Les inconvénients :

- Facilité d'applications donnant lieu a de nombreuses implémentations et choix pas toujours Justifiés.
- Malgré une solide base théorique, le choix du réseau incombe souvent a l'utilisateur car il n'existe pas de guide prouvé pour toute utilisation.
- La nature nonlineaire des RNAs peut piéger l'utilisateur dans un minimum local.

1.7 Applications des RNA

Les RNAs ont trouvé applications dans une multitude de domaines, aussi divers que différents. Un aperçu des domaines ou les RNA ont étés d'un grand apport sont :

1. Signal processing
   - Egalisation non linéaire
   - L'élimination de bruit et de l'écho.
   - Reconnaissance de signaux radar ou sonar
2. Automatique
   - Identification/modélisation des systèmes non linéaire.
   - Commande basé modèle (commande prédictive, IMC, etc).
3. Compression de données
4. Systèmes de classifications ou de diagnostiques
5. Mémoires adressable

2. Neurone simple et problème de classification

Dans ce chapitre, nous allons, tout d'abord, présenter les types de neurones simples ainsi que quelques règles d'apprentissage de base, soutenus par des exemples. Une fois cette introduction effectuée, un aperçu sur la classification en utilisant les neurones présentés sera introduite avec quelques exemples de modélisation des fonctions logiques AND et OR.

2.1 Type de neurones simples

2.1.1 Le neurone de Mc Culloch-Pitt

Le neurone de Mc Culloch-Pitt est donné dans la Figure 2.1, il se caractérise principalement par une fonction d'activation de type limitation 0 ou 1 (s=1 ou 0).

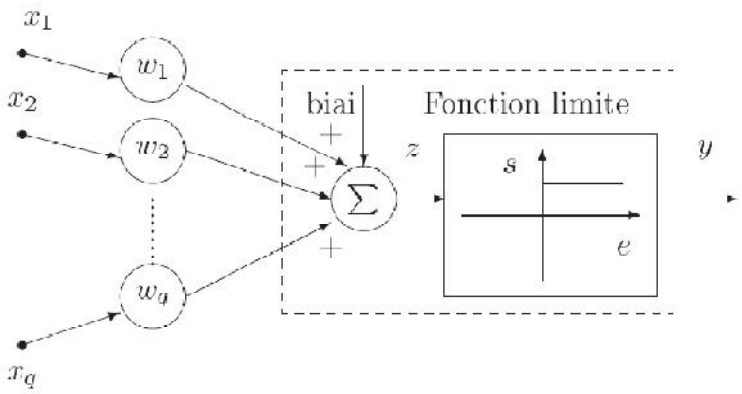

Figure 2.1 – Neurone de Mc Culloch-Pitt

2.1.2 Le Perceptron

Le perceptron, Figure 2.2, se caractérise par :

- Une fonction d'activation continue de type : $y = \frac{2}{1+e^{-z}} - 1$.
- Fonction d'activation de type squashing, ou $-1 < y < 1$.
- La fonction d'activation est différentiable.

2.2 La règle d'apprentissage de Hebb

Hebb (1949) :

Quand l'axone de la cellule A est proche d'exciter la cellule B ceci se traduit par une décharge (fire), un changement métabolique ou évolution se produit dans l'une ou l'autre des cellules. En d'autres termes, l'e_cacité de déchargement de A sur B est augmentée.

Algorithme :

$$y = f(w^T x) \quad (2.1)$$
$$\Delta w = \mu y x = \mu f(w^T x) x \quad (2.2)$$

Et, pour le neurone i et l'entrée j, on a :

$$\Delta w_{ij} = \mu y x = \mu f(w_i^T x_j) x_j \quad (2.3)$$

Avec :

- w_0 est initialisé avec des valeurs aléatoires
- L'apprentissage est purement a propagation avant (feedforward) et non supervisé.
- Les poids w_i, seront fortement influencés par des entrées fréquentes
- Une évolution sans contraintes des poids w_i.
- Une règle de mise a jour basé sur la corrélation.

2.2.1 La règle de Hebb appliqué a un neurone de type signe

Soit le neurone de type signe suivant. Il est à noter que ceci est une version du perceptron avec $\beta \to \infty => f(w^T x) = sgn(w^T x)$.

