MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE UNIVERSITE MOULOUD MAMMERI, TIZI-OUZOU

FACULTE DE GENIE ELECTRIQUE ET DE L'INFORMATIQUE
DEPARTEMENT D'ELECTRONIQUE

Mémoire de fin d'études

En vue de l'obtention

du Diplôme Master en Electronique

Option Télécommunications et Réseaux

Thème :

RECONNAISSANCE DE CARACTERES A BASE DES RESEAUX DE NEURONES

Proposé et encadré par:

M^{r}: HADDAB.S

Réalisé et présenté par :

M^{elle} : SAIKI Hakima

M^{elle}: OULTAF Kahina

Promotion: 2012-2013

Remerciement

Nous remercions dieu de nous avoir donné la force et la volonté pour mener à bien en notre travail.

Nous tenons à remercier notre promoteur et encadreur Monsieur S. HADDAB pour son aide, conseils, orientation et sa confiance qui nous ont été d'une grande importante.

Nous tenons à remercier, tous ceux qui nous ont enseigné durant toutes nos études.

Nous tenons à remercier tous ce qui ont contribué de prés ou de loin pour la réalisation de ce travail

D édicace

DEDICACE

Je dédie ce mémoire

A mes parents,

Mon frère et mes sœurs,

Mon amie et binôme Hakima.

Kahina.O

Sommaire

Chapitre III : Réalisation d'un classifieur neuronal

Table des figures

Chapitre I:

Chapitre III:

Table des tableaux

Introduction générale

Depuis son invention, l'écriture reste un moyen de communication privilégié entre les êtres humains. Bien que l'imprimerie puis l'informatique aient permis son automatisation, l'écriture est loin d'avoir disparue de notre société et les individus émettent et reçoivent de grandes quantités de documents.

Ce moyen d'échange et de communication n'est pas resté au service de la littérature mais a touché plusieurs domaines aussi nombreux que variés tel que les sciences, les mathématiques et l'informatique. Aujourd'hui, avec les technologies modernes, des systèmes de reconnaissance de l'écriture ont été développés afin de faciliter le traitement de données imprimées ou manuscrites.

Dans ce travail, on s'intéresse à la réalisation d'un système de reconnaissance de lettres à l'aide de réseaux de neurones. Nous réaliserons un classifieur à base de réseau de neurones pour reconnaître, dans un premier temps, trois lettres de l'alphabet, puis nous nous intéresserons au cas de la généralisation à un plus grand nombre de caractères.

Ce travail se compose de trois chapitres. Le premier est consacré à une présentation générale des systèmes de reconnaissance de caractères, aux composantes principales de ces systèmes ainsi qu'à un aperçu sur les différentes phases de traitement de données.

Les réseaux de neurones et leurs différents types sont introduits dans le deuxième chapitre. Nous nous intéresserons, particulièrement, à l'algorithme de rétro-propagation qui sera utilisé dans notre cas.

Dans le troisième chapitre, nous présentons la réalisation d'un classifieur neuronal pour reconnaître trois lettres puis nous généraliserons au cas de plusieurs lettres.

Introduction:

La reconnaissance de caractères fait partie d'un domaine plus vaste du traitement de signal qui est la reconnaissance de forme. L'objectif visé consiste, à partir d'une source pouvant être manuscrite ou numérique, à identifier chaque caractère (lettres, chiffres ou symboles) afin d'en extraire l'information qu'il contient.

Nous allons, dans ce qui suit, décrire le principe général et les différentes étapes de la reconnaissance de caractères ainsi que les contraintes rencontrées en fonction de la nature des caractères source.

I. Objectifs et domaines d'application:

La reconnaissance automatique de l'écriture a pour objectif de transformer un texte écrit ou une image en une représentation compréhensible par un ordinateur et facilement reproductible par un logiciel de traitement en vue de l'archivage, la modification, la réutilisation ou la transmission de l'information que ces documents contiennent.

Depuis son invention, l'écriture reste un moyen de communication privilégié entre les êtres humains. Bien que l'imprimerie puis l'informatique aient permis son automatisation, l'écriture est loin d'avoir disparue de notre société et les individus émettent et reçoivent de grandes quantités de documents. Le traitement de masse de documents apparaît alors incontournable et de nombreux travaux de recherche ont été effectués dans ce cadre. Ainsi les premiers OCR (Optical Caracter Recognition) ont fait leur apparition et les systèmes de lecture automatique d'adresses postales utilisées pour le tri de courrier ont été mis en œuvre. La lecture automatique de l'écriture manuscrite dans les formulaires s'est également développée grâce à des moyens de calcul plus puissants.

Aujourd'hui, les applications de la reconnaissance de caractères sont nombreuses et variées et connaissent ainsi plusieurs domaines d'activité parmi lesquels on peut citer les suivants :

- Les banques et les assurances pour l'authentification de chèques bancaires et la vérification de clauses de contrats.
- La poste pour la lecture des adresses et le tri automatique du courrier.
- Les télécommunications pour l'échange de fichiers informatiques à distance.
- La police et la sécurité pour la reconnaissance de numéros minéralogiques pour le contrôle routier, l'authentification et l'identification de manuscrits et l'identification du scripteur.

- Les affaires et l'industrie pour la gestion des stocks et la reconnaissance de documents techniques.
- La bureautique pour l'indexation et l'archivage automatique de documents, et pour la publication électronique.
- L'administration pour la reconnaissance de plans cartographiques et la lecture automatique de documents administratifs.
- La transcription de documents manuscrits anciens.

Le principe et les techniques à utiliser dépendent, essentiellement, de la nature initiale des caractères à identifier. Ces caractères peuvent être manuscrits, imprimés ou numérisés.

II. Les différents types d'écritures:

La reconnaissance de caractères est d'autant plus complexe que ces derniers peuvent se présenter sous différentes formes (imprimés, manuscrits). Les caractères imprimés sont moins difficiles à identifier dans la mesure où ils se présentent sous des formes standardisées avec des normes prédéfinies. Les caractères manuscrits se présentent sous des formes très diverses qui résultent des habitudes de chaque scripteur et qui diffèrent d'une personne à une autre.

II.1. Caractères manuscrits:

La reconnaissance automatique de l'écriture naturelle (caractères manuscrits) est une opération particulièrement complexe et ardue. A ce jour, il n'existe aucun algorithme capable de traiter, de façon fiable, l'intégralité des tracés qui la constituent: Un texte peut contenir, à la fois, des chiffres, des lettres minuscules, majuscules des caractères bâtons et des symboles. Par conséquent, il est capital de savoir extraire et séparer ces entités afin de les traiter grâce aux algorithmes spécifiques à chacune des catégories d'écritures qu'ils constituent.

D'une manière générale, la complexité d'un système de reconnaissance de caractères manuscrits s'évalue suivant trois critères: Style d'écriture et contraintes, nombre de scripteurs et taille de vocabulaire.

II.1.1. Styles d'écriture et contraintes:

Chaque individu écrit d'une façon qui lui est propre. Son écriture est différente de celle des autres même si l'alphabet de base est le même.

Toute la difficulté de la reconnaissance de l'écriture manuscrite réside, en majeure partie, dans cette grande variété d'écriture: Tracés différents, enchainements différents entre les lettres à l'intérieur d'un mot, etc.

Les variabilités dans l'écriture peuvent être dues à des contraintes interne et externe provenant des habitudes propres au scripteur.

La présentation de Tappert [1] indique que le texte peut subir deux types de contraintes:

- **Contraintes externes**: conduisant à une écriture précasée, zonée, guidée par cadre (exemple formulaire).
- **Contraintes internes:** provenant des habitudes propres à chaque scripteur et conduisant à une écriture détachée, groupée, scripte (bâton), purement cursive ou mixte. Il est évident que l'écriture détachée reste la plus facile à comprendre du fait de la séparation quasi immédiate des lettres; au contraire, l'écriture cursive nécessite plus d'efforts du fait de l'ambiguïté des limites entre les lettres.

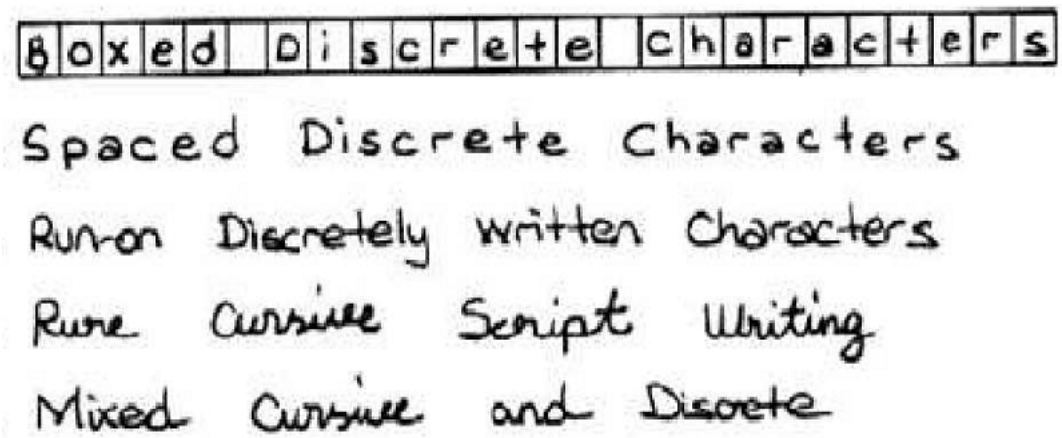


Figure. II.1. Classification de l'écriture manuscrite selon Tappert[1]

Cette figure représente les différents types d'écriture qui sont classées par Tappert [1]

II.1.2. Taille du vocabulaire:

La taille du vocabulaire est également un facteur influant sur les performances d'un système de reconnaissance ainsi que sur la méthode adoptée.

Alors que pour des tâches nécessitant un petit vocabulaire (une trentaine de mots dans le cas de la reconnaissance de montants de chèques), on cherche à reconnaitre le mot comme une entité à part entière. Pour des tâches de très grand vocabulaire on cherchera plutôt à reconnaitre les lettres au sein du mot.

Il est habituel de considérer que:

- Un petit vocabulaire est constitué d'une dizaine de mot.
- Un vocabulaire de taille moyenne comporte une centaine de mots.
- Un grand vocabulaire comporte des milliers de mots,
- Enfin, un très grand vocabulaire renvoie à plus d'une dizaine de milliers de mots.

II.1.3. Nombre de scripteurs:

La difficulté de reconnaissance augmente avec le nombre de scripteurs. En effet, plus il ya de scripteurs (droitier/gaucher, enfant/adulte, différentes professions), plus il ya de styles d'écritures différentes comme représenté dans la figure II.2. suivante :

Figure II.2. Le mot "**Rome**" écrit par différents scripteurs

La difficulté de traitement croît avec le nombre de scripteurs. Un premier niveau est la reconnaissance mono-scripteur avec apprentissage de l'écriture propre à l'utilisateur considéré. Un niveau plus général est atteint par les systèmes multi-scripteurs qui sont capables de reconnaitre l'écriture de n'importe quel scripteur.

La figure II.3 suivante représente les différents degrés de difficulté rencontrés dans la reconnaissance de caractère manuscrits. En s'éloignant du centre des axes le degré de difficulté augmente selon son type.

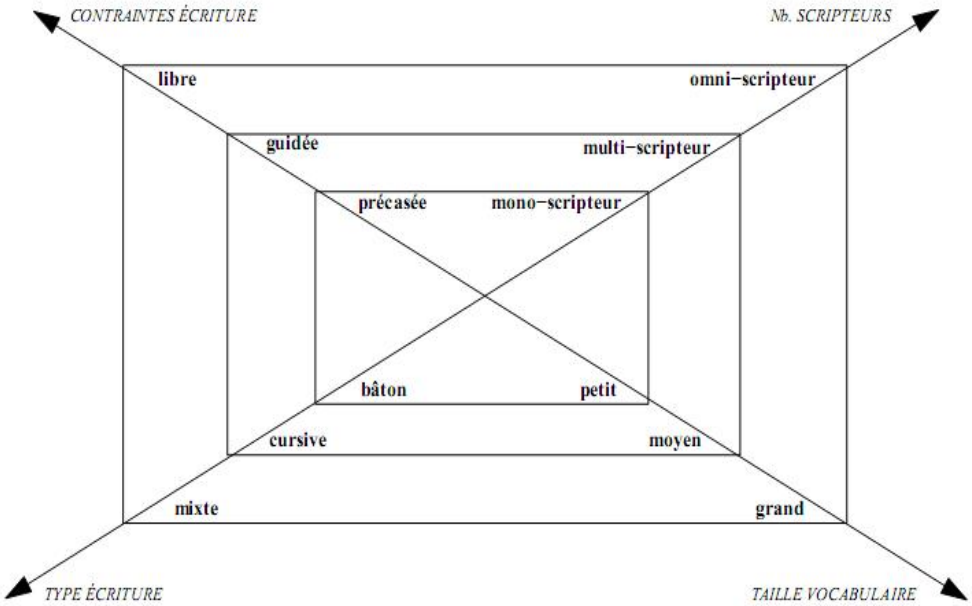


Figure II.3. La difficulté de l'écriture manuscrite

II.2.Caractères imprimés:

Même s'il semble à priori, que la reconnaissance de caractères imprimés est plus simple la grande variété de paramètres fait que le traitement reste complexe. Les mots possèdent une infinité de représentations qui dépendent des polices de caractères, de styles (gras, italique, souligné, ombré, etc.) et des mises en page différentes et complexes. Les opérations à effectuer et les résultats peuvent varier notablement.

