Université Abdelmalek Essaâdi

Faculté Polydisciplinaire de Larache

Licence d'Etudes Fondamentales

Sciences Mathématiques, Informatique et Applications

Projet de fin d'études

Etude et réalisation d'un modèle de réseaux de neurones artificiel avec un apprentissage supervise/non supervisé

Réalisée par

ISSARTI Ikrame
KARARA Mohamed
EZOUAGH Mohamed
ZGUINDOU Abdelghafour

Encadrée par

Pr. FAKHOURI Hanane
Pr. EL HAIMOUDI Khatir

Plan de travaille

Part 1: Analyse en composantes principales (ACP)

1. Principe
2. Algorithme

part 2: Les réseaux de neurones artificiels (rna)

1. Définition
2. Propriétés
3. Le modèle biologique
4. Le modèle formel
5. La structure du réseau
6. Apprentissage du réseau de neurones

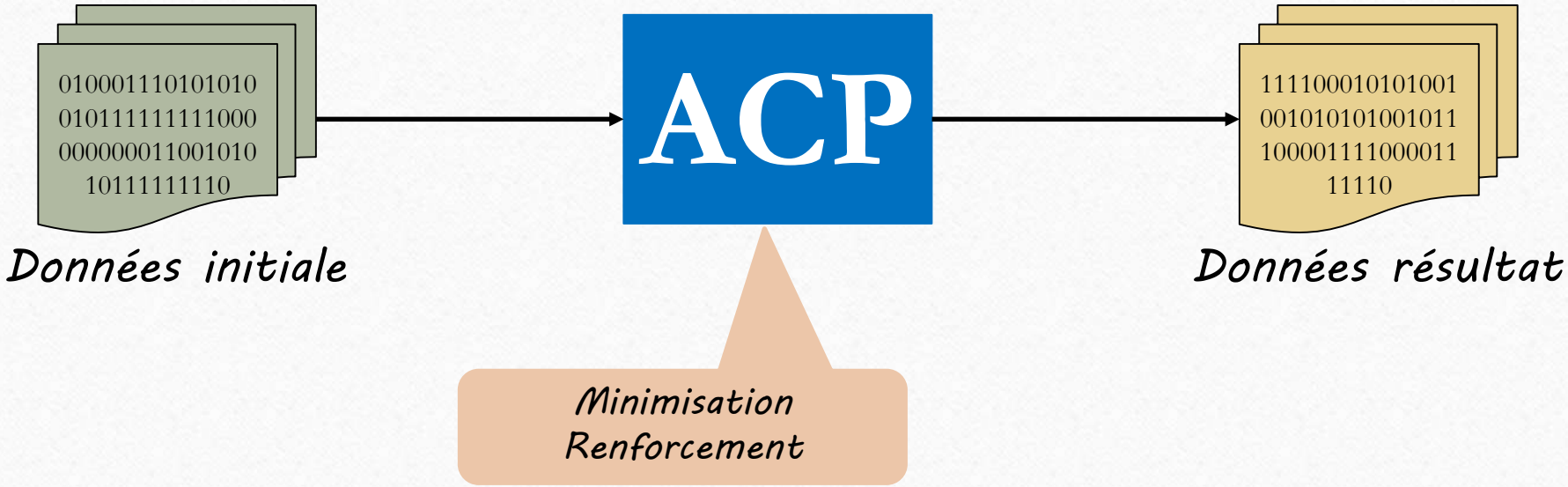
Travail 1 : La carte auto-organisatrice de kohonen (som)

travail 2 : Perceptron multicouche à retropropagation (PMR)

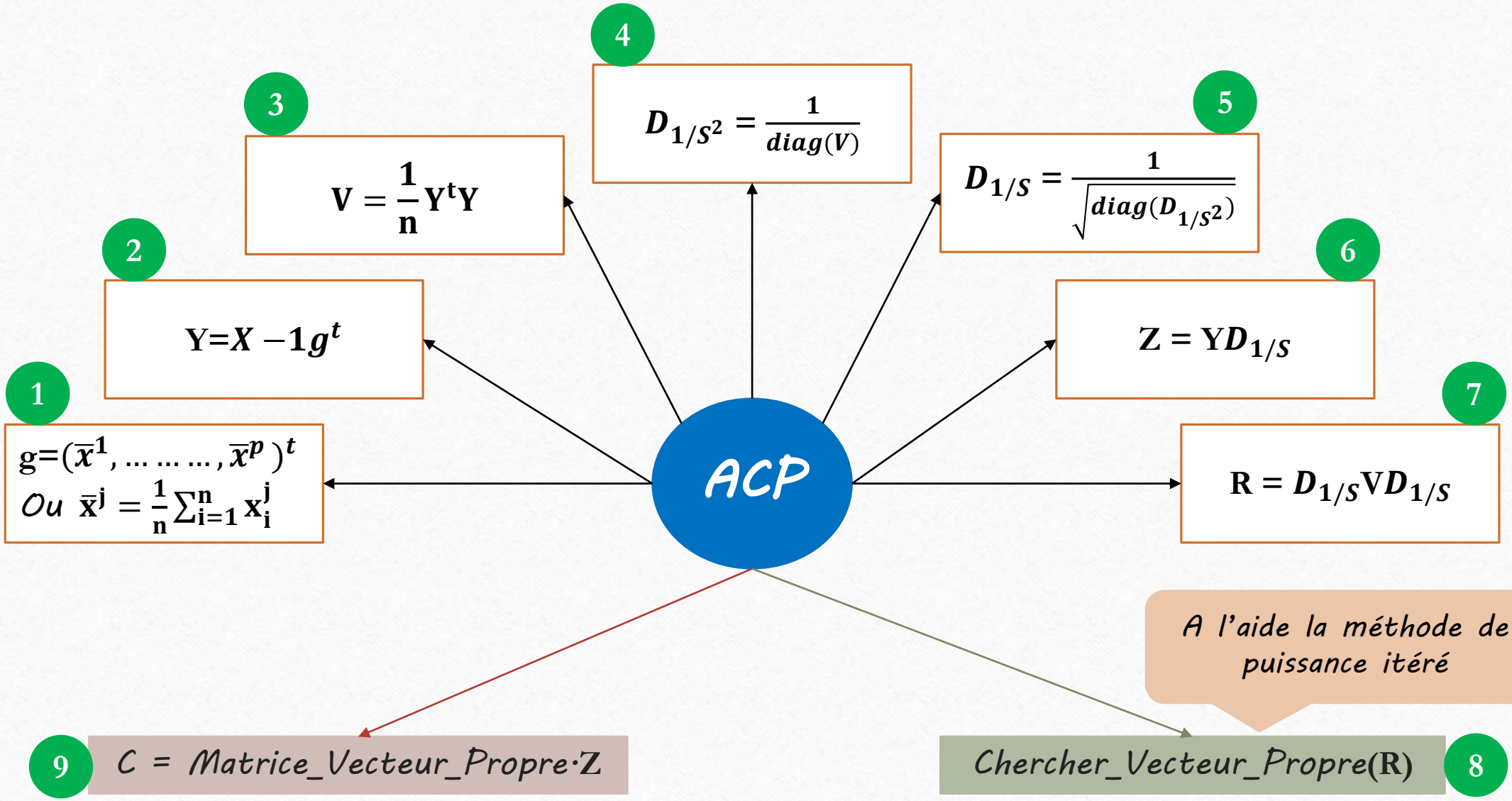
PART 1:
ACP

ACP

1. Principe



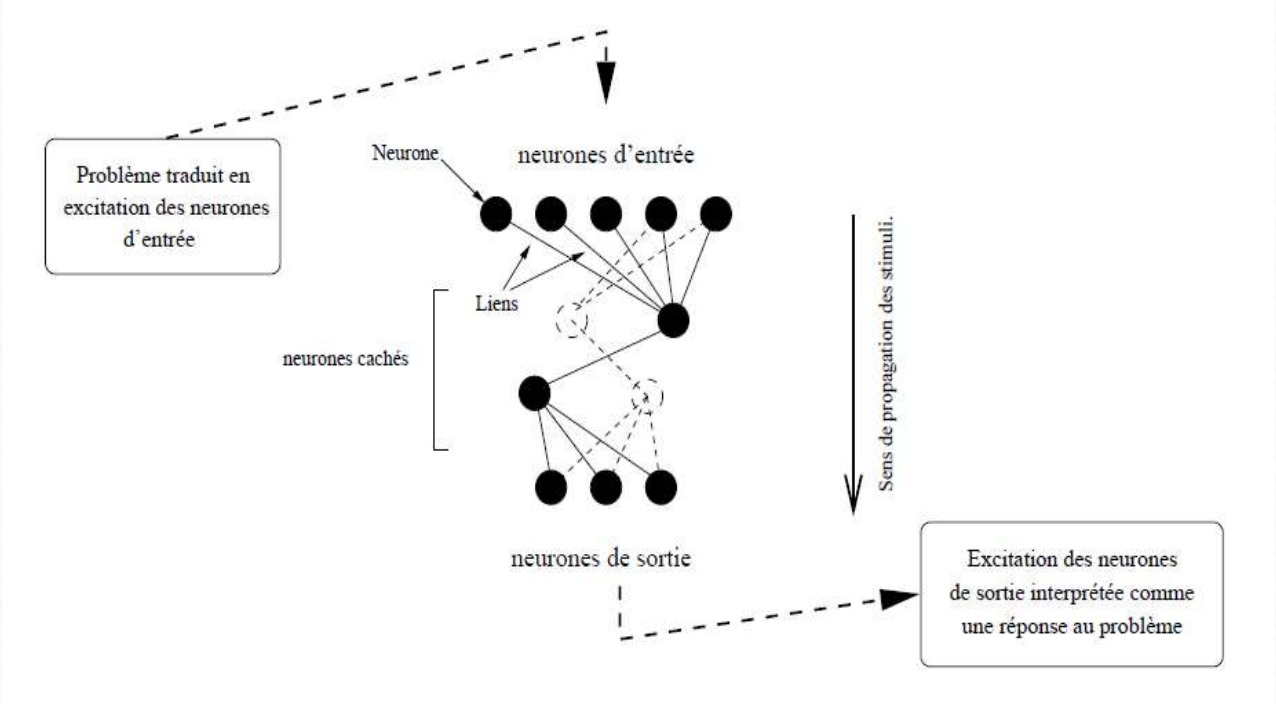
2· Algorithme



PART 2:
RNA

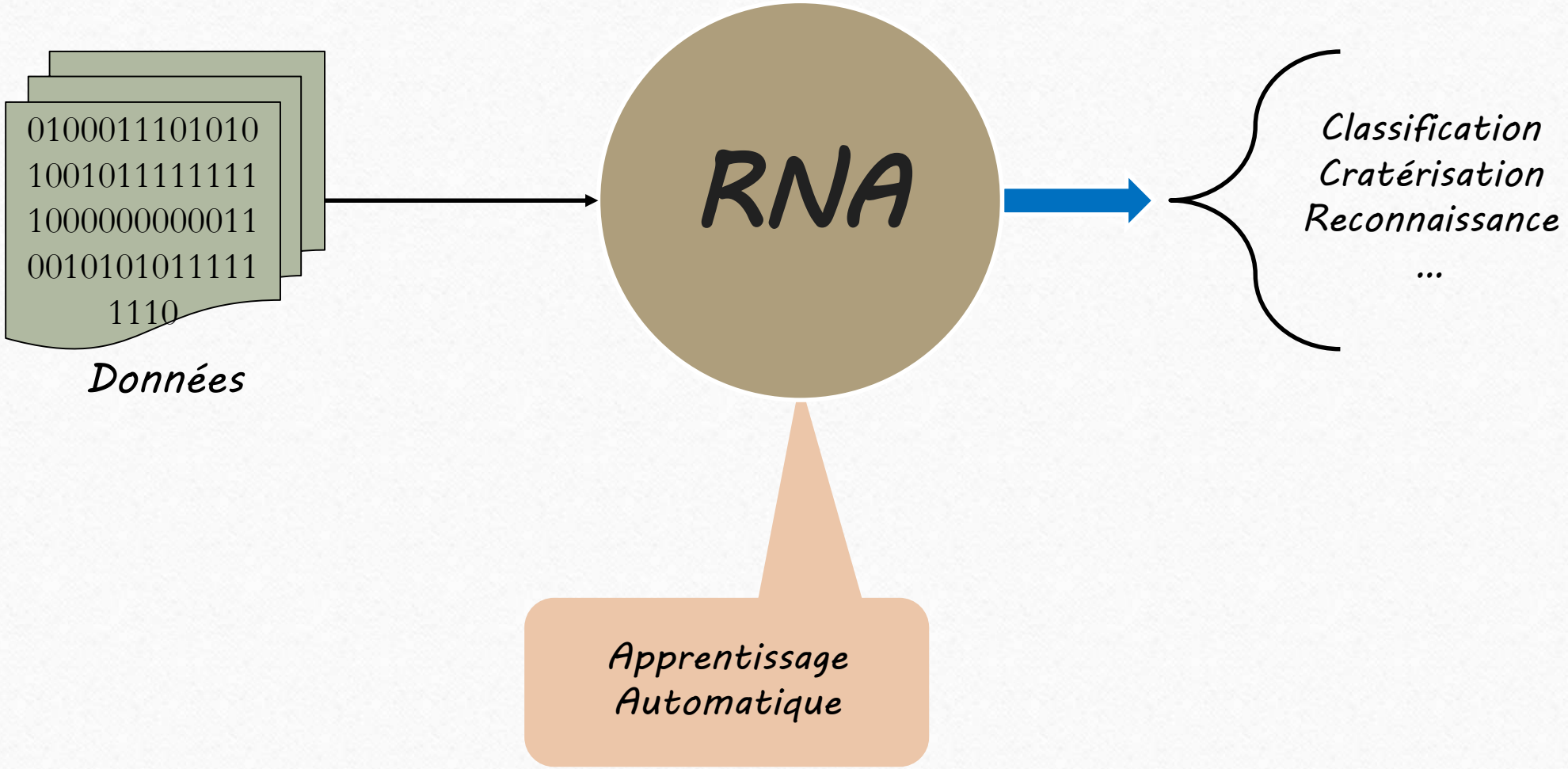
RNA

1. Définition



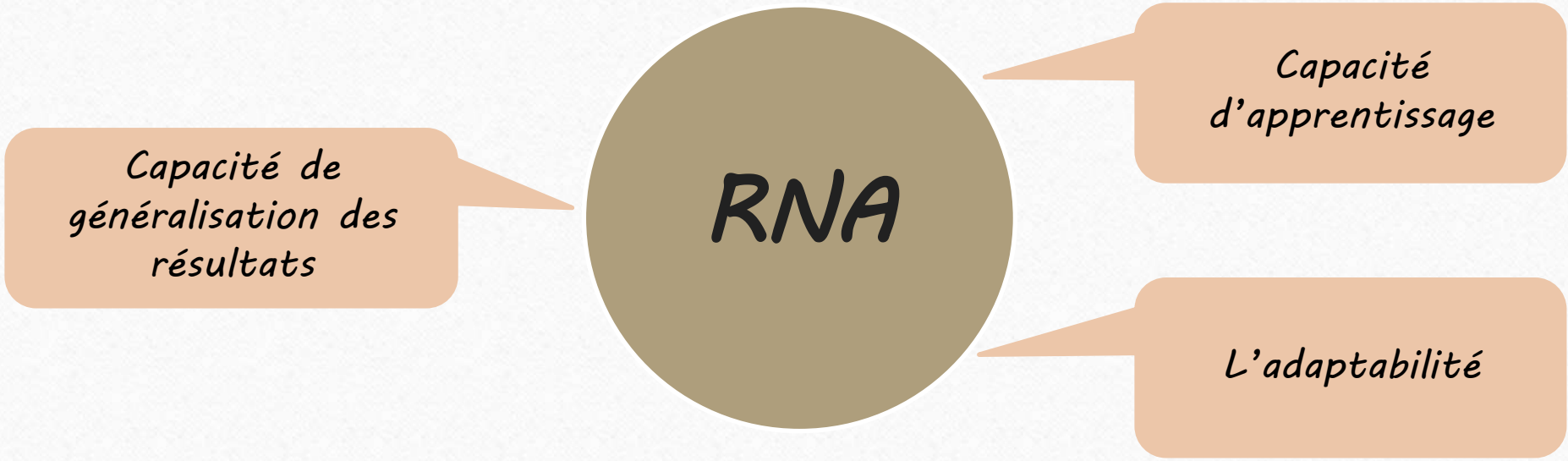
Les réseaux de neurones artificiels sont des réseaux fortement connectés de processeurs élémentaires fonctionnant en parallèle. Chaque processeur élémentaire calcule une sortie unique sur la base des informations qu'il reçoit. Toute structure hiérarchique de réseaux est évidemment un réseau

