Réseaux de neurones artificiels

Hélène Paugam-Moisy

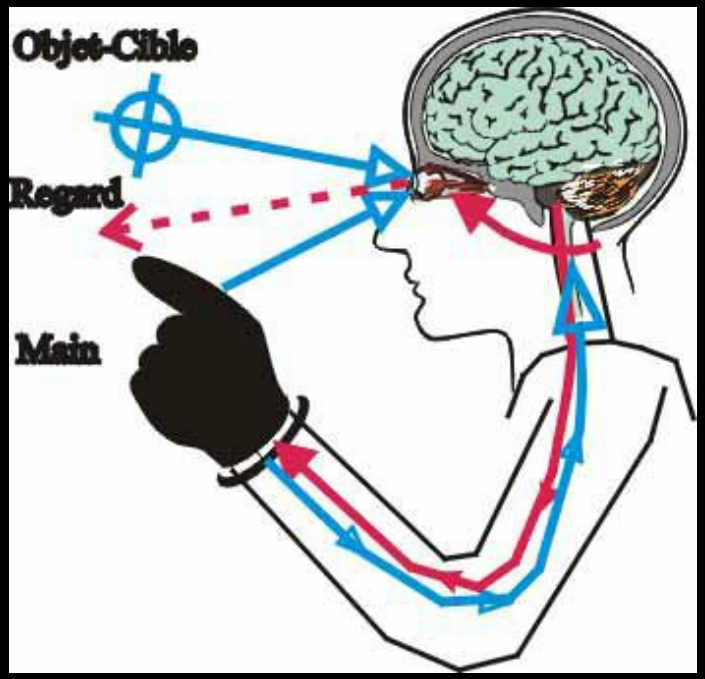
Professeur d'Informatique – Université des Antilles

Dept DMI – Labo LAMIA

hpaugam@univ-antilles.fr

Les réseaux de neurones naturels

Cerveau, neurones et intelligence

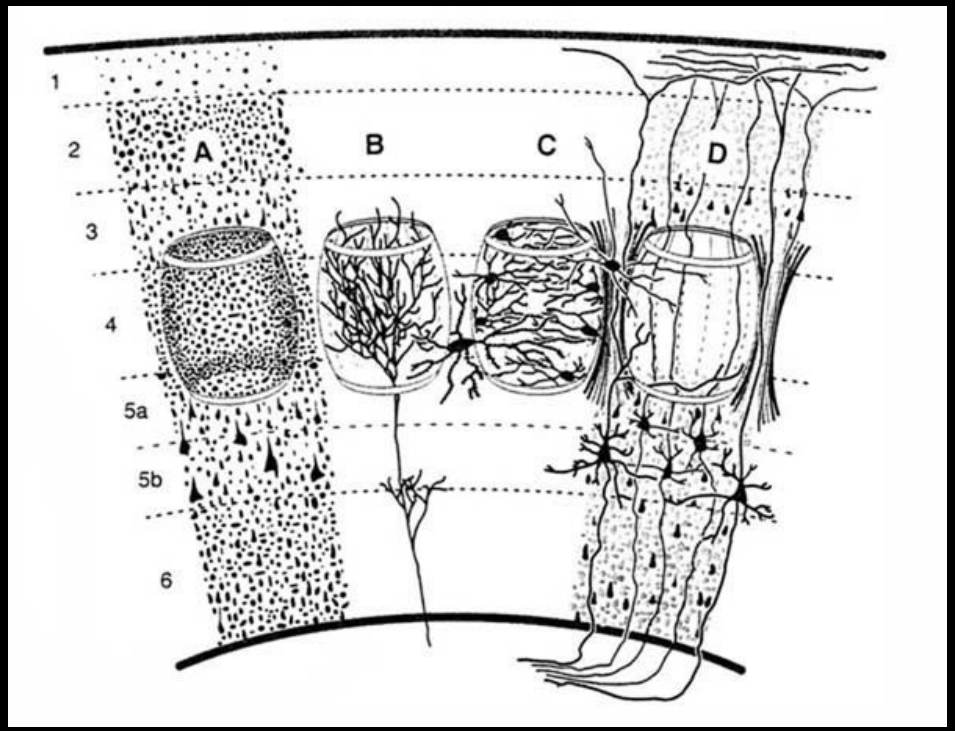


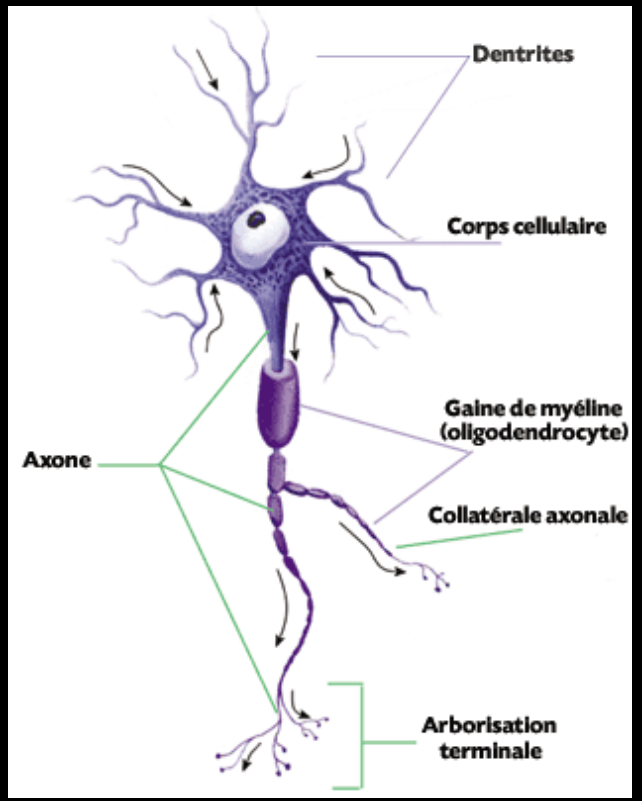
Chez les animaux, la place occupée par le cortex semble être proportionnelle au développement des facultés intelligentes.

Le cerveau humain est constitué d'environ 10^{11} neurones ;
en moyenne, chaque neurone a de l'ordre de 10^{4} connexions.

Neurones biologiques / naturels

Les neurones forment des réseaux de communication complexes, chaque neurone établissant de très nombreuses connexions avec d'autres.



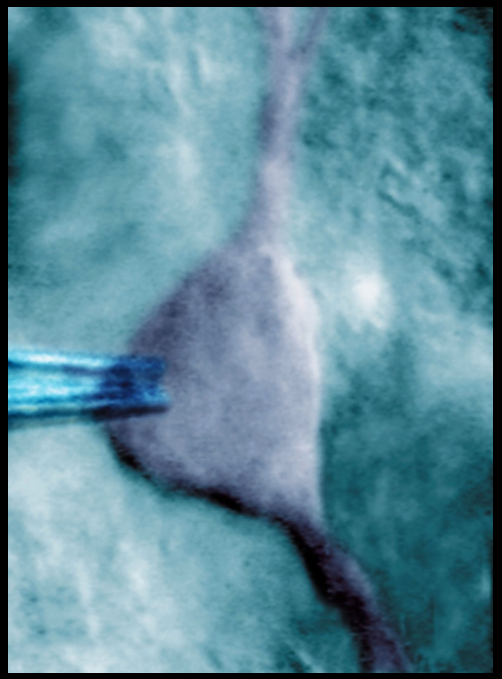


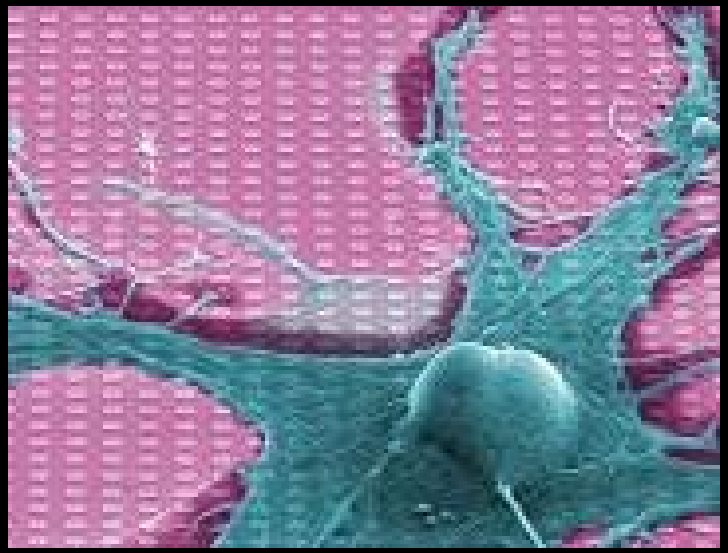
vibrisses du rat : neurones sensitifs => thalamus
(***White et Peters***) neurones thalamiques => cortex

Neurones biologiques / naturels

Première schématisation
du neurone (Dieters, 1865)

