de Hopfield utilise l'architecture des réseaux entièrement connectés et récurrents (dont les connexions sont non orientées).

Les sorties sont en fonction des entrées et du dernier état pris par le réseau.

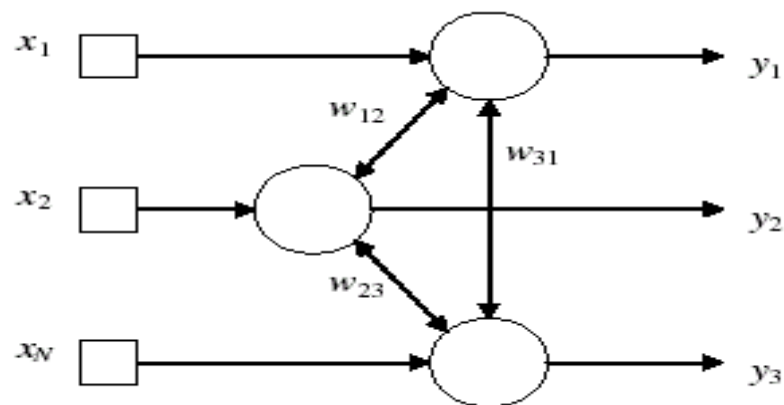


Figure 1.7 : Réseaux associatifs - modèle de Hopfield, réseau récurrent monocouche à connexité totale.

C'est un réseau avec des sorties binaires où tous les neurones sont interconnectés avec des poids symétriques, c'est-à-dire que le poids du neurone  $N_i$  au neurone  $N_j$  est égal au poids du neurone  $N_j$  au neurone  $N_i$ .

#### b) Modèle de Kohonen

- Les cartes de Kohonen sont réalisées à partir d'un réseau à deux couches, une en entrée et une en sortie.
- Notons que les neurones de la couche d'entrée sont entièrement connectés à la couche de sortie
- Chaque neurone de la couche de sortie possède des connexions latérales récurrentes dans sa couche.

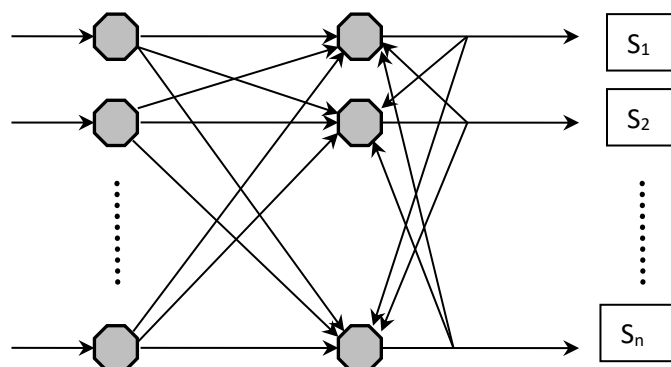


Figure 1.8 : Le modèle de Kohonen

## 5. Historique

McCULLOCH, W., et PITTS, W., 1943, "A logical Calculus of the Ideas Immanent in Nervous Activity", Bulletin of Mathematical Biophysics, 5, 1943, pp. 115-133

- Dans leur article, McCulloch et Pitts ont développé un calcul logique pour les réseaux de neurones qui unifie la neurophysiologie et la logique mathématique. Leur modèle formel suit la loi du 'tout ou rien'. Avec un nombre suffisant de ces unités simples, et des connections établies convenablement et fonctionnant d'une manière synchrone, McCulloch et Pitts ont montré qu'un réseau ainsi constitué peut, en principe, calculer toute fonction calculable.

Hebb, D.O., 1949, *The Organization of Behavior ; A Neuropsychological Theory*, New York ; Wiley

- Le développement majeur qui a suivi a été le livre « *The Organization of Behavior* » publié par Hebb en 1949, dans lequel une hypothèse explicite a été présentée pour la première fois, concernant la règle d'apprentissage physiologique se basant sur les modifications synaptiques. Hebb a proposé comme hypothèse que la connectivité du cerveau change continuellement lorsque l'organisme apprend les différentes tâches fonctionnelles, et des assemblages de neurones sont créés par de tels changements.

Le livre de Hebb a été une source d'inspiration pour le développement de systèmes apprenants et adaptatifs.

- L'article de Rochester, Holland, Haibt et Duda (1956), a peut être été la première tentative d'utilisation de la simulation par machine pour tester la théorie neuronale basée sur le postulat d'apprentissage de Hebb. Ces auteurs ont montré que *l'inhibition* doit être ajoutée pour que la théorie puisse fonctionner correctement.

Rochester, N., J.H. Holland, L.H. Haibt, and W.L. Duda, 1956 ; "Tests on a cell assembly theory of the action of the brain using a large digital computer", IRE Transactions on Information Theory, volIT-2, pp. 80-93

- Dans la même année (1956), Uttley démontre qu'un réseau de neurones à synapses modifiables peut apprendre à classer des ensembles simples d'éléments binaires dans des classes correspondantes.

. Uttley, A.M., 1956, " A theory of the mechanism of learning based on the computation of conditional probabilities", Proceedings of the First International Conference on Cybernetics, Namur, Gauthier-Villars, Paris

- En 1979, Uttley a émis l'hypothèse que l'efficacité d'une synapse variable dans le système nerveux dépend des relations statistiques entre les états fluctuants sur les deux côtés de la synapse. De cette manière, Uttley établit la relation avec la théorie de l'information.

Uttley, A.M., 1979, *Information Transmission in the Nervous System*, London ; Academic Press

- En 1952, apparaît le livre d'Ashby, dans lequel il est stipulé que le comportement adaptatif n'est pas inné mais appris, et c'est à travers l'apprentissage que l'animal (système) évolue vers le meilleur.

Le livre met en exergue d'un côté, les aspects dynamiques des organismes vivants en qualité de machines et de l'autre côté le concept de stabilité.

Ashby, W.R., 1952, *Design for Brain*, New York ; Wiley

- En 1954, Minsky rédige une thèse sur les réseaux de neurones et en 1961 publie un article portant sur l'intelligence artificielle et contenant une longue section portant sur ce qui sera appelé '**réseaux de neurones**'.

Minsky, M.L., 1954, "*Theory of neural-analog reinforcement systems and its application to the brain-model problem*", Ph.D. thesis, Princeton University, Princeton, NJ.

Minsky, M.L., 1961, "*Steps towards artificial intelligence*", *Proceedings of the Institute of Radio Engineers*, vol. 49, pp.8-30 (Reprinted in : Feigenbaum, E.A., and J.Feldman, eds, *Computers and Thought*, pp. 406-450, New York ; McGraw-Hill.)

- En 1958, Rosenblatt introduit une nouvelle approche de la reconnaissance des formes avec son travail sur le *perceptron*, une nouvelle méthode d'apprentissage supervisé. Il démontre l'important théorème de la *convergence du perceptron*.

Rosenblatt, F., 1958, "The Perceptron : A probabilistic model for information storage and organization in the brain", *Psychological Review*, vol. 65. Pp. 386- 408

- En 1960, Widrow et Hoff introduisent l'algorithme des moindres carrés (LMS) et l'utilisent pour formuler l'*Adaline* (élément linéaire adaptatif). La différence entre le perceptron et l'*Adaline* réside dans la procédure d'apprentissage.
- En 1967, Minsky publie un livre qui étend les résultats de McCulloch et Pitts et les situe dans le contexte de la théorie des automates et de la théorie du calcul.

Minsky, M.L., 1967, *Computation ; Finite and Infinite Machines*, Englewood Cliffs, NJ ; Prentice-Hall.