On a trois familles de systèmes de reconnaissance optique de caractères (OCR), à savoir les systèmes de reconnaissances mono-fonte, multi-fonte ou omni-fonte.

- Un système d'OCR mono-fonte reconnaît les caractères d'une fonte bien déterminée.
- Les systèmes d'OCR multi-fonte permettent la reconnaissance des caractères de plusieurs fontes, toutes ayant, en principe, fait l'objet d'un apprentissage unique.
- Les systèmes d'OCR omni-fonte font abstraction de l'information sur la fonte, dans la mesure où ils sont capables de reconnaître des caractères de n'importe quelle fonte, et n'importe quelle taille. Cependant les méthodes d'OCR omni-fonte sont incapables d'identifier les fontes, d'où leur intérêt limité dans les systèmes de reconnaissance structurelle de documents.

Dans de telles applications, il faut compléter se type d'OCR par un module d'identification de la fonte.

III. Reconnaissance de caractères statiques ou dynamiques:

Dans le domaine de la reconnaissance de l'écriture, deux cas distincts sont considérés. Il s'agit, d'une part, de la reconnaissance statique dite encore « hors-ligne », où les données disponibles résultent de la numérisation d'un document papier contenant de l'écriture obtenue par un scanner ou une caméra. Dans ce cas, on dispose d'une image binaire ou en niveaux de gris, ayant perdue toute information temporelle sur l'ordre des points, elle est privée de l'information spatio-temporelle, et plus délicate, elle concerne à priori, tous les documents numérisés (adresse postal, chèques bancaires, formulaire structurés ou manuscrite quelques). On dispose alors d'une image numérisée où il va falloir avant même d'aborder l'étape de reconnaissance dynamique, localiser et extraire le signal d'écriture du reste de l'image. De plus, ce mode introduit une difficulté supplémentaire relative à la variabilité du tracé en épaisseur et en connectivité, nécessitant des techniques de prétraitement.

L'autre domaine est celui de la reconnaissance dynamique, appelée encore « en ligne ». Cette fois le signal d'écriture est directement enregistré pendant sa production grâce à un dispositif spécifique (assistant personnel, tablette graphique) qui permet d'échantillonner la trajectoire du stylet fournissant ainsi les coordonnées (x, y) ordonnées dans le temps dans ce cas, la donnée est de type signal et l'approche doit tirer profit de la représentation temporelle. La reconnaissance en ligne s'effectue à partir d'une acquisition spatio-temporelle des mots cursifs sur une tablette électronique.

IV. Système de reconnaissance: Le classifieur

L'outil de base d'un système de reconnaissance est le classifieur, qui a pour but, à partir des différents caractères de la source, à ranger chacun d'entre eux dans la classe qui lui correspond. Si, par exemple, nous devons distinguer entre trois caractères A, B, ou C, nous définissons trois classes qui correspondent, chacune, à l'une des lettres. Les détails du système de classification fait l'objet du chapitre suivant.

Toutefois, un certain nombre d'opérations préalables est nécessaire pour pouvoir assurer la classification des caractères. Nous devons, ainsi, procéder à un prétraitement dans le but de faciliter l'extraction des caractéristiques des grandeurs à classifier.

Les techniques de reconnaissance de caractères ont beaucoup évoluées depuis la commercialisation des premiers OCR en particulier grâce à l'apparition des moyens de calculs puissants. Nous sommes ainsi passés des méthodes structurelles où chaque caractère était comparé à des patrons à des méthodes statistiques. Les systèmes de reconnaissance de l'écriture ont également mis à profit les progrès du traitement de l'image et de la reconnaissance de formes.

Généralement, un système de reconnaissance de formes est composé de trois étapes principales comme indique la figure: Le prétraitement, l'extraction des caractéristiques, et la classification.

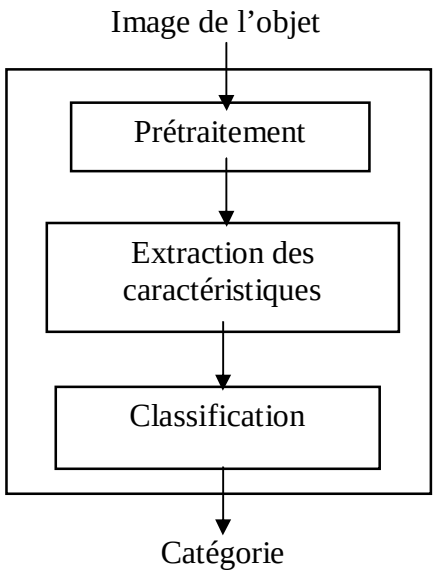


Figure IV.1. Système de reconnaissance

IV.1.Prétraitement:

Les images de l'objet digitalisé peuvent prendre différentes formes. L'un des objectifs du prétraitement des images est de réduire la variabilité qui peut exister entre les écritures de différentes formes.

Cette variabilité concerne l'inclinaison des traits, la pente des lignes ou des mots et les dimensions, notamment la hauteur de l'écriture ; le prétraitement a également pour objectif la suppression des bruits dans les images. Un classifieur ne peut pas s'appliquer efficacement aux images originales; c'est pourquoi nous avons besoin de prétraiter les images pour qu'elles soient normalisées.

Plusieurs étapes de prétraitement préliminaires sont alors nécessaires pour la reconnaissance de caractères.

IV.1.1.Réduction du bruit:

Cette étape vise à supprimer les bruits de l'image de l'entrée et à éliminer les points redondants car ces points-là peuvent causer des confusions pour le classifieur.

Le bruit est une valeur découlant habituellement de la reproduction, de la numérisation et de la transmission de l'image originale. Celui-ci ne peut pas toujours être entièrement supprimé. On utilise souvent le lissage à l'aide de filtres pour remplacer la valeur d'un pixel par la moyenne des valeurs des pixels entourant (et incluant) le pixel d'origine. Lorsqu'il s'agit d'images balayées, le lissage peut provoquer du maculage, et lorsqu'il est appliqué sur du texte en ligne, ce peut provoquer du découpage des points d'extrémité.

IV.1.2.Binarisation/seuillage:

La binarisation fait ressortir l'information utile par rapport à l'arrière plan. Elle permet de passer d'une image de niveaux de gris à une image binaire composée de deux valeurs « 0 » et « 1 », plus simple à traiter. En général, on utilise un seuil de binarisation approprié qui traduit la limite des contrastes forts et faibles dans l'image. Mais pour des images peu contrastées ou à contraste variable, il est difficile de fixer ce seuil à une valeur précise.

IV.1.3.Segmentation:

La segmentation est une procédure qui va séparer les régions dans l'image. C'est une étape importante car un classifieur n'est pas capable de reconnaître à la fois un ensemble d'objets dans l'image. Les caractères peuvent être produits en lettres attachées, et peuvent également se chevaucher, il est alors nécessaire de déterminer où débute et où prend fin un caractère; c'est pourquoi on utilise la phase segmentation. Cette procédure s'avère être une phase critique car les erreurs commises ici seront difficilement corrigeables dans les étapes ultérieures.

Selon le type de périphérique utilisé, les points élémentaires de l'image, ou pixels, peuvent prendre une valeur binaire « 0 » ou « 1 », ou une valeur quantifiée représentant un niveau de gris ou une couleur. Dans ce dernier cas l'image doit subir un processus de binarisation.

IV.1.4.Normalisation:

Après la segmentation, les tailles des images de caractères segmentés sont variables. Ce phénomène peut perturber le système de reconnaissance. On a alors besoin de normaliser les images obtenues pour éliminer l'impact de la rotation, de la translation et du facteur d'échelle.

La normalisation est une tâche nécessaire lorsque l'acquisition n'est pas réaliser avec un scanner relié au système (image existante). En effet si l'entrée du système est une image externe par rapport au système, il faut impérativement ramener les caractères à la même taille

Lorsque les deux dimensions de l'image du caractère sont inférieures aux dimensions normalisées, on peut faire de la dilatation de l'image. En revanche lorsqu'une des deux dimensions de l'image du caractère est supérieure à celle normalisée, la normalisation se fait en ré-échantillonnant selon cette direction. Dans tous les cas, il faut conserver l'aspect initial du caractère. Un classifieur va agir plus efficacement sur les images de taille homogène.

IV.1.5. Squelettisation:

La squelettisation est une technique très utilisée dans la reconnaissance des caractères. Elle permet de diminuer l'information utile en ne gardant que le squelette de la forme ; le principe est de ramener l'image du mot à une écriture linéaire d'une épaisseur égale à un pixel, en préservant la forme, la connexité et la topologie du tracé.

IV.1.6. Redressement:

Le redressement est une opération fréquente en analyse de document, souvent due à un mauvais positionnement du document sur le scanner, conduisant à une inclinaison de l'image. Dans l'écriture, il existe plusieurs méthodes de redressement tels que :

- Le redressement de la ligne de base : L'idée est de rendre horizontaux les mots à l'aide d'une transformation géométrique de type rotation isométrique des points de l'image.
- Le redressement des écritures penchées : Cette technique facilite la segmentation préalable des mots en caractères, l'idée est de trouver l'angle moyen d'inclinaison puis faire une transformation géométrique de type cisaillement de l'image.

IV.1.7.L'opération de cadrage:

Cette opération consiste à chercher la première et la dernière ligne/colonne signifiante, ensuite créer une nouvelle image cadrée à partir de l'image mère.

IV.2.Extraction des caractéristiques:

L'extraction des caractéristiques (ou primitives) est une étape très importante à partir de laquelle chaque objet sera représenté par son vecteur de caractéristiques sous forme d'une matrice, de grande dimension en général. Si cette étape n'est pas assurée correctement, l'apprentissage et la reconnaissance risquent d'aboutir à des résultats erronés.

Pour la reconnaissance de caractères, les primitives qui seront extraites doivent présenter, d'une part, une certaine invariance géométrique et, d'autre part, une certaine invariance statistique. Tandis que l'invariance géométrique signifie une tolérance des opérateurs des translations, de rotation et de changement d'échelle, l'invariance statistique signifie une tolérance du bruit inévitable.

Vu que le vecteur caractéristique est une matrice de grande dimension, si on laisse le classifieur travailler directement sur cette matrice, cela peut causer un problème de performance et d'efficacité.

En effet, même si on utilise un classifieur très performant celui-ci ne peut compenser un mauvais choix des primitives.

IV.2.1.Méthodes d'extraction des caractéristiques:

Dans la littérature, les primitives sont généralement classées de trois façons différentes :

- **En fonction du type d'image:**

Une première distinction est faite entre les primitives selon qu'elles sont extraites d'une image en niveaux de gris, d'une image binarisée, du contour ou du squelette de la forme résultant de la phase de prétraitement.

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

Figure IV.2. Image en niveaux de gris, binaire, contour et squelette

Dans le cas d'une image au niveau de gris **(a)**, on extrait des primitives de type zoning, moment géométriques ou de Zernike[2], ou des primitives discrète (largeur, hauteur, épaisseur moyenne des traits, etc.)

Dans le cas d'une image binarisée **(b),** il existe deux approches possibles : l'analyse indirecte et l'analyse directe selon que l'on passe par une extraction des composantes connexes (les méthodes couramment utilisées sont le marquage des composantes connexes et les diagrammes de Voronoi[3]) ou non. Les primitives seront, entre autres, des projections, des histogrammes, des moments géométriques ou de Zernike[2].

