République algérienne démocratique et populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université Ferhat Abass-Sétif

THÈSE

Présenté à la Faculté des Sciences
Département d'Informatique

En vue de l'obtention du diplôme de

Doctorat en science

Option : Informatique

Par

Salima Nebti

Thème

Reconnaissance de Caractères Manuscrits par Intelligence Collective

Soutenu le : O7/mars/2013

Devant le jury

Président :	**Pr**	**Abdelouahab**	**MOUSSAOUI**	**Prof Université Ferhat Abbas Sétif**
Rapporteur :	**Dr**	**Abdallah**	**BOUKERRAM**	**M.C Université Ferhat Abbas Sétif**
Examinateurs :	**Pr**	**Okba**	**KEZAR**	**Prof Université de Biskra**
	Pr	**Hamid**	**SERIDI**	**Prof Université de Guelma**
Membre invité :		**Mohamed**	**TOUAHRIA**	**Prof Université Ferhat Abbas Sétif**

Remerciements

Cette thèse fut l'opportunité d'un travail de recherche attirant qui trouve son achèvement dans ce document.
J'exprime ma profonde reconnaissance à Monsieur Abdallah Boukerram, Maître de conférences à l'Université de Sétif qui a accepté de diriger ce travail. L'accueil de Mr Boukerram, son soutien et son ouverture d'esprit ont largement contribué à la réalisation de ce travail. Ses conseils, ainsi que son optimisme ont été précieux. Je tiens à saluer sincèrement Mr Boukerram. Pour l'intérêt qu'il a porté à la lecture de ce manuscrit.

Je tiens a remercier Mr Abdelouahab Moussaoui, Professeur à l'Université de Sétif, qui m'a fait le grand honneur d'accepter la présidence du jury.

Mes remerciements s'adressent également à Mr Okba Kezar Professeur à l'Université de Biskra, Hamid Seridi Professeur à l'Université de Guelma et Mr Mohamed Touahria Maître de conférences à l'Université de Sétif, qui m'ont fait l'honneur d'accepter de juger ce travail.

Je voudrais exprimer ma plus grande reconnaissance à mes parents, mes frères et sœurs pour leurs soutiens.

PUBLICATIONS DE L'AUTEUR:

S.Nebti, S.Meshoul, M.Batouche, "Predator-Prey Optimiser for unsupervised Clustering in image segmentation", *Proceedings of the 2005 International ARAB conference on Information technology (ACIT' 2005). Pp 317-324. Amman-Jordan, 2005.*

Salima Nebti, "Une approche Coopérative Co-évolutionnaire pour la segmentation d'images couleurs", *The Third international summer school on signal processing and its applications, (I3SPA) 08- 12 July Jijel- Algeria, 2006.*

S.Nebti, S.Meshoul, M.Batouche, " segmentation d'images par une approche évolutionnaire prédateur-proie symbiotique", 5 ème séminaire national en informatique de biskra (SNIB 2006), 2006.

S.Nebti, S.Meshoul, M.Batouche, " A Socio-biological Algorithm for Color Image Segmentation", *the 3rd International Symposium on Image Video Communications over fixed and mobile networks (ISIVC2006) ,Yasmine Hammamet-Tunisia (September 13-15,2006)*

S.Nebti, S.Meshoul, M.Batouche," A Co-evolutionary Approach for Snake Based Contour detection", *Métaheuristiques 2006 (Meta'06), Yasmine Hammamet-Tunisia (November 02-04, 2006).*

S.Nebti, " Colour Image Segmentation using Coevolutionary Particle Swarm Optimisation", *Journées Ecole Doctorale & Réseau de recherche en Sciences et Technologies de l'information JED'07 – Annaba, Algérie (Mai 27- 28, 2007).*

Salima Nebti, " Une approche Rapide basée Snake pour la détection de contour", *The 4th international summer school on signal processing and its applications, (ISSSPA '07), Boumerdes, Algérie, (30 Juin- 04 Juillet 2007).*

S.Nebti, S.Meshoul. Predator Prey Optimization For Snake based Contour Detection. *International Journal of Intelligent Computing and Cybernetics, Vol 2 No 2, 2009.*

S.Nebti, A.Boukerram, A Particle Swarm Optimization based approach for Handwritten Digits Recognition. *Conférence Internationale des Technologies de l'information et de la communication CITIC'09. Algeria. 2009.*

S.Nebti, A.Boukerram. Handwritten digits recognition using an hybrid approach for MLP training. *International Arab Conference on Information Technologie. ACIT'09. Yemen 2009.*

S.Nebti, A.Boukerram. Handwritten Digits Recognition using a Symbiotic Algorithm for Training MLP/RBF Neural Networks. *International Conference On Industrial Engineering and Manufacturing ICIEM'10, May, 9-10, 2010, Batna, Algeria*

S.Nebti, A.Boukerram, Handwritten Digits Recognition Based on Swarm Optimization Methods. *The second international conference on networked digital technologies (NDT'2010), Prague, Gzech Republic.2010.*

S.Nebti, A.Boukerram. An Improved Radial Basis Function Neural Network based on a Cooperative Co-evolutionary Approach for Handwritten Digits Recognition. *International Conference On Machine Web Intelligence ICMWI'10, october, 3-5, 2010, Alger, Algeria.*

Salima Nebti, Abdellah Boukerram, "Use of Nature-inspired Meta-heuristics for Handwritten Digits Recognition", *International Journal of Computational Linguistics Research, V 1, N° 1, Page 30-40, 2010.*

S.Nebti. Color Image Segmentation based on Swarm Optimisation Methods. Lecture Notes in Computer Science, Volume 6377/2010, 277-284, ICICA'2010.

Salima Nebti, Abdellah Boukerram. "Handwritten characters recognition based on nature-inspired computing and neuro-evolution". Applied Intelligence, The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies, ISSN 0924-669X, Vol 22, N° 2, March 2012.

SOMMAIRE

Chapitre 4 : L'extraction des attributs caractéristiques

Chapitre 5 : La classification

Chapitre 6 : La combinaison de classificateurs

Chapitre 7 : L'intelligence collective et l'intelligence en essaim

Chapitre 8 : Les réseaux de neurones

Chapitre 9 : Les réseaux de neurones évolutionnaires

Chapitre 10 : La Reconnaissance de caractères par des classificateurs statistiques basés sur l'intelligence en essaim

