République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mentouri – Constantine

Faculté des Sciences de l'Ingénieur
Département d'Informatique

N° d'ordre :234/Mag/2008
Série :004/Inf/2008

**Mémoire**

**Présenté pour l'obtention du diplôme de magistère en informatique**

**Option : Systèmes d'Information & Intelligence Artificielle Distribués**

Thème :

# *Approches incrémentales pour l'apprentissage en reconnaissance de formes*

***Dirigé par :***
***Dr. F. Hachouf***

***Présenté par :***
***Hebboul Amel***

***Soutenu le : 23/06/2008***

***Devant le jury :***

| | | |
|---|---|---|
| Prof. Z. BOUFAIDA | Professeur, Université de Constantine | Président |
| Dr. M.K. KHOLLADI | Maître de Conférences, Université de Constantine | Examinateur |
| Dr. S. CHIKHI | Maître de Conférences, Université de Constantine | Examinateur |
| Dr. F. HACHOUF | Maître de Conférences, Université de Constantine | Rapporteur |

*Ma famille pour ses conseils, ses soutiens, ses amours et tendresses et ses encouragements,*

*Mon mari Riad,*

*A tous ceux qui de prés ou de loin ont contribué à ce projet.*

*A tous je dis merci.*

# بسم الله الرّحمن الرّحيم

إِنَ الحَمْدَ لله، نَحْمَدُه تعالى ونشكره على ما أتَانَا مِنْ عِلْمٍ، نَدْعُوه تعالى أَنْ يكون عِلْمًا نَافِعًا.

قال الله تعالى: " وقُلْ رَّبِّ زِدْنِي عِلْمًا "

# *Remerciements*

*Je tiens à assurer de ma  sincère et totale gratitude et de ma profonde reconnaissance mon encadreur, Mme Fella Hachouf qui, par sa confiance, ses conseils, ses explications a contribué de manière essentielle aux résultats obtenus, elle est à l'origine de ce travail, elle a participé largement à sa finalisation.*

*Je remercie vivement Mme Zizette Boufaida d'avoir accepté d'examiner ce travail et de présider ce jury.*

*Je suis très reconnaissante à Mr Salime ChiKhi et Mr Mohamed Kheireddine Kholladi d'avoir eu la gentillesse de juger ce travail.*

*J'adresse mes sincères remerciements pour l'ensemble des enseignants qui m'ont donné la passion de l'informatique, et toutes les personnes qui m'ont conduit directement ou indirectement à réaliser ce modeste travail.*

*Ma famille pour ses conseils, ses soutiens, ses amours et tendresses et ses encouragement,*
*Mon mari Riad,*
*A tous ceux qui de prés ou de loin ont contribué à ce projet.*

*A tous je dis merci.*

# Résumé

Dans le cadre de ce magistère, nous proposons deux approches incrémentales pour l'apprentissage en reconnaissance de formes. Ces approches sont des réseaux de neurones constructifs avec un apprentissage non supervisé (Incremental Neural Gas with Adapted Threshold (IGNGAT) et Incremental Neural Gas with utility parameter IGNGU). Dans IGNGAT, le seuil de similarité est calculé à partir de la base d'apprentissage et une valeur de précision fixée à priori. IGNGU est un réseau de neurones à deux couches capable de partitionner un espace de données bruitées, non stationnaires et non étiquetées. Dans la première couche, le seuil de similarité est adaptatif à la situation présente. Et pour la deuxième couche, le seuil est calculé à partir de la première couche. Sa valeur reste fixe pendant l'apprentissage de cette couche. Les deux couches suppriment le bruit par l'utilisation du paramètre d'utilité et peuvent opérer en parallèle dans le cas incrémental. Nous avons testé ces réseaux dans différents problèmes de classification de données tels que la segmentation des images de textures, la classification de logos et des cellules drépanocytaires.

# Abstract

Under this study, we offer two incremental approaches for learning in pattern recognition. These approaches are constructive neural networks with an unsupervised learning (Incremental Neural Gas with Adapted Threshold (IGNGAT) and Incremental Neural Gas with utility parameter IGNGU). In IGNGAT, A similarity threshold is computed from learning data base and a precision value *a priori* set. IGNGU is a neural network composed of two layers able to partition no stationary and no labeled noisy data. In the first layer, the similarity threshold is adaptive. For the second layer, the threshold is calculated from the first layer. Its value remains fixed during the learning of this layer. The two layers delete noise using a utility parameter and they can operate in parallel in the incremental case. Networks have been these in different classification problems such as texture image segmentation, and cells logos classification.

Introduction générale……...1

Chapitre 1 : Introduction à la reconnaissance des formes et les réseaux de neurones

I- Introduction ..... 3

II- Reconnaissance de formes ..... 4
2-1- Schéma général d'un système de reconnaissance de formes ..... 4
2-2- Les différentes étapes d'un système de reconnaissance de formes ..... 4
2-2-1- Le prétraitement ..... 4
2-2-2- Extraction des descripteurs ..... 5
2-2-3- Classification ..... 6
2-3- Les méthodes de classification ..... 7
2-3-1- Approches Statistiques ..... 7
2-3-2- Approches Structurelles (syntaxiques) ..... 8
2-3-3- Approches bio inspirées ..... 8
2-3-4- Séparateurs à large marge (Support Vector Machines SVM) ..... 10

III- Les réseaux de neurones ..... 13
3-1- Définitions de base ..... 13
3-1-1- Le neurone biologique ..... 13
3-1-2- Le neurone formel ..... 14
3-1-3- Les réseaux de neurones ..... 15
3-1-4- L'apprentissage ..... 15
3-2- Les types de réseaux de neurones ..... 18
3-2-1- Les réseaux de neurones non bouclés ..... 18
3-2-2- Les réseaux de neurones bouclés (ou récurrents) ..... 19
3-2-3-Les cartes topologiques autos-organisatrices ..... 19

IV- Conclusion ..... 20

Chapitre 2 : Les réseaux de neurones incrémentaux

I- Introduction ..... 22

II- Growing Neural Gas (GNG) ..... 22
2-1- Définition ..... 22
2-2- Principe ..... 22
2-3- Algorithme ..... 23
2-4- Discussion ..... 25

III- Incremental Growing Network Gas (IGNG) ..... 27
3-1- Définition ..... 27
3-2- Principe ..... 27
3-3- Algorithme ..... 29
3-4- Discussion ..... 30

IV- Réseau de neurones incrémental pour l'apprentissage non supervisé(IN) ... 32
4-1- Définition ..... 32
4-2- Principe ..... 32
4-3- Algorithme ..... 33

4-4- Exemples ..... 36

V- Réseau de neurones incrémental (Incremental Neural Network (INeN)) ..... 39
5-1- Définition ..... 39
5-2- Principe ..... 39
5-3- Algorithme ..... 40
5-4- Seuillage automatique ..... 40
5-5- Discussion ..... 41
VI- Conclusion ..... 41

Chapitre 3: Les approches proposées

I- Contribution ..... 42

II- Un réseau incrémental à seuillage adaptatif ..... 42
2-1- Principe général ..... 42
2-2- Algorithme ..... 43

III- Un réseau incrémental à deux couches avec un paramètre d'utilité ..... 44
3-1- Principe général ..... 44
3-2- Algorithme ..... 48
3-2-1- Algorithme d'apprentissage de la première couche ..... 48
3-2-2- Calcul du seuil $T_c$ à partir de la première couche ..... 51
3-2-3- Algorithme d'apprentissage de la deuxième couche ..... 51
3-2-4- L'étiquetage des neurones des deux couches ..... 53

IV- Conclusion ..... 53

Chapitre 4 : Les résultats expérimentaux

I- Les résultats de IGNGAT ..... 54
1-1- Les paramètres de textures ..... 54
1-1-1- Remplissage d'une matrice de co-occurrence ..... 54
1-1-2- Les caractéristiques ..... 55
1-2- Le processus de segmentation d'une image de textures ..... 57
1-3- Les images de synthèse ..... 57
1-3-1- Les images de textures fines ..... 58
1-3-2- Les images de textures grossières ..... 59
1-4- Les images aériennes ..... 59
1-4-1- Les images aériennes en milieu urbain ..... 60
1-4-2- Les images aériennes en milieu rural ..... 61
1-4-3- Les images aériennes en milieu semi urbain ..... 62
1-5- Les images médicales ..... 63

II- Les résultats de IGNGU ..... 64
2-1- Une base de données artificielle ..... 64
2-1-1- Cas passif (non incrémental) ..... 64
2-1-2- Cas incrémental ..... 66
2-2- La classification de textures ..... 69
2-2-1- Les images de synthèse ..... 70
2-2-2- Les images ariennes ..... 72
2-2-3- Les images médicales ..... 73

2-3- La classification des LOGOS.....75
2-3-1- Les moments invariants.....75
2-3-2- Cas passif.....77
2-3-3- Cas incrémental.....78
2-4- La drépanocytose.....79
III- Conclusion…….82
Conclusion générale et perspectives…….....84
Bibliographies…….....85

Fig.1.1. Les étapes d'un système de reconnaissance de formes. ..... 4
Fig.1.2. Forme générale d'un algorithme évolutionniste. ..... 10
Fig.1.3. Séparation linéaire optimale.. ..... 11
Fig.1.4. Deux classes avec une frontière très non linéaire.. ..... 12
Fig.1.5. Exemple de Neurone Biologique ..... 13
Fig.1.6. La connexion entre deux neurones biologiques. ..... 14
Fig.1.7. Un neurone formel. ..... 14
Fig.1.8. Un réseau de neurones à n entrées,. ..... 18
Fig.1.9. Un réseau de neurones bouclé à deux entrées. ..... 19
Fig.1.10. Exemple d'une carte auto organisée à une dimension.. ..... 20
Fig.1.11. Exemple d'une carte auto organisée à deux dimensions. ..... 20
Fig.2.1. Exemple d'un GNG entraîne sur des données générées à partir d'un espace d'entrée avec différentes dimensions dans des surfaces différentes. ..... 25
Fig.2.2. La distribution non stationnaire d'espace d'entrées.. ..... 26
Fig.2.3. Le GNGU apprend la distribution non stationnaire d'espace d'entrées avec succès, 27
Fig.2.4. Insertion d'un neurone quand la donnée à apprendre est trop loin de son neurone gagnant. ..... 28
Fig.2.5. Insertion d'un neurone quand la donnée à apprendre est trop loin de son deuxième neurone gagnant ..... 29
Fig.2.6. Réseau IGNG pour la visualisation de données. ..... 31
Fig.2.7. Comportement du GNG (a) et du IGNG (b) dans le cas d'un apprentissage incrémental. ..... 31
Fig.2.8. La base de données artificielles utilisées dans l'exemple. ..... 36
Fig.2.9. Le résultat du GNG dans le cas stationnaire : une classe. ..... 36
Fig.2.10. Le résultat du IN de la première couche dans le cas stationnaire : trois classes ..... 37
Fig.2.11. Le résultat du IN de la deuxième couche dans le cas stationnaire : cinq classes ..... 37
Fig.2.12. Le résultat du GNG dans le cas non stationnaire. ..... 37
Fig.2.13. Le résultat du GNGU dans le cas non stationnaire. ..... 38
Fig.2.14. Le résultat du IN de la première couche dans le cas non stationnaire. ..... 38
Fig.2.15. Le résultat du IN de la deuxième couche dans le cas non stationnaire . ..... 38
Fig.2.16. Structure d'un réseau de neurones incrémental. ..... 39
Fig.3.1. Schéma général d'apprentissage des deux couches du IGNGU. ..... 45
Fig.3.2. Ensemble de données représenté dans un espace à deux dimensions.. ..... 46
Fig.4.1. Exemple de matrice de co-occurrence. ..... 54
Fig.4.2. Le processus de segmentation d'une image de textures. ..... 57
Fig.4.3. Image à une texture. ..... 58
Fig.4.4. Espace d'entrée partitionné ..... 58
Fig.4.5. Espace d'entrée partitionné en deux classes ..... 59
Fig.4.6. Deux textures fines. ..... 59
Fig.4.7. Deux textures grossières ..... 59
Fig.4.8. Une texture grossière et une texture fine ..... 59
Fig.4.9. Image aérienne en milieu urbain . ..... 60
Fig.4.10 Le nombre de neurones dans le réseau IGNGAT et le nombre de classes détectées par rapport à la valeur de $P_r$. ..... 61
Fig.4.11. Image aérienne en milieu rural. ..... 61
Fig.4.12. Le nombre de neurones dans le réseau IGNGAT et le nombre de classes détectées par rapport à la valeur de $P_r$. ..... 62
Fig.4.13. Image aérienne en milieu semi urbain. ..... 62
Fig.4.14. Le nombre de neurones dans le réseau IGNGAT et le nombre de classes détectées par rapport à la valeur de $P_r$. ..... 62
Fig.4.15. Coupe transversal d'un crâne humain (IGNGAT) . ..... 62

Fig.4.16. Coupe longitudinal d’un crâne humain (IGNGAT) . ..... 63
Fig.4.17. La base de données artificielle. ..... 64
Fig.4.18. Le résultat du IGNGU sur la base sans bruit. ..... 65
Fig.4.19. Le nombre de neurones par rapport au temps dans le cas passif sans bruit ..... 65
Fig.4.20. Le résultat de la méthode proposée sur la base sans bruit. ..... 66
Fig.4.21. Le nombre de neurones par rapport au temps dans le cas passif avec bruit. ..... 66
Fig.4.22. Le résultat de la méthode proposée sur la base sans bruit dans le cas incrémental. 67
Fig.4.23. Le nombre de neurones par rapport au temps dans le cas incrémental sans bruit..... 68
Fig.4.24. Le résultat de la méthode proposée sur la base sans bruit dans le cas incrémental. 68
Fig.4.25. Le nombre de neurones par rapport au temps dans le cas incrémental avec bruit .... 69
Fig.4.26. Exemple d’une fenêtre de segmentation. ..... 70
Fig.4.27. Segmentation d’une image de synthèse de 5 textures avec FS=65*65 en changeant la taille de la fenêtre d’affectation.. ..... 70
Fig.4.28. La segmentation hiérarchique. ..... 70
Fig.4.29. Le résultat de la segmentation d’une image de 4 textures. ..... 71
Fig.4.30. Le résultat de la segmentation d’une image de 5 textures. ..... 72
Fig.4.31. Le résultat de la segmentation d’image au milieu rural. ..... 72
Fig.4.32. Le résultat de la segmentation d’image au milieu semi rural. ..... 73
Fig.4.33. Le résultat de la segmentation d’image au milieu urbain. ..... 73
Fig.4.34. Coupe transversal d’un crâne humain (IGNGU). ..... 74
Fig.4.35. Coupe longitudinal d’un crâne humain (IGNGU). ..... 74
Fig.4.36. Coupe longitudinal 2 d’un crâne humain (IGNGU). ..... 74
Fig.4.37. La base de logos utilisé. ..... 75
Fig.4.38. Logo2. ..... 77
Fig.4.39. Le nombre de neurones par rapport au temps dans la classification de logos dans le cas passif. ..... 78
Fig.4.40. Le nombre de neurones par rapport au temps dans les deux couches dans la classification de logos dans le cas incrémental ..... 79
Fig.4.41. Cliché de frottis sanguin d'un patient drépanocytaire. ..... 79
Fig.4.42. Processus de falciformation d’un globule drépanocytaire. ..... 80
Fig.4.43. L’image constituant la base d'apprentissage. ..... 80
Fig.4.44. Les cellules extraites à partir de l’image. ..... 81
Fig.4.45. Le nombre de neurones par rapport au temps dans les deux couches dans la classification de cellules drépanocytaires. ..... 81

Tab.1.1. Fonctions de transferts a=f(n). ..... 15
Tab.4.1. Les paramètres utilisés dans le cas passif sans bruits (base de données artificielle).. 65
Tab.4.2. Les paramètres utilisés dans le cas passif avec bruits (base de données artificielle).. 66
Tab.4.3. Les paramètres utilisés dans le cas incrémental sans bruits (base de données artificielle). ..... 67
Tab.4.4. Les paramètres utilisés dans le cas incrémental avec bruits (base de données artificielle). ..... 68
Tab.4.5. Les paramètres utilisés (images de 4 textures de Broadatz)..... 71
Tab.4.6. Les paramètres utilisés (images de 5 textures)..... 71
Tab.4.7. Les paramètres utilisés (images ariennes)..... 73
Tab.4.8. Les paramètres utilisés (images médicales). ..... 74
Tab.4.9. Les paramètres utilisés pour la création de la base d'apprentissage des logos. ..... 75
Tab.4.10. Les moments calculés sur les quatre images décrites à la figure Fig.4.38..... 77
Tab.4.11. Les paramètres utilisés dans le cas passif (logos). ..... 77
Tab.4.12. Les paramètres utilisés dans le cas passif (logos). ..... 78
Tab.4.13. Les paramètres utilisés dans le cas incrémental (logos)..... 78
Tab.4.14. Les paramètres utilisé dans la classification de cellules. ..... 81
Tab.4.15. Le résultat de la classification de cellules du IGNGU. ..... 82

# Introduction générale

La reconnaissance de formes a pour but d'identifier la classe d'un objet. Selon les applications, ces formes peuvent être des images, des signaux, des sons, etc. En mesurant des caractéristiques sur une forme (avec un capteur) nous transformons cette forme en un vecteur d'entrée (patron ou observation) [1].

Pour une machine, la classification de visages, de données médicales, de formes, sont toutes des tâches assez difficiles. Par exemple, dans le cas de la reconnaissance de caractères manuscrits, il est difficile d'énoncer une description générale qui tienne compte de toutes les variations particulières de chaque caractère. Une autre approche qui peut être utilisée pour cette tâche est celle de l'apprentissage. Ainsi, le critère pour décider si une image correspond ou non à une lettre 'A' consiste à comparer si cette image est suffisamment similaire à des '$\mathcal{A}$' vus auparavant. De ce point de vue, on ne calcule pas la classification de caractères : elle doit être apprise à partir d'exemples.

Il existe deux types d'apprentissage. L'apprentissage supervisé où les exemples sont étiquetés. Donc nous avons besoin d'un expert dans le domaine d'application. L'apprentissage non supervisé, où nous n'avons aucune information *a priori* sur les classes des exemples. Dans les deux types d'apprentissage, nous avons besoin d'une base d'exemples qui s'appelle une base d'apprentissage. La question qui se pose est : « combien d'exemples doit contenir la base pour que le système de reconnaissance des formes puisse classer toutes les formes ? » La réponse est que les performances du système de reconnaissance des formes augmentent avec la taille de la base. Malheureusement la taille de cette dernière est limitée par les capacités matérielles des ordinateurs. Une autre question se pose si des nouvelles données arrivent : « est ce que le système est capable d'apprendre ces nouvelles données sans réapprendre les données déjà apprises, et sans les détériorer ? » Pour traiter ces problèmes un nouveau type d'apprentissage apparaît. C'est l'apprentissage incrémental. Dans ce dernier, le système est capable d'apprendre des nouvelles données sans oublier les données déjà apprises. Dans le cas où la taille de la base d'apprentissage est très grande, nous divisons cette dernière en différentes parties. Et le système apprend ces parties successivement. Dans ce cadre plusieurs méthodes sont proposées tels que les réseaux de neurones et les séparateurs à large marge (SVM).

Les réseaux de neurones sont inspirés des systèmes nerveux humains et proposés en 1943 par W. McCulloch et W. Pitts. Ces réseaux ont connu plusieurs améliorations et modifications. Il existe deux grandes familles des réseaux de neurones. La première famille

contient les réseaux de neurones non bouclés et les réseaux de neurones bouclés. La deuxième famille contient les cartes auto organisatrices [2].

Les cartes auto organisatrices font partie de la famille des modèles dits *apprentissage non supervisé*. Une distinction peut être effectuée entre deux types de cartes auto organisatrices : Les cartes statiques, dont le nombre de neurones est fixé et déterminé *a priori*, par exemple, la carte de Kohonen proposée en 1982 [3]. Les cartes dynamiques, ou constructives pour lesquelles le nombre de neurones varie, à titre d'exemple, le réseau GNG proposé par Fritzik en 1995[4]. Ces cartes sont capables de partitionner un espace d'entrée en des sous ensembles sans aucune information *à priori.*

Dans ce mémoire, nous proposons deux cartes auto organisatrices et incrémentales basées sur le GNG. La première carte est un réseau de neurones incrémental à seuillage adaptatif, où nous calculons le seuil à partir de la base d'apprentissage et une valeur de précision fixée *à priori*. La deuxième carte est un réseau de neurones incrémental à deux couches. Dans ce modèle nous utilisons un paramètre d'utilité pour éliminer le bruit existant dans la base d'apprentissage.

Ce mémoire est organisé comme suit : le chapitre I est une introduction au système de reconnaissance de formes et les réseaux de neurones. Le chapitre II donne une description détaillée des réseaux de neurones incrémentaux existants. Le chapitre III décrit nos réseaux incrémentaux. Le chapitre IV présente les résultats expérimentaux des réseaux de neurones proposés. Nous terminerons par une conclusion générale qui résume les travaux effectués ainsi que les perspectives de recherche.

# chapitre 1

# Introduction

# à

# La reconnaissance de formes

# &

# Les réseaux de neurones

# I- Introduction

L’être humain cherche toujours à faciliter sa vie et la rend plus sécurisée. Il développe l’ordinateur pour automatiser le traitement des données.

Maintenant l’être humain cherche à remplacer l’homme par une machine, surtout dans les situations de travail dangereuses comme un centre nucléaire, et les tâches où l’homme prend beaucoup de temps pour les effectuer.

La première fonction qui doit être automatisée est la vision, où l’homme a la capacité de voir les formes et de les reconnaître.

Bien que l’homme sépare facilement les formes, il perd beaucoup de temps si le nombre de formes est trop élevé (par exemple, la recherche d’un mot donné dans un long texte). Donc l’être humain doit construire un système capable de capturer une forme et de la classer. Ce système est appelé un système de reconnaissance des formes.

Il existe deux principales approches pour la reconnaissance des formes : l’approche structurelle et l’approche statistique. Mais dans les dernières années les chercheurs se sont inspirés des approches de la biologie, tel que les réseaux de neurones.

Les réseaux neuronaux sont utilisés dans divers domaines dans les cadres d'applications commerciales, scientifiques, industrielles ou médicales. Les raisons de leur succès et de leur utilisation pour résoudre des problèmes sont liées à leurs caractéristiques intrinsèques suivantes :

1. L'efficacité avec laquelle les réseaux neuronaux arrivent à représenter les connaissances d'un domaine durant l'apprentissage.
2. La facilité d'utilisation et la qualité de la représentation de la connaissance qui permet de généraliser les solutions sur des exemples qui n'ont pas été vus durant l'apprentissage.

Dans ce chapitre, nous présenterons le schéma général d’un système de reconnaissance des formes. Nous détaillerons ses différentes étapes. Ensuite, nous présenterons les définitions de base liées aux réseaux de neurones avec la présentation de différents types de réseaux de neurones avec des exemples pour chaque type. Enfin, nous terminerons ce chapitre avec une conclusion.

# II- Reconnaissance de formes

## 2-1- Schéma général d'un système de reconnaissance des formes

Un système de reconnaissance de formes est représente par Fig.1.1 [5]

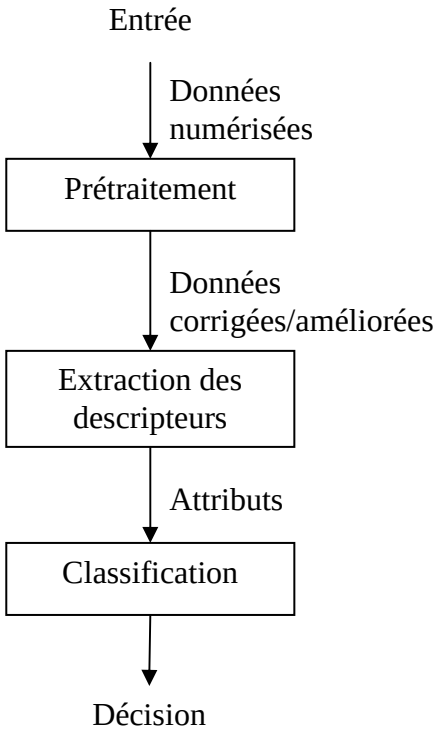



***Fig.1.1. Les étapes d'un système de reconnaissance de formes.***

L'entrée du système acquiert soit une forme ou un signal par un capteur (appareil photo, scanner, magnétophone, etc…). Cette entrée peut être une image scannée ou un signal vocal qui sera stocké dans un fichier. Par la suite plusieurs traitements sont opérés sur ces images et ces fichiers. Le but de ces prétraitements est d'éliminer les phénomènes qui provoquent une dégradation des performances du système, de réduire les bruits de quantification et de préserver la connexité des composantes connexes dans l'image. Le résultat de cette phase permettra d'extraire ou de mettre en évidence des particularités locales ou globales. Cette étape permet d'engendrer pour chaque image, un vecteur de primitives qui sert comme entrée au module dédie à la classification [5].

## 2-2- Les différentes étapes d'un système de reconnaissance de formes

### 2-2-1- Le prétraitement

Les données brutes issues des capteurs sont les représentations initiales des données à partir desquelles des traitements permettent de construire celles qui seront utilisées pour la

reconnaissance. Les données brutes sont bruitées, elles contiennent des informations parasites, et elles n'explicitent pas les informations utiles pour la reconnaissance.

Pour les prétraitements, le concepteur s'aide des connaissances qu'il possède sur les capteurs, les types de données, le problème posé et les méthodes d'apprentissage et de reconnaissance qu'il utilisera. Les prétraitements sont utiles pour éliminer des bruits qui peuvent être dus au capteur ou à des interférences avec d'autres sources de signaux (la parole en milieu sonore, l'encre du verso qui traverse le papier et dont la trace est visible sur la feuille du manuscrit, les fonds imagés des chèques, etc.).

Le traitement du signal constitue les principales sources pour les méthodes de prétraitement : filtrage (des bruits hautes fréquences, par exemple), amélioration des contrastes, extraction de contours ou de squelettes, modélisation du signal temporel ou extraction des primitives, etc.

### 2-2-2- Extraction des descripteurs

L'objectif de l'extraction des descripteurs (caractéristiques) dans le domaine de la reconnaissance consiste à exprimer les primitives sous une forme numérique ou symbolique appelée *codage*.

On appelle caractéristique (ou descripteur) une information qui peut être mesurée sur la donnée à reconnaître. Par exemple : l'amplitude moyenne d'un signal sur une fenêtre temporelle, l'énergie dans une bande de fréquence, le rapport hauteur sur largeur d'un caractère manuscrit ou le niveau de gris moyen d'une zone d'image etc.

On appelle primitive une composante élémentaire d'une forme. Les primitives ne sont pas décomposables. Par exemple : un segment de droite, une boucle ... . Le vecteur des $n$ descripteurs représente un point dans le nouvel espace à $n$ dimensions. Cette étape de la reconnaissance consiste à extraire des caractéristiques permettant de décrire de façon non équivoque les formes appartenant à une même classe de formes tout en les différenciant des autres classes. Le choix des descripteurs dépend énormément du problème posé. Malheureusement, il n'y a pas de théorie pour guider le choix des descripteurs. Ceci reste pour l'instant dans le domaine de l'art plus que dans celui de la science. [6] formule les desiderata suivants à propos des descripteurs :

***Discriminanabilité :*** un bon descripteur doit avoir des valeurs significativement différentes pour des formes appartenant à des classes différentes.

***Fiabilité :*** un bon descripteur doit avoir des valeurs très similaires (à faible variabilité) pour les formes d'une même classe. Par exemple, le diamètre peut être un descripteur intéressant pour classer des pièces de monnaies mais ne sera pas fiable pour distinguer différents arbres.

***Indépendance :*** dans un ensemble de descripteurs choisis pour un problème donné, un descripteur quelconque ne doit pas dépendre d'un autre descripteur. Par exemple, il ne faut pas utiliser le rayon et le diamètre comme deux descripteurs distincts.

***Nombre :*** le coût, la complexité et les exigences en temps de calcul d'un système de reconnaissance de formes croissent avec le nombre de descripteurs utilisés. Il faut donc choisir les meilleurs descripteurs pour diminuer ce nombre tout en satisfaisant la probabilité d'erreur acceptable, fixée à l'avance.

### 2-2-3- Classification

Classer un ensemble d'objets, c'est attribuer à chacun une classe (ou une catégorie) parmi plusieurs classes définies à l'avance. Cette tâche est appelée classification ou discrimination. Un algorithme qui réalise automatiquement une classification est appelé classifieur.

Les statisticiens appellent aussi classification la tâche qui consiste à regrouper des données qui se ressemblent dans des classes qui ne sont pas définies à l'avance [2].

Avant qu'un classifieur ne soit intégré dans un système de reconnaissance des formes, il faut avoir procédé auparavant à deux étapes : l'étape d'*apprentissage* et l'étape de *test*.

***a- L'étape d'apprentissage***

L'étape d'apprentissage consiste à caractériser les classes de formes de manière à bien distinguer les familles homogènes de formes. C'est une étape clé dans le système de reconnaissance. On distingue deux types d'apprentissage : apprentissage *supervisé* et apprentissage *non supervisé* [5].

Dans le cas de l'apprentissage supervisé, un échantillon représentatif de l'ensemble des formes à reconnaître est fourni au module d'apprentissage. Chaque forme est étiquetée par un opérateur appelé professeur. Cette étiquette permet d'indiquer au module d'apprentissage la classe dans laquelle le professeur souhaite que la forme soit rangée. Cette phase d'apprentissage consiste à analyser les ressemblances entre les éléments d'une même classe et les dissemblances entre les éléments de classes différentes pour en déduire la meilleure partition de l'espace des représentations. Les paramètres décrivant cette partition sont stockés

dans une table d'apprentissage à laquelle le module de décision se référera ensuite pour classer les formes qui lui sont présentées.

Dans le cas de l'apprentissage non supervisé, on fournit au système de reconnaissance un grand nombre de formes non étiquetées. L'étape de la classification va se charger d'identifier automatiquement les formes appartenant à une même classe.

***b- L'étape de test***

L'étape de test permet d'évaluer la performance du classifieur pour un apprentissage donné. C'est une étape importante car elle peut mettre en cause le choix des primitives ou le choix de la méthode d'apprentissage. En effet, il est difficile de trouver *a priori* les primitives pertinentes et la méthode d'apprentissage la plus adaptée au problème posé d'où l'utilité de procéder par itérations successives. Ces itérations consistent à extraire des primitives jugées utiles au problème de reconnaissance à résoudre et de tester la performance du système avec cet ensemble de primitives. Au fur et à mesure que les performances du système souhaitées ne sont pas atteintes alors il faut trouver à nouveau une nouvelle famille de primitives ou de combiner les primitives extraites avec de nouvelles primitives.

## 2-3- Les méthodes de classification

### 2-3-1- Approches Statistiques

Des approches classiques en reconnaissance des formes sont fondées sur l'étude statistique des mesures que l'on a effectuées sur les objets à reconnaître. L'étude de leur répartition dans un espace métrique et la caractérisation statistique des classes permet de prendre une décision de reconnaissance du type « plus forte probabilité d'appartenance à une classe ». Ces méthodes s'appuient en général sur des familles d'objets analogues dans l'espace de représentation [7].