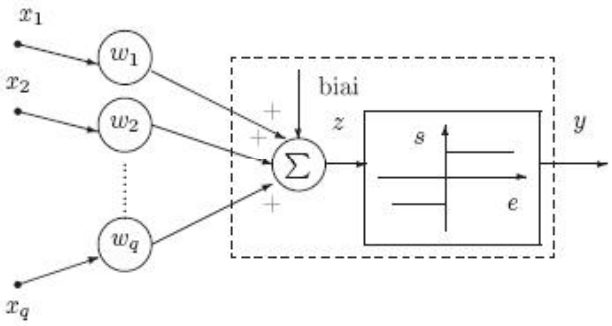

Figure 2.3 – Neurone de type signe

Soit : $w_0 = \begin{bmatrix} 1.0 \\ -1.0 \\ 0.0 \\ 0.5 \end{bmatrix}$ et $x_0 = \begin{bmatrix} 1.0 \\ -2.0 \\ 1.5 \\ 0.0 \end{bmatrix}$, $x_1 = \begin{bmatrix} 1.0 \\ -0.5 \\ -2.0 \\ -1.5 \end{bmatrix}$, $x_2 = \begin{bmatrix} 0.0 \\ 1.0 \\ -1.0 \\ 1.5 \end{bmatrix}$

Si on prend $\mu = 1$, pour des raisons simplificatives, on obtient :

$$z_0 = w_0^T x_0 = [1.0 \ \ -1.0 \ \ 0.0 \ \ 0.5] \begin{bmatrix} 1.0 \\ -2.0 \\ 1.5 \\ 0.0 \end{bmatrix} = 3$$

La mise a jour des poids ($w_{k+1} = w_k + \Delta w$) se fait comme suit :

$$w_{k+1} = w_k + sgn(z_k)x_k \tag{2.4}$$

D'ou :

$$w_1 = \begin{bmatrix} 1.0 \\ -1.0 \\ 0.0 \\ 0.5 \end{bmatrix} + 1. \begin{bmatrix} 1.0 \\ -2.0 \\ 1.5 \\ 0.0 \end{bmatrix} = \begin{bmatrix} 2 \\ -3.0 \\ 1.5 \\ 0.5 \end{bmatrix}$$

Pour les deux prochaine itérations :

$$z_1 = -0.25,\ w_2 = w_1 + sgn(z_1)x_1 = w_1 - x_1 = \begin{bmatrix} 1.0 \\ -2.5 \\ 3.5 \\ 2.0 \end{bmatrix}$$

$$z_2 = -3.0,\ w_3 = w_2 + sgn(z_2)x_2 = w_2 - x_2 = \begin{bmatrix} 1.0 \\ -3.5 \\ 4.5 \\ 0.5 \end{bmatrix}$$

La règle de Hebb utilisé avec un neurone signe, revient à rajouter ou soustraire l'entrée et le vecteur poids. Si bien sur $\mu = 1$.

2.2.2 Règle de Hebb et neurone de type Perceptron

Soit le neurone de type perceptron Figure 2.2, avec $\beta = 1, => \ f(w^T x) = y = 2/(\ 1 + e^z\ )\ - 1$.

Soit : $w_0 = \begin{bmatrix} 1.0 \\ -1.0 \\ 0.0 \\ 0.5 \end{bmatrix}$ et $x_0 = \begin{bmatrix} 1.0 \\ -2.0 \\ 1.5 \\ 0.0 \end{bmatrix}$, $x_1 = \begin{bmatrix} 1.0 \\ -0.5 \\ -2.0 \\ -1.5 \end{bmatrix}$, $x_2 = \begin{bmatrix} 0.0 \\ 1.0 \\ -1.5 \\ 1.5 \end{bmatrix}$

Si on prend $\mu = 1$, pour des raisons simplificatives, on obtient :

$$z_0 = w_0^T x_0 = [1.0 \ \ -1.0 \ 0.0 \ 0.5] \begin{bmatrix} 1.0 \\ -2.0 \\ 1.5 \\ 0.0 \end{bmatrix} = 3$$

La mise a jour des poids se fait comme suit :

$$w_{k+1} = w_k + f(z_k)x_k, \quad y = f(z) = y = \frac{2}{1 + e^{-\beta z}} - 1 \qquad (2.5)$$

D'ou :

$$w_1 = \begin{bmatrix} 1.0 \\ -1.0 \\ 0.0 \\ 0.5 \end{bmatrix} + 0.905 \begin{bmatrix} 1.0 \\ -2.0 \\ 1.5 \\ 0.0 \end{bmatrix} = \begin{bmatrix} 1.905 \\ -2.810 \\ 1.357 \\ 0.500 \end{bmatrix}$$