Enregistrement d'un neurone
vivant : micro-électrode de verre

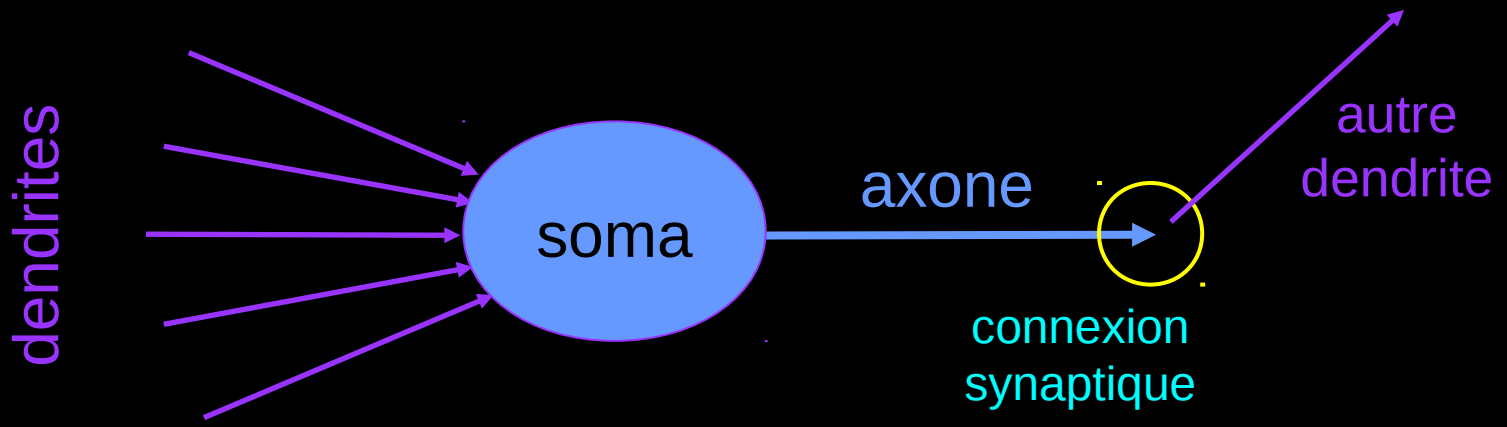




Neuropuce *(Infineon)* : 16 384 capteurs au mm^2
pour enregistrer tout signal électrique émis par
un neurone vivant, ici celui d'une "limace de boue"
[Lymnaea stagnalis]

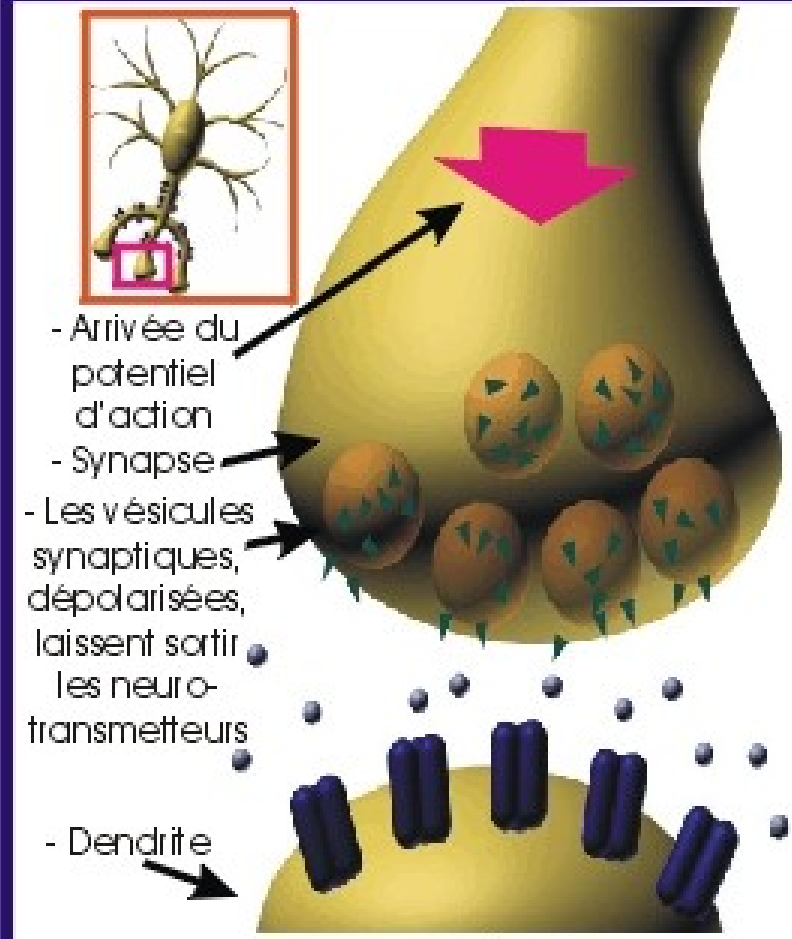
Neurones biologiques / naturels

Schématiquement, un neurone biologique est constitué d'un arbre dendritique, d'un soma et d'un axone.

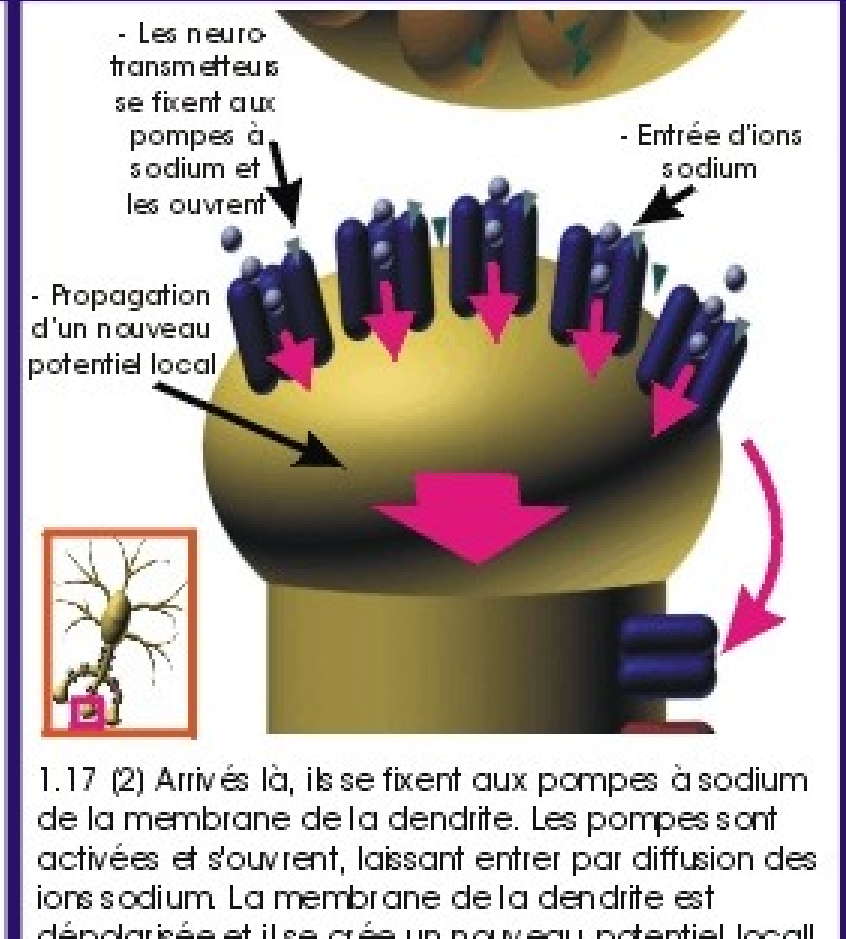


Un neurone peut émettre des impulsions sur son axone. Elles se propagent vers les dendrites des neurones auxquels il est connecté, qui les transmettent à leur corps cellulaire (soma), via des connexions pondérées par des synapses.

Connexions synaptiques



1.17 (1) Le potentiel d'action dépolarise la synapse. Les vésicules synaptiques fusionnent donc avec la membrane et libèrent leurs neurotransmetteurs. Ceux-ci diffusent jusqu'à la dendrite.



1.17 (2) Arrivés là, ils se fixent aux pompes à sodium de la membrane de la dendrite. Les pompes sont activées et s'ouvrent, laissant entrer par diffusion des ions sodium. La membrane de la dendrite est dépolarisée et il se crée un nouveau potentiel local! Celui-ci se rendra exciter le corps cellulaire pour qu'il atteigne son seuil de déclenchement et déclenche de ce fait un potentiel d'action.

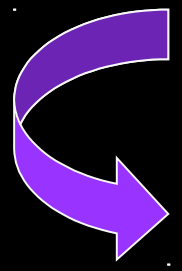
David Laflamme

Plasticité synaptique

Loi de renforcement de Hebb (1949) :

Postulat physiologique :

“quand un axone de la cellule A est assez proche pour exciter une cellule B et quand, de façon repétée et persistante, il participe à son activation, un certain processus de croissance ou un changement métabolique s’installe, dans une cellule ou dans les deux, tel que l’efficacité de A, en sa qualité de cellule qui active B, est augmentée”.



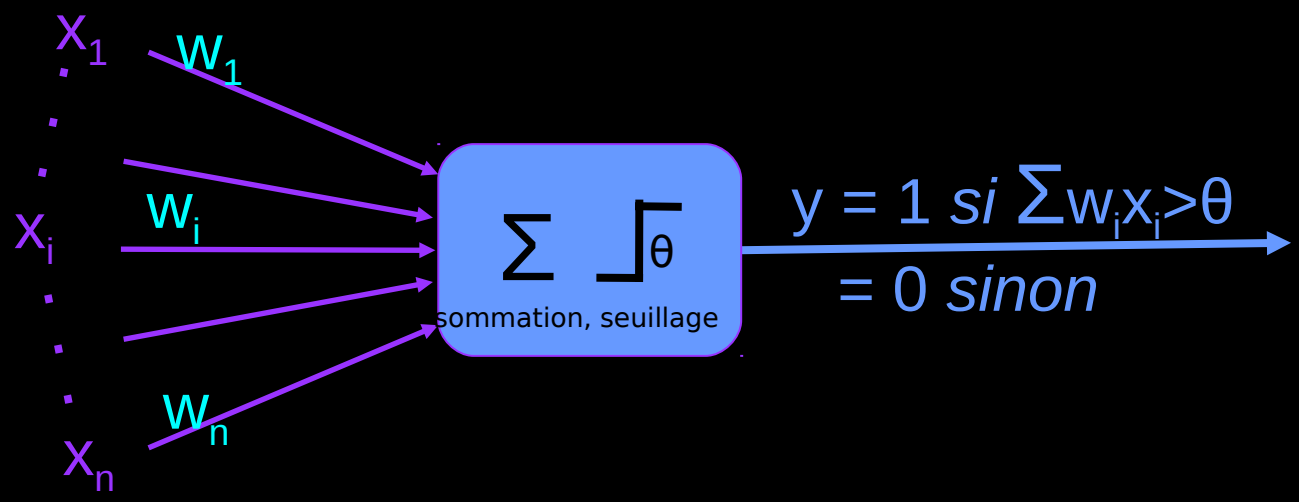
En résumé : si deux neurones sont souvent excités simultanément, alors ils augmentent la force (le poids) de leur interconnexion .