Principe


0100011101010
1001011111111
1000000000011
0010101011111
1110
Données
RNA
Classification
Cratérisation
Reconnaissance
...
Apprentissage
Automatique

RNA

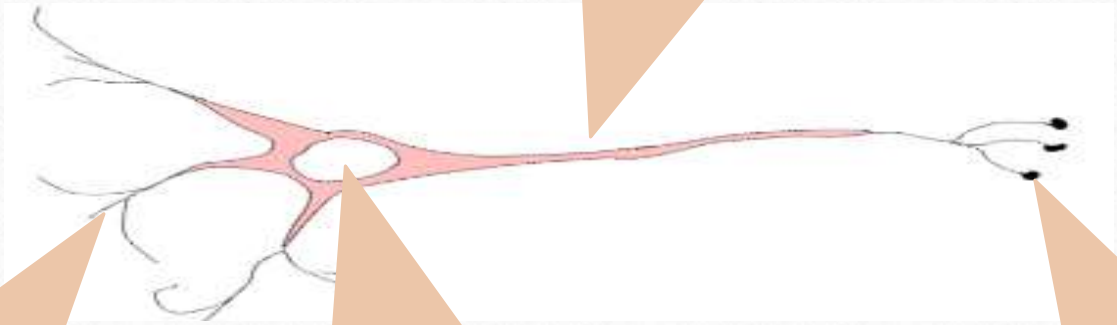
2. Propriétés



RNA

3· Le modèle biologique

l'axone transporte l'influx nerveux vers les synapses·



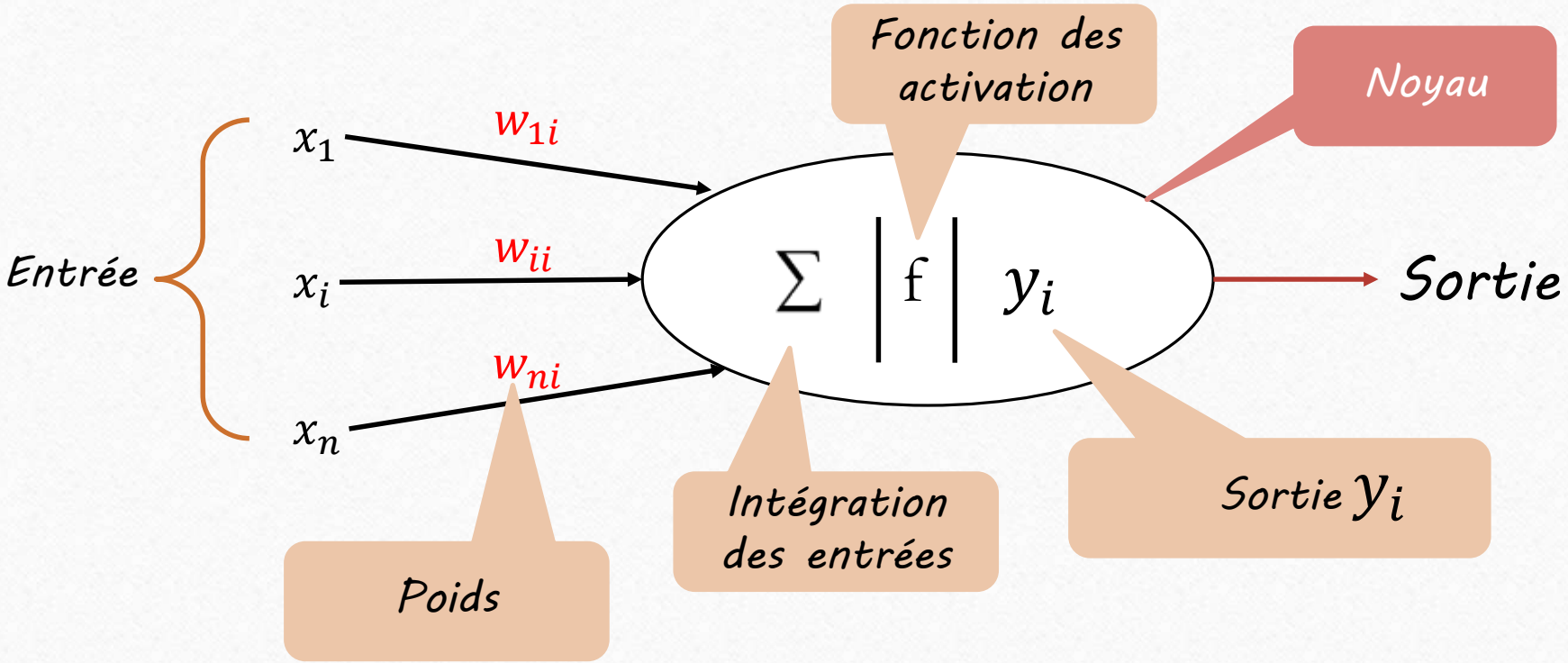
les dendrites, selon leur longueur et leur perméabilité, affectent la quantité d'influx nerveux qui se rend au noyau·

le noyau est le centre des réactions électrochimiques· Si les stimulations externes sont suffisantes, le noyau provoque l'envoi d'un influx nerveux électrique à travers l'axone·

Les synapses transmettent l'influx nerveux provenant de l'axone vers d'autres cellules à partir de neurotransmetteurs inhibiteurs ou excitateurs·

RNA

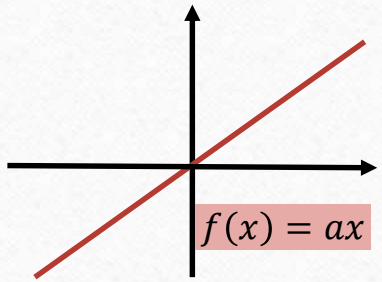
4. Le modèle formel



RNA

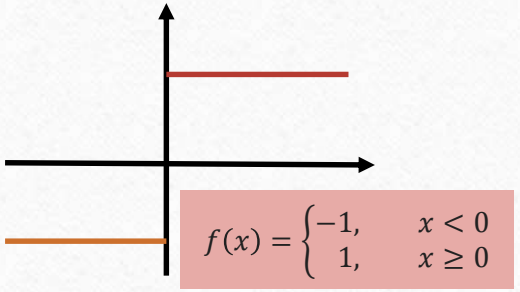
4. Le modèle formel

Les fonctions d'activation



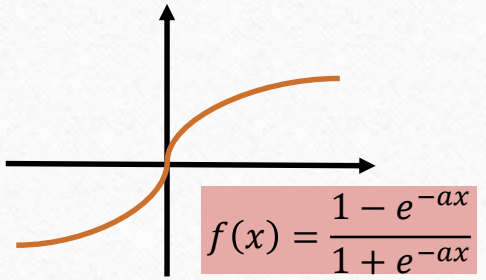
Linéaire

La plus part des phénomènes n'ont pas un comportement linéaire



Tout au rien = Seuil dur

Décision certaine même pour Les données proches de la surface De décision

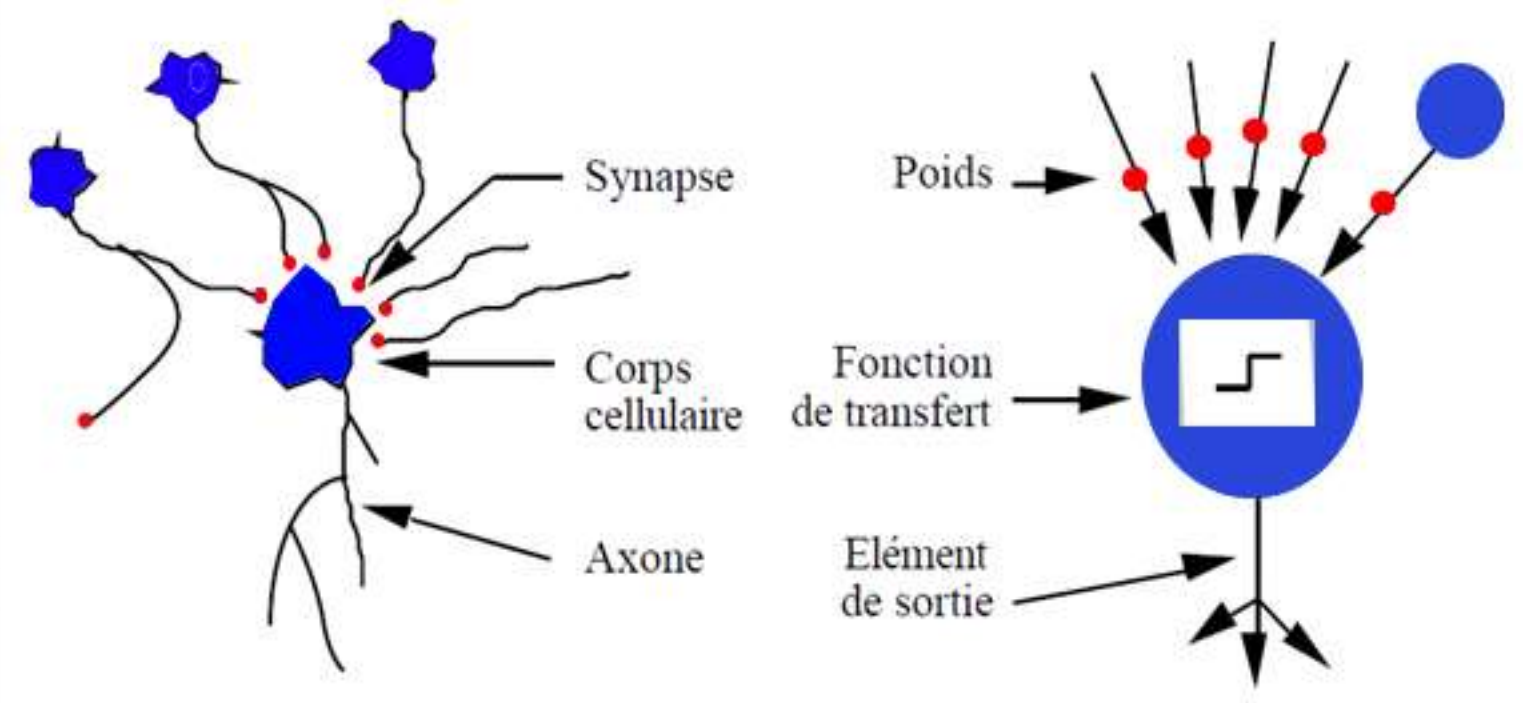


Fonction logistique = Seuil doux

Décision avec une probabilité

RNA

4. Le modèle formel / Biologique



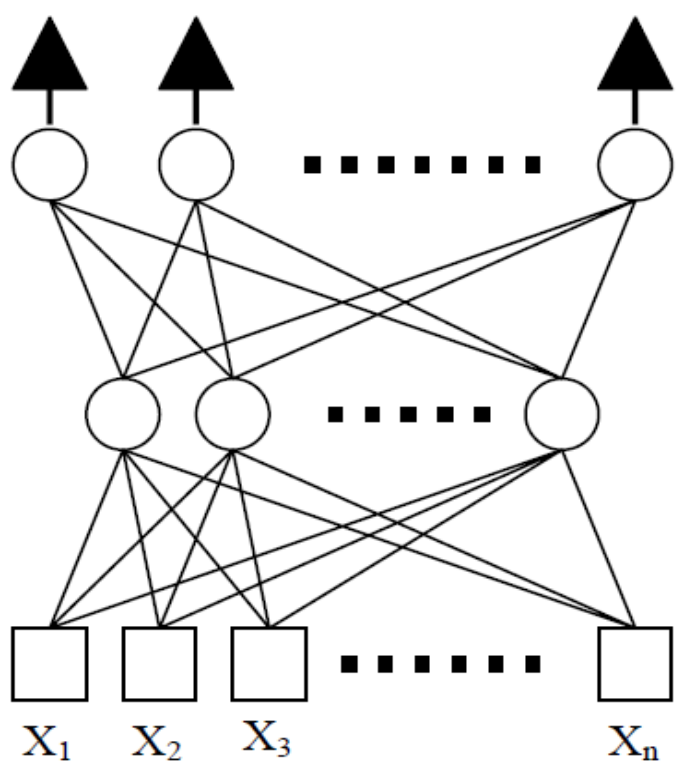
RNA

5· structure du réseau

RNA

5. La structure du réseau

Non Bouclé

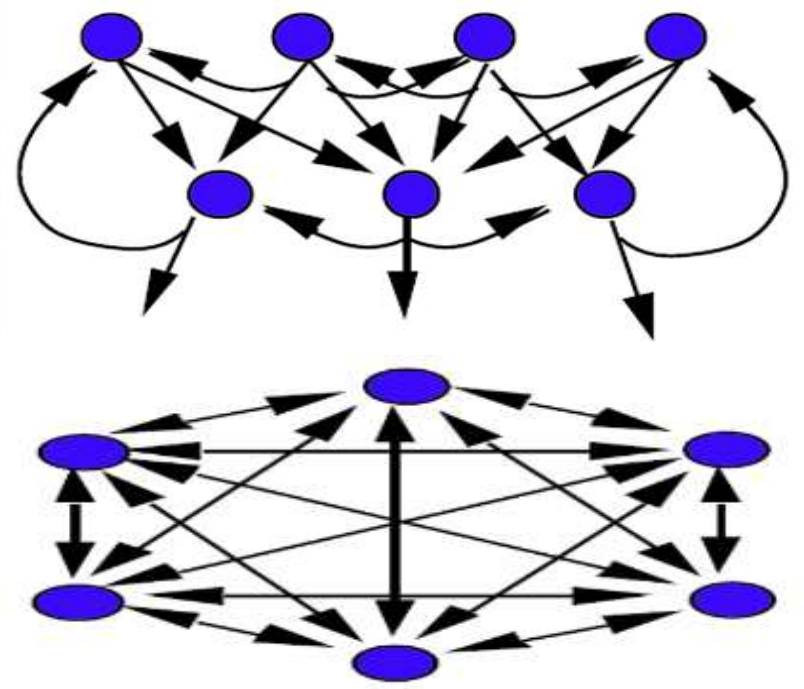


Un réseau de neurones non bouclé réalise une (ou plusieurs) fonction algébrique de ses entrées par composition des fonctions réalisées par chacun de ses neurones.

