République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique
Université 8Mai 1945 – Guelma
Faculté des sciences et de la Technologie
Département d'Electronique et Télécommunications

Mémoire de fin d'étude
pour l'obtention du diplôme de Master Académique

Domaine : **Sciences et Technologie**
Filière : **Electronique**
Spécialité : **Systèmes Electroniques**

La reconnaissance de forme par réseaux des neurones

Présenté par :

Mr SEHILI Nabil

Sous la direction de :

Mr MENASRIA Azzeddine

Mai 2014

Dédicaces

A mes parents,

A ma chère sœur,
A Merieme,Remaissa et Nawel
A mes frères,

A mes chers collègues,

A mes chers amis.

Remerciements

Nos vifs remerciements à

Monsieur MENASRIA Azzeddine ,

pour l'intérêt porté à ce travail, pour ses

encouragements.

Dédicaces

Sommaire

Problématique de la reconnaissance d'image 2

Introduction générale.....5

Chapitre I

RECONNAISSANCE DES FORMES 7

I.1. HISTORIQUE.....8

I.2. DEFINITIONS.....8

I.3. LE BUT DE LA RECONNAISSANCE DES FORMES.....9

I.4. GENERALITES.....11

I.4.1. Le monde physique.....11

I.4.2. Le codage.....12

I.4.3. Le prétraitement.....12

I.4.5. L'apprentissage.....15

I.6. LES DIFFERENTES METHODES DE LA RECONNAISSANCE DES FORMES.....16

I.6.1. L'approche statistique.....16

I.6.1.1. Méthodes paramétriques.....17

I.6.1.1.1. Méthode de Bayes.....17

I.6.1.2. Méthodes non paramétriques.....18

I.6.1.2.1 Méthode des k plus proches voisins.....18

I.6.1.2.2. Méthode des nuées dynamiques.....19

I.6.1.3. Conclusion 20

Chapitre II

LES RESEAUX DE NEURONES.....22

II.1. INTRODUCTION.....22

II.2. HISTORIQUE 23

II.3. Neurone biologique.....24

II.5. DIFFERENTS MODELES DE RESEAUX.....26

II.5.1. L'Adaline 26

II.5.2.Les réseaux de Hopfield.....26

II.5.3. Les réseaux de Kohonen.....27
II.5.4. Les perceptrons multicouches.....28
II.6. Le neurone formel généralité 28
II.6.3. Perceptron une couche 43
II.6.3. 1.Description mathématique 43
II.6.3. 2Algorithme du perceptron une couche 45
II.6.3. 3 Limitations et propriétés fondamentales.....46
II.6.4. Perceptron multicouches 46
II.6.4. 1.Description Mathématique (Algorithme de rétro-propagation)..... 47
II.6.4. 1.1.Vers la couche de sortie 48
II.5.4. 1.2.Vers une couche cachée..... 49
II.5.4. 1.3.Formule générale.....51
II.5.4. 1.4Algorithme du perceptron multicouche.....52
II.5.4. 1.4.Conclusion 53

Chapitre III

APPLICATION.....55
III.1. INTRODUCTION.....56
III.2. Définition image.....56
III.3. Prétraitement 56
III.3.1. Image en niveaux de gris.....57
III.3.2. Image binaire.....57
III.3.3. Opérations géométriques.....58
III.3.4. Détection de contours.....60
III.3.4. 1.le filtre de Sobel 60
III.3.4. 2.Histogramme – seuillage.....62
III.4. L'extraction des propriétés 62
III.5. L'apprentissage.....63
III.5. 1. Algorithme du perceptron multicouche.....64
III.6. Algorithme du classification.....65
III.7. les applications.....66
III.7.1. test et décision.....68
Conclusion.....76

Bibliographie ……..77

LISTE DES FIGURE

Chapitre I

Figure I.1. Procédure de capture de forme.....10
Figure I.2. Système de reconnaissance des formes..... 11
Figure I.3.des images obtenue par caméra.....11
Figure I.5Extraction propriétés.....15
Figure I.6. Exemple d'apprentissage.....15
Figure I.7. Les différentes méthodes de reconnaissance des formes.....16
Figure I.8. Méthode de décision du plus proche voisin.....18
Figure I.9. Méthode de k plus proches voisins.....19

Chapitre II

Figure II.1. Schéma du neurone biologique.....25
Figure II. Perceptron multicouches.....28
Figure II.2Neurone formel à plusieurs entrées avec seuil de comparaison β.....29
Figure II.3 Neurone formel binaire de McCulloch-Pitts.....30
Figure II.4 OU logique : table de vérité et valeur des poids du neurone formel. 30
Figure II.5 ET logique : table de vérité et valeur des poids du neurone Formel.....31
Figure II.6 Problème de séparabilité pour le XOR logique.....32
Figure II.7 Classes non séparables pour un perceptron.....33
Figure II.8 Réseau MLP à connections directes.....34
Figure II.9 Séparation linéaire des classes.....35
Figure II.10 Séparation de deux classes par une droite.....36
Figure II.11 Zones indéterminées (Zind) d'un classifieur linéaire.....37
Figure II.12 Droite séparatrice avant apprentissage. 39
Figure II.13 Droite séparatrice avant et après apprentissage.....39
Figure II.14 Fonction d'activation à seuil binaire 40
Figure II.15 Fonction d'activation à seuil binaire.....41

Figure II.16 Fonction logsig……..42
Figure II.17 : Identification entre neurone biologique et neurone formel……..42
La figure II.18 notations……42

Chapitre III

Figure III.1 représentation d'une image par une matrice……..56
Figure III.2 le codage par niveau de gris ……....57
Figure III.3 Image binaire……..57
Figure III.4 le redimensionnement d'image……..58
Figure III.5 Centrage……...58
Figure III.5 découpage……..60
Figure III.6 détection de contour (filtre de Sobel) ……61
Figure III.7 propriétés d'image……..62
Figure III.8 extraire les diagonales d'image……..63
Figure III.9 courbe de minimisation d'erreur triangle ……..66
Figure III.10 courbe de minimisation d'erreur rectangle……...66
Figure III.11 courbe de minimisation d'erreur cercle……67
Figure III.12 Image inconnu avant classification……...68
Figure III.13Image=triangle Après classification……...68
Figure III.14 Image inconnu avant classification……69
Figure III.15Image=rectangle Après classification……69
Figure III.16 Image inconnu avant classification……...70
Figure III.17Image=cercle Après classification……70
Figure III.18 .Image réel……71
Figure III.19 . Prétraitement (filtrage) + extraction des caractéristiques……...71
Figure III.20 Segmentation (découpage)……72
Figure III.21. Inversion de couleur……72
Figure III.22. Séparation entre les chiffres ……72
Figure III.23. Test avant classification……73
Figure III.24. Test après classification……73
Tableau III.1 données utiliser pour l'application……...74

Tableau III.2 Résultat pour l'application 01……74

Tableau III.2 Résultat pour l'application 02……75

Problématique de la reconnaissance d'image

Le problème que cherche à résoudre la reconnaissance des formes est d'associer une étiquette à une donnée qui peut se présenter sous forme d'une image ou d'un signal. Des données différentes peuvent recevoir la même étiquette, ces données sont les réalisations ou les exemplaires de la classe identifiée par l'étiquette. Par exemple, le son Q prononcé par différents locuteurs conduit à des signaux différents mais ces différences ne sont pas significatives du point de vue de l'identification du son, ces signaux sont des réalisations ou des représentations de la classe Q. De même, l'écriture manuscrite du caractère A varie d'un scripteur à l'autre mais le lecteur identifiera le caractère A pour chacune de ces réalisations.

Des méthodes générales ont été développées en reconnaissance des formes pour extraire automatiquement des informations des données sensibles afin de caractériser les classes de formes (apprentissage) et d'assigner automatiquement des données à ces classes (reconnaissance).

Parallèlement aux travaux sur les méthodes de reconnaissance, se développaient le traitement d'image, la vision par ordinateur, et le traitement de la parole. Ces domaines ont focalisé le problème de la reconnaissance sur des données spécifiques, mais par ailleurs ils ont permis de situer la reconnaissance dans un processus plus vaste d'interprétation d'images ou de compréhension de la parole.

Introduction générale

Introduction générale

La reconnaissance des formes a pour objet de simuler les activités de perception sensorielle du cerveau. En premier lieu, la perception visuelle et auditive, y compris dans des bandes spectrales non perçues par l'homme (infra rouge, radar, sonar, ...). La reconnaissance a besoin d'un modèle de l'objet. Pour l'être humain, ce modèle correspond à une représentation mentale de l'objet qui peut être apprise en retenant les caractéristiques les plus discriminantes de l'objet. Les caractéristiques peuvent être toutes sortes d'attributs de l'objet : forme, couleur, texture, taille, volume, etc.
Objet de recherches depuis l'apparition de l'informatique, cette discipline prend désormais un essor considérable dans l'industrie, car les progrès de l'électronique permettent désormais de doter les systèmes opérationnels de fortes puissances de calcul à des prix raisonnables. Elle regroupe un certain nombre de techniques allant de la simple identification d'objets isolés à l'analyse de scènes complexes ou la compréhension de la parole. Dans ces derniers cas, il faut tout à la fois segmenter en objets, identifier ces objets et établir un réseau de relations entre objets. La notion d'apprentissage est au coeur de la plupart des techniques développées. Ces techniques peuvent être utilisées soit seules dans le cas de systèmes dont la mission principale est d'identifier des objets, soit en relation étroite avec des techniques d'intelligence artificielle dans le cas de systèmes devant à la fois percevoir et raisonner sur les choses perçues.

La reconnaissance de formes est un domaine très proche de celui de l'analyse de données (utilisé pour la prévision, les études statistiques, etc.) : la reconnaissance passe souvent par une analyse de données préalable, d'une subdivision de ces données en groupes d'appartenance distincts, puis d'une prise de décision d'appartenance à l'un de ces groupes.

C'est un domaine également très proche de celui des réseaux de neurones: d'une part la reconnaissance de formes est la principale application des réseaux de neurones, et d'autre part de nombreuses méthodes de reconnaissance de formes
peuvent être formulées dans le cadre des réseaux de neurones.

Comprendre les mécanismes à l'origine des fonctions supérieures du cerveau est l'objet de recherches au carrefour de la neurobiologie, de la psychologie, de l'informatique et de la physique.

Les neurones sont des automates élémentaires dont le mode de fonctionnement s'inspire de celui des neurones biologiques. Il y a deux motivations principales : d'une part la modélisation du cerveau, et d'autre part la réalisation d'algorithmes et de machines spécialisées dans des tâches de reconnaissance des formes par exemple, dont les performances pourraient être supérieures à celles des algorithmes et ordinateurs classiques. Malgré les succès de l'application des réseaux de neurones à la reconnaissance des formes statiques, le traitement ou la reconnaissance des processus non stationnaires avec les réseaux de neurones est encore un grand défi.

Aujourd'hui les réseaux de neurones sont utilisés dans de nombreux domaines, à commencer par la vie artificielle et l'intelligence artificielle eût égard à leurs nombreuses propriétés, leur capacité d'apprentissage en particulier.

Notre travail a pour objectif de mettre au point un système de reconnaissance automatique de formes géométriques. La solution proposée est basée sur l'approche neuronale.

Donc, notre travail est de concevoir et d'expérimenter une architecture de réseaux de neurones dont le but de développé un système de reconnaissance automatique de formes.

Chapitre I

Chapitre I

Reconnaissance des formes

I.1. HISTORIQUE

La reconnaissance des formes est apparue au milieu des années cinquante à partir des mathématiques (calcul des probabilités, algèbre linéaire…), de l'informatique et des sciences de l'ingénieur. Elle s'intéresse à la conception et à la réalisation des systèmes (matériels ou logiciels) capables de percevoir et, d'interpréter, dans une certaine mesure, des signaux captés depuis le monde physique. Ultérieurement, son développement a été lié à l'intelligence artificielle dont on l'a considérée pendant longtemps comme une composante.

Les problèmes de détection classiques en traitement de signal on servi de bases pour le développement de nombreuses méthodes qui, par la suite, sont intégrées et généralisées pour d'autres données que des signaux.

Les premiers travaux que l'on peut qualifier de la reconnaissance des formes au sens actuel ont lieu à partir du début des années soixante. Dans ces travaux, on retrouve les deux disciplines fondatrices : mathématique (pour le cadre théorique, pour l'analyse et la généralisation des méthodes) et l'informatique (pour la construction et l'implémentation logiciel ou même matériel des algorithmes, pour leur parallélisassions).

I.2. DEFINITIONS

On désigne par reconnaissance de formes (ou parfois reconnaissance de motifs) un ensemble de techniques et méthodes visant à identifier des motifs à partir de données brutes afin de prendre une décision dépendant de la catégorie attribuée à ce motif. On considère que c'est une branche de l'intelligence artificielle qui fait largement appel aux techniques d'apprentissage automatique et aux statistiques.

Les formes ou motifs à reconnaître peuvent être de nature très variée. Il peut s'agir de contenu visuel (code barre, visage, empreinte digitale...) ou sonore (reconnaissance de parole), d'images médicales (rayon X, EEG, IRM...) ou multi spectrales (images satellitaires).

La reconnaissance des formes consiste à affecter une forme à la classe à laquelle elle appartient. Elle est définie comme étant l'ensemble des techniques qui permettent de

reproduire la perception humaine en vue d'automatiser complètement les opérations et les prises de décision.

Mathématiquement la reconnaissance des formes est un problème formaliser :

Soit X un espace de représentation et Ω un ensemble fini de noms (espace d'interprétation), une reconnaissance (identification) est une application

$\beta : X \longrightarrow \Omega$

Le problème est de construire β, qui à partir de $x \in X$, permet de décider à quelle classe $y \in \Omega$ l'élément x est affecté .

La reconnaissance des formes consiste donc à doter la machine d'organes comme : le scanner, la camera et d'autres moyens captant l'information extérieure sous formes variées pour l'utilisation dans les étapes ultérieures.

Après les définitions précédentes on voit les difficultés suivantes :

- Saisir l'information.
- La représenter convenablement.
- Prendre une décision d'interprétation dans cette espace de représentation.

I.3. LE BUT DE LA RECONNAISSANCE DES FORMES

Le but de la reconnaissance des formes est : «d'obtenir des algorithmes permettant de reproduire efficacement les phénomènes de perception de l'être humain » .La reconnaissance des formes est un domaine de l'informatique, qui a pour but de percevoir et de reconnaitre automatiquement les formes présentes dans les signaux et les images. Il s'agit de mettre au point des programmes et des logiciels capables de traiter la multitude des cas concrets : repérage d'objets plans, reconnaissance de défauts, reconnaissance de caractère… etc.

I.4. GENERALITES

La reconnaissance de formes possède de nombreux domaines d'application qui vont de la reconnaissance de caractères manuscrits, à la reconnaissance automatique de la parole ou encore à la reconnaissance de forme géométrique dans une image. Une des approches la plus généralement adoptée, consiste dans la définition initiale d'un « univers » d'intérêt puis dans le choix d'un descripteur sur un espace vectoriel. On a ensuite le choix soit de définir

intrinsèquement les objets de cet univers d'intérêt soit de développer un automate qui établira des classes d'équivalence par présentation d'exemples ou d'échantillons qui lui seront désignés ou non.

Quand on estime que la phase d'apprentissage est terminée, le même algorithme ou une forme modifiée sera utilisée dans une phase de reconnaissance de forme proprement dite. La reconnaissance consistant soit à reconnaître un des exemples d'apprentissage plus ou moins déformé soit plus généralement à
définir la classe d'appartenance la plus proche d'un échantillon inconnu.
On peut définir un schéma général mettant en œuvre un système de capteurs sur le monde physique permettant à un ordinateur d'acquérir des données sur l'univers physique désiré puis après un prétraitement, de disposer d'un descripteur numérique.

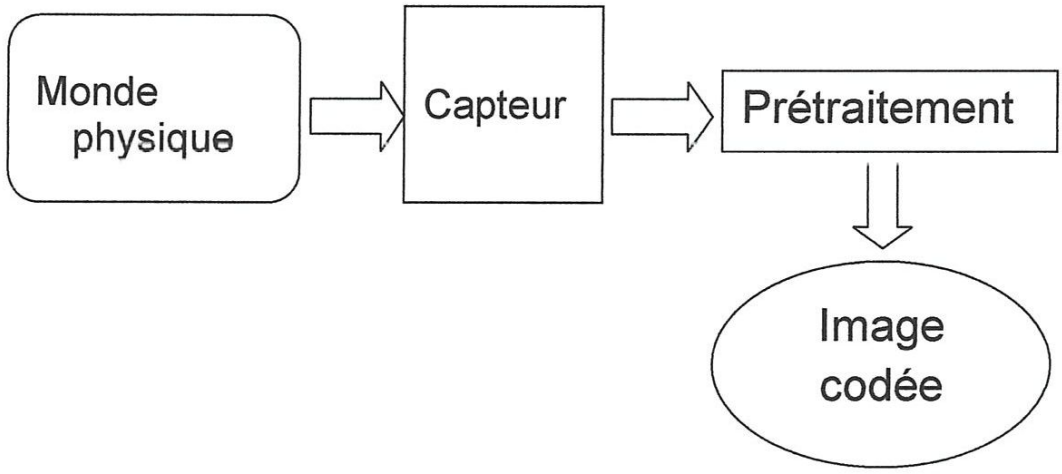


Figure I.1. Procédure de capture de forme

Le schéma général d'un système de reconnaissance des formes avec une phase d'apprentissage est donné par la figure suivante :

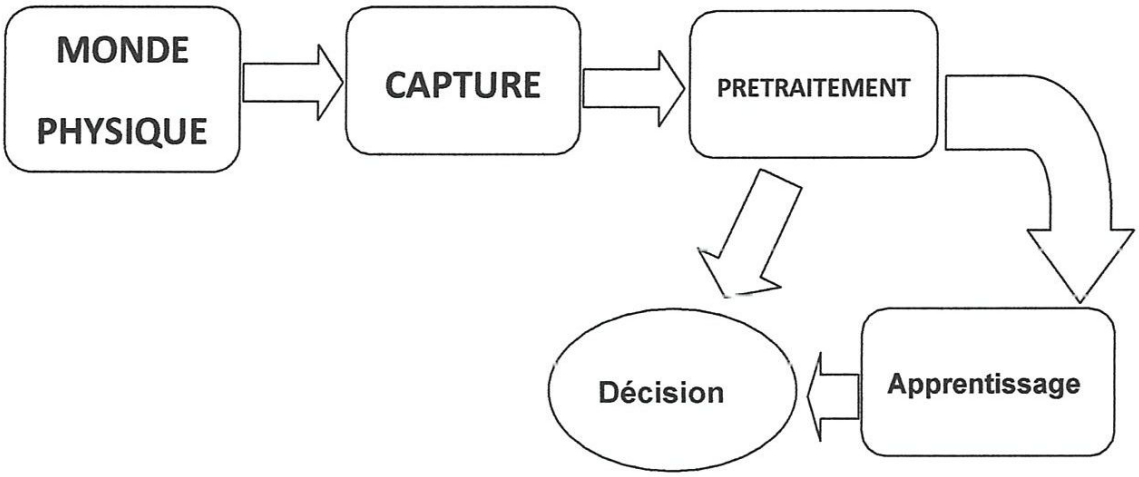


Figure I.2. Système de reconnaissance des formes

I.4.1. Le monde physique

La chaîne part du monde physique qui est un espace analogique de dimension infinie appelé espace des formes. Les objets dans cet espace sont décrits de différentes façons avec une multitude de propriétés dont il serait difficile de tenir compte à la reconnaissance des formes. La loi de passage au monde numérique nécessite forcément une sélection et par conséquent une certaine simplification.

Figure I.3.Des images obtenue par caméra

I.4.2. Le codage

L'opération de codage consiste à transformer un ensemble de données analogiques en un ensemble de données numériques de manière à pouvoir les traiter par ordinateur. Cette transformation doit se faire de la manière la plus fidèle possible, c'est-à-dire sans perte d'informations pertinentes et en conservant les propriétés essentielles de l'objet physique. En vision, on doit être capable de retrouver la même disposition des objets de la scène .

Cette étape est inclut dans l'appareil de capture (la conversion analogique numérique intégré dans l'appareil de capture).

I.4.3. Le prétraitement

L'information brute de l'opération d'acquisition peut engendrer des bruits dus au dispositif de capture lui-même, ce qui altère la qualité de l'image. Le prétraitement tente de remédier à cet inconvénient en filtrant la forme saisie. Les techniques employées dépendent totalement du mode d'acquisition, du type d'image et des informations que l'on veut retirer.