ANNEXE

LISTE DES FIGURES

LISTE DES TABLES

LISTE DES ABREVIATIONS

ABC	L'optimisation par colonie d'abeilles artificielles
ACO	L'optimisation par colonie de fourmi
AE	Algorithme évolutionnaire
ANNs	Réseau de neurones artificiel
BA	L'algorithme des abeilles
BKS	Méthode du comportement des connaissances d'espace
Bp	La méthode d'apprentissage de la rétro-propagation de gradient
C4.5	algorithme d'apprentissage d'arbre de décision
CART	algorithme d'apprentissage d'arbre de décision
CENPARMI	d'une base de données des chèques arabes manuscrits
CNN	Le réseau de neurones convolutionel
DAI	L'intelligence artificielle distribuée
FFNNs	les réseaux de neurones feed forward
GAs	les algorithmes génétiques
HMMs	Le modèle de markov caché
IA	L'intelligence artificielle
IAD	L'intelligence artificielle distribuée
ID3	algorithme d'apprentissage d'arbre de décision
***k*-NN**	Le classificateur du plus proche voisin
LeNet-5	Un réseau convolutionnel conçu pour la reconnaissance de chiffres manuscrits
LM	Méthode d'apprentissage des réseaux de neurones : Levenberg-Marquardt
MBO	L'optimisation par mariage des abeilles de miel
MLP	Le perceptron multi couche
MMC	le modèle de markov caché
MNIST	la base de données des chiffres arabes manuscrits
NEAT	une méthode, **N**euro **E**volutionnaire d'**A**ugmentation des **T**opologies des réseaux de neurones.
NIST	Une base de données : contient des caractères isolés et du texte écrits.
NISTSD19	Une base de données : contient des caractères isolés
PCA	Analyse de composantes principales
PNN	Le Réseau de neurones probabiliste
PSO	La méthode d'optimisation par essaim de particule
QNA	Méthode d'apprentissage des réseaux de neurones : quasi-Newton algorithm
RBF	Fonction à base radiale
RBFNNs	Les réseaux de neurones à base radiale
RN	réseau de neurones
RNN	réseau de neurones récurrent
SANE	L'évolution symbiotique de l'architecture des RNs
SCG	Méthode d'apprentissage des réseaux de neurones : scaled conjugate gradient
SDNNs	les réseaux de neurones à déplacement spatial
SMA	Les systèmes Multi-agent
SOM	Carte auto-organisatrice
SRN	réseau récurrent simple
SSE	la somme des erreurs quadratiques
SVM	Machine à vecteur support
SVs	les vecteurs supports

Première partie

La reconnaissance de caractères manuscrits

Chapitre 1

INTRODUCTION GENERALE

1.1 Introduction

Les améliorations faites dans la technologie des ordinateurs se sont appuyées sur la venue des techniques numériques permettant la communication avec l'ordinateur par écriture manuscrite. Parallèlement, le développement rapide des systèmes de gestion des entreprises a permis la mise en œuvre des systèmes de la gestion des documents manuscrits. La reconnaissance de l'écriture manuscrite a attiré l'attention d'un grand nombre de chercheurs depuis plusieurs décennies et reste un champ de recherche très actif dû à ses nombreuses applications [A.Vinciarelli, 2002] comme la lecture automatique des adresses postales [Y.Lu et al, 2002], [A.Brakensiek et al, 2003], des chèques bancaires [Q. Xu et al, 2001], [N. Gorski, 1999], la lecture des formulaires [T. Hirano et al, 2001] et bien d'autres.

Les travaux de cette thèse s'inscrivent dans le domaine de la reconnaissance des formes, plus particulièrement : la résolution du problème de la reconnaissance de caractères manuscrits par des méthodes natives du domaine de l'intelligence collective.

L'intelligence collective est un axe de recherche qui a fait l'objet d'investigation dans de nombreuses disciplines. L'origine de la résolution collective de problèmes est liée au domaine de l'intelligence artificielle distribuée IAD connue depuis longtemps et qui s'est engagée dans des approches plus sophistiquées qui sont les systèmes multiagents.

La résolution collective de problèmes est un domaine de la recherche qui s'intéresse aux interactions entre des entités autonomes et à l'émergence au niveau supérieur de l'ensemble d'un système complexe. Cette technique porte sur la manière de diviser un problème entre des entités relativement simples mais capables de s'auto organiser afin d'accomplir d'une façon collective un but commun. Cette vision distribuée de problèmes a suscité l'attention croissante des chercheurs pour la qualité de ses applications et ses avantages tels que le parallélisme, la vitesse, flexibilité, …etc.

L'intelligence collective s'appuie largement sur des méthodes dérivées de l'étude des sociétés de groupes vivants ou des comportements animaliers. C'est depuis l'apparition de la Vie Artificielle comme domaine que des auteurs ont commencé à s'incliner vers l'étude des capacités collectives de la résolution de problèmes de certaines espèces d'animaux sociaux en abordant simultanément les applications qui pouvaient en être dérivées en informatique comme en robotique [A.Drogoul, 1993]. Ces systèmes se sont généralement inspirés des études issues de la biologie, la sociologie, la psychologie sociale, l'éthologie, les sciences cognitives et bien d'autres. Les insectes sociaux représentent le meilleur modèle naturel de coopération d'agents réactifs pour achever des tâches complexes. L'interaction de ces agents simples permet l'émergence de phénomènes complexes

et ainsi l'émergence d'une intelligence collective. C'est la capacité des groupes d'agents simples à s'auto organiser pour produire un tout dont le comportement pouvant être qualifié d'intelligence [J.Koh et al, 1995]. De plus, ces systèmes possèdent l'avantage d'avoir des schémas d'interaction sophistiqués tels que : la coopération (la résolution d'un but commun par plusieurs agents), la coordination (l'organisation de la résolution d'un problème pour éviter les effets nuisibles et ainsi permettre un bon processus de résolution) [E. Izquierdo-Torres, 2004]. L'intelligence en essaim constitue un domaine à part entière de l'intelligence collective ou l'intelligence artificielle distribuée. L'intérêt grandissant de l'intelligence en essaim est lié à ses propriétés : robustesse aux pannes, fonctionnement décentralisé, émergence des solutions globales, …etc.

Le comportement collectif des insectes sociaux comme les abeilles, les fourmis, les guêpes, …est une source d'inspiration forte pour le domaine de la reconnaissance de caractères telle que nous le réaffirmons dans cette thèse. Nous l'abordons selon une démarche de classification basée sur les réseaux de neurones notamment le perceptron multi couche et les réseaux de neurones à base radiale. Nous allons montrer que les algorithmes issus de ce domaine très vaste peuvent être utiles dans nos applications plus particulièrement les méthodes d'intelligence en essaim qui font parties du domaine de l'intelligence artificielle distribuée ; ces méthodes sont caractérisées par une forte puissance d'exploration et d'exploitation de l'espace des solutions.

1.2 La reconnaissance de caractères :

La reconnaissance de caractères manuscrits est la transcription des données manuscrites au format numérique pour obtenir sa signification. C'est une tâche difficile due à la grande variabilité des styles d'écriture et elle est plus difficile au niveau de la reconnaissance du texte où la reconnaissance passe par plusieurs processus qui sont : la segmentation du texte en phrases, les phrases en mots et les mots en lettres pour leur reconnaissance. Dans tous ces processus, un dictionnaire contextuel joue un rôle essentiel pour interpréter correctement les différentes prétentions trouvées pendant la reconnaissance.

Un système de la reconnaissance de caractères nécessite trois étapes principales notamment : *Le prétraitement* dont l'objectif est l'extraction des caractères bruts sans bruit, ni texture en procédant par des techniques de Binarisation, de la réduction de bruit, de la normalisation, et de la squelettisation. *L'extraction de caractéristiques* est une étape cruciale dont l'objectif est d'extraire les vecteurs d'attributs constituant les informations pertinentes permettant la distinction des caractères. *La classification* : est l'étape finale dont l'objectif est la décision. Elle se fait par des techniques de classification ou d'apprentissage [G. Vamvakas, 2009], [G. Vamvakas, 2010].