RNA

5. La structure du réseau

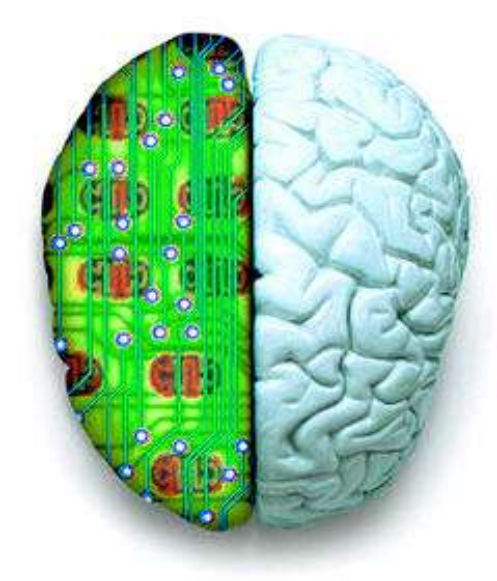
Bouclé



le graphe des connexions est cyclique : lorsqu'on se déplace dans le réseau en suivant le sens des connexions, il est possible de trouver au moins un chemin qui revient à son point de départ (un tel chemin est désigné sous le terme de « cycle »). Les connexions récurrentes ramènent l'information en arrière par rapport au sens de propagation défini dans un réseau multicouche. Ces connexions sont le plus souvent locales.

RNA

6. Apprentissage

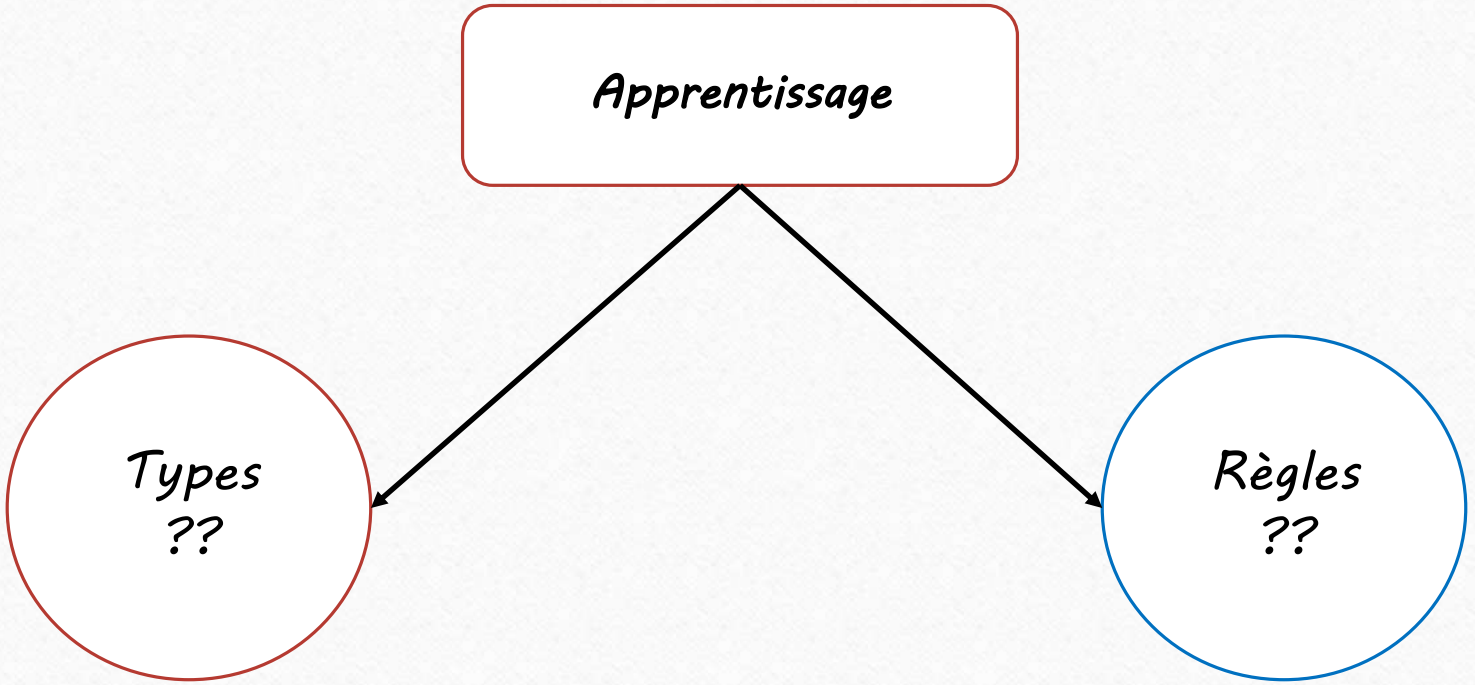


Une procédure adaptative par laquelle les connexions des neurones sont ajustées face à une source d'information.

Dans la majorité des algorithmes actuels, les variables modifiées pendant l'apprentissage sont les poids des connexions. L'apprentissage est la modification des poids du réseau dans l'optique d'accorder la réponse du réseau aux exemples et à l'expérience.

RNA

6. Apprentissage



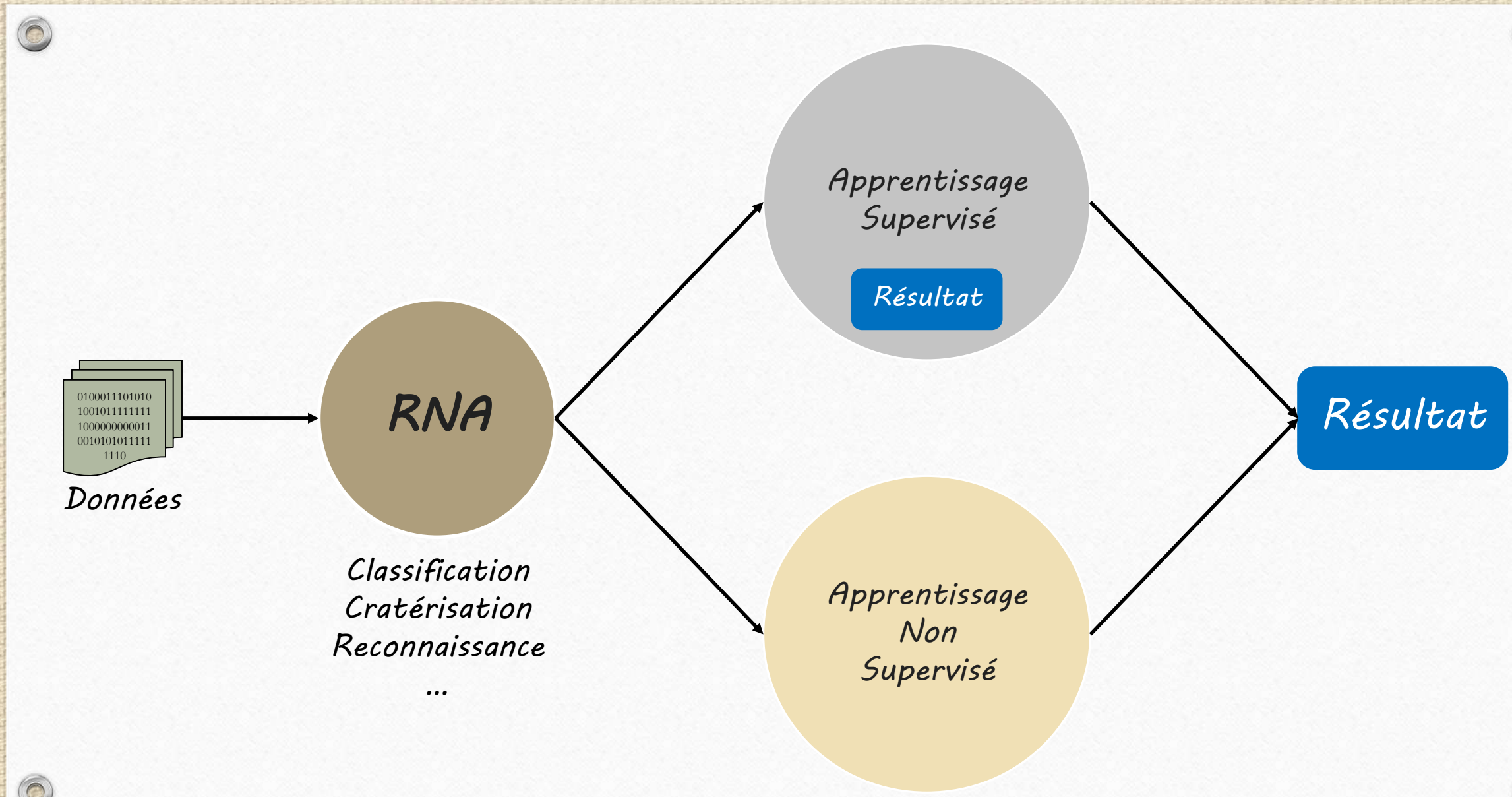
RNA

6. Apprentissage

Types d'apprentissage

Un apprentissage est dit supervisé lorsque l'on force le réseau à converger vers un état final précis, en même temps qu'on lui présente un motif. Ce genre d'apprentissage est réalisé à l'aide d'une base d'apprentissage, constituée de plusieurs exemples de type entrées-sorties (les entrées du réseau et les sorties désirées ou encore les solutions souhaitées pour l'ensemble des sorties du réseau).

L'apprentissage non supervisé consiste à ajuster les poids à partir d'un seul ensemble d'apprentissage formé uniquement de données. Aucun résultat désiré n'est fourni au réseau.
L'apprentissage ici consiste à détecter les similarités et les différences dans l'ensemble d'apprentissage.


Apprentissage
Supervisé
Résultat
RNA
0100011101010
1001011111111
1000000000011
0010101011111
1110
Données
Classification
Cratérisation
Reconnaissance
...
Apprentissage
Non
Supervisé
Résultat

RNA

5. Apprentissage

Règles d'apprentissage

Les règles d'apprentissage indiquent la quantité de modification à appliquer à chaque poids, en fonction des exemples d'entrée (et des sorties désirées associées, dans le cas de l'apprentissage supervisé) :

$$W_{ij}(t+1) = W_{ij}(t) + \Delta W_{ij}$$

Où t désigne un instant avant la modification des poids, et t+1 un instant après.
Il existe un grand nombre de règles d'apprentissage différentes. Les plus répandues sont les suivantes :

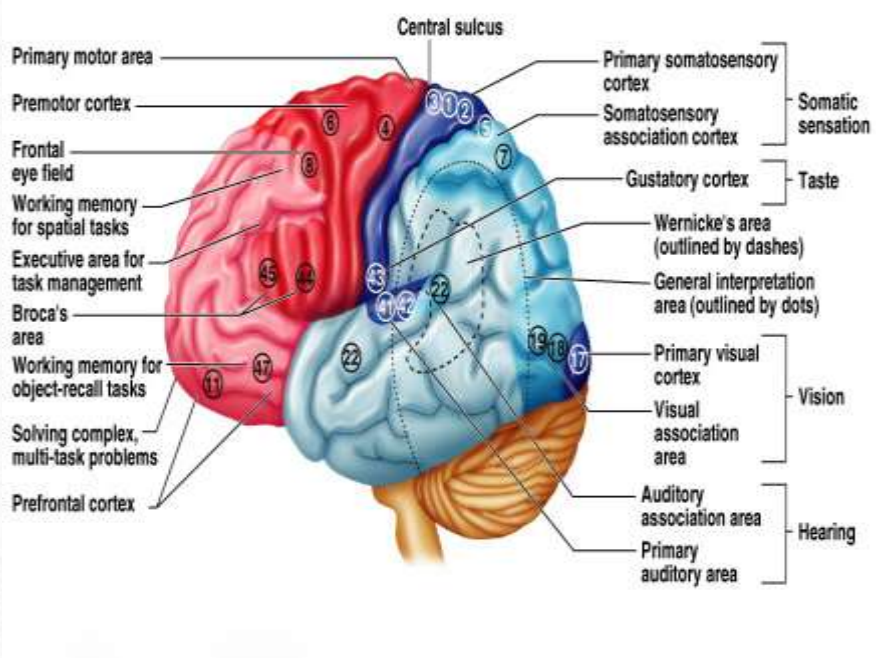
- ✓ Règle de Hebb $\Delta W_{ij} = \alpha x_i y_j$
- ✓ Règle d'apprentissage compétitif $\Delta W_{ij} = \alpha(x_i - W_{ij})y_j$
- ✓ Règle d'apprentissage supervisé $\Delta W_{ij} = \alpha x_i(y_j - \theta_j)$
- ✓ Règle de la retropropagation du gradient $\Delta W_{ij} = \alpha \gamma_j y_i$

TRAVAIL 1:

SOM

SOM

1. Principe



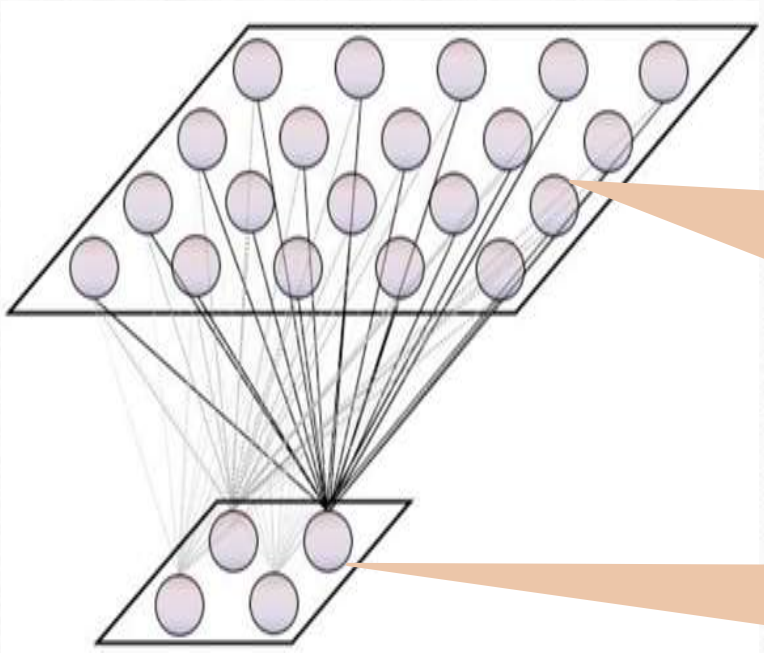
SOMs fournissent un moyen de représenter des données multidimensionnelles dans des espaces de dimensions beaucoup plus faibles - généralement une ou deux dimensions.

Cette technique crée un réseau qui stocke des informations de telle façon que toutes les relations topologiques au sein de l'ensemble d'apprentissage sont maintenues.

Ces structures intelligentes de représentation de données sont inspirées, comme beaucoup d'autres créations de l'intelligence artificielle, par la biologie ;

SOM

2. Architecture



d'une couche de sortie également appelée couche de compétition où les neurones entrent en compétition. Les valeurs provenant des neurones d'entrées sont transmises à tous les neurones de la couche de compétition, en même temps. Pour approcher des valeurs passées par les neurones d'entrées.

- d'une couche d'entrée où tout motif à classer est représenté par un vecteur multidimensionnel qualifié de vecteur d'entrée. A chaque motif est affecté un neurone d'entrée.