Les deux principales familles de méthodes utilisées sont les méthodes *paramétriques* et les méthodes *non paramétriques*.

***a- Les méthodes paramétriques***

Les méthodes paramétriques opèrent sous l'hypothèse que les classes étudiées suivent une distribution de probabilité d'une certaine forme connue *a priori*. La prise de décision consiste à déterminer la classe pour laquelle la forme inconnue présente la probabilité d'appartenance maximale. Elles exigent des bases d'apprentissage assez importantes pour une estimation correcte des paramètres de la distribution supposée. Par exemple: la règle de Bayes, et les chaînes de Markov.

***b- Les méthodes non paramétriques***

Dans le cas des méthodes non paramétriques, les lois de probabilité sont inconnues pour une des classes. Le problème revient à établir des frontières de décision entre les classes. Les techniques les plus utilisées en reconnaissance de formes sont : la méthode du plus proche voisin.

## 2-3-2- Approches Structurelles (syntaxiques)

Ces sont des approches qui sont basées sur l'extraction de primitives en prenant compte de l'information structurelle. Ces approches cherchent à structurer l'information en décrivant l'organisation topologique (la structure) de la forme à partir de ses composantes les plus élémentaires. Ces approches nécessitent une mesure de la similarité entre deux représentations structurelles. On distingue plusieurs techniques telles que les structures de graphes, les structures syntaxiques et les arbres des décisions [7].

## 2-3-3- Approches bio inspirées

Dans les dernières années, les chercheurs optimisaient les algorithmes classiques par des algorithmes inspirés de la biologie comme les algorithmes génétiques et les réseaux de neurones.

***a- Les algorithmes génétiques***

Les algorithmes génétiques, connus généralement sous le nom d'algorithmes évolutionnaires, sont des méthodes adaptatives utilisées dans les problèmes d'optimisation. Ils sont basés sur le processus d'évolution génétique des organismes biologiques à travers des générations selon la théorie de l'évolution de Darwin[1].

Ces organismes vivent ensemble dans un environnement où ils se reproduisent et partagent les mêmes ressources telles que la nourriture et les abris contre les prédateurs. Ainsi, les individus les plus forts ont plus de chance de survivre vis-à-vis de la nourriture et des prédateurs, ou bien de trouver un partenaire pour se reproduire. Ceux qui ont réussi à survivre et à trouver des partenaires vont générer un nombre relativement plus important de progénitures. Les plus faibles vont avoir peu ou pas de descendants. De ce fait, les gènes des individus les plus adaptés vont se transmettre dans plusieurs individus des générations suivantes. La combinaison des meilleurs gènes des différents ancêtres peut parfois produire

[1] *la métaphore Darwinienne : Les individus d'une espèce évoluent par reproduction et sélection pour maximiser leur performance dans leur environnement.*

des "super-individus" qui s'adaptent encore mieux que leurs parents. Ainsi, les espèces évoluent et deviennent de plus en plus adaptés à leur environnement.

En imitant ce principe, les algorithmes évolutionnaire appliqués à un problème d'optimisation font évoluer un ensemble de solutions candidates, appelé *population* d'individus. Un individu représente une solution possible du problème donné. A chaque individu est attribué une fonction d'adaptation appelée "*fitness*" qui mesure la qualité de la solution qu'il représente, souvent c'est la valeur de la fonction à optimiser (fonction objective). Ensuite, une nouvelle population des solutions possibles est produite en sélectionnant les parents parmi les meilleurs de la "*génération*" actuelle pour effectuer des *croisements* et des *mutations*. La nouvelle population contient une plus grande proportion de caractéristiques des meilleurs individus de la génération précédente. De cette façon, de génération en génération, les meilleurs gènes se propagent dans la population, en se combinant ou en échangeant les meilleurs traits.

En favorisant les meilleurs individus, les régions les plus prometteuses de l'espace de recherche sont explorées. Si l'algorithme est bien conçu (codage, fonction d'évaluation des individus et d'autres paramètres judicieusement choisis), la population convergera vers un état stationnaire minimal [8].

***Principe des Algorithmes Evolutionnistes***

Les algorithmes évolutionnistes partagent un paradigme commun. Leurs caractéristiques principales sont les suivantes (voir fig.1.2)[8]:

- ✓ Une représentation génétique du problème, c'est-à-dire un codage approprié des solutions sous forme d'individus (ou génotype);
- ✓ Une fonction d'évaluation pour sélectionner les individus selon leur "force" (*fitness)*;
- ✓ Un mode de sélection des individus à reproduire;
- ✓ Des opérateurs génétiques pour modifier les représentations associées aux individus;
- ✓ Des paramètres internes de l'algorithme (ex: taille de la population, probabilité d'application de chaque opérateur...).

La recherche s'effectue à partir d'une population de points; la progression d'un état à un autre s'effectue par des opérateurs stochastiques tels que la sélection, le croisement et la mutation.

***Opérateurs Génétiques [8]***

*Croisement* est un échange par *blocs* d'éléments entre deux chaînes pour en générer un ou deux autres. Il existe plusieurs façons de procéder à cet échange selon les contraintes du problème traité.

Début

1- **Initialisation**
2- **Evaluation**
3- TANT que (critère d'arrêt=FAUX)
4- **Evolution**
   a. **Sélection**
   b. **Recombinaison** (croisement, mutation)
   c. **Evaluation**
5- Fin Tant que;

Fin.

***Fig.1.2. Forme générale d'un algorithme évolutionniste.***

*Mutation* est une forme de l'adaptation génétique de l'individu à son environnement. Elle se manifeste par une modification brutale d'un gène. Le phénomène se produit très rarement dans la nature (de 0,1% à 5% de la population).

## 2-3-4- Séparateurs à large marge (Support Vector Machines SVM)

Les SVM sont une méthode de classification supervisée. Introduite par Vapnik en 1996, cette méthode consiste à séparer les données en trouvant le séparateur dont la marge est la plus grande possible, d'où la seconde signification des initiales SVM : « Séparateur à Vaste Marge » [9].

***a- Cas linéaire et séparable***

Supposons dans un premier temps que si l'échantillon ($x_i$, $y_i$), $i$=1,…, $l$ (où $y_i = \pm 1$ donne la classe de chaque exemple) est linéairement séparable ; il existe alors un hyperplan $w.x+b$ tel que

$$w_i x_i + b \geq 1 \quad \text{si} \quad y_i = 1 \tag{1.1}$$

$$w_i x_i + b \leq -1 \quad \text{si} \quad y_i = -1 \tag{1.2}$$

Alors la dimension VC $h$ de la famille $f(x;\alpha) = wx + b$ est bornée :

$$h \leq \min\left(r^2 A^2, n\right) + 1$$

*n est* la dimension des entrées *x*, *r* le rayon de la plus petite sphère entourant tous les points $x_i$, *et A* une constante telle que $\|w\| \leq A$.

$\Rightarrow$ Chercher l'hyperplan qui minimise $\|w\|$

L'hyperplan qui minimise $\|w\|$ est celui qui maximise la marge. Pour trouver cet hyperplan, on peut minimiser $\Psi = \frac{1}{2} w.w$ sous les contraintes $y_i . f(x_i, w, b) \geq 1, i = 1 \cdots l$, qui résument les inégalités (1.1) et (1.2). Pour cela, on écrit le Lagrangien

$$L(w, b, \alpha) = \Psi - \sum \alpha_i \{ f(x, w, b) - 1 \} \tag{1.3}$$

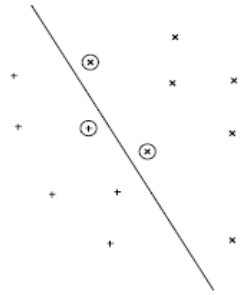


***Fig.1.3. Séparation linéaire optimale. Les trois vecteurs de support sont en cerclés [10].***

En annulant les dérivées partielles de *L* par rapport aux trois ensembles de paramètres, on déduit directement que la solution $w_0$ est une combinaison linéaire des exemples $x_i$, $w_0 = \sum y_i \alpha_i x_i$ et que les multiplicateurs de Lagrange $\Lambda = (\alpha_i)$ maximisent la fonctionnelle :

$$Q(\Lambda) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j x_i x_j \tag{1.4}$$

Sous les contraintes : $\sum_{i=1}^{l} \alpha_i^0 y_i$ et $\alpha_i \geq 0$.

Chaque point $x_i$ de l'ensemble d'apprentissage est associé à un multiplicateur $\alpha_i \geq 0$. Seuls les points tels que $\alpha_i \neq 0$ interviennent dans la solution $w_0$. Ce sont les vecteurs de support. La fonction *f* implémentée par une SVM s'écrit donc :

$$f(x) = w_0 . x + b = \sum_{i \in SV} x_i x + b \tag{1.5}$$

***b- Extension au cas non linéairement séparable***

Dans la dérivation précédente, nous avons supposé que les exemples étaient linéairement séparables. Lorsque cela n'est pas le cas, il faut relâcher les contraintes (1.1) et (1.2) en introduisant des variables d'écart $\xi_i \geq 0$. On écrit :

$$w_i x_i + b \geq 1 - \xi_i \quad \text{si} \quad y_i = 1 \tag{1.6}$$

$$w_i x_i + b \leq -1 + \xi_i \quad \text{si} \quad y_i = -1 \tag{1.7}$$

Les écarts permettent à certains points de se situer du mauvais coté de la frontière. Il faut alors minimiser $\sum_i \xi_i$, et on montre [11] que l'on obtient alors la même solution que précédemment, avec une contrainte supplémentaire $\alpha_i \leq C$, où $C$ est une constante positive qui permet de poser l'importance que l'on accorde a priori aux écarts.

Dans la formulation précédente, les exemples $x_i$ interviennent uniquement via des produits scalaires $x_i.x$.

Idée: utiliser un produit scalaire non linéaire $K(x,y)$ pour obtenir :

$$f(x) = \sum_{i \in SV} K(x_i, x) + b \tag{1.8}$$

Exemples des noyaux :

- ✓ Polynômes $K(x, y) = (x.y + 1)^p$
- ✓ Fonctions radiales $K(x, y) = \exp\left(\frac{(x-y)^2}{\sigma^2}\right)$
- ✓ Sigmoïde $K(x, y) = \tanh(bx.y - c)$

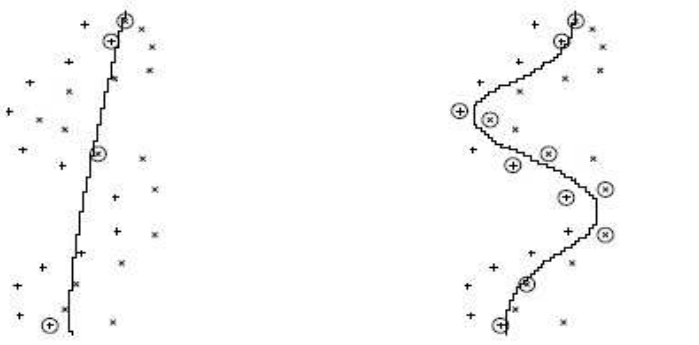



***Fig.1.4. Deux classes avec une frontière très non linéaire. A gauche, solution avec polynôme de degré 2, qui laisse plusieurs erreurs. A droite, tous les points sont séparés en utilisant un noyau polynomial de degré 7 [10].***

# III- Les réseaux de neurones

## 3-1- Définitions de base

### 3-1-1- Le neurone biologique

Le neurone, comme toute cellule est constitue de :

- ✓ **Un *corps cellulaires*** (ou *soma*), qui contient son noyau et où se déroulent les activités propres à sa vie cellulaire.
- ✓ **Un *axone* et de *dendrites*** sont structures spécialisées dans la communication avec les autres neurones. Cette communication entre cellules nerveuses s'effectue via des impulsions nerveuses. Les impulsions sont générées à l'extrémité somatique de l'axone et vont vers les terminaisons *axonales*. Là, elles affecteront tous les neurones reliés au neurone générateur, par l'intermédiaire de jonctions entre les terminaisons *axonales* et les autres cellules. Cette jonction est appelée *synapse*.

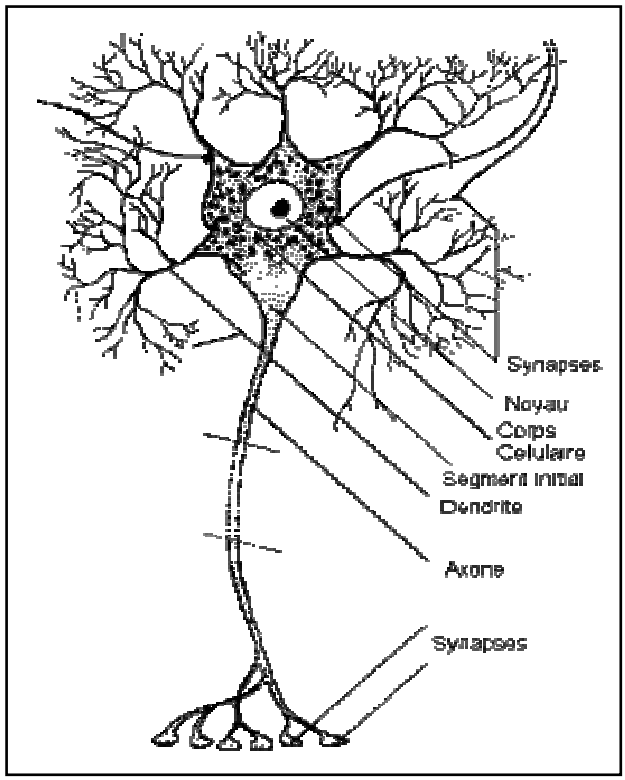



***Fig.1.5. Exemple de Neurone Biologique***

***Caractéristiques du système nerveux***

- ✓ Robuste et tolérant aux erreurs et aux "pannes".
- ✓ Flexible et doté de capacités d'apprentissage, pas besoin d'être explicitement "programmé" (plasticité synaptique).

- ✓ Capable de traiter de l'information partielle (floue), incertaine (probabiliste) ou incomplète.
- ✓ Très massivement parallèle.
- ✓ Une grande capacité de stockage.

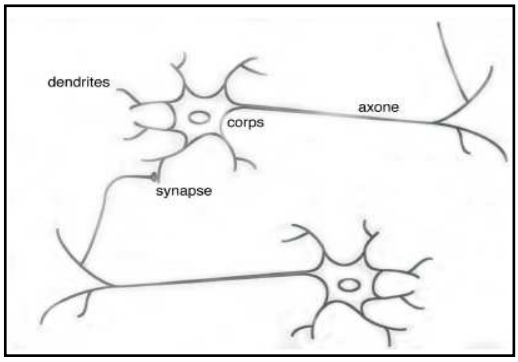



***Fig.1.6. La connexion entre deux neurones biologiques.***

## 3-1-2- Le neurone formel

Les auteurs dans [2] donnent la définition suivante :

Un neurone est une fonction non linéaire dont les paramètres sont à valeurs bornées.

***Notations***

*$x_i$ : les variables d'entrées d'un neurone(stimules),$X_j=[x_1\ x_2 \ldots x_n]'$, c'est le vecteur d'entrées du neurone j.*

*$w_i$: les paramètres du neurone (poids), $W_j=[w_1\ w_2 \ldots w_n]'$, c'est le vecteur de poids du neurone j.*

*f : la fonction d'activation.*

*$y_j$: la sortie du neurone j.*

*$f(X_j,W_j)=y_j$, il existe plusieurs types de la fonction f(sigmoïde, tangente,…)(voir Tab.1.1) .*

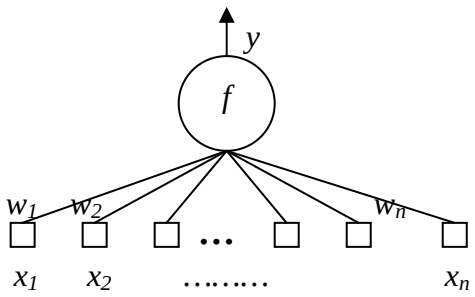



***Fig.1.7. Un neurone formel.***

| Nom de la fonction | Relation d'entrée/sortie | Icône | Nom Matlab |
|---|---|---|---|
| seuil | $a = 0 \quad \text{si } n < 0$<br>$a = 1 \quad \text{si } n \geq 0$ |  | hardlim |
| seuil symétrique | $a = -1 \quad \text{si } n < 0$<br>$a = 1 \quad \text{si } n \geq 0$ |  | hardlims |
| linéaire | $a = n$ |  | purelin |
| linéaire saturée | $a = 0 \quad \text{si } n < 0$<br>$a = n \quad \text{si } 0 \leq n \leq 1$<br>$a = 1 \quad \text{si } n > 1$ |  | satlin |
| linéaire saturée symétrique | $a = -1 \quad \text{si } n < -1$<br>$a = n \quad \text{si } -1 \leq n \leq 1$<br>$a = 1 \quad \text{si } n > 1$ |  | satlins |
| linéaire positive | $a = 0 \quad \text{si } n < 0$<br>$a = n \quad \text{si } n \geq 0$ |  | poslin |
| sigmoïde | $a = \frac{1}{1+\exp^{-n}}$ |  | logsig |
| tangente hyperbolique | $a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$ |  | tansig |
| compétitive | $a = 1 \quad \text{si } n \text{ maximum}$<br>$a = 0 \quad \text{autrement}$ |  | compet |

***Tab.1.1. Fonctions de transferts a=f(n).***

## 3-1-3- Les réseaux de neurones

Un neurone réalise simplement une fonction non linéaire, paramétrée, par ses variables d'entrées. L'intérêt des neurones réside dans les propriétés qui résultent de leur association en réseaux, c'est-à-dire de la composition des fonctions non linéaires réalisées par chacun des neurones.

## 3-1-4- L'apprentissage

On appelle *apprentissage* des réseaux de neurones la *procédure* qui consiste à *estimer les paramètres* des neurones du réseau, afin que celui-ci remplisse au mieux la tâche qui lui est affectée [2]. Dans le cadre de cette définition on peut distinguer deux type d'apprentissage : l'apprentissage *supervisée* et l'apprentissage *non supervisée*.

***a- L'apprentissage supervisée***

L'apprentissage dit *supervisé* est caractérisé par la présence d'un *professeur* qui possède une connaissance approfondie de l'environnement dans lequel évolue le réseau de neurones. En pratique, les connaissances de ce professeur prennent la forme d'un ensemble de $Q$ couples de vecteurs d'entrée et de sortie que nous noterons $\{(X_1, y_1), (X_2, y_2), \ldots, (X_Q, y_Q)\}$, où $X_j$ désigne une entrée et $y_j$ la cible pour cette entrée, c'est-à-dire les sorties désirées du réseau. Chaque couple $(X_j, y_j)$ correspond donc à un cas d'espèce de ce que le réseau devrait produire (la cible) pour une entrée donnée. Pour cette raison, l'apprentissage supervisé est aussi qualifié d'apprentissage par des exemples.

***b- L'apprentissage non supervisée***

L'apprentissage dit non supervisée ou encore «auto organisée» est caractérisé par *l'absence* complète de professeur, Nous ne disposons donc que d'un environnement qui fournit des entrées, et d'un réseau qui doit apprendre sans intervention externe. En assimilant les entrées de l'environnement à une description de son état interne, la tâche du réseau est alors de modéliser cet état le mieux possible. Pour y arriver, il importe d'abord de définir une mesure de la qualité pour ce modèle, et de s'en servir par la suite pour optimiser les paramètres libres du réseau, c'est-à-dire ses poids. A la fin de l'apprentissage, le réseau a développé une habilité à former des représentations internes des entrées de l'environnement permettant d'encoder les caractéristiques de ceux-ci et, par conséquent, de créer automatiquement des classes d'entrée similaires.

***c- L'apprentissage incrémental***

Les auteurs dans [12] définissent l'apprentissage incrémentale par la possibilité d'améliorer les performances d'un classifieur en reprenant une phase d'apprentissage lorsque des exemples supplémentaires apportant un complément d'information devient disponible. Les méthodes classiques prennent généralement pour hypothèse que la phase de recueil de données constituant la base d'apprentissage a lieu avant toute construction du classifieur et que la base constituée est « statistiquement significative », c'est-à-dire qu'elle contient un nombre suffisant d'exemples pour approcher les caractéristiques statistiques de la population à classer. Si toute fois ces caractéristiques sont inconnues, on se contente de recommander de prendre une base d'apprentissage « la plus grande possible » et on confirme la validité des résultats en utilisant une base de test contenant des données aux caractéristiques statistiques analogues (pouvant être obtenues par extraction aléatoire de la base d'apprentissage initiale avant apprentissage).

L'apprentissage incrémental présente l'avantage d'éliminer a priori le problème de la représentativité de la base d'apprentissage.

La question qui se pose est, « si un réseau permet de prendre en compte de nouvelles données, mais qu'advient-il des connaissances déjà acquises ? ».

Dans l'apprentissage incrémentale, un réseau de neurones doit apprendre des nouvelles données sans détériorer des données déjà acquises. Ce problème est connu sous le nom de dilemme « plasticité/stabilité » [15]:

- ✓ ***Plasticité :*** capacité à assimiler de nouvelles données.

- ✓ ***Stabilité :*** capacité à ne pas oublier les données déjà apprises.

***d- L'apprentissage compétitif***

L'apprentissage compétitif, comme son nom l'indique, consiste à faire compètitionner les neurones d'un réseau pour déterminer lequel sera actif à un instant donné. Contrairement aux autres types d'apprentissage où, généralement, tous les neurones peuvent apprendre simultanément et de la même manière, l'apprentissage compétitif produit un «vainqueur» ainsi que, parfois, un ensemble de neurones «voisins» du vainqueur. Et seuls ce vainqueur et, potentiellement, son voisinage bénéficient d'une adaptation de leur poids. On dit alors que l'apprentissage est local car limité à un sous-ensemble des neurones du réseau.

Une règle d'apprentissage compétitif comporte les éléments suivants :

- ✓ Un ensemble de neurones identiques (même type) sauf pour les valeurs de leurs poids synaptiques;
- ✓ Une limite imposée à la «force» d'un neurone ;
- ✓ Un mécanisme permettant aux neurones de compétitionner pour le droit de répondre à un certain sous-ensemble de stimuli d'entrée, de manière à ce qu'un seul neurone de sortie soit actif à la fois.

Ainsi, les neurones individuels peuvent apprendre à se spécialiser sur des sous-ensembles de stimuli similaires pour devenir des détecteurs de caractéristiques.

Dans leur forme la plus simple, les réseaux de neurones qui utilisent l'apprentissage compétitif sont souvent constitués d'une seule couche de neurones de sortie, totalement connectée sur les entrées. Un neurone vainqueur modifiera ses poids synaptiques en les rapprochant (géométriquement) d'un stimulus d'entrée p pour lequel il a battu tous les autres neurones lors de la compétition :

$$\Delta w = \begin{cases} \eta(p - w) & \textit{Si le neurone est vainqueur} \\ 0 & \textit{Autrement} \end{cases}$$

Où $0 < \eta < 1$ correspond à un taux d'apprentissage. Un neurone qui ne gagne pas la compétition ne modifiera aucunement ses poids. Il ne sera donc pas affecté par le stimulus en question. Parfois, on définit également un voisinage autour du neurone gagnant et on applique une règle similaire sur les voisins, mais avec un taux d'apprentissage différent :

$$\Delta w = \begin{cases} \eta_1 (p - w) & \textit{Si le neurone est vainqueur} \\ \eta_2 (p - w) & \textit{Si le neurone est voisin du vainqueur} \\ 0 & \textit{Autrement} \end{cases}$$

Avec $\eta_2 \leq \eta_1$.

L'apprentissage compétitif est surtout utilisé dans le contexte d'un apprentissage non supervisé.

## 3-2- Les types de réseaux de neurones

Les réseaux de neurones sont divisés en deux grandes familles, la première famille contient les réseaux de neurones non bouclés et les réseaux de neurones bouclés, et la deuxième famille contient les cartes auto organisatrices [2].

### 3-2-1- Les réseaux de neurones non bouclés

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

L'information circulant des entrées vers les sorties sans *retour en arrière* : si l'on représente le réseaux comme un graphe dont les nœuds sont les neurones et les arêtes les connections entre ceux-ci, le graphe d'un réseau non bouclé est *acyclique* [2].

Un exemple des réseaux de neurones non bouclé est les réseaux à couches sont souvent appelés perceptrons multicouche (ou MLP pour Multi-Layer Perceptron). Illustre par la figure Fig.1.8.

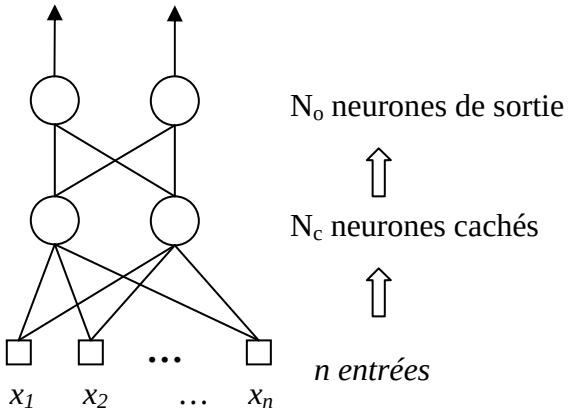



***Fig.1.8. Un réseau de neurones à n entrées, une couche de $N_c$ neurones cachés, et $N_o$ neurones de sorties.***

## 3-2-2- Les réseaux de neurones bouclés (ou récurrents)

Un réseau de neurones bouclé à temps discret réalise une (ou plusieurs) équation aux différences non linéaires, par composition des fonctions réalisées par chacun des neurones et des retards associés à chacune des connexions (voir Fig.1.9) [2].

Les chiffres dans les carrés indiquent le retard attaché à chaque connexion, multiple de l'unité de temps (ou période d'échantillonnage) T. le réseau contient un cycle, qui part du neurone 3, va au neurones 4, et revient au neurones 3.

Tout cycle du graphe des connexions d'un réseau de neurones bouclé doit comprendre au moins une connexion de retard non nul.

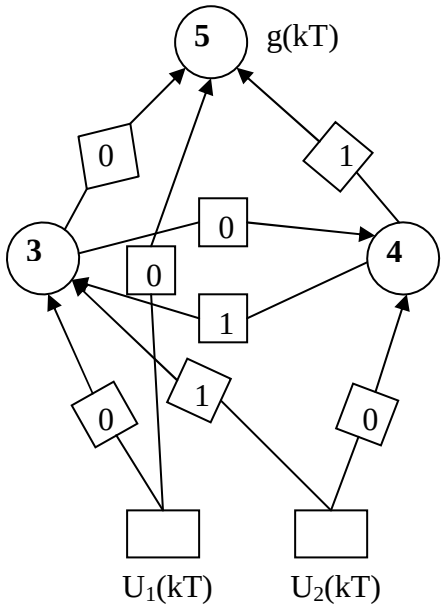



***Fig.1.9. Un réseau de neurones bouclé à deux entrées.***

## 3-2-3-Les cartes topologiques autos-organisatrices

Les cartes topologiques auto organisatrices font partie de la famille des modèles dits *apprentissage non supervisé.*

La particularité la plus importante des cartes auto organisatrices est qu'elles rendent possible la comparaison des groupements qui ont été réalisés directement à partir des données. Une observation est affectée à un groupe qui est projeté en un nœud de la carte. La comparaison des projections liées à deux observations distinctes permet d'estimer la proximité des groupes dont elles sont issues. Les observations « semblables » ont la même projection ; si les projections sont différentes, la dissemblance grandit avec la distance qui existe entre les projections ; cette distance est calculée sur la carte. Ainsi, l'espace des sous

ensembles s'identifie à la carte. Et il est possible, d'une certaine manière, de regarder simultanément l'espace des sous ensembles et celui des observations.

Une distinction peut être effectuée entre deux types de cartes auto organisatrices :

- ✓ Les cartes statiques, dont le nombre de neurones est fixé et déterminé *a priori*, par exemple, la carte de Kohonen proposée en 1982 [3].
- ✓ Les cartes dynamiques, ou constructives, pour lesquelles le nombre de neurones varie, par exemple, le réseau GNG proposé par Fritzik en 1995[4] .

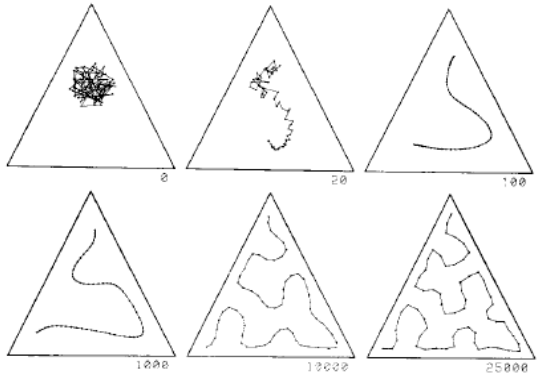



***Fig.1.10. Exemple d'une carte auto organisée à une dimension. Les stimuli d'apprentissage sont distribués uniformément à l'intérieur d'un triangle.***

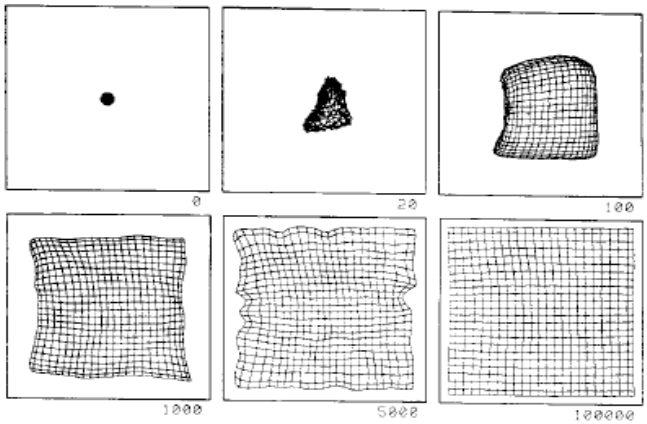



***Fig.1.11. Exemple d'une carte auto organisée à deux dimensions. Les stimuli d'apprentissage sont distribués uniformément à l'intérieur d'un carré.***

# IV- Conclusion

De manière générale, on peut dire que les systèmes de reconnaissance des formes sont des fonctions de classification qui associent à toute forme inconnue sa classe la plus probable.

Un système de reconnaissance des formes contient principalement trois modules. Un module de prétraitement de données qui est utile pour éliminer des bruits qui peuvent être dus au capteur ou à des interférences avec d'autres sources de signaux. Un module d'extraction

des descripteurs des données qui se charge de faciliter la tâche de reconnaissance en présentant les données en entrée sous une forme adaptée à la classification. Un module de classification qui va classer les formes inconnues. Avant qu'un classifieur ne soit intégré dans un système de reconnaissance des formes, il faut avoir procédé auparavant à deux étapes : l'étape d'*apprentissage* et l'étape de *test*.