Des primitives de type zoning ou des primitives discrètes (largeur, hauteur, épaisseur moyenne des traits, etc.)

Dans le cas d'un contour **(c),** on extrait d'avantage des profils, des descripteurs de Fourier [4] ou d'ondelettes, des splines et également de type zoning, code de Freeman[5], contour code et on utilise également des contours actifs tels que les snakes ainsi que des primitives discrètes (périmètre, compacité, excentricité, etc.)

Enfin dans le cas d'un squelette **(d),** les primitives extraites seront des descripteurs de Fourier [4], des descriptions de graphe, des primitives discrètes (nombre de boucles, de croisements, rapport largeur sur hauteur et présence d'un point isolé, etc.)

- **Selon les primitives globales et locales :**

La deuxième distinction peut être effectuée entre les primitives globales et locales.

Les primitives globales : Ayant pour but de représenter au mieux la forme générale d'un caractère. On cite entre autres la transformée de Fourier [4], la transformée de Hough [6] qui détecte les lignes dans les images, ainsi que des primitives discrètes précédemment citées.

Les primitives locales : calculées dans une fenêtre glissante qui parcourt les pixels de l'image avec un pas d'analyse qui dépendent de la modélisation, du type de primitives et de la taille de l'image.

Parmi ces primitives on trouve la transformée de Fourier fenêtrée (Gabor[7]) et les dérivées qui extraient une information sur la direction des traits, les moments ainsi que des primitives discrètes décrite auparavant.

- Selon les primitives topologiques ou statistiques :

La troisième distinction est effectuée entre les primitives topologique ou statistiques :

- Primitives topologiques ou métriques : On compte dans une forme ou un échantillon le nombre de trous, évalue les concavités, ou mesure des pentes et autres paramètres de

courbures et d'orientations, ou mesure la longueur et l'épaisseur des traits, ou détecter les croisements et les jonctions des traits, mesurer les surfaces et périmètres, etc.

Primitives topologique structurelles**:** Elles sont généralement extraites, non pas de l'image brute, mais à partir du squelette ou contour de la forme. Ainsi on ne parle plus de trous mais de boucles ou de cycles dans une représentation du caractère.

- Primitives statistique : Elles véhiculent une information distribuée sur toute l'image et représentées par un histogramme qui indique le nombre de pixels sur chaque ligne ou colonne de l'image.

On peut citer aussi l'approche fondée sur une moyenne des pixels situés à l'intérieur d'un masque rectangulaire (zoning). Il faut, pour cela, construire une matrice de masques recouvrant la totalité de la forme qui permet une représentation statistique des valeurs correspondante à chaque masque.

V. Classification:

La classification est l'opération de base d'un système de reconnaissance de caractères. Son principe consiste à partir d'une source d'entrée constituée de plusieurs caractères différents dans notre cas à reconnaitre chacun d'entre eux et à lui attribuer, en sortie, une classe propre à chaque caractère. Deux caractères différents (exemple: Lettre A et Lettre B) seront donc rangés dans deux classes différentes.

Pour pouvoir être distingués l'un de l'autre par le classifieur, chaque lettre doit être caractérisée par des paramètres pertinents choisis de telle sorte qu'il n'y ait pas de risque de confusion entre elles.

Une fois ces paramètres définis, le classifieur doit, dans un premier temps, apprendre à les reconnaitre. Cette étape dite, phase d'apprentissage, consiste à injectés à l'entée du système, l'ensemble des caractères de la source autant de fois qu'il le faut jusqu'à ce que la structure interne du classifieur, qui s'auto- corrige au fur et à mesure à l'étape de reconnaissance.
Durant cette dernière, chaque caractère de la source d'entrée pourra alors être reconnu et rangé dans la classe qui lui correspond.

Le rôle d'un classifieur est de déterminer, parmi un ensemble fini de classe, à laquelle appartient un objet donné.
Un tel système doit être capable de modéliser les frontières qui séparent les classes les unes des autres. Cette modélisation est appelée fonction discriminante, à partir de laquelle la classification s'effectue.

De nombreux classifieurs existent et sont plus ou moins bien adaptés à la reconnaissance de l'écriture.

V.1.Les principaux systèmes de classification:

Parmi les classifieurs les plus utilisés, on peut citer les SVM (Machines à Vaste marge), les classifieurs statistiques par HMM (Hidden MarkovModel [8]) et les RDN (Réseaux de Neurones).

V.1.1.Les SVM :

Les machines à vecteur de support appelées aussi classificateur à marge optimale ou encore séparateurs à vaste marge, SVM ont été introduites par Vapnik [9]. Leur principe est de maximiser la marge entre les classes. Il faut donc déterminer l'hyperplan maximisant cette marge.

Les SVM offrent des performances intéressantes pour la reconnaissance de caractère manuscrites, mais ils sont peu applicables à la reconnaissance de mots (sauf éventuellement avec une segmentation explicite des mots en lettres).

En effet, ils travaillent avec des données en dimension fixe et ne permettent donc pas d'introduire la variabilité de longueur des mots. De plus ils ont l'inconvénient d'avoir un temps de calcul assez long en phase d'apprentissage comme en phase de reconnaissance.

V.1.2.Les approches markoviennes :

Les modèles Markov sont couramment utilisés pour la reconnaissance de l'écriture manuscrite. En effet, ce sont des outils statistiques puissants permettant de calculer la probabilité d'appartenance d'une forme à une classe. La forme est vue comme un ensemble d'observation émises par des états cachés. La modélisation markovienne est bien adaptée à l'écriture manuscrite car elle permet d'intégrer les variabilités de longueur des mots.

V.1.3.Les réseaux de neurones :

L'idée principale est qu'un neurone formel est capable de réaliser des calculs élémentaires comme la séparation d'un vecteur en deux classes, chaque classe étant déterminée par le poids du neurone. Le problème est alors de choisir quel coefficient affecter aux poids pour réaliser une séparation optimale. La multiplication des neurones permet de séparer plusieurs classes : Il faut donc réaliser un choix sur la topologie du réseau. Les RDN sont bien adaptés à la reconnaissance de forme globale telle que des caractères isolés sur une représentation en

dimension fixe.

Conclusion:

Nous avons donc donné un aperçu général sur la reconnaissance des caractères et les outils nécessaires pour cet objectif.

Dans notre travail nous utilisons un système de reconnaissance à base de réseau de neurones. Le chapitre suivant décrit en détail le fonctionnement de ce type de système.

Introduction :

Les réseaux de neurones ont été utilisés fréquemment en tant que classifieur, par exemple pour l'identification d'anomalies. Dans ce qui suit nous commençons par expliquer le principe d'utilisation des réseaux de neurones puis nous exploiterons leurs propriétés pour la classification en vue de la reconnaissance de caractères. Nous détaillerons la manière de construire un classifieur à base de réseaux de neurones et nous mettrons en évidence les avantages et les limitations des réseaux de neurones par rapport aux autres classifieurs.

I. Historique :

A l'origine, Warren McCulloch et Walter Pitts [10] présentent, en 1943, le neurone formel qui est une abstraction du neurone physiologique. Ils montrent, théoriquement, que les réseaux de neurones formels simples peuvent réaliser des fonctions logiques, arithmétiques et symboliques complexes.

Les travaux de McCulloch et Pitts [10] n'ont pas donné d'indications sur la manière d'adapter les coefficients synaptiques qui relient les neurones entre eux. En 1949, D.Hebb [11] présente, dans son ouvrage « The Organization of Behavior » [12] une règle d'apprentissage qui permet de modifier les valeurs des coefficients synaptiques en fonction de l'activité des unités qu'ils relient.

En 1958, F.Rosenblatt [13] développe le modèle du perceptron. Ce dernier comporte deux couches de neurones : Une couche cachée et une couche liée à la prise de décision. C'est le premier système artificiel capable d'apprendre par expérience.

Dans la même période, le modèle de l'ADALINE (ADAptive LINar Element) a été présenté par B.Widrow et Hoff [14] Ce modèle sera par la suite le modèle de base des réseaux multi-couches.

En 1969, M.Minsky et S.Papert [15] publient un ouvrage mettant en exergue quelques limitations théoriques du Perceptron, notamment l'impossibilité de traiter des problèmes non linéaires.

En 1972, T.Kohonen [16] présente ses travaux sur les mémoires associatives et propose des applications à la reconnaissance de formes.

C'est en 1982 que J.Hopfield [17] présente son étude d'un réseau complètement rebouclé, dont il analyse la dynamique.

En 1984, c'est le système de rétro-propagation du gradient de l'erreur qui est le sujet le plus débattu dans le domaine.

Aujourd'hui, nous retrouvons les réseaux de neurones solidement implantés dans diverses industries, et leurs applications concernent des domaines aussi nombreux que variés.

II. Les réseaux de neurones :

L'évolution de la théorie des réseaux de neurones formels est inspirée directement du développement des travaux biologiques sur le cerveau humain.

II.1. Le neurone biologique :

Le cerveau est l'organe de commande le plus complexe et le plus inconnu de la biologie de l'homme ou de l'animal. Les cellules nerveuses, appelées neurones, sont les éléments de base du système nerveux central. Ce dernier en posséderait environ cent (100) milliards.

Dans leur organisation générale et leur système biochimique, les neurones possèdent de nombreux points communs avec les autres cellules. Ils présentent, cependant, des caractéristiques particulières qui se distinguent par quatre fonctions spécialisées qu'ils assument :

- Recevoir des signaux en provenance des neurones voisins ;
- Intégrer ces signaux ;
- Engendrer un influx nerveux ;
- Conduire et transmettre l'influx nerveux à un neurone capable de le recevoir.

Le neurone biologique possède trois (03) principales composantes : Les dendrites, le corps cellulaire et l'axone (figure II.1).

- **Les dendrites** : constituées d'un maillage de récepteurs nerveux, acheminent vers le corps du neurone les signaux électriques provenant d'autres neurones.
- **Le corps du neurone** : Agit comme un intégrateur en accumulant les charges électriques.
- Lorsque le neurone devient suffisamment excité (charge accumulée dépasse un certain seuil) par un processus électrochimique, il engendre un potentiel électrique qui se propage à travers **l'axone**.

  Le contact entre l'axone et les dendrites d'un autre neurone s'effectue au niveau des **synapses**.

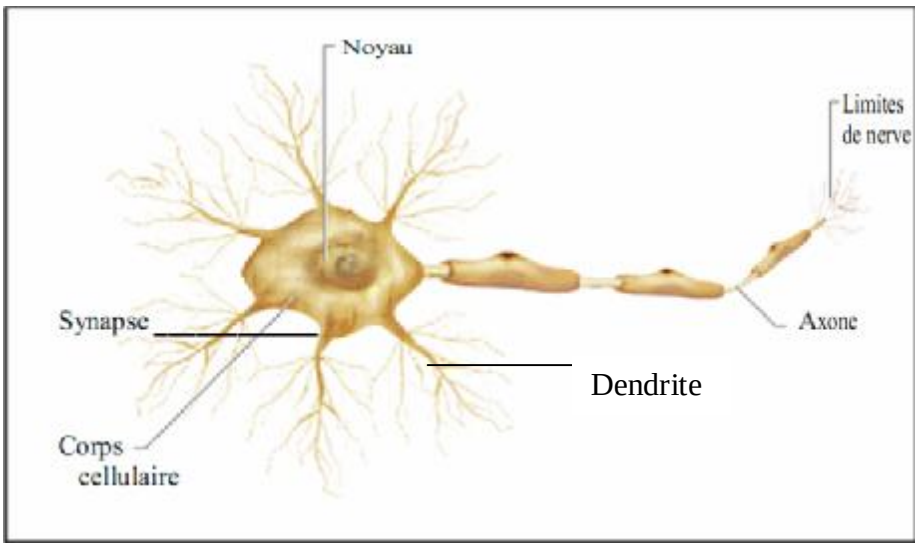


Figure II.1.Schéma du neurone biologique.

II.2. Structure général du réseau de neurone:

II.2.1. Neurone formel :

Inspiré du système biologique, le neurone formel est un automate caractérisé par un petit nombre de fonctions mathématiques. Il traite un signal recueilli (signal d'entrée) à travers ses connexions entrantes pour fournir un signal de sortie calculé par la fonction de transfert. En générale, on considère que chaque neurone fournit une information additive aux unités de calcul (neurones) qui lui sont connectés.