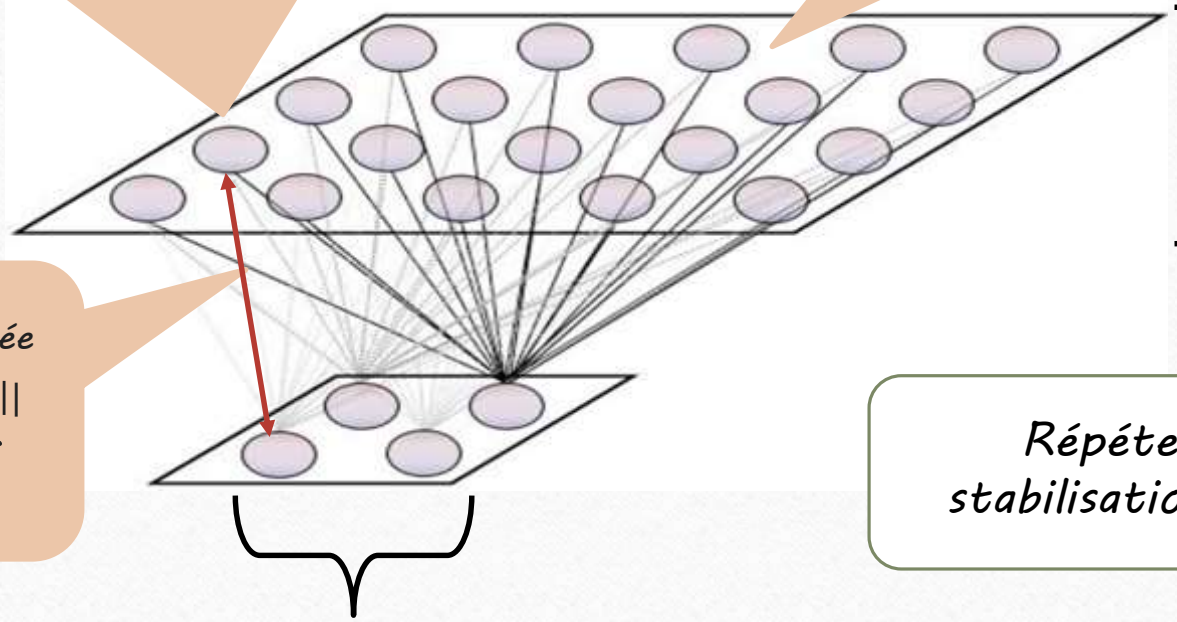
3· Algorithme d'apprentissage

Initialisation des paramètres:

- ✓ Poids d'entrée (petites valeurs aléatoire)
- ✓ Forme et taille du voisinage
- ✓ Taux d'apprentissage

Mise à jour des poids du neurone j et de ses voisins, le vecteur poids du neurone j' est modifié par

$$W_{j'}(t+1) = W_{j'}(t)\alpha(t)h_{jj'}(x - W_{j'}(t))$$



La carte SOM (Couche Sortie)

Déterminer le neurone vainqueur de chaque entrée

$$\left|\left|W_j - x\right|\right| = \min_n ||W_n - x||$$

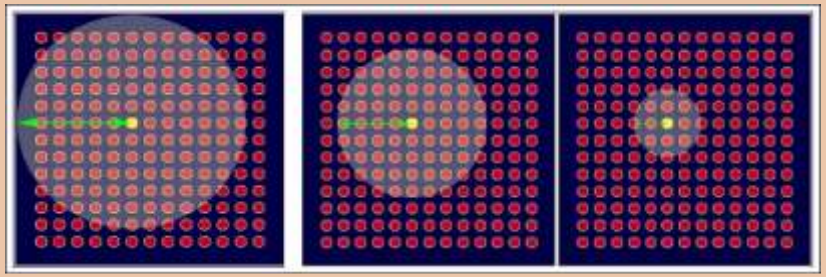
j : indice du neurone vainqueur
n : le nombre de neurones

Répéter jusqu'à stabilisation du réseau

Couche d'entrées

Réglage des paramètres

Diminution de la taille du voisinage

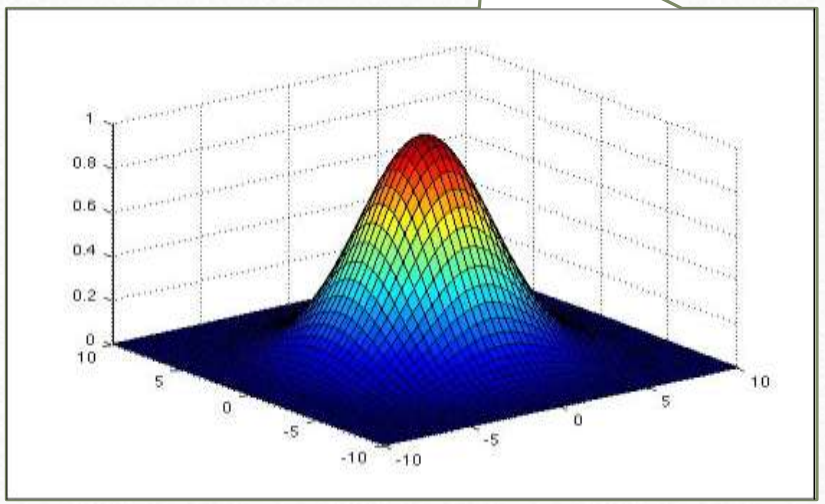


$$\sigma(t) = \exp(-t\frac{\ln(GridSize)}{MaxIteration})$$

Forme du voisinage

$$h_{jj'} = \exp(\frac{-d_{jj'}^2}{2\sigma(t)^2})$$

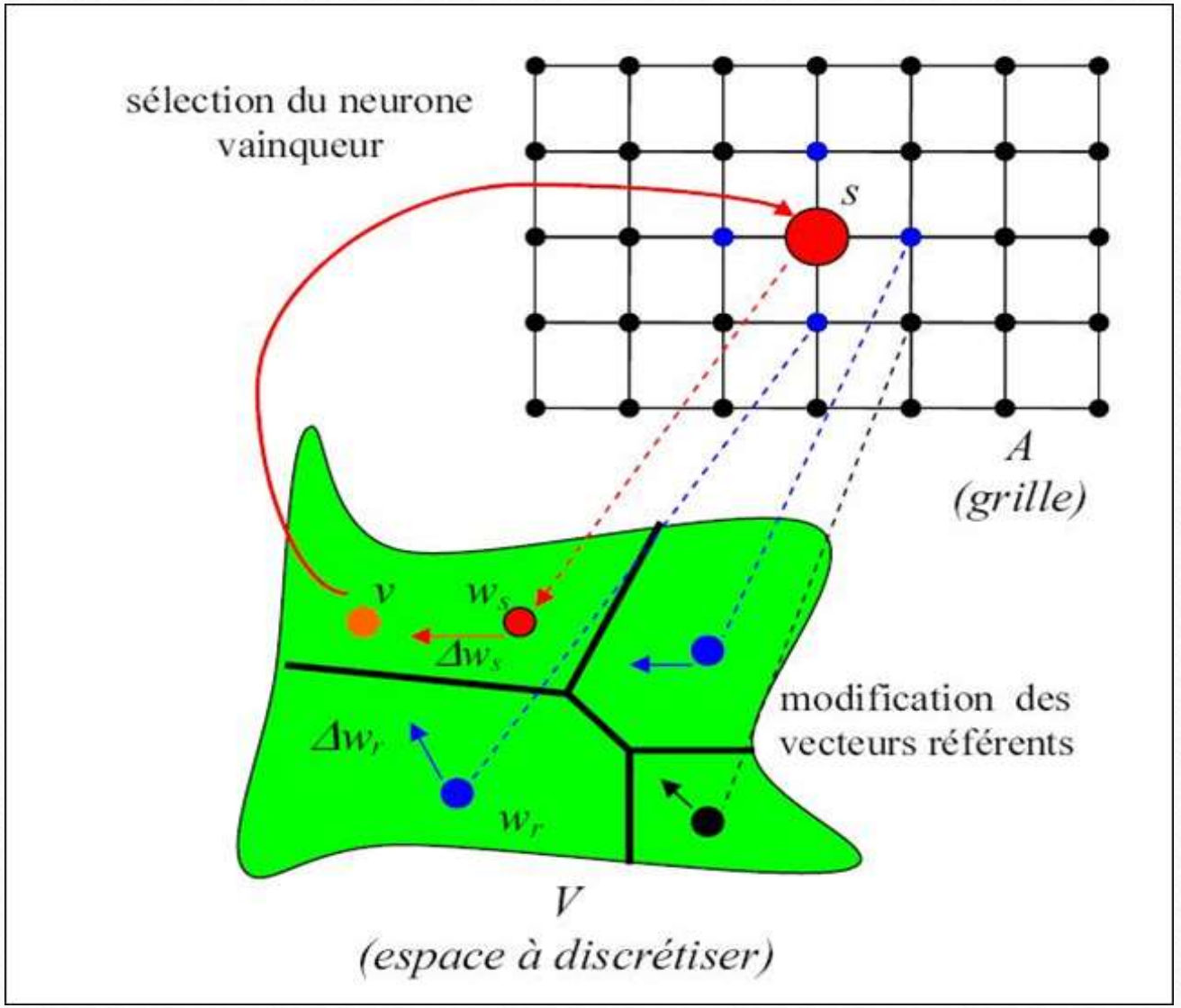
J est l'indice du neurone vainqueur
J' est l'indice d'un neurone voisin
$d_{jj'}$ Est la distance entre le neurone vainqueur et les autre neurones



Diminution du taux d'apprentissage

Pour assurer la convergence de l'apprentissage on peut faire varier le taux d'apprentissage en fonction du nombre d'itérations· La valeur initiale peut être grande, par exemple égale à 1·

$$\alpha(t) = \alpha_0 * e^{-t\frac{1}{MaxIteration}}$$


sélection du neurone
vainqueur
s
A
(grille)
v
w_s
Δw_s
modification des
vecteurs référents
Δw_r
w_r
V
(espace à discrétiser)

SOM

4. Propriétés

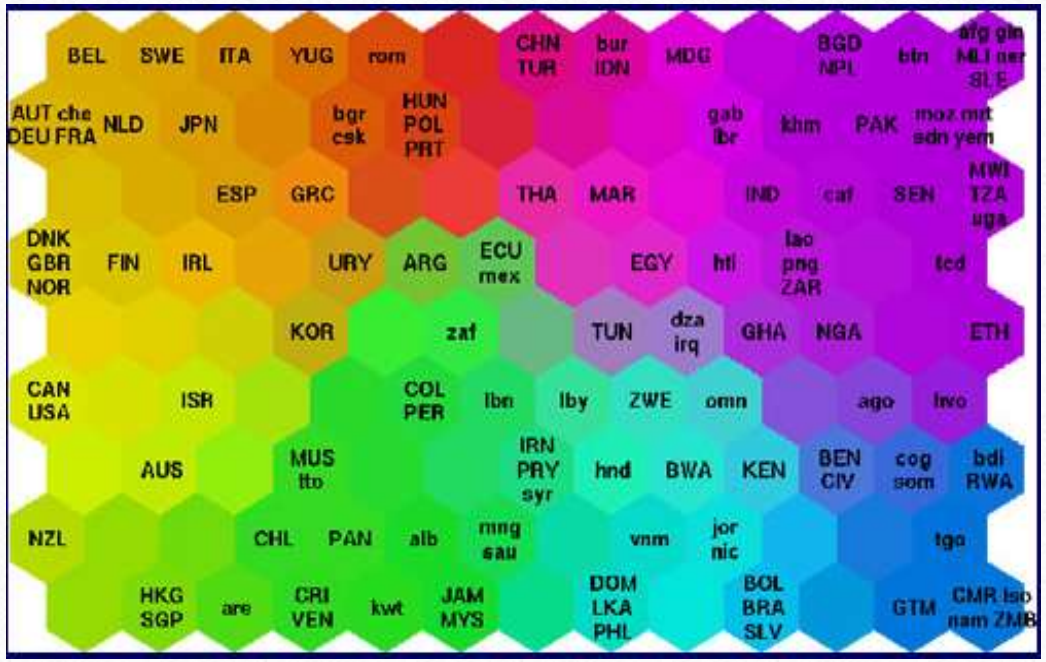
- ✓ Similitude des densités dans l'espace d'entrée :
  La carte reflète la distribution des points dans l'espace d'entrée.
- ✓ Préservation des relations topologiques
- ✓ Les ancêtres des cartes auto-organisatrices, les algorithmes comme "k-moyennes", réalisent la discrétisation de l'espace d'entrée en ne modifiant à chaque cycle d'adaptation qu'un seul vecteur référent. Leur processus d'apprentissage est donc très long. L'algorithme de Kohonen profite des relations de voisinage dans la grille pour réaliser une discrétisation dans un temps très court.

SOM

5. Exemples d'application

La classification

World Poverty Map



SOMs sont couramment utilisés comme aides à la visualisation. Ils peuvent le rendre facile pour nous les humains de voir les relations entre les grandes quantités de données.

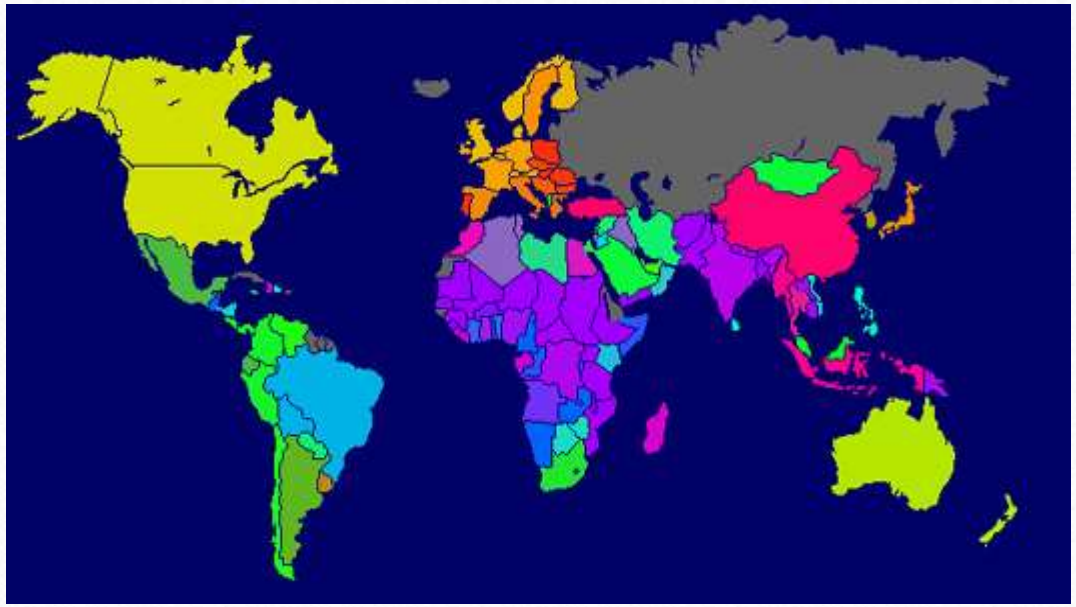
Un SOM a été utilisé pour classer les données statistiques décrivant divers facteurs de qualité de vie tels que l'état de santé, la nutrition, les services d'enseignement, etc.