=> c’est le principal fondement des règles d’apprentissage, dans les réseaux de neurones artificiels.

Les réseaux de neurones artificiels

Neurones artificiels

Le premier modèle mathématique de neurone est le neurone formel de McCulloch et Pitts (1943).



Loi du “ tout ou rien ” : le neurone émet ou non une impulsion sur son axone, selon que la somme pondérée de ses entrées dendritiques dépasse ou non son seuil θ.

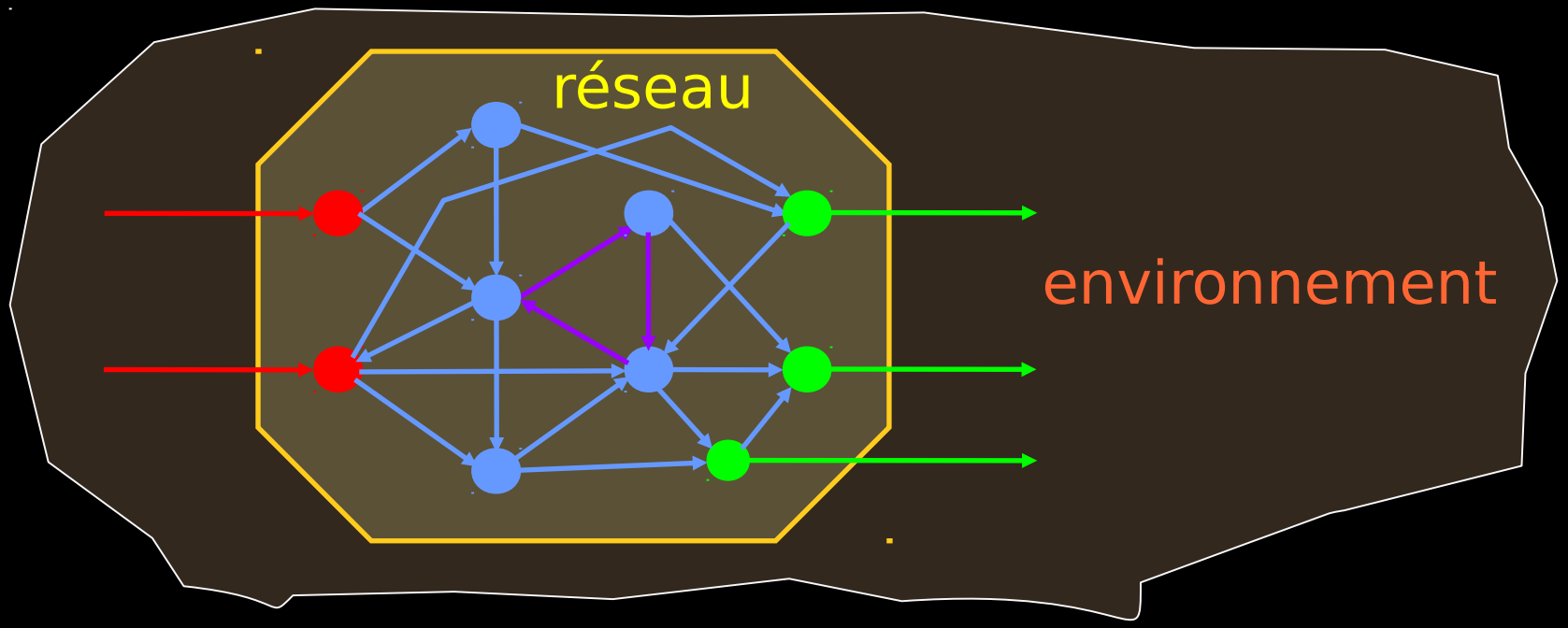
Réseaux de neurones artificiels

Un réseau de neurones artificiels est un ensemble de neurones formels, connectés entre eux selon une certaine architecture / topologie.

Les activités se propagent de neurone en neurone selon une certaine dynamique.

Les poids des connexions sont modifiés / adaptés par une règle d'apprentissage, à partir d'exemples.

Architecture d'un réseau



neurone d'entrée

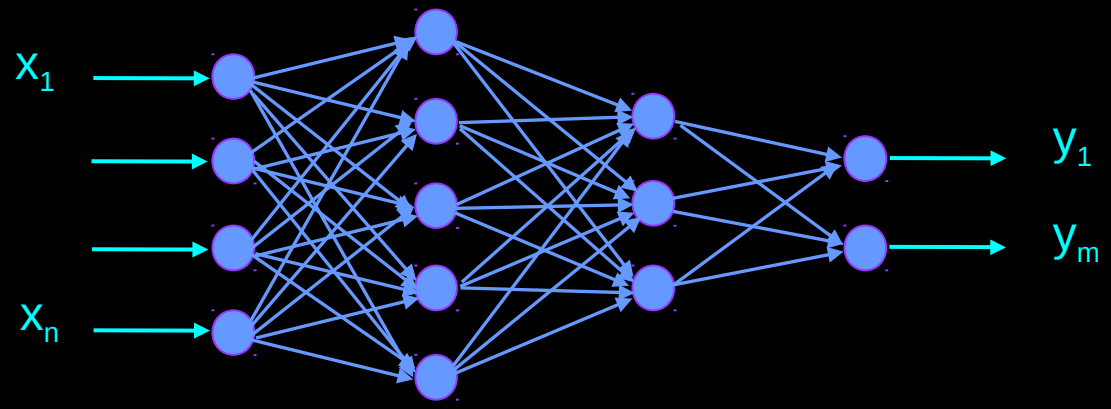
neurone caché

neurone de sortie

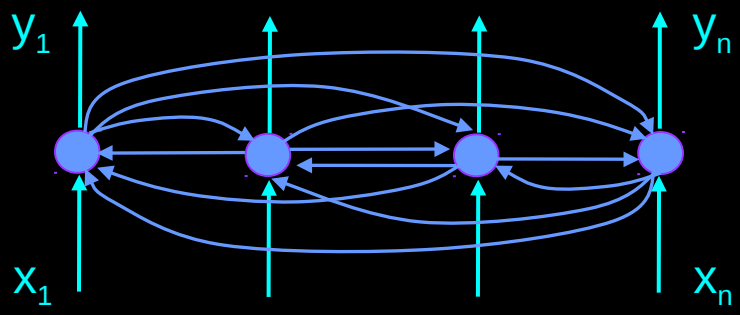
Le graphe d'interconnexion peut être quelconque ; il peut y avoir des circuits (boucles).

Topologies et dynamiques

Les architectures varient, entre deux schémas extrêmes :



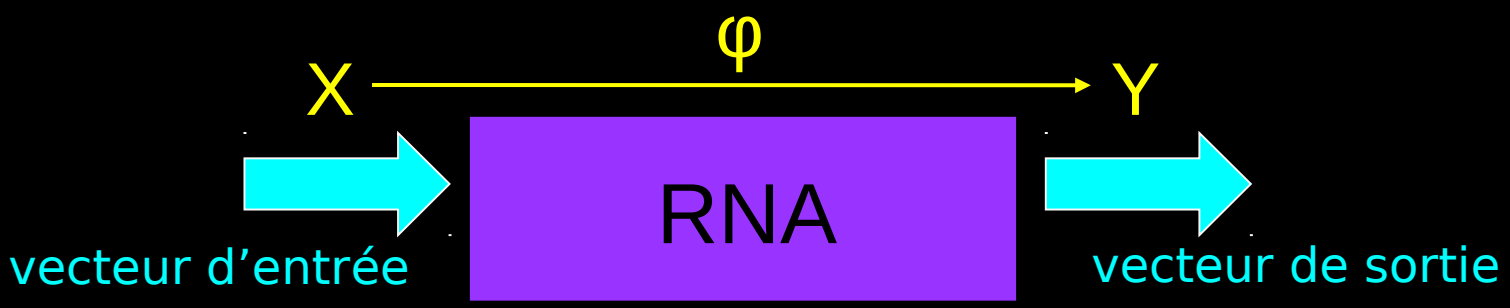
topologie en couches => dynamique feedforward



topologie bouclée => dynamique récurrente

Réseaux de neurones artificiels

On peut aussi voir un réseau de neurones artificiels comme un outil de modélisation de type “ boîte noire ” :



Le RNA réalise une association d’un espace d’entrée X vers un espace de sortie Y. La “ fonction ” φ est définie par les caractéristiques du réseau. Ceux-ci sont ajustés au problème par une règle d’apprentissage
à partir d’une base d’exemples.

Apprentissage des poids synaptiques

Le plus couramment, les règles d'apprentissage modifient les poids synaptiques (nbs réels) qui relient les neurones.

On associe habituellement des poids positifs à des connexions excitatrices et des poids négatifs à des connexions inhibitrices

cependant...