La valeur de la somme pondérée, ***S*** est simplement, la somme des sorties des différents neurones en amont avec lesquels il est connecté, plus une valeur seuil ***b*** (biais interne). La figure II.2 résume la structure détaillée du neurone.

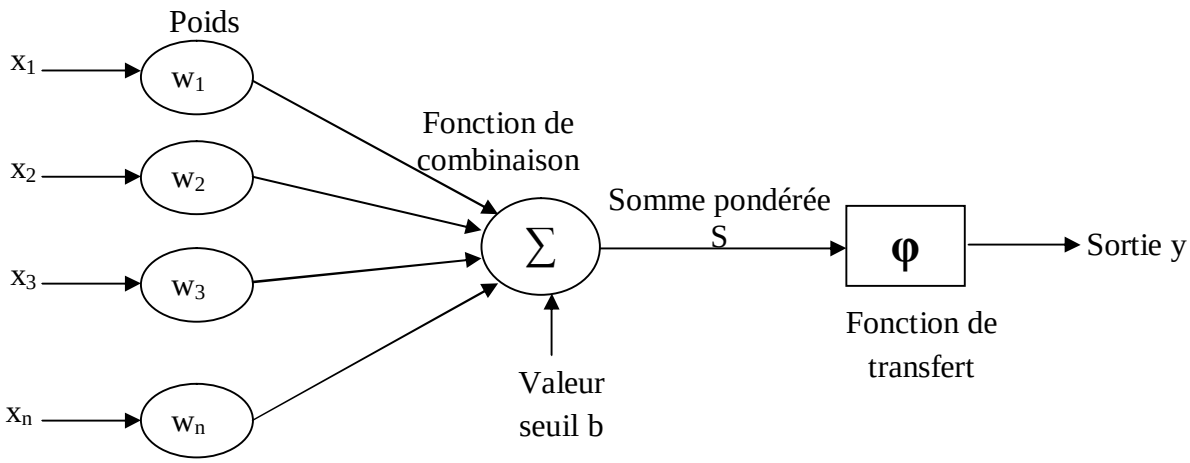


Figure. II.2.Modèle générale du neurone formel

$[x \ldots \ldots . x_n]$: *Représente les entrées de i neurone*;

$[w_1 \ldots \ldots . w_n]$: *Représente les poids associés au* $n^{ème}$ *entrée du neurone* ;

$S = \sum_{i=1}^{n} w_{i,k} \times x_n + b$: *Représente la valeur de la somme pondéré* ;

φ: *Représente la fonction d'activation* ;

$y = \varphi(S)$: *Représente la sortie du neurone* ;

b: *Représente le biais interne* ;

II.2.1.1. Le signal d'entrée :

Dans un réseau de ***n*** neurones chacun d'entre eux reçoit en général de **1** à ***n*** stimulations en provenance des autres cellules. Les valeurs, binaires ou réelles, sont pondérées puis additionner par la fonction φ.

Si le neurone dont nous calculons l'entrée est le neurone ***i***, nous notons W_{ik} représentant la valeur réelle la pondération correspondant à l'arc reliant le neurone ***k*** au neurone ***i*** ce qui correspond au, poids de la connexion.

II.2.1.2. La fonction d'activation (ou fonction de transfert) :

La fonction d'activation représente l'un des paramètres les plus importants du réseau de neurones. En effet, l'état des neurones peut être à valeurs binaires ou réelles et la fonction d'activation peut fortement varier suivant le type de réseau et le modèle de réseau à étudier. Les deux fonctions couramment utilisées sont la fonction signe, pour les neurones à états binaires, et la fonction sigmoïde pour les neurones à état réels.

Les principales fonctions d'activation sont récapitulées dans le tableau suivant :

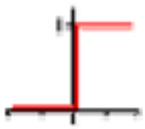
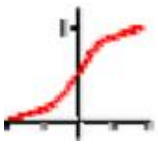
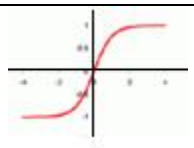
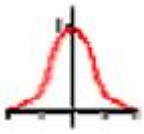
Nom de la fonction	Relation entrée/sortie	allure
Fonction signe		
Fonction sigmoïde	————————	
Fonction hyperbolique	————————	
Fonction Gaussienne	————————	

Tableau II.1. Les fonctions d'activation

II.2.1.3. Le signal de sortie :

Un réseau de neurones peut comporter plusieurs sorties comme il peut avoir qu'une seule pour un grand nombre d'entrées, elles peuvent être de type signal binaire ou réel, la sortie ***y*** est en général du même type que les entrées.

II.2.2. Modélisation d'un neurone formel :

La modélisation consiste à mettre en œuvre un système de réseaux neuronaux sous un aspect non pas biologique mais artificiel. Cette modélisation mathématique suppose une correspondance avec chaque élément composant le neurone biologique.

Le tableau suivant montre la relation entre le neurone biologique et le neurone formel.

Neurone biologique	Neurone formel
Synapses	Poids de connexion
Axone	Signal de sortie
Dendrites	Signal d'entrée
Corps cellulaire (somma)	Fonction d'activation

Tableau II.2. Relation entre le neurone biologique et le neurone formel

II.3. Types des réseaux de neurones :

Un neurone réalise une fonction non linéaire paramétrée de ses entrées. Un réseau de neurones est un maillage de plusieurs neurones organisés en couches.

On peut distinguer essentiellement deux familles de réseaux de neurones en fonction du chemin suivi par le signal qui le travers : Les réseaux non bouclés appelés aussi feed-forward et les réseaux bouclés.

II.3.1. Les réseaux non bouclés :

Il s'agit des réseaux appelés aussi réseaux de type Perceptron ou statiques, et dans lesquels l'information se propage de couche d'entrée vers la couche de sortie sans retour en arrière, en suivant les connexion on peut pas revenir au neurones de départ. Figure II.3

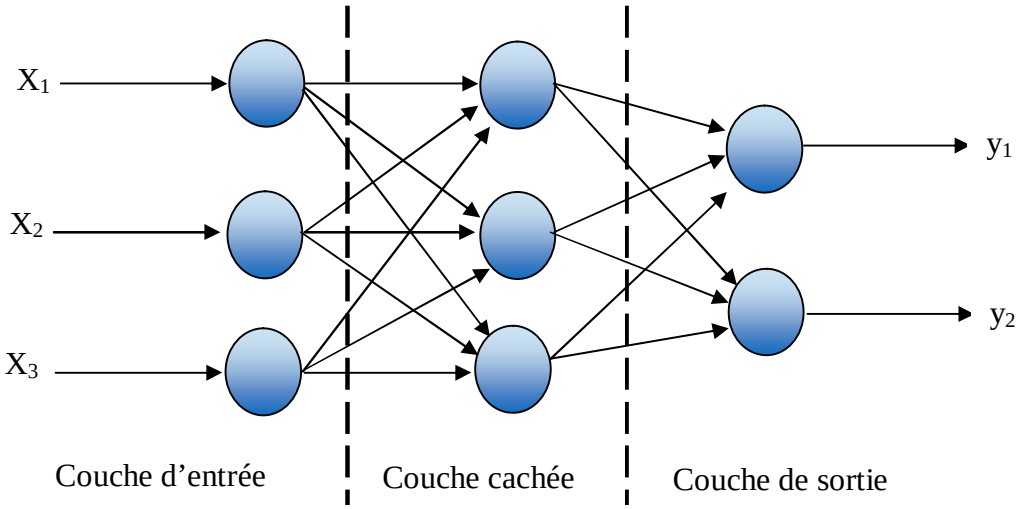


Figure II.3. Réseau de neurones non bouclé.

II.3.2. Les réseaux bouclés ou récurrents :

Par opposition aux réseaux non bouclés, les réseaux récurrents peuvent contenir des chemins bouclés, passant plusieurs fois par un même neurone (figure II.4). Grâce à cette structure cyclique, un stimulus entrant peut être partiellement ou totalement remis en question par l'état antérieur du réseau ou par l'arrivée de stimuli postérieurs. Ce type de réseau a donc des capacités théoriques supérieures à celles des réseaux non récurrents. Les réseaux récurrents présentent donc une dynamique complexe due aux multiples rétroactions internes.

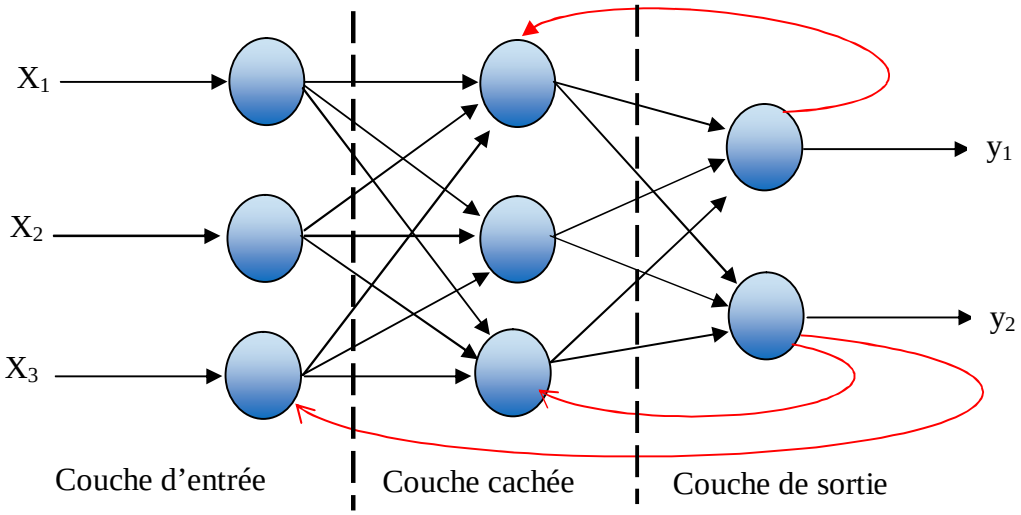


Figure. II.4. Réseau de neurones bouclé.

II.4. Apprentissage :

L'apprentissage est un processus d'adaptation des paramètres d'un système pour donner une réponse désirée à une entrée ou stimulation quelconque. Un réseau de neurone nécessite une phase d'apprentissage durant laquelle ce réseau construit les liens qui lui permettant d'effectuer la mémorisation, la classification et la prédiction du type de données présentées.

L'algorithme d'apprentissage est l'opération qui permet de modifier les poids selon l'évolution de l'erreur entre la sortie obtenue et la sortie désirée. Il contrôle tout le comportement futur du réseau de neurones. La capacité de mémorisation et de généralisation du réseau réside dans la configuration de ses poids.

On trouve trois types d'algorithme d'apprentissage : L'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage par renforcement.

II.4.1. Apprentissage supervisé :

Dans ce mode, un professeur qui connaît parfaitement la sortie désirée ou correcte guide le réseau en lui apprenant à chaque étape le bon résultat. Donc l'apprentissage, ici, consiste à comparer le résultat obtenu avec le résultat désiré, puis corriger les poids de connexion de façon à réduire l'erreur commise par le réseau, c'est-à-dire la différence entre la sortie désirée et la réponse à l'entrée correspondante.

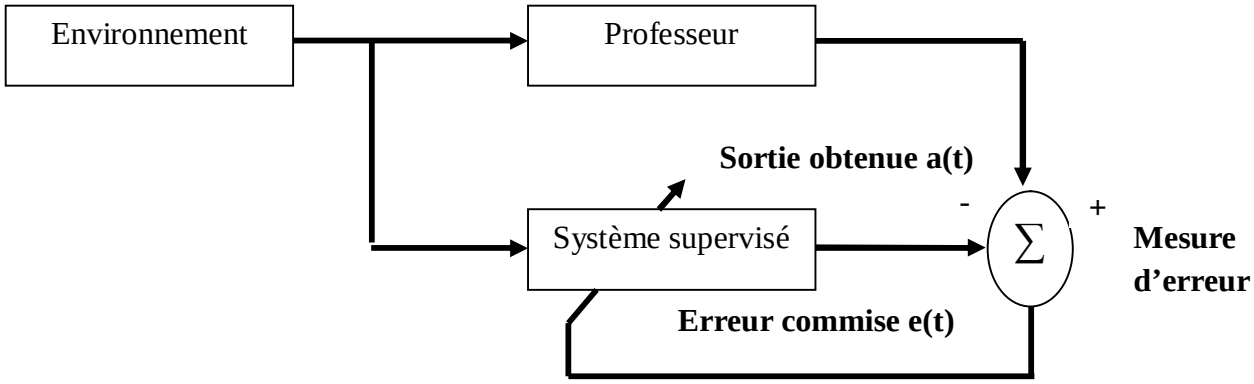


Figure. II.5. Schéma bloc d'apprentissage supervisé

II.4.2. Apprentissage non supervisé :

Dans l'apprentissage non supervisé, le réseau modifie ses paramètres en tenant compte seulement des informations d'une base d'entrées (il n y a pas de professeur). C'est pourquoi on l'appelle, quelquefois, auto apprentissage. Ces méthodes n'ont pas besoins de sorties désirées préétablies, ce sont des réseaux à dynamique autonome (figure II.6).