Plusieurs approches ont été proposées pour la reconnaissance des formes. Les approches structurelles représentent les mécanismes de classification sous la forme d'une structure (par exemple les arbres). L'approche statistique représente chaque classe par une fonction qui renvoie pour chaque forme la probabilité qu'elle appartienne à la classe en question. Les approches bio-inspirées qui tentent de reproduire des mécanismes naturels, les plus populaires sont les réseaux neurones et les algorithmes génétiques. Dans les dernières années, une nouvelle approche est apparue ; c'est le séparateur à large marge (SVM). SVM est une méthode de classification supervisée binaire (séparer deux classes).

Un réseau de neurones est un système composé de plusieurs unités de calcul simples fonctionnant en parallèle, dont la fonction est déterminée par la structure du réseau, la solidité des connexions, et l'opération effectuée par les éléments ou noeuds. C'est ce que nous détaillerons dans le prochain chapitre.

# chapitre 2

# Les réseaux de neurones incrémentaux

# I- Introduction

Le clustering est un problème connu dans le domaine d'analyse de données [13]. Il sert à regrouper un ensemble d'objets ayant des caractéristiques homogènes en un nombre inconnu de classes. Parmi les méthodes de clustering, notre choix s'est porté sur les réseaux de neurones incrémentaux à apprentissage non supervisé. Il y'a trois raisons qui justifient l'utilisation de ce type de réseaux:

- ✓ Le nombre de classes est inconnu. Celui-ci est détecté automatiquement durant l'apprentissage avec ce type de réseaux.
- ✓ L'ensemble des données est représenté par des vecteurs de caractéristiques de dimension variable.
- ✓ Ces réseaux incrémentaux peuvent détecter cette dimension et la représenter. L'ensemble de données peut être incomplet c'est-à-dire une nouvelle classe peut être créée. Les réseaux incrémentaux peuvent ajouter des neurones (des classes).

Dans ce chapitre nous présenterons quatre réseaux constructifs, que nous comparons à la fin du chapitre.

# II- Growing Neural Gas (GNG)

## 2-1- Définition

Le réseau «Growing Neural Gas» [4](GNG) est un réseau constructif qui ne pose *a priori* aucune hypothèse sur la topologie de l'espace des entrées. Un réseau minimal est initialement créé et de nouveaux neurones ainsi que de nouvelles connexions entre les neurones sont ajoutés au fil de l'apprentissage non supervisé.

## 2-2- Principe

La topologie du réseau est représentée par un graphe $G = [N(t), C(t)]$ où $N(t)$ désigne l'ensemble des sommets du graphe au temps $t$ et $C(t)$ l'ensemble de ses arêtes. À chaque sommet est associé un neurone caractérisé par un vecteur de poids synaptiques $w_i$ (vecteur de référence) ainsi qu'un signal d'erreur $e_i$. Ce dernier servira à accumuler l'erreur de modélisation attribuable au neurone $i$ et guidera le choix de l'emplacement où nous ajouterons périodiquement de nouveaux sommets dans le graphe. Les arêtes du graphe, liant deux sommets $i$ et $j$, correspondent quant à elles à des connexions entre les neurones sous-jacents. À chaque connexion $\{i, j\}$ est associé un âge $a_{i,j}$ . Une connexion jeune implique une

vraisemblance élevée de la relation topologique, alors qu'au contraire, une connexion âgée signifie une vraisemblance faible de cette relation. Lorsque l'âge d'une connexion dépassera un certain seuil, celle-ci pourra mourir et disparaître du graphe. Comme nous le verrons plus loin, à la fois les connexions et les neurones peuvent apparaître et disparaître du graphe tout au long du processus d'apprentissage. C'est pourquoi les ensembles *N* et *C* dépendent tous les deux du temps.

L'algorithme du GNG comprend un certain nombre de paramètres essentiels. Tout d'abord, il y a $\tau$ qui définit la période de temps entre les ajouts de neurone, c'est-à-dire qu'à toutes les $\tau$ itérations, nous ajouterons un neurone quelque part dans le graphe. L'apprentissage du GNG, tout comme celui du Kohonen, est de type compétitif. Le neurone gagnant ainsi que ses voisins sont déplacés dans la direction d'un stimulus en proportion de sa distance et d'un certain taux d'apprentissage. Le GNG utilise deux taux distincts, un pour le neurone gagnant, $\eta_g$, et l'autre pour ses voisins immédiats, $\eta_v$. Contrairement au réseau de Kohonen, cependant, ces taux demeurent fixes tout au long de l'apprentissage. Également, le voisinage est fixé à 1, c'est-à-dire que seuls les voisins directs (ces sont les neurones qui ayant une connexion avec le neurone) du neurone gagnant se déplacent. Les connexions entre les neurones ne peuvent dépasser un âge maximum $a_{max}$, sinon elle meurent et disparaissent. Finalement, on utilise aussi un paramètre $\alpha$ pour contrôler l'oubli des signaux d'erreur associés aux neurones du réseau.

## 2-3- Algorithme

Initialisation : $N(0) = \{x, y\}$ et $C(0) = \{\{x, y\}\}$ avec $w_x(0)$ et $w_y(0)$ initialisés aléatoirement avec de petits poids synaptiques dans l'espace des entrées ; $e_x(0) = e_y(0) = a_{x,y} = 0$.

Fixer $t_{\max}, \tau, \eta_g, \eta_v, a_{\max}$, $\alpha_1$ et $\alpha_2$.

t=1.

Répéter tant que $t \le t_{\max}$ :

(1) Choisir aléatoirement une entrée $p(t)$ parmi l'ensemble d'apprentissage.

(2) Déterminer les deux neurones gagnants $g_1$ et $g_2$ les plus proche de $p(t)$ :

$$g_1 = \arg\min_{i \in N(t)} \|p(t) - w_i(t)\|$$

$$g_2 = \arg\min_{i \in N(t) - \{g_1\}} \|p(t) - w_i(t)\|$$

(3) Incrémenter les âges de toutes les connexions adjacentes à $g_1$ :

$$\forall \{i, g_1\} \in C(t) : a_{i,g_1} = a_{i,g_1} + 1$$

(4) Incrémenter l'erreur associée au premier gagnant :

$$e_{g_1} = e_{g_1} + \left\| p(t) - w_{g_1}(t) \right\|^2$$

(5) Déplacer les vecteurs de référence de $g_1$ et de ses voisins directs dans la direction de $p(t)$ :

$$\Delta w_{g_1}(t) = \eta_{g_1} \left[ p(t) - w_{g_1} \right]$$

$$\Delta w_v(t) = \eta_v \left[ p(t) - w_v \right], \; v \in \Lambda_{g_1}$$

Où $\Lambda_{g_1} = \left\{ i \in N(t) \middle| i \neq g_1 et \{i, g_1\} \in C(t) \right\}$

(6) Si $\{g_1, g_2\} \in C(t)$ alors $a_{g_1,g_2} = 0$ sinon

$$C(t) = C(t) \cup \{\{g_1, g_2\}\}, \quad a_{g_1,g_2} = 0 .$$

(7) Retirer de $C(t)$ toutes les connexions pour les quelles $a_{i,j} > a_{\max}$ ; retirer aussi tous les neurones qui se retrouveraient isolés suite à la mort de connexions.

(8) Si $t \bmod \tau = 0$ alors :

a. Déterminer le neurone $q$ possédant l'erreur maximum

$$q = \arg\max_{i \in N(t)} e_i$$

b. Déterminer le neurone $r$, voisin de $q$, possédant aussi l'erreur maximum :

$$r = \arg\max_{i \in \Lambda_q} e_i$$

c. Insérer un nouveau neurone $x$ à mi-chemin entre $q$ et $r$ :

$$w_x(t) = \frac{w_q(t) + w_r(t)}{2}, \qquad e_q = \alpha_1 e_q, \quad e_r = \alpha_1 e_r, \quad e_x = e_q$$

d. Remplacer la connexion $\{q, r\}$ par les deux connexions $\{q, x\}$ et $\{x, r\}$ avec

$$a_{q,x} = a_{x,r} = 0$$

(9) Réduire les signaux d'erreur de tous les neurones :

$$e_i = \alpha_1 e_i, \quad \forall i \in N(t)$$

(10) $t = t + 1$

(11) Fin tant que

La figure Fig.2.1 [4] illustre la capacité qu'a le GNG à apprendre la topologie des données. Pour cet exemple, les données d'apprentissage ont été générées à partir de quatre distributions uniformes : la première dans un volume tridimensionnel en forme de prisme rectangulaire, la deuxième sur une droite au bout de la surface et la dernière le long d'un anneau au bout du segment de droite. La figure Fig.2.1 montre le graphe du réseau à différents instants durant

l'apprentissage, Initialement (fig.2.1.(a)), on commence avec un réseau de deux neurones liés par une seule connexion. Au fil des itérations, des neurones sont ajoutés dans le graphe aux endroits où l'erreur de modification est maximum. A la fin, non seulement le réseau a appris à modéliser l'espace d'entrée, mais il a aussi réussi à apprendre la topologie des différentes formes échantillonnées. Les paramètres utilisés dans cet exemple sont $\tau = 100$, $\eta_g = 0.2$, $\eta_v = 0.006$, $\alpha_1 = 0.5$, $\alpha_2 = 0.995$, $a_{\max} = 50$.

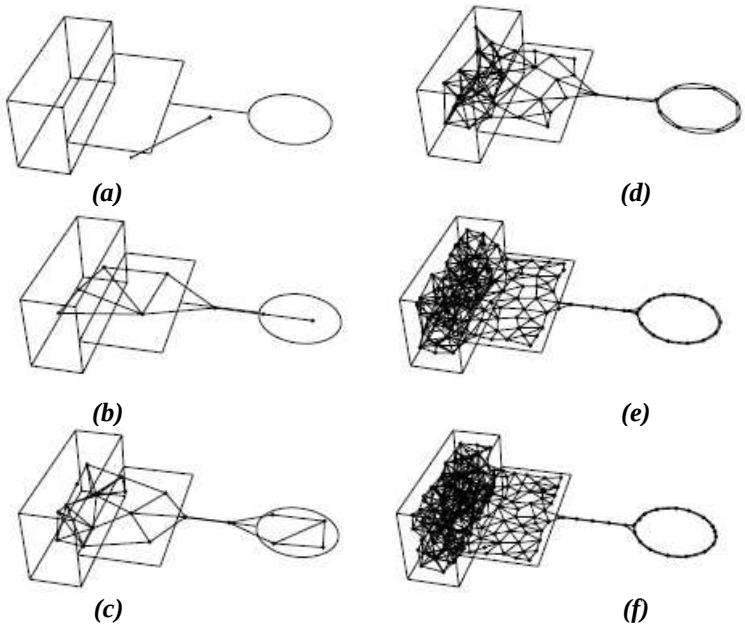


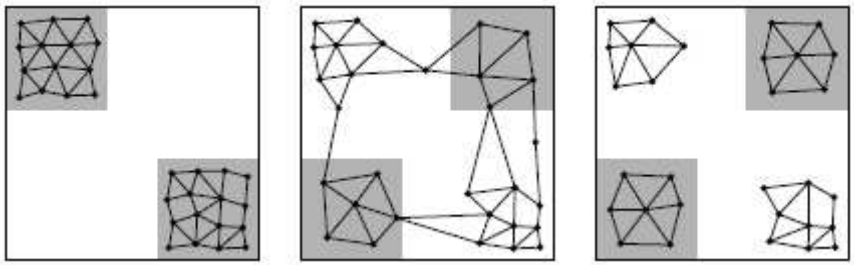

***Fig.2.1. Exemple d'un GNG entraîne sur des données générées à partir d'un espace d'entrée avec différentes dimensions dans des surfaces différentes. a) le réseau initial. b) le réseau après 600 itérations. c) le réseau après 1800 itérations, les figure (d), (e) et (f) montrent le réseau après 5000, 15000, 2000 respectivement [4].***

## 2-4- Discussion

Le GNG a une structure constructive, non fixée à priori. L'insertion des neurones dans le réseau GNG est faite chaque $\tau$ itérations et on ne supprime que les neurones isolés. Donc le nombre de neurones peut devenir très grand, ce qui conduit aux phénomènes de *sur apprentissage* (le dépassement de nombre de neurones). Pour cette raison, on limite le nombre de neurones avant l'apprentissage [14].

Dans le cas où la distribution des données est non stationnaire (cas d'apprentissage en ligne) c'est-à-dire les données changent leurs positions dans l'espace d'entrée pendant le processus d'apprentissage, le GNG est incapable de prendre correctement ce cas de figure. La figure Fig.2.2 montre comment le GNG prend une base d'apprentissage avec une distribution des données non stationnaire. La figure Fig.2.2.(c) montre que des neurones ont été adapté aux nouvelles régions.

Dans [14], Fritzik ajoute au GNG un paramètre pour supprimer les neurones inutiles, c'est la valeur locale d'utilisation, et à chaque itération adapter la valeur d'erreur de tous les neurones :



*(a)* *(b)* *(c)*

***Fig.2.2. La distribution non stationnaire d'espace d'entrées. a) initiale état de l'espace d'entrées avec le GNG. b) l'état du réseau après 5000 itérations. c) après 50000 itérations [14].***

### *Erreur locale*

La valeur de l'erreur locale du neurone gagnant s'adapte comme suit :

$$e_{g_1} = e_{g_1} + \left\| p(t) - w_{g_1}(t) \right\|^2$$

Dans chaque itération le vecteur de référence du neurone gagnant et ses voisins immédiats sont adaptés. Donc il faux adapter la valeur de l'erreur locale de tous les neurones par la formule suivante :

$e_c = e_c + \beta e_c$ Pour tout neurone *c*

Où $\beta$ est une constante prédéfinie.

### *La valeur locale de l'utilisation*

La valeur locale de l'utilisation du neurone gagnant s'adapte comme suit :

$$u_{g_1} = u_{g_1} + \left\| p(t) - w_{g_2}(t) \right\|^2 - \left\| p(t) - w_{g_1}(t) \right\|^2$$

Pour les autres neurones :

$u_c = u_c + \beta u_c$ Pour toute neurone *c*

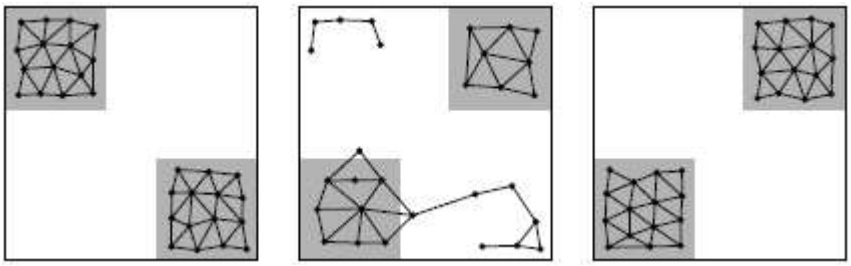
### *Le critère de suppression*

On supprime le neurone *c* qui a la valeur locale minimale d'utilisation à condition qu'elle vérifie le critère suivant :

$e_q / u_c > k$

Où $q = \arg\max_{i \in N(t)} e_i$ , $c = \arg\min_{i \in N(t)} u_i$ et $k$ constante prédéfinie.

Le nouveau GNG est appelé GNGU. La figure Fig.2.3 montre comment le GNGU apprend la même base de la figure Fig.2.2.



***(a) 20000 itérations*** ***(b) 25000 itérations*** ***(c) 40000 itérations***

***Fig.2.3. Le GNGU apprend la distribution non stationnaire d'espace d'entrées avec succès, k= 3.***

Pour le GNGU comme pour le GNG, il faux limiter le nombre de neurones avant l'apprentissage [14]. Cette approche est constructive et permet de prendre en compte de nouvelles données, mais qu'advient il des connaissance déjà acquises ?

Dans [15] les auteurs proposent un nouvel algorithme s'inspirant fortement de celui proposé par Fritzik mais qui comporte quelques avantages par rapport à celui –ci.

# III- Incremental Growing Network Gas (IGNG)

## 3-1- Définition

IGNG [15] est un modèle constructif et incrémental de cartes auto organisatrices. Comme le modèle Growing Neural Gas, le IGNG n'impose aucune contrainte sur la structure du réseau. Celui-ci est mis à jour de façon continue par un apprentissage Hebbien compétitif.

## 3-2- Principe

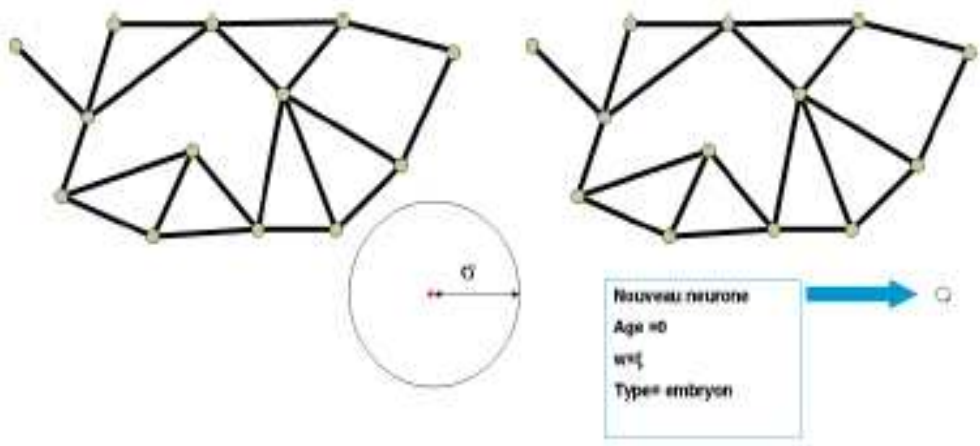
Chaque neurone possède un âge, $age_n$, un vecteur de référence, $w_n$, et un type (mature ou embryon), $type_n$. A chaque connexion est associé un âge. Une connexion jeune implique une vraisemblance élevée de la relation topologique, alors qu'au contraire une connexion âgée signifie une vraisemblance faible de cette relation. Lorsque l'âge d'une connexion dépassera un certain seuil, celle-ci pourra mourir et disparaître. Afin de construire le réseau, on effectue avant l'apprentissage de chaque donnée $p(t)$ un test défini par l'équation (2.1) où $T$ est un seuil fixé a priori. Ce test sert à vérifier si l'insertion d'un neurone est nécessaire. S'il n'existe

pas au moins deux neurones qui satisfont cette équation, alors l'ajout d'un nouveau neurone est effectué.

$$\|p(t) - w_n\| \leq T \qquad (2.1)$$

$w_n$ :vecteur de référence d'un neurone n

Quand un neurone est inséré, c'est un neurone de type embryon avec un âge nul. Initialement le réseau est vide. A chaque itération, nous cherchons le neurone gagnant $g_1$ . Si celui-ci n'existe pas (réseau vide) ou s'il ne satisfait pas la condition de l'équation (2.1), un nouveau neurone est ajouté avec $w_{new} = p(t)$ (voir Fig.2. 4 (a)).



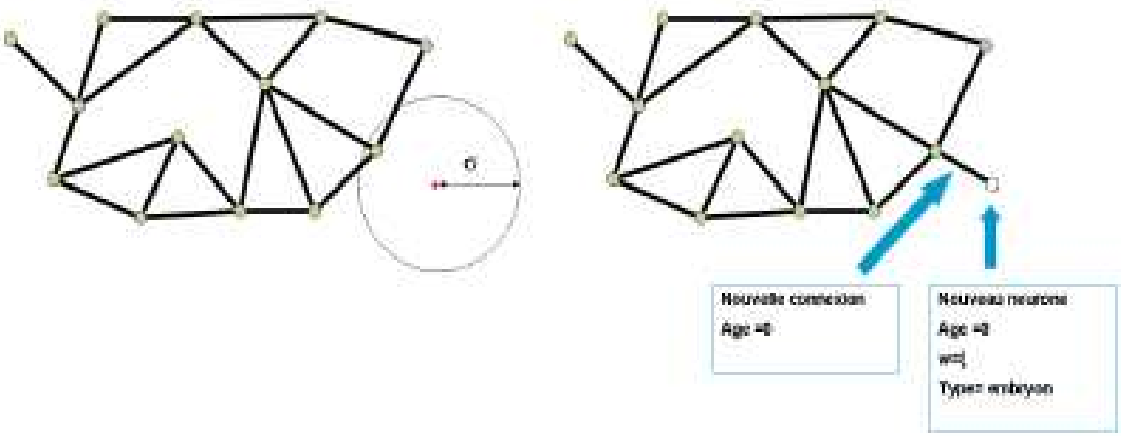

***(a) une nouvelle donnée*** ***(b) création d'un nouveau neurone***

***Fig.2.4. Insertion d'un neurone quand la donnée à apprendre est trop loin de son neurone gagnant.***

Un deuxième cas peut se produire quand un neurone gagnant $g_1$ satisfaisant l'équation (2.1) existe. Nous recherchons alors le deuxième neurone $g_2$ le plus proche de cette nouvelle donnée. Si $g_2$ n'existe pas ou s'il ne satisfait pas le test, un nouveau neurone est ajouté avec $w_{new} = p(t)$ et une connexion est créée entre ce neurone et $g_1$ (voir Fig. 2. 5).

Enfin, le dernier cas de figure correspond à l'existence d'au moins deux neurones qui satisfont le test de l'équation (2.1). Aucun neurone n'est ajouté dans cette situation et l'adaptation des vecteurs de référence se fait comme pour le GNG. L'âge des connexions émanant de $g_1$ est incrémenté. Comme dans l'apprentissage Hebbien compétitif, nous ajoutons une connexion entre les deux plus proches neurones de la donnée. Si celle-ci existe déjà, son âge est réinitialisé à zéro. Toutes les connexions ayant un âge supérieur à $a_{\max}$ sont supprimées, et si ceci a pour conséquence l'isolement d'un neurone, celui-ci est supprimé.

L'âge de tous les neurones connectés à $g_1$ est incrémenté, et tous les neurones embryons avec un âge supérieur à $a_{mature}$ deviennent des neurones matures.




***(a) une nouvelle données*** ***(b) création d'un nouveau neurone***

***Fig.2.5. Insertion d'un neurone quand la donnée à apprendre est trop loin de son deuxième neurone gagnant***

## 3-3- Algorithme

Initialisation : $N(0) = \{\ \}$ et $C(0) = \{\ \}$.

Fixer $\eta_g, \eta_v, a_{\max}, a_{mature}$ et $T$.

(1) Tant que *le critère d'arrêt n'est pas satisfait* faire

(2) Choisir aléatoirement une entrée $p(t)$ parmi l'ensemble d'apprentissage.

(3) Déterminer le premier neurone gagnant $g_1$ le plus proche de $p(t)$ :

$$g_1 = \arg\min_{i \in N(t)} \left\| p(t) - w_i(t) \right\|$$

(4) Si $\left\| p(t) - w_{g_1} \right\| > T$ alors :

a. Insérer un nouveau neurone $n_{new}$ avec $w_{new} = p(t)$.

b. Retourner à l'étape (1).

(5) Déterminer le deuxième neurone gagnant $g_2$ le plus proche de $p(t)$ :

$$g_2 = \arg\min_{i \in N(t) - \{g_1\}} \left\| p(t) - w_i(t) \right\|$$

(6) Si $\left\| p(t) - w_{g_2} \right\| > T$ alors :

a. Insérer un nouveau neurone $n_{new}$ avec $w_{new} = p(t)$.

b. Créer une connexion entre $g_1$ et le nouveau neurone :

$$C(t) = C(t) \cup \{\{g_1, n_{new}\}\}, \quad a_{g_1, n_{new}} = 0.$$

c. Retourner à l'étape (1).

(7) Incrémenter les âges de toutes les connexions adjacentes à $g_1$ :

$$\forall\{i, g_1\} \in C(t) : a_{i,g_1} = a_{i,g_1} + 1$$

(8) Déplacer les vecteurs de références de $g_1$ et de ses voisins directs dans la direction de $p(t)$ :

$$\Delta w_{g_1}(t) = \eta_{g_1}\left[p(t) - w_{g_1}\right]$$

$$\Delta w_v(t) = \eta_v\left[p(t) - w_v\right],\ v \in \Lambda_{g_1}$$

Où $\Lambda_{g_1} = \left\{i \in N(t) \middle| i \neq g_1 et \{i, g_1\} \in C(t)\right\}$

(9) Si $\{g_1, g_2\} \in C(t)$ alors $a_{g_1,g_2} = 0$ sinon

$$C(t) = C(t) \cup \{\{g_1, g_2\}\}, \quad a_{g_1,g_2} = 0.$$

(10) Retirer de $C(t)$ toutes les connexions pour les quelles $a_{i,j} > a_{\max}$ ; retirer aussi tous les neurones qui se retrouveraient isolés suite à la mort de connexions.

(11) Incrémenter l'âge de tous les neurones voisins directs du neurone gagnant $g_1$ :

$$age_v = age_v + 1, \quad v \in \Lambda_{g_1}$$

(12) Pour *chaque neurone embryon* faire :

Si $age_n \geq a_{mature}$ alors : $type_n = 'mature'$

(13) Fin tant que.

## 3-4- Discussion

La notion de neurones matures ou embryons a surtout été introduite pour doter le réseau IGNG d'une tolérance au bruit. Le paramètre $a_{mature}$ correspond donc au nombre d'activations qu'il faut à un neurone pour que celui-ci ne soit plus considéré comme résultant de données bruitées. Par conséquent, ce paramètre est à régler en fonction du bruit présent dans la base d'apprentissage.

*T* est le paramètre qui demande le plus de connaissances sur le problème à traiter. Il correspond au rayon de l'hyper sphère représentant le volume maximal dans l'espace de description que doit considérer un neurone. Il est le paramètre le plus important dans notre approche et doit encore malheureusement être fixé *a priori*.

L'algorithme proposé ne conduit pas aux phénomènes de *sur-apprentissage* rencontrés par les autres réseaux évolutifs, puisque l'ajout d'un nouveau neurone se fait seulement si c'est nécessaire.

La distribution de la figure Fig.2.6 est la même qui est utilisée pour tester les capacités du réseau GNG. Cet exemple montre que le réseau IGNG donne des résultats équivalents au

GNG pour la visualisation de la topologie des données dans le cadre d'un apprentissage passif.

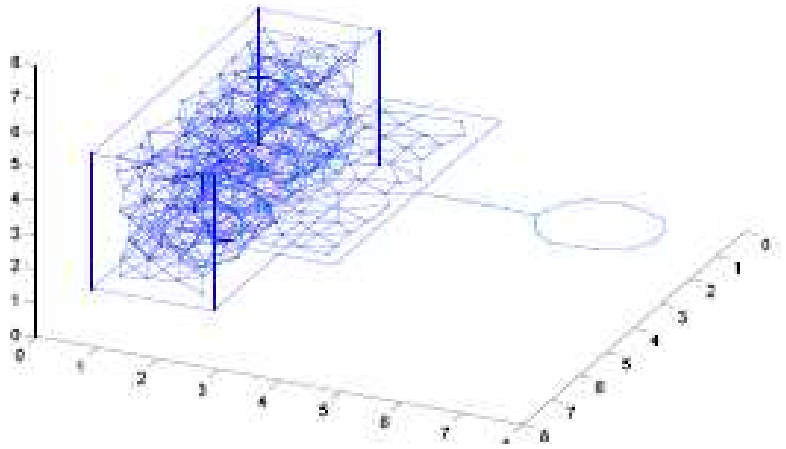


***Fig.2.6. Réseau IGNG pour la visualisation de données.***

Pour montrer les limites du réseau GNG pour la visualisation de la topologie dans le contexte d'un apprentissage incrémental, dans [15] les auteurs utilisent un problème synthétique composé de deux classes en deux dimensions avec une distribution sphérique pour la première classe et cubique pour la deuxième classe. L'apprentissage est effectué en deux étapes, une première étape où le réseau apprend la topologie de la première classe, puis dans un deuxième temps, les données de la deuxième classe sont présentées au réseau.

La figure Fig.2.7 nous montre que pour pouvoir apprendre la topologie de la deuxième classe (distribution cubique), le réseau GNG a détérioré la connaissance acquise sur la topologie de la première classe (distribution sphérique). Ce problème est connu sous le nom de dilemme (plasticité/stabilité). Contrairement au réseau GNG, le réseau IGNG se comporte parfaitement et préserve bien les connaissances déjà modélisées. Il répond donc bien au dilemme (plasticité/stabilité).

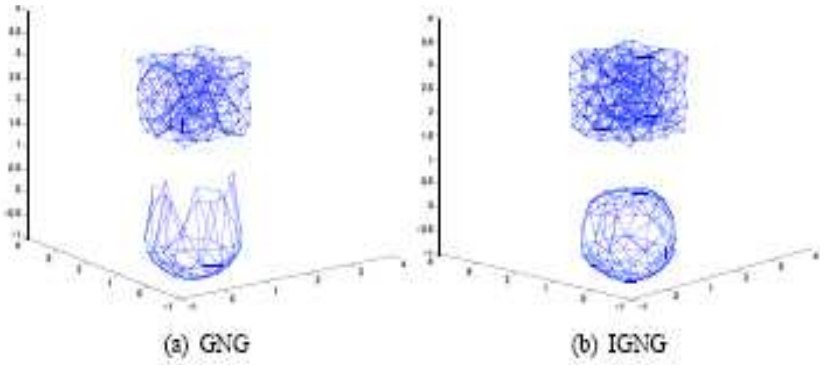



***Fig.2.7. Comportement du GNG (a) et du IGNG (b) dans le cas d'un apprentissage incrémental.***

# IV- Réseau de neurones incrémental pour l'apprentissage non supervisé (IN)

## 4-1- Définition

IN est un modèle des réseaux incrémentaux proposée par les auteurs dans [16], basé sur le regroupement hiérarchique et par partition.

IN est un réseau à deux couches. Chacune d'elle est une carte auto organisatrice. Son algorithme d'apprentissage est inspiré du modèle GNG. La première couche est utilisée pour modéliser et réduire l'espace d'entrées en un nombre plus petit de données en générant la structure topologique de cet espace. La deuxième couche utilise les vecteurs de références de neurones de la première couche comme ensemble d'apprentissage. Cette couche retourne le nombre de classes et donne les prototypes typiques [16].

## 4-2- Principe

Pour les deux couches, à une nouvelle donnée est affecté un nouveau neurone, quand la distance euclidienne entre cette donnée et le premier plus proche neurone (ou le deuxième plus proche neurone) est supérieur à un seuil $T$ adapté d'une manière permanente à la situation présente dans la première couche. Pour la deuxième couche, la distance typique intra-classes et la distance typique entre-classes sont calculées à partir des neurones générés dans la première couche. En suite on donne un seuil $T_c$ constant qui doit être supérieur à la distance d'intra classe et inférieur à la distance entre classes.

Pour l'insertion des neurones dans les classes, on adopte la méthode utilisée par le GNG, d'insérer un neurone entre le neurone $q$ qui a le plus grand signal d'erreur accumulée et son voisin $f$ qui a le plus grand signal d'erreur accumulée. GNG insère le neurone sans aucun contrôle si l'insertion est utile ou pas, mais le IN contrôle l'utilité d'insertion de nouveau neurone par un paramètre d'utilisation $M_i$ et le rayon d'erreur $R_i$. Cette évaluation assure que l'insertion d'un nouveau neurone décrémente l'erreur et contrôle la stabilité du nombre de neurones.

L'adaptation des vecteurs de référence et la construction des connexions se fait comme pour le GNG.

