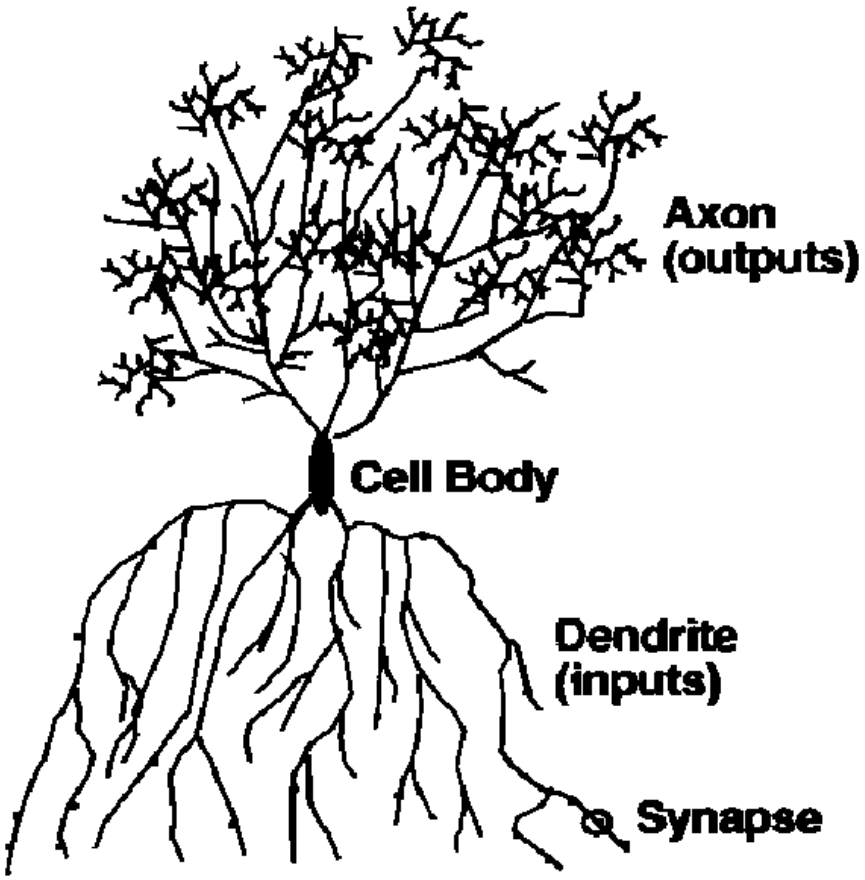
Cours IFT6266,
Premiers réseaux de neurones: le Perceptron

- **Biologie et informatique.** Chez l'humain, le siège de l'intelligence et de l'apprentissage est le cerveau. On peut s'en inspirer pour élaborer des algorithmes d'apprentissage: les réseaux de neurones artificiels (modèles connexionistes, voir les deux volumes [4], et pour la connection neurobiologique, [1]).



Comment est-ce que le traitement de l'information dans le cerveau et l'ordinateur de Von Neumann diffèrent?

Ordinateur de Von Neumann	Cerveau
calcul et mémoire séparés et centralisés	calcul et mémoire intégrés et distribués
programme = séquence d'instructions	calcul = satisfaction de multiples contraintes
exécution d'un sous-programme à la fois	combinaison simultanée de multiples sources d'information
un seul processeur très rapide	des centaines de milliards d'unités de calcul très lentes

- **Traitement de l'information dans le cerveau.**
  - Parallélisme massif: O(10^{11}) unités simples mais variées.
  - Connectivité (synapses: environ 1000/neurone).
  - Opération du neurone: beaucoup d'entrées, une sortie (envoyées à beaucoup d'autres neurones).
    1. intégration (sommation) spatiale (contributions des differentes synapses) et

temporelle (délais de transmissions, convolution temporelle),

2. non-linéarité (saturation du potentiel d'action).

– Adaptivité (lors de l'apprentissage):

  * nouvelles synapses,
  * mort de synapses,
  * modification de l'efficacité synaptique.

Mais on ne connait pas vraiment "l'algorithme d'apprentissage".

- **Le neurone formel**: une approximation très grossière de l'opération du neurone [2]. C'est un classifieur linéaire (sortie binaire). Un réseau (suffisamment complexe) de neurones formels peut représenter n'importe quelle fonction booléenne ou n'importe quelle partition de R^n.