Représantation par le niveau de gris ou binaire

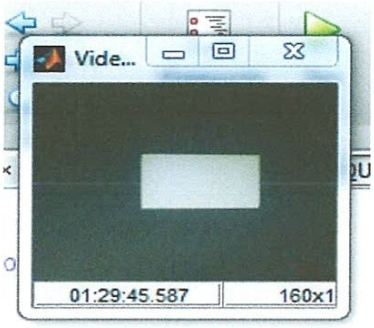
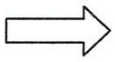


image réel



RECT3 =

102	102	101	102	102	102	102	102	102	102
102	101	102	102	101	102	102	101	102	102
102	102	101	102	102	101	102	102	102	102
102	103	104	107	108	107	106	105	103	103
103	102	115	135	136	132	128	126	109	103
104	104	119	140	141	138	135	133	112	103
107	106	122	142	141	141	139	137	115	104
108	107	123	142	141	140	138	135	114	104
109	110	113	116	115	114	112	109	107	105
111	110	110	110	110	109	108	106	106	106

niveau de gris

RECT2 =

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Image binaire= algorithme seuil (image niveau de gris)

Figure I.4. Type de représentation image après prétraitement

I.4.3.1. Le filtrage

Le filtrage est l'élimination des informations non utile donc minimisation d'erreur dans la phase apprentissage et classification, pour notre application on utilise le filtre de Sobel pour détection de contour.

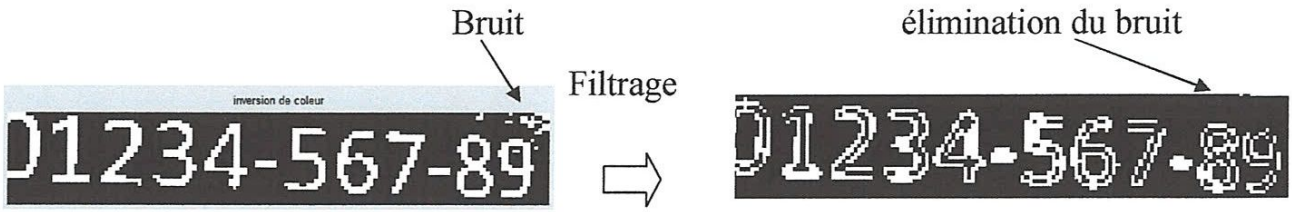


Figure I.5. Filtrage

I.4.4. L'extraction des propriétés

Le but de l'analyse est d'extraire les propriétés caractéristiques de l'objet et de les exprimer sous forme numérique ou symbolique. La représentation obtenue servira de base aux étapes ultérieures d'apprentissage et de reconnaissance.

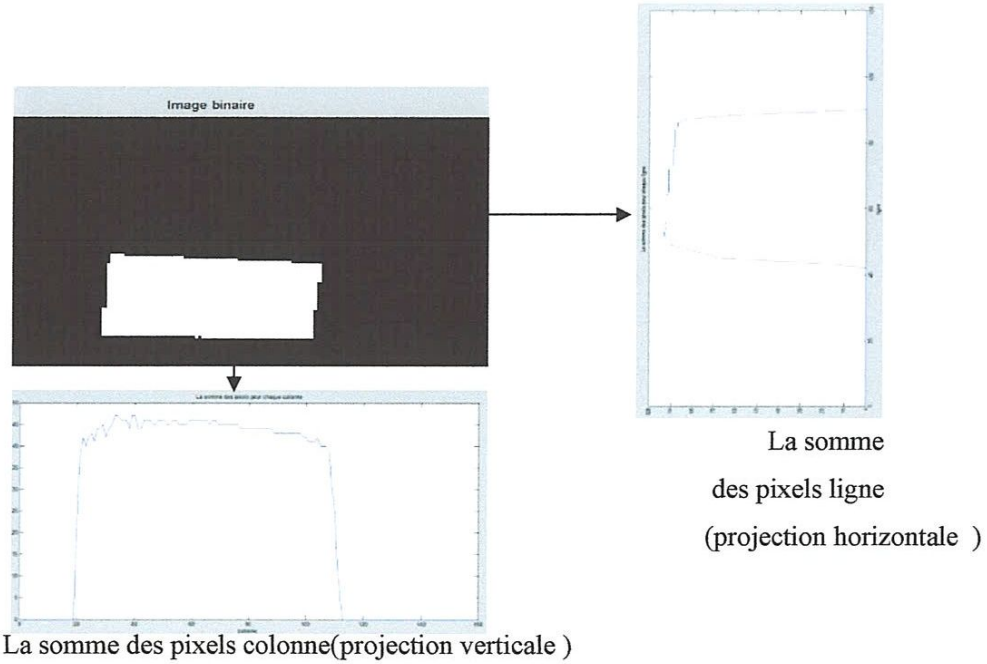
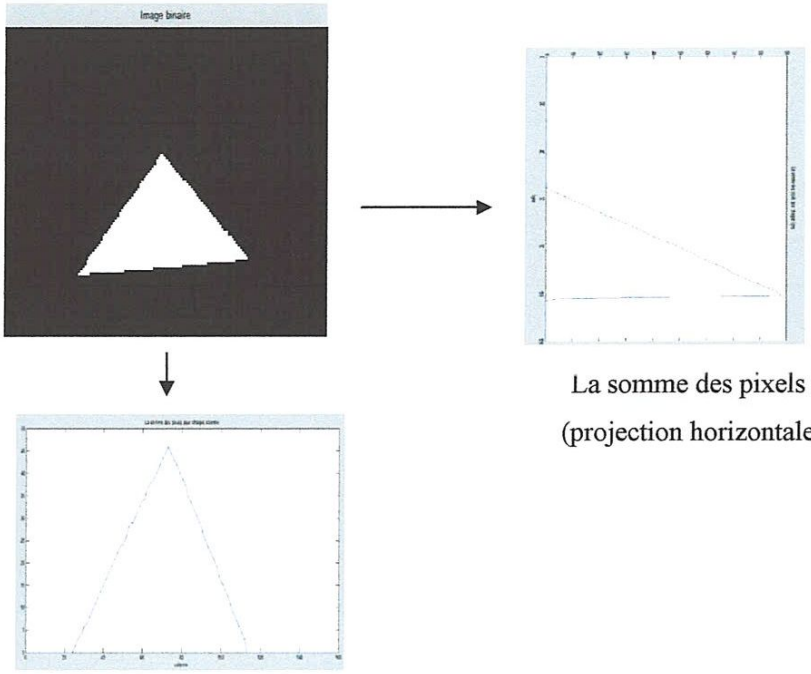


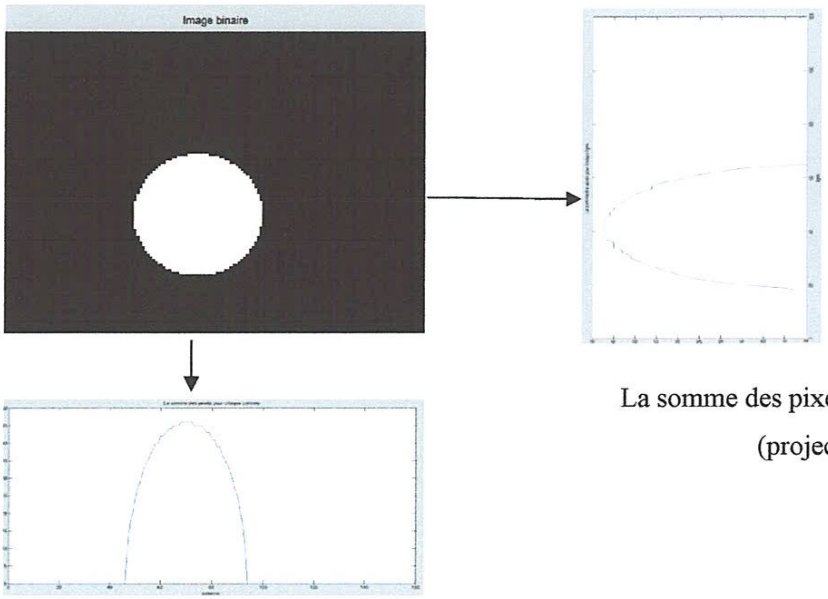
Figure I.6.a.Extraction les propriétés pour un rectangle (caractéristiques)



La somme des pixels ligne
(projection horizontale)

La somme des pixels colonne (projection verticale)

Figure I.6.b.Extraction les propriétés pour un triangle (caractéristiques)



La somme des pixels ligne
(projection horizontale)

La somme des pixels colonne(projection verticale)

Figure I.6.c.Extraction les propriétés pour un cercle (caractéristiques)

Donc l'image et représenter par deux vecteur seulement vecteur ligne et vecteur colonne ce qu'il permette de minimiser le temps de traitement, apprentissage et reconnaissance.

I.4.5. L'apprentissage

Dans le cas d'apprentissage il s'agit de fournir au système un ensemble de formes dont on connaît les classes. C'est cet ensemble d'apprentissage qui va permettre de régler le système de reconnaissance de façon à ce qu'il soit capable de reconnaître de nouvelles formes.

Les paramètres d'un classifieur sont ajustés à partir de données d'apprentissage, sans regarder si la réponse du classifieur est correcte ou non pendant cette modification. Dans le cas supervisé, au contraire, les erreurs commises par le classifieur sont exploitées dans l'ajustement des paramètres.

Dans la plupart des algorithmes, le nombre de classes des données est initialement indiqué à l'algorithme.

En fait le rôle du module d'apprentissage consiste à caractériser chaque classe par exemple par des relations entre les paramètres définissant la forme. Par exemple, supposons que des formes soient définies par des vecteurs dans R et supposons disposer d'échantillons de deux classes d'objets. On peut alors représenter ces échantillons dans l'espace R des paramètres.

L'apprentissage peut alors par exemple consister à trouver automatiquement une courbe séparant les échantillons de chacune des deux classes. C'est la recherche de cette courbe qui va être l'apprentissage des deux classes.

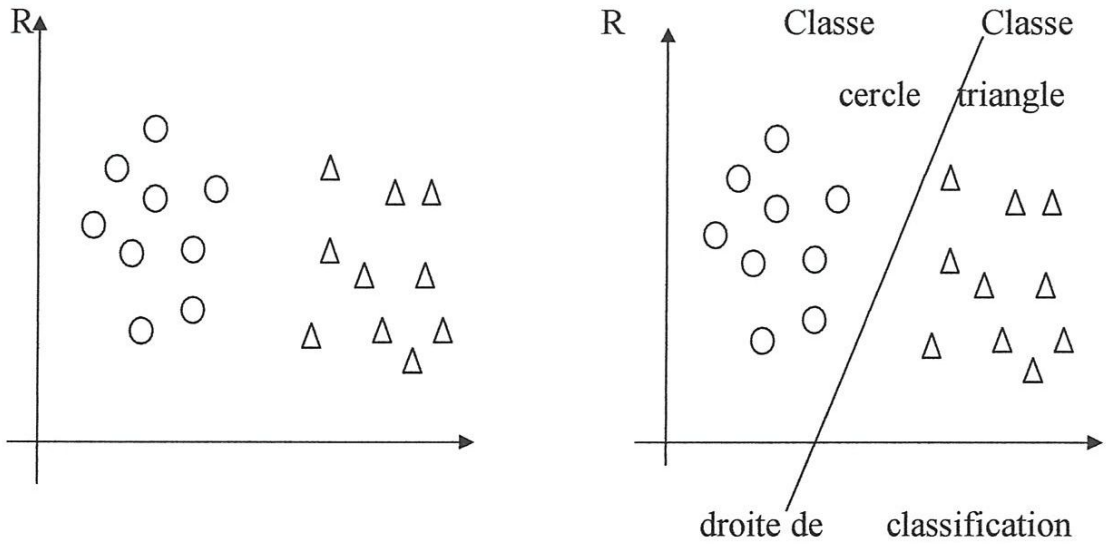


Figure I.6. Exemple d'apprentissage

I.6. LES DIFFERENTES METHODES DE LA RECONNAISSANCE DES FORMES

Généralement, il y a trois approches : l'approche statistique basée sur des calculs de probabilités et statistiques, l'approche linguistique basée sur les formes graphiques des objets et l'approche connexionniste basée sur les réseaux de neurones. Ces méthodes son résumées sur le schéma suivant :

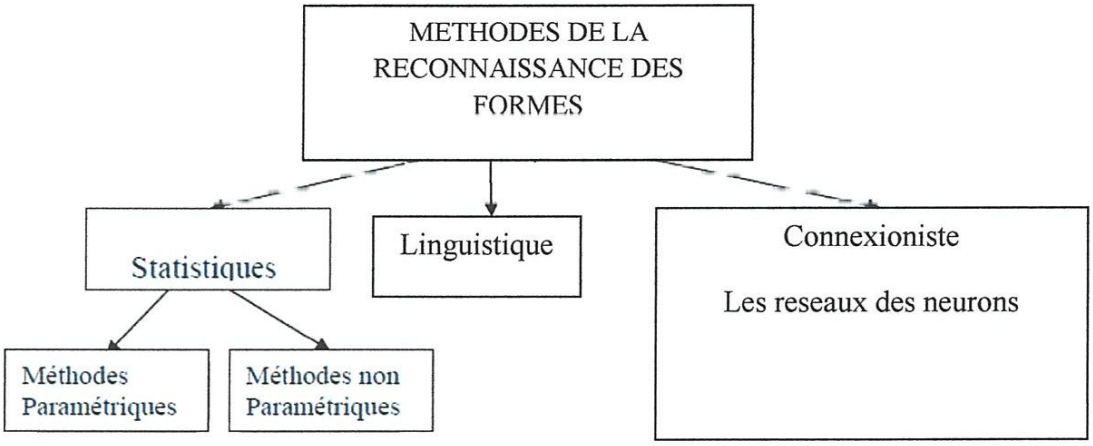


Figure I.7. Les différentes méthodes de reconnaissance des formes

I.6.1. L'approche statistique

L'application des méthodes statistiques à la reconnaissance des formes a été formalisée par Chow en 1965 Plusieurs systèmes existants s'inspirent de ces méthodes.

Elle consiste à reconnaître une forme en calculant certaines caractéristiques statistiques qui décrivent la forme, elle est la plus ancienne et parmi les méthodes les plus utilisées en reconnaissance des formes.

Dans cette approche, l'espace d'interprétation est constitué de classes nommées où chaque classe regroupe plusieurs vecteurs ayant des caractéristiques communes et ayant la même interprétation.

Par exemple : un caractère qui s'écrit de plusieurs façons. On représente chaque façon d'écriture par un vecteur, l'ensemble des vecteurs obtenus définit une classe, l'ensemble des classes obtenues construit le dictionnaire d'apprentissage ou l'ensemble des formes déréférence.

La forme inconnue est représentée par un vecteur de dimension n, en effectuant n mesures sur la forme à reconnaître; à la décision nous distinguons d'une façon générale deux grands types de méthodes :

1) Les méthodes dites paramétriques où l'on donne un modèle de distribution de chaque classe avec la probabilité d'appartenance d'un point donné à une classe.

2) Les méthodes dites non paramétriques où l'on cherche à définir les frontières des classes dans l'espace de représentation.

I.6.1.1. Méthodes paramétriques

Le but de ces méthodes est de minimiser l'erreur moyenne d'une mauvaise classification, elles essaient d'éliminer le risque de décider l'appartenance d'une forme inconnue à une classe ne lui correspondant pas.

Les méthodes paramétriques supposent la connaissance des lois de probabilité des observations et des classes.

Chaque classe est décrite par un modèle de distribution (loi de probabilité), en général gaussienne. Ce type de méthodes donne de bons résultats à condition que la répartition des points dans chaque classe soit régulière pour qu'on puisse trouver une loi de probabilité connue, et que le nombre d'échantillons soit important pour avoir une bonne précision des calculs.

I.6.1.1. Méthode de Bayes

Dans la classification bayesienne, les classes et les données à classer sont considérées comme des variables aléatoires.

Les classes sont en nombre fini. Ce sont donc des discrètes et elles sont décrites par des probabilités.

Les données sont des variables aléatoires continues, donc elles sont décrites par des fonctions de densité de probabilité.

Dans le cas où nous aurions toutes les informations en termes de probabilités a priori sur les objets, nous pouvons utiliser le théorème de Bayes qui permet de transformer ces probabilités a priori en probabilités a postériori, c'est-à-dire à trouver la classe la plus probable.

I.6.1.2. Méthodes non paramétriques

Le but de ces méthodes est la définition des frontières des classes dans l'espace de représentation de façon à classer le point inconnu par une série de tests simples.

I.6.1.2.1 Méthode des k plus proches voisins

La forme à classer est comparée aux autres formes ayant déjà été classées, et on lui affecte la classe la plus représentée parmi les k plus proches d'elle.

Dans le cas particulier k=1, c'est la classe de la forme la plus proche de la forme à classer qui est affectée à cette dernière.

La notion de proximité utilisée est quantifiée par une mesure de similarité. La mesure de similarité la plus utilisée est la distance euclidienne.

Pour que l'algorithme puisse démarrer, il faut utiliser un certain nombre d'exemples dont la classe est connue, auxquels vont être comparés les premiers vecteurs à classer.

- Règle de décision du plus proche voisin

Cette règle suppose disposer d'une distance dans l'espace des formes, et on est donc capable de calculer une distance entre une forme x quelconque à un échantillon.

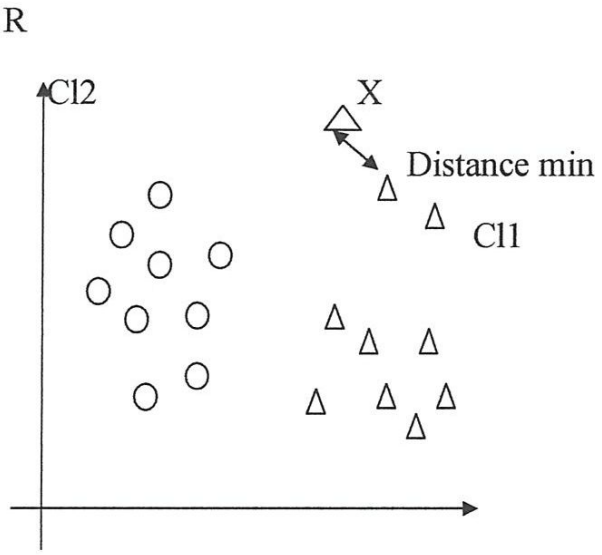


Figure I.8. Méthode de décision du plus proche voisin

Le principe de décision consiste tout simplement à calculer la distance de la forme inconnue X à tous les échantillons fournis. La forme est alors classée dans la classe de l'échantillon le plus proche (par exemple ci-dessus X serait classé dans la classe Cl1).

- Règle de décision des k plus proches voisins

Il s'agit d'une extension de la règle précédente (plus proche voisin) : pour une forme inconnue X à classer, on va examiner la distance de X à tous les échantillons, qui définissent toutes les classes, puis on sélectionne les k plus proches échantillons et on affecte X à la classe majoritaire parmi ces k échantillons. Dans l'exemple suivant et pour 3 voisins nous classerions X dans la classe Cl2 :

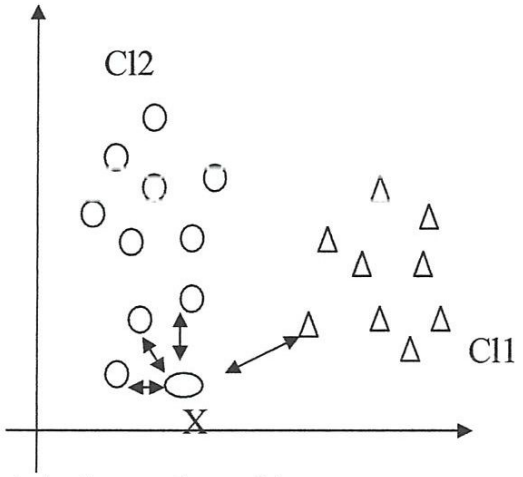


Figure I.9. Méthode de k plus proches voisins

I.6.1.2.2. Méthode des nuées dynamiques

La principale différence avec les k plus proches voisins est que chaque vecteur à classer n'est pas comparé à tous les exemples déjà classés, mais à un seul vecteur représentatif de chaque classe.

Les principales caractéristiques de cette méthode sont les suivantes :

- Chaque classe est représentée par un vecteur : on l'appelle noyau, ou représentant, ou encore prototype.
- Ce représentant est obtenu par combinaison des vecteurs affectés à sa classe par l'algorithme. Souvent, on le prend égal à la moyenne de ces vecteurs.

- Chaque vecteur à classer est comparé à tous les noyaux au moyen d'une mesure de similarité (le plus souvent, la distance euclidienne). Ce vecteur est associé à la classe dont le noyau est le plus proche.
- Le noyau est mis à jour à chaque affectation d'un nouveau vecteur à la classe qu'il représente. L'effet de cette modification est un rapprochement du noyau vers le nouveau vecteur.
- Le représentant initial pour chaque classe peut être un vecteur de cette classe choisi aléatoirement, mais l'algorithme aura plus de chances de s'exécuter efficacement si on utilise plutôt un vecteur représentatif de cette classe, c'est-à-dire situé aux alentours du centre(par exemple égal à la moyenne de quelques points connus de cette classe).

C'est une methode peu couteuse en calculs car chaque vecteur a classer n'est comparé qu'à un seul autre vecteur pour chaque classe : le noyau.

I.6.1.3. conclusion

On peut dire que les méthodes statistiques ont un avantage majeur car elles utilisent un modèle mathématique rigoureux (réduction de la dimension de l'espace de représentation, la classification se fait à l'aide de lois de probabilités…etc.). D'autre part, elles présentent des inconvénients qui peuvent alourdir la reconnaissance (nombreux calculs à faire et nombreuses données à stocker…), et aussi ces méthodes sont peu adéquates à certaines applications (comme l'analyse de scènes dans une image…) car elles ne tiennent pas compte de contraintes structurelles et contextuelles de formes.