SOM

6. Exemples d'application

La classification

World Poverty Map

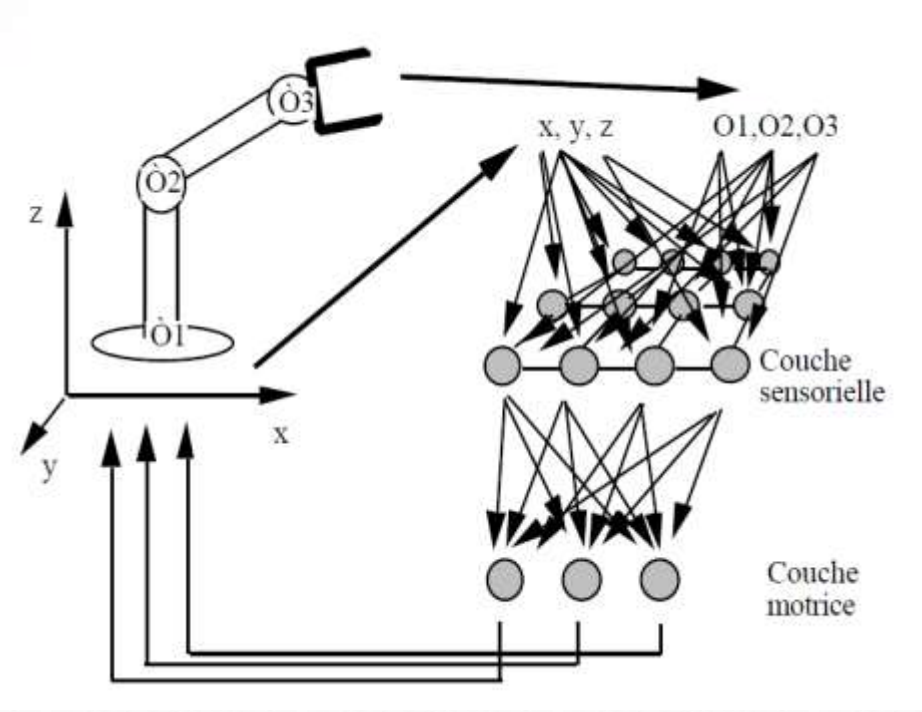


Cette information de couleur peut alors être tracée sur une carte du monde comme ceci (Image) Cela rend très facile pour nous de comprendre les données de la pauvreté

SOM

6. Exemples d'application

Application à la robotique



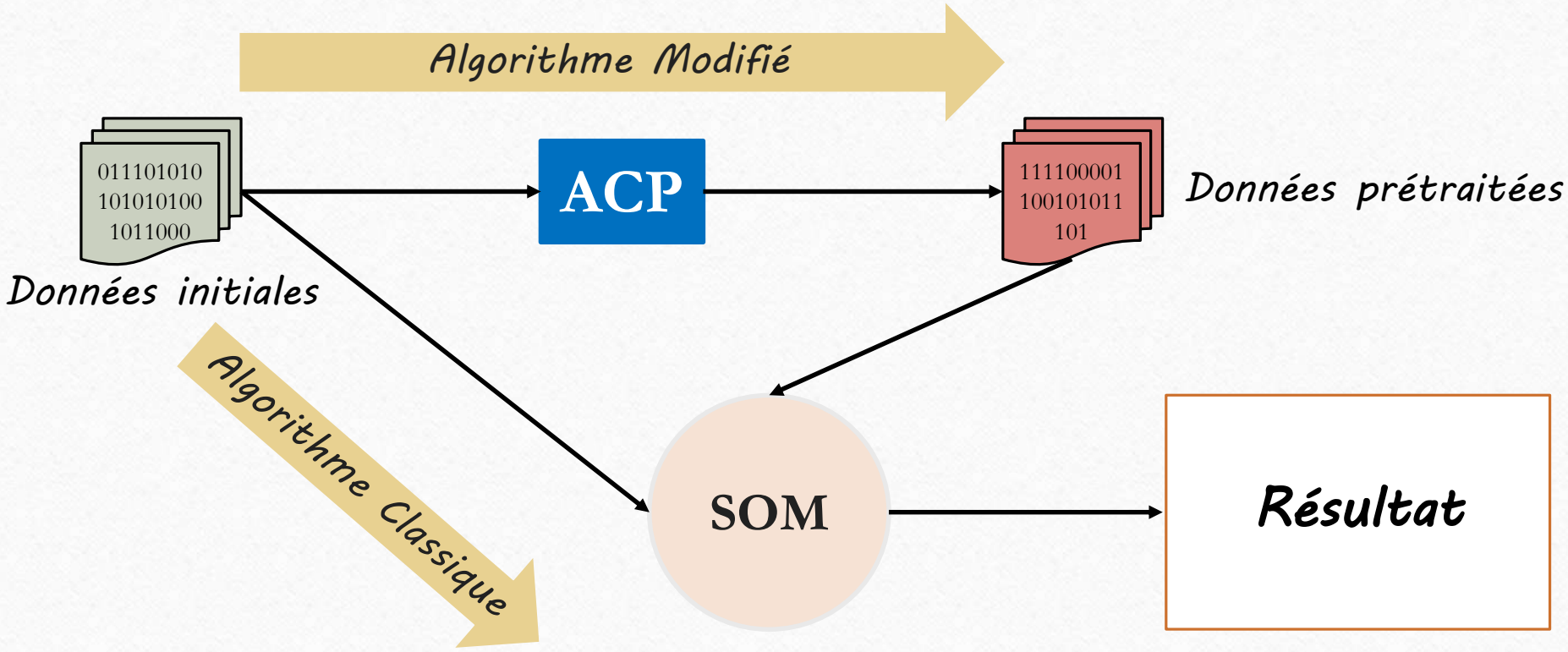
On utilise la capacité de représentation spatiale de la carte auto-organisatrice pour piloter un bras de robot.

Il s'agit de mettre en correspondance l'espace cartésien dans lequel travaille l'opérateur humain avec l'espace du robot (coordonnées de chacun de ses axes en valeurs angulaires).

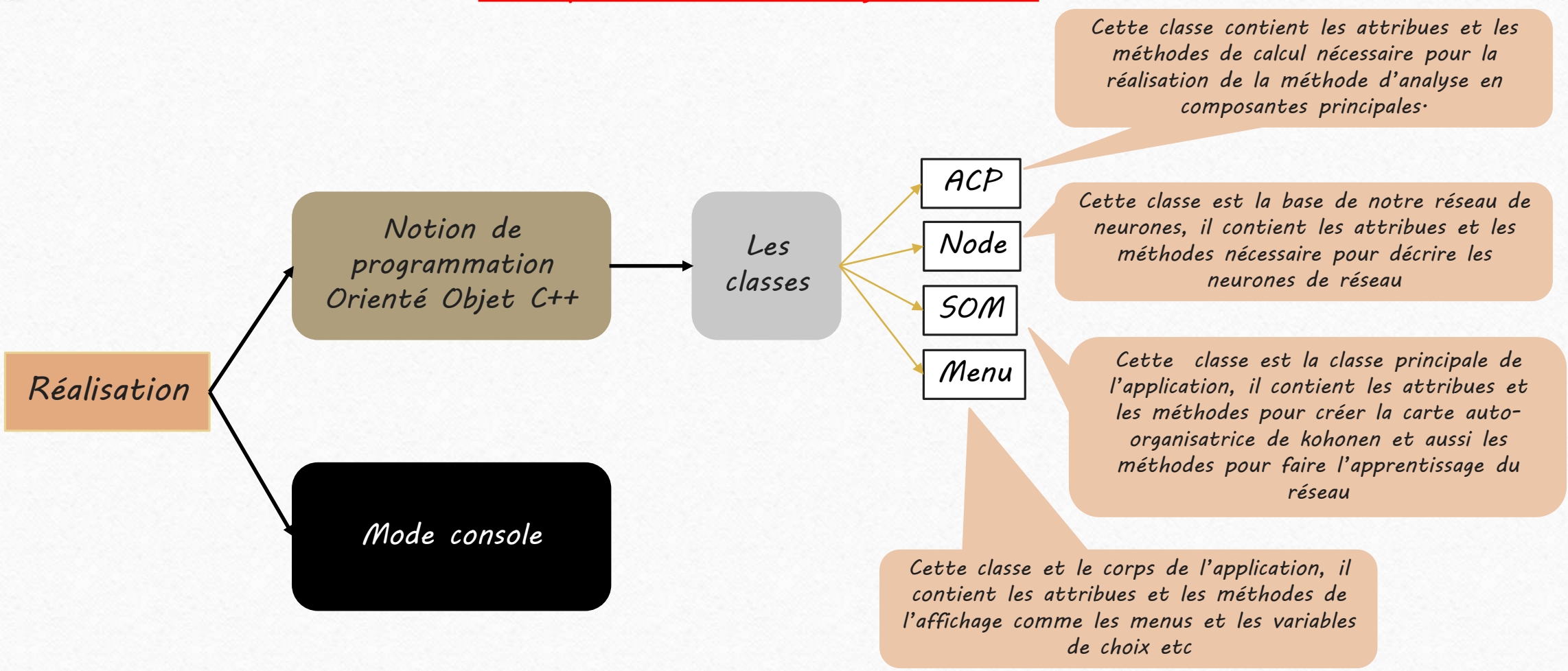
SOM

1. Réalisation

Les structures fonctionnelles élaborées à la base des deux algorithmes classiques et modifiés



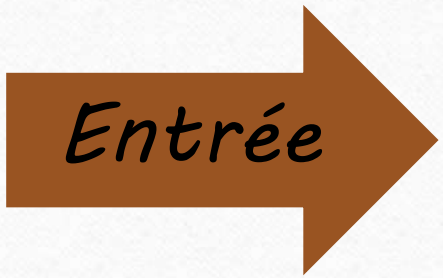
Description de modèle logiciel réalisé



2. Les testes

Test de classification des alphabets

A	A’	T	T’	C	C’	F	F’	H	H’
0 1 0	0 1 0	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	1 1 1	1 0 1	1 0 0
1 1 1	0 1 1	0 1 0	0 0 0	1 0 0	1 0 0	1 1 1	1 1 0	1 1 1	1 1 1
1 0 1	1 0 1	0 1 0	0 1 0	1 1 1	1 1 0	1 0 0	1 0 0	1 0 1	1 0 1



A	010111101
A’	010011101
T	111010010
T’	111000010
C	111100111
C’	111100110
F	111111100
F’	111110100
H	101111101
H’	100111101

Création de la carte

```
ENTRER LA DIMENSION DE VOTRE CARTE SOM : 6

ENTRER LE NOMBRE MAXIMALE D'ITERATIONS : 1000

ENTRER L'EREUR D'APPRENTISAGE MINIMALE : 0.00001

ENTRER LE TAUX D'APPRENTISAGE INITIALE : 0.5
```

L'initialisation des vecteurs des poids

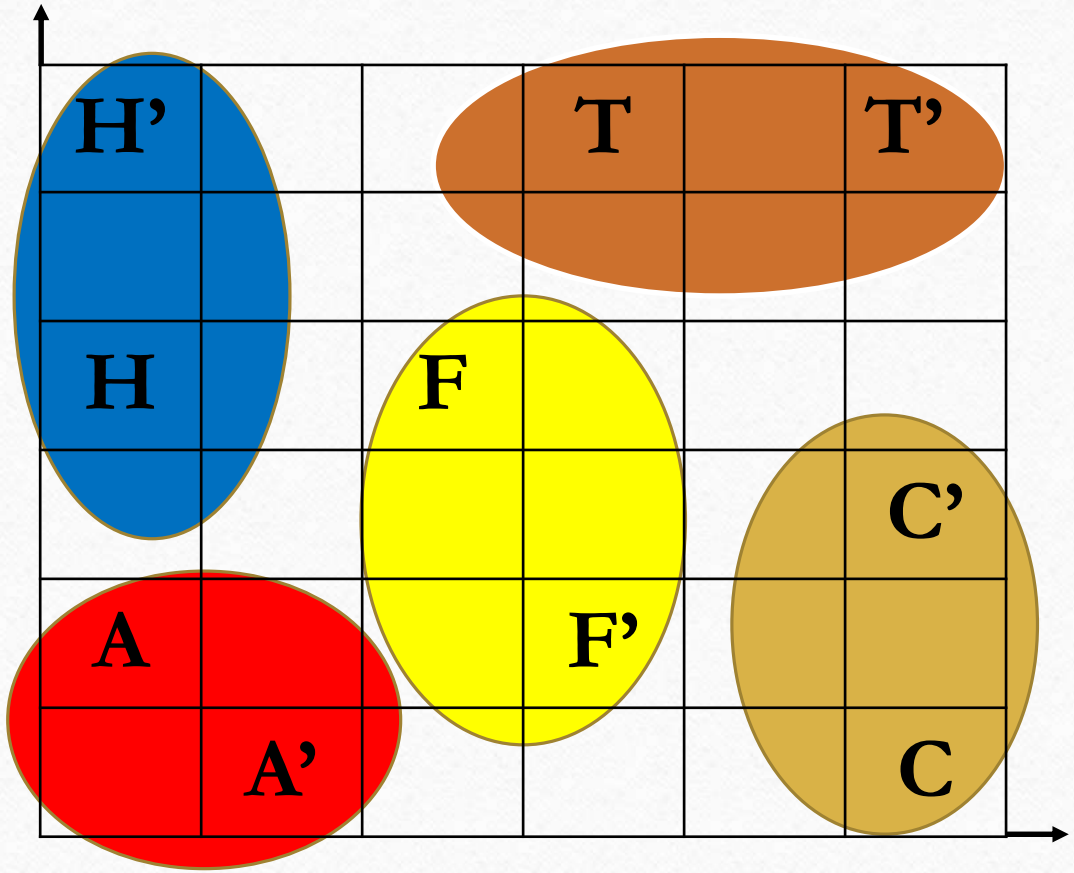
```
Node 1 ->(0,0) :    0.00588989     0.707855     0.100616     0.623047      0.86322     0.491486     0.747314     0.496887     0.380096
Node 2 ->(0,1) :      0.785339     0.552795     0.357086     0.955688     0.630829     0.176575     0.374237     0.131622     0.743256
Node 3 ->(0,2) :      0.951691     0.611969     0.027832     0.329834    0.0559082     0.639191     0.131622     0.847046     0.864288
Node 4 ->(0,3) :      0.596863     0.721619     0.853943     0.014679     0.126465     0.707886     0.617126      0.21756    0.0659485
Node 5 ->(0,4) :      0.168915     0.624084     0.340973     0.319397     0.367554      0.66098     0.802368     0.806854      0.52652
Node 6 ->(0,5) :      0.611084     0.798157     0.900574     0.144806     0.630157     0.402405     0.253693     0.136536     0.855164
Node 7 ->(1,0) :     0.0661621     0.427795     0.573334     0.302277     0.548035     0.225555      0.31134     0.110626     0.808014
Node 8 ->(1,1) :      0.134705     0.284241     0.788086     0.895203     0.789612     0.743774     0.615204     0.361115     0.856628
Node 9 ->(1,2) :      0.228485     0.863556     0.229431     0.249542     0.542389     0.984802    0.0538025    0.0814209     0.524658
Node 10->(1,3) :      0.426788    0.0946655     0.258789      0.89151     0.232758     0.146545     0.125092      0.93161    0.0801086
Node 11->(1,4) :     0.0470886    0.0587158     0.336395     0.914673      0.39859      0.43277     0.946136     0.837158      0.53421
Node 12->(1,5) :      0.842072     0.693512     0.397675     0.259155    0.0043335     0.525574     0.954773     0.398682     0.241089
Node 13->(2,0) :      0.585541     0.255127      0.68399     0.945251     0.435486     0.890198    0.00717163     0.940948     0.601532
Node 14->(2,1) :      0.786133      0.57666     0.142426     0.222321     0.382996    0.00427246     0.417908    0.0822449     0.659912
Node 15->(2,2) :      0.855072    0.0648499     0.811035     0.662048     0.691467     0.802673     0.530121     0.685608     0.142761
Node 16->(2,3) :      0.689484     0.727875      0.77771    0.0310669     0.868652     0.644501     0.706543    0.0854492     0.551971
Node 17->(2,4) :      0.947876    0.0587769     0.274963     0.145172      0.98175     0.619965     0.292236     0.922455     0.367523
Node 18->(2,5) :      0.694519     0.218628     0.155914      0.24054     0.521423     0.902252     0.106415     0.902618     0.441711
Node 19->(3,0) :    0.0801086     0.782074     0.171783     0.974365     0.775848     0.870361     0.210632     0.456604    0.00375366
Node 20->(3,1) :      0.75061     0.114044     0.404694     0.311127     0.992584    0.0385742     0.252075      0.18927     0.247681
Node 21->(3,2) :     0.153503       0.6203     0.885345     0.939056     0.195648     0.779633     0.645538     0.656738     0.909119
Node 22->(3,3) :     0.455444     0.921265     0.664154     0.150757     0.636322     0.569519     0.422028     0.942993     0.540558
Node 23->(3,4) :     0.578552     0.536438      0.25528     0.396942      0.35025    0.0366211     0.795227     0.196503     0.070282
Node 24->(3,5) :     0.389404     0.590729    0.0709229     0.197662     0.155884     0.644318     0.454315     0.604279     0.697327
Node 25->(4,0) :     0.441284     0.684448     0.396515     0.835693     0.592194     0.199585     0.949432     0.875977     0.391693
Node 26->(4,1) :     0.987427     0.185303     0.895691     0.574432     0.442047     0.627319     0.708496    0.0496216     0.285553
Node 27->(4,2) :     0.260193     0.407623     0.895294     0.710175     0.728424     0.896027     0.393555       0.3974     0.903839
Node 28->(4,3) :     0.308594     0.388092     0.570557     0.353546     0.733673     0.745331     0.736359      0.73941     0.139313
Node 29->(4,4) :     0.200165     0.272583      0.68042     0.911102     0.367828     0.517639     0.109375      0.90741     0.203064
Node 30->(4,5) :     0.517944     0.654816     0.438507     0.687622    0.0904236    0.0795593    0.0757751    0.0278015     0.355072
Node 31->(5,0) :      0.30722     0.697266      0.14267     0.394836    0.0678406      0.67572     0.724915       0.1987        0.694
Node 32->(5,1) :     0.615692     0.654877     0.543976    0.0982971     0.545349    0.0496216     0.974548      0.46402      0.96994
Node 33->(5,2) :     0.727844     0.530945     0.679901      0.69278     0.372101     0.388611    0.0499573     0.809753     0.169556
Node 34->(5,3) :    0.0608215     0.310547     0.784454     0.212463     0.321899      0.48465   0.00915527      0.43985     0.507843
Node 35->(5,4) :     0.605927     0.758514     0.401062     0.349792     0.842621     0.942291     0.623291     0.997253    0.0534668
Node 36->(5,5) :     0.562836     0.863525     0.386749     0.306396      0.28421    0.0267029     0.612549     0.391174    0.0186157
```