- **Séparabilité linéaire**. Un ensemble d'exemples (dans un problème de classification) est linéairement séparable si il existe un classifieur linéaire qui peut tous les apprendre. Si on choisit des fonctions de $\mathcal{R}^n$ vers $\{$-1,1$\}$ de manière aléatoire, on se rend compte que les exemples tirés de la plupart d'entre ces fonctions ne sont pas séparables quand le nombre d'exemples est plus grand que la dimension de l'entrée. Plus précisemment, si nous avons n entrées binaires, il existe 2^n vecteurs d'entrées possibles, et 2^{2^n} fonctions binaires. Or, la fraction de ces fonctions représentables par un neurone formel est plus petite que $\frac{2^{n^2}}{n!}$ 3(Lewis & Coates 1967). Pour des entrées réelles, la fraction des 2^l étiquetages binaires de l points qu'un neurone formel peut représenter est 1 quand $l \leq n+1$ et $\frac{1}{2^{l-1}} \sum_{i=0}^{n} choose(l-1,i)$ quand $l \geq n+1$ ($choose(l,i)$ est le nombre de combinaisons de i éléments dans l).

- **L'algorithme du Perceptron** [5,6], permet d'apprendre les paramètres d'un neurone formel quand les exemples sont **linéairement séparables** (il existe donc une solution qui donne la bonne réponse sur les exemples d'apprentissage). Pour simplifier la notation on va inclure une entrée égale à 1 dans x, ce qui fait que le w_i correspondant est le "biais". Soit $f(x) = signe(w'x)$ et des exemples (x, y), $x \in \mathcal{R}^n$, $y \in \{$-1,1$\}$. Après chaque exemple (x, y)

  – nouveau $w_i \leftarrow w_i + \eta \, 1_{y \neq f(x)} \, y \, x_i$

  où η est un scalaire (le pas de gradient) dont la valeur n'est pas importante. c'est à dire que si il n'y a pas d'erreur, pas de modification, sinon, on change w de manière à ce que $w'x$ change dans la bonne direction. Le **théorème du Perceptron** nous dit que cet algorithme converge en un temps fini quand les exemples sont linéairement séparables (i.e., quand il existe w qui donne 0 erreur d'apprentissage).

  Notons que la règle de mise à jour des paramètres ci-dessus peut être interprété comme une descente de gradient stochastique sur un critère qui n'est pas le nombre d'erreurs

de classification (celle-là a une dérivée 0 presque partout, donc pas utilisable) mais un critère qui l'approxime, la somme des $-1_{f(x)\neq y} w'xy$ sur les exemples (x, y). Ce critère est donc 0 quand il n'y a pas d'erreur, et il augmente linéairement en fonction de la distance signée entre x et la surface de décision, quand x est du mauvais côté. Si on interprète cette règle de mise à jour comme une descente de gradient stochastique, alors on peut montrer la convergence asymptotique (c'est moins bien) même dans le cas où les exemples ne sont pas linéairement séparables, mais il y a des conditions sur η (il doit décroitre assez vite mais pas trop, voir le feuillet sur la descente de gradient).

References

1. Gluck, M. and Rumelhart, D. (1990). *Neuroscience and connectionist theory*. Lawrence Erlbaum, London.

2. McCulloch, W. and Pitts, W. (1943). A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5.

3. P. M. Lewis II and C. L. Coates, *A realization procedure for threshold gate networks*, Proceedings of the Third Annual Symposium on Switching Circuit Theory and Logical Design (Chicago, Illinois), American Institute of Electrical Engineers, 7–12 October 1962, pp. 159–168.

4. Rumelhart, D., McClelland, J., and the PDP Research Group (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1. MIT Press, Cambridge.

5. F. Rosenblatt, *The perceptron — a perceiving and recognizing automaton*, Tech. Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca, N.Y., 1957.

6. F. Rosenblatt, *Principles of neurodynamics*, Spartan, New York, 1962.

7. Y. LeCun, *Une procédure d'apprentissage pour réseau à seuil assymétrique*, Cognitiva 85: A la Frontière de l'Intelligence Artificielle, des Sciences de la Connaissance et des Neurosciences (Paris 1985), CESTA, Paris, 1985, pp. 599–604.

8. D.B. Parker, *Learning logic*, Tech. Report TR–47, Center for Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, MA, 1985.

9. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, *Learning representations by back-propagating errors*, Nature **323** (1986), 533–536.

10. B. Widrow, *Generalization and information storage in networks of adaline "neurons"*, Self-Organizing Systems 1962 (Chicago 1962) (M.C. Yovits, G.T. Jacobi, and G.D. Goldstein, eds.), Spartan, Washington, 1962, pp. 435–461.