Dans ce cas on doit choisir pour notre travail une méthode connexionniste donc la méthode Réseaux des neurones.

Chapitre II

Chapitre II

Les réseaux de neurones

II.1. INTRODUCTION

Conçus à l'origine par des biologistes pour étudier le cerveau humain, les réseaux de neurones sont maintenant principalement utilisés dans le domaine de l'intelligence artificielle.

L'idée principale est de modéliser l'entité de base du cerveau humain, le neurone, puis d'en assembler plusieurs afin de se rapprocher du raisonnement humain.

Comme leur nom l'indique, les réseaux de neurones sont organisés autour d'un ensemble de neurones interconnectés de manière à former un système avec une ou plusieurs entrées et une ou plusieurs sorties.

Les réseaux de neurones, constituent une approche permettant d'aborder sous des angles nouveaux les problèmes de perception, de mémoire, d'apprentissage et de raisonnement. Ils s'avèrent aussi des alternatives très prometteuses pour contourner certaines des limitations des ordinateurs classiques. Grâce à leur traitement parallèle de l'information et à leurs mécanismes inspirés des cellules nerveuses (neurones), ils infèrent des propriétés émergentes permettant de solutionner des problèmes jadis qualifiés de complexes .

Les réseaux de neurones artificiels ou formels sont des structures la plupart du temps simulés par des algorithmes exécutés sur les ordinateurs d'usage général, parfois sur des machines ou même des circuits spécialisés, qui prennent leurs inspirations à partir du fonctionnement élémentaire des systèmes nerveux. Ils sont utilisés essentiellement pour résoudre des problèmes de classification, de reconnaissance de formes, d'association, d'extraction de caractéristique, et d'identification,… etc.

II.2. HISTORIQUE

Les premiers travaux sur les réseaux de neurones remontent à 1943, par la Mc Culloch et Pitts. Ils ont modélisé le fonctionnement du système nerveux à partir d'éléments abstraits, ayant les propriétés des neurones biologiques connues à l'époque. Ce sont eux les premiers à montrer que des réseaux de neurones formels simples peuvent réaliser des fonctions logiques, arithmétiques et symboliques complexes.

En 1949 D. Hebb, qui donna aux réseaux de neurones un premier mécanisme d'apprentissage. La combinaison de ces deux travaux donne ainsi une structure très simple, un neurone et une règle d'apprentissage.

En 1959, F. Rosenblatt développa le modèle du Perceptron. Il construit le premier neuroordinateur basé sur ce modèle et l'appliqua au domaine de la reconnaissance de formes. Ce modèle est une première tentative du neurone orienté vers le traitement automatique de l'information.

En 1960, B. Widrow développa le modèle Adaline (Adaptive Linear Element). Dans sa structure, le modèle ressemble au Perceptron, cependant la loi d'apprentissage est différente. Celle-ci est à l'origine de l'algorithme de rétropropagation du gradient, très utilisé aujourd'hui avec les Perceptrons multicouches.

En 1969, M. Minsky et S. Papert publient une analyse rigoureuse des propriétés du Perceptron, qui met en évidence les limitations théoriques du modèle. Ces limitations concernent notamment l'impossibilité théorique de traiter avec ce modèle, des problèmes non linéaires. Ils étendent implicitement ces limitations à tout modèle de réseaux de neurones artificiels. Ils montrent cependant que les Perceptrons multicouches peuvent lever ces problèmes, mais n'apportent pas de réponses aux difficultés posées par l'apprentissage des couches internes de ces réseaux. Cette publication a pour conséquence de diminuer fortement subsides et les recherches dans le domaine.

Entre 1969 et 1982, toutes les recherches ne sont pas interrompues. De grands noms travaillent toujours durant cette période comme S. Grossberg et T. Kohonen. S. Grossberg publie des travaux sur la théorie de la résonance adaptative qui tente d'inclure des mécanismes d'attention dans des modèles de réseaux à deux couches rebouclées. Kohonen propose un modèle d'auto-organisation qui manifeste des capacités de développement d'une organisation à partir d'une stimulation seule.

En 1982, grâce à la publication de J. Hopfield , on obtient à nouveau une croissance d'intérêt pour les réseaux de neurones artificiels. Il présente une théorie du fonctionnement et des possibilités des réseaux de neurones. La présentation de son article est très anticonformiste, car à l'inverse de tous les travaux présentés jusque là, son modèle fixe préalablement le comportement à atteindre et construit, à partir de là, la structure et la loi d'apprentissage correspondant au résultat escompté. En 1983, la machine de Boltzmann est le premier modèle connu apte à traiter de manière satisfaisante les limitations recensées dans le cas du Perceptron. Mais l'utilisation pratique et en particulier la convergence de l'algorithme, est extrêmement longue.

C'est en 1985 que la méthode de rétropropagation du gradient est découverte et que les limitations du Perceptron mises en avant par M. Minsky sont enfin levées.

Aujourd'hui le domaine des réseaux de neurones est toujours en plein développement, des centaines de compagnies sont impliquées dans des développements d'applications de réseaux de neurones.

II.3. Neurone biologique

En biologie, un neurone est une cellule nerveuse dont la fonction est de transmettre un signal électrique dans certaines conditions. Il agit comme un relai entre une couche de neurones et celle qui la suit. Les caractéristiques des neurones sont encore mal connues (et font l'objet de recherches) mais on connait leur principe d'action[2].

Le corps d'un neurone est relié d'une part à un ensemble de dendrites (entrées du neurones) et d'autre part à un axone, partie étirée de la cellule, qui représentera pour nous sa sortie. Le neurone étudié est connecté aux neurones qui l'environnent : il reçoit au niveau de ses dendrites les signaux électriques des neurones "en amont", propagés par les axones de ces derniers. Les charges électriques s'accumulent dans le neurone jusqu'à dépasser un certain seuil : à ce moment la transmission du signal électrique se déclenche via son axone vers d'autres neurones "en aval".

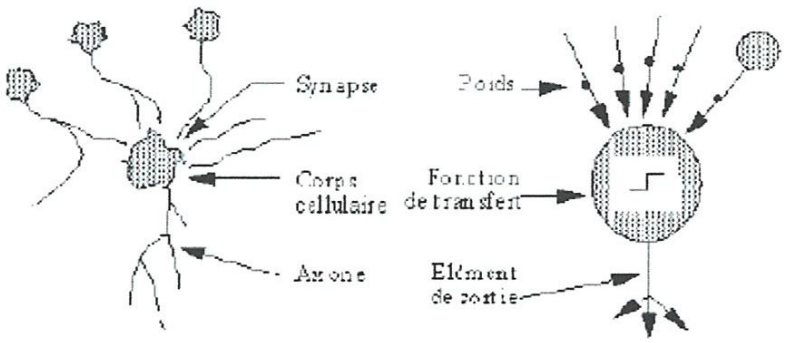


Figure II.1. Schéma du neurone biologique et artificiel

On remarque que les liaisons axone/dendrite entre deux neurones (connexions synaptiques) ne sont pas toutes de la même efficacité. Ainsi l'entrée associée à une certaine dendrite du neurone pourra avoir plus d'influence qu'une autre sur la valeur de sortie. On peut représenter la qualité de la liaison par un poids, sorte de coefficient s'appliquant au signal d'entrée. Le poids sera d'autant plus grand que la liaison est bonne. Un poids négatif aura tendance à inhiber une entrée, tandis qu'un poids positif viendra l'accentuer.

II.4. RESEAUX DE NEURONES ARTIFICIELS

L'idée générale des réseaux de neurones artificiels (RNA) est de trouver une nouvelle procédure de calcul pour produire une nouvelle génération d'ordinateur. Cette idée, inspirée de la biologie du cerveau humain, est différente de la méthode conventionnelle utilisée sur les ordinateurs.

Cette nouvelle technique, semble plus humaine que mécanique. L'homme à toujours rêvé de la machine qui parle, pense, oublie et se rappelle. Le but n'est pas de produire une machine humaine ce qui est impossible, mais de copier et imiter l'approche naturelle du cerveau humain au profit de la science. Un RNA peut être considéré comme une boite noire, qui Sommation Seuillage

Axone Les dentrites Corps cellulaire reçoit des signaux d'entrée et produit des signaux de sortie c'est un modèle mathématique composé d'un grand nombre d'éléments de calculs organisés sous forme de couches interconnectées.

Les réseaux de neurones artificiels sont des réseaux fortement connectés de processeurs élémentaires fonctionnant en parallèle. Chaque processeur calcule une sortie unique sur la

base des informations qu'il reçoit. Toute structure hiérarchique de réseaux est évidemment un réseau[3].

II.5. DIFFERENTS MODELES DE RESEAUX

A la fin des années 50, la neurobiologie ne disposait pas encore des techniques modernes de l'électrochimie pour étudier les structures du cerveau. L'idée de certains chercheurs fût alors d'utiliser les modélisations des neurones et de l'évolution des synapses pour simuler des réseaux de neurones. Ceux-ci pourraient exhiber des comportements proches de ceux du cerveau et ainsi fournir des explications sur le fonctionnement de celui-ci.

II.5.1. L'Adaline

Le modèle de l'Adaline (ADAptive LINear Element) est issu des recherches de Widrow et Hoff sur les filtres adaptatifs, une Adaline est composée d'un seul neurone et d'une couche d'interconnexion de N poids (N entrées et une sortie).

L'Adaline est un réseau de neurones avec une fonction d'activation sigmoïde.

L'Adaline est un système adaptatif simple, il cherche à trouver une relation possible entre les formes d'entrées et des réponses que l'on désire obtenir.

Une importante généralisation de la règle d'apprentissage du perceptron a été présentée par Widrow et Hoff, procédure d'apprentissage des moindres carrés, aussi connue comme règle des deltas. Cette procédure est employée pour minimiser l'erreur quadratique entre la forme d'entrée et la réponse désirée qui est choisie. Le problème est de déterminer les coefficients synaptiques (les poids) de manière telle que les réponses Entrées-Sorties soient correctes, pour un nombre élevé d'individus (d'exemples) pris dans l'ensemble des individus d'apprentissage. La moyenne des erreurs doit être minimisée (au sens des moindres carrés).

II.5.2.Les réseaux de Hopfield

Il s'agit d'un réseau constitué de neurones à deux états (-1 et 1, ou 0 et 1), dont la loi d'apprentissage est la règle de Hebb (1949), qui veut qu'une synapse améliore son activité si et seulement si l'activité de ses deux neurones est corrélée (c'est-à-dire que le poids d'une connexion entre deux neurones augmente quand les deux neurones sont activés au même temps) .

II.5.3. Les réseaux de Kohonen

Contrairement aux réseaux de Hopfield où les neurones sont modélisés de la façon la plus simple possible, on recherche ici un modèle de neurone plus proche de la réalité. Ces réseaux sont inspirés des observations biologiques du fonctionnement des systèmes nerveux de perception des mammifères.

Une loi de Hebb modifiée (tenant compte de l'oubli) est utilisée pour l'apprentissage. La connexion est renforcée dans le cas ou les neurones reliés ont une activité simultanée, et diminuée dans le cas contraire (alors qu'il ne se passait précédemment rien dans ce cas). Ceci se résume par la formule :

$dw / dt = k * S * e - B(S) * w$

Où w est le poids associé à une certaine connexion, e la valeur que le neurone reçoit en entrée par cette connexion, S la valeur qu'il renvoie en sortie (toujours positive), B(S) la fonction d'oubli et k une constante positive.

Les neurones très proches (physiquement) interagissent positivement (le poids des connexions est augmenté autour d'une certaine zone quand une synapse est activée), négativement pour les neurones un peu plus loin, et pas du tout pour les neurones éloignés. Ceci crée un "amas" de neurones activés et contribue à spécialiser certains neurones : pour une entrée donnée, une sortie particulière sera activée alors que les autres resteront inertes. Nous utilisons aussi parfois des lois de concurrence entre les neurones (création et destruction de neurones selon certains critères).

Ceci permet de résoudre certains problèmes, tels le problème du voyageur de commerce (comment relier n villes par le chemin le plus court).

Les réseaux de Kohonen ont des applications dans la classification, le traitement de l'image, l'aide à la décision et l'optimisation.

II.5.4. Les perceptrons multicouches

Ils sont une amélioration du perceptron comprenant une ou plusieurs couches intermédiaires, dites couches cachées, dans le sens où elles n'ont qu'une utilité intrinsèque pour le réseau de neurones et pas de contact direct avec l'extérieur. Chaque neurone n'est relié qu'aux neurones des couches directement précédente et suivante, mais à tous les neurones de ces couches.

Les PMC utilisent, pour modifier leurs poids, un algorithme de rétropropagation du gradient, qui est une généralisation de la règle de Widrow-Hoff. Il s'agit toujours de minimiser l'erreur quadratique. On propage la modification des poids de la couche de sortie jusqu'à la couche d'entrée. Les PMC agissent comme un séparateur non linéaire et peuvent être utilisés pour la classification, le traitement de l'image ou l'aide à la décision.

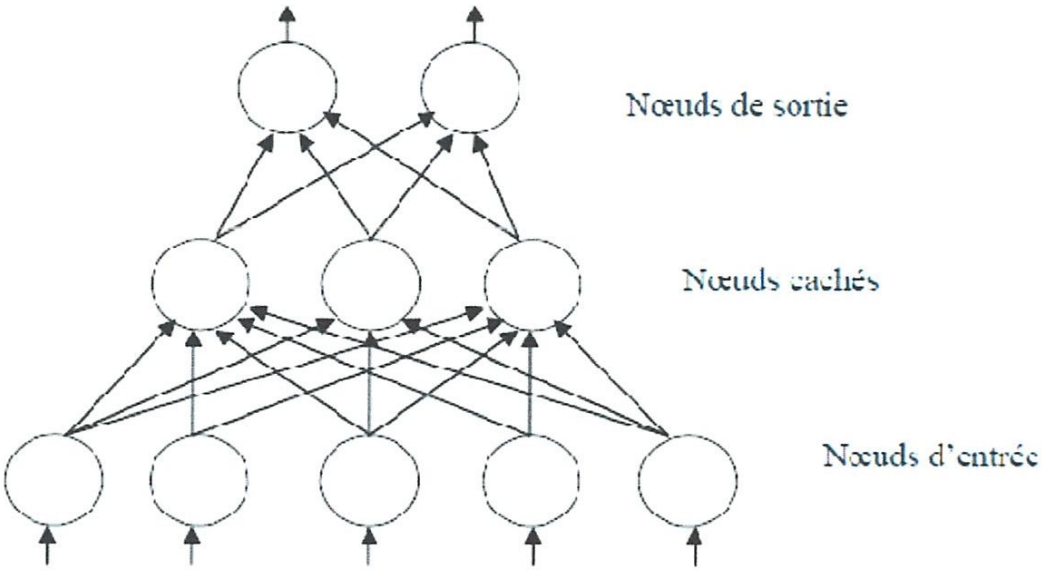


Figure II. Perceptron multicouches

II.6. Le neurone formel généralité

Le premier modèle réalisé en laboratoire fut un automate binaire ou à seuil qui représentait le neurone formel de McCulloch et Pitts (voir figure II.2)[2].

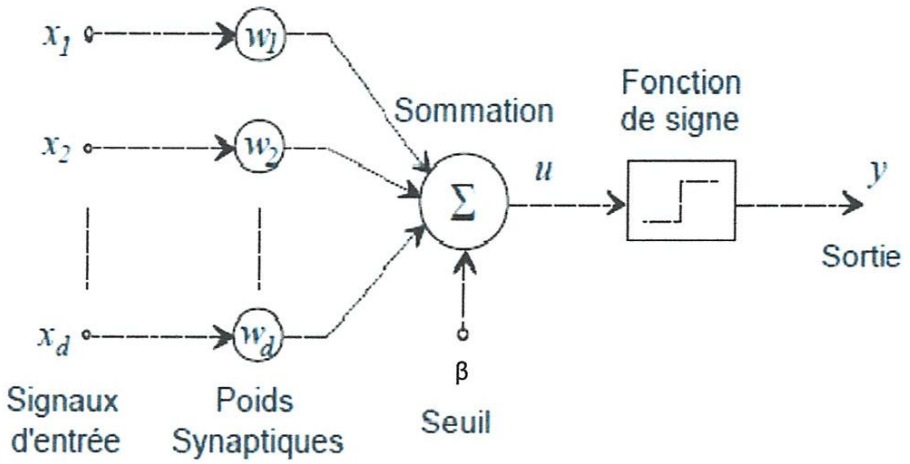


Figure II.2Neurone formel à plusieurs entrées avec seuil de comparaison β

La fonction de seuil utilisée dans ce modèle s'apparentait à la fonction suivante :

$$f(x) = 1 \text{ si } x>P \text{ et } f(x) = 0 \text{ si } x<P \qquad (1)$$

Le neurone formel est un modèle mathématique simplifié du neurone biologique. Le neurone formel est constitué de plusieurs entrées dont chacune est multipliée par un poids. L'ensemble des résultats de chacune des entrées est présenté à un sommateur pour en former une combinaison linéaire. À la figure II.2, on schématise un neurone formel avec ses entrées, ses poids ainsi que le module de sommation qui pondère les entrées.

On appelle apprentissage la recherche de la valeur qui doit être donnée à chacun des poids pour permettre au réseau de classifier convenablement des données. Les méthodes d'apprentissage les plus courantes est l'apprentissage supervisé et non-supervisé. Les seules opérations permises pour ce type de neurone sont les opérations ET, OU, et NON, des opérations qui permettent de résoudre uniquement des problèmes linéairement séparables.

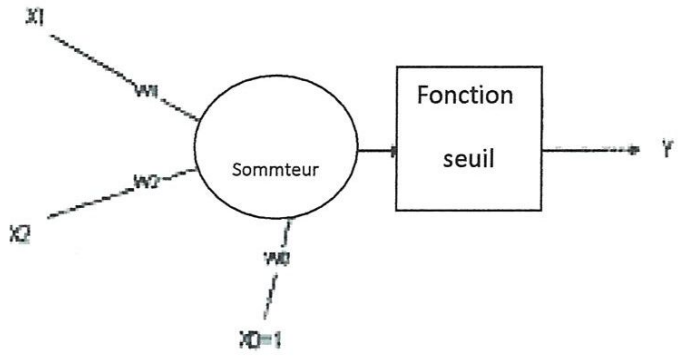


Figure II.3 Neurone formel binaire de McCulloch-Pitts.

Pour bien comprendre le fonctionnement d'un neurone formel, nous allons présenter quelques exemples liés aux fonctions logiques. Nous utiliserons, pour ce faire, les fonctions ET, OU qui, comme nous le verrons, sont des fonctions linéairement séparables et la fonction logique XOR (ou exclusif) qui se trouve être une fonction qui n'est pas linéairement séparable par un neurone formel.

Nous verrons comment l'ajout d'une seconde couche peut résoudre ce type de problème qui se rencontre fréquemment dans les applications quotidiennes.

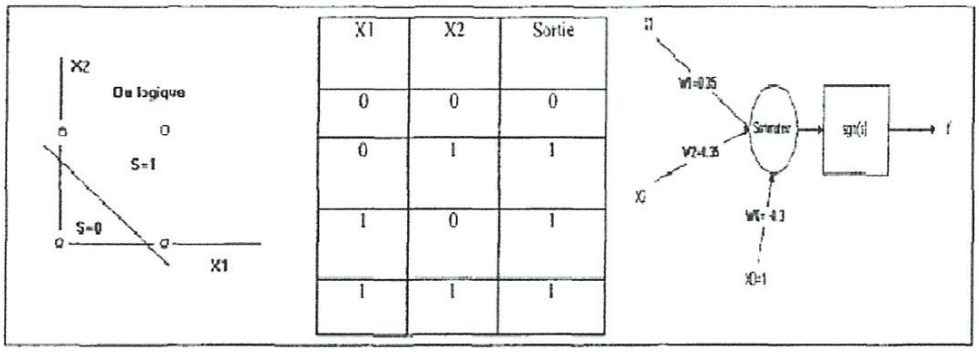
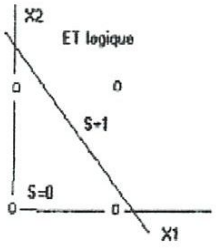


Figure II.4 OU logique : table de vérité et valeur des poids du neurone formel.

Pour comprendre le OU logique, on présente celui-ci sous forme de table de vérité.
On peut voir, sur la figure II.4, les valeurs des entrées et de la sortie pour chacun des états. Les valeurs des poids se stabilisent après apprentissage. Si on présente sur Xi et X2 la valeur zéro, on veut à la sortie du sommateur le résultat suivant:
0.35(0) + 0.35(0) - 0.3 = -0.3 < 0, Y = 0. Si la fonction sgn(s) donne 1 pour S>0 et 0 pour S<0, alors la sortie Y pour cet état donnera la valeur 0, ce qui correspond au comportement de la porte OU. Si nous appliquons maintenant X1=l et X2=0 à la sortie du sommateur, nous obtenons :

0.35(1)+ 0.35(0)-0.3 = +0.05 > 0, Y = 1. Nous aurons le même résultat pour X1=0 et X2=l. Si X1 et X2 sont tous deux égaux à un (1), la sortie vaudra
0.35(1)+ 0.35(1)-0.3 =+0.4>0, Y = 1

Pour ce qui est de la porte ET, avec des valeurs différentes de poids, nous obtenons les sorties désirées pour chacun des états de la porte ET.