Comme dans le GNGU, on supprime les neurones dans les régions de petite densité, mais dans le IN on utilise une nouvelle stratégie : si le nombre d'utilisation $M_i$ d'un neurone est inférieur à la moyenne du nombre d'utilisation de tous les neurones multiplié par un

paramètre *c*, et que ce neurone a un seul voisin ou c'est un neurone isolé, alors il faut supprimer ce neurone.

## 4-3- Algorithme

Le même algorithme est utilisé pour les deux couches. La différence entre les deux couches est l'ensemble d'apprentissage de la deuxième couche, constitue par les vecteurs de référence de neurones générés par la première couche. Un seuil constant est utilisé dans la deuxième couche à la place d'un seuil adaptatif utilisé dans la première couche.

Initialisation : $N(0)=\{x,y\}$ et $C(0)=\{\ \}$ avec $w_x(0)$ et $w_y(0)$ initialisés aléatoirement dans l'espace des entrées ; $e_x(0)=e_y(0)=0$ et $M_x=M_y=0$

Fixer $t_{\max}, \tau, a_{\max}$, $\alpha_1$, $\alpha_2, \alpha_3$, $\beta$ et $\gamma$.

*t=1.*

Tant que $t \le t_{\max}$ :

(1) Choisir aléatoirement une entrée $p(t)$ parmi l'ensemble d'apprentissage.

(2) Déterminer les deux neurones gagnants $g_1$ et $g_2$ les plus proche de $p(t)$ :

$$g_1 = \arg\min_{i \in N(t)} \left\| p(t) - w_i(t) \right\|$$

$$g_2 = \arg\min_{i \in N(t)-\{g_1\}} \left\| p(t) - w_i(t) \right\|$$

(3) Si $\left\| p(t) - w_{g_1} \right\| > T_{g_1}$ ou $\left\| p(t) - w_{g_2} \right\| > T_{g_2}$ alors :

a. Insérer un nouveau neurone $n_{new}$ avec $w_{new} = p(t)$.

b. Retourner à l'étape (1).

(4) Si $\{g_1, g_2\} \in C(t)$ alors $a_{g_1,g_2} = 0$ sinon

$$C(t) = C(t) \cup \{\{g_1, g_2\}\}, \quad a_{g_1,g_2} = 0.$$

(5) Incrémenter les âges de toutes les connexions adjacentes à $g_1$ :

$$\forall \{i, g_1\} \in C(t) : a_{i,g_1} = a_{i,g_1} + 1$$

(6) Incrémenter l'erreur associée au premier gagnant :

$$e_{g_1} = e_{g_1} + \left\| p(t) - w_{g_1}(t) \right\|$$

(7) Incrémenter le nombre d'utilisation associée au premier gagnant :

$$M_i = M_i + 1$$

(8) Déplacer les vecteurs de référence de $g_1$ et de ses voisins directs dans la direction de $p(t)$ :

$$\Delta w_{g_1}(t) = \eta_{g_1}\left[p(t) - w_{g_1}\right]$$

$$\Delta w_v(t) = \eta_v\left[p(t) - w_v\right],\ v \in \Lambda_{g_1}$$

Où $\Lambda_{g_1} = \left\{i \in N(t) \middle| i \neq g_1 et \{i, g_1\} \in C(t)\right\}$

(9) Retirer de $C(t)$ toutes les connexions pour les quelles $a_{i,j} > a_{\max}$ ; retirer aussi tous les neurones qui se retrouveraient isolés suite à la mort de connexions.

(10) Si $t \bmod \tau = 0$ alors :

a. Déterminer le neurone $q$ possédant l'erreur maximum :

$$q = \arg\max_{i \in N(t)} e_i$$

b. Déterminer le neurone $r$, voisin de $q$, possédant aussi l'erreur maximum :

$$r = \arg\max_{i \in \Lambda_q} e_i$$

c. Insérer un nouveau neurone $x$ à mi-chemin entre $q$ et $r$ :

$$w_x(t) = \frac{w_q(t) + w_r(t)}{2},$$

$$e_x = \alpha_1(e_q + e_r)$$

$$M_x = \alpha_2(M_q + M_r)$$

$$R_x = \alpha_3(R_q + R_r)$$

$$e_q = \beta e_q,\quad e_r = \beta e_r$$

$$M_q = \gamma M_q,\ M_r = \gamma M_r$$

d. Si $e_i / M_i > R_i\ (\forall i \in \{q, r, x\})$ alors :

i. L'insertion est non réussie et le nouveau neurone est supprimé.

ii. Tous les paramètres sont restaurés.

Sinon

iii. $e_i / M_i = R_i\ (\forall i \in \{q, r, x\})$

iv. Remplacer la connexion $\{q, r\}$ par les deux connexions $\{q, x\}$ et $\{x, r\}$ avec $a_{q,x} = a_{x,r} = 0$

e. Pour tous les neurones, si le neurone $i$ a un seul voisin et $M_i < c\sum_{j=1}^{N_a} M_j / N_a$ alors supprimer le neurone $i$. où $N_a$ est le nombre de neurones et $c$ est le taux d'utilité dont sa valeur comprise entre $1 \geq c > 0$.

(11) $t = t + 1$

(12) Fin tant que

***Les paramètres***

***a- Le seuil***

La première couche

- ✓ A l'insertion d'un nouveau neurone $i$ son seuil $T_i$ est égal à $+\infty$.
- ✓ Quand le neurone $i$ est le premier ou le deuxième neurone gagnant, modifier le seuil $T_i$ par :
  - o Si le neurone a des voisins : $T_i = \max_{c \in \Lambda_i} \|w_c - w_i\|$.
  - o Si le neurone est isolé : $T_i = \min_{c \in N - \{i\}} \|w_c - w_i\|$

La deuxième couche :

On calcule la distance intra-classes et la distance inter-classes :

- ✓ La distance intra-classe est la distance moyenne de toutes les connexions : $d_w = \frac{1}{N_C} \sum_{\{i,j\} \in C} \|w_i - w_j\|$.
- ✓ La distance inter-classes est la distance minimale entre deux classes : $d_b(C_i, C_j) = \min_{i \in C_i, j \in C_j} \|w_i - w_j\|$.
- ✓ Le seuil doit être supérieur à la distance intra-classes et inférieur à la distance inter-classes.

***b- Les paramètres d'adaptation des vecteurs de références***

- ✓ Du premier neurone gagnant $\eta_g = \frac{1}{t}$.
- ✓ Des neurones voisins du neurone gagnant $\eta_v = \frac{1}{100t}$,

Où $t$ est représente le nombre d'itérations.

## 4-4- Exemples

Pour montrer l'avantage du IN, dans [16] les auteurs l'appliquent sur une base de données artificielles de 2-dimentions. Cette base contient cinq classes A, B, C, D et E, la classe E est divisée en 3 parties pour montrer l'efficacité du réseau dans le cas non stationnaire. De plus elle est bruitée par 10% de données (voir la figure Fig.2.8). Les paramètres utilisés sont $\tau = 100, a_{\max} = 100, c = 1$. Pour le GNG et le GNGU, le nombre limite de neurones est 300 neurones.

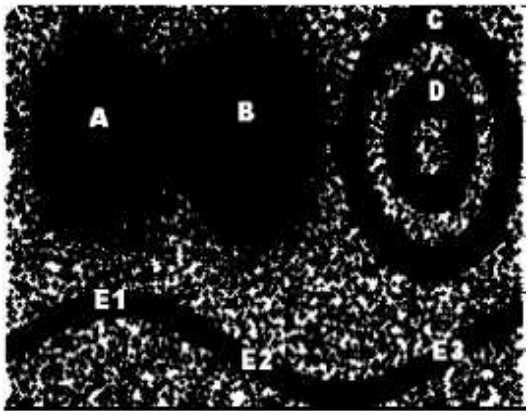



***Fig.2.8. La base de données artificielles utilisées dans l'exemple.***

Pour le cas stationnaire, dans [16] les auteurs choisissent aléatoirement 100 000 exemples. Le GNG détecte la structure topologique mais il rassemble les cinq classes dans une seul classe à cause du bruit (fig2.9). La première couche de IN détecte trois classes (Fig.2.10) et deuxième détecte le cinq classes(Fig.2.11).

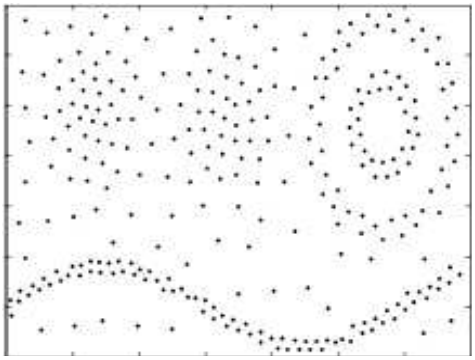


***Fig.2.9. Le résultat du GNG dans le cas stationnaire : une classe.***

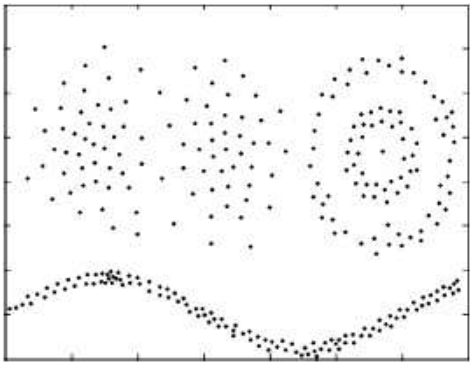


*Fig.2.10. Le résultat du IN de la première couche dans le cas stationnaire : trois classes*

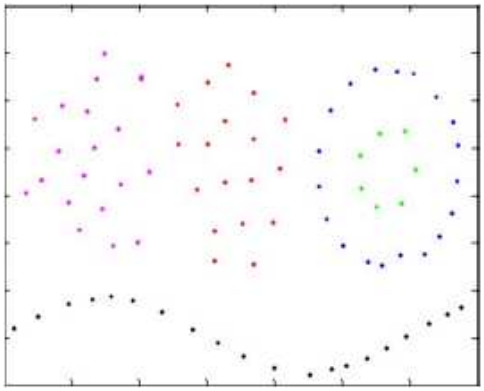


*Fig.2.11. Le résultat du IN de la deuxième couche dans le cas stationnaire : cinq classes*

Pour le cas non stationnaire, dans [16] les auteurs divisent l'ensemble d'apprentissage en parties A, B, C, E1, E2, E3, et choisissent aléatoirement 20 000 exemples dans chaque partie. Ensuite l'apprentissage des différents parties est effectue successivement. Le GNG n'a pas des résultats intermédiaires (voir Fig.2.12) et il ne peut pas d'éliminer le bruit. Le GNGU supprime les parties déjà détectées (voir Fig.2.13). Il ne garde que la dernière partie. La figure Fig.2.14 représente le résultat de la première couche de IN. A prés l'apprentissage de chaque partie la première couche de IN reporte sa structure topologique. La figure Fig.2.15 représente le résultat de la deuxième couche de IN.

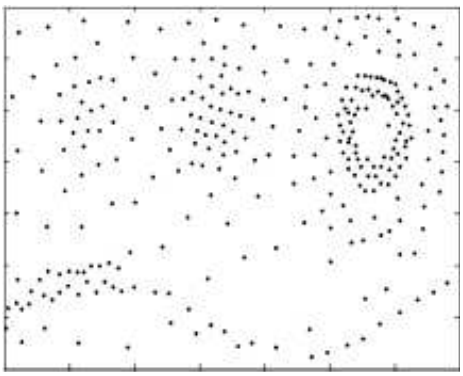


*Fig.2.12. Le résultat du GNG dans le cas non stationnaire.*

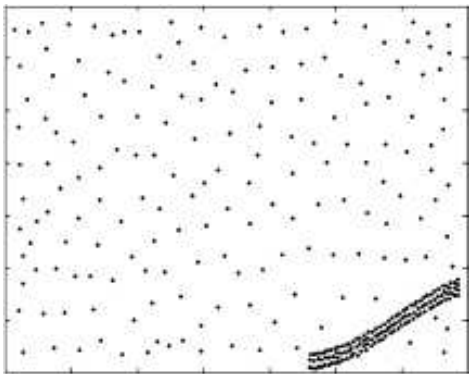


*Fig.2.13. Le résultat du GNGU dans le cas non stationnaire.*

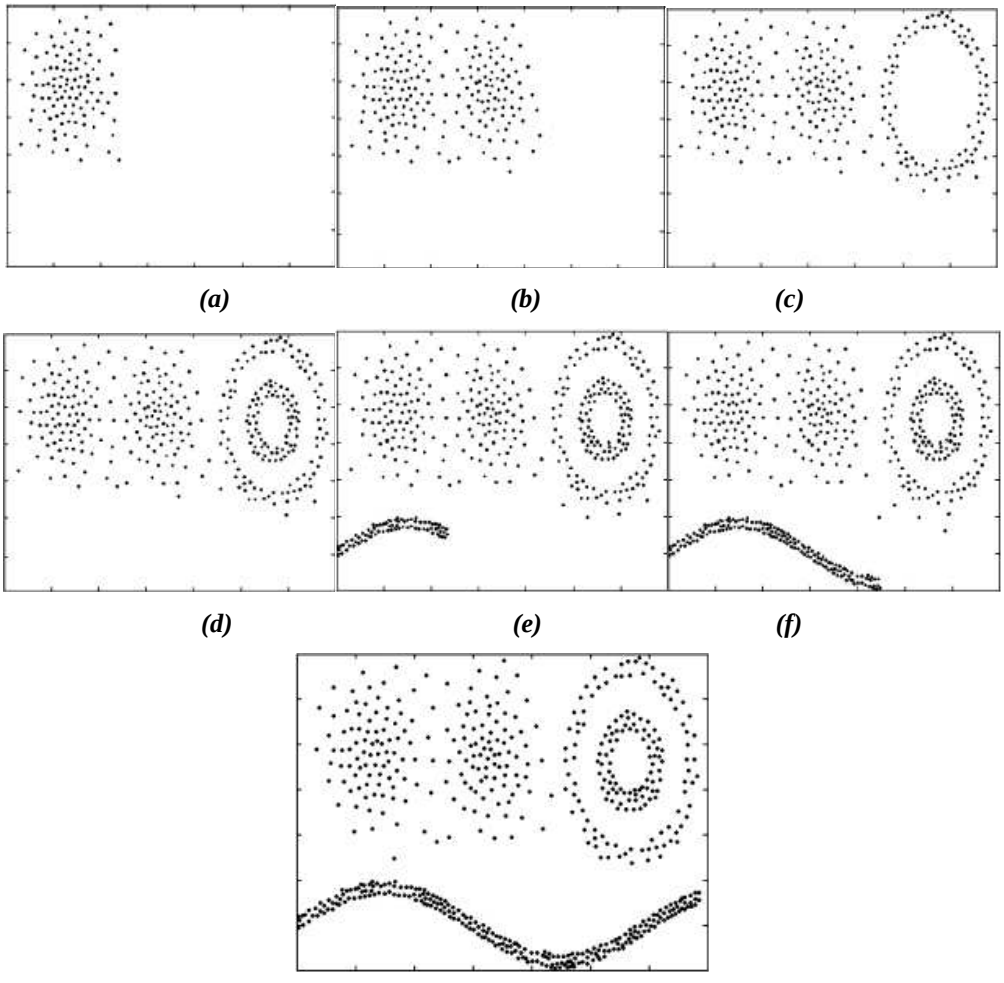



*Fig.2.14. Le résultat du IN de la première couche dans le cas non stationnaire.*

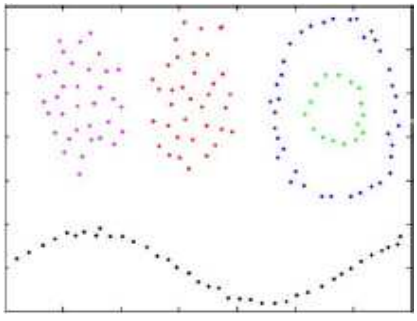


*Fig.2.15. Le résultat du IN de la deuxième couche dans le cas non stationnaire : cinq classes sont détectées.*

# V- Réseau de neurones incrémental (Incremental Neural Network (INeN))

## 5-1- Définition

C'est un réseau de neurones à apprentissage non supervisé [17]. Il possède deux couches, comme le montre la figure Fig.4.16, les neurones dans la première couche ont un vecteur de références et un compteur d'utilisation, où chaque neurone représente une classe. Ce nombre est déterminé automatiquement. Le neurone gagnant signifie qu'un seul neurone sera activé à la fois. Chaque neurone de sortie correspondra à une classe. Les étiquettes affectées aux neurones de sortie sont conservées dans la seconde couche appelée couche index.

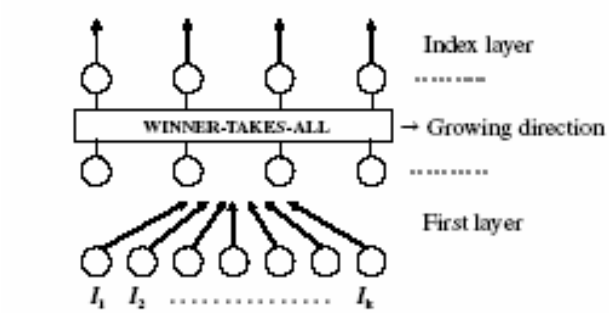



***Fig.2.16. Structure d'un réseau de neurones incrémental.***

## 5-2- Principe

INeN calcule la distance Euclidienne entre les neurones du réseau et le vecteur de référence en entrée. La distance minimale est comparé à un seuil bas $T$ pour décider si le vecteur d'entrée est représenté par un neurone du réseau ou non. Si la distance minimale est inférieure au seuil $T$ alors le compteur du neurone gagnant $g_1$ le plus proche du vecteur d'entrée est incrémenté et son vecteur de référence est modifié par la relation :

$$\Delta w_{g_1}(t) = \eta_{g_1} \lfloor p(t) - w_{g_1} \rfloor \qquad (2.2)$$

*Sinon* un nouveau neurone est inclus dans le réseau, son vecteur de référence est égal au vecteur de référence en entrée et son compteur est nul. Une nouvelle étiquette est affectée à ce neurone.

Un histogramme est utilisé pour contrôler le nombre de neurones créé. Il informe sur le nombre de classes présentes dans l'espace d'entrée. Les valeurs stockées dans les compteurs déterminent les amplitudes de l'histogramme. On utilise un seuil Th pour supprimer les neurones inutiles, sa valeur est déterminée après l'analyse de l'histogramme du réseau.

## 5-3- Algorithme

Initialisation : $N(0) = \{\ \}$ et *indice=0.*

Fixer $\eta_g$ , $T$ et $itera_{max}$.

(1) Tant que $t \le itera_{\max}$ faire

(2) Choisir aléatoirement une entrée $p(t)$ parmi l'ensemble d'apprentissage.

(3) Déterminer le neurone gagnant $g_1$ le plus proche de $p(t)$ :

$$g_1 = \arg\min_{i \in N(t)} \left\| p(t) - w_i(t) \right\|$$

(4) Si $\left\| p(t) - w_{g_1} \right\| > T$ alors :

a. *Indice=indice+1*

b. Insérer un nouveau neurone $n_{new}$ avec $w_{new} = p(t)$ , $M_{new}=0$ *et* $C_{new}=indice$.

c. Retourner à l'étape (1).

(5) Déplacer les vecteurs de références de $g_1$ dans la direction de $p(t)$ :

$$\Delta w_{g_1}(t) = \eta_{g_1} \left\lfloor p(t) - w_{g_1} \right\rfloor$$

$$M_i=M_i+1$$

(6) *t=t+1*

(7) Fin tant que.

(8) Calculer le seuil *Th* à partir de l'histogramme du réseau.

(9) Supprimer tous les neurones ayant une valeur de compteur inférieure à *Th.*

## 5-4- Seuillage automatique

Dans [18] un seuil T est calculé automatiquement avant l'apprentissage par la formule :

$$AT(X) = \sqrt{\frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{N} \left( x_{ij} - m_j \right)^2} \qquad (2.\ 3)$$

Où $X$ est la matrice de l'ensemble d'apprentissage de dimension *M*N*. Chaque ligne représente un vecteur de caractéristiques. Donc $X$ possède $M$ vecteurs de caractéristiques de dimension $N$, et $m_j$ est la moyenne de caractéristiques dans la colonne $j$. Le seuil automatique représente la distribution des vecteurs de caractéristiques dans l'espace de données.

## 5-5- Discussion

Les neurones de INeN sont les centres des classes des données, donc, INeN ne peut détecter que les classes de forme sphérique.

# VI- Conclusion

Tous les réseaux cités précédemment sont des réseaux constructifs et dynamiques où il n'existe aucune contrainte sur ces structures. Dans le tableau suivant, nous allons citer les avantages et les inconvénients pour chaque réseau.

| ***Les réseaux*** | ***Avantages*** | ***Inconvénients*** |
|---|---|---|
| ***GNG*** | - Extrait la topologie de données. | - Il faut limiter le nombre de neurones.<br>- N'élimine pas le bruit.<br>- Non adapté pour l'apprentissage incrémental. |
| ***IGNG*** | - Extrait la topologie de données.<br>- Si le seuil est bien calculé, IGNG détecte les classes.<br>- Elimine le bruit. | - Le seuil est fixé avant l'apprentissage du réseau. |
| ***IN*** | - Extraction de la topologie de données.<br>- Le seuil est calculé automatiquement durant l'apprentissage.<br>- Eliminer le bruit.<br>- Détecter les classes dans la deuxième couche. | - Le réseau est non incrémental en entièr la deuxième couche attend jusqu' à que la première couche termine l'apprentissage de toutes les données. |
| ***INeN*** | - Calcule automatique du seuil avant l'apprentissage.<br>- L'algorithme d'apprentissage simple. | - Ne détecte que les classes des données de forme sphérique. |

Dans le chapitre suivant, nous allons exposer une approche constructive et incrémentale pour l'apprentissage non supervisée qui a les avantages de toutes les approches citées précédemment ;

# chapitre 3

# Les approches proposées

# I- Contribution

Le GNG est un réseau constructif mais non incrémental. IGNG est un réseau incrémental mais il faut fixer le seuil de classification avant l'apprentissage. La première couche de IN est incrémental mais le réseau en entier est non incrémental. INeN ne détecte que les classes sphériques. Dans ce chapitre nous proposerons deux réseaux incrémentaux à seuillage adaptatif pour l'apprentissage non supervisé. Le premier est un réseau incrémental à seuillage adaptatif en calculant le seuil par une formule à partir de la base d'apprentissage. Le deuxième réseau incrémental a deux couches. Le seuil dans la première couche est calculé pendant son apprentissage. Et à partir des neurones de la première couche nous calculons un seuil fixe pour l'apprentissage de la deuxième couche. Un paramètre d'utilité est utilisé dans les deux couches pour éliminer le bruit et éviter le dépassement de nombre de neurones pendant l'apprentissage. Ces deux couches opèrent en parallèle dans le cas incrémental.

# II- Un réseau incrémental à seuillage adaptatif

## 2-1- Principe général

Le réseau IGNGAT (Incremental Growing Neural Gas Automatique Threshold) est constitué de deux couches. La première couche contient les vecteurs de caractéristiques représentant les classes et la deuxième couche contient les étiquettes de ces classes.

Dans la première couche, chaque ensemble des neurones connectés ont la même étiquette dans la deuxième couche.

La première couche est construite selon le même principe que IGNG sauf que nous éliminons la notion d'âge des neurones. En plus dans le IGNG, le seuil $T$ est fixé *a priori*. $T$ est le paramètre qui demande le plus de connaissances sur le problème à traiter. Il correspond au rayon de l'hyper sphère représentant le volume maximal dans l'espace de description que doit considérer un neurone. La question que nous nous posons est quelle valeur affecter à ce seuil ? Dokur dans [16] propose une formule simple pour calculer le seuil de segmentation des images ultrason par le réseau INeN :

$$T = AT(X) = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(x_{ij} - m_j\right)^2} \qquad (3.1)$$

Où $X$ est la matrice de l'ensemble d'apprentissage de dimension $M*N$. Chaque ligne représente un vecteur de caractéristiques. Donc $X$ possède $M$ vecteurs de caractéristiques de

dimension $N$, et $m_j$ est la moyenne de caractéristiques dans la colonne $j$. Le seuil adaptatif représente la distribution des vecteurs de caractéristiques dans l'espace de données.

Le réseau IGNG est constitué par des sous réseaux qui ont un ou plusieurs neurones connectés. Chaque sous réseau représente une classe. Donc l'espace de l'influence d'un sous réseau de IGNG qui représente une classe, est plus grande. Pour résoudre ce problème, il faut diminuer la valeur du seuil. Dans le IGNGAT, nous diminuons la valeur du seuil par la division de la formule précédente par un nombre entier positif qui s'appelle la précision $P_r$ comme suit :

$$T = AT(X, P_r) = \frac{\sqrt{\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(x_{ij} - m_j\right)^2}}{P_r} \tag{3.2}$$

## 2-2- Algorithme

Initialisation : $N(0) = \{\ \}$ et $C(0) = \{\ \}$.

Fixer $\eta_g, \eta_v, a_{\max}$ *et* $P_r$.

Calculer $T = AT(X, P_r) = \dfrac{\sqrt{\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(x_{ij} - m_j\right)^2}}{P_r}$

(14) Tant que *le critère d'arrêt n'est pas satisfait* faire

(15) Choisir aléatoirement une entrée $p(t)$ parmi l'ensemble d'apprentissage.

(16) Déterminer le premier neurone gagnant $g_1$ le plus proche de $p(t)$ :

$$g_1 = \arg\min_{i \in N(t)} \left\| p(t) - w_i(t) \right\|$$

(17) Si $\left\| p(t) - w_{g_1} \right\| > T$ alors :

d. Insérer un nouveau neurone $n_{new}$ avec $w_{new} = p(t)$.

e. Retourner à l'étape (1).

(18) Déterminer le deuxième neurone gagnant $g_2$ le plus proche de $p(t)$ :

$$g_2 = \arg\min_{i \in N(t) - \{g_1\}} \left\| p(t) - w_i(t) \right\|$$

(19) Si $\left\| p(t) - w_{g_2} \right\| > T$ alors :

a. Insérer un nouveau neurone $n_{new}$ avec $w_{new} = p(t)$.

b. Créer une connexion entre $g_1$ et le nouveau neurone :

$$C(t) = C(t) \cup \{\{g_1, n_{new}\}\}, \qquad a_{g_1, n_{new}} = 0.$$

f. Retourner à l'étape (1).

(20) Incrémenter les âges de toutes les connexions adjacentes à $g_1$ :

$$\forall\{i, g_1\} \in C(t) : a_{i,g_1} = a_{i,g_1} + 1$$

(21) Déplacer les vecteurs de références de $g_1$ et de ses voisins directs dans la direction de $p(t)$ :

$$\Delta w_{g_1}(t) = \eta_{g_1}\lfloor p(t) - w_{g_1}\rfloor$$

$$\Delta w_v(t) = \eta_v\lfloor p(t) - w_v\rfloor,\ v \in \Lambda_{g_1}$$

Où $\Lambda_{g_1} = \{i \in N(t) | i \neq g_1 et \{i, g_1\} \in C(t)\}$

(22) Si $\{g_1, g_2\} \in C(t)$ alors $a_{g_1,g_2} = 0$ sinon

$$C(t) = C(t) \cup \{\{g_1, g_2\}\},\quad a_{g_1,g_2} = 0.$$

(23) Retirer de $C(t)$ toutes les connexions pour les quelles $a_{i,j} > a_{\max}$ ; retirer aussi tous les neurones qui se retrouveraient isolés suite à la mort de connexions.

(24) Fin tant que.

# III- Un réseau incrémental à deux couches avec un paramètre d'utilité

## 3-1- Principe général

Le IGNGU est un modèle constructif et incrémental à deux couches de cartes autos organisatrices. Le IGNGU comme tous le réseaux inspirés du GNG, n'impose aucune contrainte sur la structure du réseau.

Le principe général du IGNGU est semblable à celui du IN, la première couche extrait la structure topologique de l'espace d'entrées et élimine partiellement le bruit qui existe dans cet espace. La deuxième couche apprend la structure topologique extraite par la première couche et donnera la structure topologique finale de l'espace d'entrées avec les classes typiques des données en éliminant le bruit restant (voir Fig.3.1).

Dans IN, la deuxième couche attend que la première terminera son apprentissage pour démarrera le sien. Et si des nouvelles données arrivent après que la deuxième couche termine son apprentissage, celle-ci réapprend les données déjà apprises avec les nouvelles données. Pour éviter ces cas de figure nous désactiverons les neurones de la première couche après que la deuxième les apprend. Donc dans notre méthode les neurones des deux couches ont un type actif ou désactif. Les deux couches peuvent travailler en parallèle, c'est-à-dire après que la

première couche apprend une partie de l'espace d'entrées, la deuxième couche démarrera l'apprentissage de cette partie. En plus de cet avantage pendant l'apprentissage des deux couches, seuls les neurones actifs qui sont supprimés. Donc on ne supprime pas les neurones désactifs existants dans l'espace de donnés de petite densité dans les données déjà apprises (dans le cas des classes linéaires et dans les frontières des classes).