Pour les deux prochaines itérations :

$$y_1 = \ \ 0.077, \ w_2 = w_1 \ + \ 0.077x_1 = \begin{bmatrix} 1.828 \\ -2.772 \\ 1.512 \\ 0.616 \end{bmatrix}$$

$$y_2 = -0.932, \ w_3 = w_2 + 0.932x_2 = \begin{bmatrix} 1.828 \\ -3.700 \\ 2.440 \\ 0.783 \end{bmatrix}$$

La règle de Hebb utilisé avec un neurone continue de type Perceptron, revient à rajouter ou soustraire, une fraction (déterminé par β) l'entrée et le vecteur poids.

2.3 Règle d'apprentissage du Perceptron

La regle d'apprentissage du perceptron (Rosenblatt 1958) est caractérisé par :

- La règle est applicable aux neurones de type signe.
- Le signal d'apprentissage est la différence entre l'actuelle et la réponse désirée.
- La nécessité d'un signal désiré implique un apprentissage supervisé.

Soit le perceptron, avec le signal d'apprentissage d :

Algorithme :

$$e = d - y, \quad y = sgn(w^T x) \tag{2.6}$$

$$\Delta w = \mu e x = \mu[d - sgn(w^T x)]x \tag{2.7}$$

Etant donné que d=1 ou -1,

$$\Delta w = \pm 2\mu x \tag{2.8}$$

Exemple :

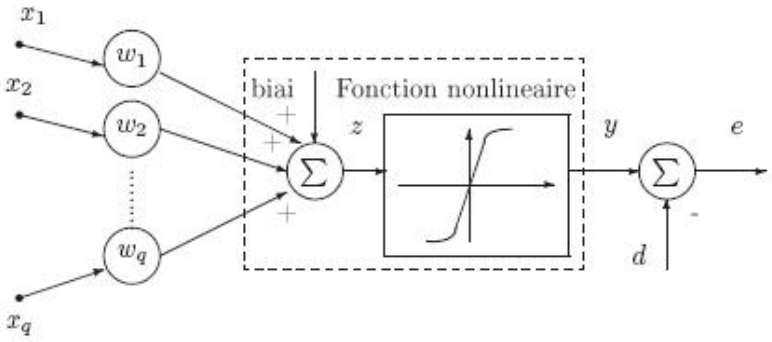

Figure 2.4 – Neurone utilisé avec la régle du Perceptron

Soit : $w_0 = \begin{bmatrix} 1.0 \\ -1.0 \\ 0.0 \\ 0.5 \end{bmatrix}$, $d_0 = -1.0$, $d_1 = -1$, $d_2 = 1$,

et $x_0 = \begin{bmatrix} 1.0 \\ -2.0 \\ 0.0 \\ -1.0 \end{bmatrix}$, $x_1 = \begin{bmatrix} 0.0 \\ 1.5 \\ -0.0 \\ -1.0 \end{bmatrix}$, $x_2 = \begin{bmatrix} -1.0 \\ 1.0 \\ 0.5 \\ -1.0 \end{bmatrix}$

$$z_0 = w_0^T x_0 = [1.0 \ \ -1.0 \ \ 0.0 \ \ 0.5] \begin{bmatrix} 1.0 \\ -2.0 \\ 0.0 \\ -1.0 \end{bmatrix} = 2.5$$

A noter que, $sgn(2.5) \neq d_0$. Si on prend $\mu = 0.1$, on obtient :

$$w_1 = w_0 + 0.1(-1 - 1)x_0$$

$$w_1 = \begin{bmatrix} 1.0 \\ -1.0 \\ 0.0 \\ 0.5 \end{bmatrix} - 0.2 \begin{bmatrix} 1.0 \\ -2.0 \\ 1.5 \\ -1.0 \end{bmatrix} = \begin{bmatrix} 0.8 \\ -0.6 \\ 0.0 \\ 0.7 \end{bmatrix}$$

$$z_1 = w_1^T x_1 = [0.0 \ 1.5 \ -0.5 \ -1.0] \begin{bmatrix} 0.8 \\ -0.6 \\ 0.0 \\ 0.7 \end{bmatrix} = -1.6$$

A noter que, $sgn(-1.6) = d_1$. La mise a jour des poids n'est pas necessaire.