Les vecteurs des poids après l'apprentissage

Node 1 ->(0,0) :	3.34547e-029	1	3.55714e-029	0.468023	1	1	1	1	1.11031e-029	1
Node 2 ->(0,1) :	7.04168e-026	1	7.03913e-026	0.999784	1	1	1	1	7.85072e-030	1
Node 3 ->(0,2) :	0.528009	0.471991	0.528009	1	1	1	1	1	7.75934e-031	1
Node 4 ->(0,3) :	1	2.51216e-009	1	1	1	1	1	1	4.46811e-032	1
Node 5 ->(0,4) :	1	1.63694e-017	0.468533	1	1	1	1	1	2.99141e-033	1
Node 6 ->(0,5) :	1	2.96318e-024	4.27199e-009	1	1	1	1	1	2.16917e-027	1
Node 7 ->(1,0) :	8.66641e-026	1	8.66874e-026	0.000162223	1	1	1	1	5.06363e-030	1
Node 8 ->(1,1) :	1.2007e-013	1	1.2007e-013	0.468023	1	1	1	1	7.86384e-039	1
Node 9 ->(1,2) :	0.608828	0.669762	0.608828	1	1	1	1	1	5.2561e-036	0.721409
Node 10->(1,3) :	1	0.468974	1	1	1	1	1	1	3.36644e-038	0.531026
Node 11->(1,4) :	1	0.278326	0.608248	1	1	1	1	1	2.47198e-032	0.721674
Node 12->(1,5) :	1	9.64847e-008	4.58197e-007	1	1	1	1	1	9.64797e-008	1
Node 13->(2,0) :	0.00227034	1	0.00227034	0.00227034	1	0.99773	1	1	1.24867e-027	0.99773
Node 14->(2,1) :	0.997587	1	0.997587	0.997587	1	0.00241334	1	1	1.53749e-035	0.00241334
Node 15->(2,2) :	1	1	1	1	1	0.997601	1	1	1.77093e-033	7.23503e-014
Node 16->(2,3) :	1	1	1	1	1	1	1	1	3.23424e-026	1.03421e-020
Node 17->(2,4) :	1	1	1	0.997616	1	0.997616	0.997616	0.00238399	1.3048e-010	
Node 18->(2,5) :	1	1	1	4.09848e-007	1	4.09848e-007	4.09848e-007	1	2.28738e-007	
Node 19->(3,0) :	1	1	1	1	1	1.42292e-007	1	1.80196e-007	3.22488e-007	
Node 20->(3,1) :	1	1	1	1	1	2.32204e-011	1	2.16692e-011	1.66432e-011	
Node 21->(3,2) :	1	1	1	1	1	0.00193058	1	8.47612e-011	8.30127e-029	
Node 22->(3,3) :	1	1	1	1	1	1	1	1.38879e-007	1.89313e-029	
Node 23->(3,4) :	1	1	1	0.0022984	1	0.0022984	0.0022984	0.997702	4.69867e-031	
Node 24->(3,5) :	1	1	1	1.96359e-012	1	1.96359e-012	1.96359e-012	1	4.29727e-025	
Node 25->(4,0) :	1	1	1	1	0.00234671	4.42998e-028	1	0.997653	0.997653	
Node 26->(4,1) :	1	1	1	1	0.996961	1.01151e-031	1	0.00303883	0.00180416	
Node 27->(4,2) :	1	1	1	1	0.00237267	6.04902e-014	1	0.997627	3.02863e-014	
Node 28->(4,3) :	1	1	1	0.999999	0.000337489	0.000336533	0.999999	0.999663	2.16536e-028	
Node 29->(4,4) :	1	1	1	2.01633e-010	0.456638	6.56875e-011	2.01633e-010	1	1.09388e-031	
Node 30->(4,5) :	1	1	1	3.78659e-027	0.468151	3.78202e-027	3.78199e-027	1	1.92073e-028	
Node 31->(5,0) :	1	1	1	1	2.46821e-021	4.76784e-025	1	1	1	
Node 32->(5,1) :	1	1	1	1	1.46573e-013	4.66677e-036	1	1	0.468279	
Node 33->(5,2) :	1	1	1	1	2.12983e-012	1.6627e-029	1	1	2.00842e-009	
Node 34->(5,3) :	1	1	1	1	4.12148e-010	2.88242e-029	1	1	3.02036e-029	
Node 35->(5,4) :	1	1	1	2.96525e-007	4.10426e-010	2.86142e-026	2.96525e-007	1	1.6236e-027	
Node 36->(5,5) :	1	1	1	4.96681e-025	1.92441e-009	4.32865e-026	9.92967e-025	1	3.01768e-026	

Résultat finale

```
ENTREE [0]  -> 0  1  0  1  1  1  1  0  1
WINNER [0,1]-> 7.04168e-026  1  7.03913e-026  0.999784  1  1  1  7.85072e-030  1

ENTREE [1]  -> 0  1  0  0  1  1  1  0  1
WINNER [1,0]-> 8.66641e-026  1  8.66874e-026  0.000162223  1  1  1  5.06363e-030  1

ENTREE [2]  -> 1  1  1  0  1  0  0  1  0
WINNER [3,5]-> 1  1  1  1.96359e-012  1  1.96359e-012  1.96359e-012  1  4.29727e-025

ENTREE [3]  -> 1  1  1  0  0  0  0  1  0
WINNER [5,5]-> 1  1  1  4.96681e-025  1.92441e-009  4.32865e-026  9.92967e-025  1  3.01768e-026

ENTREE [4]  -> 1  1  1  1  0  0  1  1  1
WINNER [5,0]-> 1  1  1  1  2.46821e-021  4.76784e-025  1  1  1

ENTREE [5]  -> 1  1  1  1  0  0  1  1  0
WINNER [5,2]-> 1  1  1  1  2.12983e-012  1.6627e-029  1  1  2.00842e-009

ENTREE [6]  -> 1  1  1  1  1  1  1  0  0
WINNER [2,3]-> 1  1  1  1  1  1  1  3.23424e-026  1.03421e-020

ENTREE [7]  -> 1  1  1  1  1  0  1  0  0
WINNER [3,1]-> 1  1  1  1  1  2.32204e-011  1  2.16692e-011  1.66432e-011

ENTREE [8]  -> 1  0  1  1  1  1  1  0  1
WINNER [0,3]-> 1  2.51216e-009  1  1  1  1  1  4.46811e-032  1

ENTREE [9]  -> 1  0  0  1  1  1  1  0  1
WINNER [0,5]-> 1  2.96318e-024  4.27199e-009  1  1  1  1  2.16917e-027  1
```

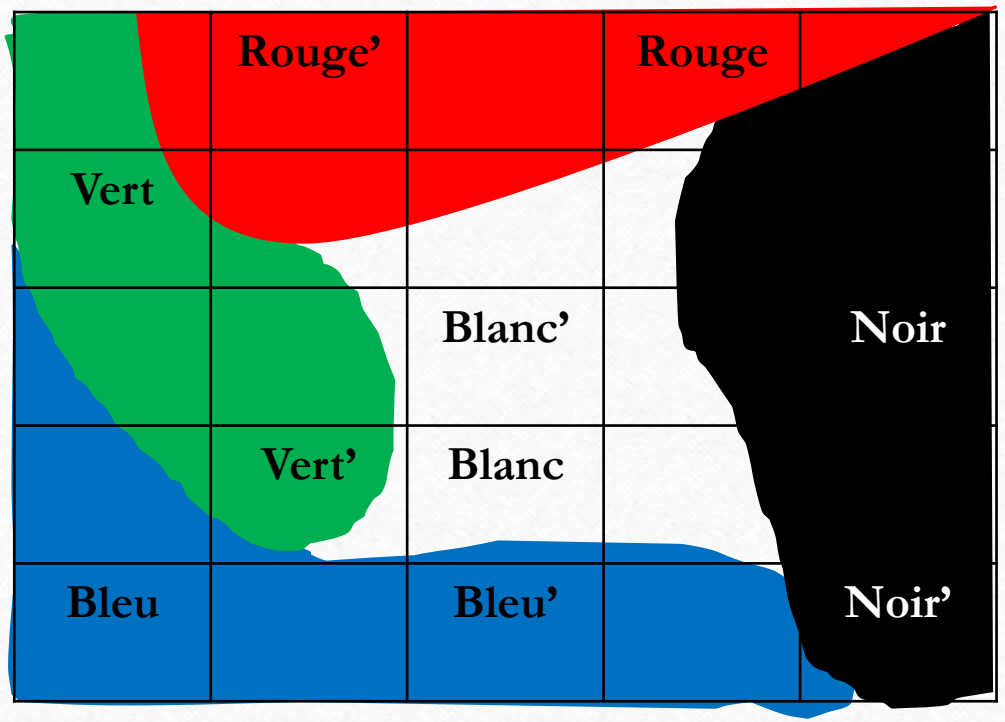

H’
T
T’
H
F
C’
A
F’
A’
C

Test de classification des colores

Entrée

Noir	1	1	1
Noir'	1	21	31
Rouge	255	0	0
Rouge'	255	0	131
Bleu	0	0	255
Bleu'	99	140	255
Vert	0	255	0
Vert'	160	255	181
Blanc	255	255	255
Blanc'	247	255	239

Code RGB



Capitales européens / température

	Janvier	Février	Mars	Avril	Mai	Juin	Juillet	Août	Septembr	Octobre	Novembre	Décembre
Athènes	9.1	9.7	11.7	15.4	20.1	24.5	27.4	27.2	23.8	19.2	14.6	11
Berlin	-0.2	0.1	4.4	8.2	13.8	16	18.3	18	14.4	10	4.2	1.2
Bruxelles	3.3	3.3	6.7	8.9	12.8	15.6	17.8	17.8	15	11.1	6.7	4.4
Budapest	-1.1	0.8	5.5	11.6	17	20.2	22	21.3	16.9	11.3	5.1	0.7
Copenhag	-0.4	-0.4	1.3	5.8	11.1	15.4	17.1	16.6	13.3	8.8	4.1	1.3
Dublin	4.8	5	5.9	7.8	10.4	13.3	15	14.6	12.7	9.7	6.7	5.4
Helsinki	-5.8	-6.2	-2.7	3.1	10.2	14	17.2	14.9	9.7	5.2	0.1	-2.3
Kiev	-5.9	-5	-0.3	7.4	14.3	17.8	19.4	18.5	13.7	7.5	1.2	-3.6
Cracovie	-3.7	-2	1.9	7.9	13.2	16.9	18.4	17.6	13.7	8.6	2.6	-1.7
Lisbonne	10.5	11.3	12.8	14.5	16.7	19.4	21.5	21.9	20.4	17.4	13.7	11.1
Londres	3.4	4.2	5.5	8.3	11.9	15.1	16.9	16.5	14	10.2	6.3	4.4
Madrid	5	6.6	9.4	12.2	16	20.8	24.7	24.3	19.8	13.9	8.7	5.4

```
                    LES DONNEES APRES L'ACP
-------------------------------------------------------------------------------


-0.816187
  6.91716
 -1.08833
-0.154157
 0.869163
 -2.29097
 -1.36594
 -4.91621
 -2.54492
 -2.08852
  4.90306
 -0.73669
  3.31254
```

```
ENTRER LA DIMENTION DE VOTRE CARTE              : 5
ENTRER LE MAX D'ITERATION POUR CALCULER(R/LR)   : 1000
ENTRER LE TAUX D'APPRENTISSAGE                  : 0.5
ENTRER L'ERREUR MINIALE D'APPRENTISSAGE         : 0.00001
```

Carte classique

1000 itérations

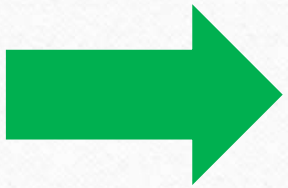
Erreur moyen=0,9

Carte amelioré

1000 itérations

Erreur moyen=0,2

```
ENTREE [1]  -> 6.91716
WINNER [1,0]-> 6.29426

ENTREE [2]  -> -1.08833
WINNER [3,1]-> -1.03916

ENTREE [3]  -> -0.154157
WINNER [1,4]-> -0.282626

ENTREE [4]  -> 0.869163
WINNER [2,1]-> 1.23729

ENTREE [5]  -> -2.29097
WINNER [4,1]-> -2.14904

ENTREE [6]  -> -1.36594
WINNER [3,3]-> -1.28149

ENTREE [7]  -> -4.91621
WINNER [4,4]-> -4.33586

ENTREE [8]  -> -2.54492
WINNER [3,4]-> -2.52152

ENTREE [9]  -> -2.08852
WINNER [4,0]-> -2.0601

ENTREE [10]  -> 4.90306
WINNER [0,1]-> 4.63288

ENTREE [11]  -> -0.73669
WINNER [2,3]-> -0.87864

ENTREE [12]  -> 3.31254
WINNER [0,2]-> 3.55799
```



	Bruxelles		Kiev	Helsinki
		Londres	Dublin	
Madrid				
Lisbonne		Budapest	Berlin	Copenhague
	Athènes			Cracovie

TRAVAIL 2:

PMR

1. Principe
2. L'architecture du réseau
3. Algorithme d'apprentissage
4. Propriétés
5. Exemples d'application
6. Réalisation
7. Les testes