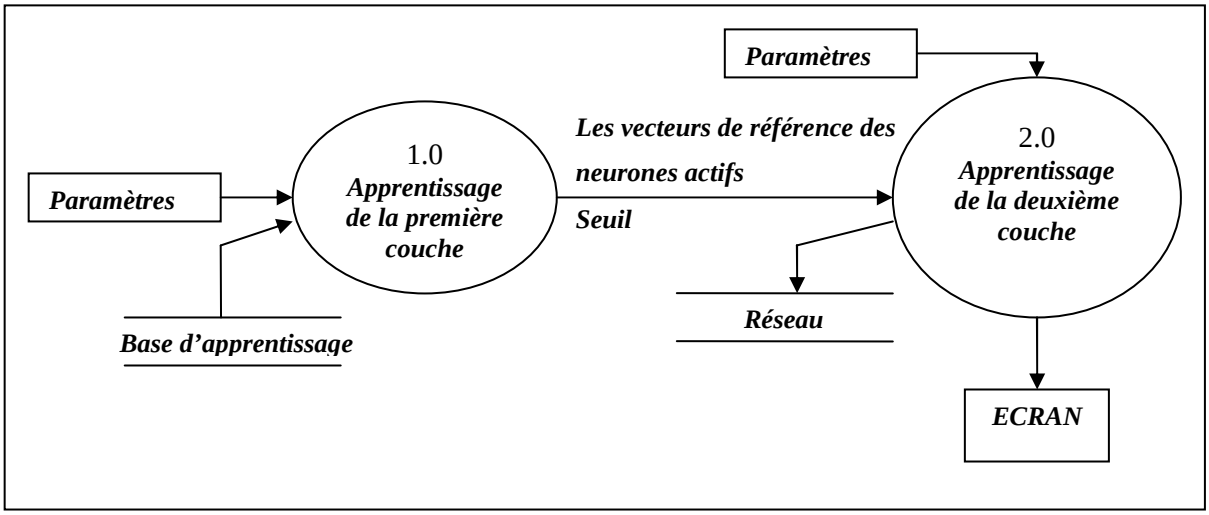


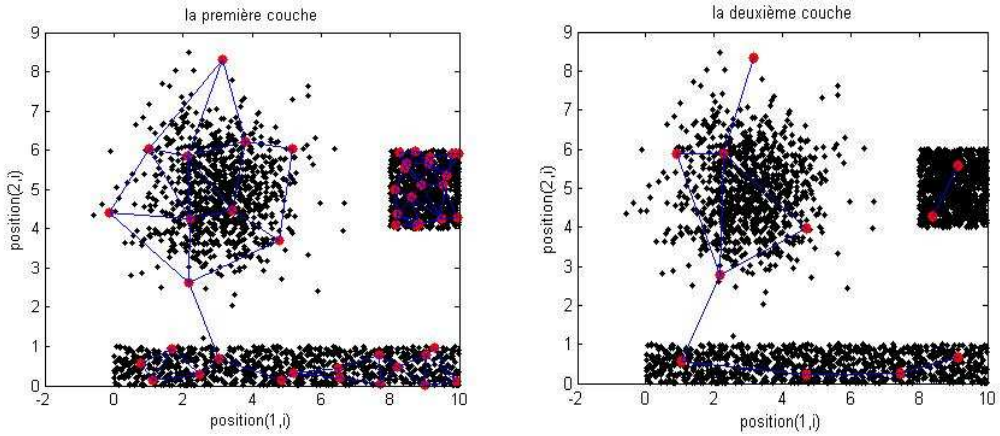

***Fig.3.1. Schéma général d'apprentissage des deux couches du IGNGU.***

Nous avons utilisé un taux d'utilité pour supprimer les neurones inutiles ou les neurones qui représentent le bruit, comme dans le IN. Et pour éviter de supprimer les nouveaux neurones, nous associons à chaque neurones un âge. La suppression est faite chaque $\tau$ itérations où, on supprime les neurones actifs ayant un seul voisin, un nombre d'activation inférieur à la moyenne des nombres d'activations des neurones actifs multipliée par un taux d'utilité c et un âge supérieur à $\tau$.

Nous associons à chaque connexion de deux neurones dans la première couche, deux paramètres : l'âge et un compteur d'utilisation qui s'incrémente si l'âge de cette connexion devient 0. Pendant l'apprentissage nous supprimons les connexions ayant un âge supérieur à l'âge maximum des connexions au bien les connexions ayant une valeur de nombre d'utilisation inférieur à une valeur prédéfinie et un âge inférieur à l'âge moyen des connexions.

Nous avons testé le IN sur une base de données artificielle contenant 3 classes. La première classe à une forme sphérique où les données sont distribuées par la loi normale. La deuxième classe à une forme rectangulaire et la troisième classe a une forme linéaire où les données sont distribuées uniformément. Dans chaque classe il y a 10 000 exemples. Donc la base contient 30 000 exemples. La première couche détecte 2 classes (voir Fig.3.2 (a)) et la deuxième couche détecte aussi 2 classes ( voir Fig.3.2(b)).

Dans le IN le seuil d'un nouveau neurone est infini. Donc celui-ci a un grand espace d'influence, il est possible que le IN rassemble les classes proches dans l'espace de données. Nous avons affecté au seuil du nouveau neurone la distance entre celui-ci et le plus proche neurone de lui.




*(a)* *(b)*

***Fig.3.2. Ensemble de données représenté dans un espace à deux dimensions. (a) la première couche du réseau IN détecte deux classes, (b) la deuxième couche du IN détecte deux classes.***

Chaque neurone *n* des deux couches possède:

- ✓ Un vecteur de référence,"$w_n$ ".
- ✓ Un type, "$type_n$" (actif ou désactif).
- ✓ Un nombre d'activation "$M_n$ ".
- ✓ Une étiquette "$C_n$ " indique la classe du neurone.
- ✓ Un seuil " $T_n$".
- ✓ Un age "$age_n$ ".

Les neurones de la première couche possèdent en plus:

- ✓ un signal d'erreur $e_n$
- ✓ un rayon d'erreur $R_n$.

Chaque connexion est associée à un âge. Une connexion jeune implique une vraisemblance élevée de la relation topologique, alors qu'au contraire une connexion âgée signifie une vraisemblance faible de cette relation. Lorsque l'âge d'une connexion dépassera un certain seuil, celle-ci pourra mourir et disparaître. Afin de construire le réseau, on effectue avant l'apprentissage de chaque donnée $p(t)$ un test défini par l'équation (3.3). Ce test sert à vérifier si l'insertion d'un neurone est nécessaire. S'il n'existe pas au moins deux neurones qui satisfont cette équation, alors l'ajout d'un nouveau neurone est effectué.

$$\| p(t) - w_n \| \leq T_n \tag{3.3}$$

$w_n$ : Vecteur de référence d'un neurone *n* et $T_n$ son seuil.

Quand un neurone est inséré, c'est un neurone de type ***actif*** et les autres paramètres sont nuls. Initialement la couche est vide. A chaque itération, nous cherchons le neurone gagnant $g_1$ . Si celui-ci n'existe pas (réseau vide) ou s'il ne satisfait pas la condition de l'équation (3.3), un nouveau neurone est ajouté avec $w_{new} = p(t)$ . Dans la première couche, si le réseau est non vide, le seuil est adapté par la formule (3.4), dans le cas contraire, nous affectons au seuil l'infini:

$$T_{new} = \| p(t) - w_{g_1} \| \tag{3.4}$$

Un deuxième cas peut se produire quand un neurone gagnant $g_1$ satisfaisant l'équation (3.3) existe. Nous recherchons alors le deuxième neurone $g_2$ le plus proche de la donnée. Si $g_2$ n'existe pas ou s'il ne satisfait pas le test, un nouveau neurone est ajouté avec $w_{new} = p(t)$ et une connexion est créée entre ce neurone et $g_1$. Dans la première couche le seuil est adapté par la formule (3.4).

Enfin, le dernier cas de figure correspond à l'existence d'au moins deux neurones qui satisfont le test de l'équation (3.3). Aucun neurone n'est ajouté dans cette situation et l'adaptation des vecteurs de référence se fait comme pour le GNG. L'âge des connexions émanant de $g_1$ est incrémenté. Comme dans l'apprentissage Hebbien compétitif, nous ajoutons une connexion entre les deux plus proches neurones de la donnée. Si celle-ci existe déjà, son âge est réinitialisé à zéro. Toutes les connexions ayant un âge supérieur à $a_{\max}$ sont supprimées, et si ceci a pour conséquence l'isolement d'un neurone, celui-ci est supprimé. Il faut activer le premier neurone gagnant et ses voisins directs, c'est-à-dire le type de neurones devient ***actif***. Le nombre d'activations du premier neurone gagnant est incrémenté et l'âge de tous les neurones est incrémenté. Dans la première couche les seuils des deux neurones gagnants sont adaptés par la formule (3.5):

$$T_i = \max_{c \in \Lambda_i} \| w_c - w_i \| \tag{3.5}$$

Où $\Lambda_i$ est l'ensemble des voisins directs du neurone *i*.

Chaque $\tau$ itérations on supprime:

Les connexions entre les neurones actifs ayant une valeur de compteur d'utilisation inférieur à une valeur prédéfinie et un âge inférieur à l'âge moyenne des connexions.

- ✓ Les neurones actifs isolés.
- ✓ Les neurones actifs ayant un seul voisin, un nombre d'activation inférieur à la moyenne des nombres d'activations des neurones actifs multipliée par un taux d'utilité *c* et un âge supérieur à $\tau$, où $0 < c \leq 1$, pour supprimer le bruit et éviter le dépassement du nombre des neurones.

Si la base d'apprentissage est divisée en plusieurs parties. A chaque fois une couche terminera d'apprendre une partie, elle désactivera tous ses neurones et les étiquettera (deux neurones connectés appartiennent à la même classe). La première couche avant de désactiver ses neurones, passe à la deuxième couche les vecteurs de référence des neurones actifs, ensuite elle les désactivera.

On désactive les neurones pour éviter deux cas, le premier est de supprimer les données déjà acquises et le deuxième de repasser les données déjà apprises par les deux couches. C'est-à-dire réapprentissage de la deuxième couche des données déjà apprises.

# 3-2- Algorithme

## 3-2-1- Algorithme d'apprentissage de la première couche

Initialisation

Si la couche est vide

Insérer deux nouveaux neurones avec des vecteurs de référence choisis aléatoirement dans l'espace des entrées :

$N(0) = \{x, y\}$ et $C(0) = \{\ \}$ avec $w_x(0)$ et $w_y(0)$ initialisés aléatoirement dans l'espace des entrées ; $e_x(0) = e_y(0) = 0$, $M_x = M_y = 0$, $R_x = R_y = 0$, $T_x = T_y = +\infty$, $age_x = age_y = 0$.

Sinon

Désactiver tous les neurones ;

Fixer $t_{\max}, \tau, a_{\max}$, $\alpha_1$, $\alpha_2, \alpha_3$, $\beta$ et $\gamma$.

t=1.

Tant que $t \leq t_{\max}$ :

(1) Choisir aléatoirement une entrée $p(t)$ parmi l'ensemble d'apprentissage.

(2) Déterminer le première neurone gagnant $g_1$ le plus proche de $p(t)$ :

$$g_1 = \arg\min_{i \in N(t)} \| p(t) - w_i(t) \|$$

(3) Si $\left\|p(t)-w_{g_1}\right\| > T_{g_1}$ alors :

e. Insérer un nouveau neurone actif $n_{new}$ avec $w_{new} = p(t)$ et $T_{new} = \left\|p(t)-w_{g_1}\right\|$.

f. Retourner à l'étape (1).

(4) Déterminer le deuxième neurone gagnant $g_2$ le plus proche de $p(t)$

$$g_2 = \arg \min_{i \in N(t)-\{g_1\}} \left\|p(t)-w_i(t)\right\|$$

(5) Si $\left\|p(t)-w_{g_2}\right\| > T_{g_2}$ alors :

i. Insérer un nouveau neurone actif $n_{new}$ avec $w_{new} = p(t)$ et $T_{new} = \left\|p(t)-w_{g_1}\right\|$.

ii. Créer une connexion entre $n_{new}$ et $g_1$ :

$$C(t) = C(t) \cup \{\{g_1, n_{new}\}\}, \quad a_{g_1,n_{new}} = 0 \text{ et } c_{g_1,g_2} = 0.$$

iii. Retourner à l'étape (1).

(6) Si $\{g_1, g_2\} \in C(t)$ alors $a_{g_1,g_2} = 0$ sinon

$$C(t) = C(t) \cup \{\{g_1, g_2\}\}, \quad a_{g_1,g_2} = 0 \text{ et } c_{g_1,g_2} = c_{g_1,g_2} + 1.$$

(7) Incrémenter les âges de toutes les connexions adjacentes à $g_1$ :

$$\forall \{i, g_1\} \in C(t) : a_{i,g_1} = a_{i,g_1} + 1$$

(8) Incrémenter le signal d'erreur associé au premier gagnant :

$$e_{g_1} = e_{g_1} + \left\|p(t)-w_{g_1}(t)\right\|$$

(9) Incrémenter le nombre d'activation associée au premier gagnant :

$$M_i = M_i + 1$$

(10) Calculer le rayon d'erreur de $g_1$ :

$$R_{g_1} = e_{g_1} / M_{g_2}$$

(11) Déplacer les vecteurs de référence de $g_1$ et de ses voisins directs dans la direction de $p(t)$ et les activer :

$$\Delta w_{g_1}(t) = \eta_{g_1} \left[p(t)-w_{g_1}\right]$$

$$\Delta w_v(t) = \eta_v \left[p(t)-w_v\right], \; v \in \Lambda_{g_1}$$

Où $\Lambda_{g_1} = \left\{i \in N(t) \middle| i \neq g_1 et \{i, g_1\} \in C(t)\right\}$

(12) Retirer de $C(t)$ toutes les connexions pour les quelles $a_{i,j} > a_{\max}$ ; retirer aussi tous les neurones qui se retrouveraient isolés suite à la mort de connexions.

(13) Calculer le seuil des deux gagnants $g_1$ et $g_2$ :

$$T_i = \max_{c \in \Lambda_i} \| w_c - w_i \|$$

(14) Incrémenter l'age de tous les neurones actifs.

(15) Si $t \bmod \tau = 0$ alors :

i. Supprimer les connexions entre les neurones actifs ayant une valeur de compteur d'utilisation inférieur à une valeur prédéfinie et un âge inférieur à l'âge moyen des connexions.

ii. Supprimer les neurones actifs isolés.

iii. Supprimer tous les neurones actifs ayant un seul voisin, un âge inférieur à $\tau$ et un nombre d'activation $M_i < c \sum_{j=1}^{N_a} M_j / N_a$ . où $N_a$ est le nombre de neurones actifs et *c* est le taux d'utilité sa valeur comprise entre 0 et 1

iv. Déterminer le neurone $q$ possédant la valeur d'erreur maximal :

$$q = \arg\max_{i \in N(t)} e_i$$

v. Déterminer le neurone $r$ , voisin de $q$ , possédant aussi la valeur d'erreur maximal :

$$r = \arg\max_{i \in \Lambda_q} e_i$$

vi. Insérer un nouveau neurone $x$ à mi-chemin entre $q$ et $r$ :

$$w_x(t) = \frac{w_q(t) + w_r(t)}{2},$$

$$e_x = \alpha_1 (e_q + e_r)$$

$$M_x = \alpha_2 (M_q + M_r)$$

$$R_x = \alpha_3 (R_q + R_r)$$

$$T_x = \| w_x - w_q \|$$

$$e_q = \beta e_q, \quad e_r = \beta e_r$$

$$M_q = \gamma M_q, \; M_r = \gamma M_r$$

vii. Si $e_x / M_x > R_x$ alors :

i. L'insertion est non réussie et le nouveau neurone est supprimé.

ii. Tous les paramètres sont restores.

Sinon

1. $e_i / M_i = R_i \; (\forall i \in \{q, r, x\})$

2. Remplacer la connexion $\{q,r\}$ par les deux connexions $\{q,x\}$ et $\{x,r\}$ avec $a_{q,x} = a_{x,r} = 0$

(16) t=t+1 retourner à (1).

(17) Fin tant que.

### 3-2-2- Calcul du seuil $T_c$ à partir de la première couche

i. Calculer la distance intra-classes : $d_w = \frac{1}{N_C} \sum_{\{i,j\} \in C} \left\| w_i - w_j \right\|$

ii. Calculer la distance enter-classes : $d_b(C_i, C_j) = \min_{i \in C_i, j \in C_j} \left\| w_i - w_j \right\|$

iii. Si la première couche contient une seule classe, le seuil est égal à la première distance supérieure à la distance intra classe. Sinon, le seuil doit être supérieur à la distance intra-classes et inférieur à la distance enter-classes.

### 3-2-3- Algorithme d'apprentissage de la deuxième couche

Initialisation

Construire l'ensemble d'apprentissage à partir des vecteurs de référence de neurones actifs de la première couche.

Si *la deuxième couche est vide* alors

Insérer deux nouveaux neurones avec des vecteurs de référence choisis aléatoirement à partir de l'ensemble d'apprentissage :

$N(0) = \{x, y\}$ et $C(0) = \{\ \}$ avec $w_x(0)$ et $w_y(0)$ initialisés aléatoirement dans l'espace des entrées ; $M_x = M_y = 0$ , $T_x = T_y = T_c$.

Sinon

Désactiver tous les neurones ;

Fixer $t_{\max}, \tau, \eta_g, \eta_v, a_{\max}$ .

t=1.

Tant que $t \le t_{\max}$ :

(1) Choisir aléatoirement une entrée $p(t)$ parmi l'ensemble d'apprentissage.

(2) Déterminer le première neurone gagnant $g_1$ le plus proche de $p(t)$ :

$$g_1 = \arg\min_{i \in N(t)} \left\| p(t) - w_i(t) \right\|$$

(3) Si $\left\| p(t) - w_{g_1} \right\| > T_{g_1}$ alors :

a. Insérer un nouveau neurone actif $n_{new}$ avec $w_{new} = p(t)$ et $T_{new} = T_c$.

b. Retourner à l'étape (1).

(4) Déterminer le première neurone gagnant $g_1$ le plus proche de $p(t)$ :

$$g_2 = \arg \min_{i \in N(t) - \{g_1\}} \left\| p(t) - w_i(t) \right\|$$

(5) Si $\left\| p(t) - w_{g_2} \right\| > T_{g_2}$ alors :

a. Insérer un nouveau neurone actif $n_{new}$ avec $w_{new} = p(t)$ et $T_{new} = T_c$.

c. Créer une connexion entre $n_{new}$ et $g_1$ :

$$C(t) = C(t) \cup \{\{g_1, n_{new}\}\}, \quad a_{g_1, n_{new}} = 0.$$

d. Retourner à l'étape (1).

(6) Si $\{g_1, g_2\} \in C(t)$ alors $a_{g_1, g_2} = 0$ sinon

$$C(t) = C(t) \cup \{\{g_1, g_2\}\}, \quad a_{g_1, g_2} = 0.$$

(7) Incrémenter les âges de toutes les connexions adjacentes à $g_1$ :

$$\forall \{i, g_1\} \in C(t) : a_{i, g_1} = a_{i, g_1} + 1$$

(8) Incrémenter le nombre d'activation associée au premier neurone gagnant et au deuxième neurone gagnant:

$$M_i = M_i + 1$$

(9) Déplacer les vecteurs de référence de $g_1$ et de ses voisins directs dans la direction de $p(t)$ et les activer :

$$\Delta w_{g_1}(t) = \eta_{g_1} \left[ p(t) - w_{g_1} \right]$$

$$\Delta w_v(t) = \eta_v \left[ p(t) - w_v \right], \; v \in \Lambda_{g_1}$$

Où $\Lambda_{g_1} = \left\{ i \in N(t) \middle| i \neq g_1 et \{i, g_1\} \in C(t) \right\}$

(10) Retirer de $C(t)$ toutes les connexions pour les quelles $a_{i,j} > a_{\max}$ ; retirer aussi tous les neurones qui se retrouveraient isolés suite à la mort de connexions.

(11) Incrémenter l'âge de tous les neurones actifs.

(12) Si $t \bmod \tau = 0$ alors :

a. Supprimer les neurones actifs isolés.

b. Supprimer tous les neurones actifs ayant un seul voisin, un âge inférieur à $\tau$ et un nombre d'activation $M_i < c\sum_{j=1}^{N_a} M_j / N_a$ . où $N_a$ est le nombre de neurones actifs et *c* est le taux d'utilité sa valeur comprise entre 0 et 1.

(13) t=t+1 retourner à (1).

(14) Fin tant que.

### 3-2-4- L'étiquetage des neurones des deux couches

1. Initialiser tous les neurones par non classés.
2. Prendre un neurone non étiqueté et lui affecter une étiquette.
3. affecter la même étiquette à tous les neurones non classés connectés à ce neurone par un chemin.
4. Répéter jusqu'à ce que tous les neurones soient étiquetés.

## IV- Conclusion

IGNGAT est un réseau incrémental à seuillage adaptif, par le calculant du seuil à partir de la base d'apprentissage et une valeur de précision.

IGNGU est un réseau incrémental à deux couches. La première couche réduit l'espace d'entrée en des sous ensembles par l'utilisation d'un seuil de similarité adaptatif, et élimine le bruit existant dans cet espace par l'utilisation d'un paramètre d'utilité. À partir de la première couche nous calculons le seuil utilisé dans la deuxième couche. Cette dernière apprend les vecteurs de référence des neurones actifs de la première couche et donne les classes typiques des données et élimine le bruit restant. Ces deux couches opèrent en parallèle dans le cas incrémental. Les classes des données sont représentées par des sous réseaux.

Nous avons appliqué le IGNGAT dans la segmentation des images de textures, et le IGNGU sur une base de données artificielle, la segmentation des images de textures et sur la classification des logos. Dans le prochain chapitre nous présentons, les résultats en détail.

# chapitre 4

# Les résultats expérimentaux

# I- Les résultats de IGNGAT

Nous avons appliqué l'algorithme proposé IGNGAT pour la segmentation des images de textures de différents types (fine –grossière – périodique et apériodique). Ces images sont constituées d'images de synthèse extraites de l'album de Broadatz [19], d'images aériennes et des images médicales.

Le choix de la base d'apprentissage est faite par la sélection aléatoire sur l'image. La taille de la fenêtre utilisée dans l'apprentissage et le choix des paramètres de texture utilisés dépendent du type d'image à segmenter.

La valeur du paramètre $P_r$ dépend du type d'images et des détails demandés. Nous montrons ci-dessous l'effet de ce paramètre sur le résultat de la segmentation.

## 1-1- Les paramètres de textures

Nous avons utilisé la matrice de co-occurrence qui constitue un outil puissant pour l'analyse des textures. La matrice de co-occurrence *Mc* d'une région d'intérêt *ROI* est définie pour tout couple de niveaux de gris (a, b) et une translation *d* par :

$$Mc\ (a, b)=card\left\{(s,s+d)\in \Re^2 / A[s]=a, A[s+d]=b\right\} \qquad (4.1)$$

*Mc (a, b)* est donc le nombre de couples de sites *(s, s+d)* de la région considérée, séparé par le vecteur de translation *d* ayant pour niveau de gris *a* et *s+d* pour niveau de gris *b*. Pour une image quantifiée sur *L* niveau de gris, la matrice *Mc (a, b)* est une matrice *LxL*.

### 1-1-1- Remplissage d'une matrice de co-occurrence

La figure Fig.4.1 représente un exemple de calcul de la matrice de co-occurrence calculée sur une région de 25 pixels, quantifié sur six niveaux de gris et avec une translation d= (0,1) (un pixel vers la droite).

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 1 | 3 | 4 |
| 2 | 3 | 1 | 2 | 4 |
| 3 | 3 | 2 | 1 | 1 |
| 5 | 3 | 5 | 2 | 4 |
| 1 | 1 | 5 | 2 | 2 |

***Région de 25 sites***

| | | | | | |
|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | 2 | 2 | 1 | 0 | 1 |
| 2 | 2 | 1 | 1 | 2 | 0 |
| 3 | 1 | 1 | 1 | 1 | 1 |
| 4 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 2 | 1 | 0 | 0 |

***matrice de co-occurrence***

***Fig.4.1. Exemple de matrice de co-occurrence.***

Vu que les matrices de co-occurrence contiennent une masse trop importante d'informations, elles sont difficilement manipulables dans leur intégralité (256 valeurs dont beaucoup de zéros dans les exemples calculés au cours de notre étude). Quatorze indices prenant en compte l'ensemble de la matrice Mc(a, b) ont été définis par Haralick [20], Ils correspondent à des caractères descriptifs des textures (contraste, homogénéité, etc.…). Nous proposons ici les indices les plus fréquemment utilisés. Ils sont présentés sous forme normalisée, c'est-à-dire qu'ils prennent des valeurs comprises entre 0 et 1.

## 1-1-2- Les caractéristiques

***a) La normalisation***

*Nc* est le paramètre de normalisation

$$Nc = \sum_{a=1}^{L} \sum_{b=1}^{L} Mc(a,b) \tag{4.2}$$

***b) Homogénéité locale***

Cet indice est d'autant plus élevé que l'on retrouve souvent le même couple de pixels. Ce qui est le cas quand le niveau de gris est uniforme ou quand il y a périodicité dans le sens de la translation. Il est défini par la relation suivante :

$$Hom = \frac{1}{Nc} \sum_{a=1}^{L} \sum_{b=1}^{L} \frac{Mc(a,b)}{\left[1+(b-a)^2\right]} \tag{4.3}$$

***c) Le contraste (Inertie)***

Mesure les variations locales des niveaux de gris. Si elles sont importantes alors le contraste sera élevé, donc il existe peu de régions homogènes.

$$Cont = \frac{1}{Nc} \sum_{a=1}^{L} \sum_{b=1}^{L} (a-b)^2 \, Mc(a,b) \tag{4.4}$$

***d) L'Entropie***

Mesure la complexité de l'image. Elle permet de caractériser le degré de granularité de l'image. Plus l'entropie est élevée et plus la granularité est grossière.

$$Ent = -\frac{1}{Nc} \sum_{a=1}^{L} \sum_{b=1}^{L} Mc(a,b) \log\left(\frac{Mc(a,b)}{Nc}\right) \tag{4.5}$$

***e) La variance***

Caractérise la distribution des niveaux de gris autour de la valeur moyenne µ.

$$\mu_x = \sum_{a=1}^{L}\sum_{b=1}^{L} aMc(a,b) \;;\; \mu_y = \sum_{a=1}^{L}\sum_{b=1}^{L} bMc(a,b)$$

$$\mathrm{Var_x} = \frac{1}{Nc}\sum_{a=1}^{L}\sum_{b=1}^{L}(a-\mu_x)^2 Mc(a,b) \tag{4.6}$$

$$\mathrm{Var_y} = \frac{1}{Nc}\sum_{a=1}^{L}\sum_{b=1}^{L}(b-\mu_y)^2 Mc(a,b)$$

***f) L'énergie***

Cet indice mesure l'homogénéité de l'image. L'énergie a une valeur d'autant plus faible qu'il y'a peu de zones homogènes.

$$E = \frac{1}{Nc^2}\sum_{a=1}^{L}\sum_{b=1}^{L}\left(Mc(a,b)\right)^2 \tag{4.7}$$

***g) La corrélation***

Décrit la corrélation entre les lignes et les colonnes de la matrice de co-occurrence. Plus les valeurs sont uniformément distribuées dans la matrice et plus cet indice est grand.

$$Cor = \frac{1}{Nc}\sum_{a=1}^{L}\sum_{b=1}^{L} abMc(a,b) \tag{4.8}$$

***h) L'uniformité-1 (first_order)***

D'autant plus élevée qu'un même niveau de gris apparaît dans la texture.

$$Dir = \frac{1}{Nc}\sum_{a=1}^{L} Mc(a,a) \tag{4.9}$$

***i) L'uniformité-2 (second_order)***

Cet indice est d'autant plus important que la texture possède une orientation privilégiée dans le sens de la translation.

$$U = \frac{1}{Nc^2}\sum_{a=1}^{L}(Mc(a,a))^2 \tag{4.10}$$

## 1-2- Le processus de segmentation d'une image de textures

Il est défini par l'organigramme de la figure Fig.4.2. Pour segmenter une image de textures nous calculons le vecteur de caractéristiques d'une fenêtre de segmentation (FS). Si la distance entre ce vecteur et le neurone gagnant inférieure au seuil, nous affectons son étiquette au pixel centre de cette fenêtre.

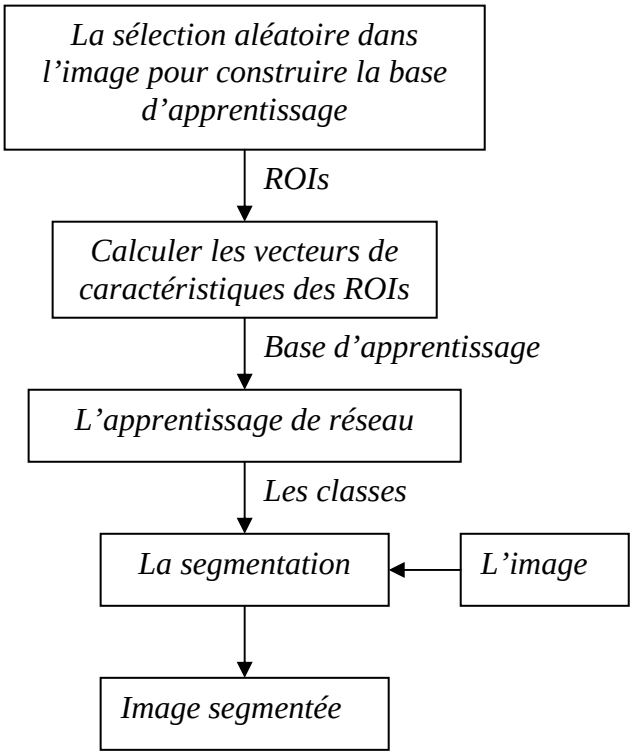



***Fig.4.2. Le processus de segmentation d'une image de textures.***

## 1-3- Les images de synthèse

Les images de synthèse sont utilisées pour montrer l'influence de la taille de la fenêtre d'apprentissage, la taille de la fenêtre de segmentation et la précision $P_r$ sur les résultats de la segmentation. Les paramètres utilisés par l'algorithme ont été fixés à $a_{\max}$ =10 pour l'âge maximal que peut avoir une connexion, $\eta_g$ =0.09 pour la pas d'adaptation du neurone gagnant, $\eta_v$ =0.01 pour le pas d'adaptation des voisins du neurone gagnant.

Nous avons deux types d'images de textures : fine et grossière. Nous avons utilisé 20 images de taille 320x200 pixels pour chacune des deux textures

La figure Fig.4.3 montre l'image originale et deux résultats de segmentation. Pour $P_r$ = 1, IGNGAT détecte une seule classe mais pour $P_r$=2, il détecte 3 classes. Fig4.4 montre le réseau IGNGAT avec l'ensemble d'apprentissage pour les deux résultats de classification. Pour $P_r$=1 il existe une seule classe et le nombre de neurones est égal à 7, et pour $P_r$=2, il

existe trois classes et le nombre de neurones est égal à 11. Donc la valeur de $P_r$ influe sur le nombre de neurones dans le réseau et le nombre de classes.

La figure Fig.4.4 montre aussi, que le IGNGAT ne partitionne pas l'espace d'entrée seulement, mais il extrait aussi sa topologie.

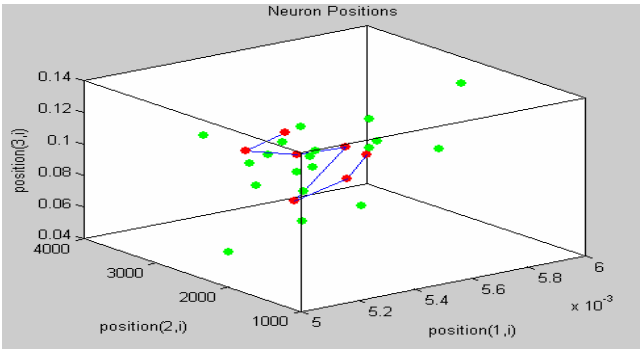
Après la segmentation d'une image de synthèse avec une texture, nous montrons dans les figures Fig.4.5 et Fig.4.6 l'image originale constituée de deux textures fines et l'image segmentée. La fenêtre d'apprentissage (FA) ainsi que la fenêtre de segmentation (FS) sont de taille 15*15 et $P_r$=1. Pour les textures grossières, il faux agrandir la taille de la fenêtre de façon à englober le motif constituant la texture comme le montrent Fig.4.7 pour une image de texture grossière et Fig.4.8 pour une image de texture fine et grossière.