X1	X2	Sortie
0	0	0
0	1	0
1	0	0
1	1	1

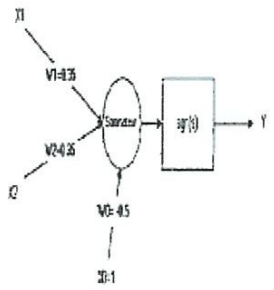
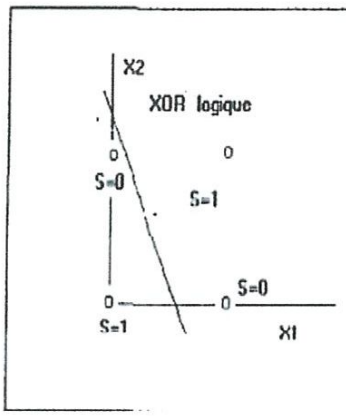


Figure II.5 ET logique : table de vérité et valeur des poids du neurone formel

Pour les fonctions logiques OU et ET, il n'y a aucun problème au niveau de la séparabilité des classes. Par contre, un neurone formel ne pourra résoudre un problème de type XOR. Comme on peut le voir à la figure II.6, la seule manière d'arriver à résoudre ce type de problème est d'ajouter une nouvelle couche contenant un neurone.



X1	X2	Sortie
0	0	0
0	1	1
1	0	1
1	1	0

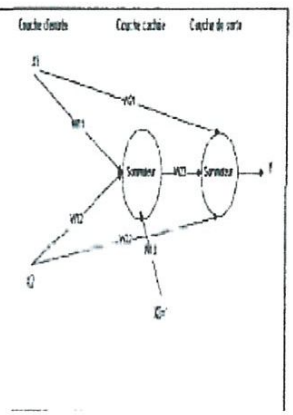


Figure II.6 Problème de séparabilité pour le XOR logique

A la figure II.7, on a une droite de séparation qui ne peut séparer les deux classes.

Pour résoudre ce problème, Minsky et Papper proposèrent une configuration en

multicouches mais il restait à trouver comment effectuer l'ajustement des paramètres de poids et de biais dans les couches cachées.

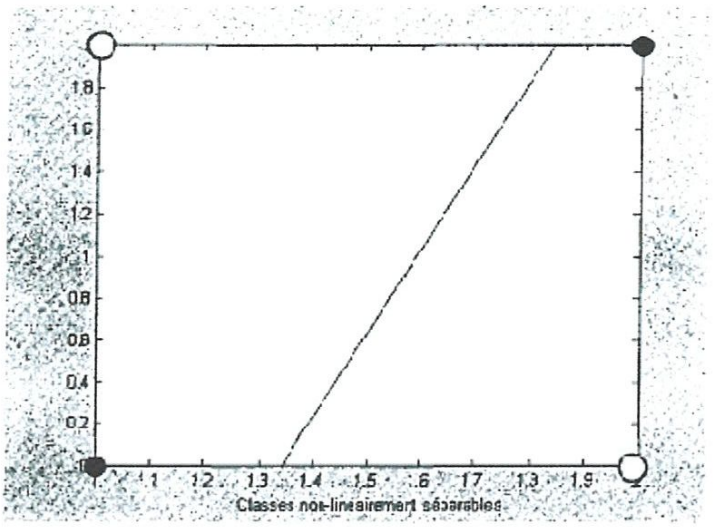


Figure II.7 Classes non séparables pour un perceptron

Entre les années 1970 et 1980, plusieurs chercheurs travaillèrent à la résolution de ce problème et découvrirent finalement un algorithme qui permettait de résoudre ce problème de configuration des poids et des biais.

Cela ouvrit la porte à une véritable révolution dans l'utilisation des RNA (Réseau de Neurones Artificielles) dans toutes les sphères d'activités.

Les réseaux à couches multiples que l'on nomme MLP (Multi-Layer Perceptron)

seront étudiés un peu plus loin ainsi que la règle d'apprentissage de rétropropagation du gradient de l'erreur.

On peut voir sur la figure II.8 la couche d'entrées Xi, Yi représentant la couche cachée et finalement la couche de sortie Zi.

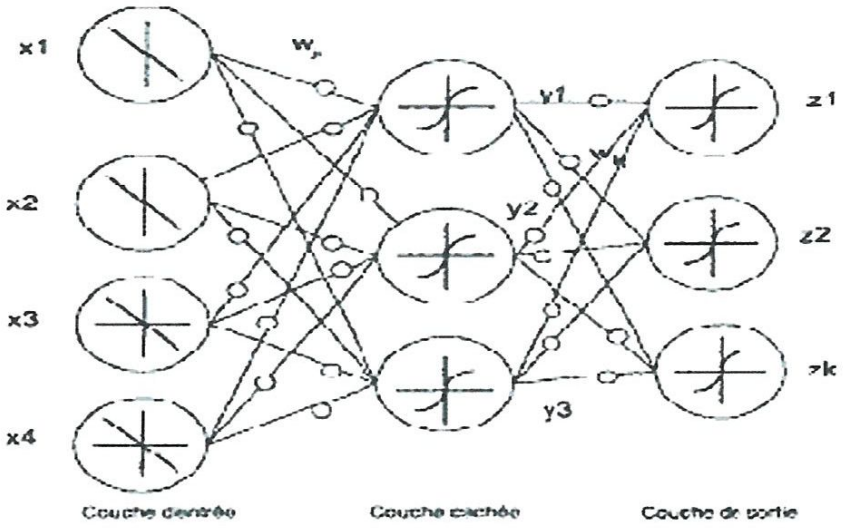


Figure II.8 Réseau MLP à connections directes

Le neurone formel, de par sa nature, agit donc comme un séparateur à fonction discriminante linéaire. L'ajout de plusieurs neurones à la couche d'entrée produit un discriminateur linéaire complexe. Dans le cas d'une fonction discriminante à deux variables, le neurone formel est composé de 2 entrées correspondant aux variables X1 et X2, de poids correspondant à W1et W2 et d'un biais W0.

L'entrée et les poids suivent la relation :

$$S_j = \sum_i W_{ji} X_i \qquad (2)$$

Comme nous le verrons Un peu plus loin, le neurone formel représente l'équivalent (en termes mathématiques) d'une fonction discriminante linéaire.

II.6.1 Discrimination linéaire à deux classes

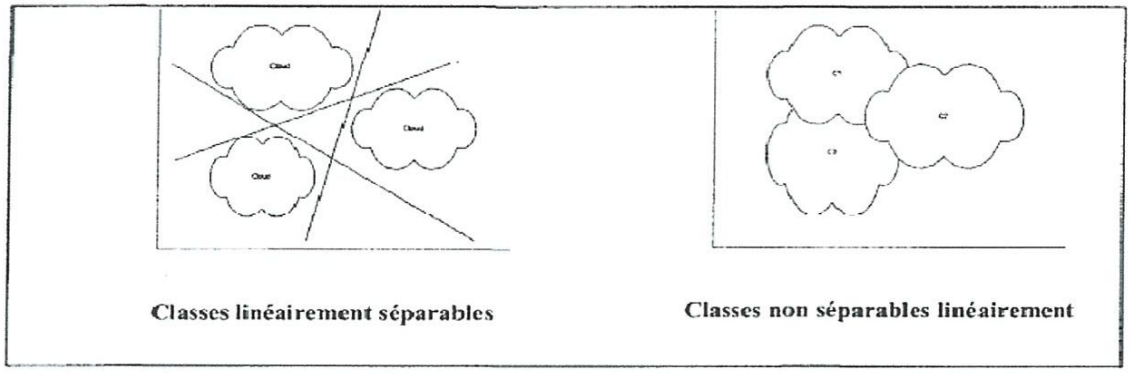


Figure II.9 Séparation linéaire des classes

L'objectif de départ de la discrimination linéaire est de considérer la séparation entre deux classes par une fonction linéaire simple, soit dans ce cas-ci une droite. Dans la figure II.10, deux classes sont séparées par une droite dont l'équation dépend de deux variables soit la variable XI et la variable X2; ces variables étant les caractéristiques propres à chacun des objets, caractéristiques produisant un espace à 2 dimensions dans ce cas-ci, à n dimensions dans le cas général.

Dans le cas d'une forme, on choisirait par exemple la largeur et la hauteur comme caractéristiques des objets.

Pour représenter mathématiquement la classification à deux classes, nous devons représenter graphiquement une séparation de classe à partir d'une droite tracée dans un espace à deux dimensions. L'équation d'une droite est donnée par la relation suivante :

$$d(x, y) = ax+by+c= w1x1+w2x2+w3=0 \qquad (3)$$

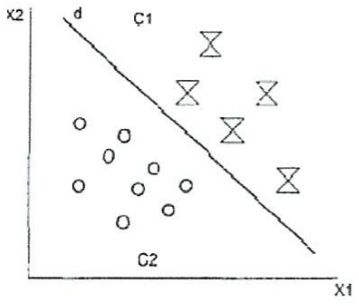


Figure II.10 Séparation de deux classes par une droite

Cette équation est une fonction discriminante linéaire et en réarrangeant cette équation nous mettons en évidence le rôle joué par les paramètres de poids et parameter de biais:

$$y = \frac{-w_1}{w_2}x - \frac{w_3}{w_2} \quad \text{ou} \quad y = mx + b \tag{4}$$

Nous avons remplacé X1 et X2 par x et y, W1 et W2 sont appelés des poids tandis que W3 est appelé le biais. Les paramètres de poids et de biais sont les paramètres qui ajustent la pente et l'ordonnée à l'origine de la droite de séparation. Sous une forme plus généralisée, nous pouvons considérer les variables de caractérisation de la forme par Xi, et les paramètres d'ajustement de la droite de séparation par Wi; pour donner finalement la fonction de discrimination suivante sous une forme plus compacte :

$$d(v) = w_0x_0 + w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_ix_i - \sum_{i=0}^{n} W^T{}_1X_i - 0 \tag{5}$$

où $W^T{}_i = (w_0, w_1, w_2, \ldots, w_i)$ et $X_i = (x_0, x_1, x_2, \ldots, x)$

avec $x_0 = 1$ et w_0 est le biais.

II est évident, selon la figure II.10, que pour d(v)>0 la donnée se classe dans Ci tandis que pour d(v)< 0 la donnée est classée dans C2 :

$$\begin{aligned} v &\in C_1 \quad \text{si } d(v) > 0 \\ v &\in C_2 \quad \text{si } d(v) < 0 \\ v &\in \text{Frontière} \quad \text{si } d(v) = 0 \end{aligned} \tag{6}$$

Ce concept se généralise pour la séparation à plusieurs classes, mais il peut arriver d'avoir des régions qui demeurent indéterminées comme on peut le voir à la figureII.11.

Aussi, dans le cas de problèmes dont les classes ne sont pas linéairement séparables, il arrive que l'on choisisse des sous-classes telles que l'on puisse les séparer convenablement[6].

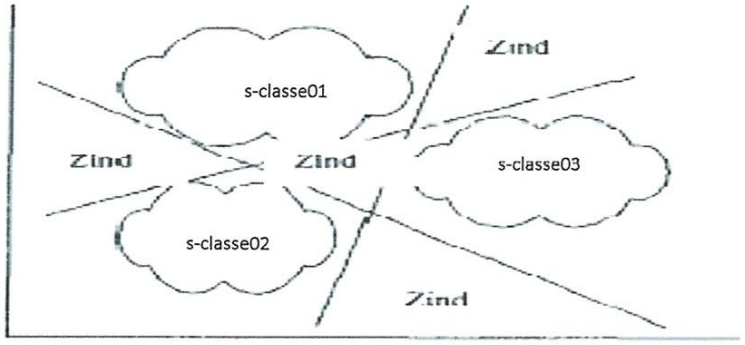


Figure II.11 Zones indéterminées (Zind) d'un classifieur linéaire.

Dans un problème à deux classes, les paramètres de la droite, du plan ou de l'hyperplan qui sont nécessaires lors de la séparation des classes doivent pouvoir se modifier aisément et c'est là un problème qui fut résolu par les méthodes d'apprentissage.

Présentement, deux méthodes d'apprentissage sont couramment utilisées, soit l'apprentissage supervisé et l'apprentissage non-supervisé.

L'apprentissage est dit supervisé si les différentes familles des formes sont connues a priori et si la tâche d'apprentissage est guidée par un superviseur ou professeur.

L'apprentissage se déroule de la manière suivante : on choisit un sous-ensemble S de l'ensemble E des formes à reconnaître. Pour chacune des formes appartenant à ce sous-ensemble, le professeur indique dans quelle classe doit se trouver chacune des formes.

Lors de l'étape d'apprentissage, les meilleures séparatrices sont calculées pour distinguer les classes d'appartenance. Une fois complétée cette étape, on applique alors l'ensemble E des formes au modèle ainsi défini. Il est entendu que le sous-ensemble se doit d'être représentatif de toutes les données pour comporter au moins un représentant par classe. Le problème dans ce type d'apprentissage est qu'il faut un nombre élevé d'éléments dans le sous-ensemble S

pour une détermination précise des droites de séparation de classes. Dans l'expérimentation, on choisit un petit ensemble de données pour chacun des objets que l'on présente en exemple au réseau de neurones. Ce sous-ensemble de données se doit d'être assez représentatif pour caractériser l'objet à être reconnu par le réseau.

L'apprentissage non-supervisé regroupe des données en formant des nuages vectoriels séparés selon une distance prédéfinie. Aucun exemple n'est nécessaire, la méthode d'apprentissage est basée sur une recherche statistique selon des caractéristiques bien précises attribuées au départ. On utilise pour ce faire soit la mesure de distance euclidienne ou soit la mesure de distance de Manhattan comme logique de séparation des nuages vectoriels. On annote à la toute fin les classes ainsi trouvées. Pour ne citer que quelques méthodes, on retrouve la méthode d'apprentissage bayésienne et la méthode GMM (Gaussian Mixture Method).
Peu importe l'algorithme d'apprentissage, il faut que les classes soient linéairement séparables, sinon on aurait des erreurs de classification. On peut citer comme exemple l'algorithme du perceptron ou l'algorithme d'Arkadev et Braverman .L'algorithme du perceptron est plus simple. Sa convergence vers la solution est un peu plus lente mais certaine0. Voici un exemple de calcul des poids et du biais en utilisant l'algorithme du perceptron :

Étape 1 -Choisir des valeurs aléatoires pour les poids w1,w2 et le biais θ

Étape 2 -Appliquer le vecteur d'entrée x(i)

Étape 3 -Calculer la valeur de sortie S $S\,(S = w_1 x_1 + w_2 x_2 + \theta)$

Étape 4 - Si la valeur de sortie est égale à la valeur cible, S = d(i)
retourner à l'étape 2 sinon continuer à l'étape 5

Étape 5 -La variation du biais est égale à la valeur de la cible et la variation du poids w(i) est égale à la valeur de l'entrée x(i) fois la valeur de la cible

désirée : $d\theta = \gamma(d_i - o_i)dw_i = \gamma x(i)(d_i - o_i)$

Étape 6 - $\theta = \theta + d\theta$, $w_i = w_i + dw_i$ et retourner à l'étape 2.

Dans l'étape 5, le symbole y spécifie le pas d'apprentissage. Si le pas est trop petit, la convergence risque d'être faussée et si le pas est trop grand, il y a un risque de ne pas arriver à une convergence. Dans notre exemple, nous avons utilisé un pas unitaire. Après plusieurs passages de l'ensemble des vecteurs d'entrée, les poids et les biais seront ajustés de telle sorte que lors de l'application de toutes les données, celles-ci se classeront automatiquement, et que la valeur de sortie S correspondra à la classe qui est associée à la donnée d'entrée.

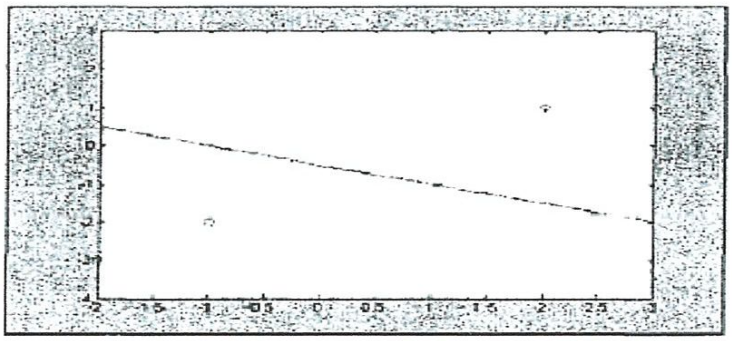


Figure II.12 Droite séparatrice avant apprentissage.

Il faut prendre soin de remarquer que les échelles sont différentent même si les coordonnées des points demeurent identiques dans la figure II.12 et II.13.

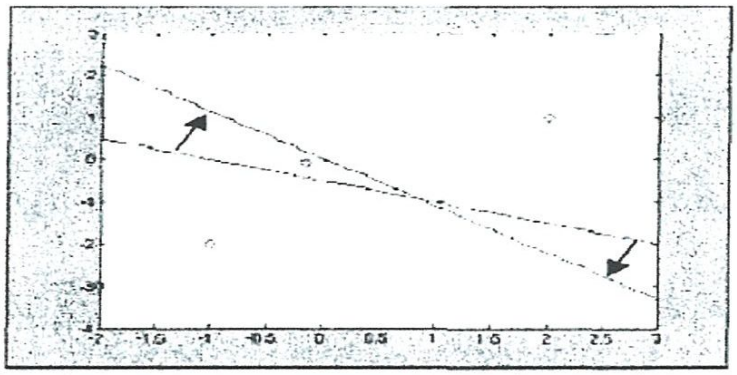


Figure II.13 Droite séparatrice avant et après apprentissage.

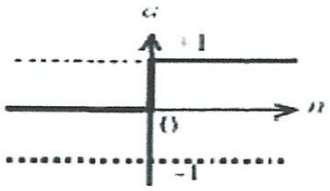
Les figures II.12 et II.13 représentent le déplacement de la droite séparatrice avant et après apprentissage. Sur la figure II.12, la droite sépare deux classes représentées par deux points distincts.

Dans l'algorithme du perceptron, il arrive que les coefficients des poids soient à zéro parce que le taux d'apprentissage est inexistant et que la valeur des poids varie dans ce cas très fortement; il faudra dans ce cas choisir de nouvelles valeurs pour l'initialisation des poids car lorsque ceux-ci ont une valeur nulle, le calcul n'évolue plus ayant atteint un faux minimum.

On doit alors reprendre l'ensemble de tous les vecteurs de la base d'apprentissage et les représenter de nouveau dans un ordre aléatoire pour donner une chance à l'algorithme de converger rapidement. La figure II.13 présente deux droites alors que nous ne devrions voir qu'une seule droite, ceci pour indiquer l'évolution de la pente de la droite lorsqu'un nouvel objet est inséré dans la base d'apprentissage.

Voyons maintenant une autre approche qui nous est donnée par le concept de neurone formel. Le neurone formel est un modèle très simplifié simulant le fonctionnement d'un neurone biologique. Son fonctionnement s'apparente aux fonctions discriminantes linéaires. Les neurones formels servent comme brique de base des réseaux de neurones étudiés dans ce document. Nous verrons également qu'il existe d'autres fonctions d'activation dépendant du type de réseau et des règles d'apprentissage[3].

Dans le cas d'une classification binaire, la fonction d'activation utilisée est la fonction de Heaviside ou fonction seuil qui se nomme «hardlim» dans MATLAB. Cette fonction (figure II.14) produit une valeur discrète égale à zéro (0) ou un (1) indiquant le résultat de la sortie. Par opposition, les fonctions discriminantes linéaires ont une fonction d'activation continue (fonction « purelin » dans MATLAB) voir figure II.15.



A= hardlim(n)

Figure II.14 Fonction d'activation à seuil binaire

Le résultat en sortie de la fonction « purelin » est le même que celui présenté à l'entrée de la fonction. Cette fonction se retrouve généralement à l'entrée des réseaux multicouches (MLP) étant donné que sa dérivée première existe. Nous verrons qu'il est important pour les réseaux multicouches d'avoir des fonctions d'activation non linéaires pour l'étape d'apprentissage car l'algorithme de rétropropagation du gradient de l'erreur doit sa fonctionnalité de la non linéarité des fonctions d'activation.

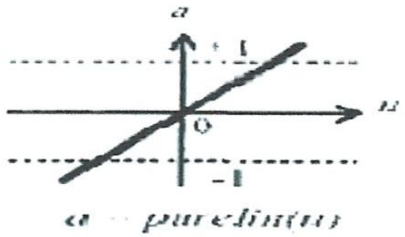


Figure II.15 Fonction d'activation à seuil binaire

Dans le cas où la fonction d'activation est non-linéaire, nous avons les fonctions « logsig » et « tansig ». La fonction « logsig » évolue entre 0 et 1 tandis que la fonction « tansig » évolue entre moins un (-1) et plus un (1). L'utilisation d'une fonction nonlinéaire telle que le sigmoïde vient du fait que sa dérivée existe et qu'elle est simple à mettre en oeuvre. Par exemple, pour la fonction « logsig », l'équation se traduit comme suit :

$$F(s)=\frac{1}{1+e^{-s}} \quad \text{et} \quad F'(s) = F(s)(1 - F(s)) \tag{7}$$

Le fait que la dérivée de cette fonction soit exprimable, de cette manière, diminue les temps de calcul pour les réseaux MLP à plusieurs neurones. Dans MATLAB, la fonction « logsig » a l'allure de celle présentée à la figure II.16.