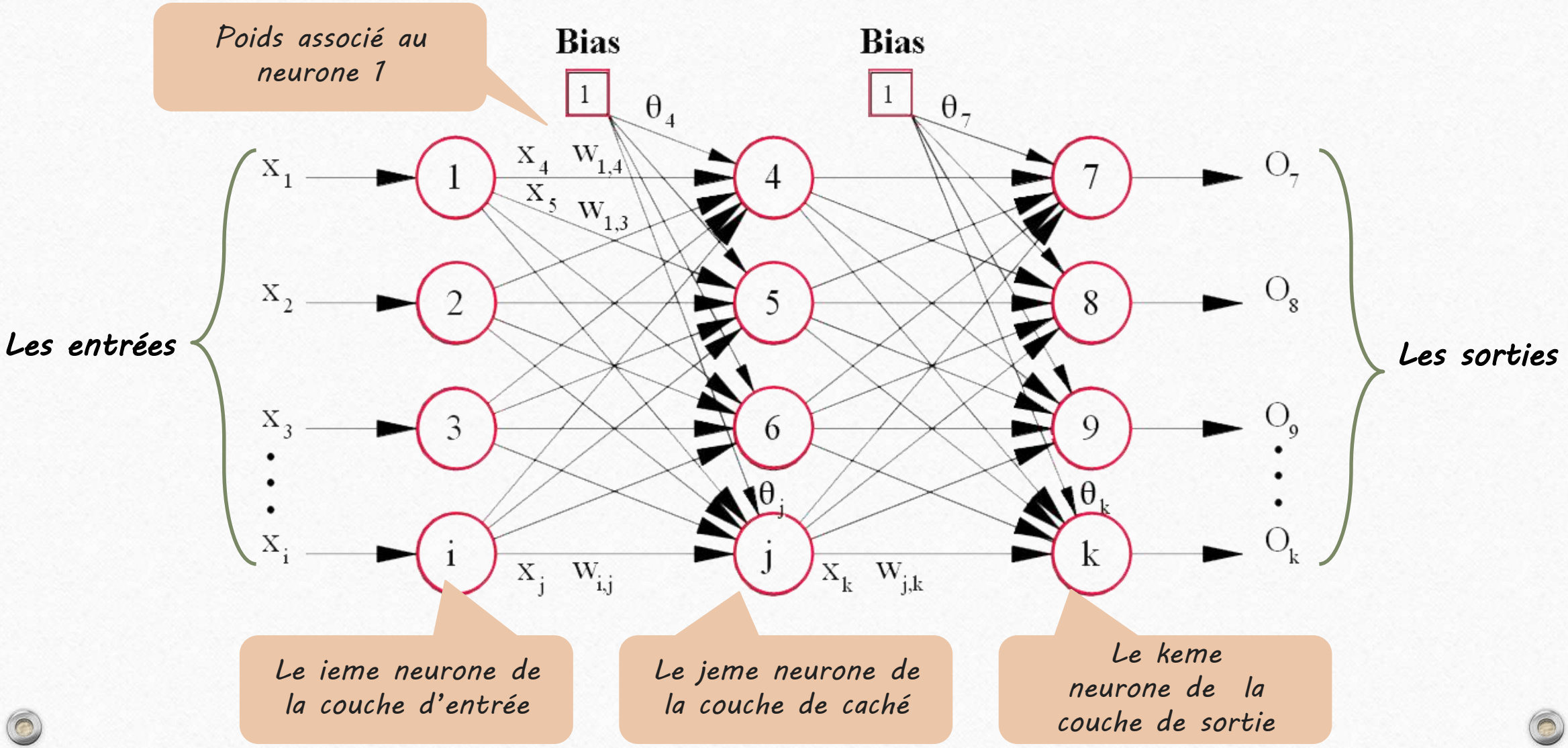
PMR

1. Principe

La rétropropagation basée sur la diminution du gradient d'erreur par rapport aux poids du réseaux

$$E = \frac{1}{2}\sum_{i=0}^{m}(y_i - \theta_i)^2$$

2. L'architecture du réseau

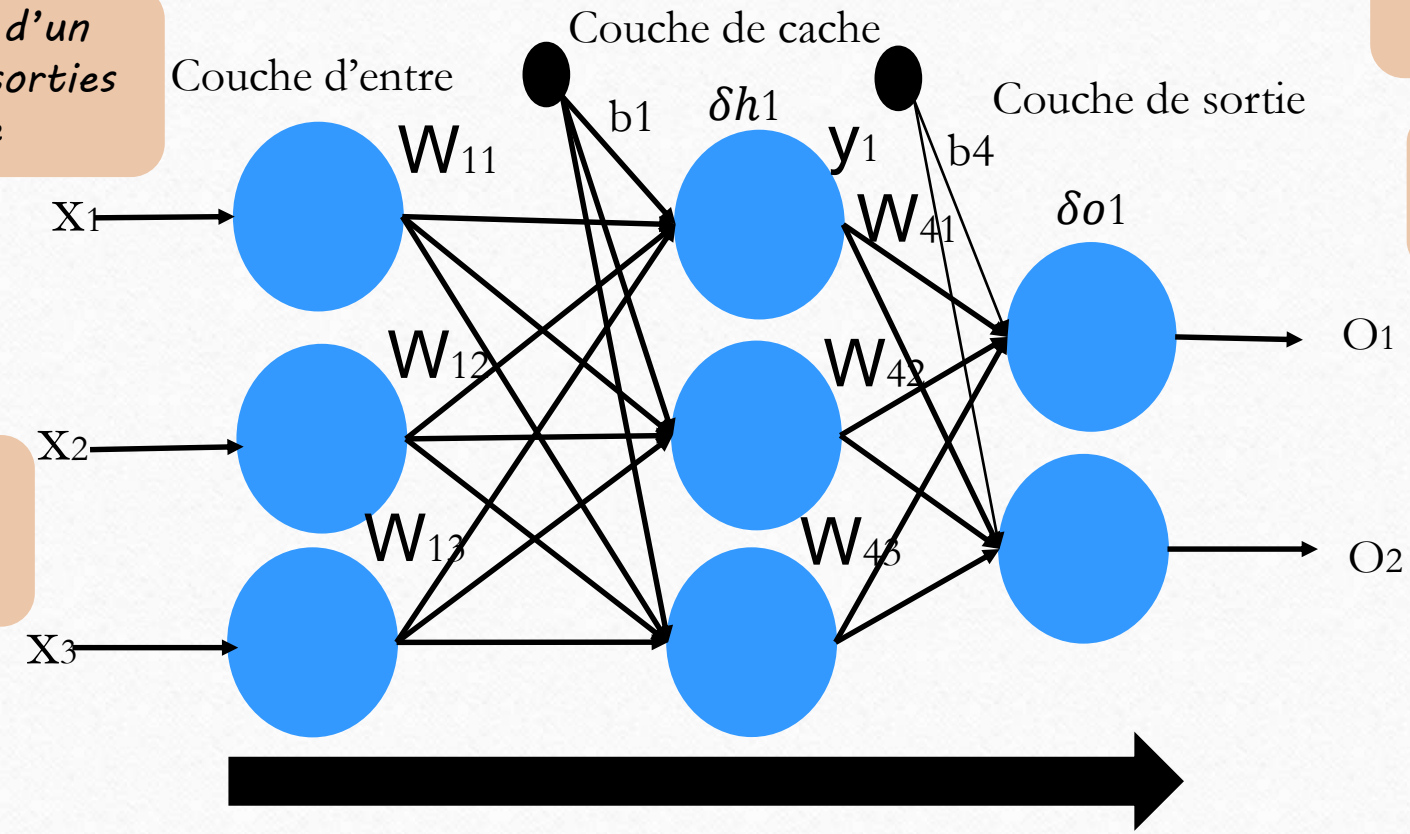


3· Algorithme d'apprentissage

3 · Calcul des sorties du RNA à partir des entrées qui lui sont appliquées et calcul de l'erreur entre ces sorties et les sorties désirés à apprendre

2 · Application d'un vecteur entrées-sorties à apprendre

1 · Initialiser aléatoirement les coefficients wij



Calcul la somme $X=\sum X_j W_{ij} + b_j$

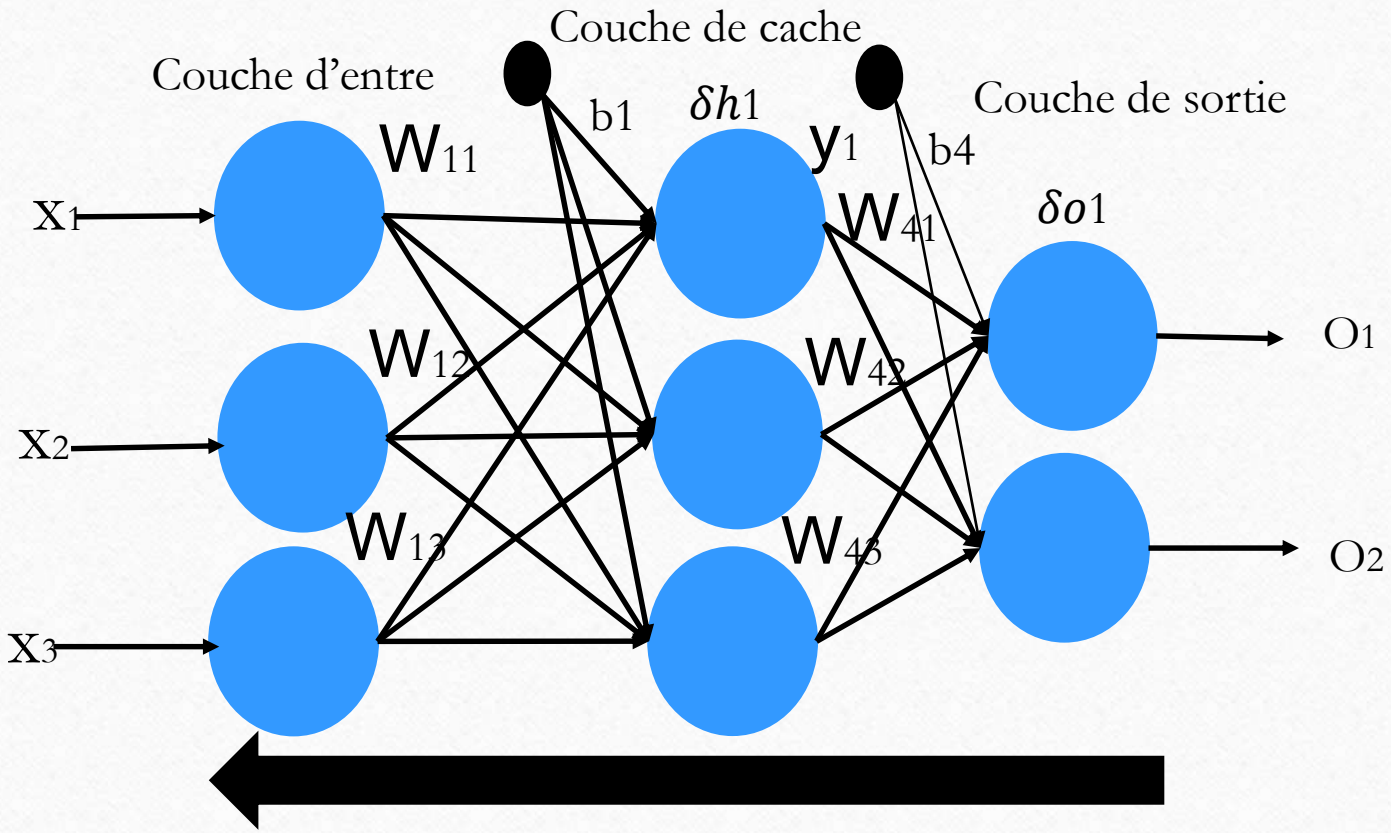
$$F(x) = \frac{1}{1 + e^{-X}}$$

$S=Di - Oi$

Di

4 · Correction des poids des liens entre les neurones de la couche de sortie et de la première couche cachée selon l'erreur présente en sortie

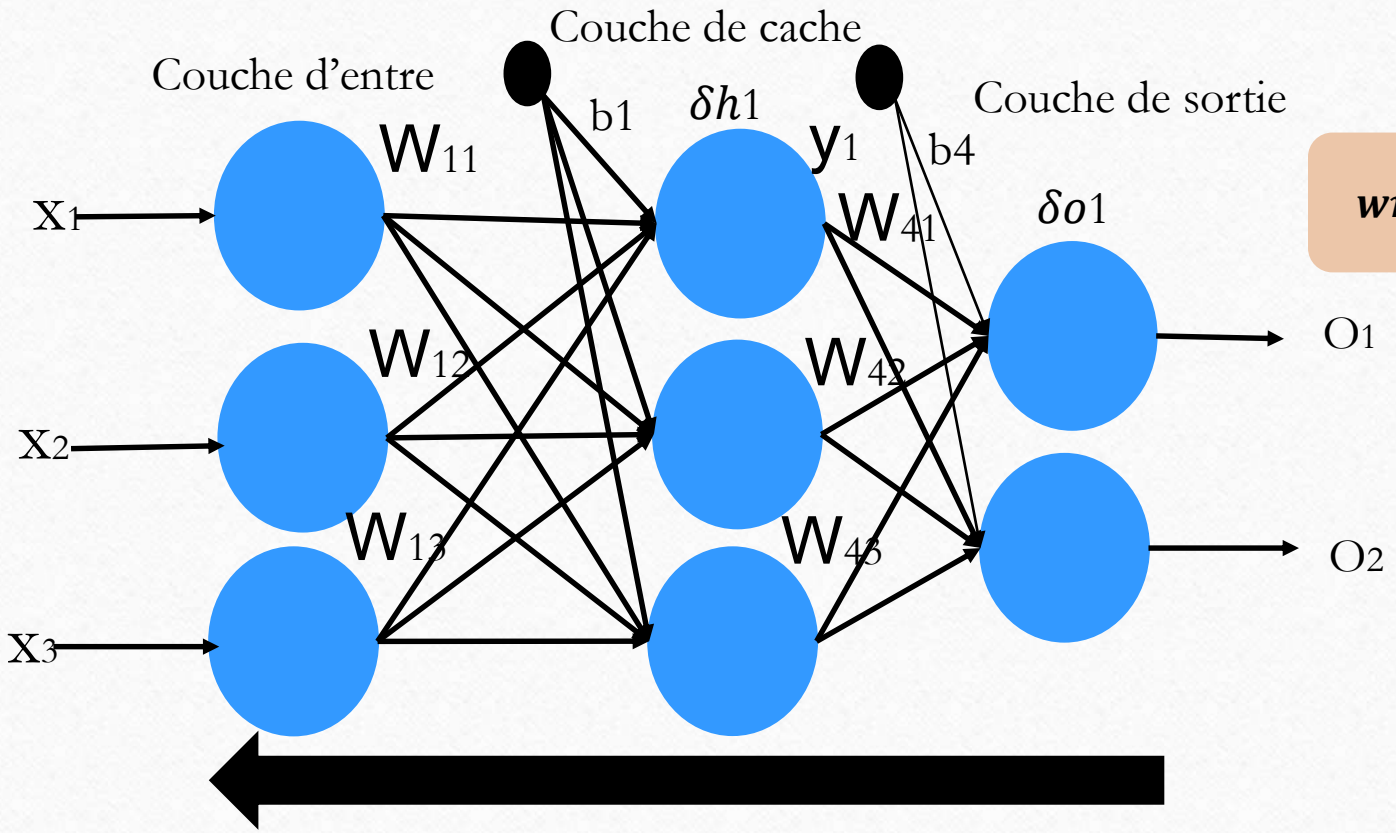
$$\delta oi = f'(\sum yjWjk + bi) * (di - Oi)$$
$$\delta hj = f'(\sum xiWij + bj)(\sum \delta oiwij)$$
$$wn = wn + \eta * \delta o1 * O1 + \alpha * wn$$



$$f'(x) = f(x)[1 - f(x)]$$

5·Propagation de l'erreur sur la couche précédente et correction des poids des liens entre les neurones de la couche cachée et ceux en entrées·