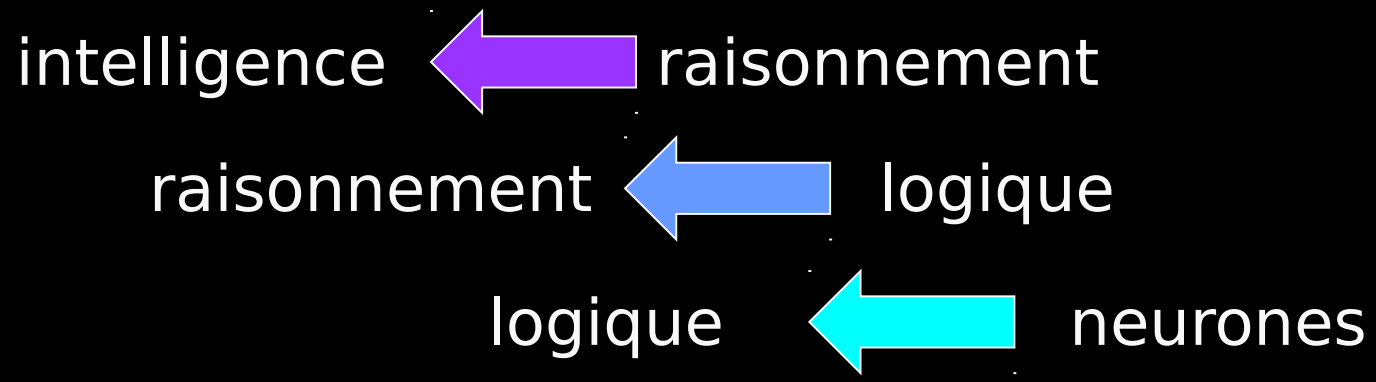
les règles d'apprentissage des modèles connexionnistes peuvent modifier le signe d'un poids synaptique donné, ce qui n'est guère réaliste, sur le plan biologique.

Motivation initiale

La motivation initiale n'était pas non plus très biologique :

l'idée sous-jacente à la définition du neurone formel était de chercher à montrer que la pensée intelligente résultait du fonctionnement des neurones du cerveau...

mais, à l'époque, on pensait que :



Synthèse des fonctions booléennes

Ainsi, les premiers travaux de McCulloch et Pitts ont été de réaliser la synthèse des fonctions booléennes élémentaires à partir de réseaux de neurones formels.

Parallèlement, les mathématiciens tentaient de démontrer que tout raisonnement pouvait se formaliser par des combinaisons complexes de règles logiques.

donc, dans l'esprit des travaux de Turing :

grand reseau de neurones

raisonnement complexe, intelligent

Historique du connexionnisme

1940

Neurone formel de McCulloch & Pitts

Loi de renforcement de Hebb

} notions fondatrices

1950

Perceptron de Rosenblatt

1960

Adaline de Widrow

1970

1980

Réseau de Hopfield

Cartes auto-organisatrices de Kohonen

Réseaux MLP Rumelhart et al.

1990

Réseaux RBF Moody & Darken

Support Vector Machines Vapnik

2000

20...

Apprentissage et généralisation

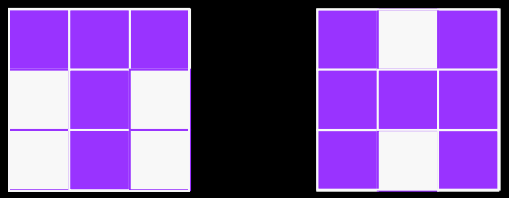
vers Premiers Modeles

Neurones jouets (empruntés à Alexander, 1990)

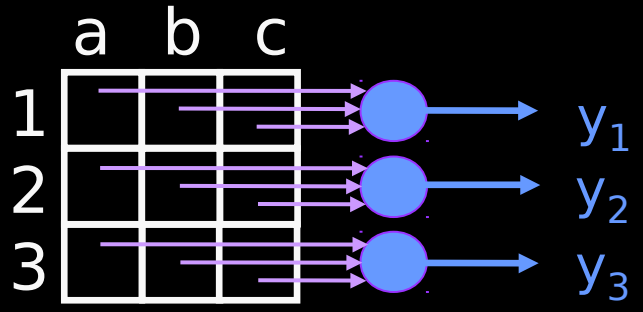
Codage d'un caractère : 3x3 pixels

0 = blanc
1 = violet

Base d'exemples :
un "T" et un "H"



Réseau = 3 neurones jouets



Règle d'apprentissage : un nouveau motif force la sortie du neurone à 0 (resp. à 1) selon qu'il est plus proche, au sens de la distance de Hamming, de l'exemple pour lequel elle répond 0 (1) et en cas d'égalité, la sortie reste indéterminée [on notera "?"].

Reconnaissance des caractères

mo -tif	a_1	b_1	c_1	y_1
	0	0	0	?
	0	0	1	?
	0	1	0	?
	0	1	1	?
	1	0	0	?
H	1	0	1	0
	1	1	0	?
T	1	1	1	1

mo -tif	a_2	b_2	c_2	y_2
	0	0	0	?
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
	1	0	1	?
	1	1	0	?
H	1	1	1	0

mo -tif	a_3	b_3	c_3	y_3
	0	0	0	?
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
H	1	0	1	0
	1	1	0	?
	1	1	1	?

Sorties des 3 neurones, sur simple présentation des exemples :

- T --> les 3 unités répondent 1
- H --> les 3 unités répondent 0

Application de la règle d'apprentissage

mo -tif	a_1	b_1	c_1	y_1
	0	0	0	?
	0	0	1	?
	0	1	0	?
	0	1	1	?
	1	0	0	?
H	1	0	1	0
	1	1	0	?
T	1	1	1	1

mo -tif	a_2	b_2	c_2	y_2
	0	0	0	?
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
	1	0	1	?
	1	1	0	?
H	1	1	1	0

mo -tif	a_3	b_3	c_3	y_3
	0	0	0	?
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
H	1	0	1	0
	1	1	0	?
	1	1	1	?

$^{T}\, d_{Hamming}$ [(0,0,0) , (1,1,1)] = 3 $^{H}\, d_{Hamming}$ [(0,0,0) , (1,0,1)] = 2

(0,0,0) est plus proche du H que du T  $y_1 := 0$

Application de la règle d'apprentissage

mo -tif	a_1	b_1	c_1	y_1
	0	0	0	0
	0	0	1	?
	0	1	0	?
	0	1	1	?
	1	0	0	?
H	1	0	1	0
	1	1	0	?
T	1	1	1	1

mo -tif	a_2	b_2	c_2	y_2
	0	0	0	?
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
	1	0	1	?
	1	1	0	?
H	1	1	1	0

mo -tif	a_3	b_3	c_3	y_3
	0	0	0	?
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
H	1	0	1	0
	1	1	0	?
	1	1	1	?

$^{T}d_{Hamming}[(0,0,0),(0,1,0)] = 1$ $^{H}d_{Hamming}[(0,0,0),(1,1,1)] = 3$

(0,0,0) est plus proche du T que du H → $y_2 := 1$

Application de la règle d'apprentissage

mo -tif	a_1	b_1	c_1	y_1
	0	0	0	0
	0	0	1	?
	0	1	0	?
	0	1	1	?
	1	0	0	?
H	1	0	1	0
	1	1	0	?
T	1	1	1	1

mo -tif	a_2	b_2	c_2	y_2
	0	0	0	1
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
	1	0	1	?
	1	1	0	?
H	1	1	1	0

mo -tif	a_3	b_3	c_3	y_3
	0	0	0	?
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
H	1	0	1	0
	1	1	0	?
	1	1	1	?

$T\ d_{Hamming}$ [(0,0,1) , (0,1,0)] = 2 $H\ d_{Hamming}$ [(0,0,1) , (1,1,1)] = 2

(0,0,1) est à égale distance du T et du H → y_2 := "?"

Quand l'apprentissage est terminé ...

mo -tif	a_1	b_1	c_1	y_1
	0	0	0	0
	0	0	1	0
	0	1	0	1
	0	1	1	1
	1	0	0	0
H	1	0	1	0
	1	1	0	1
T	1	1	1	1

mo -tif	a_2	b_2	c_2	y_2
	0	0	0	1
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
	1	0	1	0
	1	1	0	?
H	1	1	1	0

mo -tif	a_3	b_3	c_3	y_3
	0	0	0	1
	0	0	1	0
T	0	1	0	1
	0	1	1	1
	1	0	0	0
H	1	0	1	0
	1	1	0	1
	1	1	1	0

Ici, il reste seulement quelques cas d'indétermination et seulement pour le neurone associé à la ligne 2.

... on va chercher à généraliser

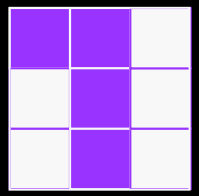
Le modèle doit pouvoir servir à reconnaître de nouveaux caractères, différents des deux exemples appris.