$$z_2 = w_2^T x_2 = [1.0 \ -1.0 \ 0.5 \ -1.0] \begin{bmatrix} -1.0 \\ 1.0 \\ 0.5 \\ -1.0 \end{bmatrix} = -0.75$$

A noter que, $sgn(-0.75) = d_1$. La mise a jour des poids est necessaire.

$$w_3 = w_2 + 0.1(1+1)x_2$$

$$w_3 = \begin{bmatrix} 0.8 \\ -0.6 \\ 0.0 \\ 0.7 \end{bmatrix} + 0.2 \begin{bmatrix} 1.0 \\ -1.0 \\ 0.5 \\ -1.0 \end{bmatrix} = \begin{bmatrix} 0.6 \\ -0.4 \\ 0.1 \\ 0.5 \end{bmatrix}$$

Remarques :

Il n'ya pas de justification évidente pour l'inclusion du vecteur d'entré, *x*, dans l'équation (2.7).
Le choix du coefficient d'apprentissage ¹, n'est pas spécifié.
La règle du perceptron peut être résumé à ce qui suit :

Avec un neurone de Mc Culloch-Pitt
$w_{k+1} = wk + \mu x$ si $y = 0$ et $d = 1$
$w_{k+1} = w_k - \mu x$ si $y = 1$ et $d = 0$
$w_{k+1} = w_k$ si $y = d$

Avec un neurone de type signe (perceptron avec $\beta \rightarrow 1$)
$w_{k+1} = wk + 2\mu x$ si $y = 0$ et $d = 1$
$w_{k+1} = w_k - 2\mu x$ si $y = 1$ et $d = 0$
$w_{k+1} = w_k$ si $y = d$

Il peut être prouvé que le vecteur poids, *w*, converge.

2.3.1 Convergence de la règle du perceptron

Théoreme : S'il existe un vecteur de poids, qui classifie correctement l'ensemble d'apprentissage (supposé linéairement séparable), alors la règle d'apprentissage trouvera un, et un seul, vecteur de poids, *w¤*, dans un nombre fini d'itérations.

Assomptions :

- Il existe au moins un vecteur de poids *w¤*.
- Il existe un nombre _ni de vecteurs d'apprentissage.
- La fonction d'activation est de type unipolaire (0 ou 1).

Preuve :

A la *k* ieme itération, on a :

$$w_{k+1} = w_k + \mu e_k x_k, \quad e_k = d_k - y_k \tag{2.9}$$

Si on soustrait w^* des deux cotés de l'équation (2.9), on obtient :

$$w_{k+1} - w^* = w_k - w^* + \mu e_k x_k \tag{2.10}$$

Si y_k est correctement classé, alors il n'a pas de mise a jour. Si on prend la distance euclidienne de l'equation (2.10), on obtient :

$$\|w_{k+1} - w^*\|^2 - \|w_k - w^*\|^2 + \mu^2\|x_k\|^2 + 2\mu e_k (w_k - w^*)^T x_k \tag{2.11}$$

Il peut etre prouvé que pour cur chaque vecteur, x_k, non classé, on a :

$$e_k w^T x_k = \quad |w^T x_k| \leq 0 \tag{2.12}$$

et aussi :

$$e_k w^{*T} x_k = |w^{*T} x_k| \geq 0 \tag{2.13}$$

D'ou :

$$\|w_{k+1} - w^*\|^2 = \|w_k - w^*\|^2 + \mu^2\|x_k\|^2 - 2\mu\left(|w^{*T} x_k| + |w^T x_k|\right) \tag{2.14}$$

Si on choisi un pas d'apprentissage sullisament petit,

$$\mu^2 \ll \mu \tag{2.15}$$

L'équation (2.14), peut être ramené a :

$$\|w_{k+1} - w^*\|^2 \approx \|w_k - w^*|^2 - 2\mu\left(|w^{*T} x_k| + w^T x_k|\right) \tag{2.16}$$

Sachant que $\mu \geq 0$, et que $|w^{*T} x_k| + |w^T x_k| \geq 0$, doit décroître au fil des itérations. Toutefois, $\|.\|$ne peut pas prendre de valeurs négatives, elle doit donc converger dans un nombre fini d'itérations. A noter que cette norme ne dois pas nécessairement converger vers 0 ($w = w¤$).