## 1-3-1- Les images de textures fines

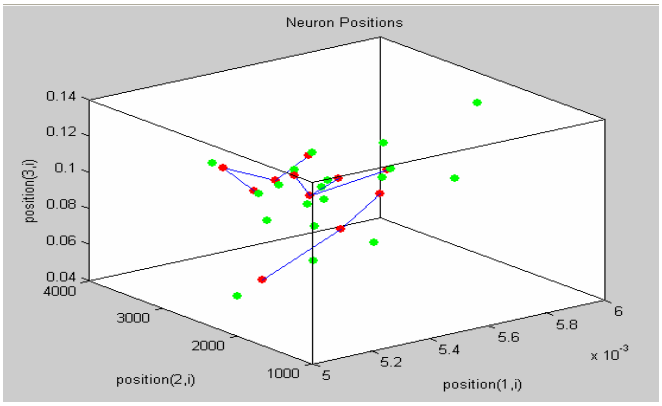
***a) Pour une classe.***

***(a) (b) (c)***

***Fig.4.3. Image à une texture (a) image originale, image segmentée (b) $P_r$=1, (c) $P_r$=2.***




***(a)***




***(b)***

***Fig.4.4. Espace d'entrée partitionné, (a)$P_r$=1, (b)$P_r$=2***

***b) Pour deux classes***

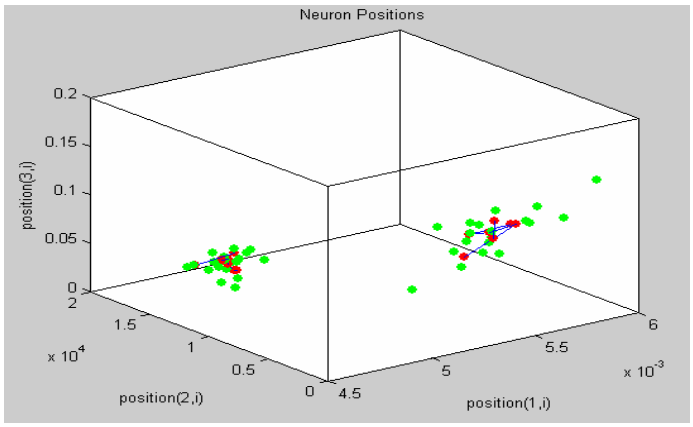
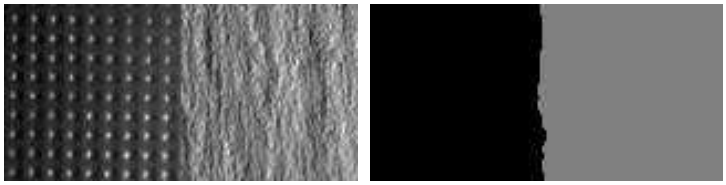


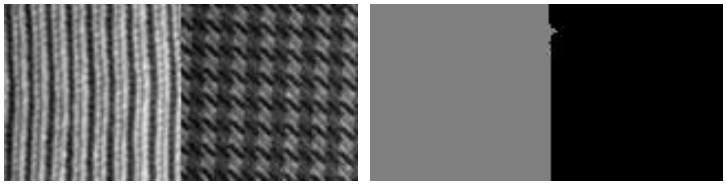

***Fig.4.5. Espace d'entrée partitionné en deux classes***



***(a)*** ***(b)***

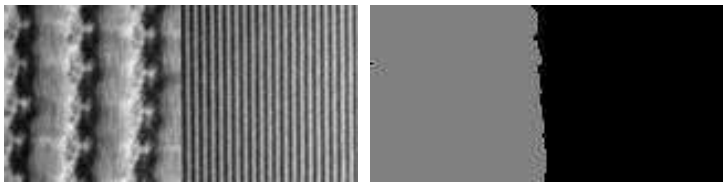
***Fig.4.6. Deux textures fines: (a) image originale, (b) image segmentée. Pr=1, FA=15*15 et FS=15*15.***

## 1-3-2- Les images de textures grossières



***(a)*** ***(b)***

***Fig.4.7. Deux textures grossières: (a) image originale, (b) image segmentée. FA=65*65 et FS=35*35.***



***(a)*** ***(b)***

***Fig.4.8. Une texture grossière et une texture fine : (a) image originale, (b) image segmentée. FS=65*65 et FS=55*55.***

## 1-4- Les images aériennes

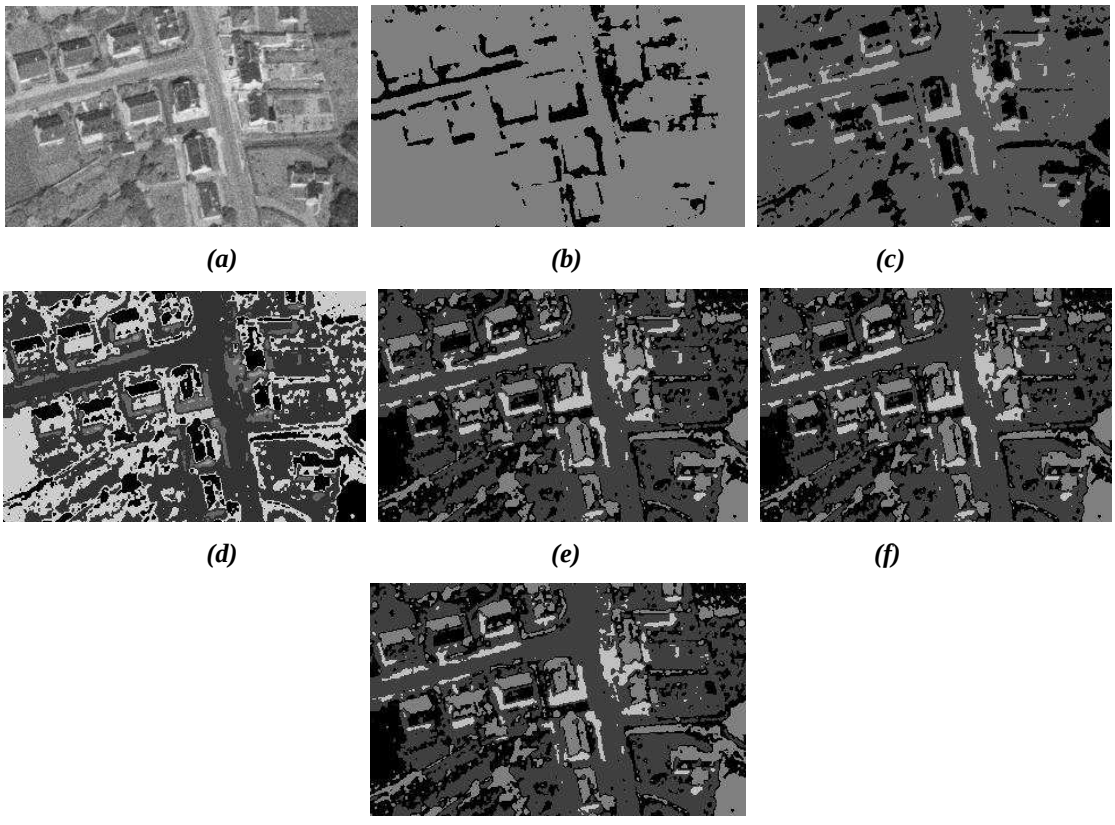
Pour la base d'exemples des images de synthèse, nous connaissons le nombre de textures qui existent dans l'image avant la segmentation, pour cela nous avons pris un nombre fixe d'exemple pour chaque texture, mais pour les autres types d'images, nous prendrons des échantillons aléatoirement.

Les paramètres :

- La fenêtre de segmentation **3x3.**
- La fenêtre d'apprentissage **3x3.**
- Les caractéristiques utilisées : **l'homogénéité et la corrélation.**
- $a_{\max}$ =10
- $\eta_g$ =0.09
- $\eta_v$ =0.01

## 1-4-1- Les images aériennes en milieu urbain

La figure Fig.4.9 montre les résultats de la segmentation d'une image aérienne en milieu urbain en changeant la valeur de $P_r$ de 1 jusqu'à 6. Les trois premiers résultats (Fig.4.9(b), Fig.4.9(c) et Fig.4.9(d)) montrent que le nombre de classes augmente, mais les trois derniers résultats montre une stabilisation de nombre de classes (les trois images donnent presque le même résultat de segmentation).



***(a)*** ***(b)*** ***(c)***

***(d)*** ***(e)*** ***(f)***

***(g)***

***Fig.4.9. Image aérienne en milieu urbain : (a) image originale, image segmentée : (b) Pr=1, (c) Pr=2, (d) Pr=3, (e) Pr=4, (f) Pr=5, (g) Pr=6.***

Nous avons continué l'apprentissage du réseau IGNGAT de l'image Fig.4.9(a) jusqu'à $P_r$=21, et nous avons dessiné le graphe qui montre le changement du nombre de neurones dans le réseau et le nombre de classes, par rapport au changement de la valeur de $P_r$. Ce graphe est représenté par la figure Fig.4.10. En regardant le graphe nous remarquons que le nombre de classes et le nombre de neurones reste invariant dans un intervalle. Le nombre de classes est constant entre 4 et 6 classes et le nombre de neurones entre 11 et 18 neurones pour un $P_r$ situé entre 6 et 21.

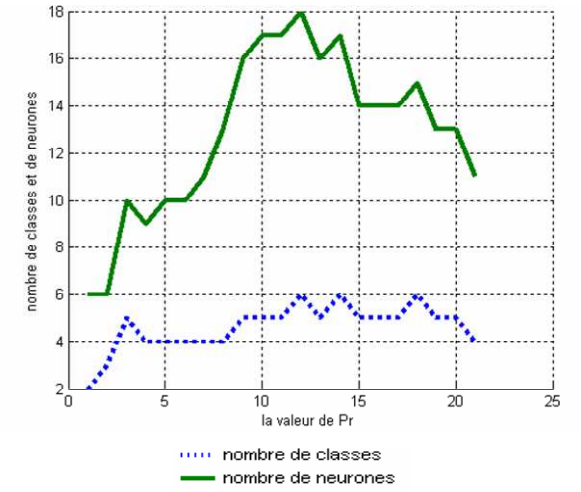
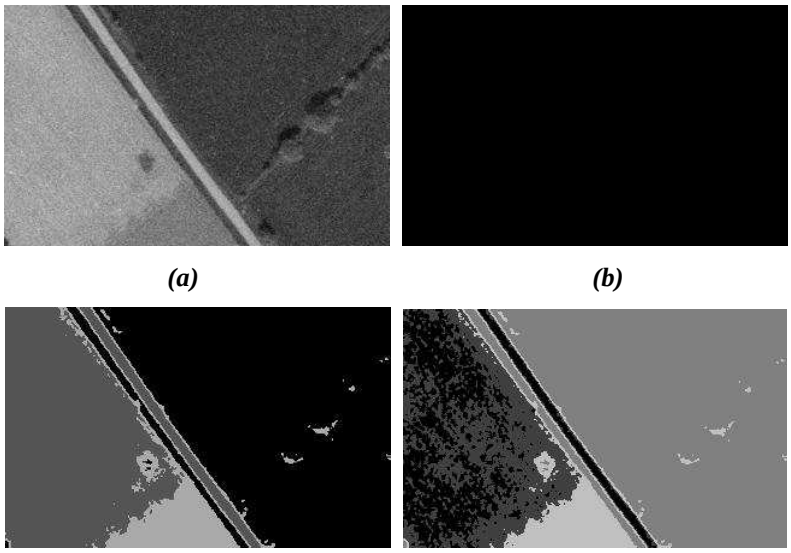



***Fig. 4.10 Le nombre de neurones dans le réseau IGNGAT et le nombre de classes détectées par rapport à la valeur de $P_r$.***

## 1-4-2- Les images aériennes en milieu rural



***(a)*** ***(b)***

***(c)*** ***(d)***

***Fig.4.11. Image aérienne en milieu rural : (a) image originale, image segmentée :(b) $P_r$=1, (c) $P_r$=2, (d) $P_r$=3.***

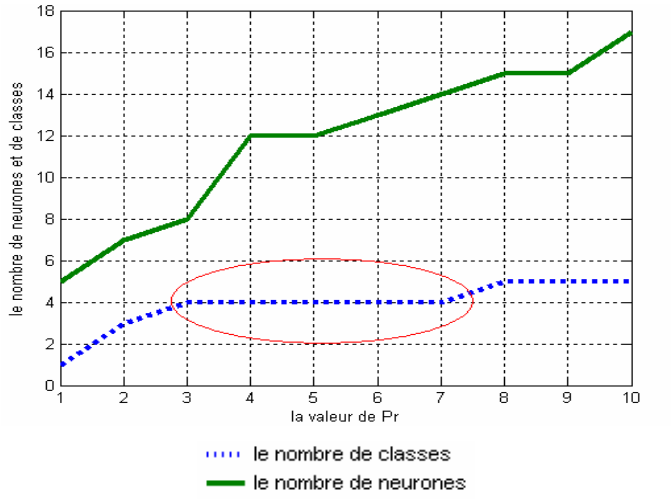
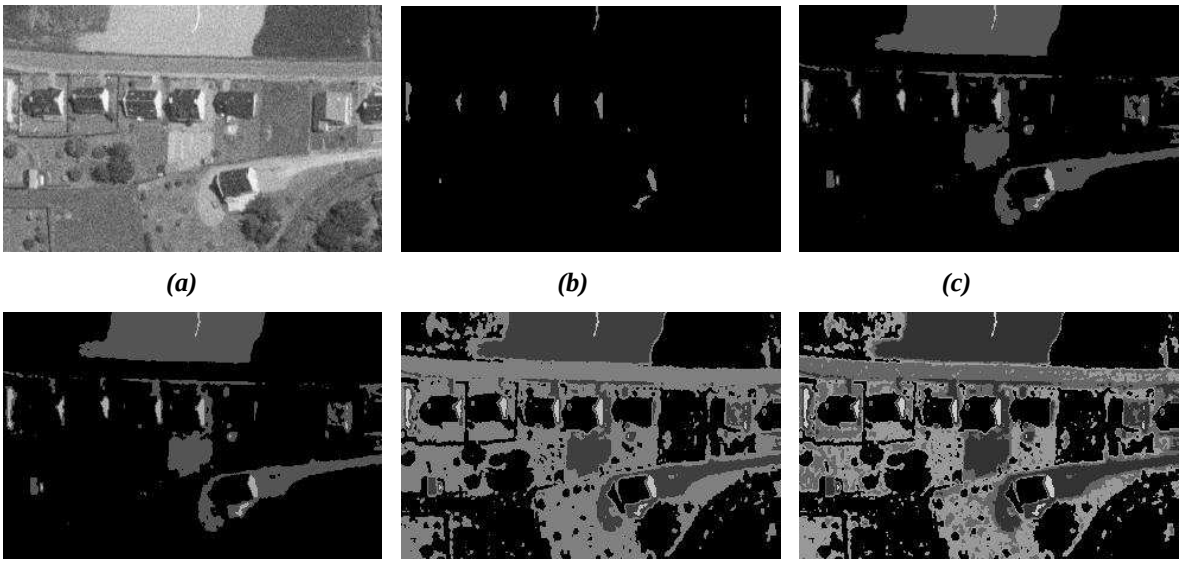



***Fig. 4.12. Le nombre de neurones dans le réseau IGNGAT et le nombre de classes détectées par rapport à la valeur de $P_r$.***

## 1-4-3- Les images aériennes en milieu semi urbain



***(a)*** ***(b)*** ***(c)***

***(d)*** ***(e)*** ***(f)***

***Fig.4.13. Image aérienne en milieu semi urbain : (a) image originale, image segmentée : (b) $P_r$=1, (c) $P_r$=2, (d) $P_r$=3, (e) $P_r$=4, (f) $P_r$=5.***

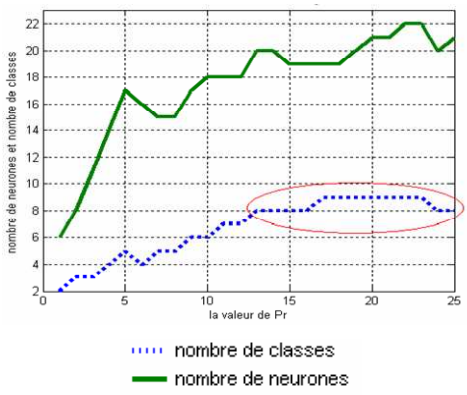
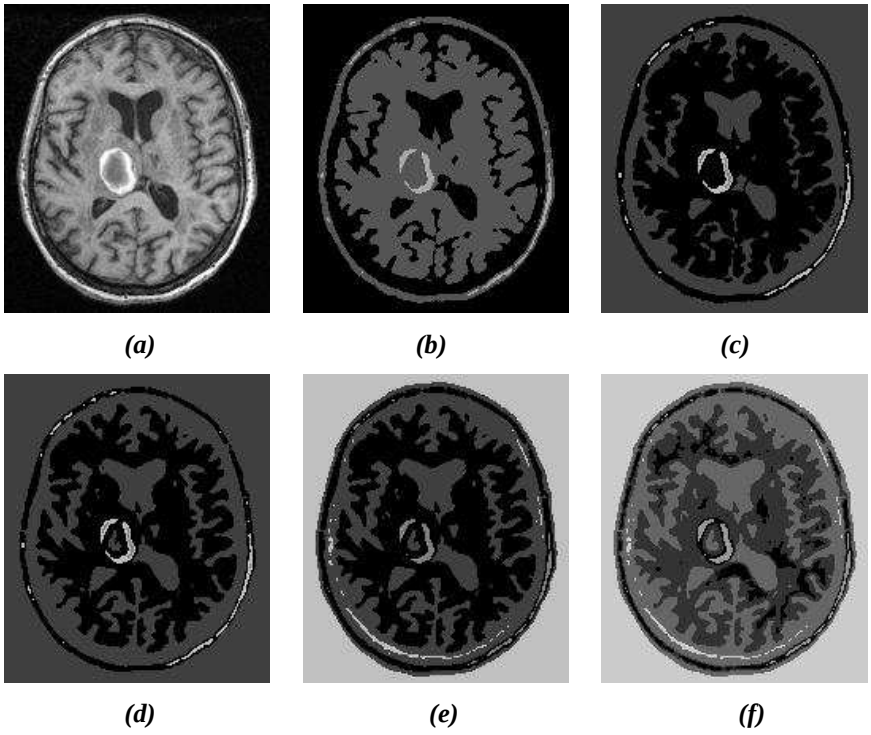



***Fig.4.14. Le nombre de neurones dans le réseau IGNGAT et le nombre de classes détectées par rapport à la valeur de $P_r$.***

Nous remarquons la même chose dans l'image en milieu rural et semi urbain, et en plus nous remarquons que la valeur de $P_r$ dépend de la taille des régions dans l'image segmentée c'est-à-dire, si l'image a des région homogènes de petite taille la valeur de $P_r$ doit être grande comme dans l'image en milieu semi urbain ($P_r$ entre 13 et 25), et le contraire si l'image a des régions homogènes de grande taille, comme dans l'image en milieu rural($P_r$ entre 3 et 7).

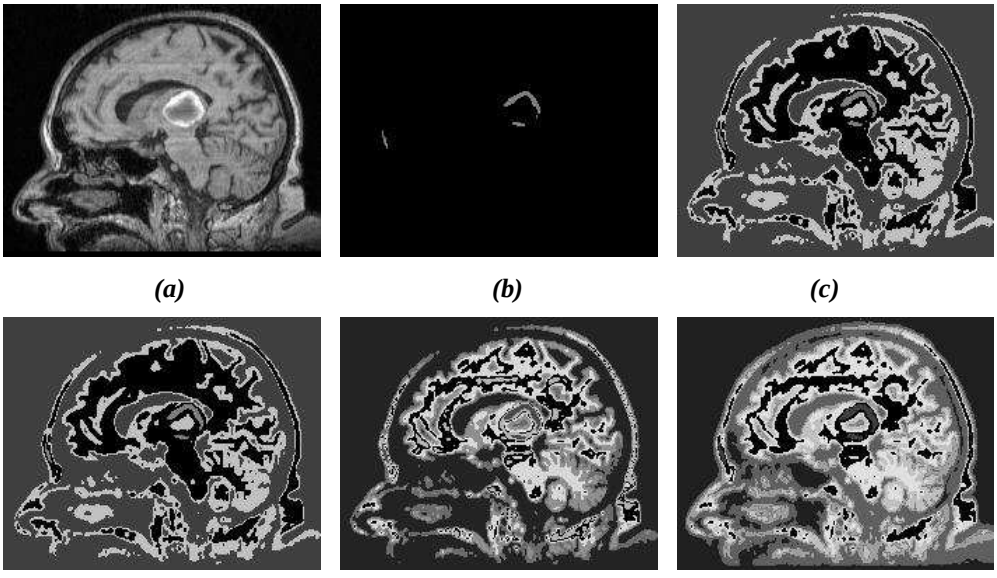
## 1-5- Les images médicales



***(a)*** ***(b)*** ***(c)***

***(d)*** ***(e)*** ***(f)***

***Fig.4.15. Coupe transversal d'un crâne humain (IGNGAT) (a) image originale, image segmentée (b) $P_r$=1, (c) $P_r$=3, (d) $P_r$=7, (e) $P_r$=14, (f) $P_r$=21 .***



***(a)*** ***(b)*** ***(c)***

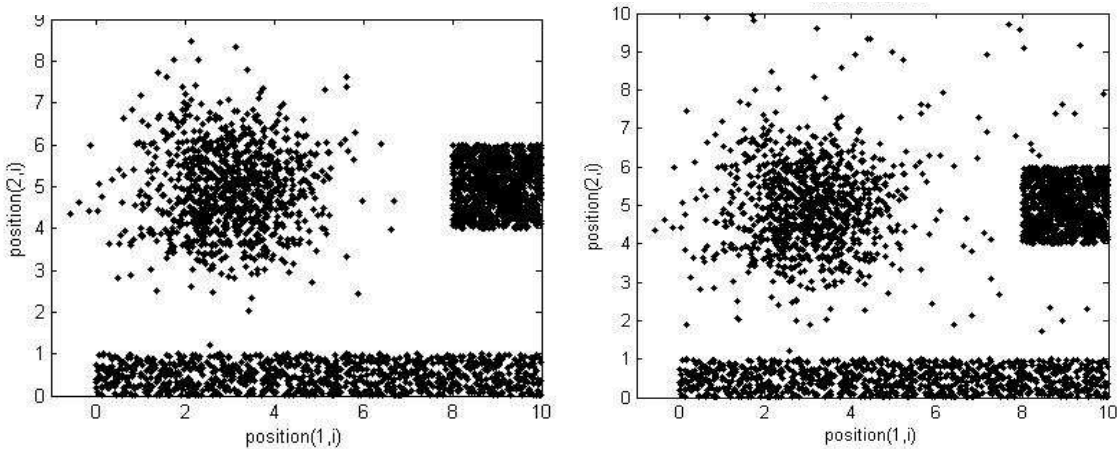
***(d)*** ***(e)*** ***(f)***

***Fig.4.16. Coupe longitudinal d'un crâne humain (IGNGAT) (a) image originale, image segmentée (b) $P_r$=1, (c) $P_r$=3, (d) $P_r$=7, (e) $P_r$=14, (f) $P_r$=21 .***

# II- Les résultats de IGNGU

## 2-1- Une base de données artificielle

Dans une première étape, nous avons testé la méthode proposée sur une base de données artificielle contenant 3 classes. La première classe a une forme sphérique où les données sont distribuées par la loi normale. La deuxième classe a une forme rectangulaire et la troisième classe a une forme linéaire où les données sont distribuées uniformément dans les formes représentée par la figure Fig.4.17. (a) qui présente la base de données artificielle utilisée dans le cas sans bruits. Nous avons ajouté 3.5% de bruit dans la base de données artificielles représentée par la figure Fig.4.17. (b). dans chaque classe il y a 10 000 exemples. Donc la base contient 30 000 exemples.




***(a)*** ***(b)***

***Fig.4.17. La base de données artificielle, (a) sans bruit, (b) avec bruit.***

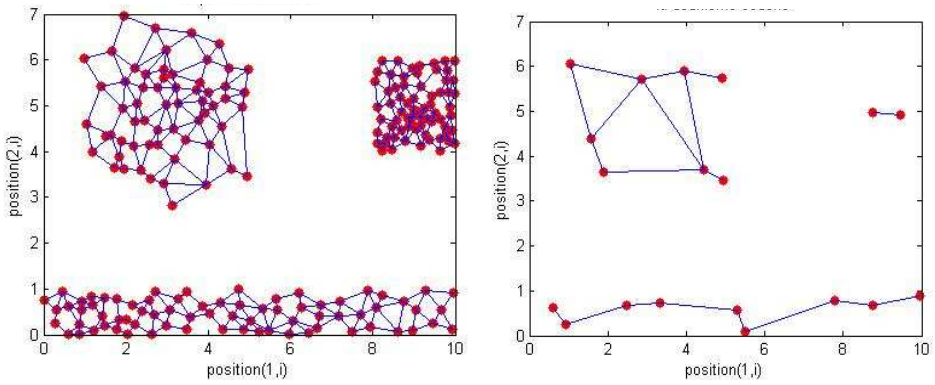
### 2-1-1- Cas passif (non incrémental)

***a) Sans bruit***

Les paramètres utilisés pour l'apprentissage des deux couches sont cités dans le tableau suivant. Dans chaque itération nous prenons aléatoirement un exemple de la base d'apprentissage (voir le résultat dans la figure Fig.4.18). La figure Fig 4.19 présente le changement de nombre de neurones dans les deux couches chaque $\tau$ itérations. Dans la première couche, le nombre de neurones augmente. Et après, 150 $\tau$ itérations, il se stabilise dans l'intervalle [200,210] (voir Fig.4.19 (a)). Et dans la deuxième couche le nombre de neurones reste égal à 19 neurones après les premières $\tau$ itérations (voir Fig.4.19 (b)).

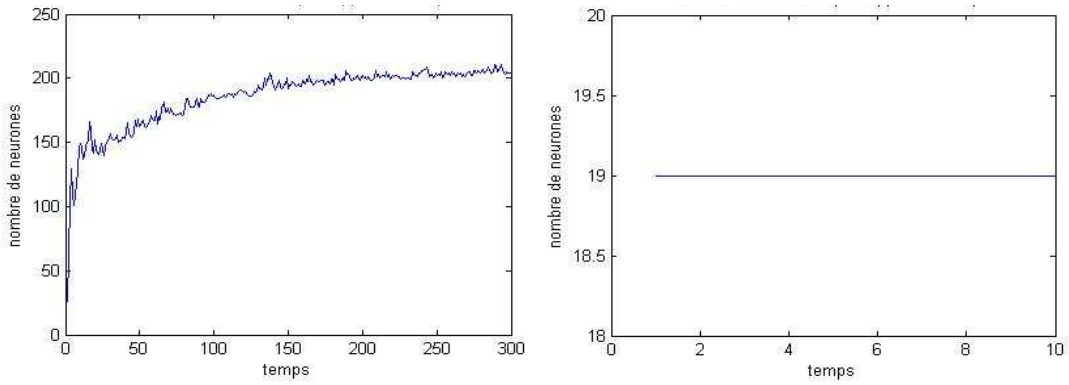
| Les paramètres / Les couches | $a_{max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 100 | 100 | 1 | *1/t* | *1/(100*t)* | 30 000 |
| Deuxième couche | 50 | Nombre de neurones de la première couche | 0.01 | *1/t* | *1/(100*t)* | $\tau$*10 |

***Tab.4.1. Les paramètres utilisés dans le cas passif sans bruits (base de données artificielle).***




***(a)*** ***(b)***

***Fig.4.18. Le résultat du IGNGU sur la base sans bruit, (a) la première couche, (b) la deuxième couche.***




***(a)*** ***(b)***

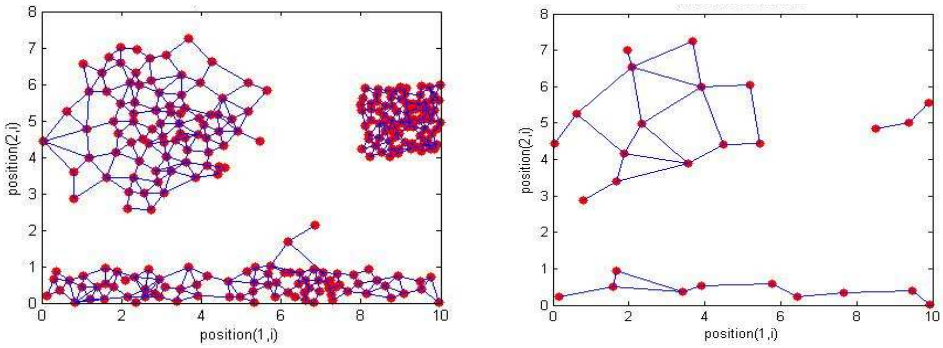
***Fig.4.19. Le nombre de neurones par rapport au temps dans le cas passif sans bruit, (a) pour la première couche, (b) pour la deuxième couche.***

### *b) Avec bruit*

Les paramètres utilisés pour l'apprentissage des deux couches sont cités dans le tableau suivant. Dans chaque itération nous prenons aléatoirement un exemple de la base d'apprentissage (voir le résultat dans Fig.4.20.). A cause du bruit, le nombre de neurones dans les deux couche ne se stabilise pas rapidement mais après un certain temps. Dans la première couche, le nombre de neurones se stabilise après 200 $\tau$ itérations dans l'intervalle [250, 260] (voir Fig.4.21 (a)). Et dans la deuxième couche il se stabilise après 9 $\tau$ itérations à 27 neurones. Donc le IGNGU est capable d'apprendre des données en présence de bruit en éliminant ce dernier.

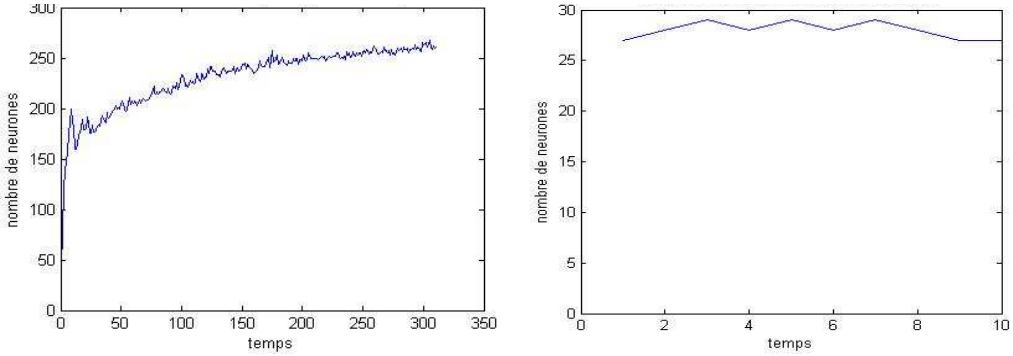
| Les paramètres / Les couches | $a_{max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 25 | 100 | 1 | *1/t* | *1/(100*t)* | 30 000 |
| Deuxième couche | 50 | Nombre de neurones de la première couche | 0.05 | *1/t* | *1/(100*t)* | $\tau$ *10 |

***Tab.4.2. Les paramètres utilisés dans le cas passif avec bruits (base de données artificielle).***




***(a)*** ***(b)***

***Fig.4.20. Le résultat de la méthode proposée sur la base sans bruit, (a) la première couche, (b) la deuxième couche.***




***(a)*** ***(b)***

***Fig.4.21. Le nombre de neurones par rapport au temps dans le cas passif avec bruit, (a) pour la première couche, (b) pour la deuxième couche.***

## 2-1-2- Cas incrémental

Pour monter l'efficacité de la méthode proposée dans le cas incrémental, nous avons divisé la base de données artificielle en deux parties. La première partie contient la classe sphérique plus la moitié de la classe linéaire et la deuxième partie contient le reste de la classe linéaire plus la classe rectangulaire. Dans les deux cas sans et avec bruit. Ensuite l'apprentissage des différentes parties est effectue successivement, pour les deux couches. Dans chaque itération nous prenons aléatoirement un exemple de la partie de la base d'apprentissage.