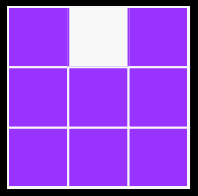
	neurone	1
	0 0 0	0
	0 0 1	0
	0 1 0	1
	0 1 1	1
	1 0 0	0
H	1 0 1	0
	1 1 0	1
T	1 1 1	1

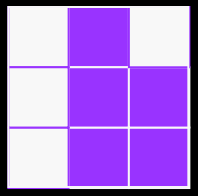
	neurone	2
	0 0 0	1
	0 0 1	?
T	0 1 0	1
	0 1 1	?
	1 0 0	?
	1 0 1	0
	1 1 0	?
H	1 1 1	0

	neurone	3
	0 0 0	1
	0 0 1	0
T	0 1 0	1
	0 1 1	1
	1 0 0	0
H	1 0 1	0
	1 1 0	1
	1 1 1	0

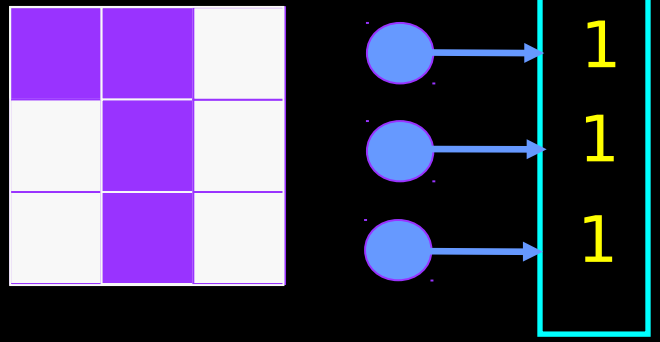
Que va répondre le réseau pour les motifs ci-dessous ?







Utilisation du réseau en généralisation



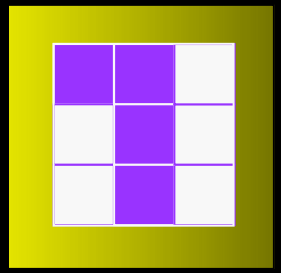
même réponse
que pour le :

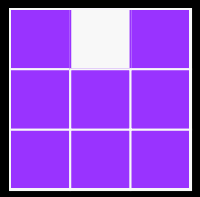
	neurone			1
	0	0	0	0
	0	0	1	0
	0	1	0	1
	0	1	1	1
	1	0	0	0
H	1	0	1	0
	1	1	0	1
T	1	1	1	1

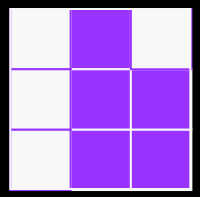
	neurone			2
	0	0	0	1
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
	1	0	1	0
	1	1	0	?
H	1	1	1	0

	neurone			3
	0	0	0	1
	0	0	1	0
T	0	1	0	1
	0	1	1	1
	1	0	0	0
H	1	0	1	0
	1	1	0	1
	1	1	1	0

Que répond le réseau pour le premier caractère (à gauche) ?

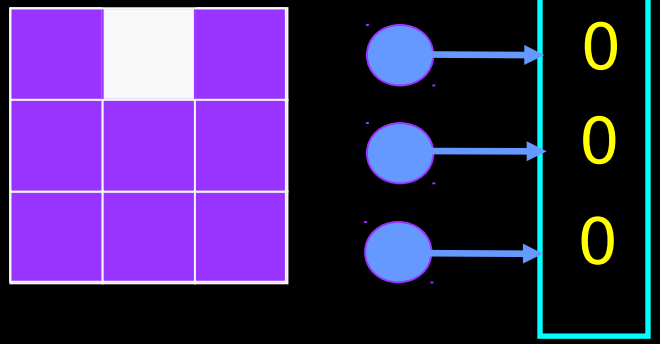






c’est un T bruité

Utilisation du réseau en généralisation



même réponse
que pour le :

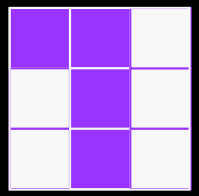
H

neurone				1
	0	0	0	0
	0	0	1	0
	0	1	0	1
	0	1	1	1
	1	0	0	0
H	1	0	1	0
	1	1	0	1
T	1	1	1	1

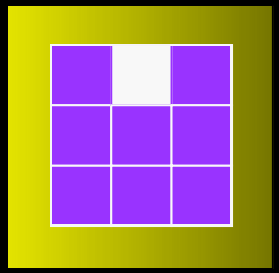
neurone				2
	0	0	0	1
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
	1	0	1	0
	1	1	0	?
H	1	1	1	0

neurone				3
	0	0	0	1
	0	0	1	0
T	0	1	0	1
	0	1	1	1
	1	0	0	0
H	1	0	1	0
	1	1	0	1
	1	1	1	0

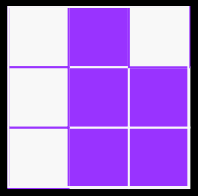
Que répond le réseau pour le second caractère (au milieu) ?



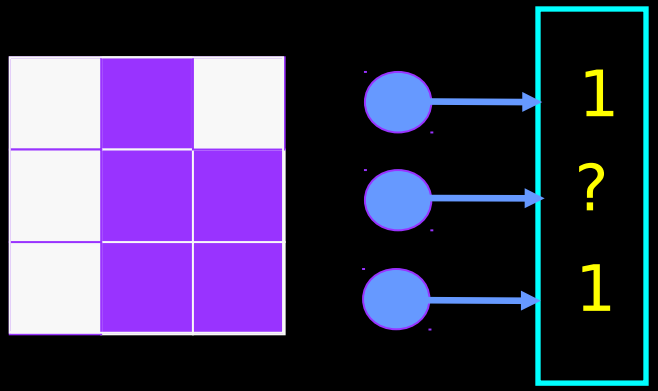
T (bruité)



c’est un H bruité



Utilisation du réseau en généralisation



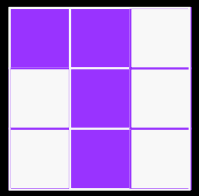
même réponse
que ..... RIEN !

	neurone			1
	0	0	0	0
	0	0	1	0
	0	1	0	1
	0	1	1	1
	1	0	0	0
H	1	0	1	0
	1	1	0	1
T	1	1	1	1

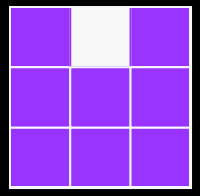
	neurone			2
	0	0	0	1
	0	0	1	?
T	0	1	0	1
	0	1	1	?
	1	0	0	?
	1	0	1	0
	1	1	0	?
H	1	1	1	0

	neurone			3
	0	0	0	1
	0	0	1	0
T	0	1	0	1
	0	1	1	1
	1	0	0	0
H	1	0	1	0
	1	1	0	1
	1	1	1	0

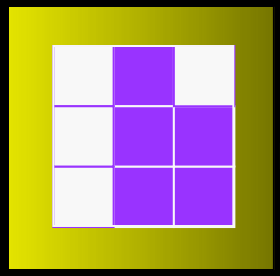
Que répond le réseau pour le dernier caractère (à droite) ?



T (bruité)



H (bruité)



caractère
non reconnu

Premiers modèles connexionnistes

App. et Gene.

Historique du connexionnisme

1940

Neurone formel de McCulloch & Pitts

Loi de renforcement de Hebb

} notions fondatrices

1950

1960

Perceptron de Rosenblatt

Adaline de Widrow

1970

1980

Réseau de Hopfield

Cartes auto-organisatrices de Kohonen

Réseaux MLP Rumelhart et al.

1990

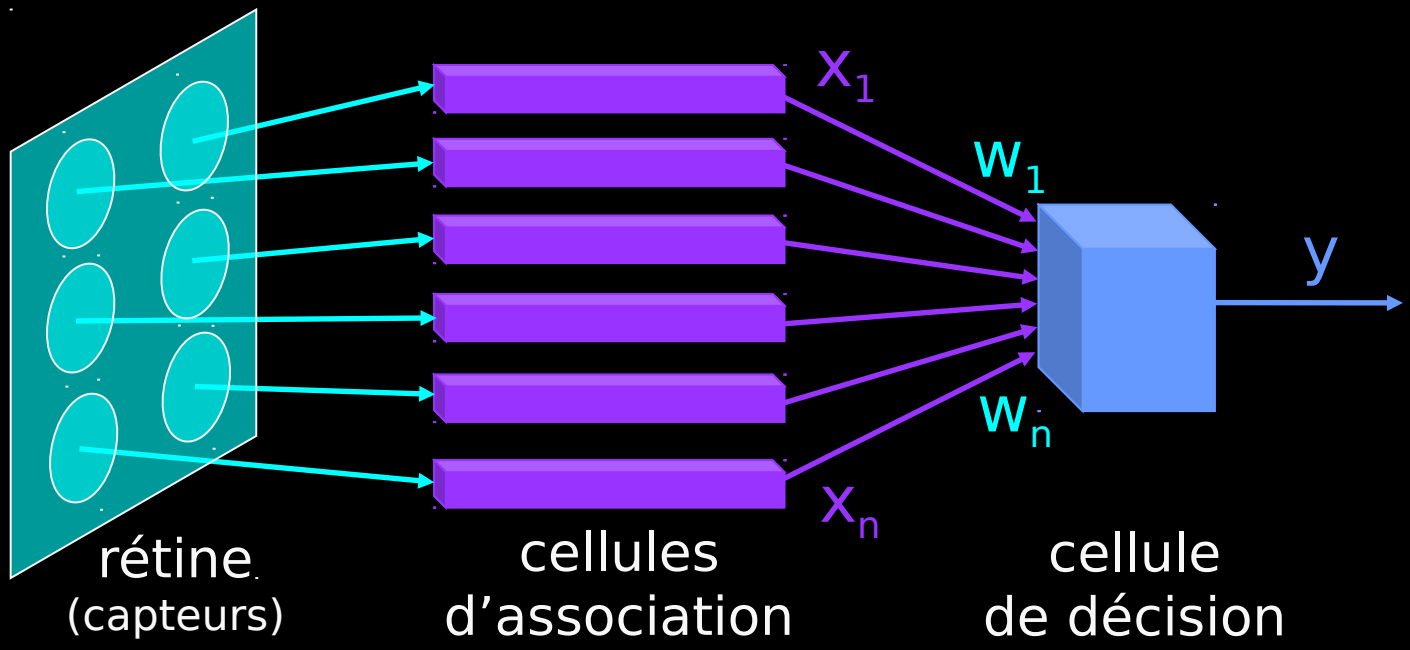
Réseaux RBF Moody & Darken

Support Vector Machines Vapnik

2000

20...