Les systèmes de la reconnaissance de caractères peuvent être distingués en deux catégories : les systèmes holistiques et les systèmes analytiques. Dans la première catégorie, l'unité à reconnaître est le mot, l'extracteur des attributs caractéristiques le plus employé est le calcul des profils horizontaux et verticaux. Les méthodes de cette catégorie sont efficaces puisqu'elles ne nécessitent pas la segmentation du mot et ainsi préviennent les erreurs de segmentation. Néanmoins, la taille des données d'apprentissage et du lexique employés sont énormes. En effet, elles nécessitent une plus longue durée d'apprentissage. Dans l'approche analytique, l'unité à

reconnaître est le caractère, la segmentation des mots en caractères est indispensable. Par conséquent, le taux de reconnaissance diminue dû aux erreurs de segmentation. Ces erreurs sont généralement réduites par utilisation des méthodes telles que les modèles de markov caché (HMMs) [A.T. Nghiem et al, 2005].

Les méthodes de classification de caractères peuvent être structurelles ou statistiques. Dans l'approche structurelle, le caractère est décrit par des primitifs structurels. La classification est basée sur des graphes de grammaire ou des règles de comparaison de primitifs. L'avantage de cette approche est qu'elle est rapide puisque elle n'exige pas un grand nombre d'exemples d'apprentissage mais la détermination de tels attributs caractéristiques est délicate. Dans l'approche statistique, chaque caractère est décrit par des distributions probabilistes. Le processus de la reconnaissance emploie des méthodes telles que les réseaux de neurones, les SVM et les HMMs, plus particulièrement, les modèles de Markov caché (HMMs) qui ont reçu un grand intérêt puisqu'ils modélisent les relations existantes entre les caractères de la même chaîne (par exemple, un mot). En outre, ils s'adaptent aux variations de la longueur du texte manuscrit. Cependant, ils ne sont pas efficaces pour la classification des caractères isolés en comparaison aux réseaux de neurones et les SVMs. Par conséquent, de nouvelles tendances combinant les modèles de markov caché et les réseaux de neurones ont été proposées [A.T. Nghiem et al, 2005].

1.3 Travaux associés

1.3.1 Les réseaux de neurones

La majorité des systèmes de la reconnaissance de caractères sont basés sur les réseaux de neurones dus de leur capacité de généralisation (c à d leur capacité à traiter de nouveaux exemples différents des exemples d'apprentissage). Les réseaux de neurones ont été largement répandus dans le domaine de la reconnaissance d'écriture manuscrite et des résultats prometteurs ont été parvenus, particulièrement pour les chiffres manuscrits en utilisant " le Perceptron multicouche" (MLP). Les raisons derrière leur utilisation consistent principalement en leur capacité à explorer des espaces de recherches complexes due à la dynamique qu'elles suivent. Néanmoins, les réseaux de neurones présentent quelques difficultés : Un réseau de neurones est une boîte noire dans le sens qu'il n'est pas interprétable et ne pourrait pas être expliqué à travers les exemples d'entrée [*A.*Vesely, 2003], [A. Pinkus, 1999]. Un réseau de neurones nécessite un temps énorme et une mémoire significative pour son apprentissage, mais une fois entraîné, ses poids sont enregistrés et le test devient très rapide. En outre, il est difficile de définir l'ensemble des données d'apprentissage permettant la meilleure généralisation pour un problème donné.

Les réseaux de neurones peuvent être distingués en quatre classes basées sur l'architecture adoptée [L.Jin et al, 1995], [J.Vesanto et al, 2000] :

- Les réseaux de neurones feed-forward (FFNNs ou MLPs)
- Les Réseaux de neurones à Base Radiale (RBFs)
- Les Cartes auto-organisatrices (SOMs)
- Les Réseaux de neurones Récurrents (RNNs)

Les réseaux du type MLP sont basés sur le calcul du produit scalaire entre les entrées de chaque neurone et leurs poids synaptiques correspondants et puis de calculer la sortie de chaque neurone à travers une fonction qui est généralement la fonction sigmoïde. En revanche, les RBFs sont basés sur le calcul de la distance euclidienne entre les entrées et les centres des fonctions représentant les neurones cachés dits à base radiale. Les cartes auto-organisatrices (SOMs) possèdent une structure complètement différente où les nœuds (neurones) sont uniformément organisés sous la forme d'une grille hexagonale ou rectangulaire. Les SOMs sont basés sur le principe de réduction de l'espace d'entrée à une dimension inférieure représentant l'espace de la carte auto-organisatrice. Les coordonnées d'un nœud de la carte disposant le vecteur de poids le plus proche au vecteur d'entrée seront assignées au vecteur d'entrée [J.Vesanto et al, 2000]. Les réseaux de neurones récurrents (RNNs) sont généralement indiqués pour les problèmes dynamiques (temporels) grâce à leur capacité de mémorisation à travers les cycles dans leurs graphes d'interconnexions [L.Jin et al, 1995].
Des réseaux de neurones plus récents ont été également développés parmi lesquels on distingue les réseaux de neurones convolutionnels, les réseaux de neurones à délais temporels et les réseaux de neurones à déplacement spatial (Space Displacement Neural Networks : SDNN).

1.3.2 L'apprentissage des réseaux de neurones

L'apprentissage d'un réseau de neurones implique l'ajustement de ses poids et biais basé sur un processus itératif qui réduit au minimum la valeur d'une fonction particulière. En fait, les algorithmes d'apprentissage peuvent être distingués en deux classes : l'apprentissage supervisé où la sortie prévue doit être fournie à l'avance, et l'apprentissage non supervisé où la sortie prévue n'est pas exigée à l'avance, elle résulte après l'étape d'apprentissage ; ce dernier est généralement employé pour le clustering.

De nombreuses techniques ont été développées pour l'apprentissage des réseaux de neurones. Les techniques que nous avons utilisées dans ce document sont basées sur l'optimisation par intelligence en essaim. Nos travaux sont essentiellement inspirés des approches neuro-évolutionnaires bien décrites dans un chapitre de la deuxième partie de ce document. Les algorithmes évolutionnaires ont été justifiés très efficaces pour la conception et l'apprentissage des réseaux de neurones [J.Branke, 1995].
L'apprentissage des réseaux de neurones à base des algorithmes évolutionnaires peut apporter une grande amélioration par rapport aux réseaux de neurones conventionnels qui aboutissent souvent à des performances sous optimales. De même, les architectures des réseaux de neurones peuvent profiter de leur mise en œuvre par ces méthodes de recherche globale en montrant une amélioration des performances. De plus, la sélection de primitives par les algorithmes évolutionnaires fournis des résultats prometteurs en termes de vitesse et exactitude.