### *a) Sans bruit*

Les paramètres utilisés pour l'apprentissage des deux couches sont cités dans Tab.4.3.

| Les paramètres / Les couches | $a_{max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 100 | 100 | 1 | *1/t* | *1/(100*t)* | 15 000 |
| Deuxième couche | 50 | Nombre de neurones de la première couche | 0.01 | *1/t* | *1/(100*t)* | $\tau$ *10 |

***Tab.4.3. Les paramètres utilisés dans le cas incrémental sans bruits (base de données artificielle).***

La figure Fig.4.22 nous présente les résultats d'apprentissage du IGNGU pour chaque partie des deux couches. Cette figure montre deux choses : le réseau IGNGU est incrémental c'est-à-dire, il est cabale d'apprendre des nouvelles données sans détruire les données déjà prises même si celles-ci ne sont plus disponible pendant l'apprentissage des nouvelles données. Et que les deux couches peuvent opérer en parallèle. C'est-à-dire, pendant que la deuxième couche effectue l'apprentissage de la première partie, la première couche démarre l'apprentissage de la deuxième partie. Et la figure Fig.4.23 nous montre que le nombre de neurones, pour les deux couches pendant l'apprentissage des deux parties se stabilise.

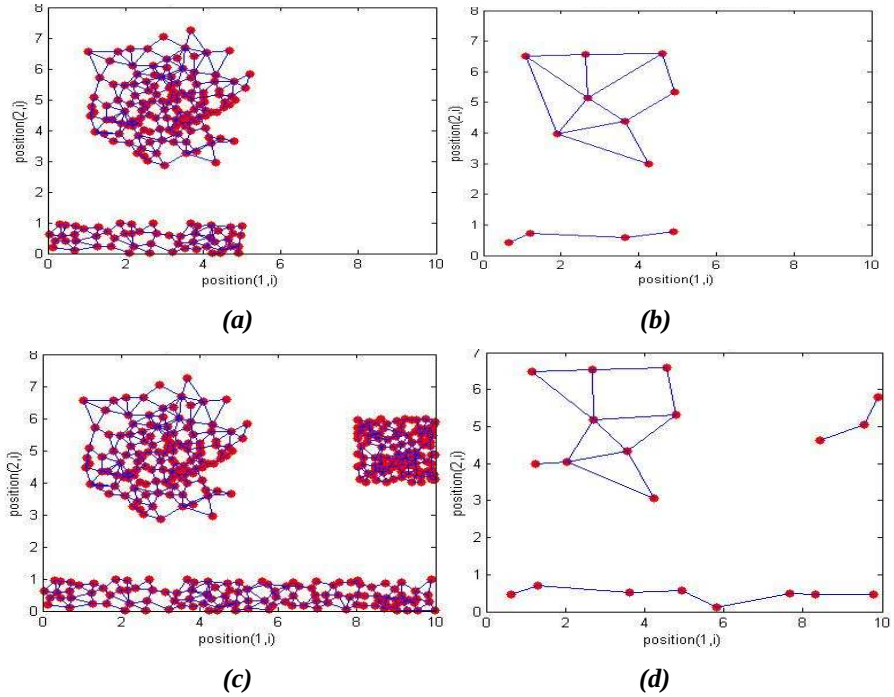
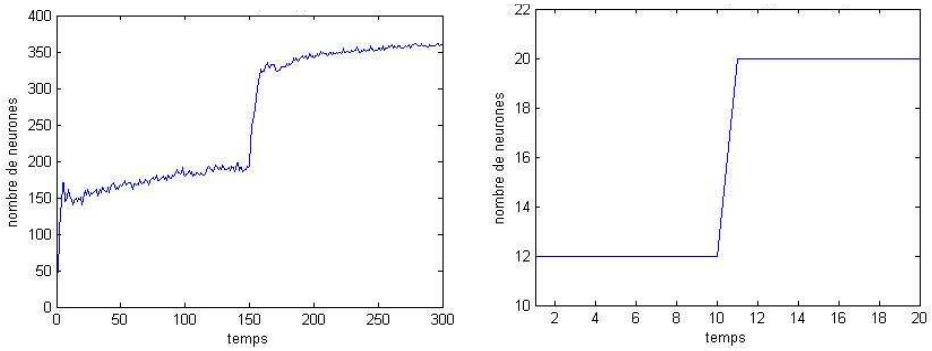



***Fig.4.22. Le résultat de la méthode proposée sur la base sans bruit dans le cas incrémental. La première partie : (a) la première couche, (b) la deuxième couche. La deuxième partie : (c) la première couche, (b) la deuxième couche.***

### *b) Avec bruit*

Nous avons ajouté 3.5% du bruit dans chaque partie. Les paramètres utilisés pour l'apprentissage des deux couches sont cités dans tab.4.4.

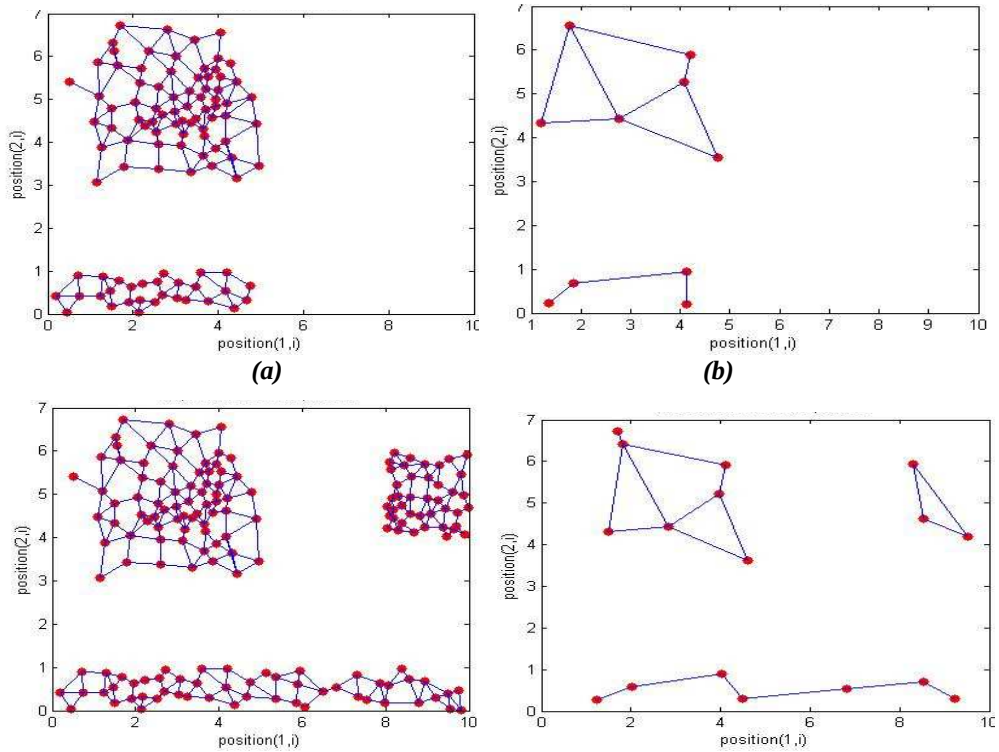



*(a)* *(b)*

***Fig.4.23. Le nombre de neurones par rapport au temps dans le cas incrémental sans bruit, (a) pour la première couche, (b) pour la deuxième couche.***

| Les paramètres / Les couches | $a_{\max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 25 | 25 | 1 | *1/t* | *1/(100*t)* | 15 500 |
| Deuxième couche | 50 | Nombre de neurones de la première couche | 0.05 | *1/t* | *1/(100*t)* | $\tau$ *10 |

***Tab.4.4. Les paramètres utilisés dans le cas incrémental avec bruits (base de données artificielle).***

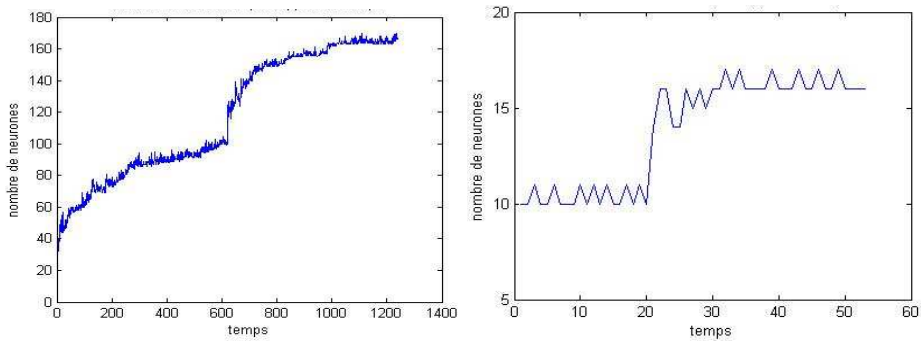



*(a)* *(b)*

*(c)* *(d)*

***Fig.4.24. Le résultat de la méthode proposée sur la base sans bruit dans le cas incrémental. La première partie : (a) la première couche, (b) la deuxième couche. La deuxième partie : (c) la première couche, (b) la deuxième couche.***

La figure Fig.4.24 montre que même avec le bruit, IGNGU est un réseau incrémental capable d'éliminer ce dernier. Le nombre de neurones se stabilise dans un intervalle dans les deux couches pendant l'apprentissage des deux parties.




***(a)*** ***(b)***

***Fig.4.25. Le nombre de neurones par rapport au temps dans le cas incrémental avec bruit, (a) pour la première couche, (b) pour la deuxième couche.***

## 2-2- La classification de textures

Nous avons appliqué l'algorithme proposé IGNGU pour la segmentation des images de textures de différents types (fine –grossière – périodique et apériodique). Ces images sont constituées d'images de synthèse extraites de l'album de Broadatz [18], d'images aériennes et des images médicales.

Le choix de la base d'apprentissage est faite par la sélection aléatoire sur l'image. La taille de la fenêtre utilisée dans l'apprentissage et le choix des paramètres de texture utilisés dépendent de type d'image à segmenter.

Dans la segmentation des images de textures, il existe un grand problème dans les frontières des régions. Si nous utilisons une grande fenêtre de segmentation, les frontières de régions seront mal détectées et si, nous utilisons une petite fenêtre de segmentation les détails dans les textures sont détectés. La solution qui nous proposons, est d'utiliser des fenêtres de tailles différentes dans la segmentation d'images. C'est-à-dire nous segmentons l'image avec une fenêtre de segmentation grande N*N. Et nous gardons pour chaque pixel l'étiquette et la distance. Puis nous diminuions la taille de la fenêtre (N1*N1 où N1<N) et vérifions que si un pixel dans l'image segmentée est non classé ou si la distance entre le vecteur de caractéristiques de la ROI du pixel et le neurone gagnant est plus petite que l'ancienne distance où la fenêtre est plus grande. Nous affectons alors à ce pixel l'étiquette du neurone gagnant. Et comme dans les images de textures nous nous intéressons à l'ensemble des pixels et non à un pixel donc nous affectons l'étiquette du neurone gagnant à une fenêtre autour du pixel centre, que nous appelons la fenêtre d'affectation (FF) N2*N2 où N2<N (voir Fig. 4. 26). Cette méthode améliore le résultat de la segmentation dans le cas de textures grossières (voir Fig. 4. 27), dans cette figure nous voyons que le résultat de segmentation est amélioré avec l'augmentation de la taille de la fenêtre d'affectation. La région noire contient les pixels non classés.

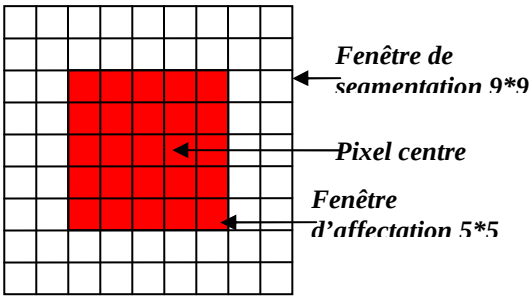



*Fig.4.26. Exemple d'une fenêtre de segmentation.*

*(a)* *(b)* *(c)* *(d)* *(e)*

*Fig.4.27. Segmentation d'une image de synthèse de 5 textures avec FS=65*65 en changeant la taille de la fenêtre d'affectation. (a) l'image originale, les images segmentées : (b) FF=1*1, (c) FF=15*15, (d) FF=25*25, (e) FF=35*35.*

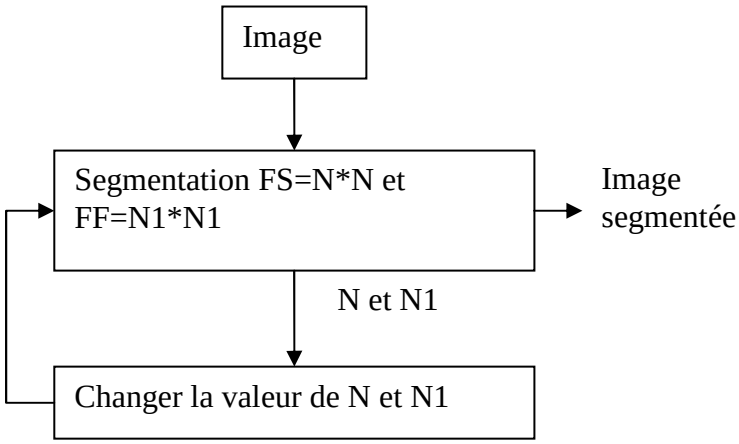



*Fig.4.28. La segmentation hiérarchique.*

## 2-2-1- Les images de synthèse

### *a) Les images de 4 textures de Broadatz*

Les paramètres de texture utilisés sont l'homogénéité et l'inertie. Les fenêtres d'apprentissage utilisées sont de taille 65*65, 55*55, 45*45. Les fenêtres de segmentation utilisées sont de taille 65*65, 55*55, 45*45, 35*35, 25*25, 15*15, 5*5.

Pour la base d'apprentissage nous choisissons 20 exemples pour chaque texture et pour chaque fenêtre d'apprentissage. C'est-à-dire nous avons 240 exemples dans la base d'apprentissage.

Les paramètres utilisés :

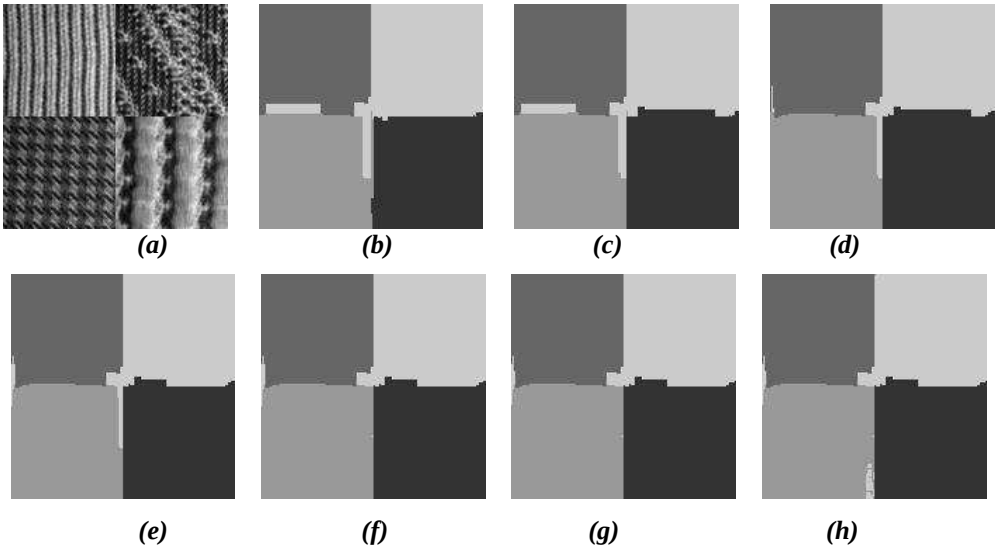
| Les paramètres / Les couches | $a_{\max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 60 | 120 | 1 | 0.01 | 0.001 | 2 400 |
| Deuxième couche | 20 | Nombre de neurones de la première couche | 0.01 | 0.01 | 0.001 | $\tau$ *10 |

***Tab.4.5. Les paramètres utilisés (images de 4 textures de Broadatz).***

### *b) Les images de 5 textures*

Les paramètres de texture utilisés sont l'homogénéité et la corrélation. Les fenêtres d'apprentissage utilisées sont de taille 65*65, 55*55, 45*45 et 35*35. Les fenêtres de segmentation utilisées sont de taille 65*65, 55*55, 45*45, 35*35, 25*25, 15*15, 5*5.

Pour la base d'apprentissage nous choisissons 40 exemples pour chaque texture et pour chaque fenêtre d'apprentissage. C'est-à-dire nous avons 800 exemples dans la base d'apprentissage.



***(a)*** ***(b)*** ***(c)*** ***(d)***

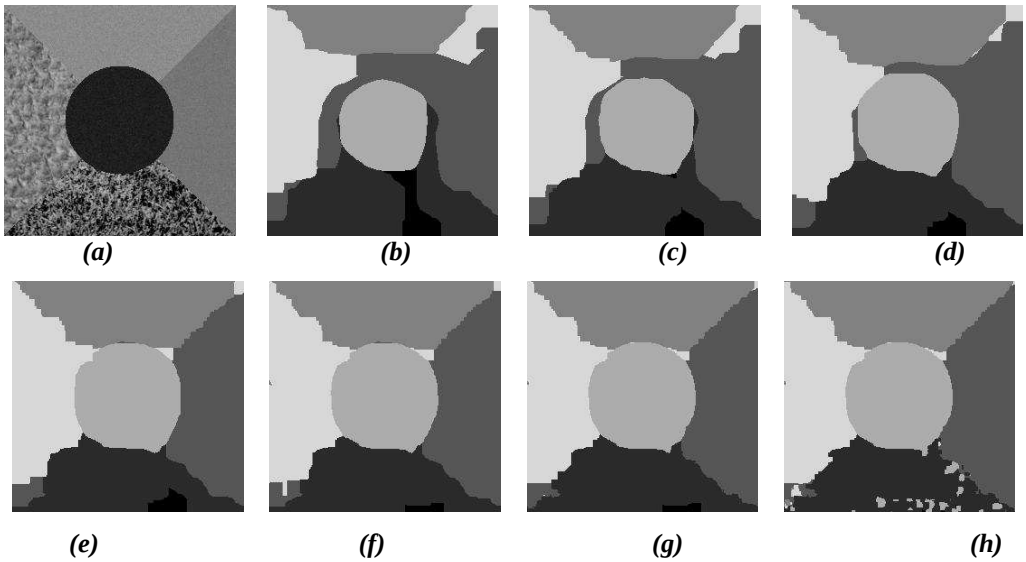
***(e)*** ***(f)*** ***(g)*** ***(h)***

***Fig.4.29. Le résultat de la segmentation d'une image de 4 textures, (a) l'image originale, les images segmentées : (b) FS=65*65 et FF=65*65, (c) FS=55*55 et FF=55*55, (d) FS=45*45 et FF=45*45, (e) FS=35*35 et FF=35*35, (f) FS=25*25 et FF=25*25, (g) FS=15*15 et FF=15*15, (h) FS=5*5 et FF=5*5.***

Les paramètres utilisés :

| Les paramètres / Les couches | $a_{\max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 20 | 20 | 1 | 0.01 | 0.001 | 8 000 |
| Deuxième couche | 20 | Nombre de neurones de la première couche | 0.05 | 0.01 | 0.001 | $\tau$ *10 |

***Tab.4.6. Les paramètres utilisés (images de 5 textures).***



*(a)* *(b)* *(c)* *(d)*

*(e)* *(f)* *(g)* *(h)*

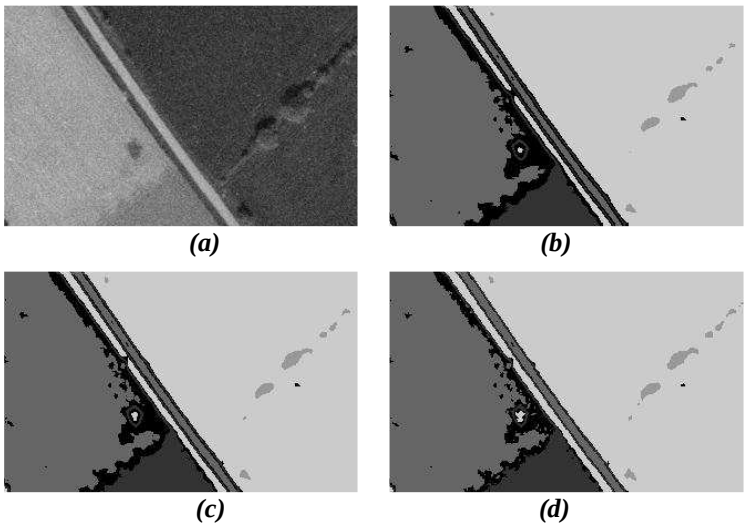
***Fig.4.30. Le résultat de la segmentation d'une image de 5 textures, (a) l'image originale, les images segmentées : (b) FS=65*65 et FF=35*35, (c) FS=55*55 et FF=35*35, (d) FS=45*45 et FF=35*35, (e) FS=35*35 et FF=35*35, (f) FS=25*25 et FF=25*25, (g) FS=15*15 et FF=15*15, (h) FS=5*5 et FF=5*5.***

## 2-2-2- Les images ariennes

Les paramètres de texture utilisés sont l'homogénéité et la corrélation. Les fenêtres d'apprentissage utilisées sont de taille 7*7, 5*5, 3*3. Les fenêtres de segmentation utilisées sont de taille 7*7, 5*5, 3*3.

Pour la base d'apprentissage nous choisissons 120 exemples pour chaque fenêtre d'apprentissage. C'est-à-dire nous avons 480 exemples dans la base d'apprentissage.

Les paramètres utilisés sont présentés dans le tableau Tab.4.7.



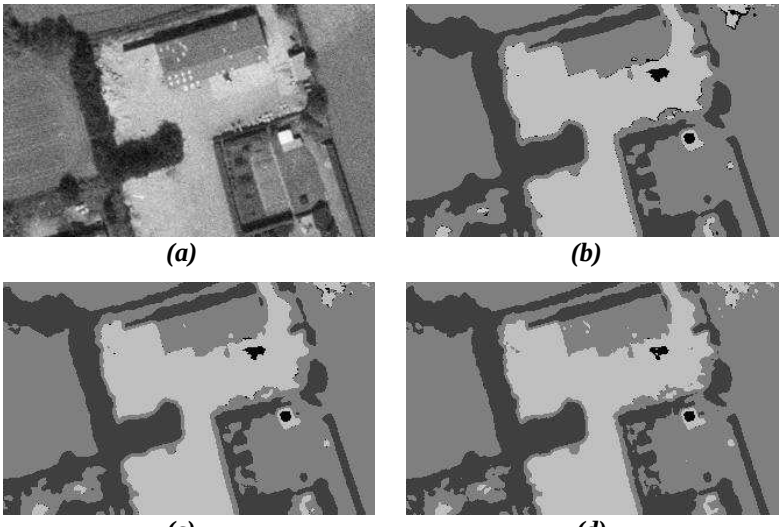
*(a)* *(b)*

*(c)* *(d)*

***Fig.4.31. Le résultat de la segmentation d'image au milieu rural, (a) l'image originale, les images segmentées : (b) FS=7*7 et FF=1*1, (c) FS=5*5 et FF=1*1, (d) FS=3*3 et FF=1*1.***

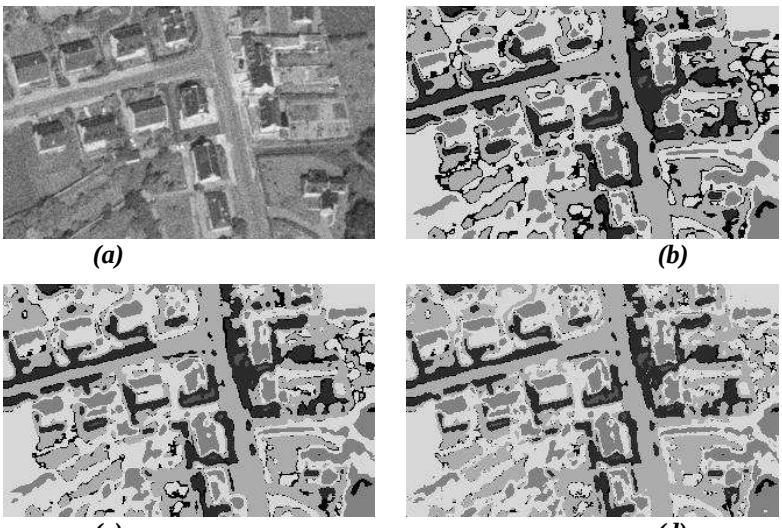
| Les paramètres / Les couches | $a_{max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 20 | 20 | 1 | 0.01 | 0.001 | 4 800 |
| Deuxième couche | 20 | Nombre de neurones de la première couche | 0.05 | 0.01 | 0.001 | $\tau$ *10 |

***Tab.4.7. Les paramètres utilisés (images ariennes).***



***(a)*** ***(b)*** ***(c)*** ***(d)***

***Fig.4.32. Le résultat de la segmentation d'image au milieu semi rural, (a) l'image originale, les images segmentées : (b) FS=7*7 et FF=1*1, (c) FS=5*5 et FF=1*1, (d) FS=3*3 et FF=1*1.***



***(a)*** ***(b)*** ***(c)*** ***(d)***

***Fig.4.33. Le résultat de la segmentation d'image au milieu urbain, (a) l'image originale, les images segmentées : (b) FS=7*7 et FF=1*1, (c) FS=5*5 et FF=1*1, (d) FS=3*3 et FF=1*1.***

## 2-2-3- Les images médicales

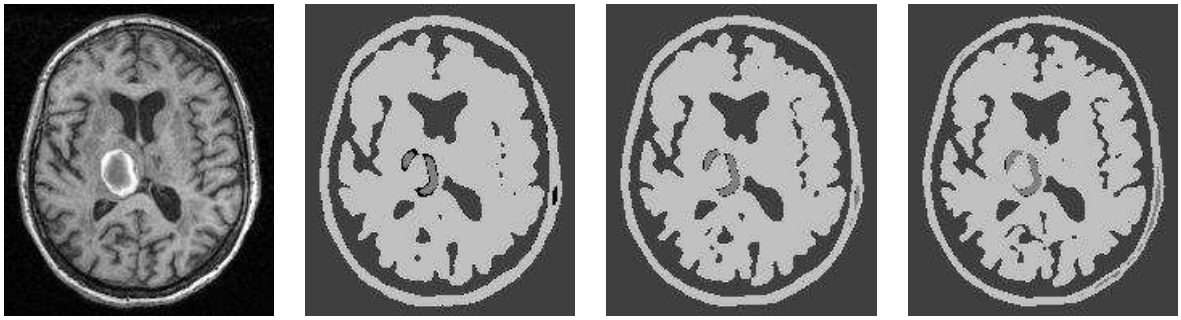
Les paramètres de texture utilisés sont l'homogénéité et la corrélation. Les fenêtres d'apprentissage utilisées sont de taille 7*7, 5*5, 3*3. Les fenêtres de segmentation utilisées sont de taille 7*7, 5*5, 3*3.

Pour la base d'apprentissage nous choisissons 60 exemples pour chaque fenêtre d'apprentissage. C'est-à-dire nous avons 240 exemples dans la base d'apprentissage.

Les paramètres de la méthode utilisés :

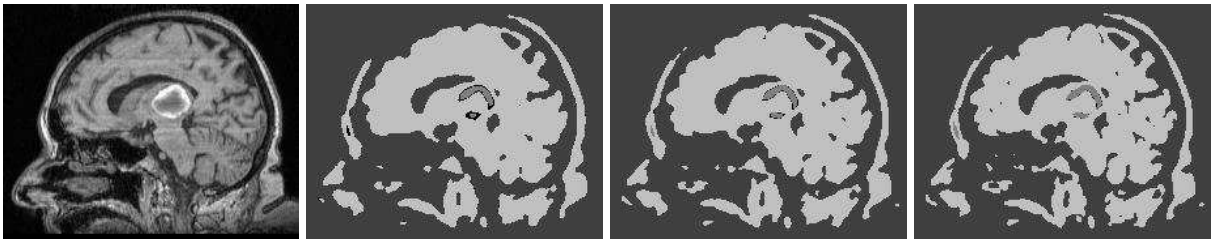
| Les paramètres / Les couches | $a_{max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 20 | 60 | 1 | 0.01 | 0.001 | 2 400 |
| Deuxième couche | 20 | Nombre de neurones de la première couche | 0.05 | 0.01 | 0.001 | $\tau$ *10 |

***Tab.4.8. Les paramètres utilisés (images médicales).***



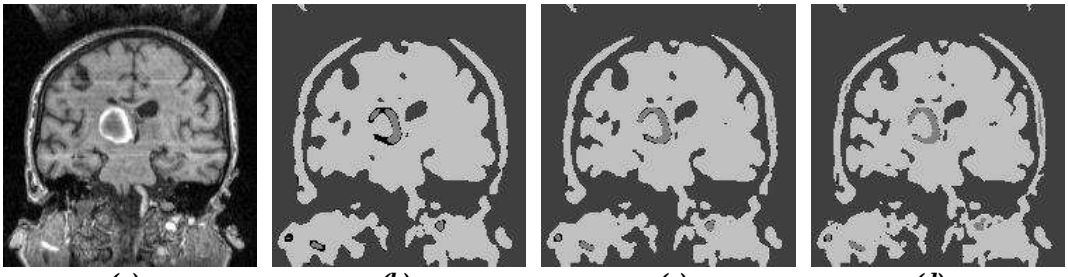
***(a)*** ***(b)*** ***(c)*** ***(d)***

***Fig.4.34. Coupe transversal d'un crâne humain (IGNGU), (a) l'image originale, les images segmentées : (b) FS=7*7 et FF=1*1, (c) FS=5*5 et FF=1*1, (d) FS=3*3 et FF=1*1.***



***(a)*** ***(b)*** ***(c)*** ***(d)***

***Fig.4.35. Coupe longitudinal d'un crâne humain (IGNGU), (a) l'image originale, les images segmentées: : (b) FS=7*7 et FF=1*1, (c) FS=5*5 et FF=1*1, (d) FS=3*3 et FF=1*1.***



***(a)*** ***(b)*** ***(c)*** ***(d)***

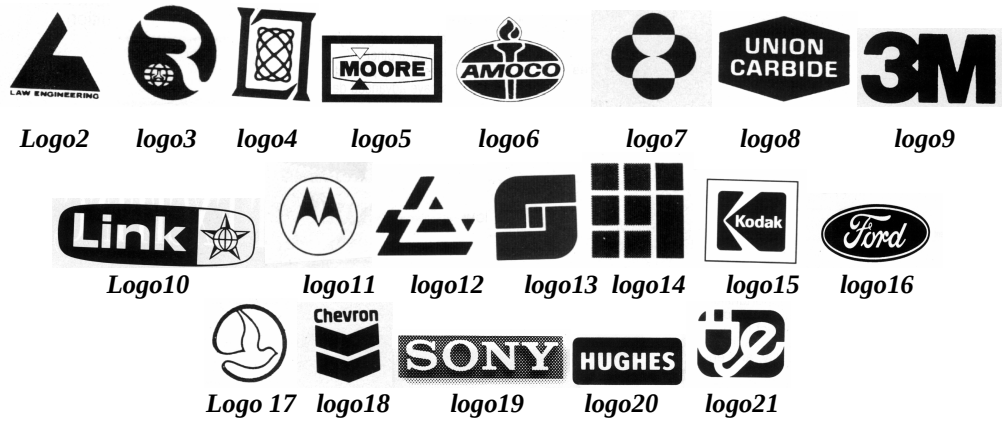
***Fig.4.36. Coupe longitudinal 2 d'un crâne humain (IGNGU), (a) l'image originale, les images segmentées : (b) FS=7*7 et FF=1*1, (c) FS=5*5 et FF=1*1, (d) FS=3*3 et FF=1*1.***

## 2-3- La classification des LOGOS

Nous avons appliqué la méthode proposée pour la classification des logos. Nous avons utilisé les moments invariants pour la caractérisation des logos. Dans la base d'apprentissage nous avons pris 20 logos (voir Fig.4.36), et pour chaque logo, nous avons créé 24 échantillons par rotation et changement d'échelle.

| *Facteur de changement d'échelle* | *0.2* | *0.5* | *0.7* | *0.8* | *2* | *4* | *6* | *8* |
|---|---|---|---|---|---|---|---|---|
| *Degré de rotation* | *5°* | *15°* | *50°* | *60°* | *-30°* | *-110°* | *140°* | *-90°* |
| | *0.2 et 5°* | *0.5 et 15°* | *0.7 et 50°* | *0.8 et 60°* | *2 et -30°* | *4 et -110°* | *6 et 140°* | *8 et -90°* |

***Tab.4.9. Les paramètres utilisés pour la création de la base d'apprentissage des logos.***




***Logo2 logo3 logo4 logo5 logo6 logo7 logo8 logo9***

***Logo10 logo11 logo12 logo13 logo14 logo15 logo16***

***Logo 17 logo18 logo19 logo20 logo21***

***Fig.4.37. La base de logos utilisé.***

### 2-3-1- Les moments invariants

***a) Les moments standards***

L'équation générale des moments est la suivante [6]:

$$M_{i,j} = \sum_{k=1}^{K} \sum_{l=1}^{L} (k)^i (l)^j g(k,l)$$

Où *g(k ,l)* est le niveau de gris de l'image défini pour *k=1,…,K* et *l=1,…,L*.