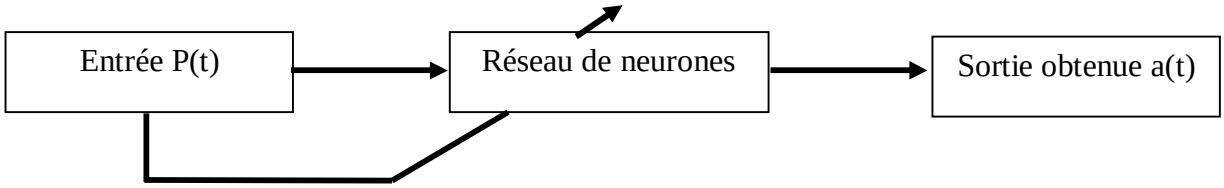


Figure. II.6. Schéma bloc de l'apprentissage non supervisé

II.4.3. Apprentissage renforcé :

Le renforcement est, en fait, une sorte d'apprentissage supervisé. Dans cette approche, le réseau doit apprendre la corrélation entrées/sorties via une estimation de son erreur, c'est-à-dire du rapport échec/succès.

Le réseau va donc tendre à maximiser un index de performance qui lui est fourni, appelé signal de renforcement. Le système étant capable ici, de savoir si la réponse qu'il fournit est correcte ou non, mais il ne connaît pas la bonne réponse.

II.5. Propriétés des réseaux de neurones :

Les réseaux de neurones possèdent certaines propriétés très intéressantes :

II.5.1. L'approximation universelle :

Propriété : *Toute fonction bornée suffisamment régulière peut être approximée uniformément, avec une précision arbitraire, dans un domaine fini de l'espace de ses variable, par un réseau de neurones comportant une couche de neurones cachés en nombre fini, possédant tous la même fonction d'activation, et un neurone de sortie linéaire.* HORNIK et al.1989, HORNIK et al. 1990, HORNIK 1991. [18]

Cette propriété ne donne pas de méthode pour trouver les paramètres du réseau et n'est pas spécifique aux réseaux de neurones. Il existe bien d'autres familles de fonctions paramétrées possédant cette propriété, c'est le cas notamment des ondelettes.

II.5.2. L'approximation parcimonieuse :

Lorsqu'on veut modéliser un processus à partir des ordonnées, on cherche toujours à obtenir les résultats les plus satisfaisants possible avec un nombre minimum de paramètres ajustables (les poids de connexions), HORNIK [18] 1993 a montré que :

Si l'approximation dépend des paramètres ajustables de manière non linéaire, elle est plus parcimonieuse que si elle dépend linéairement des paramètres.

Plus précisément, le nombre de paramètres croît exponentiellement avec le nombre de variables dans le cas des fonctions linéaires par rapport à leurs paramètres, alors qu'il croît linéairement avec ce nombre pour les fonctions non linéaires.

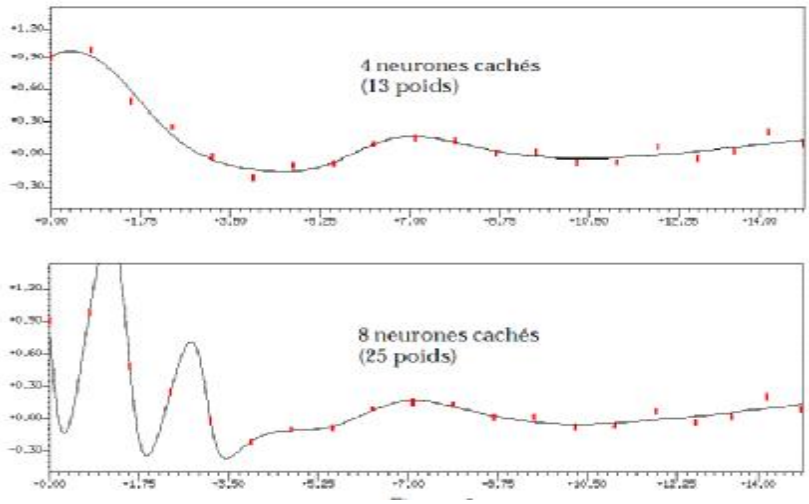


Figure. II.7. Approximation parcimonieuse.

On constate que le réseau le plus parcimonieux (4 neurones cachés, soit 13 coefficients) produit un meilleur ajustement qu'un réseau trop riche en coefficients (8 neurones cachés, soit 25 coefficients).

II.5.3. La généralisation et le sur-apprentissage :

Un réseau de neurones nécessite une phase d'apprentissage. Lors de cette phase, le réseau doit posséder un critère d'arrêt d'apprentissage.

Si le réseau apprend trop longtemps, un phénomène de sur-apprentissage risque de survenir, entraînant une dégradation des performances de généralisation. Ainsi, dans un réseau de neurones, à partir d'un certain nombre d'époques d'entraînement, le système se sur-spécialise par rapport à la base de données d'apprentissage. Il perd alors sa capacité de généralisation par rapport aux données de test provenant de la même source mais qui n'ont pas encore été traitées par le système. La figure(II.8) représente ce phénomène.

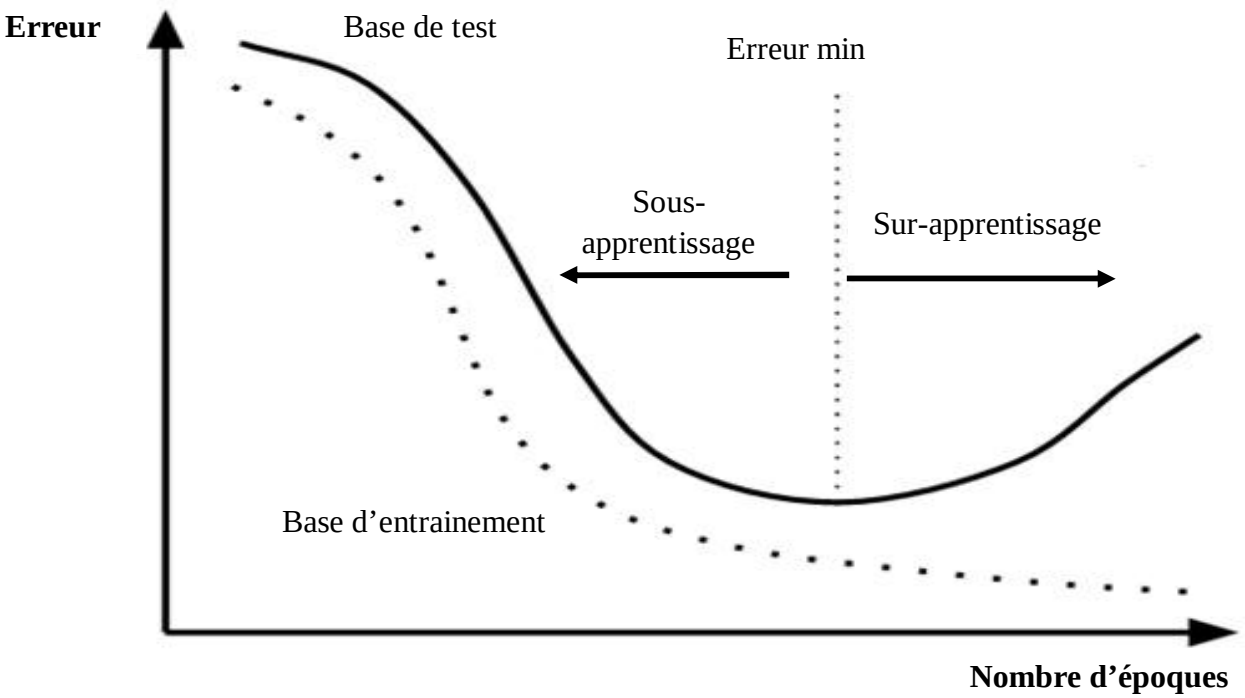


Figure. II.8. Schématisation de l'erreur en fonction du nombre d'époques lors de la phase d'apprentissage.

Cette figure montre l'erreur de généralisation en fonction du nombre d'époques d'apprentissage. On constate que l'erreur d'apprentissage commence par décroître. Ainsi, le réseau de neurones devient meilleur d'époque en époque pour classifier la base d'entraînement.

Par contre, à partir d'un certains nombre d'époques, on remarque que l'erreur sur la base de test augmente. Cet effet est dû au fait que le réseau s'est sur-spécialisé pour la base d'entraînement et qu'il perd sa capacité de généralisation. Ainsi, il faut arrêter l'entraînement du système à l'époque le plus proche possible de la frontière entre le sous-apprentissage et le sur-apprentissage.

II.6. Algorithme de rétro-propagation de l'erreur:

Parmi les différentes structures possibles, la technique de **rétro-propagation du gradient** (*Back propagation* en anglais) est l'une des plus utilisées. Cette méthode permet de calculer le gradient de l'erreur pour chaque neurone de sortie puis corrige les poids de connexions en fonction du gradient obtenu. De la dernière couche vers la première. De façon abusive, on appelle souvent technique de rétro-propagation du gradient, l'algorithme classique de correction des erreurs basé sur le calcul du gradient grâce à la rétro-propagation et c'est cette méthode qui est présentée ici.

Dans le cas des réseaux de neurones, les poids synaptiques qui contribuent à engendrer une erreur importante se verront modifiés de manière plus significative que les poids qui ont engendré une erreur marginale. Ce principe fonde les méthodes de type algorithme du gradient, qui sont efficacement utilisées dans des réseaux de neurones multicouches comme les perceptrons multicouches. L'algorithme du gradient a pour but de converger de manière itérative vers une configuration optimisée des poids synaptiques. Cet état peut être un minimum local de la fonction à optimiser et idéalement, un minimum global de cette fonction (dite fonction de coût). Les poids dans le réseau de neurones sont au préalable initialisés avec des valeurs aléatoires. On considère ensuite un ensemble de données qui vont servir à l'apprentissage. Chaque échantillon possède ses valeurs cibles qui sont celles que le réseau de neurones doit à terme prédire lorsqu'on lui présente le même échantillon. L'algorithme se présente comme ceci :

Soit un échantillon $\vec{x}$ que l'on met à l'entrée du réseau de neurones et la sortie $\vec{t}$ recherchée pour cet échantillon.

- On propage le signal en avant dans les couches du réseau de neurones :

$$x_k^{(n-1)} \rightarrow \; x_j^{(n)}$$

- La propagation vers l'avant se calcule à l'aide de la fonction d'activation g, de la fonction d'agrégation h (souvent un produit scalaire entre les poids et les entrées du neurone) et des poids synaptiques $\vec{w}_{jk}$ entre le neurone $x_k^{(n-1)}$ et le neurone $x_j^{(n)}$.

$$x_j^{(n)} = g^{(n)}(h_j^{(n)}) = g^{(n)}(\sum_k w_{jk}^{(n)} x_k^{(n-1)})$$

- Lorsque la propagation vers l'avant est terminée, on obtient à la sortie le résultat $\vec{y}$
- On calcule alors l'erreur entre la sortie donnée par le réseau $\vec{y}$ et le vecteur désiré $\vec{t}$ à la sortie pour cet échantillon. Pour chaque neurone **i** dans la couche de sortie, on calcule :

$$e_i^{sortie} = g'(h_i^{sortie})[t_i - y_i]$$

où g' est la dérivée de g

- On propage l'erreur $e_i^{(n)} \mapsto e_j^{(n-1)}$ vers l'arrière grâce à la formule suivante :

$$e_j^{(n-1)} = g'^{(n-1)}(h_j^{(n-1)}) \sum_i w_{ij} e_i^{(n)}$$

Avec: $e_j^{(n)} = \sum_i [t_i - y_i] \frac{\partial y_i}{\partial h_j^{(n)}}$

- On met à jour les poids dans toutes les couches :

$$\Delta w_{ij}^{(n)} = \lambda e_i^{(n)} x_j^{(n-1)}$$

Où λ représente le taux d'apprentissage (de faible magnitude et inférieur à 1).

