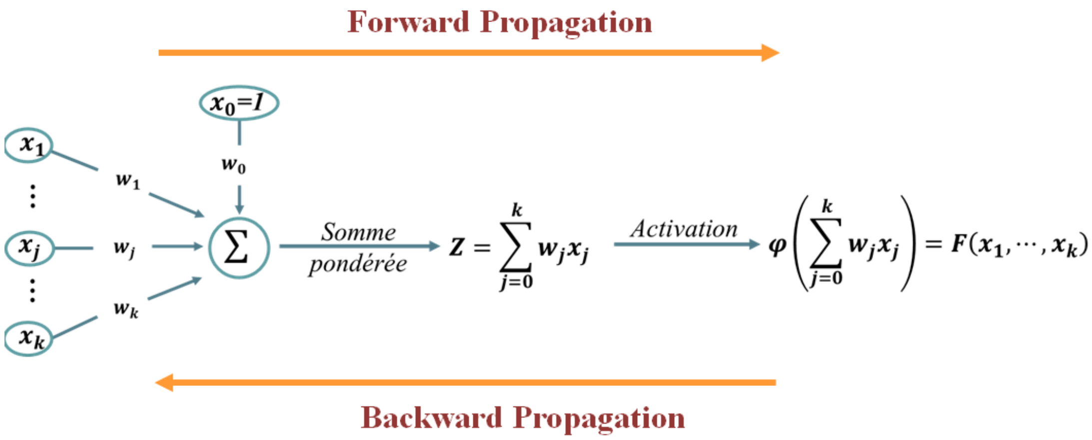
Université Mohammed Premier
Faculté des Sciences
Département d'Informatique
Oujda

Année universitaire : **2023- 2024**
Master : **Ingénierie Informatique**
Module : **Ingénierie Linguistique**
Série 1 : **Réseaux de neurones**

Nous examinons dans ce TP la méthodologie de construction pas à pas d'un perceptron comme illustré dans la figure ci-dessous.



On dispose d'un ensemble d'apprentissage de données $(x^i, e_i)_{i=1,\cdots,N}$ où $x^i = (x_j^i)_{j=1,\cdots,k} \in \mathbb{R}^k$ est la $i^{ème}$ entrée et $e_i \in \mathbb{R}$ est l'étiquette de cette entrée.

La fonction d'activation φ est la fonction ***sigmoïde*** définie par : $\varphi(x) = \frac{1}{1+e^{-x}}$.

La fonction erreur globale est définie pour le vecteur de poids $w = (w_0, \cdots, w_k)$ par

$$Err(w) = \frac{1}{N}\sum_{i=1}^{N} Err_i(w) = \frac{1}{N}\sum_{i=1}^{N}\left(F(x^i) - e_i\right)^2 = \frac{1}{N}\sum_{i=1}^{N}\left[\varphi\left(\sum_{j=0}^{k} w_j x_j^i\right) - e_i\right]^2$$

L'ajustement des poids du modèle se fera selon la formule de descente de gradient suivante :

$$w^{m+1} = w^m - \delta \,\mathrm{grad}\, \mathrm{E}rr(w^m)$$

où le vecteur des poids initiaux $w^0 = (w_0^0, w_1^0, \cdots, w_k^0)$ est choisi aléatoirement, et le pas δ est choisi entre 0 et 1.

Pour calculer grad $\mathrm{E}rr(w)$ défini par :

$$grad\left(Err(w)\right) = \frac{1}{N}\sum_{i=1}^{N} grad\left(Err_i(w)\right) \qquad \text{avec} \qquad grad\left(Err_i(w)\right) = \left(\frac{\partial Err_i(w)}{\partial w_l}\right)_{0\le l\le k}$$

on pose pour une donnée (x, e), où $x = (x_j)_{1\le j\le k}$ et e est l'étiquette de x :

$$z(w) = \sum_{j=0}^{k} w_j x_j \quad (x_0 = 1) \qquad \text{et} \quad s(z) = \varphi(z) = \frac{1}{1+e^{-z}}$$

Ainsi, l'erreur relative à cette donnée est exprimée par :

$$Err(w) = \left[\varphi\left(\sum_{j=0}^{k} w_j x_j\right) - e\right]^2 = [s(z) - e]^2$$

Comme

$$\frac{\partial Err}{\partial s} = 2(s-e), \qquad \frac{\partial s}{\partial z} = \frac{e^{-z}}{(1+e^{-z})^2} = s(1-s), \qquad \frac{\partial z}{\partial \mathrm{w}_l} = x_l$$

alors

$$\begin{aligned}\frac{\partial Err(w)}{\partial \mathrm{w}_l} &= \frac{\partial E(w)}{\partial s} \times \frac{\partial S(w)}{\partial z} \times \frac{\partial z(w)}{\partial \mathrm{w}_l} \\ &= 2(s-e)s(1-s)x_l\end{aligned}$$

et par suite,

$$\begin{aligned} grad\big(Err_i(w)\big) &= \left(\frac{\partial Err_i(w)}{\partial \mathrm{w}_l}\right)_{0 \le l \le k} \\ &= \left(2\left(\varphi\left(\sum_{j=0}^{k} w_j x_j^i\right) - e\right)\varphi\left(\sum_{j=0}^{k} w_j x_j^i\right)\left(1 - \varphi\left(\sum_{j=0}^{k} w_j x_j^i\right)\right)x_l^i\right)_{0 \le l \le k} \\ &= 2\left(\varphi\left(\sum_{j=0}^{k} w_j x_j^i\right) - e_i\right)\varphi\left(\sum_{j=0}^{k} w_j x_j^i\right)\left(1 - \varphi\left(\sum_{j=0}^{k} w_j x_j^i\right)\right)x^i \end{aligned}$$

1. Créer les fonctions ***sigmoïde(x)*** et ***derivee_sigmoide(s)*** qui calculent respectivement les valeurs de la fonction d'activation sigmoïde et de sa dérivée en un point donné.
2. Créer la fonction ***sortie_Perceptron(x,w)*** qui calcule la sortie du perceptron à partir des vecteurs ***x*** et ***w***.
3. Créer la fonction ***initialisation(k)*** qui génère aléatoirement le vecteur $w^0 = \left(w_0^0, w_1^0, \cdots, w_k^0\right)$ des poids initiaux.
4. Créer la fonction ***gradient_local(x, e, w)*** qui calcule le gradient de la fonction erreur appliqué à l'exemple ***x*** ayant l'étiquette ***e***.
5. Créer la fonction ***gradient_Globale(X, E, w)*** qui calcule le gradient de la fonction erreur globale relative à tous les exemples x^i du corpus *X* et au vecteur des étiquettes $E = (\mathrm{e}_i)_{i=1,\cdots,N}$.
6. Créer la fonction ***perceptron(X, E, pas, n_iter)*** qui permet de construire un perceptron à partir du corpus d'apprentissage *X* et du vecteur des étiquettes ***E***. le pas de la méthode de descente de gradient est ***pas***, et ***iter*** représente le nombre d'itération.
7. Créer une fonction ***get_IO_data()*** qui permet de construire la matrice des exemples ***X***, et le vecteur des étiquettes ***E,*** et tester la fonction ***perceptron()*** sur ces données :

	X					E
Exemple 1	1	0	1	0		1
Exemple 2	1	0	1	1		1
Exemple 3	0	1	0	1		0