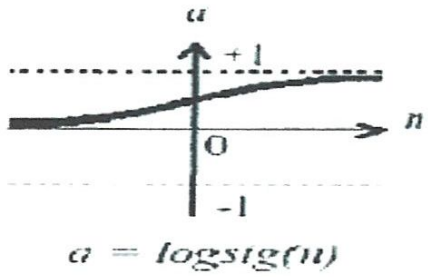


Figure II.16 Fonction logsig

II.6.2.Neurone formel (Résumé)

La figure II.17 présente le fonctionnement d'un neurone formel . Il s'agit d'un composant calculatoire faisant la somme pondérée des signaux reçus en entrée (calculant la quantité h) à laquelle on applique une fonction de transfert, ici la fonction seuil , afin d'obtenir la réponse de la cellule (notée y)[4].

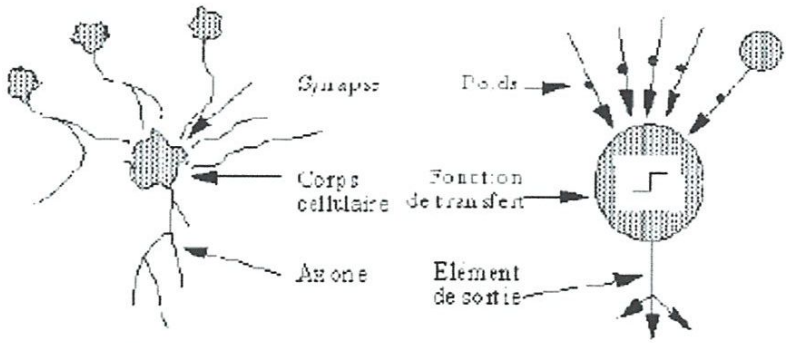
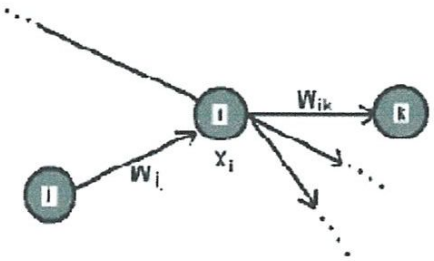


Figure II.17 : Identification entre neurone biologique et neurone formel

La figure II.18 donne les notations que nous utilisons.



La figure II.18 notations

En reprenant les notations de l'illustration II.18, la sortie d'un neurone est définie par :

$$y=H\left(\sum_{j=1}^{n}(\omega_j x-u)\right) \qquad (8)$$

où u désigne le seuil de la fonction de transfert H (Heavy-side, sigmoïde,...). Il est possible d'une part de généraliser ce comportement à n'importe quelle cellule i, il suffit de rajouter un indice supplémentaire à toutes les quantités afin de spécifier leur relation avec i, d'où l'on obtient :

$$y_i=H(h_i),\ h_i=\sum_{j=1}^{n_i}(\omega_{ij}x_j \quad u_i) \qquad (9)$$

II.6.3. Perceptron une couche

le perceptron se veut un modèle de l'activité perceptive. Il s'agit d'associer des configurations présentées en entrée sur la rétine à des réponses. Constitué de 2 couches de cellules (et donc une couche de connexions), l'une pour la saisie (la rétine), l'autre pour la réponse. Imaginons un perceptron constitué d'une rétine 6X6 et d'une couche de sortie de 26 cellules. La tâche du perceptron sera de reconnaître les lettres présentées sur la rétine, chaque lettre sera associée à une cellule de sortie. Ainsi, lorsque l'on présente un A on souhaite que la première cellule soit active et toutes les autres inactives.

II.6.3. 1.Description mathématique

Le nombre de cellules (de chaque couche) est adapté à la dimension de l'espace des valeurs.

À chaque cellule de la couche de sortie on associe une valeur θi appelée seuil d'activation, ainsi qu'une fonction de transfert fi qui renvoie la valeur +1 si la somme pondérée des entrées est supérieure au seuil et 0 sinon (on se place dans l'hypothèse de

cellules binaires i.e. deux états au choix, 0 et 1 ou -1 et 1). Plus formellement, si on note ωij la connexion de la cellule j (rétine) vers la cellule i (sortie), on pose

$$a_i = \sum_i (\omega_{ij} s_j - \theta_i) \tag{10}$$

où sj désigne le signal transitant entre les cellules j et i .

L'objectif est d'entraîner le réseau jusqu'à ce que la valeur obtenue en sortie soit égale à la valeur désirée (on est dans le cadre d'un apprentissage supervisé).

Pour cela on utilise la règle de Hebb (du nom de son inspirateur Donald Hebb) qui consiste à renforcer les connexions entre deux cellules si elles répondent similairement et à diminuer cette valeur sinon. On pose :

$$\omega_{ij} = \omega_{ij} + \eta (d_i - o_i) s_j \tag{11}$$

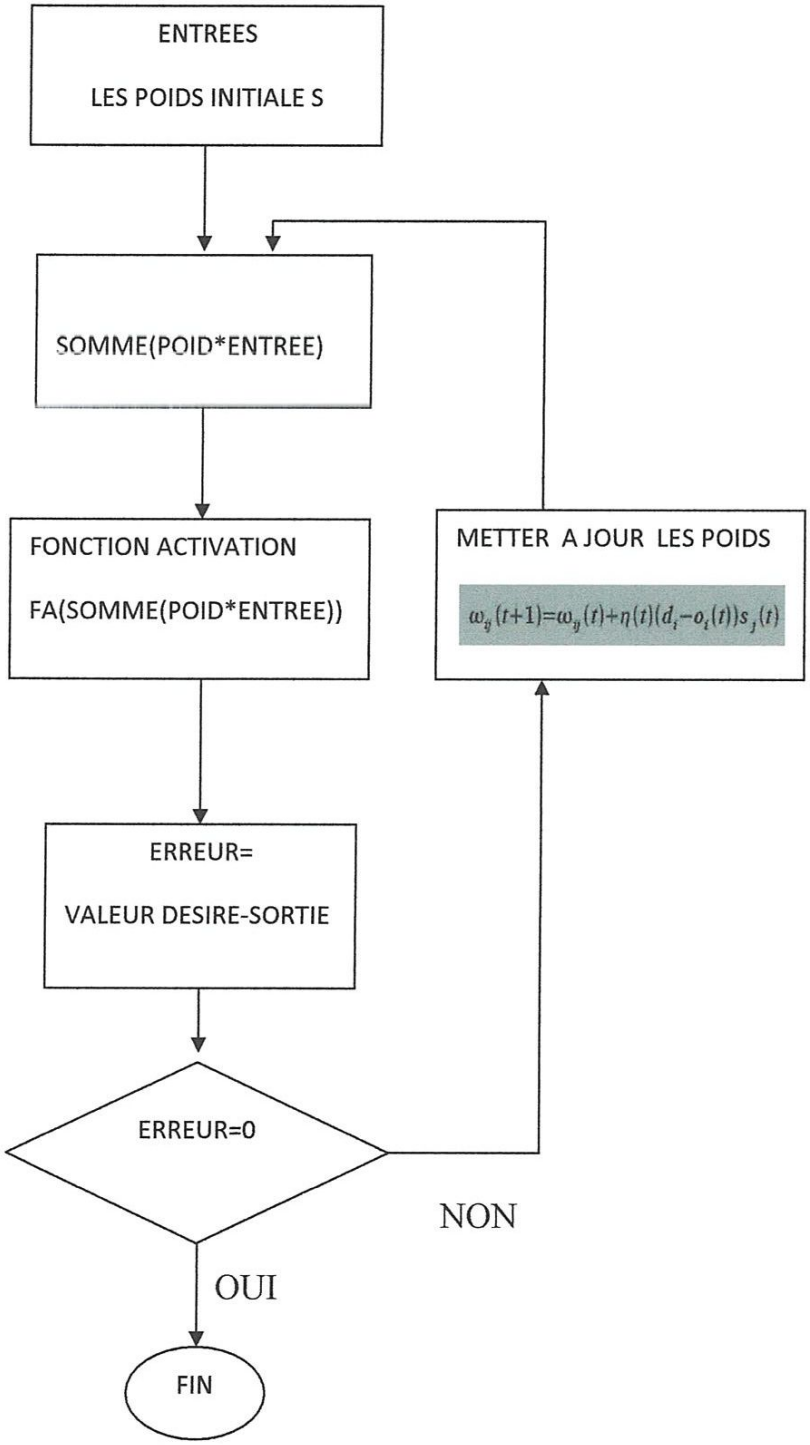
ou encore, si on rend le temps explicite dans l'équation

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \eta(t)(d_i - o_i(t)) s_j(t) \tag{12}$$

dans laquelle ai(t) désigne la valeur calculée (au temps t) par le réseau sur la sortie i, tandis que di correspond à la valeur désirée ; sj(t) correspond au signal transitant sur la connexion de j à i de poids ωij(t), η(t) étant le pas (ou coefficient) d'apprentissage au temps t. Ce pas sert à moduler l'impact de l'erreur sur la connexion.

II.6.3. 2Algorithme du perceptron une seul couche

L'algorithme d'apprentissage :



II.6.3. 3 Limitations et propriétés fondamentales

Rosenblatt a démontré la convergence de l'algorithme d'apprentissage du perceptron : « Si le problème admet une solution, alors l'algorithme trouvera une solution au problème en un temps fini et ce indépendamment de l'ordre de présentation des données ».

L'ordre de présentation des données n'influe pas sur la découverte de la solution mais sur les étapes du traitement.

Le perceptron ne résout que des problèmes linéairement séparables.

Tout problème non linéairement séparable peut être transformé en un problème équivalant linéairement séparable, mais il n'existe pas d'algorithme permettant d'automatiser la transformation.

II.6.4. Perceptron multicouches

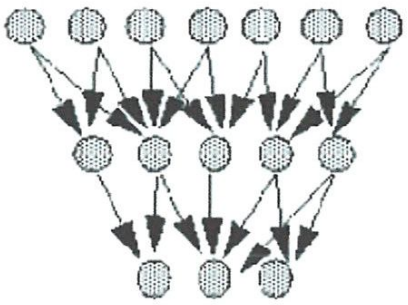


Figure II.19 : Réseau à couches multiples

La séparation linéaire proposée par le perceptron à une seule couche n'est pas suffisante pour tout les classes des problèmes (linéaire, non linéaire). Le perceptron multicouche offre une alternative intéressante et très utilisée.

Noter algorithme de reconnaissance des formes est basé sur cette approche (Application chapitre III).

On considère un réseau possédant $l+1$ couches de connexions, c'est-à-dire avec l couches cachées $H^1 \ldots H^l$, auxquelles on rajoute la rétine, notée R et la couche de sortie notée S. Chaque couche de cellules contient un certain nombre de cellules, notés $N_0, N_1, N_2, \ldots, N_l$,

N_{l+1}. Une connexion joignant une cellule i à une cellule j sera notée ω_{ji}^{k} si elle appartient à la $k^{ème}$ couche de connexions. La sortie de la cellule i appartenant à la $k^{ème}$ couche de cellules sera notée $s_i^k = f_i(a_i^k)$; lorsque k=0, (la cellule appartient à la rétine) on pose $s_i^0 = x_i$, lorsque k= l +1 (la cellule appartient à la couche de sortie) on pose $s_i^k = a_i$. On appelle **activation** de la cellule i de la $k^{ème}$ couche la valeur :

$$\forall k, 1<k<l+1, a_i^k = \sum_{j-1}^{N_k} (\omega_{ij}^{(k-1)} s_j^{(k-1)} - 0_i) \tag{13}$$

Enfin on notera, d_i^p la valeur désirée sur la cellule i pour le pattern p , et o_i^p la valeur obtenue pour le pattern p sur la sortie i.

II.6.4. 1.Description Mathématique (Algorithme de rétro-propagation)

Maintenant que les notations sont définies, nous allons voir comment on peut généraliser la règle delta dans le cadre du PMC. On considérera que toutes les fonctions de transfert utilisées sont continues et dérivables. On définit l'erreur globale du système comme la somme des erreurs partielles obtenues pour chaque pattern fourni en entrée. En d'autre terme :

$$E = \sum_p E_p = \sum_p \sum_{i=1}^{N_{(l+1)}} \left| \frac{1}{2}(d_i^p - o_i^p) \right| \tag{14}$$

On cherche maintenant à minimiser l'erreur quadratique instantanée du système, par la méthode du gradient stochastique. On définit la variation sur un poids ωij pour un pattern p par :

$$\Delta_p \omega_{ij} = -\eta \frac{(\partial E_p)}{(\partial \omega_{ij})} \tag{15}$$

Il est maintenant nécessaire de calculer cette expression. Nous allons scinder le calcul en deux parties, d'une part pour les connexions allant vers la couche de sortie (conditions aux bords), et d'autre part pour les connexions aboutissant sur une couche cachée.

II.6.4. 1.1. Vers la couche de sortie

Dans ce cas, on rappelle que $s_i = o_i^p$. On définit les notations suivantes :

$$s_i^{(l+1)} = s_i$$
$$a_i^{(l+1)} = a_i$$
$$\omega_{ij}^{l} = \omega_{ij}$$

On peut décomposer la variation du poids ωij en deux terme, par la règle de la dérivée de fonction composée:

$$\frac{(\partial E_p)}{(\partial \omega_{ij})} = \frac{(\partial E_p)}{(\partial s_i)} \frac{(\partial s_i)}{(\partial \omega_{ij})} \tag{16}$$

Le premier terme s'évalue directement de la relation (14):

$$\frac{(\partial E_p)}{(\partial s_i)} = -(d_i^p - o_i^p) \tag{17}$$

Le second terme nécessite une deuxième décomposition, car la sortie s_i est encore fonction de l'activation a_i du neurone i.

$$\frac{(\partial s_i)}{(\partial \omega_{ij})} = \frac{(\partial s_i)}{(\partial a_i)} \frac{(\partial a_i)}{(\partial \omega_{ij})} \tag{18}$$

Le premier terme de cette décomposition n'est autre que la dérivée de la fonction de transfert évaluée pour l'activation a_i du neurone i:

$$\frac{(\partial s_i)}{(\partial a_i)} = f'_i(a_i) \tag{19}$$

Le deuxième terme vient directement de la relation (13):

$$\frac{(\partial a_i)}{(\partial \omega_{ij})} = s_j^l \tag{20}$$

En résumé on trouve :

$$\Delta_p \omega_{ij} = -\eta \frac{(\partial E_p)}{(\partial \omega_{ij})} = \eta (d_i^p - o_i^p) f'_i(a_i) s_j^l \tag{21}$$

II.5.4. 1.2.Vers une couche cachée

Il va falloir maintenant exprimer la modification des poids pour une connexion qui ne va pas sur la couche de sortie.

On suppose que l'on souhaite modifier le poids d'une connexion ω_{ij} appartenant à une couche c, $1 \leq c \leq |$. Il s'agit donc d'arriver à exprimer E_p en fonction de ω_{ij}^c, on va le faire en utilisant la composition des dérivées. Regardons comment cela fonctionne pour c=| avant de généraliser à n'importe quelle valeur de c.

En analysant les formules, on constate que ω_{ij}^1 sert au calcul de a_i^1 qui lui-même permet le calcul de s_i^1, cette valeur est injectée au niveau des connexions ω_{ki}^{l+1} et contribue donc aux erreurs obtenues lors du calcul de chaque o_k^p.

Ce qui nous permet d'écrire :

$$\frac{(\partial E_p)}{(\partial \omega_{ij}^l)} = \sum_{k=1}^{N_{l+1}} \left(\frac{(\partial E_p)}{(\partial s_k^{(l+1)})} \frac{(\partial s_k^{(l+1)})}{(\partial a_k^{(l+1)})} \frac{(\partial a_k^{(l+1)})}{(\partial s_i^l)} \frac{(\partial s_i^l)}{(\partial a_i^l)} \frac{(\partial a_i^l)}{(\partial \omega_{ij}^l)} \right) \tag{22}$$

Il est possible maintenant de calculer chacun des termes intervenants dans la somme.

Le premier terme s'évalue directement de la relation (14):

$$\frac{(\partial E_p)}{(\partial s_k^{(l+1)})} = -(d_k^p - o_k) \tag{23}$$

Le deuxième terme n'est autre que la dérivée de la fonction de transfert du $k^{ième}$ neurone de la couche l+1, évaluée pour l'activation de ce même neurone:

$$\frac{(\partial s_k^{(l+1)})}{(\partial a_k^{(l+1)})} = f'_k(a_k^{(l+1)}) \tag{24}$$

Le troisième terme s'obtient par dérivation de l'expression générale de l'activation d'un neurone, décrite par la relation (13):

$$\frac{(\partial a_k^{(l+1)})}{(\partial s_i^l)} = \omega_{ki}^l \tag{25}$$

Le quatrième terme, comme pour le deuxième, est la dérivée de la fonction de transfert, mais évaluée pour le neurone i de la couche l

$$\frac{(\partial s_i^l)}{(\partial a_i^l)} = f'_i(a_i^l) \tag{26}$$

Le dernier terme, enfin, découle comme le troisième de la dérivation de la relation d'activation (13):

$$\frac{(\partial a_i^l)}{(\partial \omega_{ij}^l)} = s_j^{(l-1)} \tag{27}$$

À partir de là, il est immédiat d'étendre l'égalité pour une couche quelconque c telle que $1 \leq c \leq l$. Et ainsi bâtir la formule générale mise en oeuvre dans le cadre de la rétro-propagation du gradient. Pour une couche c quelconque, il suffit de dire que ω_{ij}^{c} participe au calcul de a_{i}^{c}, ce dernier permet de calculer s_{i}^{c}, cette valeur étant ensuite injectée sur les connexions issues de i et aboutissant sur les cellules k de la couche c+1, contribuant ainsi aux erreurs de la couche c+1, d'où l'expression suivante :

$$\frac{\partial E_p}{\partial \omega_{ij}^{c}} = \sum_{k=1}^{N_{c+1}} \left(\frac{\partial E_p}{\partial s_k^{(c+1)}} \frac{\partial s_k^{(c+1)}}{\partial a_k^{(c+1)}} \frac{\partial a_k^{(c+1)}}{\partial s_i^{c}} \frac{\partial s_i^{c}}{\partial a_i^{c}} \frac{\partial a_i^{c}}{\partial \omega_{ij}^{c}} \right) \tag{28}$$

II.5.4. 1.3.Formule générale

Formule Adapter à n'importe quelle couche de connexions. Pour cela on va poser :

$$\delta_i^{c} = -\frac{\partial E_p}{\partial s_i^{c}} \tag{29}$$

Ce qui va nous permettre de reformuler la variation de poids :

$$\Delta_p \omega_{ij}^{c} = \eta \, \delta_i^{c} s_j^{(c-1)}, \forall c, 1 < c < l+1 \tag{30}$$

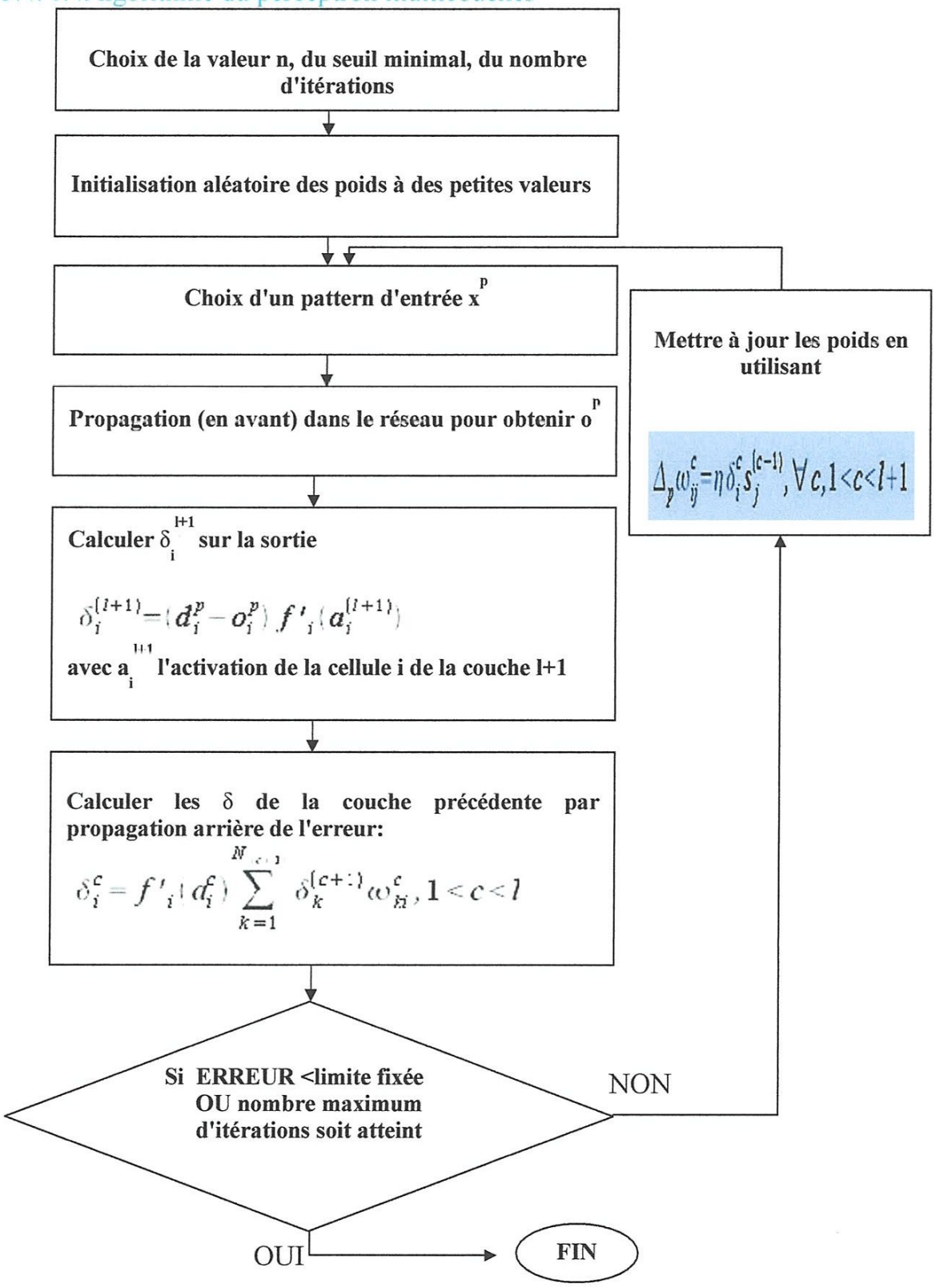
Avec :

$$\delta_i^{(l+1)} = (d_i^p - o_i^p) f'_i(a_i^{(l+1)}) \tag{31}$$

$$\delta_i^{c} = f'_i(a_i^{c}) \sum_{k=1}^{N_{c+1}} \delta_k^{(c+1)} \omega_{ki}^{c}, 1 < c < l \tag{32}$$

Cette réécriture nous permet ainsi d'obtenir une procédure récursive pour le calcul des δ pour chaque cellule du réseau et nous rend possible la définition del'algorithme général de rétro-propagation du gradient pour l'apprentissage dans un PMC.