1.3.3 L'approche neuro-évolutionnaire

Le courant des approches évolutionnaires dans la modélisation des réseaux de neurones en est un exemple très important, notamment en ce qui concerne la structure et poids optimaux pour un problème donné. Particulièrement, trois types d'adaptations des réseaux de neurones peuvent être investis dans le cadre des problèmes de la reconnaissance de formes à savoir l'adaptation des poids, de la structure et des données d'apprentissage par des algorithmes évolutionnaires, tels

que, les algorithmes génétiques, les stratégies d'évolution, la programmation génétique et bien d'autres. Nous avons spécifié le chapitre 8 pour détailler cet axe de recherche jugé intéressant.

1.3.4 Les algorithmes Co-évolutionnaires

Très récemment, les algorithmes co-évolutionnaires, tels que, les algorithmes co-évolutionnaires coopératifs ou les algorithmes co-évolutionnaires compétitifs ont gagné plus d'attention dans le domaine d'optimisation des fonctions. Les raisons derrière leur utilisation consistent principalement dans leur capacité à explorer de grandes et complexes espaces de recherche dues à la dynamique qu'elles suivent et également la connaissance minimum qu'elles exigent au sujet du problème à résoudre.

Dans ce travail de thèse, nous essayons surtout de faire apparaître certaines de leurs capacités à travers des approches de la reconnaissance de caractères que nous avons développées. De ce fait, nous montrons la transition entre les algorithmes évolutionnaires et les algorithmes co-évolutionnaires jugés plus efficaces.

Les algorithmes évolutionnaires traditionnels basés sur la sélection naturelle sont des modèles simplifiés inspirés de la nature, qui imitent l'évolution biologique indépendante des espèces dans un environnement statique, ce qui est essentiellement différent de l'évolution réelle des espèces dans la nature [R.P. Wiegand, 2004], [L.Pagie et al, 1997]. Les espèces réelles Co-évoluent et vivent dans un environnement en cours d'évolution [M.Kirley, 2002]. La Coévolution à l'inverse d'une évolution simple est le processus des influences réciproques entre deux, ou plusieurs populations [R.P. Wiegand, 2004], [J.P.Cartlidge, 2004], souvent appelées des espèces. Ces espèces évoluent simultanément par interaction entre elles de telle sorte que l'évaluation d'une espèce résulte du processus évolutionnaire des autres espèces vivantes. Le but de la coévolution est de produire une dynamique ressemblante à celle de la course aux armements. Informellement, dans la course aux armements : les meilleures performances sont obtenues par chacune des espèces tout en incrémentant les performances des autres espèces. L'idée derrière cette nouvelle notion est qu'un système pourra mieux évoluer par échange de performances. La Coévolution à l'inverse d'une évolution simple se fonde principalement sur la notion de la « fitness subjective ». À l'inverse de la fitness objective utilisée dans les algorithmes génétiques traditionnels, la fitness subjective est une mesure dépendante et sensible à l'état des autres individus interagissant l'un avec l'autre.
Il existe deux formes principales de la Coévolution : La Coévolution compétitive ; où les espèces coadaptées sont guidées par des objectifs en opposition (par exemple la fitness d'une espèce est inversement proportionnelle à la fitness de l'autre espèce) [J.Paredis et al, 1997], et la Coévolution coopérative où les espèces évoluent ensemble afin de résoudre les divers éléments du même problème à résoudre [R.P. Wiegand, 2004]. L'exemple le plus connu de la coévolution compétitive sont les réseaux de tri de Hillis [H. Lipson et al, 2005] ; Dans ce modèle, les réseaux de tri et les cas de tests co-évoluent l'un avec l'autre pour trier des listes de nombres avec un nombre minimal des échanges et de comparaisons. Dans l'approche de Hillis un réseau de tri représente un hôte et le cas de test représente un parasite. Les hôtes opèrent sur des listes des nombres de longueurs fixes. La fitness d'un hôte est le pourcentage des listes correctement triées et la fitness d'un parasite est le pourcentage des cas de test qui sont inexactement triés par un

hôte. Hillis a illustré que la Coévolution des cas de tests peut fournir de meilleures performances en comparaison à d'autres méthodes traditionnelles qui emploient des cas de tests statiques [W. D. Hillis, 1990].
La coévolution coopérative est un outil efficace pour la résolution des problèmes séparables composés de sous-problèmes distincts [R.P. Wiegand, 2004], La coévolution coopérative a été principalement employée pour la tâche d'optimisation des fonctions [M. A.Potter et al, 1994]. Dans les algorithmes co-évolutionnaires coopératifs: des individus collaborateurs sont choisis d'autres populations, et puis combinés avec l'individu courant pour former une solution complète afin de permettre son évaluation, des gains de fitness sont alors affectés à l'individu courant en se basant sur la stratégie choisie d'affectation de crédit. La qualité des résultats dans les systèmes co-évolutionnaires coopératifs est principalement dépendante de la stratégie d'affectation de crédit entre les individus des espèces différentes. Il a été montré dans [M. A.Potter et al, 1994] que la stratégie optimiste est le meilleur choix pour l'optimisation des fonctions. Cependant, les systèmes co-évolutionnaires présentent quelques inconvénients : les plus connus sont : le désengagement, la sur-spécialisation et le cycling [R.P. Wiegand, 2004], [J.P.Cartlidge, 2004].

1.4 Problématique

Les énormes services offerts par les systèmes bancaires, les systèmes de tri postal, et les systèmes de gestion des formulaires ne sont pas assurés 100 % du fait de la très grande variabilité des styles d'écriture de différentes personnes. La difficulté de la reconnaissance de caractères manuscrits est due principalement aux variations des styles d'écriture. Dans cette thèse, nous avons traité une problématique majeure rencontrée en reconnaissance de caractères qui est la proposition d'un système de reconnaissance de caractères permettant de s'adapter aux changements des styles d'écriture pour en augmenter les performances.

Des solutions abondantes ont été proposées pour résoudre ce problème en concevant des systèmes qui emploient souvent des techniques à base des réseaux de neurones. De telles solutions souffrent du problème des optimums locaux causés par les algorithmes d'apprentissage employés qui sont souvent des méthodes d'optimisation par descente de gradient. Les recherches conduites dans le domaine de la reconnaissance de caractères ont montré qu'une amélioration considérable de performances peut être réalisée en employant des méthodes de recherche globales telles que les algorithmes évolutionnaires et les algorithmes d'intelligence en essaim. Les travaux de cette thèse sont fondés sur deux aspects unifiés qui sont l'intelligence en essaim et les réseaux de neurones évolutionnaires.

La démarche suivie dans cette thèse se situe dans le cadre d'étude des systèmes émergents afin de les adapter au problème de la reconnaissance de caractères. Nous nous intéressons à la modélisation collective d'un système de la reconnaissance fondée à la fois sur des classificateurs efficaces, tels que, les réseaux de neurones et sur des algorithmes d'intelligence en essaim tels que l'optimisation par les algorithmes d'abeilles ou l'optimisation par essaim de particules. Cette hybridation tente de combler les lacunes des anciennes méthodes d'apprentissage.