***b) Le centre de gravité***

Le centre de gravité d'une image $(\bar{k}, \bar{l})$ est défini par les relations suivantes :

$$\bar{k} = \frac{M_{1,0}}{M_{0,0}}, \quad \bar{l} = \frac{M_{0,1}}{M_{0,0}}$$

Où $M_{0,0}$ est la surface de la forme dans le cas d'une image binaire.

Afin de rendre les moments $M_{i,j}$ invariants par translation, on les définit en choisissant $(\bar{k},\bar{l})$ comme origine. On parle alors de moments centrés $\mu_{i,j}$ dont la définition formelle est la suivante :

$$\mu_{i,j} = \sum_{k=1}^{K}\sum_{l=1}^{L}(k-\bar{k})^{i}(l-\bar{l})^{j} g(k,l)$$

Afin de rendre les moments $\mu_{i,j}$ invariants par rotation, les moments suivants ont été introduits :

$$M_1 = \mu_{20} + \mu_{02}$$

$$M_2 = (\mu_{20} - \mu_{02})^2 + 4\mu_{11}$$

$$M_3 = (\mu_{30} - \mu_{12})^2 + (3\mu_{21} - \mu_{03})^2$$

$$M_4 = (\mu_{30} + \mu_{12})^2 + (\mu_{21} + \mu_{03})^2$$

$$M_5 = (\mu_{30} - 3\mu_{12})(\mu_{30} + \mu_{12})[(\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2] + (3\mu_{21} - \mu_{03})(\mu_{21} + \mu_{03})[3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2]$$

$$M_6 = (\mu_{20} - \mu_{02})[(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] + 4\mu_{11}(\mu_{30} + \mu_{12})(\mu_{21} + \mu_{03})$$

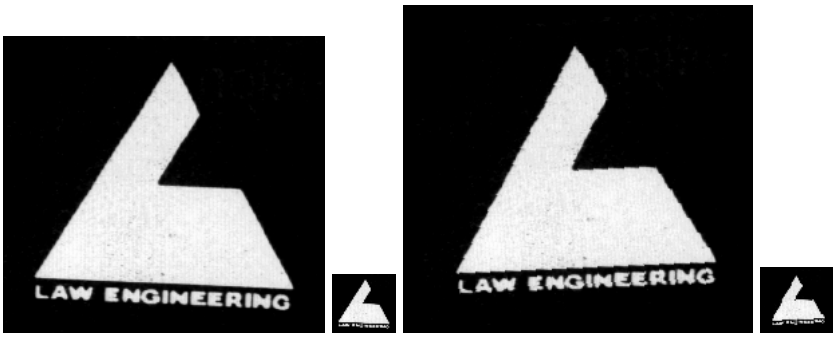
$$M_7 = (3\mu_{21} - \mu_{03})(\mu_{30} + \mu_{12})[(\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2] - (\mu_{30} - 3\mu_{12})(\mu_{21} + \mu_{03})[3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2]$$

Il faut noter que les six premiers moments $M_1$ à $M_6$ sont de plus invariants par réflexion alors que le dernier $M_7$ change de signe lors d'une réflexion. Afin de rendre les moments $M_i$ décrits précédemment invariants par grossissement, on remplace dans l'équation les moments centrés $\mu_{ij}$ par les moments centrés standard $\eta_{ij} = \dfrac{\mu_{ij}}{\mu_{00}^{\gamma}}$.

Où $\gamma = (i+j)/2 + 1$.

Après substitution, on obtient finalement les moments invariants par translation, rotation et changement d'échelle appelés $M_i^{'}$. Et normaliser par Log($M_i^{'}$).

La figure Fig.4.38 présente un exemple des transformations effectue sur l'image de logo2. Et le tableau présente les moments calculés sur chaque image.




*(a)* *(b)* *(c)* *(d)*

***Fig.4.38. Logo2, (a) l'image originale, changement d'échelle S=0.2, (c) rotation R=5°, (d) changement d'échelle plus rotation, S=0.2 et R=5°.***

| ***Les moments*** | ***(a) originale*** | ***(b) S=0,2*** | ***(c) R=5°*** | ***(d) S=0,2 R=5°*** |
|---|---|---|---|---|
| $M_1$ | *-3,0358* | *-3,0358* | *-3,0359* | *-3,0354* |
| $M_2$ | *-8,1579* | *-8,1474* | *-8,156* | *-8,1163* |
| $M_3$ | *-9,8172* | *-9,8215* | *-9,7719* | *-9,7815* |
| $M_4$ | *-11,173* | *-11,312* | *-11,175* | *-11,323* |
| $M_5$ | *-21,887* | *-22,155* | *-21,892* | *-22,078* |
| $M_6$ | *-15,257* | *-15,401* | *-15,258* | *-15,412* |
| $M_7$ | *-21,432* | *-21,631* | *-21,435* | *-21,663* |

***Tab.4.10. Les moments calculés sur les quatre images décrites à Fig.4.38.***

## 2-3-2- Cas passif

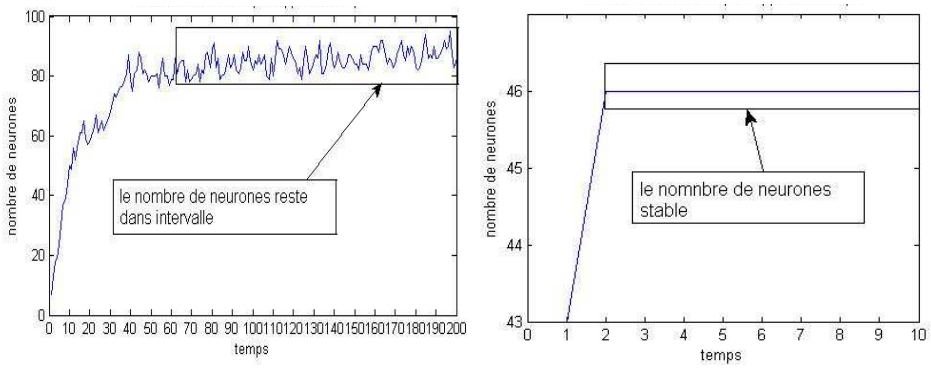
L'apprentissage du IGNGU est effectué sur la base d'apprentissage, où nous prendrons aléatoirement un exemple de la base à chaque itération. Le tableau suivant présente les paramètres utilisés dans les deux couches.

| Les paramètres / Les couches | $a_{\max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 25 | 25 | 0.1 | 0.01 | 0.001 | 5000 |
| Deuxième couche | 25 | Nombre de neurones de la première couche | 0.01 | 0.01 | 0.001 | $\tau$*10 |

***Tab.4.11. Les paramètres utilisés dans le cas passif (logos).***

Le résultat de IGNGU est 20 classes, la figure Fig.4.39 montre le changement du nombre de neurones dans la première couche et la deuxième couche.

Nous avons testé ce réseau par une base de test constituée de 200 exemples (10 exemples pour chaque logo). Le taux de rejet est égal à 14.5%, le taux de fausse classification est égal à 3.5%. Et le taux de bonne classification est égal à 82%.




*(a)* *(b)*

***Fig.4.39. Le nombre de neurones par rapport au temps dans la classification de logos dans le cas passif, (a) la première couche, (b) la deuxième couche.***

| | | | |
|---|---|---|---|
| *Facteur de changement d'échelle* | *0.1* | *0.6* | *7* |
| *Degré de rotation* | *30°* | *-5°* | *90°* |
| | *0.1 et 30°* | *0.6 et -5°* | *7 et 90°* |

***Tab.4.12. Les paramètres utilisés dans le cas passif (logos).***

## 2-3-3- Cas incrémental

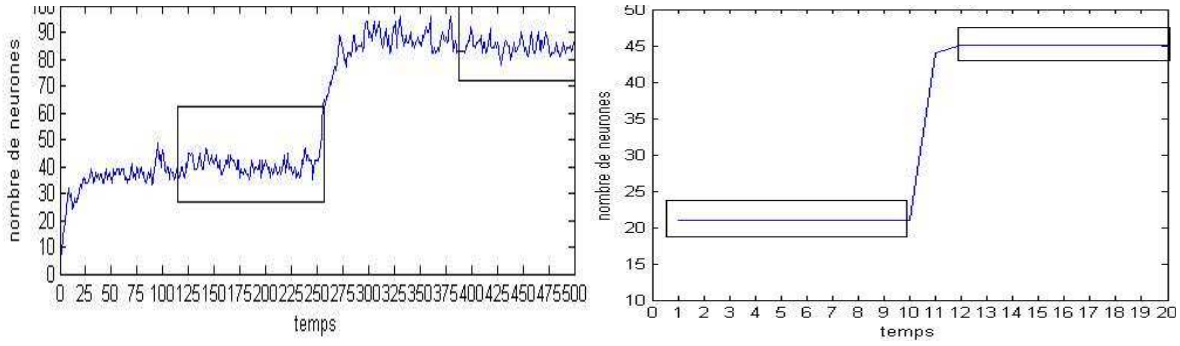
Nous avons divisé la base d'apprentissage en deux parties. Ensuite l'apprentissage des différentes parties est effectué successivement. Pour les deux couches, nous prendrons aléatoirement un exemple de la base à chaque itération. Le tableau suivant présente les paramètres utilisés dans les deux couches pour les deux parties.

| Les paramètres / Les couches | $a_{\max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 10 | 10 | 0.1 | 0.01 | 0.001 | 2500 |
| Deuxième couche | 10 | Nombre de neurones de la première couche | 0.01 | 0.01 | 0.001 | $\tau$*10 |

***Tab.4.13. Les paramètres utilisés dans le cas incrémental (logos).***

Le résultat de IGNGU est 21 classes, la figure Fig.4.40 montre le changement du nombre de neurones dans la première couche et la deuxième couche pour les deux parties.

Nous avons testé ce réseau par la même base de teste utilisée dans le cas passif, le taux de rejet est égal à 27.5%. Le taux de fausse classification est égal à 0.5%. Et le taux de bonne classification est égal à 72%.




***(a)*** ***(b)***

***Fig.4.40. Le nombre de neurones par rapport au temps dans les deux couches dans la classification de logos dans le cas incrémental, (a) la première couche, (b) la deuxième couche.***

## 2-4- La classification des cellules drépanocytose

La drépanocytose est une maladie génétique qui provoque un manque d'oxygénation du sang par la falciformation des globules rouges.

Cette grave maladie héréditaire touche en particulier les personnes originaires d'Afrique Noire, d'Afrique du Nord et des Antilles. La fréquence particulièrement élevée d'apparition de ce mal dans certaines de ces zones nécessite la mise en place de dépistages systématiques, dès la naissance. Plusieurs tests existent, des tests chimiques, des tests génétiques, mais aussi des tests visuels sur la qualité apparente des globules rouges observés au microscope. C'est ce dernier type de test qui nous intéresse ici. Il semble en effet que cette méthode, même si elle n'est pas suffisante à l'établissement d'un diagnostic définitif, soit la plus facilement généralisable, du fait du manque de moyens de bon nombre de zones infectées, et du faible coût en équipement que cette technique nécessite. De facto, vu la forme particulière des globules drépanocytaires, il semble intéressant de rendre automatique cette procédure de test, pour d'une part offrir un traitement plus rapide, donc permettant de traiter un plus grand nombre de patients, et d'autre part avoir un traitement objectif des clichés microscopiques, moyennant les informations préalables d'un expert (voir Fig.4.41).

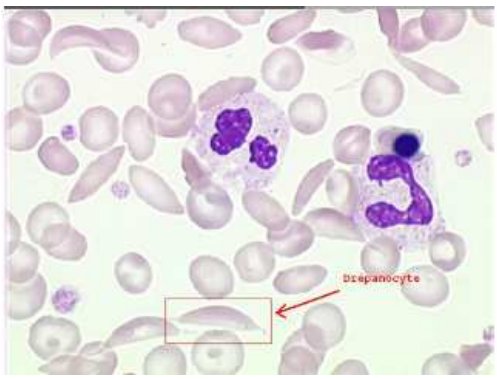



***Fig.4.41. Cliché de frottis sanguin d'un patient drépanocytaire.***

Les globules rouges prennent progressivement une forme de faucille, suivant les conditions dans lesquelles ils évoluent. Cette progressivité implique des formes de cellules intermédiaires (Fig.4.42), difficiles à classer par un médecin, à fortiori par une machine.

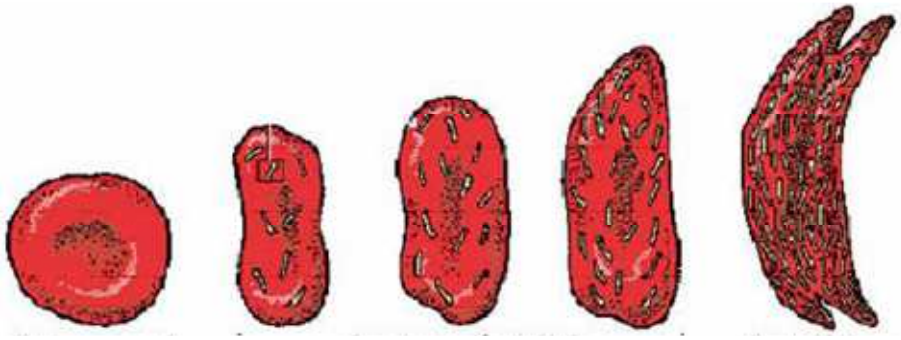


***Fig.4.42. Processus de falciformation d'un globule drépanocytaire.***

Rousseau dans [21] classe les cellules par un SVM en deux classes une classe pour les cellules malades et une autre classe pour les cellules saines, parce que les SVMs sont des classifieurs binaires. Mais le résultat de classifieur de Rousseau sur l'image de la figure Fig.4.43 est totalement erroné à cause des cellules de formes intermédiaires.

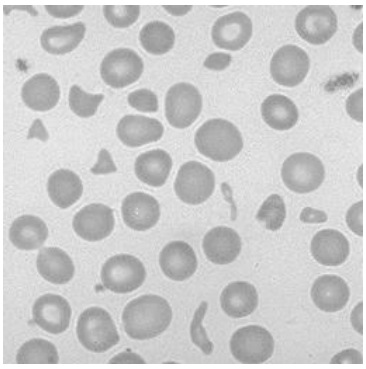


***Fig.4.43. L'image constituant la base d'apprentissage.***

Nous avons testé le réseau IGNGU pour classer les cellules de cette image (voir Fig.4.44). Cette dernière contient 35 cellules dont deux cellules drépanocytaires. Nous avons utilisé le même vecteur de caractéristiques utilisé par Rousseau [21]. Il contient 11 descripteurs, 7 sont les moments invariants et 4 descripteurs géométriques présentés ci dessous :

$$d_1 = \frac{4 * \pi * aire}{(perimetre)^2}$$

$$d_2 = \frac{aire\_du\_cercle\_inscrit}{aire\_du\_cercle\_circonscrit}$$

$$d_3 = \frac{(grand\_axe)^2}{aire}$$

$$d_4 = \frac{aire}{\left(\frac{grand\_axe}{2}\right)^2 * \pi}$$

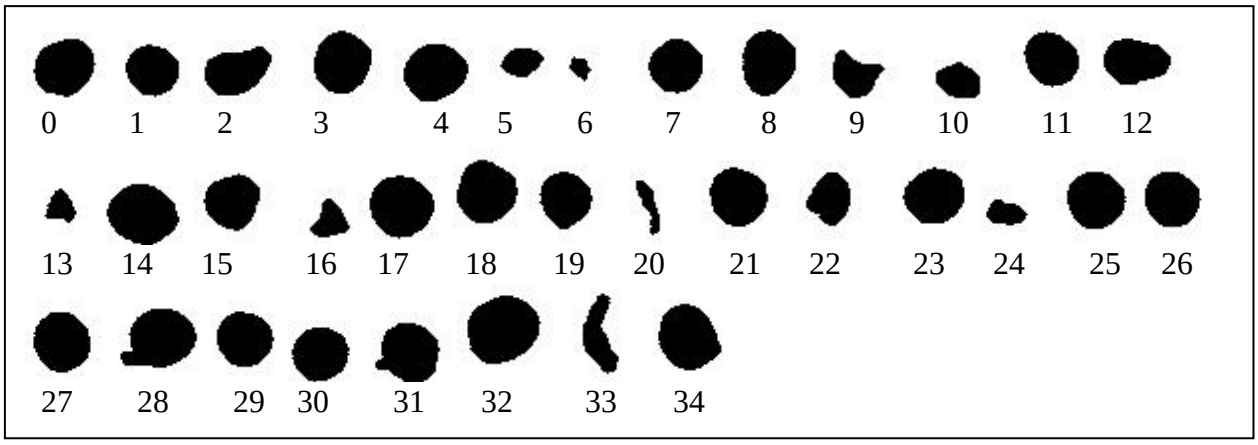
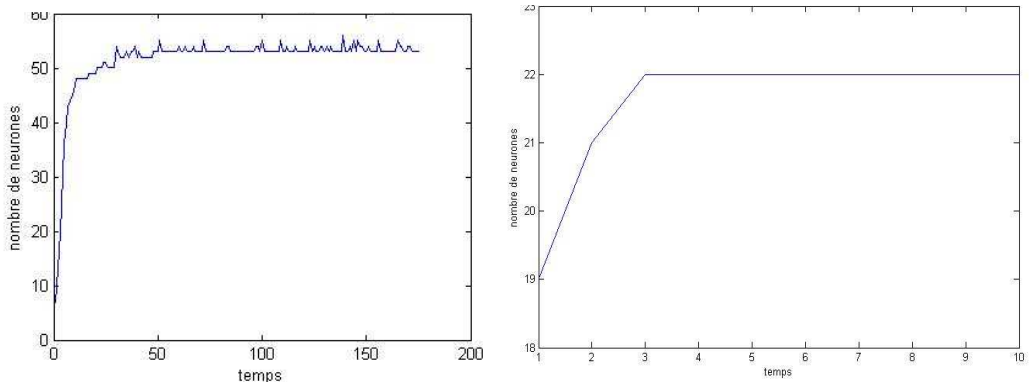



***Fig.4.44. Les cellules extraites à partir de l'image.***

Les paramètres du IGNGU utilisés :

| Les paramètres / Les couches | $a_{\max}$ | $\tau$ | $c$ | $\eta_g$ | $\eta_v$ | Nombre d'itérations |
|---|---|---|---|---|---|---|
| Première couche | 20 | 20 | 0.1 | *0.01* | *0.001* | 3 500 |
| Deuxième couche | 20 | Nombre de neurones de la première couche | 0.01 | *0.01* | *0.001* | $\tau$*10 |

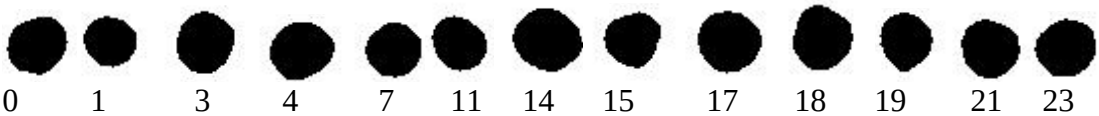
***Tab.4.14. Les paramètres utilisé dans la classification de cellules.***




***(a)*** ***(b)***

***Fig.4.45. Le nombre de neurones par rapport au temps dans les deux couches dans la classification de cellules drépanocytaires, (a) la première couche, (b) la deuxième couche.***

IGNGU détecte 7 classes montrées dans le tableau Tab.4.15. La classe 2 contient les cellules saines, la classe 7 contient les cellules malades, les classes 1, 3, 4 et 6 contiennent les cellules de formes intermédiaires. La cellule 28 est une erreur de classification, c'est une cellule saine, mais la segmentation ayant inclue à la forme de la cellule un élément extérieur (la partie grise en bas à gauche), le calcul des descripteurs a été faussé. La classe de rejet contient la cellule 33 qui est une cellule malade dans une étape plus avancé que la classe 7 c'est-à-dire la cellule 20.

| Les classes | Les cellules |
|---|---|
| 0 |  33 |
| 1 |  2 10 12 24 |
| 2 |  0 1 3 4 7 11 14 15 17 18 19 21 23  25 26 27 29 30 32 34 |
| 3 |  8 22 31 |
| 4 |  5 6 |
| 5 |  28 |
| 6 |  9 13 16 |
| 7 |  20 |

***Tab.4.15. Le résultat de la classification de cellules du IGNGU.***

# III- Conclusion

Nous avons appliqué le IGNGAT dans la segmentation des images de textures. Les résultats de la segmentation dépendent fortement de la valeur de la précision $P_r$. Les détails dans l'image segmentée apparaissent dans le même sens que la valeur donnée à $P_r$.

Nous avons testé le IGNGU sur une base de données artificielle, la segmentation des images de textures, la classification des logos et des cellules drépanocytaires.

Les résultats sur la base de données artificielle nous montrent que le IGNGU partitionne correctement l'espace d'entrée dans les cas passif et incrémental avec et sans bruit. Dans le cas incrémental le IGNGU est capable d'apprendre des nouvelles données sans réapprendre, oublier ou détériorer les données déjà apprises.

Dans la segmentation des images de textures, les caractéristiques utilisées sont extraites à partir de la matrice de cooccurrence. Plusieurs tailles de fenêtres d'apprentissage et une segmentation hiérarchique sont utilisées afin de ne pas détecter les détails dans les textures grossières et pour traiter le problème de frontières. Les résultats obtenus sont très encourageants pour tous les types d'images de textures. Pour les images de synthèse de quatre textures, de bons résultats ont été obtenu après l'utilisation d'une fenêtre de segmentation 35*35. Quant aux images de cinq textures, une fenêtre de segmentation 15*15 est suffisante pour obtenir des bons résultats. Dans ce type d'images la taille de la fenêtre d'affectation est égale à la taille de la fenêtre de segmentation, parce que les régions sont grossières. Mais dans les images ariennes et médicales, nous affectons l'étiquette de la classe au pixel centre parce que les régions sont fines. Dans les images ariennes, les champs, les arbres, les toits, les pavés et les routes sont détectés. Et dans les images médicales, la matière grise, la matière blanche et le liquide cérébro-spinal sont aussi détectés.

Dans la classification de logos, le IGNGU détecte les 20 classes. Le taux de bonne classification est acceptable, le taux de rejet est un peu élevé et le taux de fausse classification est faible, par ce que les caractéristiques utilisées sont insuffisantes.

IGNGU partitionne la base de cellules en 7 classes, une classe de cellules saines, une classe de cellules malades, et des classes pour les formes intermédiaires. IGNGU donne de meilleurs résultats que les SVMs dans la classification de cellules drépanocytaires. Il détecte les classes de formes intermédiaires.

Dans toutes les expériences, le nombre de neurones dans les deux couches du réseau se stabilise à une valeur ou dans un intervalle. Donc notre réseau ne souffre pas de dépassement du nombre de neurones.

# Conclusion générale

Dans le cadre de ce magistère nous avons proposé deux approches incrémentales pour l'apprentissage non supervisé d'un classifieur. Ce classifieur est un réseau de neurones constructif où l'ajout d'un neurone est effectué à l'arrivée d'une nouvelle donnée, s'il n'existe pas deux neurones ayant une distance euclidienne par rapport à celle-ci inférieure à un seuil de similarité. Ce dernier est adaptatif. Il est calculé pendant l'apprentissage de réseau.

Les approches proposées sont un réseau incrémental à seuillage adaptatif (IGNGAT) et un réseau de neurones incrémental à deux couches avec un paramètre d'utilité (IGNGU).

Dans le IGNGAT, le seuil est calculé par une formule à partir des données dans la base d'apprentissage et une valeur de précision. Nous l'avons testé pour la segmentation des images de textures de différents types. Les résultats obtenus sont dépendants de la valeur de la précision. Dès que la valeur de la précision augmente les détails dans les images de textures apparaissent.

Le IGNGU est un réseau incrémental à deux couches avec un paramètre d'utilité. Les deux couches sont des réseaux de neurones incrémentaux capables de partitionner un espace d'entrée bruité par l'utilisation du paramètre d'utilité. Celui-ci est utilisé pour éviter le dépassement du nombre de neurones dans la première couche. Dans le cas où la base est apprise en plusieurs parties, les deux couches peuvent opérer en parallèle. Le seuil de similarité de la première couche est calculé pendant l'apprentissage. Mais dans la deuxième couche, le seuil est calculé à partir des neurones générés dans la première couche ; sa valeur reste fixe pendant l'apprentissage.

Dans les deux approches proposées, il n'existe pas de méthode pour calculer les paramètres du réseau $\tau$ et $a_{\max}$. Ces valeurs sont déterminées expérimentalement. Donc nos futures recherches s'orienteront vers l'optimisation de ces paramètres. Bien que les résultats des approches proposées soient bons par l'utilisation de l'apprentissage non supervisé, nous pensons que les performances des réseaux augmentent si nous utilisons un apprentissage hybride. Aussi nos futures études concerneront la construction d'un système d'apprentissage incrémental distribué et parallèle.

[1] P. Henniges, *PSO pour l'apprentissage supervisé des réseaux neuronaux de type fuzzy artmap*. Thèse de l'école de technologie supérieure université du Québec, Montréal, Le 25 Juillet 2006.

[2] G. Dreyfus et al., *Réseaux de neurones, méthode et applications.* 2 éditions, EYROLLES, 2004.

[3] T. Kohonen, *Self-Organization and Associative Memory*. Springer-Verlag, New York, 1988.

[4] B. Fritzke, "A growing neural gas network learns topologies". NIPS, pp 625–632, 1995.

[5] N. Benahmed, *Optimisation de réseaux de neurones pour la reconnaissance de chiffres manuscrits isolés : sélection et pondération des primitives par algorithmes génétiques.* Thèse de l'école de technologie supérieure université du Québec, Montréal, le 1 Mars 2002.

[6] M. Kunt et al., *Reconnaissance des formes et analyse de scènes.* Traitement de l'information : volume 3, presses polytechniques et universitaires romandes, 2000.

[7] L. Miclet, *Méthode structurelle pour la reconnaissance des formes*. EYROLLES, collection technique et scientifique de télécommunication, 1984.

[8] C. K. Bounsaythip, *Des algorithmes heuristiques et évolutionnistes. Application à la résolution du problème de placement de formes irrégulières*. Thèse de l'université des sciences et technologies de Lille, 9 Octobre1998.

[9] F. Lafarge, X. Descombes, J. Zerubia, *Noyaux texturaux pour les problèmes de classification par SVM en télédétection*. Rapport INRIA, N° 5370, Novembre 2004.

[10] E. Viennet, "Machines à vecteurs de support et réseaux de neurones", Université Paris 13, Laboratoire de d'Informatique de Paris Nord, Mai 1999.

[11] V. Vapnik, *The nature of statistical learning theory*. Springer, New York, 1995.

[12] Y. Lecourtier et al., "Réseaux d'Yprels, classification et apprentissage incrémental", université de Rouen, Laboratoire Images et Informatique Industrielle, Traitement du Signal 1995.

[13] A. K. Jain et R. C. Dubes, *Algorithms for clustering data*, Prentice-Hall advanced reference series, Prentice Hall, Inc., Upper Saddle River, NJ, 1988.

[14] B. Fritzke. "A self organizing network that can follow non stationary distributions", Proceedings of ICANN-97, pp 613–618, 1997.

[15] Y. Prudent et A. Ennaji. "Extraction incrémentale de la topologie des données". Laboratoire PSI, Université et INSA de Rouen .F-76821, France, 2000.

[16] S. Furao et O. Hasegaw. "An incremental network for on-line unsupervised classification and topology learning". R2-52, 4259 Nagatsuta, Midori-ku, Yokohama 226-8503, Japan, April 2005.

[17] M. N. Kurnaz et al., "An incremental neural network for tissue segmentation in ultrasound images", Comput, Methods Programs Biomed, 2006.

[18] Z. İşcan, et al., “Ultrasound image segmentation by using wavelet transform and self-organizing”, Neural Network, 2006.

[19] P. Brodatz, *Photographic album for artists and designers*, Dover publication, New-York 1966.

[20] R. M. Haralick, “Statistical and structural approaches to textures”. Processing of IEEE, vol 67, N5, pp 786-804, 1979.

[21] F. Rousseau, *Description et classification de cellules drépanocytaires*. Mémoire de Master, Université d'Evry Val d'Essonne, 2007.