III. **Les limites des réseaux de neurones :**

Les réseaux de neurones artificiels ont besoin de cas réels servant d'exemples pour leur apprentissage (on appelle cela la *base d'apprentissage*). Ces cas doivent être d'autant plus nombreux que le problème est complexe et que sa topologie est peu structurée. Par exemple, on peut optimiser un système neuronal de lecture de caractères en utilisant le découpage manuel d'un grand nombre de mots écrits à la main par de nombreuses personnes.

Chaque caractère peut alors être présenté sous la forme d'une image brute, disposant d'une topologie spatiale à deux dimensions, ou d'une suite de segments presque tous liés.
La topologie retenue et la complexité du phénomène modélisé, et le nombre d'exemples doivent être en rapport. Sur un plan pratique, cela n'est pas toujours facile car les exemples peuvent être soit en quantité absolument limitée ou trop onéreux à collecter en nombre suffisant.

Il y a des problèmes qui se traitent bien avec les réseaux de neurones, en particulier ceux de *classification* en *domaines convexes* (c'est-à-dire tels que si des points A et B font partie du domaine, alors tout le segment AB en fait partie aussi). Des problèmes comme "*Le nombre d'entrées à 1 (ou à zéro) est-il pair ou impair* ?" se résolvent en revanche très mal : pour affirmer de telles choses sur 2 puissance N points, si on se contente d'une approche naïve mais homogène, il faut précisément N-1 couches de neurones intermédiaires, ce qui nuit à la généralité du procédé.

Un exemple caricatural, mais significatif est le suivant : Disposant en entrée du seul poids d'une personne, le réseau doit déterminer si cette personne est une femme ou bien un homme. Les femmes étant statistiquement un peu plus légères que les hommes, le réseau fera toujours un peu mieux qu'un simple tirage au hasard : cet exemple dépouillé indique la simplicité et les limitations de ces modèles mais il montre également comment l'étendre : L'information "port d'une jupe", si on l'ajoute, on aurait clairement un coefficient synaptique plus grand que la simple information de poids.
Les réseaux de neurones formels, tel que nous les avons définit possèdent un point remarquable qui est à l'origine de leur pratique dans des domaines très divers.

IV. Classification et approximation de fonction :

Considérons un problème à deux classes ***« A »*** et ***« B »,*** et considérons la fonction ***f(x)*** qui vaut **(+1)** si l'objet décrit par le vecteur ***X*** appartient à la classe ***« A »,*** et qui vaut **(0)** si cet objet appartient à la classe ***« B ».***

L'approximation, au sens des moindres carrés, de la fonction ***f(x)*** n'est autre que la probabilité d'appartenance de l'objet **X** à la classe ***« A ».*** Or, nous venons de voir que les réseaux de neurones constituent de bon approximateurs pour l'évaluation d'une probabilité d'appartenance, suivie d'une décision quant à l'appartenance de l'objet à l'une ou l'autre classe.

Contrairement à une croyance largement répandue, les réseaux de neurones ne sont pas limités à une prise de décision binaire; bien au contraire, ils sont intrinsèquement capables de

réaliser une estimation d'une probabilité d'appartenance.

Cette propriété est souvent ignorée, alors qu'elle est particulièrement précieuse lorsqu'on veut introduire un réseaux de neurones, comme outil de classification dans des systèmes complexes, contenant par exemple d'autres classificateurs, fondés sur des principes différents, on utilisant des représentations différentes pour les formes à classer.

IV.1. Classification neuronale et non neuronale :

Nous venons de justifier le fait que les réseaux de neurones fournissent une estimation de la probabilité d'appartenance à une classe. Les réseaux de neurones ne sont pas les seuls à avoir cette propriété ; il existe, comme indiqué dans le chapitre précédent, d'autres méthodes de classifications, dites paramétriques, dont le principe consiste à faire une hypothèse sur la forme de distribution de probabilité d'appartenance, et à ajuster les paramètres de cette distribution. Le résultat final est donc, en théorie, le même que celui que l'on obtient à l'aide d'un réseau de neurones.

Toutefois, l'avantage que présente un réseau de neurone dans ce domaine, c'est peut être une certaine facilité de mise en œuvre, et certainement une parallélisassions facile si le réseau est réalisé électroniquement, ou s'il est implanté sur ordinateur parallèle.

Lorsque on cherche à résoudre un problème de classification, on a affaire à des formes (par exemple des chiffres manuscrits, des phénomènes, des pièces mécaniques, etc.) susceptible d'appartenir à des catégories, ou classes différentes. Un classifieur est capable d'attribuer une classe à une forme inconnue qui lui est présentée, ou de refuser de lui attribuer une classe si la forme inconnue est trop éloignée des formes utilisée pour l'apprentissage.

Les performances d'un classifieur dépendent essentiellement de la représentation des formes c'est-à-dire du prétraitement effectué sur les données brutes.

Si la représentation des formes est bien choisie, et si le réseau est bien conçu, les réseaux de neurones offrent des performances équivalentes à celles des milliers de classifieurs en termes de taux de reconnaissance. Mais il se distingue de ceux-ci par une plus grande facilité d'implantation sous forme de circuits spécialisés très rapides.

Comme nous l'avons indiqué plus haut, les réseaux de neurones contrairement aux idées reçues, ne se contentent pas de prendre la décision d'attribuer un objet à une classe. Ils permettent d'obtenir une information beaucoup plus riche: La probabilité d'appartenance de l'objet à chacune des classes. A partir de cette information il est possible de prendre une décision ceci permet notamment de combiner cette probabilité avec des informations

provenant par exemple, d'autres systèmes de classification, utilisant d'autres algorithmes ou d'autres représentations des formes. La vue caricaturale du réseau de neurones prenant une décision d'appartenance ou de rejet doit donc être oubliée au profit d'une décision plus complète de ce que savent réellement apporter les réseaux dans le domaine de classification automatique.

À titre d'exemple de système industriel complexe utilisant les réseaux de neurones en collaboration avec d'autres techniques, citons l'utilisation par la poste dans le tri automatique faisant intervenir les réseaux de neurones pour la reconnaissance de code postal, etc.

Conclusion :

Nous venons donc de voir que la reconnaissance de caractères nécessite l'utilisation d'un classifieur ayant pour but de distinguer entre des caractères différents. Nous avons opté pour l'utilisation d'un classifieur à base de réseaux de neurones et expliqué comment concevoir ce système.

Dans ce qui suit, nous présentons notre application de reconnaissance de lettres avec un classifieur neuronal. Nous développerons le cas de la reconnaissance de trois lettres puis on généralisera dans le cas de plusieurs caractères.

Introduction :

Notre objectif étant de pouvoir reconnaître chaque caractère parmi un ensemble d'autre, nous allons utiliser les réseaux de neurones et exploiter les propriétés de ce dernier pour la conception d'un classifieur. Le principe consistera alors, à définir chaque caractère à partir d'un certain nombre de paramètres (attributs) et à faire en sorte que le classifieur puisse ranger un caractère donné dans sa classe qui lui correspond.

Nous allons dans un premier temps appliquer notre système à la reconnaissance de trois lettres A, B, C, puis nous généralisons au cas d'un plus grand nombre de caractères.

Le réseau de neurones utilisé est de type rétro-propagation de l'erreur réalisé sous le logiciel MATLAB version 7.10.0. (R2010a).

I. Conception du classifieur :

La conception du classifieur nécessite de suivre les différentes étapes suivantes :

I.1. Préparation des données:

La première étape consiste à affecter à chaque caractère (lettre) un certain nombre de paramètres, appelés primitives ou attributs, qui caractériseront, de manière significative, chacune de ces lettres de façon à pouvoir les distinguer les unes des autres. Les lettres seront alors introduites à l'entrée du classifieur à partir de ces paramètres qui les caractérisent.

I.1.1. Choix des paramètres caractérisant une lettre :

A- Critères de choix :

Cette étape du choix des paramètres a donc pour but de construire un vecteur caractéristique décrivant, globalement, chaque lettre, cette description étant basée sur l'extraction de leurs primitives significatives.

Cependant, il n'y a pas de théorie pour guider le choix de ces primitives. Ceci reste, pour l'instant, dans le domaine de l'art plus que dans celui de la science. On peut toutefois formuler les critères suivants à propos des primitives :

- **Discriminabilité** : Une bonne primitive doit avoir des valeurs significativement différentes pour des caractères appartenant à des classes différentes.
- **Fiabilité :** Une bonne primitive doit avoir des valeurs très similaires (à faible variabilité) pour les formes d'une même classe.
- **Indépendance :** Dans un ensemble de primitives choisies, une primitive quelconque ne

doit pas dépendre d'une autre primitive.

- **Nombre :** Le coût, la complexité et les exigences en temps de calcul d'un système de reconnaissance croissent avec le nombre de primitives utilisées.

Il faut donc choisir les meilleures primitives adéquates qui permettent de diminuer leur nombre tout en satisfaisant une probabilité d'erreur acceptable, fixée au préalable.

B- Paramètres choisis:

Le système de reconnaissance de caractère utilise la notion de classification du nombre de caractères à identifier en autant de classes. Dans notre application, nous ne nous intéresserons pas aux technique de traitement d'image citées dans les sections précédente (numérisation de données et leurs prétraitement afin d'extraire les primitives). Nous prendrons comme bases de données des caractères issus de Microsoft Word.

Par ailleurs, nous avons choisit trois primitives : Chaque lettre est caractérisée par trois paramètres qui sont:

- Le rapport de la surface noire sur la surface blanche de la lettre (espace occupé par la lettre).
- Le nombre de segments linéaires contenus dans la lettre: représente le nombre de droites qui forme une lettre exemple (la lettre **(A)**, contient trois segments reliés entre eux.
- Le nombre de boucles dans la lettre : C'est l'ensemble des cercles ou bien des anneaux qui peuvent exister dans une lettre exemple. (la lettre **(B)** contient deux boucles).

I.1.2. Source des données:

Pour tester le bon fonctionnement du classifieur à base de réseau de neurones, Nous avons sélectionné trois lettres, A B et C. Puis nous avons caractérisé chacune de ces lettres par trois valeurs numériques qui correspondent aux attributs précédemment définis.

Nos données (lettres) sont prélevées de Microsoft Word version 2010 avec les caractéristiques suivantes:

- Type de police: Times New Roman
- Taille de police: 72
- Texte Gras.
- Lettres majuscules.

A B C

Figure I.1. Représentation des les lettres A, B, C sur le Word

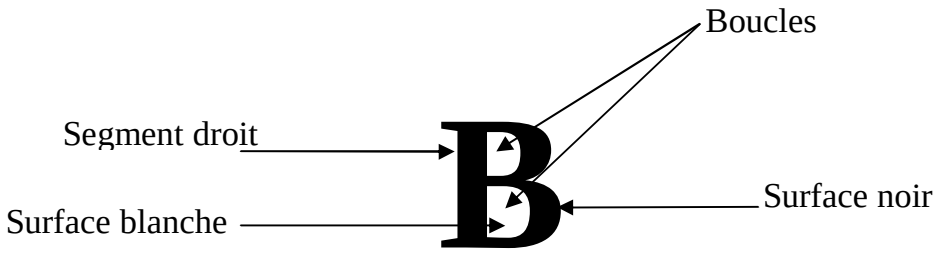


Figure I.2. Présentation de la lettre B avec ses paramètres

I.2. Représentation des données:

Chaque lettre est représentée par un vecteur d'entrée contenant les trois valeurs des paramètres qui la définissent, à savoir le rapport de surface noir/blanc, le nombre de segment linéaires et le nombre de boucles. Sachant qu'à chaque entrée du classifieur correspond une sortie cible (classe), nous devons aussi définir le vecteur cible de chaque lettre.

Le tableau suivant représente les lettres, leurs paramètres choisis ainsi que les cibles (classes) correspondant à chaque lettre.

Lettres	**La surface N/B**	**Nombre de segment droits**	**Nombre de boucles**	**Cible (classe) Targets**
A	0.7	3	0	**001**
B	0 .9	1	2	**010**
C	0.6	0	1	**100**

Tableau I.1 : Tableau représentatif des lettres A, B, C

I.3. Conception du réseau (Méthodologie) :

Les réseaux de neurones sont des méthodes statistiques appropriées pour résoudre les problèmes non linéaires. Étant donné la nature non linéaire de notre problématique de reconnaissance des lettres, Les réseaux de neurones semble être une méthode pouvant

donner des résultats satisfaisants.