Nous focalisons notre étude autour de deux familles de problématiques typiques du domaine, qui sont : l'optimisation de la performance des réseaux de neurones par des algorithmes métaphoriques basés sur des principes de l'intelligence collective et l'optimisation de la

reconnaissance de caractères manuscrits par des techniques de combinaison de classificateurs et d'extraction des attributs caractéristiques.

1.5 Objectifs

L'objectif de ce travail est d'établir un système de reconnaissance de caractères manuscrits s'appuyant sur une vision d'intelligence distribuée afin d'améliorer le taux de reconnaissance sur différents types de caractères. Il s'agit d'apporter des contributions au niveau de la reconnaissance de caractères manuscrits en utilisant des techniques partant des phénomènes naturels tels que les comportements collectifs d'oiseaux, des abeilles, des insectes sociaux en vue de résoudre ce problème de manière efficace.

Plusieurs types de procédés peuvent être employés dans le but d'améliorer la qualité de reconnaissance. Cette thèse est consacrée aux améliorations apportées par des méthodes de l'intelligence collective au perceptron multicouche et les réseaux de fonctions à base radiale quand ils se sont appliqués au problème de la reconnaissance de caractères manuscrits. Notre ambition était de réaliser un système de la reconnaissance de caractères manuscrits permettant la meilleure performance basée sur des méthodes caractérisées d'intelligence collective.

Les approches développées s'inscrivent dans un champ plus large qui est la résolution de problèmes par émergence: elles proposent des réseaux de neurones adaptatifs évolués par des modèles métaphoriques tels que le modèle de fourragement des abeilles ou le modèle des groupes collectifs d'oiseaux.

1.6 Originalité et contributions principales

Les systèmes de reconnaissance de caractères manuscrits présentés dans cette thèse portent sur quatre types de contributions à savoir la classification par des méthodes d'intelligence en essaim, les dynamiques d'apprentissage des réseaux de neurones par coévolution, les techniques de sélection d'attributs caractéristiques et les techniques de combinaison de classificateurs. L'utilisation de la coévolution au sein d'un système de reconnaissance de caractères est l'une des particularités de notre travail. D'autre part, nous montrons comment peuvent se construire et se combiner les résultats de différents réseaux de neurones. De plus, nous disposons d'une abondante littérature et d'outils puissants concernant le sujet à développer.
Après avoir mis en lumière les concepts associés et certaines limitations des approches existantes, nous avons explicité les approches développées dans le contexte de la reconnaissance de caractères par l'intelligence collective. Nous avons commencé par simplifier les méthodes d'optimisation les plus récentes afin de les adapter à nos besoins de modélisation.
Les contributions principales de cette thèse autour de notre proposition d'un système de la reconnaissance de caractères manuscrits basé sur des méthodes d'intelligence collective sont :
L'hybridation des réseaux de neurones avec des méthodes assez récentes telles que les algorithmes d'abeille et l'optimisation par essaim de particules qui figure parmi les propositions rares de la littérature et plus particulièrement dans le domaine de la reconnaissance de caractères manuscrits.

Nous avons présenté deux approches principales : la première est statistique et la seconde est à la base des réseaux de neurones en employant deux types de réseaux de neurones qui sont le MLP et le RBF respectivement. Les cadres d'applications sont tout d'abord orientés vers la reconnaissance des chiffres arabes manuscrits, d'autres résultats en employant des caractères latins (anglais) sont expliqués par l'exemple.

Nous avons examiné l'apport des algorithmes de recherche globale dont leur propriété est l'intelligence collective à savoir : PSO, BA et ABC et décrire comment ces algorithmes peuvent être employés pour résoudre le problème de la reconnaissance de caractères manuscrits. Ces algorithmes de base ont été décrits en détail dans un chapitre spécifique, car ils constituent les éléments de référence particulièrement pour notre travail tout au long de cette thèse. Nous avons également introduit ces algorithmes comme méthodes d'apprentissage de deux types de réseaux de neurones qui sont le MLP et le RBF. Les performances de ces deux réseaux avec différentes méthodes d'adaptation et différents extracteurs de caractéristiques ont été examinées. Nous avons procédé aussi à une comparaison des performances de ces approches développées à base des réseaux de neurones vis-à-vis des réseaux de neurones MLP et RBF conventionnels.

Nous avons tenté de montrer que les résultats trouvés ouvrent la voie à de nouvelles approches et interprétations sur la problématique de l'amélioration de la performance du système de la reconnaissance de caractères manuscrits.
Nous estimons que l'objectif de modélisation basé sur le comportement collectif est atteint et peut- être mieux que les méthodes d'optimisation standard. Car nos modèles apportent une vraie exploitation et exploration de l'espace des caractères d'apprentissage ; nous avons pu mettre en évidence deux résultats importants qui découlent de nos approches basées sur le principe de la combinaison de classificateurs et les résultats obtenus sont suffisants pour confirmer le potentiel de nos approches.

1.7 Organisation de la thèse

Cette thèse est organisée en trois grandes parties : la première, porte sur la reconnaissance de l'écriture manuscrite suivie des étapes du système de la reconnaissance de caractères ; la seconde partie s'intéresse aux concepts liés au domaine de l'intelligence collective et de l'intelligence en essaim et enfin la troisième est dédiée à l'étude expérimentale des approches développées.

Partie I : cette partie présente le domaine d'investigation de notre étude. Elle est constituée de cinq chapitres : le premier chapitre est dédié aux principales étapes liées à la reconnaissance de caractères manuscrits ainsi qu'aux concepts clés qui lui sont indissociables à savoir les techniques de prétraitement, le traitement, la réduction du lexique et le post-traitement. Le deuxième chapitre présente les techniques de la segmentation de l'écriture manuscrite. Le troisième chapitre décrit les méthodes d'extraction des attributs caractéristiques. Le quatrième chapitre présente les méthodes de classification des caractères manuscrits les plus célèbres et les plus efficaces. Cette partie se termine par la description des méthodes de la combinaison de classificateurs dues à leur grand intérêt dans le domaine de la reconnaissance de caractères.

Partie II : la seconde partie présente les notions fondamentales de l'intelligence collective et l'intelligence en essaim, ses définitions, ses caractéristiques, des exemples de méthodes

nouvelles d'optimisation, des modèles de réseaux de neurones récents. Les techniques de base concernant les réseaux de neurones évolutionnaires sont également présentées dans le dernier chapitre de cette partie.

Partie III :

Cette partie est constituée de trois chapitres. Le premier chapitre expose le système de reconnaissance basé sur une approche statistique utilisant les différents attributs caractéristiques et les différents classificateurs d'intelligence en essaim. Le deuxième chapitre montre l'application d'une approche co-évolutionnaire sur deux types de réseaux de neurones soit le MLP et le RBF dans le cadre de la reconnaissance de chiffres manuscrits. Le dernier chapitre est consacré à la conception d'un système de la reconnaissance de caractères que nous avons développés selon différentes sources d'inspiration basées sur l'intelligence en essaim et les réseaux de neurones. Une approche basée sur la combinaison série du perceptron multi-couche et une approche de classification inspirée du comportement collectif des abeilles suivie d'une approche basée sur la combinaison de plusieurs classificateurs ayant différents extracteurs caractéristiques. Les résultats obtenus y sont également bien explicités.