II.5.4. 1.4Algorithme du perceptron multicouches



II.5.4. 1.4. Conclusion

Tout ce passe comme si on utilisait le réseau dans deux directions, un calcul ``en avant'' pour déterminer la réponse du réseau à une ``stimulation'', puis un calcul en ``arrière'' pour évaluer les modifications à apporter aux connexions afin d'améliorer le comportement global du système. Il est important de remarquer que le réseau ne peut influer que sur les poids des connexions et (éventuellement) sur les seuils des cellules. Il ne peut, en aucun cas modifier son architecture ni les fonctions de transferts utilisées. De plus pour que le calcul soit possible il est impératif que les fonctions de transfert soient continues (et donc dérivables).

Par ailleurs, du fait même des propriétés sur le calcul du gradient on n'a plus la garantie de la convergence, c'est pourquoi on a besoin d'introduire une borne sur le nombre d'itérations (époques) De plus, comme on effectue des calculs sur machine, on sait que ces calculs sont entachés d'erreur, il devient donc illusoire de vouloir atteindre les valeurs optimales, d'où la nécessité d'introduire un seuil minimal d'erreur (ε). Enfin, comme le processus d'apprentissage est initialisé aléatoirement, il est nécessaire d'effectuer plusieurs simulations avec les mêmes paramètres initiaux - ceci nous assure une stabilité statistique de l'apprentissage.

On peut remarquer que le réseau est sujet à de très nombreux paramètres, la plupart sont à la charge du concepteur et peu de latitude est laissée au réseau. Une fois choisies l'architecture (nature du réseau, nombre de couches, nombre de cellules par couches) et les bases d'apprentissage, de test et de validation, il reste encore de nombreux paramètres à déterminer, parmi ceux-ci on peut citer :

1.-La fonction de transfert

2.-La fonction d'erreur

3.-Le pas d'apprentissage

Chapitre III

APPLICATION

L'organisation du chapitre III

Application 01 :

Les formes choisi sont des forme géométrique (triangle, rectangle et cercle)

donc trois classes.

Application 02 :

Les formes choisi sont des chiffres dans un cadre (plaque immatriculation automobile).

L'objectif est de prendre une photo par la webcam et la classer selon

la notion perceptron multicouches dans la classe approprié.

- obtenir une représentation des données à traiter qui soit manipulable en machine.
- élimination des bruits, normalisation, amélioration des contrastes, segmentation, etc...
- obtenir une représentation des données compatible avec les outils d'apprentissage et de décision utilisés.
- à partir d'un ensemble d'exemplaires et d'un modèle, construire une représentation des classes (apprentissage).
- valider les décisions de l'analyse sur la base des classes connues, pour une éventuelle modification du système, ou de certaines de ses parties.

III.1. INTRODUCTION

La notion d'image:

Une image réelle est obtenue à partir d'un signal continu bidimensionnel comme par exemple un appareil photo ou une caméra... Sur un ordinateur, on ne peut pas représenter de signaux continus, on travaille donc sur des valeurs discrètes.

III.2. Définition image:

Une image numérique est définie comme un signal fini bidimensionnel échantillonné à valeurs quantifiées dans un certain espace de couleurs. Elle est constituée de points (pixels).

Signal fini : une image possède des dimensions finies, exemple : 640x480, 800x600 points…

Signal bidimensionnel : une image possède deux dimensions : largeur, hauteur.

Signal échantillonné : les pixels d'une image sont régulièrement espacés sur une grille carrée.

Valeurs quantifiées : les valeurs des pixels appartiennent à un intervalle borné connu.

Espace de couleur : il existe de nombreuses façon de percevoir les couleurs d'une image, l'espace de représentation le plus connu est l'espace rgb (rouge-vert-bleu).

Autrement dit, une image est une matrice MxN de valeurs entières prises sur un intervalle borné [0,Ng] où Ng est la valeur maximale du niveau de gris[5].

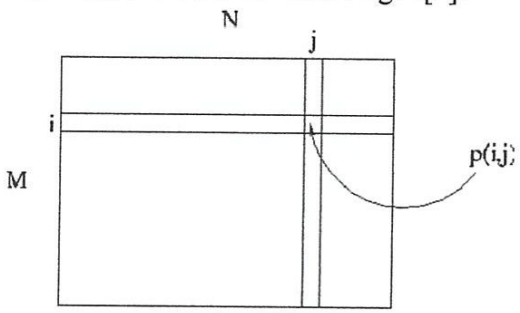


Figure III.1 représentation d'une image par une matrice

p(i,j) est le niveau de gris du pixel de coordonnées ligne i et colonne j dans l'image. p(i,j) Î[0, Ng]. Les valeurs des niveaux de gris sont des entiers.

III.3. Prétraitement

Les données brutes issues de capteurs sont les représentations initiales de la réalité à partir desquelles des traitements permettent de construire les données qui seront utilisées pour la reconnaissance. Les entrées brutes sont bruitées, elles contiennent des informations parasites, et elles n'explicitent pas les informations utiles pour la reconnaissance.

III.3.1. Image en niveaux de gris

Une image ne niveaux de gris autorise un dégradé de gris entre le noir et le blanc. En général, on code le niveau de gris sur un octet (8 bits) soit 256 nuances de dégradé. L'expression de la valeur du niveau de gris avec Ng = 256 devient: p(i,j) Î [0, 255].

Opération 01 :

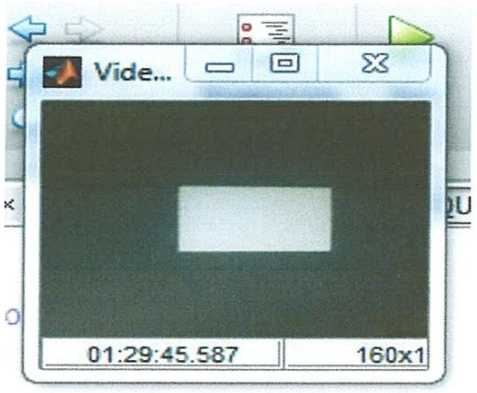


image réel

RECT3 =

102	102	101	102	102	102	102	102	102	102
102	101	102	102	101	102	102	101	102	102
102	102	101	102	102	101	102	102	102	102
102	103	104	107	108	107	106	105	103	103
103	102	115	135	136	132	128	126	109	103
104	104	119	140	141	138	135	133	112	103
107	106	122	142	141	141	139	137	115	104
108	107	123	142	141	140	138	135	114	104
109	110	113	116	115	114	112	109	107	105
111	110	110	110	110	109	108	106	106	106

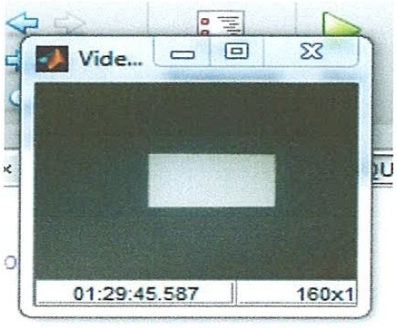
niveau de gris

Figure III.2 le codage par niveau de gris

III.3.2. Image binaire

Une image binaire est une image MxN où chaque point peut prendre uniquement la valeur 0 ou 1. Les pixels sont noirs (0) ou blancs (1). Le niveau de gris est codé sur un bit (Binary digIT). Dans ce cas, Ng >= seuil donc p(i,j) = 1 sinon p(i,j) = 0.

Opération 02 :



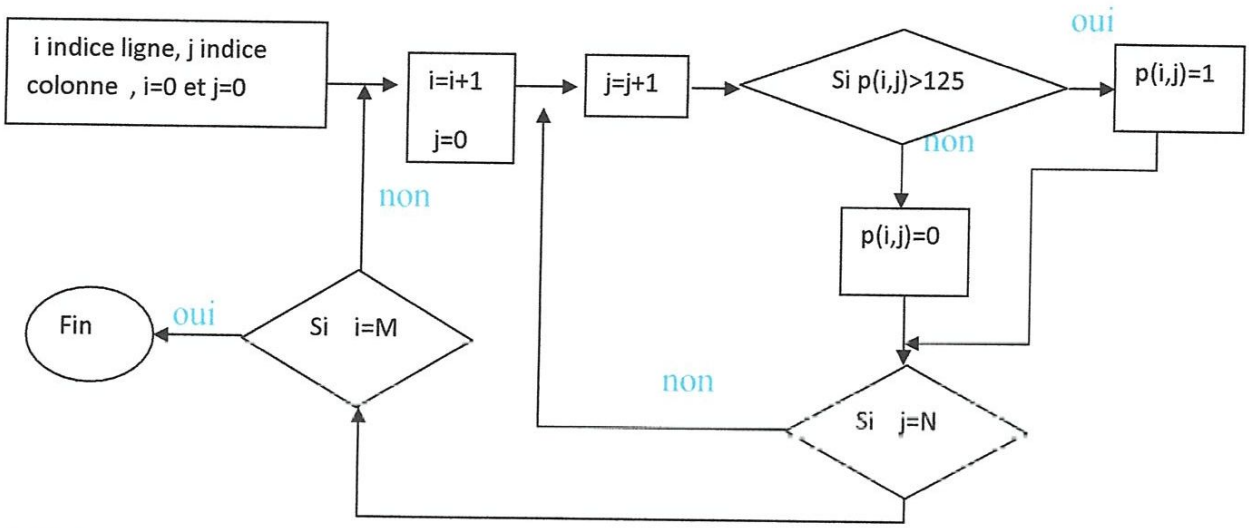
RECT2 =

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Figure III.3 Image binaire

Algorithme transformation niveau de gris vers le binaire (voir figure III.1)

L'image est une matrice MxN



III.3.3. Opérations géométriques

Les opérations géométriques classiques sont permises avec la boîte à outils de traitement d'images: rotation, changement de taille, découpage...

Opération géométrique 01 : changement de taille

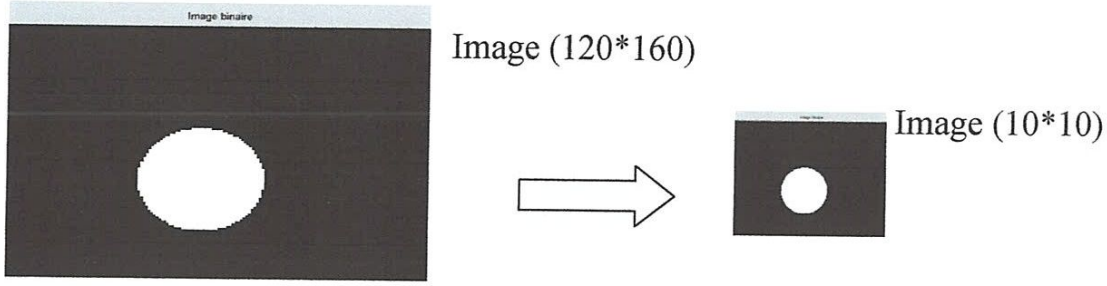


Figure III.4 Le redimensionnement d'image

Opération géométrique 02 : Centrage

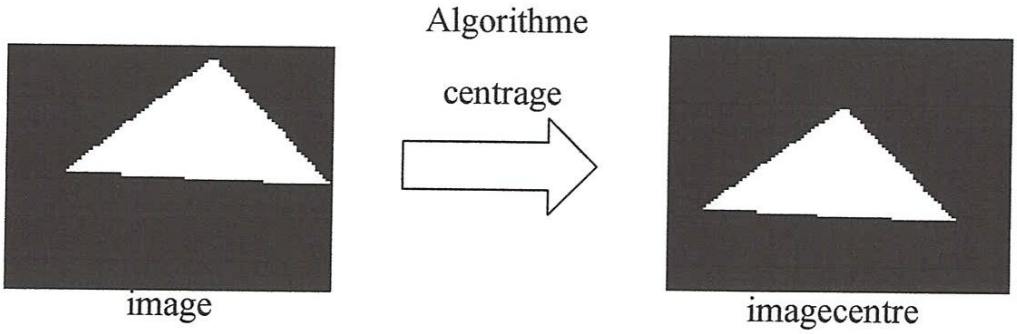
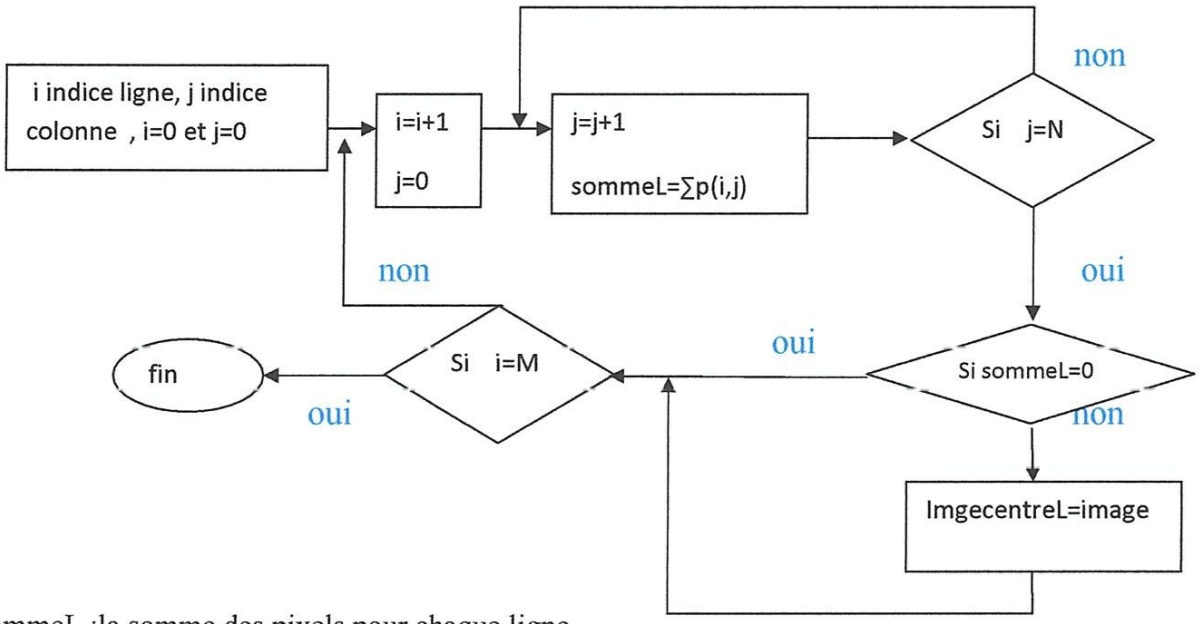


Figure III.5.Centrage

Algorithme centrage : on a l'image binaire MxN



sommeL :la somme des pixels pour chaque ligne .

ImgecentreL: la nouvelle image .

On reprend la même procédure pour les colonnes .

Opération géométrique 03 : découpage

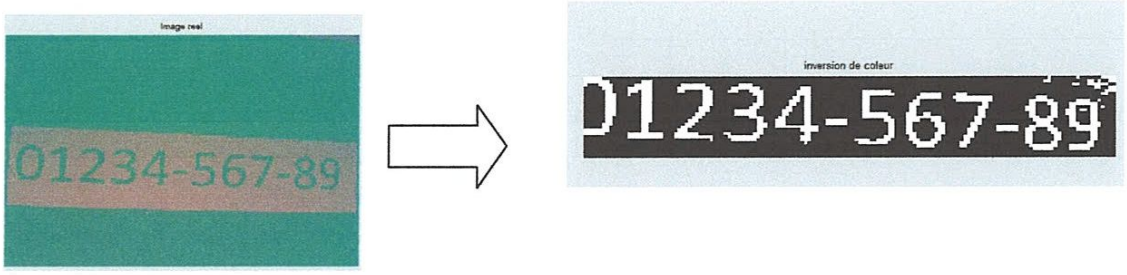


Image avant traitement et découpage

Image après traitement et découpage

Figure III.5 .Segmentation (découpage)

III.3.4. Détection de contours

La détection de contours permet de repérer les différents objets qui constituent la scène de l'image.

Il existe de nombreuses méthodes pour trouver les contours des objets, la plupart sont basées sur les dérivées premières et secondes de l'image.

III.3.4. 1.Le filtre de Sobel

Le principe de ce filtre est que l'opérateur calcule le gradient de l'intensité de chaque pixel. Ceci indique la direction de la plus forte variation du clair au sombre, ainsi que le taux de changement dans cette direction. On connaît alors les points de changement soudain de luminosité, correspondant probablement à des bords, ainsi que l'orientation de ces bords.

L'opérateur utilise des matrices de convolution. La matrice (généralement de taille 3×3) subit une convolution avec l'image pour calculer des approximations des dérivées horizontale et verticale. Soit A l'image source, Gx et Gy deux images qui en chaque point contiennent des approximations respectivement de la dérivée horizontale et verticale de chaque point. Ces images sont calculées comme suit:

$$\mathbf{G_x} = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{et} \quad \mathbf{G_y} = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

En chaque point, les approximations des gradients horizontaux et verticaux peuvent être combinées comme suit pour obtenir une approximation de la norme du gradient:

$$\mathbf{G} = \sqrt{\mathbf{G_x}^2 + \mathbf{G_y}^2}$$

Opération filtrage : filtre de Sobel

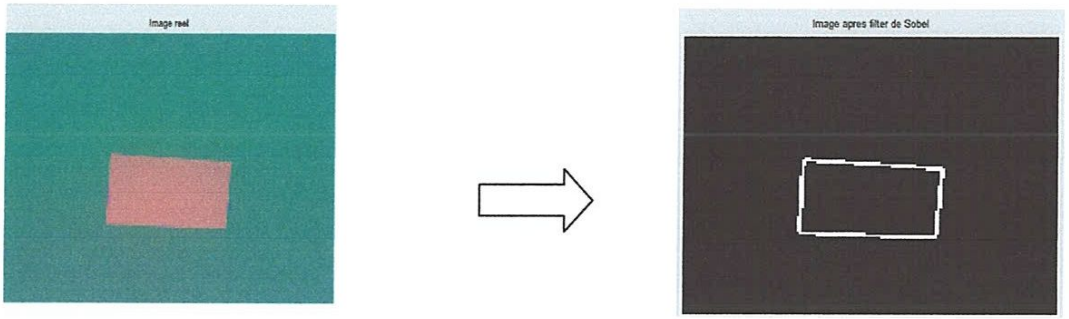


Image réel image après détection de contour

Figure III.6 détection de contour (filtre de Sobel)

III.3.4. 2.Histogramme - seuillage

L'histogramme d'une image donne la répartition de ses niveaux de gris. Ainsipour une image qui possède 256 niveaux de gris, l'histogramme représente le niveau de gris en fonction du nombre de pixels à ce niveau de gris dans l'image.