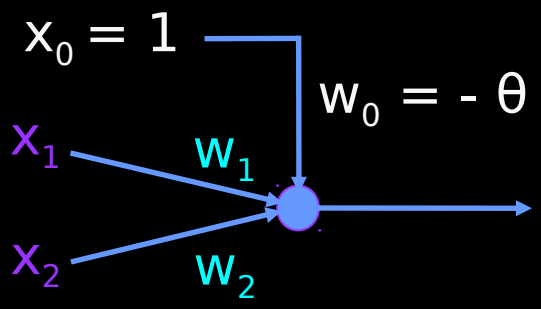
Le Perceptron (Rosenblatt, 1958)



La cellule de décision est un neurone formel (neurone à seuil) à poids variables, le reste du dispositif est fixe.

$$y = \text{signe}\left(\sum_{i=1}^{n} w_i x_i - \theta \right)$$

Seuil du neurone => “biais”



$y = 1$ si $w_1x_1 + w_2x_2 > \theta$

$y = 0$ sinon

Notion de biais :

on inclut le seuil dans les connexions, avec un poids $w_0 = - \theta$, en créant une entrée virtuelle et fixe, $x_0 = 1$

On étudie alors le signe de W^TX, avec $W=(w_0,w_1,w_2)$ et $X=(x_0, x_1, x_2)$

$w_1x_1+w_2x_2 > \theta \iff w_0x_0+w_1x_1+w_2x_2 > 0 \iff W^TX > 0$

Règle d'apprentissage du Perceptron

INIT : Choisir une valeur quelconque pour W

TEST : Choisir un exemple X dans $S = S^+ \cup S^-$

si $X \in S^+$ et $W^T X > 0$ alors aller en TEST

si $X \in S^+$ et $W^T X \leq 0$ alors aller en ADD

si $X \in S^-$ et $W^T X \leq 0$ alors aller en TEST

si $X \in S^-$ et $W^T X > 0$ alors aller en SUB

ADD : Remplacer W par W + X

Aller en TEST

SUB : Remplacer W par W – X

Aller en TEST

apprentissage supervisé

par correction d'erreur

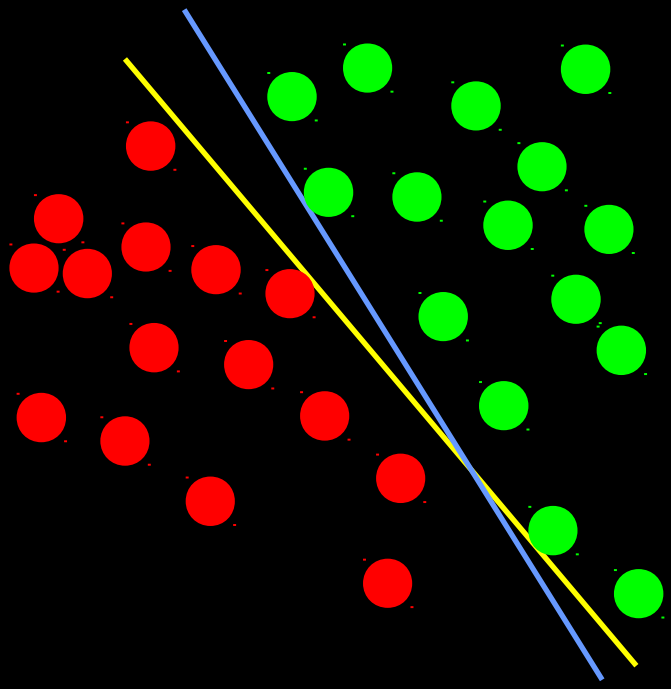
Théorème de convergence

Soit S un ensemble quelconque de vecteurs unitaires.
S'il existe un vecteur unitaire W* et un réel $\delta>0$ tels que
$W^{*T}X > \delta$ pour tout X dans S^+
et
$W^{*T}X < -\delta$ pour tout X dans S^-
alors le programme définissant la règle d'apprentissage du Perceptron passera par les procédures ADD et SUB seulement un nombre fini de fois (convergence algo).

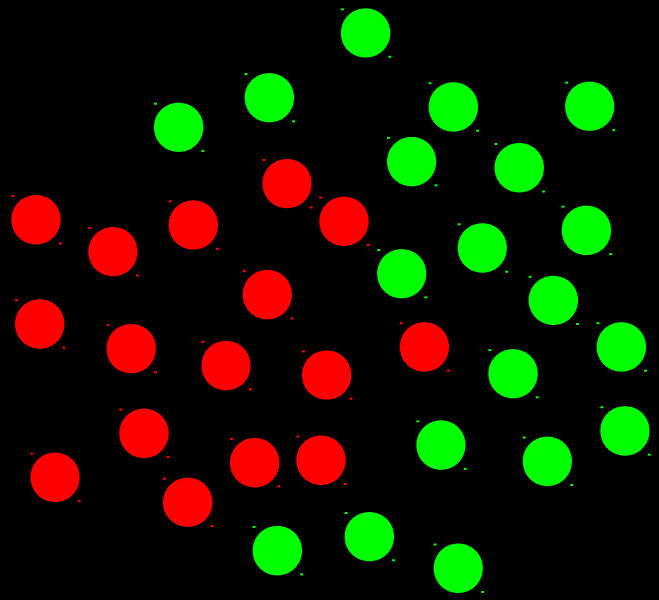
mais...

un tel W* n'existe que si les exemples de S^+ et S^- sont linéairement séparables.

Linéairement séparable ou pas ?



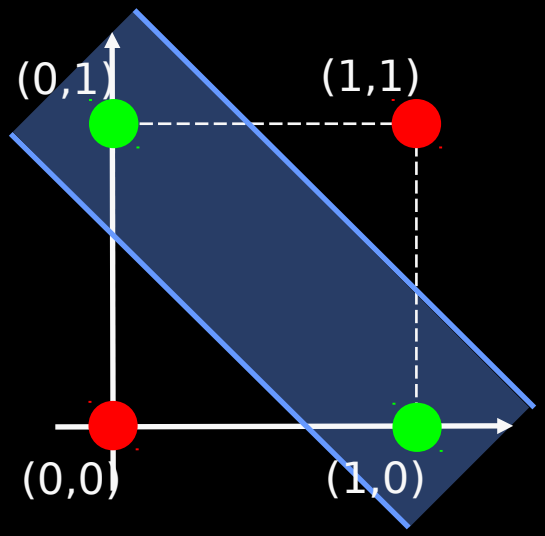
Ces deux classes sont linéairement séparables



Celles-ci ne le sont PAS

Synthèse des fonctions booléennes

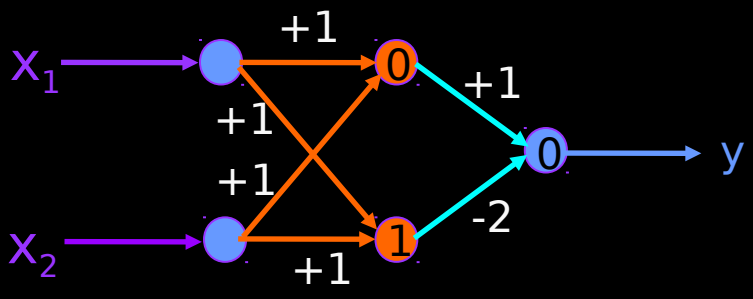
x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0



Une seule droite ne suffit pas à séparer les quatre exemples

le XOR n'est pas linérairement séparable

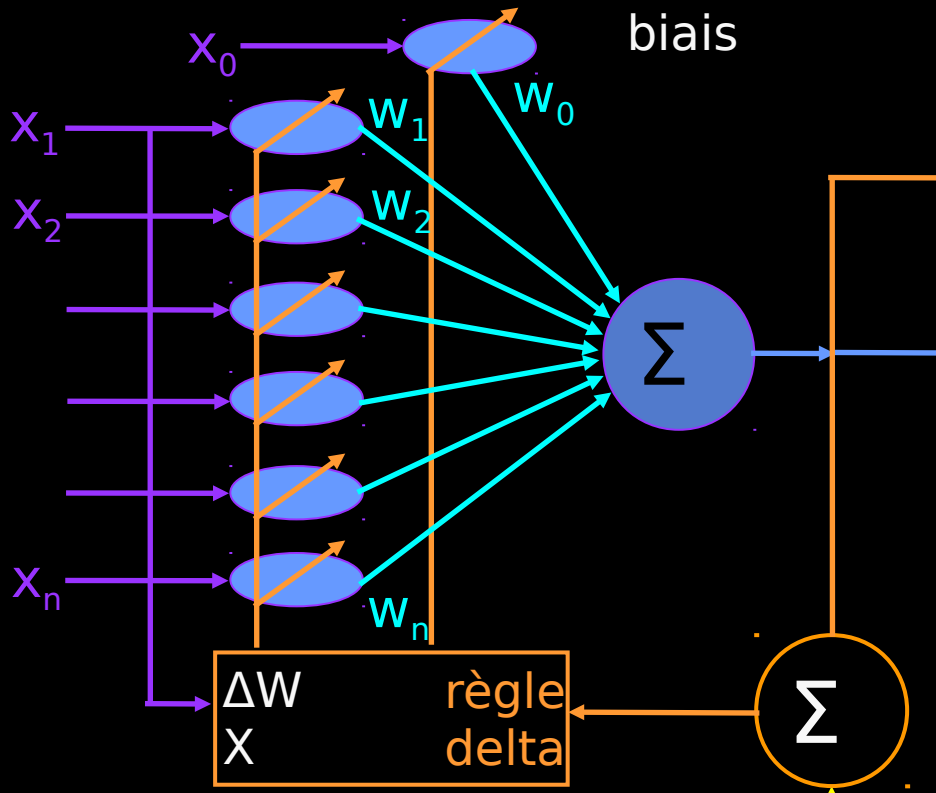
Un réseau à une couche cachée est nécessaire pour réaliser le XOR :



mais... la règle d'apprentissage du Perceptron ne marche plus !