Une conclusion reprenant les points forts et des limitations en essayant de mettre les approches mises en œuvre, ponctuées par des perspectives termine cette thèse.

Chapitre 2
RECONNAISSANCE DE L'ECRITURE MANUSCRITE

2.1 Introduction

La reconnaissance de l'écriture manuscrite est une voie traditionnelle qui a attiré l'attention des chercheurs depuis plusieurs décennies et reste un domaine de recherche très ouvert dû à son grand nombre d'applications pratiques. Aujourd'hui, les avancées réalisées dans ce domaine se concrétisent par un certain nombre d'applications comme la lecture automatique de chèques bancaires, d'adresses postales ou les adresses de formulaires. C'est un domaine très actif, d'autant plus que les interfaces homme-machine s'orientent de plus en plus vers la communication manuscrite comme les tablettes PC, les applications orientées stylo …etc.
La reconnaissance de l'écriture manuscrite est la transcription des données manuscrites au format numérique et puis de retrouver sa signification. Cette tâche requiert toujours une étape de base dite la segmentation, c-à-d la séparation des graphèmes, des mots ou des chaînes numériques. Le problème de la segmentation des caractères manuscrits constitue le défi principal de la reconnaissance de l'écriture manuscrite ainsi plus que la segmentation est correcte plus que le système de reconnaissance est meilleur.
L'une des applications particulière de la reconnaissance est la reconnaissance du montant chiffre sur chèques bancaires du fait de la présence de bruits, le chevauchement de chiffres et même la fragmentation d'un chiffre en plusieurs parties difficilement identifiables.
Dans ce 2ème chapitre nous allons présenter un état de l'art des méthodes principales de la reconnaissance de l'écriture manuscrite, notamment, la reconnaissance de caractères, de mots manuscrits, de phrases et les différentes étapes permettant la reconnaissance des caractères manuscrits. Ce chapitre présente également une synthèse des méthodes et techniques employées dans le domaine de la reconnaissance de caractères.

2.2 La reconnaissance de formes

La reconnaissance de formes est un domaine de l'intelligence artificielle qui vise à distinguer des formes au niveau de la perception. C'est un domaine très vaste qui trouve ses applications dans divers domaines tels que : la robotique, la médecine, la lecture optique de documents et bien d'autres. Le but de la reconnaissance de formes est de classifier des données basées sur des informations a priori qui sont habituellement des mesures statistiques ou des observations qui définissent des points dans un espace donné. Les formes à reconnaître sont souvent des formes géométriques ou descriptibles par des formules mathématiques, telles que, les cercles, les courbes, splines, … Elles peuvent aussi être de nature plus complexe comme les chiffres et les lettres. Un système de la reconnaissance de formes devrait mettre en œuvre les étapes suivantes :
La segmentation de l'image d'entrée en objets isolés, l'extraction des caractéristiques les plus discriminantes possibles et un classificateur probabiliste ou statistique tel que les réseaux de neurones, le k-plus-proche voisin, les SVMs, les arbres de décision, les modèles de markov cachés…etc. Les applications de ce domaine sont très variées et nécessitent des connaissances expertes du domaine d'application [G. Vamvakas et al, 2009].

2.3 La reconnaissance de l'écriture manuscrite

Le but de la reconnaissance de l'écriture manuscrite est d'interpréter le contenu des données numérisées et de produire une description de cette interprétation dans un format désiré. L'intérêt remarquable de ce domaine de la recherche a été démontré par plusieurs applications réelles telles que : les systèmes de la reconnaissance automatique et traitement des adresses postales, la reconnaissance des blocs d'adresse sur les feuilles des impôts, identification du montant sur chèques bancaires ou encore la lecture d'un texte manuscrit.

La reconnaissance d'écriture en ligne et hors-ligne :

La reconnaissance de l'écriture en ligne traite un flux de données qui provient d'un capteur pendant que l'utilisateur écrit, tandis que la reconnaissance de l'écriture hors-ligne traite des données qui ont été obtenues à partir d'un document manuscrit (scanné) [J. Park, 1999]. Autrement dit, dans la reconnaissance de l'écriture manuscrite en-ligne, l'acquisition de l'écriture se fait au cours du tracé, alors que dans la reconnaissance de l'écriture hors-ligne, l'acquisition se fait à partir d'un document papier numérisé [P.M. Lallican, 1999].
La reconnaissance de l'écriture hors-ligne est plus complexe que celle qui est en-ligne due à la présence du bruit dans le procédé d'acquisition des images et la perte d'information temporelle telle que l'ordre d'écriture et la vitesse [M.E. Morita, 2003]. L'écriture en-ligne par contre comporte plus d'information sur le tracé que l'écriture hors-ligne, notamment l'ordre temporel des points. Les systèmes de reconnaissance de l'écriture en-ligne sont généralement plus faciles et plus efficaces que les systèmes de reconnaissance hors-ligne [P.M. Lallican, 1999]. Les systèmes de reconnaissance hors-ligne et en-ligne sont également distingués par les applications pour lesquelles sont employées. La reconnaissance hors-ligne est consacrée au traitement des chèques bancaires, le tri des courriers postaux, la lecture des formulaires commerciaux,… etc, alors que la reconnaissance en-ligne est principalement employée aux domaines de l'informatique industrielle et de sécurité tels que la certification d'auteur et la vérification de signature [M.E. Morita, 2003].