I.3.1. Type et Architecture du réseau :

Il existe différentes manières d'organiser les neurones, C'est ce qui définit l'architecture et le modèle du réseau. Dans notre cas le réseau choisi est du type rétro-propagation de l'erreur qui permet de résoudre des problèmes de classification non linéaire.

A-Données d'entrée et sorties : Un réseau de neurone se comporte en réalité comme une boîte noire qui reçoit des données d'entrées et les sorties sont comparées aux données cibles préalablement définies. Par conséquent les sorties désirées ou cibles sont rangées dans une matrice de 1 et 0.

B- Nombre de neurones : Chaque lettre possède trois paramètres caractéristiques par conséquent on utilisera trois neurones pour la matrice d'entrée.

C- Nombre de couches : Le réseau utilisé et de type rétro-propagation ; ce dernier et généralement construit par une couche d'entrée et au minimum une couche cachée et une sortie qui fournira les trois classes présentées dans le tableau précédant.

D- Les fonctions d'activation : En général il est préférable dans le réseau rétro-propagation d'utiliser la fonction sigmoïde pour la couche d'entrée et la fonction linéaire pour la sortie. La figure suivante explique la structure correspondante :

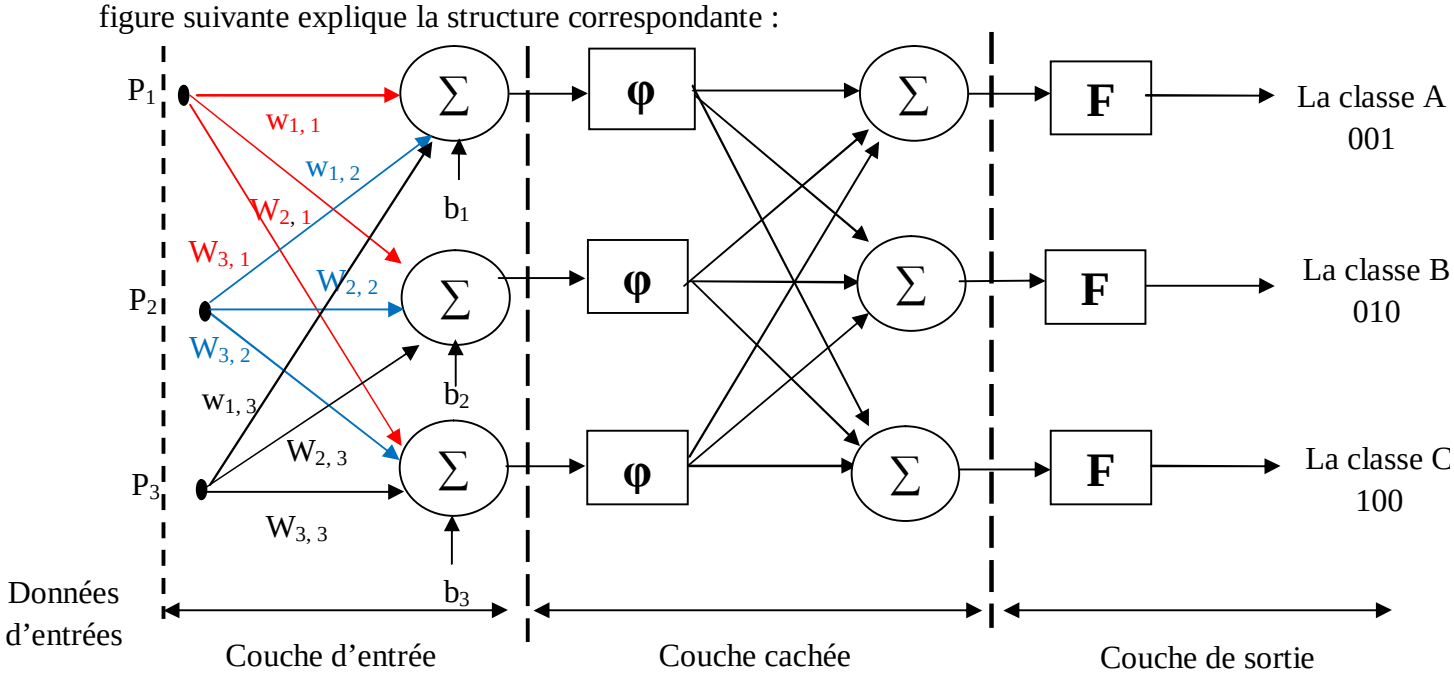


Figure I.3 : Architecture du réseau de neurone utilisé

I.4. Structure du programme:

Le programme est réalisé sous MATLAB avec les commandes résumées dans le tableau suivant :

Commandes	***Significations***
Targets	*Cibles du réseau*
Input	*Les entrées du réseau*
newff	*Fonction de création d'un modèle réseau de neurones type rétro-propagation de l'erreur*
tansig	*Fonction d'activation sigmoïde*
purelin	*Fonction d'activation linéaire*
traingd	*Algorithme d'apprentissage descente de gradient*
net	*L'objet réseau*
tr	*Informe sur le progrès d'apprentissage*
train	*Fonction d'apprentissage*
lr	*Taux d'apprentissage*
epochs	*Nombre d'époque (itération)*
Sim	*Fonction de simulation du réseau*
Out	*La sortie obtenue du réseau*
testInputs	*Les données test*
outtest	*La sortie obtenue des données test*
confusion	*Evalue le taux de reconnaissance*
mse	*Erreur quadratique*

Tableau I.2. Tableau des commandes MATLAB

Pour que le programme puisse fonctionner correctement il ya des étapes à respecter :

A-Préparation des données :

Dans notre programme on saisi la matrice des vecteur d'entrée ***« Input »,*** et la matrice de données cibles ***« Targets ».***

B- Création de l'objet réseau:

Nous avons utilisé la fonction **« *newff* »** pour la création d'un modèle réseau de neurones type rétro-propagation de l'erreur, qui prend en argument les matrices ***« Input », « Targets »***

avec trois neurones d'entrée, ainsi que la fonction d'activation ***« tansig », « purelin »*** et l'algorithme d'apprentissage ***« traingd ».***

net=newff(Input,Targets,3,{'tansig','purelin'},'traingd');

C-Initialisation des poids et biais :

La première couche prend ses poids des données d'entrées, la seconde prend ses poids de la couche précédente et la dernière est celle de sortie des résultats. Les poids sont mis à jours au fur et à mesure du processus d'entraînement avec la fonction de transfert appropriée ***« tansig ».***

D-Entrainement ou apprentissage du réseau :

L'application divise aléatoirement les vecteurs d'entrées et cibles en trois ensembles :
60% des données pour l'entrainement du réseau, 20% des données pour valider que le réseau se généralise afin qu'il cesse de s'exercer avant que le réseau se déstabilise (l'overffiting) qui infeste beaucoup l'optimisation et l'algorithme d'apprentissage ; ces données servent d'indice de mesure de performance durant l'étape d'apprentissage. Enfin, 20% de données restantes sont employées pour effectuer des essais indépendants de la généralisation de réseau.

E-Lancement de l'apprentissage :

On lance l'apprentissage du réseau par la fonction ***[net,tr]=train(net,Input,Targets)*** qui prend comme argument **« *net* »** l'objet réseau, les matrices **« *Input* »** et **« *Targets* ».** Cette fonction retourne le paramètre **« *tr* »** qui nous informe sur les progrès du réseau.
Durant l'apprentissage on tient compte des paramètres suivants :

- Le taux d'apprentissage **lr** dont la valeur est comprise entre (0 et 1). Quand il est à une valeur élevé l'algorithme d'apprentissage devient instable, quand il est à une valeur faible l'algorithme d'apprentissage prend du temps pour converger.
- **Epoch** indiquent que l'entrainement s'arrête lorsque l'algorithme atteint le nombre epoch,

F- Simulation du réseau :

A la fin de la phase d'apprentissage, on simule tout le réseau avec la fonction **« *sim* », *out=sim(net,Input)*** qui prend en argument, **« *net* »** objet réseau, **« *Input* »** matrice d'entrée. Ainsi la simulation des données test par la même fonction **« *sim* » *(outtest=sim(net,testInput).*** Les données réservées pour le test vont permettre d'évaluer d'une manière indépendante la performance du réseau.

La fonction **« *sim* »** va générer les valeurs obtenues du réseau qui seront comparées aux données cibles.

G- Calcul d'erreur:

On compare la sortie obtenue avec la cible désirée par une soustraction entre ***« Targets »*** et ***« out ».***

H- Calcul de la matrice de confusion:

Cette matrice permet d'afficher le pourcentage de classification, par la fonction ***« confusion », [c,cm,ind,per]=confusion(Targets,out);*** cette fonction prend comme argument ***« Targets »*** les données cible et **« *out* »** la réponse du réseau et nous retourne les paramètres suivants :

- **c** : Valeur de confusion qui représente la valeur des échantillons mal classifié.
- **cm**: Matrice de confusion.
- **ind**: Indice de 1 dans la matrice.
- **per** : Pourcentages (de valeurs classées correctement).

I.4.1. Discussion et interprétation des résultats :

Après définition de la problématique, notre programme doit aller chercher toutes les données, aussi bien celles permettant l'apprentissage que celles utilisées pour tester la validité de notre réseau. Chaque neurone représente une catégorie, quand un vecteur d'entrée de la catégorie approprié est appliqué au réseau, le neurone correspondant doit produire un (1) et les autres (0).

Pour avoir une reconnaissance parfaite du réseau on a testé l'influence du paramètre epoch, Au début de l'exécution de notre programme on a remarqué que les résultats obtenus ne sont pas stable pour plusieurs valeurs d'epochs de (10 à 20), malgré le fait que nous avons effectué un nombre de simulation très élevé mais le taux de reconnaissance varie entre 33.3%, 66.7% et un taux parfait de 100% à l'epochs 20 mais après un très grand nombre de simulations.

Pour améliorer nos résultats on a pensé à fixer les valeurs des poids d'entrées à des très petites valeurs, puis on a refait la simulation du programme, avec une valeur d'epochs 100, lr=0.1, les résultats obtenus sont affichés dans le tableau suivant:

Remarque: Le nombre de simulations de toutes les applications a dépassé 30 simulations.

nombre de simulation	Nombre d'itérations	Erreur (mse)	Pourcentage de classification (%)		Lettre classée
			Taux de reconnaissance	Taux d'échec	
1	2	$10^{-1.2}$	66.7	33.3%	A, B
2	8	$10^{-1.8}$	100%	0%	A, B, C

Tableau I.3. Tableau des résultats obtenus pour trois lettre A, B, C

Nous avons un taux de reconnaissance de 100% pour 8 itérations (classement des trois A, B, C); le taux de reconnaissance est de 66.7% pour 2 itérations (classement de deux lettres A, B).

En parallèle nous avons analysé l'erreur mse commise à chaque itération, on a constaté que cette erreur diminue et passe de $10^{-1.2}$ à $10^{-1.8}$ pour une reconnaissance parfaite.

Lors du passage d'un taux de reconnaissance de 66,7% à un taux de 100% le réseau a effectuer une mise à jour des poids initiaux en calculant la matrice d'erreur qui est la différence entre la sortie désirée et la sortie obtenue.

Matrice d'erreur :

$$\begin{bmatrix} 0.4893 & 0.1779 & 0.0258 \\ 0.6509 & 0.3153 & 0.1425 \\ 1.3238 & 0.2495 & 0.2848 \end{bmatrix}$$

Les erreurs commises sur chaque lettre n'ont pas encor atteint la valeur nulle d'où la nécéssité d'une mise à jour des poids pour avoir une reconnaissance parfaite de 100%.

Matrice de confusion pour le taux de 100% représenté dans la figure suivante:

Figure I.4.Matrice de confusion

Cette figure montre différents pourcentages d'erreurs qui se sont produites pendant la formation. Les cellules diagonales dans chaque table montrent le nombre de cas qui ont été correctement classifié, et les autres cellules montrent les cas mal classifiés. La cellule en bas à droite montre tous les pourcentages de cas correctement classifiés (100 pour cent) et tout les pourcentages de cas mal classifies (0 pour cent).

Ces résultats montrent que l'identification est parfaite pour une division de données par 60%, 20%, 20% ce qui donne une donnée par ensemble car on a que trois données d'entrée, l'ensemble peut avoir plusieurs données dans le cas où le nombre de classe est important.