L'Adaline : Adaptive Linear Element

(Widrow, 1960)



Inspiration : traitement du signal

mais... le modèle a les mêmes limitations que le Perceptron

Règle d'apprentissage de l'Adaline

Règle de Widrow-Hoff, encore appelée règle delta

A chaque instant t :

présentation d'un exemple X_t

calcul de la sortie linéaire :

$$s(t) = \Sigma\, w_i(t)x_i(t) = W_t^T X_t$$

présentation de la sortie désirée sd(t)

calcul de l'erreur quadratique :

$$E(t) = \tfrac{1}{2}\,(sd(t) - s(t))^2$$

mise à jour des poids :

$$(\forall i \in \{1,\ldots,n\})\quad w_i(t+1) = w_i(t) + \alpha\,(sd(t) - s(t))\,x_i(t)$$

ce qui peut s'écrire : $\Delta W_t = \alpha\,(sd_t - W_t^T X_t)\,X_t$

apprentissage supervisé, par descente en gradient

Les modèles “classiques” de RNA

Historique du connexionnisme

1940

Neurone formel de McCulloch & Pitts

Loi de renforcement de Hebb

} notions fondatrices

1950

Perceptron de Rosenblatt

1960

Adaline de Widrow

1970

1980

Réseau de Hopfield

Cartes auto-organisatrices de Kohonen

Réseaux MLP Rumelhart et al.

1990

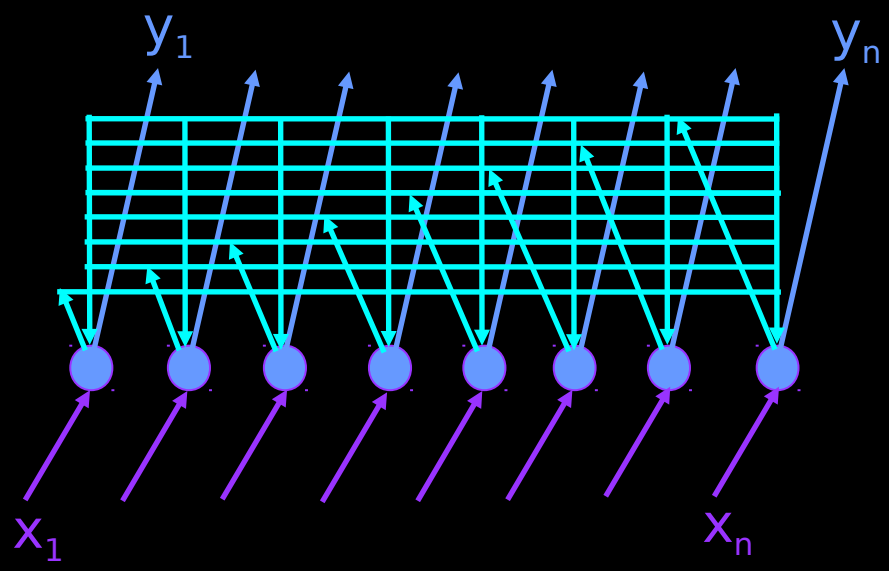
Réseaux RBF Moody & Darken

Support Vector Machines Vapnik

2000

20...

Réseau de Hopfield (Hopfield, 1982)



Architecture du réseau :

Chaque neurone est connecté à tous les autres neurones.

mémoire associative (auto-association)

Dynamique du réseau :

$y_i(0) = x_i$

$$y_i(t+1) = \text{signe}\left(\sum_{j=1}^{n} w_{ij} y_j(t) - \theta_i \right)$$

sorties calculées : $y_i(t_{cvg})$ pour i allant de 1 à n

Apprentissage du réseau de Hopfield

Règle de Hebb, quantitative

Ensemble S d'exemples X^s, à composantes bipolaires

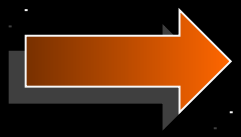
$$(\forall i \in \{1,\ldots,n\}) \quad x_i^s \in \{-1,+1\}$$

Calcul (statique) des poids du réseau :

$$(\forall i \in \{1,\ldots,n\}) \quad w_{ii} = 0$$

$$(\forall (i,j) \in \{1,\ldots,n\}^2 / i \neq j) \quad w_{ij} = \sum_{X^s \in S} x_i^s x_j^s$$

La matrice des poids est symétrique, à diagonale nulle

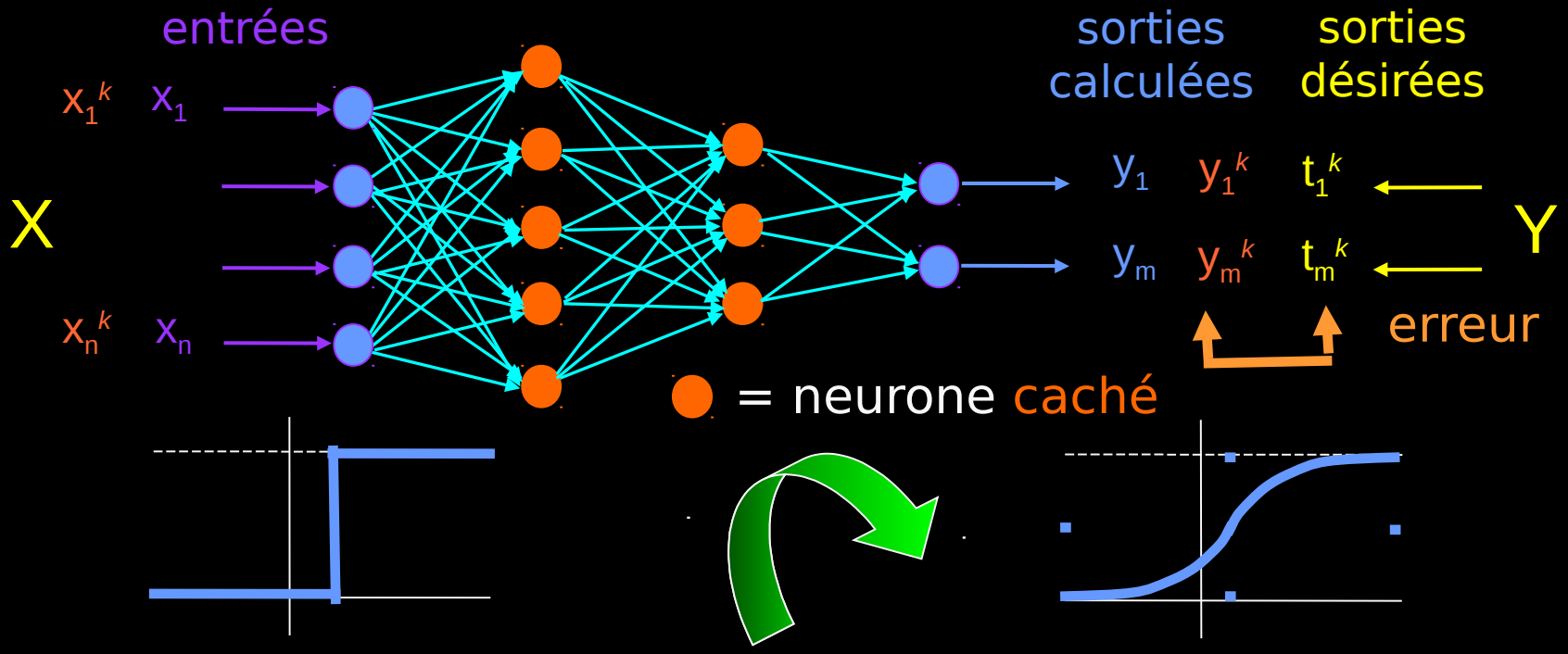


(sous certaines conditions)

la dynamique converge
vers les exemples appris.

Réseau multicouche, ou réseau MLP

(Rumelhart et al., 1986)



neurone à seuil N_j

[comme dans le Perceptron]

$y_j = \text{signe}(\Sigma w_{ij} x_i - \theta_j)$

neurone sigmoïde N_j

Ex. $\sigma = \tanh$

$y_j = \sigma (\Sigma w_{ij} x_i - \theta_j)$

Apprentissage par rétro-propagation

Apprentissage par rétro-propagation du gradient d'erreur

- Initialiser les poids w_{ij} aléatoirement, sur chaque couche

Répéter, jusqu'à critère d'arrêt, et pour chaque exemple X^s de S :

$\forall s$

- présenter les x_i^s aux neurones d'entrée
- calculer les activités, de couche en couche, jusqu'aux sorties y_j^s
- mesurer l'erreur entre sorties calculées y_j^s et sorties désirées sd_j^s
- en sens inverse, calculer les gradients et modifier les poids w_{ij}

Grand nombre de passes, avant et arrière,
afin de stabiliser les valeurs des poids.