»img = imread('rectangle.tif');

» histo = imhist(img,256);

» figure;plot(histo);

On sait que les niveaux de gris à zéro correspondent au noir et que les niveaux

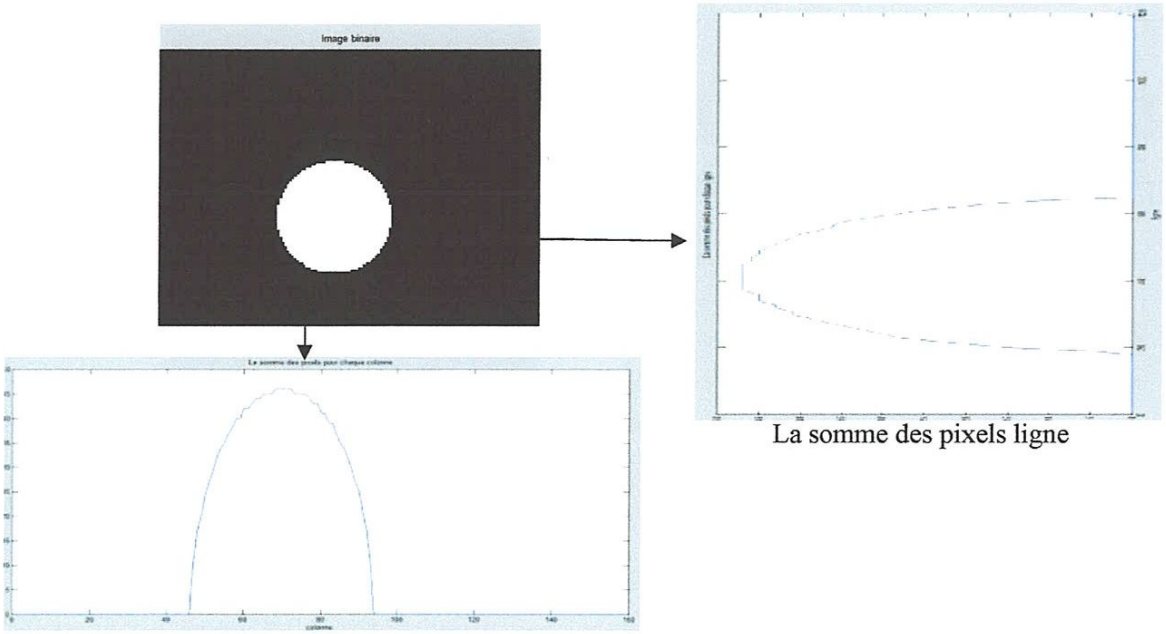
de gris à 1 indiquent le blanc. L'histogramme donne donc une excellente idée de laséparation entre quelque chose qui est clair et quelque chose qui est foncé dans

l'image. Typiquement, une utilisation de ce fait est le seuillage d'une image, ce termedésigne la définition d'un seuil au-dessus ou en-dessous duquel on va garder certaines valeurs de niveaux de gris.

III.4. L'extraction des propriétés

L'extraction des propriétés d'objet et de les exprimer sous forme numérique ou symbolique. On a inge binaire de dimension (i,j).

Application 01 : on prend la somme de chaque ligne divisé par le nombre de colonne et on la remplit dans un vecteur ligne la même chose pour les colonnes .



La somme des pixels ligne

La somme des pixels colonne

Figure III.7 Propriétés d'image

Application 02 : on prend les deux vecteurs diagonale pour chaque image .

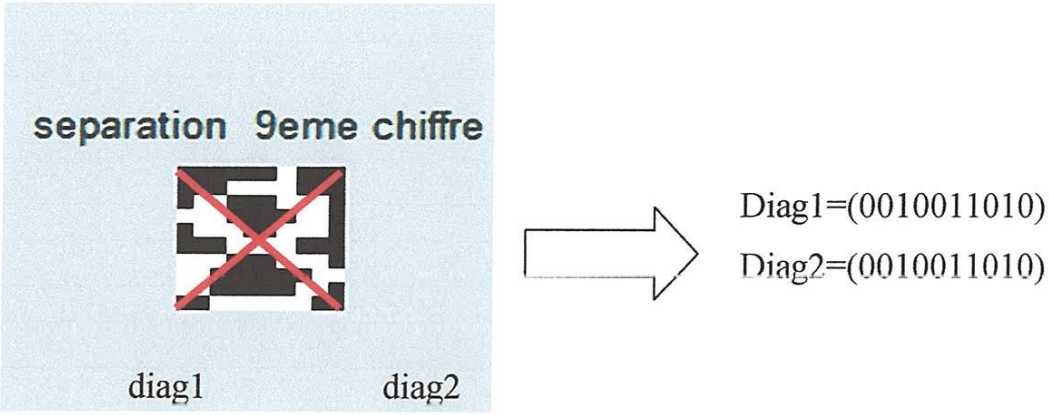
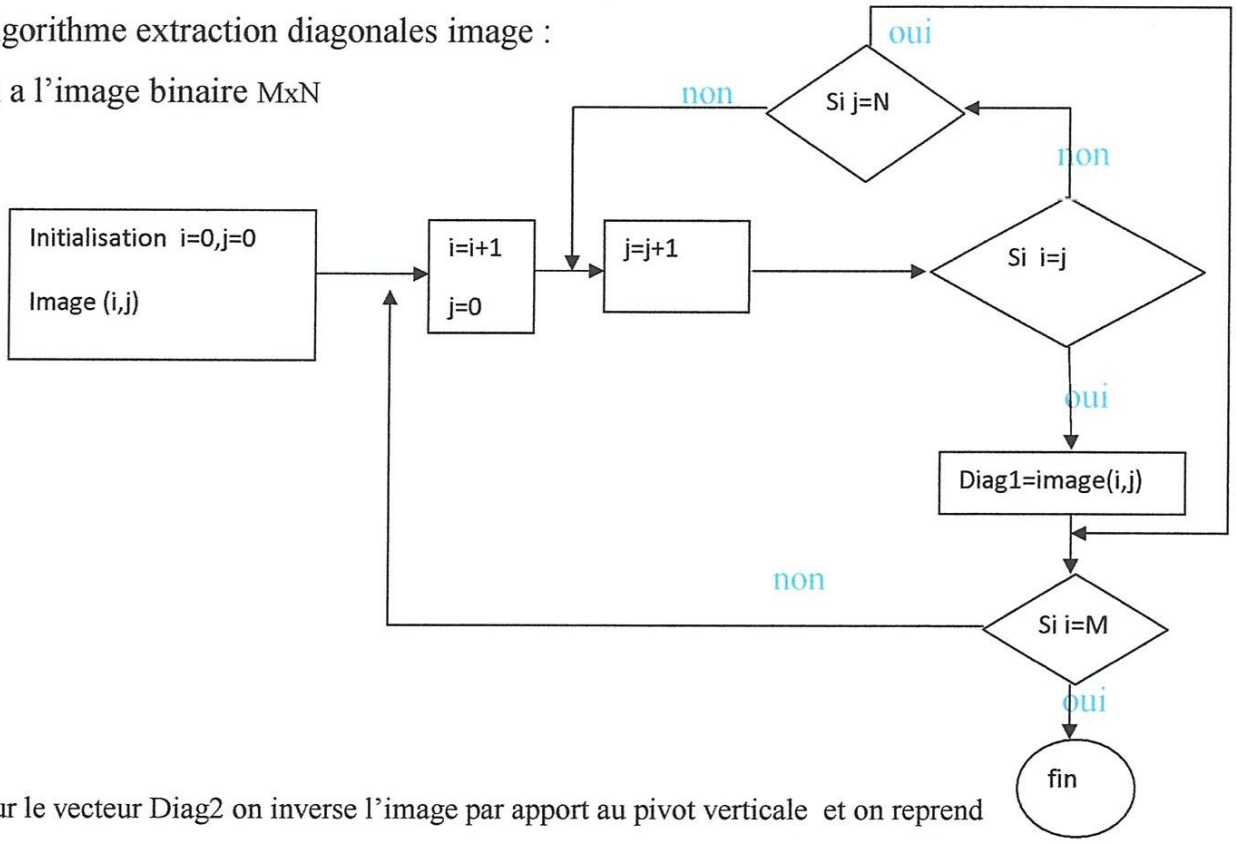


Figure III.8 Extraire les diagonales d'image

Algorithme extraction diagonales image :

on a l'image binaire MxN

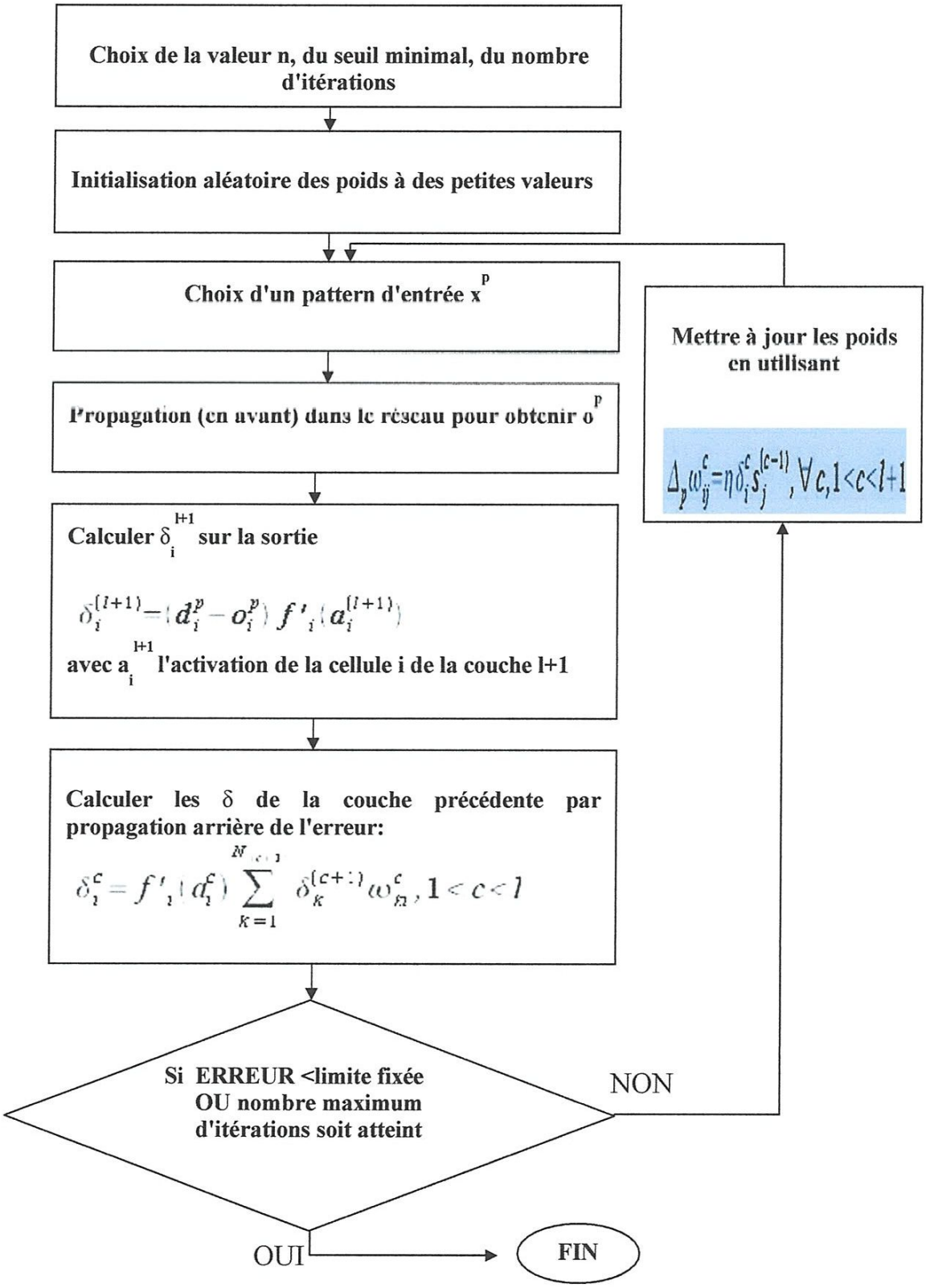


Pour le vecteur Diag2 on inverse l'image par apport au pivot verticale et on reprend la même procédure.

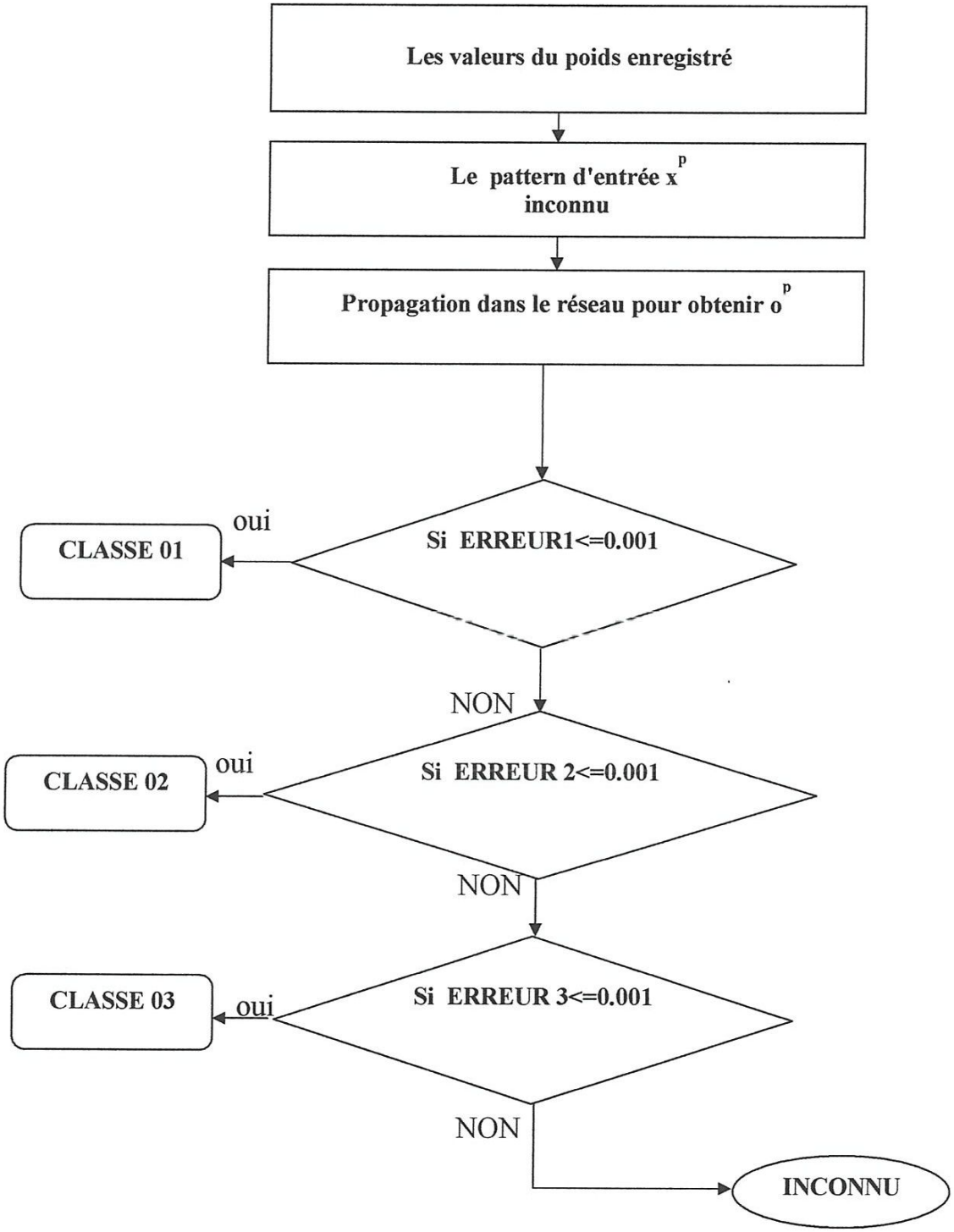
III.5. L'apprentissage

Dans le cas d'apprentissage il s'agit de fournir au système un ensemble de formes dont on connaît les classes.

III.5. 1Algorithme du perceptron multicouches [1]



III.6. Algorithme du classification



III.7. Les applications

Application 01 : les courbe de minimisation d'erreur :

Nombre des neurones cachées :3

Itération :100

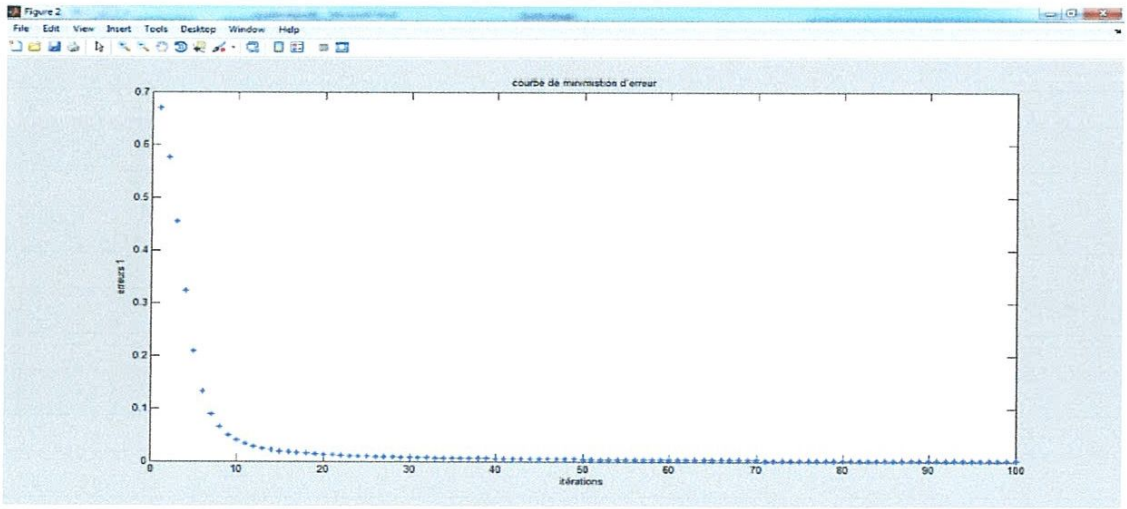


Figure III.9 courbe de minimisation d'erreur triangle(erreur min=0.01)

Nombre des neurones :6

Itération :100

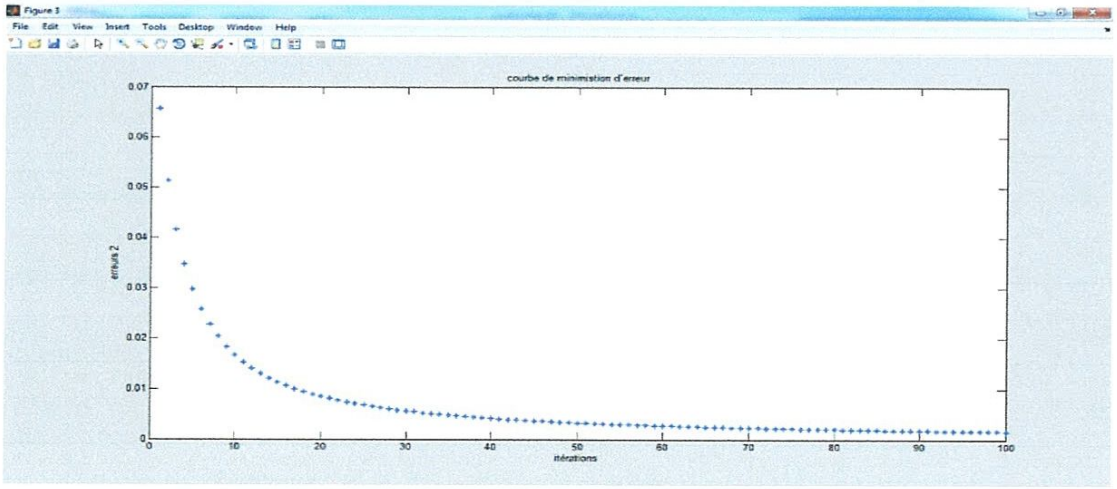


Figure III.10 courbe de minimisation d'erreur rectangle(erreur min=0.003)

Application 02 : les courbe de minimisation d'erreur :

Nombre des neurones :10

Itération :200

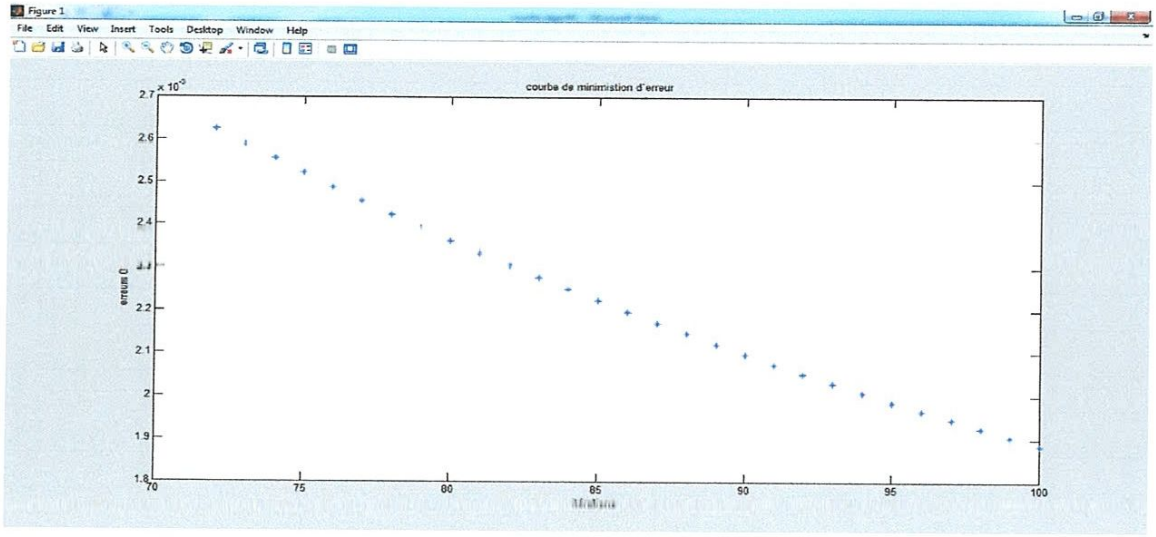


Figure III.11 courbe de minimisation d'erreur cercle(erreur min=0.0016)

On peut dire que plus le nombre d'itération et de neurone du couche cachée augmente plus la minimisation de l'erreur sera plus bonne.