2.4 Formes du texte manuscrit

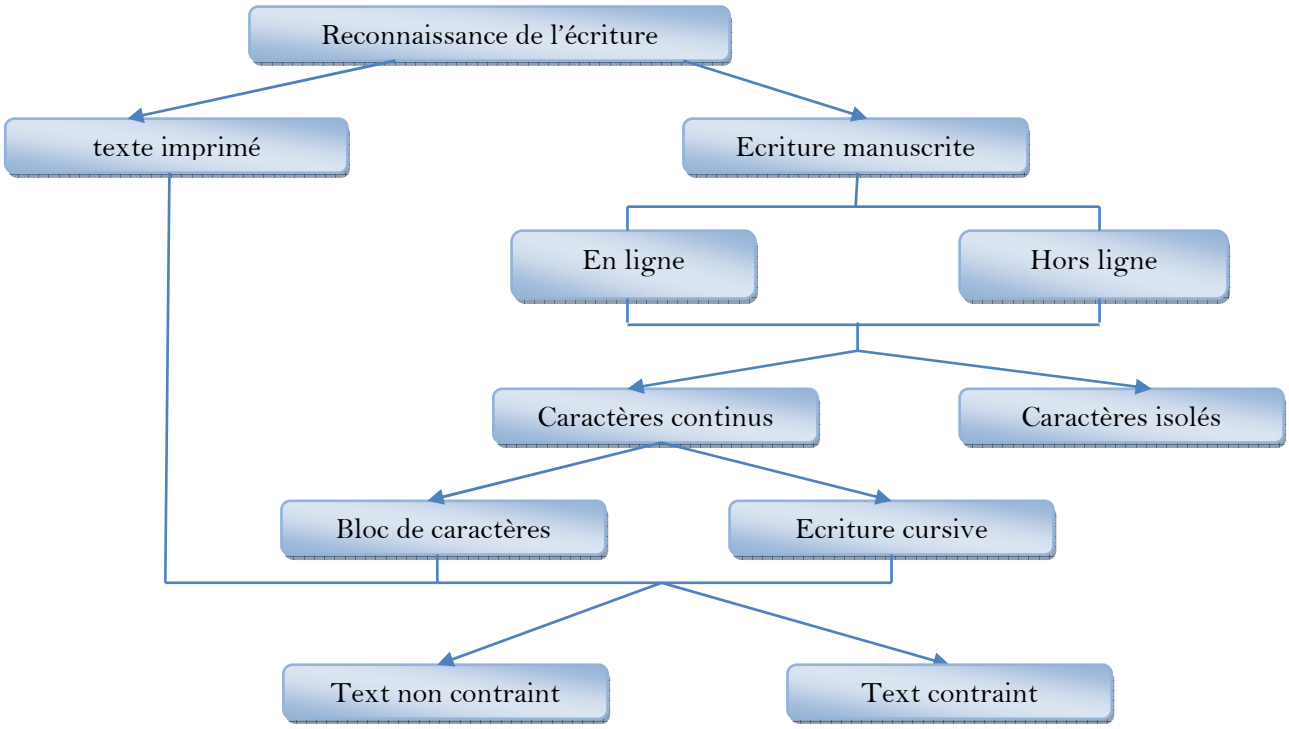
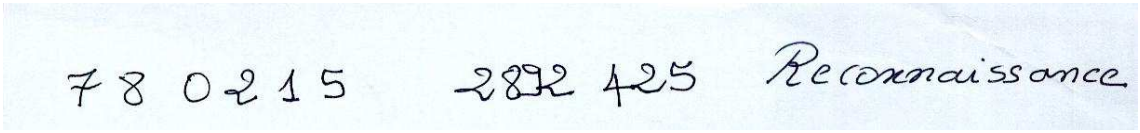


Figure 2.1 : Les différentes formes du texte manuscrit [J. Laaksonen, 1997].

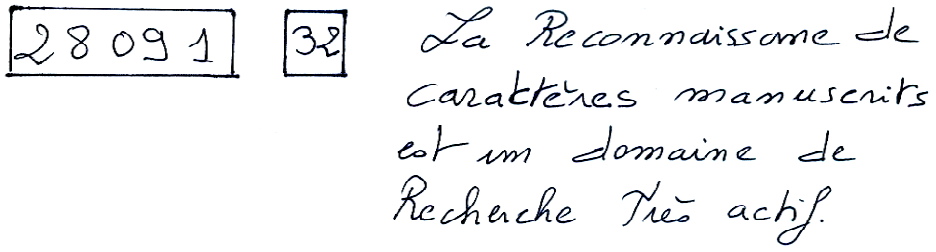
Il existe plusieurs formes du texte manuscrit : ***caractères isolés*** ou ***continus, texte cursif*** ou ***non cursif et texte contraint*** **ou** ***non contraint*** que nous allons présenter. La reconnaissance des caractères *isolés* est simple ; elle procède par la classification des caractères ou chiffres isolés sans avoir besoin d'une étape de segmentation alors que le texte *continu* nécessite de segmenter l'image d'entrée avant la phase de la classification. L'entrée du système de reconnaissance d'un texte *imprimé* est toujours considérée isolée. Un cas particulier du texte *continu* est la reconnaissance de l'écriture *cursive*. La segmentation dans ce cas est beaucoup plus difficile et des méthodes comme le modèle de Markov caché (HMM) sont souvent employées. L'écriture *Non cursive* est également connue en tant que caractères isolés [J. Laaksonen, 1997].



a) caractères isolés (b) texte continu (c) texte cursif

Figure 2.2: les différents types d'écritures.

Un texte imprimé ou manuscrit peut être également *contraint* ou *non contraint.* Le texte contraint inclut des positions où l'utilisateur est conseillé d'écrire en remplissant des emplacements prédéfinis. Ainsi, le système a des informations a priori de ce qu'il va reconnaître et où se trouve le texte. Dans le cas du texte *non contraint*, l'utilisateur a une page blanche qu'elle peut être complétée par n'importe quelle information et dans n'importe quelle direction. Dans la pratique, un texte constitué de plus d'une ligne sans contrainte d'entrée ni le nombre de mots dans chaque ligne est considéré *non contraint* (unconstrained) [J. Laaksonen, 1997].
Des exemples des textes contraints et non contraint sont montrés dans la figure suivante :



(a) texte contraint (b) texte non contraint

Figure 2.3: Exemples de texte contraint et non contraint.

De plus, chaque langue possède ses propres caractéristiques telles que le nombre de lettres et l'écriture de ces lettres dans un mot.
Prenant l'exemple de la langue Arabe : L'arabe est écrit de droite à gauche et elle est toujours cursive. Elle a 29 lettres de base et huit diacritiques. Chaque lettre a plusieurs formes selon sa position dans le mot. Par exemple, la lettre Ain a quatre formes : Forme isolée (ع), initiale, médiale, et forme finale, respectivement de droite à gauche (ععع). En plus, les lettres Hamza, Teh, et Alef ont d'autres formes. Dans un mot, chaque lettre peut être reliée de la droite à la lettre précédente. Cependant, il y a six lettres qui ne soient pas reliées de la gauche à la prochaine lettre. Ces lettres ont seulement les formes isolées et finales. Quand une de ces six lettres est présente dans un mot, le mot est subdivisé en deux *mots secondaires*. Par exemple, le mot arabe (عربية) a deux mots secondaires : le premier mot secondaire se compose d'Ain initial et de Reh final, et le deuxième mot secondaire se compose de Beh initial, de Yeh médial, et de Teh final. Quelques séquences de lettres ont des ligatures spéciales quand ils viennent dans un mot. Par exemple, le Lam suivie de Alef est habituellement dessinée (لا), et Meem suivi de Hah est souvent dessiné (محمد) [G.Abandah et al, 2008].

Les travaux sur la reconnaissance de l'écriture manuscrite peuvent être distingués en quatre catégories, selon si l'unité de reconnaissance est un caractère, un mot, une phrase, ou encore une partie d'un texte [J. Park, 1999], comme nous allons montrer par la suite.

2.5 La Reconnaissance de caractères

La reconnaissance de caractères est le problème de classification des caractères pré isolés dans un ensemble d'alphabet. La plupart des chercheurs ont adopté l'approche classique de la reconnaissance de forme dans laquelle l'étape du prétraitement est suivie de l'extraction de caractéristiques et enfin la classification [J. Park, 1999].