Apprentissage par rétro-propagation

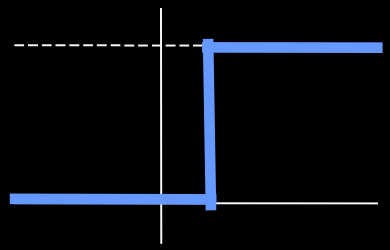
Généralisation, calcul effectué par le réseau :

- présenter les x_i aux neurones d'entrée
- calculer les activités, de couche en couche, jusqu'aux sorties y_j

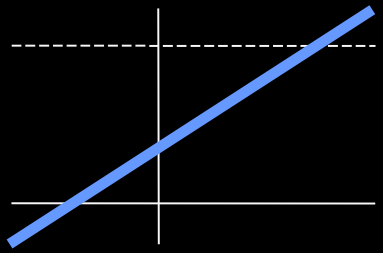
Une seule passe, avant (feedforward)

calcul très rapide, une fois que les poids du réseau ont été fixés

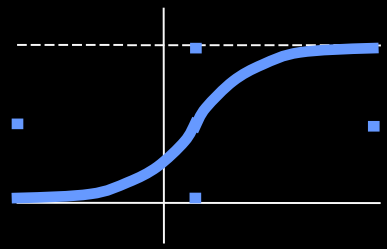
Différents types de neurones



neurone à seuil
(signe ; Heaviside)
$f = H$



neurone linéaire
(fx identité)
$f = I$



neurone sigmoïde
(exp ; tanh)
$f = \sigma$

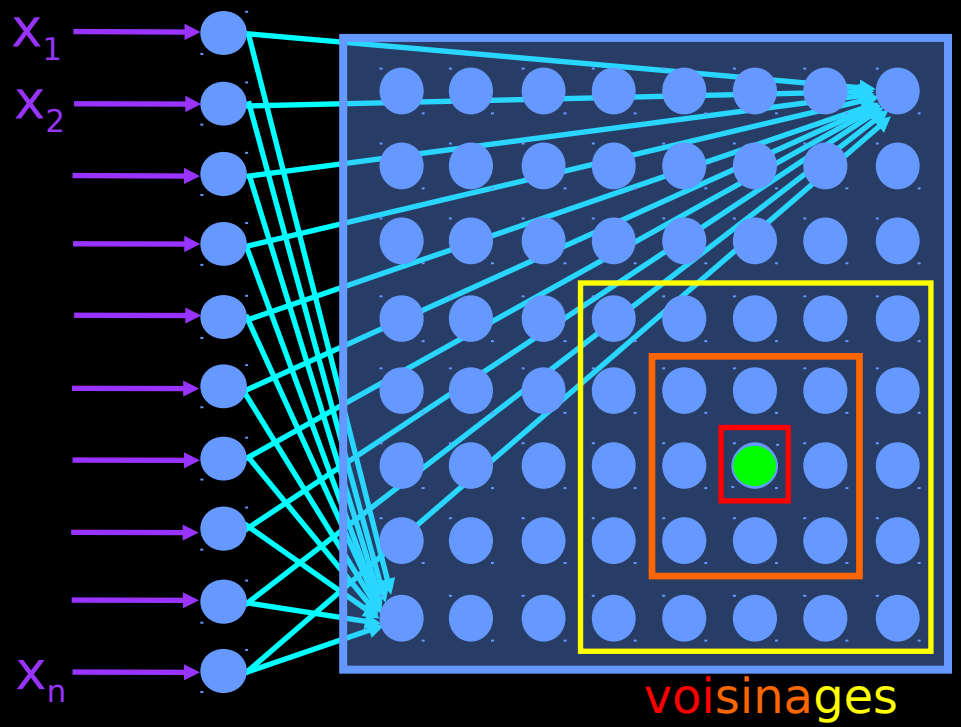
$$y = f\left(\sum_{i=1}^{n} w_i x_i - \theta \right)$$

calculs basés sur
le produit scalaire

Dans certains modèles (cartes de Kohonen, réseaux RBF) :

calculs basés sur la distance $y = g(\| X - W \|)$

Cartes auto-organisatrices (Kohonen, 1984)



⟵ carte 2D

Architecture du réseau :

Chaque neurone d'entrée est connecté à chaque neurone de la carte.

Dynamique du réseau : feedforward

● = neurone vainqueur, i.e. le plus proche de l'entrée

Règle d'auto-organisation

Au temps t=0, initialiser les poids W(0) aléatoirement.
Pour chaque exemple X(t) tiré au temps t d'une base S:

t := t+1

présenter au réseau les entrées $x_i(t)$ de l'exemple,

sélectionner le neurone vainqueur N_{j*} défini par :

$$d_{j*}(t) = \min_j (d_j(t) / 1 \leq j \leq n)$$
$$= \min_j (\Sigma_i (x_i(t) - w_{ij}(t))^2)$$

mettre à jour les poids de N_{j*} et de ses voisins :

si $j \in V_{j*}$, $\forall i$, $w_{ij}(t) := w_{ij}(t\text{-}1) + \alpha(t)(x_i(t)\text{-}w_{ij}(t\text{-}1))$

sinon, $\forall i$, $w_{ij}(t) := w_{ij}(t\text{-}1)$

apprentissage non supervisé, par auto-organisation

Utilisation d'une carte de Kohonen

Pour utiliser la carte, en généralisation,
il faut procéder à un étiquetage de la carte.

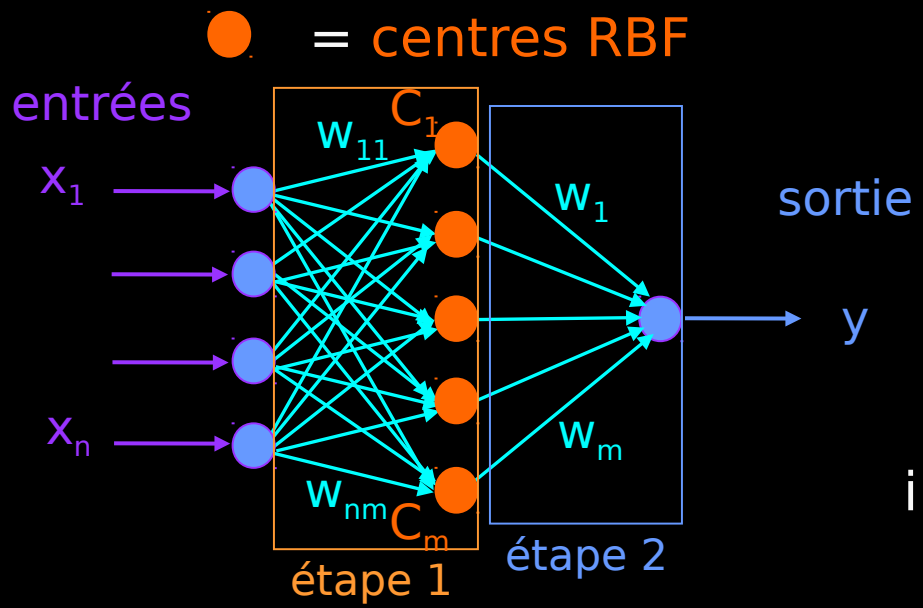
	a	a	c	k	k	k	k	l	l
b	b	c	c	c	f	k	l	l	l
d	b	e	c	f	f	f	f	l	t
d	e	e	e	f	f	g	g	t	t
h	h	e	s	s	g	g	g	t	t
h	h	h	s	s	s	g		q	t
i	h	m	m	s	s	o	q	q	q
i	m	m	m	n	o	o	o	q	r
j	m	m	n	n	n	o	p	r	r
j	j	m	n	n	p	p	p	p	r

Exemple :
reconnaissance des caractères
à moins que....

reconnaissance
des formes
de leurs tracés

Extraction de caractéristiques communes

Les réseaux RBF (Radial Basis Function)



couche cachée :
neurones distance
couche de sortie :
neurone linéaire

Chaque neurone caché est appelé “centre RBF”, il représente un prototype dans l’espace d’entrée X.

φ étant une fonction RBF, par exemple une gaussienne,

$(\forall i \in \{1,\dots,m\})$ centre C_i : $y_i = \varphi\left(\| X - W_i \|\right)$

sortie du réseau, linéaire : $y = \sum_{i=1}^{n} w_i y_i - \theta$

Apprentissage dans les réseaux RBF

Moody et Darken, 1988

On peut apprendre tous les paramètres par une métode de descente en gradient.

ou plutôt...

Apprentissage en deux étapes :

Etape 1 :

recherche des centres RBF par une méthode de clustering
=> couramment, on utilise la méthode des K-means

Etape 2 :

on apprend les poids de sortie par une méthode LMS

Conclusion

Domaines d’application multiples

Les réseaux de neurones artificiels sont devenus des outils de modélisation et de calcul destinés aux ingénieurs :

- Reconnaissance des formes, classification, clustering
- Prédiction, prévision de séries temporelles
- Contrôle, identification de systèmes
- Compression de données
- Bioinformatique
- Robotique

Des exemples :

repérer l’approche d’une tempête
identifier de multiples espèces de fleurs
reconnaître une voix dans un téléphone mobile
distinguer les raisins utilisés pour faire un bon vin
diagnostiquer les conditions d’une conduite dangereus
reconnaître un visage que l’on n’a pas vu récemment

Qualités démontrées par la théorie

Ces RNA sont, d'ailleurs, des outils très performants :

- Calculabilité : plus puissants qu'une machine de Turing
- Complexité : NP-complétude du " loading problem "
- Capacité : MLP et RBF sont approximateurs universels
- Théorie statistique de l'apprentissage, PAC-learning

mais...

les méthodes mises en oeuvre pour les améliorer sont de plus en plus souvent des méthodes déjà connues...

et surtout...

on s'est beaucoup éloigné de l'inspiration biologique !

FIN