Le pas d'apprentissage idéal :

Si on réécrit l'équation (2.14) en terme d'erreur, $\varepsilon = w\text{-}w¤$ on obtient :

$$\|\varepsilon_{k+1}\|^2 = \|\varepsilon_k\|^2 + \mu^2\|x_k\|^2 - 2\mu\left(|w^{*T} x_k| + |w^T x_k|\right) \tag{2.17}$$

Si on minimise, ε_k par rapport a μ, on obtient :

$$\frac{\partial}{\partial\mu}\|\varepsilon_{k+1}\|^2 = \frac{\partial}{\partial\mu}\left(\|\varepsilon_k\|^2 + \mu^2\|x_k\|^2 - 2\mu\left(|w^{*T} x_k| + |w^T x_k|\right)\right) \tag{2.18}$$

$$0 = 2\mu\|x_k\|^2 - 2\left(|w^{*T} x_k| + |w^T x_k|\right) \tag{2.19}$$

$$\mu_{opt} = \frac{|w^{*T} x_k| + |w^T x_k|}{\|x_k\|^2} = \frac{|(w^* + w_k)^T x_k|}{\|x_k\|^2} \tag{2.20}$$

En pratique, il n'est pas possible d'évaluer μ_{opt}, car cette dernière contient $w¤$ dans sa formulation. Et il est souvent suffisant d'interpréter μ_{opt} comme la distance optimale qui puisse ramener w_k au long de la direction de x_k.

2.4 Classification et séparabilité linéaire

2.4.1 Classification :

Le but de la classification (pattern classification) est d'assigner un objet physique, évènement ou phénomène a une classe (ou catégorie) préalablement défini. On peut citer comme exemples :

- Fonction booléennes.
- Pattern de pixels, e.x. afficheur 7 segments.
- Quantification de vecteurs, e.x. Transformation Analog/digital.
- Mémoire associative e.x. reconnaissance de caractères.

2.4.2 Fonctions discriminantes :

Problème :

Supposons qu'il existe un vecteur de forme (patterns) a p dimensions : $x_1, x_2.., x_p$ et que la classification de chaque forme (élément) est connue, il existe n classes. La dimension du vecteur des éléments p est généralement plus importante que le nombre de classes n. Il est aussi raisonnable d'assumer que n est plus important que le nombre de catégories R.
L'appartenance a une catégorie donnée, est déterminé par le classifieur basée sur la comparaison de R fonctions discriminantes $g_1(x)$; $g_2(x)$; … ; $g_R(x)$ calculé pour le vecteur d'entré x. L'élément x appartient a la catégorie i si et seulement si :

$$g_i(x) > g_j(x) \quad for \quad i,j = 1,2,\cdots,R, \quad i \neq j \tag{2.21}$$

L'équation definissant la frontière de la décision est :

$$g_i(x) \quad g_j(x) = 0 \tag{2.22}$$

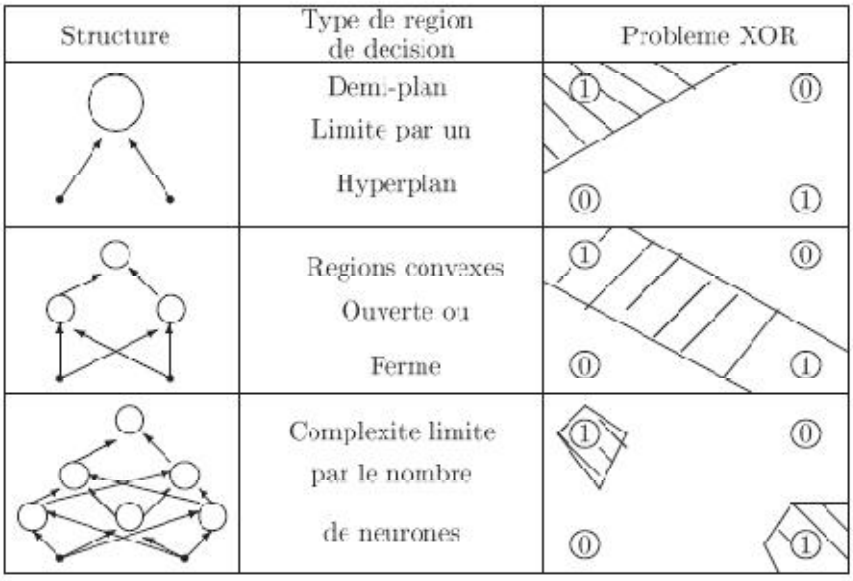

Figure 2.5 – Classification de la fonction XOR par RNA