III.5. test et décision

Application 01(formes géométrique): image réelle obtenue à partir du webcam

-Test1

Avant classification :

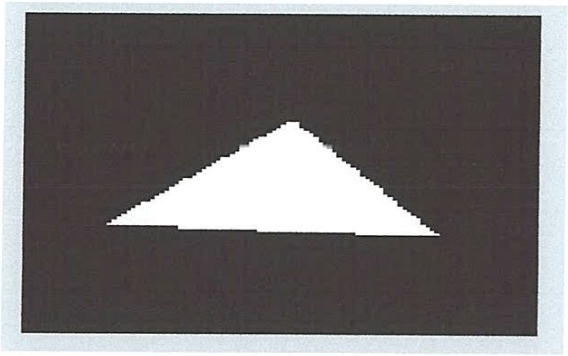


Figure III.12 Image inconnu avant classification

Après classification :

Figure III.13 Image=triangle après classification

La figure III.13 représente le résultat de classification d'une forme géométrique selon les trois classe triangle, cercle et rectangle.

-Test 2

Avant classification :

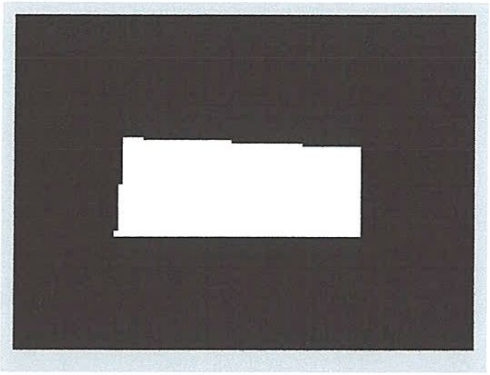


Figure III.14 Image inconnu avant classification

Après classification :

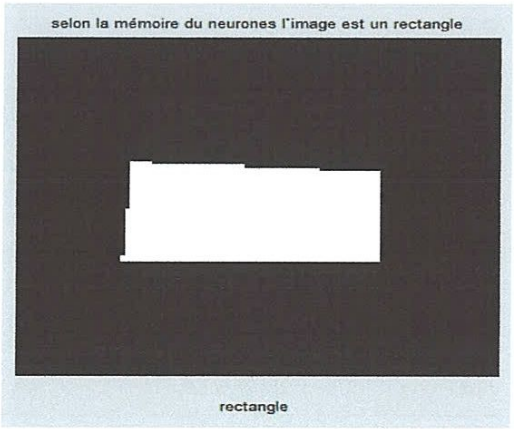


Figure III.15 Image=rectangle après classification

La figure III.15 représente le résultat de classification d'une forme géométrique selon les trois classe triangle, cercle et rectangle.

-Test3

Avant classification :

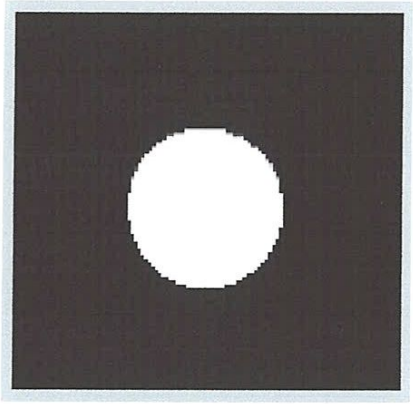


Figure III.16 Image inconnu avant classification

Après classification :

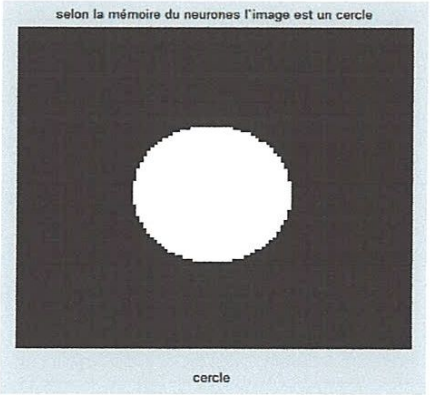


Figure III.17 Image=cercle Après classification

La figure III.17 représente le résultat de classification d'une forme géométrique selon les trois classe triangle, cercle et rectangle.

Application 02(chiffres):

Etape01 :Image réelle obtenue à partir du webcam

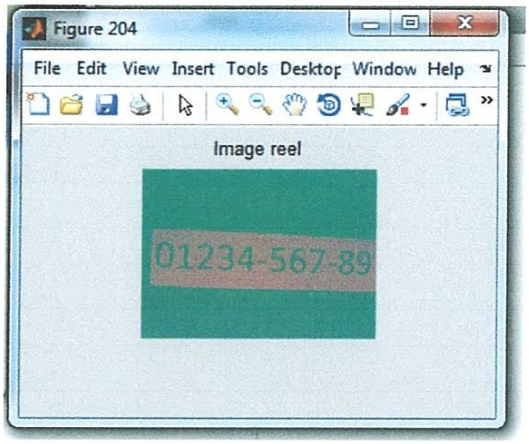


Figure III.18 .Image réel

Etape02 : Prétraitement (filtrage) + extraction des caractéristiques

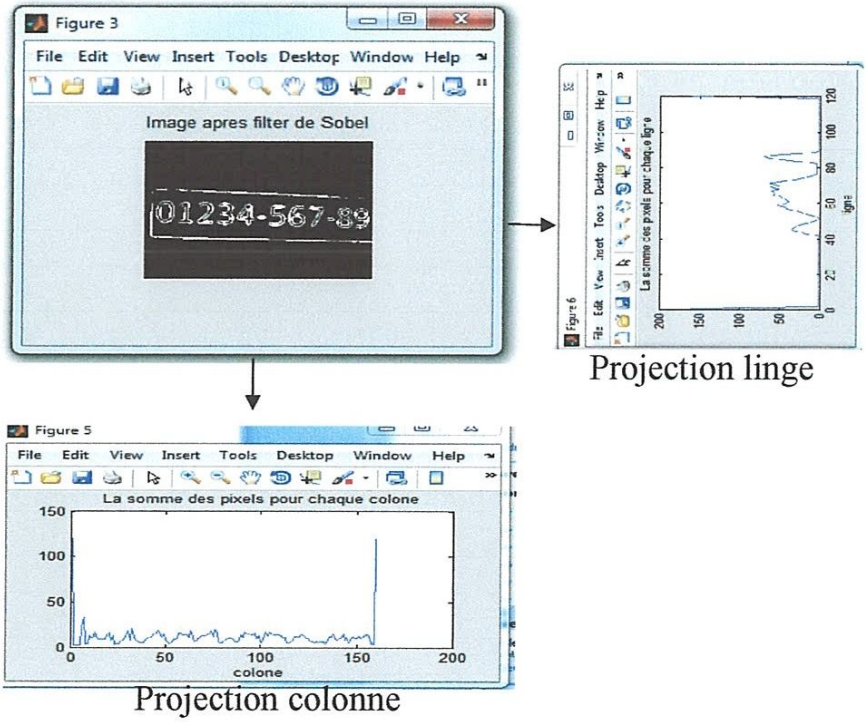


Figure III.19 . Prétraitement (filtrage) + extraction des caractéristiques

Etape03 : Segmentation (découpage)

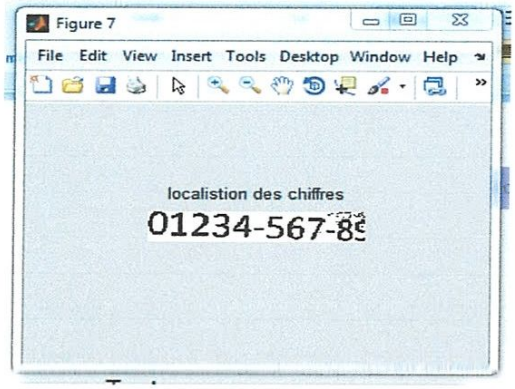


Figure III 20 Segmentation (découpage)

Etape04 : Inversion de couleur

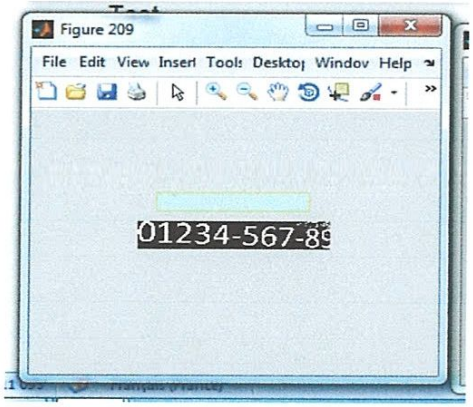


Figure III.21. Inversion de couleur

Etape05 : Segmentation (découpage ou séparation entre les chiffres)

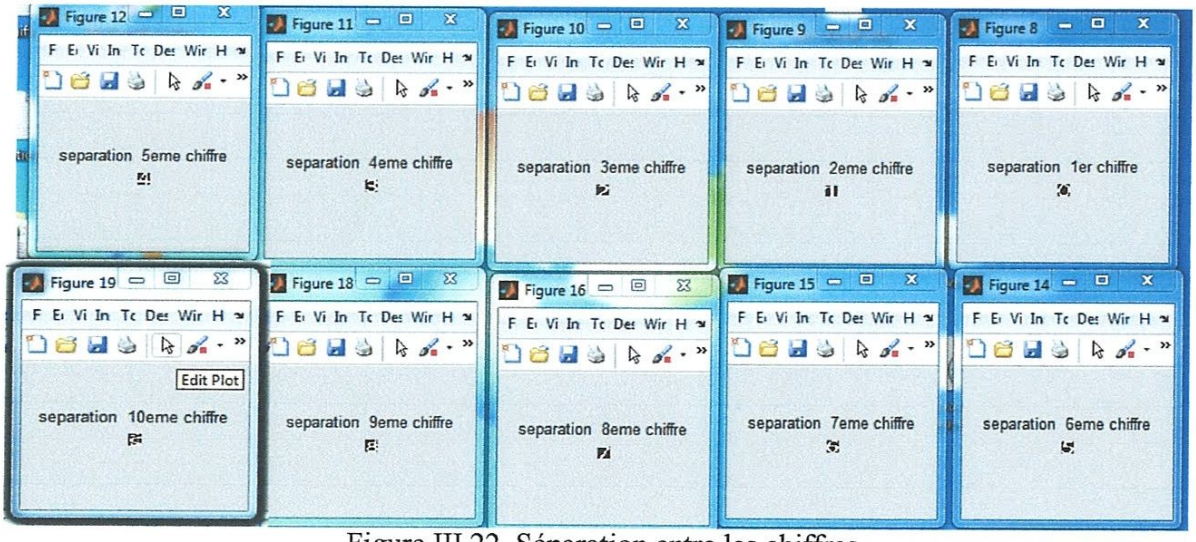


Figure III.22. Séparation entre les chiffres

-Test

Avant classification :

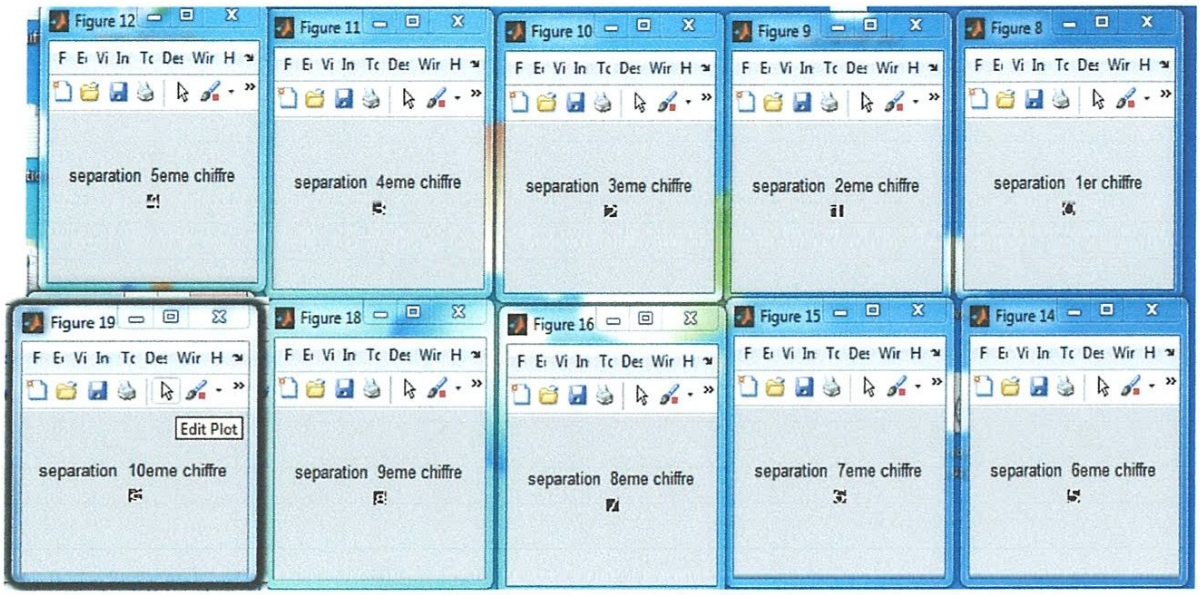


Figure III.23. Test avant classification

Après classification :

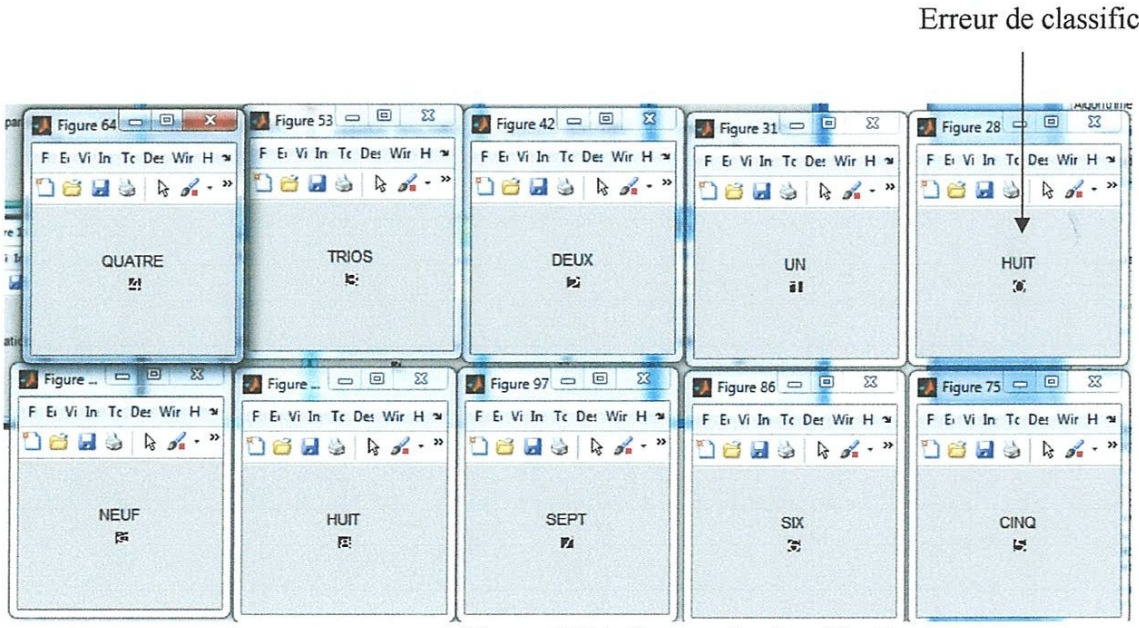


Figure III.24. Test après classification

Dans la deuxième application on doit prendre une image immatricule et puis la traiter selon la procédure de filtrage et la découpage du zone porteuse des chiffres et après la segmentation (séparation des chiffres) ,chaque image représente un seul chiffre donc chaque image sera classifier selon les dix classe (0…….9) .

La figure III .24 représente les résultats de classification d'une image immatricule modifier, cette image est utilisée dans la base d'apprentissage.

- Les données utiliser pour l'application :

Les paramètres	Les valeurs
Nombre image d'apprentissage	10
Nombre image test (non utiliser dans la base d'apprentissage)	40
Nombre neurones cachées	30
Taux d'apprentissage	0.8
Nombre d'itération	1000
Nombre neurones de sortie	3
Nombre neurones d'entrée	20

Tableau III.1 données utiliser pour l'application

- Les résultats pour application 01 :

Nombre de neurones cachées	Nombre d'itérations	Taux reconnaissance Pour rectangle(%)	Taux pour Cercle (%)	Taux pour Triangle (%)
3	200	60%	55%	48%
5	500	65%	60%	56%
10	800	70%	63%	60%
15	900	72%	78%	66%
20	1000	80%	80%	72%
30	1000	95%	92%	86%

Tableau III.2 Résultat pour l'application 01

➢ Les résultats pour application 02 : taux = pourcentage de reconnaissance

Nombre de neurones cachées	Nombre d'itérations	Taux Pour zero (%)	Taux pour Un (%)	Taux pour Deux (%)	Taux pour trois (%)	Taux pour quatre (%)	Taux pour cinq (%)	Taux pour six (%)	Taux pour sept (%)	Taux pour Huit (%)	Taux pour Neuf (%)
3	200	30	32	29	35	40	38	41	50	36	37
5	500	42	38	40	45	35	44	39	48	32	46
10	800	49	42	45	48	39	45	44	50	36	48
15	900	55	49	51	56	45	50	52	58	43	57
20	1000	66	59	60	63	55	61	69	64	57	63
30	1000	70	67	72	77	73	79	72	69	71	68

Tableau III.2 Résultat pour l'application 01

-dans le cas de l'application 01 on peut remarquer que plus on augmente le nombre de neurone (pas toujours vrai) plus le taux de reconnaissance augmente, pour le nombre d'itération la relation proportionnelle est toujours vrai.

-pour l'application02 les même remarques sont distingué, mais avec dix classes non linéairement séparable ,un faible nombre d'exemple d'apprentissage et peut-être le choix non convenable d'exemple d'apprentissage tout ces critères nous a donné un faible taux d'aprentissage.

Les solution proposer :

- Augmentation du nombre d'exemple d'apprentissage.
- Utilisation de deux ou trois couches cachés
- Utilisation du plusieurs vecteurs du caractéristique dans la phase d'apprentissage et de classification.

Conclusion

L'algorithme de reconnaissance des formes par réseaux de neurone offre une multitude d'application dans le monde réel exemple classification des produits selon leur forme dans l'industrielle, identification des personnes a partir de reconnaissance d'immatricule automobile.

L'Objectif de ce travail est de construire un algorithme qui peut reconnaitre les formes géométrique dans l'application 01 et les chiffre dans un immatricule automobile dans l'application 02.

Notre travail (la reconnaissance des formes par PMC) donne des résultats très satisfaisants sur les formes géométriques (plus de 90% de réussite sur les images de bonne qualité).

Le taux de reconnaissance est plus faible (70% de réussite sur les images de bonne qualité) dans la deuxième application puisque le nombre des classes augmente, l'angle de capture image a une grande influence sur l'étape de segmentation donc un échec de séparation entre les chiffre s'explique par l'échec de reconnaissance.

Les étapes de prétraitement et d'extraction des caractéristiques sont les plus importants critères pour la réussite d'algorithme apprentissage et décision.

On a bénéficié beaucoup de connaissance dans ce travail sur la reconnaissance de forme par réseaux de neurones la méthode la plus utilisé dans l'intelligence artificielle et la plus intéressante dans les nouvelles recherches.

Bibliographie

[1]M.mokhtari,M.Marie (1998).Applications de MATLAB 5 et SIMULINK 2

[2]Buessler J-L, (1999). Architectures neuro-mimétiques modulaires, Application à l'asservissement visuel de systèmes robotiques, thèse de l'Université de Haute-Alsace, U.F.R. Des Sciences et Techniques

[3]Blayo F. et Verleysen M., (1996). Que sais-je ? Les réseaux de neurones, Presse Universitaires de France

d'Espagnat B., (1998). Physique et réalité, Diderot Editeur

[4]http://www.sm.u-brdeaux2.fr/~corsini/Cours/mainRNA/mainRNA.html

[5]http://www.grapa.univ-lille.fr/polys/apprentissage/

[6]http://pcrochat.online.fr/webus/tutorial/