6·Boucler à la 2eme étape avec vecteur d'entrées-sorties tant que les performances du RNA (erreur sur les sorties) ne sont pas satisfaisantes



$$wn = wn + \eta * \delta h1 * O1 + \alpha * wn$$

PMR

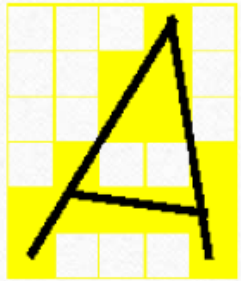
4. Propriétés

- ✓ la capacité d'apprentissage
- ✓ Ce n'est pas le seul modèle multicouche, mais c'est le plus courant.
- ✓ Neurones réparties en couches : entrée, cachées (traitement), et décision.
- ✓ Apprentissage supervisé : cherche les paramètres de réseaux qui minimise l'erreur
- ✓ entre le calculé et désiré par la Retropropagation de gradient.
- ✓ Les flots de signal vont dans une seule direction.
- ✓ trouve des solutions à certains problèmes non linéaires.
- ✓ Les perceptrons peuvent représenter toutes les fonctions linéairement séparables

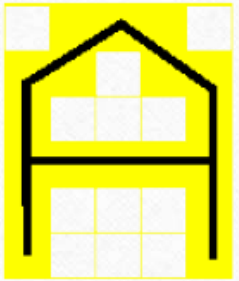
PMR

5. Exemples d'application

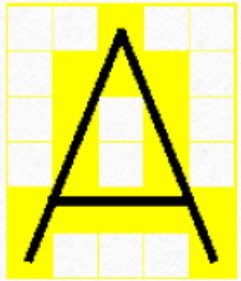
la reconnaissance des caractères



$\bar{x}_1$



$\bar{x}_2$



$\bar{t}$

$$\bar{x}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} = (0\,0\,0\,1\,0\,0\,0\,1\,1\,0\,0\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,1)$$

C: 

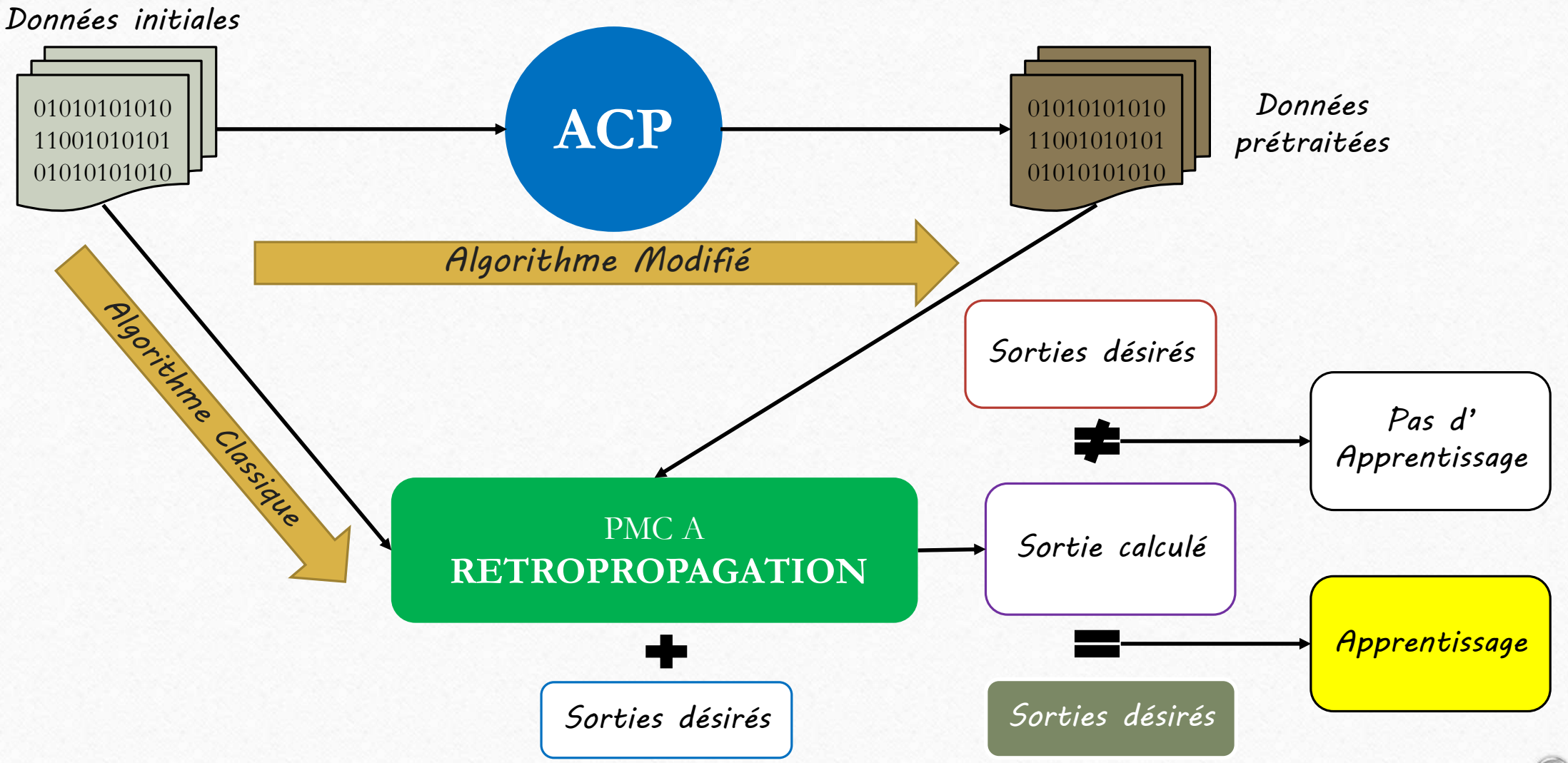
L: 

T: 

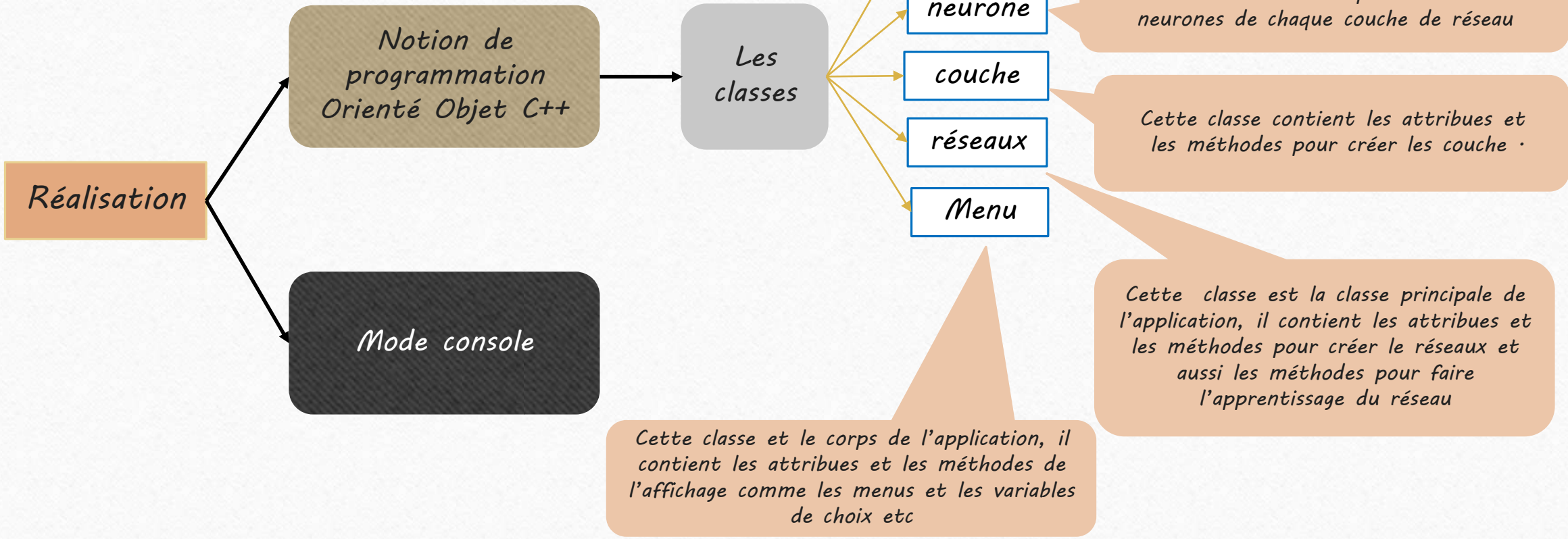
5. Exemples d'application

X1	X2	XOR
0	0	0
0	1	1
1	1	0

6· Réalisation



Description de modèle logiciel réalisé



7. Les testes

Tester l'application

la reconnaissance des caractères

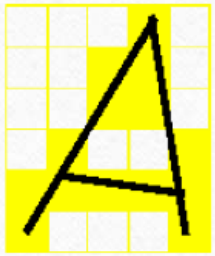
Exemple 1:

C:  =111100111

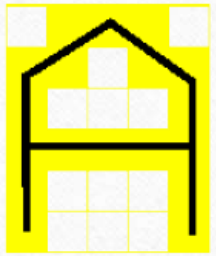
L:  =100100111

T:  =111010010

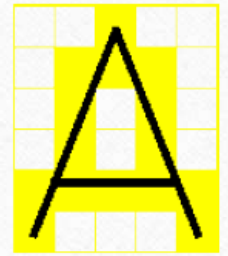
Exemple 2:



$\bar{x}_1$



$\bar{x}_2$



$\bar{t}$

$$\bar{x}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} = (0\,0\,0\,1\,0\,0\,0\,1\,1\,0\,0\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1\,1\,1\,1\,1\,0\,0\,0\,1)$$

```
DONNER LE NOMBRE DE NEURON DE ENTRE :3
DONNER LE NOMBRE DE NEURON DE SORTIE :9
DONNER LE NOMBRE DE COUCHE CACHE :1
DONNER LE NOMBRE DE NEURON DE 1 COUCHE CACHE :3
```

```
                LES ENTTRES :
                 1 1 1 1 0 0 1 1 1
                 1 0 0 1 0 0 1 1 1
                 1 1 1 0 1 0 0 1 0

SORTIE DESIREE: 1 RESULT: 0.992668
SORTIE DESIREE: 1 RESULT: 0.907756
SORTIE DESIREE: 1 RESULT: 0.907972
SORTIE DESIREE: 1 RESULT: 0.94638
SORTIE DESIREE: 0 RESULT: 0.0534376
SORTIE DESIREE: 0 RESULT: 0.00526095
SORTIE DESIREE: 1 RESULT: 0.946302
SORTIE DESIREE: 1 RESULT: 0.992587
SORTIE DESIREE: 1 RESULT: 0.946248
SORTIE DESIREE: 1 RESULT: 0.99902
SORTIE DESIREE: 0 RESULT: 0.0681903
SORTIE DESIREE: 0 RESULT: 0.0680361
SORTIE DESIREE: 1 RESULT: 0.999943
SORTIE DESIREE: 0 RESULT: 5.83185e-005
SORTIE DESIREE: 0 RESULT: 0.00047015
SORTIE DESIREE: 1 RESULT: 0.999942
SORTIE DESIREE: 1 RESULT: 0.998987
SORTIE DESIREE: 1 RESULT: 0.999943
SORTIE DESIREE: 1 RESULT: 0.965073
SORTIE DESIREE: 1 RESULT: 0.998017
SORTIE DESIREE: 1 RESULT: 0.998058
SORTIE DESIREE: 0 RESULT: 0.0649331
SORTIE DESIREE: 1 RESULT: 0.935304
SORTIE DESIREE: 0 RESULT: 0.0358428
SORTIE DESIREE: 0 RESULT: 0.0650337
SORTIE DESIREE: 1 RESULT: 0.965125
SORTIE DESIREE: 0 RESULT: 0.0651038
```

```
NEU_input 1     Poid                                    Bias
          -0.000625629                                0.373302
             0.281793
           -0.0966521
              0.40437
           -0.292505
            0.239937
           -0.175146
            0.447981
            -0.41142

NEU_input 2     Poid                                    Bias
           -0.087054                                0.0829493
            0.429472
           -0.355251
            0.256767
           -0.151997
          0.00749229
          -0.0457015
            0.182226
          -0.0736564

NEU_input 3     Poid                                    Bias
           -0.494263                                0.303583
            0.222846
          -0.0595416
          0.00233467
          -0.0044557
             0.18894
           -0.265831
            0.285592
           -0.300882

            **************************************************
hindin 0        Poid                                    Bias
          -0.0831172                                0.176061
            0.331523
           -0.225394

          -0.0285195                                0.401303
            0.303842
```

Tester l'application

La résolution d'un problème non linéairement séparable: ou exclusif XOR

X1	X2	XOR
0	0	0
0	1	1
1	1	0

```
DONNEE LE NOMBRE DES ITERATIONS :1000
DONNEE LE TAUX D APPRENTISAGE :0.5
DONNEE LE MOMENTUM  :0.4
```

```
                LES ENTTRES :
                 0 0
                 0 1
                 1 1
SORTIE DESIREE: 0 RESULT: 0.0234454
SORTIE DESIREE: 1 RESULT: 0.946314
SORTIE DESIREE: 0 RESULT: 0.0193437
```

Tester avec ACP

Colores	Poids	Année	Km	Km/h	SD
Noir=1	3500	1992	5600	260	0,1
Bleu=2	6500	1993	8600	120	0,5
Vert=3	8500	2000	1000	100	0,3

```
DONNEE LE NOMBRE DES ITERATIONS :1000
DONNEE LE TAUX D APPRENTISAGE :0.5
DONNEE LE MOMENTUM  :0.4
```

```
DONNER LE NOMBRE DE NEURON DE ENTRE :4
DONNER LE NOMBRE DE NEURON DE SORTIE :1
DONNER LE NOMBRE DE COUCHE CACHE :1
DONNER LE NOMBRE DE NEURON DE 1 COUCHE CACHE :3
```

```
LES ENTTRES :
 -2.28489 -0.838457
 -0.289358 1.4228
 2.57424 -0.584346

SORTIE DESIREE: 0.1 RESULTAT: 0.100095
SORTIE DESIREE: 0.5 RESULTAT: 0.49995
SORTIE DESIREE: 0.3 RESULTAT: 0.3
```

Tester sans ACP

Colores	Poids	Année	Km	Km/h	SD
Noir=1	3500	1992	5600	260	0,1
Bleu=2	6500	1993	8600	120	0,5
Vert=3	8500	2000	1000	100	0,3

```
DONNEE LE NOMBRE DES ITERATIONS :1000
DONNEE LE TAUX D APPRENTISAGE :0.5
DONNEE LE MOMENTUM  :0.4
```

```
DONNER LE NOMBRE DE NEURON DE ENTRE :4
DONNER LE NOMBRE DE NEURON DE SORTIE :1
DONNER LE NOMBRE DE COUCHE CACHE :1
DONNER LE NOMBRE DE NEURON DE 1 COUCHE CACHE :3
```

```
                LES ENTTRES :
                 1 3500 1992 5600 260
                 2 6500 1993 8600 120
                 3 8500 2000 1000 100

SORTIE DESIREE: 0.1 RESULT: 0.294401
SORTIE DESIREE: 0.5 RESULT: 0.294401
SORTIE DESIREE: 0.3 RESULT: 0.318384
```

MERCI DE VOTRE ATTENTION