- **Performance du classifieur:**

La figure suivante montre une parcelle de terrain des erreurs quadratique mean squard error (mse) des données d'apprentissage (courbe bleu), de validation (courbe verte) et de test (courbe rouge) en fonction de nombre d'itération (Epochs).

La courbe de validation signifiant que le réseau se généralise montre que l'erreur quadratique (mse) commence par une large valeur $10^{0.2}$ et diminue vers une petite valeur $10^{-1.8}$ à l'epoch 8, autrement dit ceci montre que le réseau apprend. L'apprentissage sur les données d'entrainement continu pendant que cet apprentissage réduit l'erreur de validation, après que le réseau ait mémorisé l'ensemble d'entrainement, la formation est arrêtée. Cette technique évite automatiquement la déstabilisation du réseau. De plus on remarque que la courbe test S'évolue de la même manière que la courbe de validation signifiant que les résultats sont

satisfaisants.

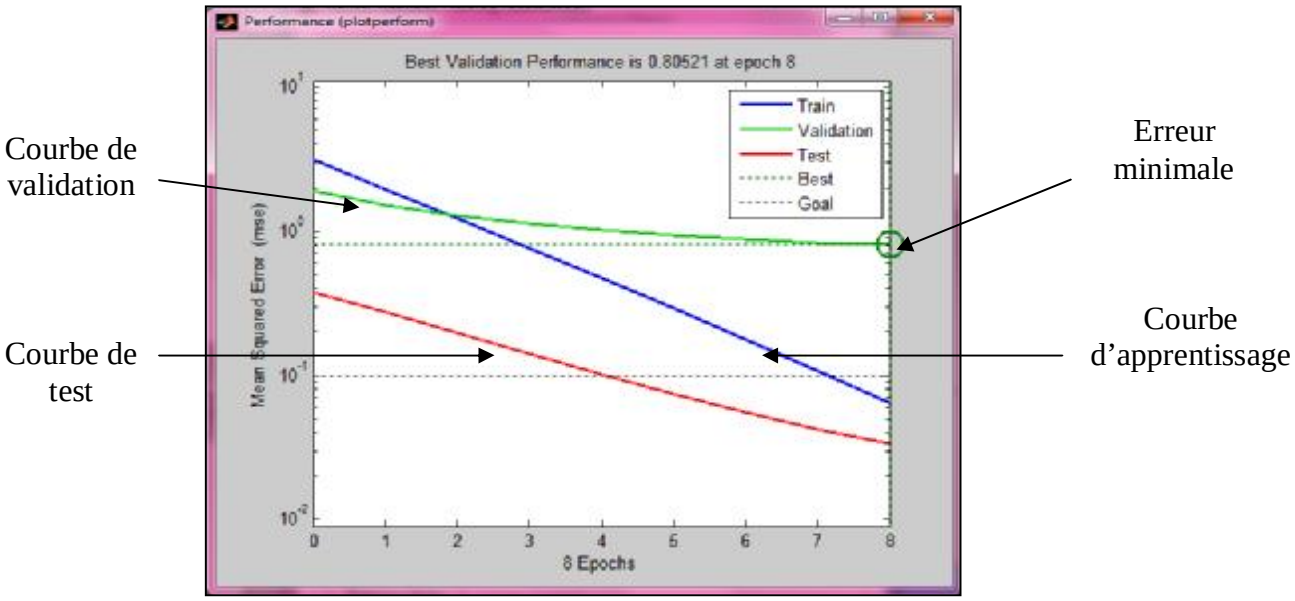


Figure I.4. Courbe de la performance du réseau

II. Généralisation:

II.1. Reconnaissance de six lettres:

La reconnaissance de caractères ne se limite pas à trois lettres alphabétiques.

En augmentant le nombre de lettres à 6 lettres comment va devenir le comportement du classifieur ?

Le plan de l'application est le même sauf que le nombre de valeurs (entrées/ sortis) augmente. Nous avons rajouté trois autres paramètres pour définir les lettres: Le nombre de croisements, le nombre de courbures et la taille des lettres.

Les résultats obtenus après simulation sont affichés dans le tableau suivant:

Nombre de simulation	Nombre d'itérations	Erreur (mse)	Pourcentage de classification (%)		Lettres classées
			Taux de reconnaissance	Taux d'échec	
1	34	$10^{-1.05}$	50%	50%	D, E, F
2	48	$10^{-1.25}$	50%	50%	D, E, F
3	52	$10^{-1.15}$	50%	50%	D, E, E

Tableau I.4. Tableau des résultats obtenus pour six lettres A, B, C, D, E, F

A partir des résultats du tableau en remarque une stabilité totale du taux de reconnaissance qui vaut 50%, avec une reconnaissance de trois lettres alphabétiques et une erreur carrée qui passe de $10^{-1.05}$ à $10^{-1.15}$. Le nombre d'itération est passé de 34 à 52 itérations.
On vient de constater qu'en augmentant les valeurs d'entrées le classifieur prend du temps pour reconnaitre les lettres 34 à 52 itérations avec un taux juste de 50%.

II.2. Généralisation totale:

Si on généralise notre application pour l'ensemble des lettres (26 lettres) et les chiffres de (0) à (9), on a besoin:

- d'une base de données de grande dimension, qui doit être enregistrée et appeler dans MATLAB lors de la constitution du programme, comme on a besoin aussi d'un programme de calcul des valeurs numérique des paramètres, et de faire appel au technique de traitement d'image ce qui n'est pas l'intérêt de notre travail.
- Le choix du nombre de couche et le nombre de neurones par couche sont la première étape à faire pour pouvoir utiliser les fonctions d'apprentissage du module de réseau de neurone de MATLAB. En général il n'y a pas de méthode précise pour trouver le nombre de couches ou le nombre de neurones par couche. Il faut tester un maximum de possibilités jusqu'à trouver celle qui donne le meilleur résultat possible.
- augmenter le nombre de primitives extraites en plus du nombre de segments droits, de boucle et de la surface noir/blanc. le classifieur peut confondre les primitive de même catégorie pour plusieurs lettre, comme exemple la lettre A, et F possèdent trois segments chacune qui devraient être classées dans deux classe différente c'est ainsi qu'il faut définir d'autre primitives pour les séparer.
- prendre en considération le nombre d'itérations (epoch) qui doit être élevé, ainsi que le temps d'exécution et la mémoire occupé par l'application.

Dans le cas où on n'obtient pas des résultats satisfaisant on effectue les étapes suivantes :

- Remettre à zéro les poids et biais initiaux du réseau à de nouvelles valeurs.
- Augmenter le nombre de neurones.
- Augmenter le nombre de vecteur d'entrainement.
- Augmenter le nombre de valeurs d'entrées si l'information appropriée est disponible.
- Essayer un algorithme d'apprentissage différent.

Conclusion :

Les réseaux de neurones ont amené une solution satisfaisante pour l'estimation de la Classification des trois lettre A, B, C avec un résultat total de 100%. Ces résultats ouvrent le champ à d'autres tests pour estimer la classification pour toutes les lettres. Dans le cas de la reconnaissance de 6 lettres alphabétiques, la reconnaissance est de 50% ces résultats explique la complexité de la reconnaissance lors de l'augmentation du nombre de caractères et amène le classifieur à une confusion entre les caractères, alors il faut augmenter le nombre d'attributs qui caractèrise le caractère bien distinct l'un de l'autre.

Cela nous a permis d'enfin voir une utilisation concrète des principes sur la reconnaissance de forme.

Nous avons présenté dans ce chapitre notre modélisation pour un système de reconnaissance. Notre approche est basée sur un classifieur neuronal. Malgré un taux de réussite de 100% que nous estimons encourageant, notre système est loin d'atteindre la perfection car le nombre de lettres n'est pas important ainsi que leurs paramètres.

Conclusion générale:

Dans le but de la reconnaissance de caractères "Lettres alphabétiques" à base des réseaux de neurones, nous avons présenté des généralités sur la reconnaissance de caractères, y compris les phases de traitement de données, avant de passer à l'étape classification.

Nous avons définit les réseaux de neurones ainsi que les types d'apprentissages, nous avons abordé particulièrement l'algorithme de rétro-propagation de l'erreur qui a été utilisé dans notre application.

On a terminé par la réalisation d'un classifieur de lettres à base de réseaux de neurones sous MATLAB, et enfin on a montré que le classifieur neuronal arrive facilement à reconnaitre parfaitement trois lettres (A, B, C) après deux simulations avec un taux de reconnaissance 100%, ces résultats sont très satisfaisants. On a pensé de faire une généralisation pour six lettres (A, B, C, D, E, F), ceci nous a amené à conclure que le taux de reconnaissance diminue à 50% avec une reconnaissance de trois lettres seulement sur six. Les résultats sont logiques car notre classifieur trouve des difficultés à distinguer plusieurs lettres à la fois, ce qui nécessite d'ajouter des paramètres d'entrés pertinents caractérisant les lettres a fin d'éviter la confusion de la classification. Dans le cas d'une généralisation sur tous les caractères (chiffres, lettres, symboles) il faut bien une technique de mesure des paramètres ainsi que des logiciels de traitement de documents pour l'assurance d'une base de donnée bien définit.

Les résultats obtenus pendant notre application peuvent être améliorés si nous avons utilisé des bases de données bien précises, et des techniques de traitement de données citées dans le premier chapitre qui vont rendre les résultats meilleurs et fiables.

Référence et bibliographie

Thèses :

- Benmahjoub Khalida, " Prévision de précipitation météorologique par réseaux de neurones", thèse magister, département électronique UMMTO, année 2007.
- Chikhi Kamel, "Reconnaissance de visages à base des HMMS standards", thèse ingénieur, département électronique, UMMTO, année 2007.
- Khadraoui Mouhamed, "Application des réseaux de neurones pour la classification des visages", département électronique, UMMTO, année 2007.
- Lfticene Lounis, "Application des réseaux de neurones à fonction radiale de base pour la classification d'arythmie cardiaque ", thèse ingénieur, département électronique, UMMTO, année 2007. [10], [11], [12], [13], [14], [15], [16], [17], [18].
- Melanie Lemaitre, "Approche markovienne bidimensionnelle d'analyse et de reconnaissance de documents manuscrits", thèse de doctorat, université de paris, année 2007. [1], [2], [3], [4], [5], [6] , [7], [8], [9].
- Bernard Gosselin, "Application de réseaux de neurones artificiels à la reconnaissance de caractères", thèse doctorat, faculté polytechnique de Mons, année 1996.
- G.Dreyfus, "Les réseaux de neurones", laboratoire électronique, école supérieur de physique et chimie, année 1998.
- Emile Poisson, Christian Viard-Gaudin, "Réseaux de neurones à convolution: Reconnaissance de l'écriture manuscrite non contrainte", école polytechnique de l'université de Nantes-France.
- A.Belaid et Y.Belaid, "reconnaissance automatique de l'écriture et du document", InterEditions, année 1992.
- Aziz rebial, "Une approche hybride pour la reconnaissance d'écriture arabe manuscrite arabe", thèse magister, université Constantine, année 2007.
- Abdelhak boukhrona, "Reconnaissance de Caractères Imprimés Omni-fonte", 3ème conférence international: SETIT, Tunisie, Mars 2005.
- Brahim Farou[1]. Samir Hallaci[1]. Hamid Seridi[2], "Système Neuro-Markovien pour la reconnaissance de l'écriture manuscrite arabe à vocabulaire limité ", [1] Université Guelma, [2] université Reims.
- Abdelhak Boukhrona," Reconnaissance de Caractères Imprimés Omni-fonte", 3ème conférence international : SETIT, Tunisie, Mars 2005.

Ouvrages:

- Presse universitaires "Les réseaux de neurones", Grenoble 1994.
- Teghem Jacques, "Optimisation en recherche opérationnelle", Paris Lavoisier 2002 ;
- M.Moukhtari, " MATLAB 5.2, 5.3 et simulink pour étudiant et ingénieur ", Edition Berlin Springer 2000.
- Personnz leon, "Réseaux de neurones formels pour la modélisation, commande et classification", Edition Paris CNRS 2003.

Logiciel:

- MATLAB version 7.10.0. (R2010a),

Sites web:

http://www.livrespourtous.com

http://www.touslescours.fr

http://www.pdfqueen.com

http://fr.wikipedia.org

http://www.bepdf.com

http://www.yopdf.com

http://www.memoireonline.com