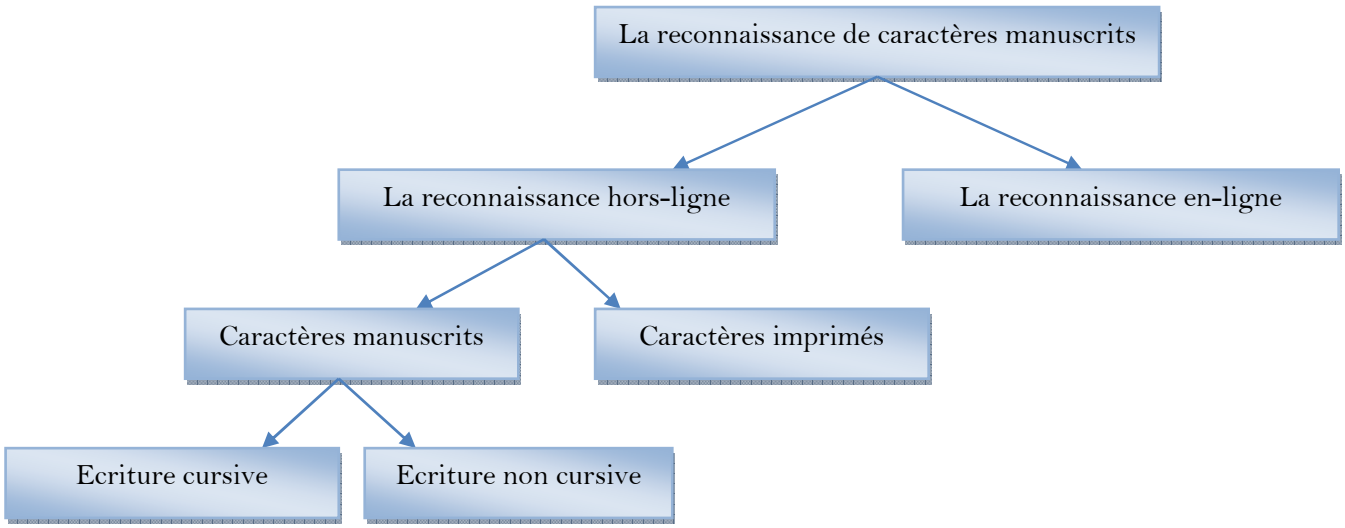


Figure 2.4: Classification des systèmes de la reconnaissance de caractères [M.E. Morita, 2003]

2.6 La Reconnaissance des mots

Les algorithmes de la reconnaissance hors-ligne des mots manuscrits prennent habituellement deux entrées : l'image du mot isolé et un lexique représentant les hypothèses possibles pour ce mot [J. Park, 1999]. Le lexique est la liste des interprétations permises des données manuscrites. Ce lexique est habituellement déterminé par l'application actuelle. Ceci vise à diminuer la complexité du problème puisque l'ambiguïté rend beaucoup des caractères non reconnaissables sans se rapporter au contexte [M.E. Morita, 2003]. La taille de lexique est limitée par l'application (par exemple, en identifiant la somme en lettres sur chèques bancaires, les seules transcriptions permises sont des nombres écrits en lettres). Encore une autre réduction de lexique peut être réalisée en analysant les données manuscrites elles-mêmes et en extrayant à partir du lexique toutes les interprétations incompatibles (si un mot manuscrit ne présente pas des ascendants ni des descendants, on accepte que les transcriptions composées de lettres sans ascendants ou descendants) [A.Vinciarelli, 2002].

Le but de la reconnaissance de mots est d'affecter un score d'appariement à chaque entrée du lexique ou de choisir la meilleure entrée parmi l'ensemble de lexique. Les approches de la

reconnaissance des mots manuscrits peuvent être classées en deux catégories principales : les approches analytiques et les approches holistiques [J. Park, 1999].

Les approches analytiques et les approches holistiques :

Dans une approche analytique, l'étape de la segmentation des mots en segments liés aux caractères est nécessaire. C'est une tâche difficile due aux problèmes tels que la fragmentation ou le chevauchement de caractères. Cette opération est en faite la plus difficile en raison de l'ambiguïté produite des mots manuscrits. Par conséquent, la plupart des méthodes analytiques réussies utilisent des stratégies segmentation- par- reconnaissance [M.E. Morita, 2003].

Dans une approche holistique, la reconnaissance du mot est effectuée en considérant le mot tout entier. Dans ce cas, il n'y a pas besoin de fractionner le mot en segments. À l'inverse des méthodes analytiques, les méthodes holistiques sont contraintes aux applications ayant une petite taille de lexique comme dans les systèmes de traitement des chèques bancaires [M.E. Morita, 2003].

Les méthodes holistiques sont employées pour la reconnaissance des mots manuscrits dans les applications ayant un lexique de petite taille. Alors que les méthodes analytiques avoir besoin de la segmentation d'un mot avant sa reconnaissance, elles se sont avantageuses par rapport aux approches précédentes dans le sens où il est plus fiable de reconnaître un ensemble de caractères que des mots entiers. En effet, la fréquence d'apparition d'un mot est bien inférieure à celle de ses lettres, qui sont partagées par tous les mots du lexique. En plus, les méthodes holistiques ne satisfont pas le critère de portabilité, pour chaque nouvelle application, on doit former l'ensemble des mots du lexique [M.E. Morita, 2003].

La plupart des travaux de la reconnaissance des mots manuscrits présupposent que le mot est segmenté par un algorithme approprié au domaine d'application avant d'être traité par le système de reconnaissance. Certains d'entre eux sont décrits comme suit [M.E. Morita, 2003] :

Scagliola *et al* établit un système de la reconnaissance de mots cursifs, dans lequel la reconnaissance est améliorée à travers plusieurs sources d'informations telles que la relation entre les primitifs d'une hypothèse (un mot), la position de l'hypothèse par rapport à la ligne de base et la ligne supérieure (upper-line), le nombre de primitifs des lettres appartenant à différentes classes de mots, … etc [C. Scagliola et al, 2000].

Kim *et al* établit un système hybride HMM-MLP pour la reconnaissance des mots de l'écriture cursive. Ils ont conçu une segmentation explicite basée HMM et une approche holistique basée MLP. Les contributions principales de ce travail sont l'approche basée HMM et une nouvelle méthode combinant deux classificateurs distincts. Des expérimentations ont été effectuées sur les mots cursifs de la base de données CENPARMI [J. H. Kim et al, 2000a].

2.7 La reconnaissance de phrases

Quelques travaux sur la reconnaissance de caractères et mots ont été étendus au texte manuscrit pour des applications pratiques comme l'interprétation des adresses postales, le traitement automatique des chèques bancaires, ... etc. Les efforts liés à la reconnaissance de phrases manuscrites présupposent souvent que les mots sont isolés ou peuvent être parfaitement isolés. Généralement les systèmes de la reconnaissance de phrases suivent les étapes de traitement suivantes : segmentation en mots, reconnaissance des mots et un post-traitement. Des images segmentées en mots sont envoyées à un système de reconnaissance de mots et des contraintes

linguistiques et contextuelles sont ensuite employées dans l'étape du post-traitement pour compléter le procédé du choix de la phrase [J. Park, 1999]. En d'autres approches, pour compenser les erreurs dues à la segmentation en mots un à plusieurs segments se sont envoyés au système de la reconnaissance des mots et le meilleur chemin trouvé constitue la phrase reconnue.
Quelques applications typiques de la reconnaissance de phrases sont la lecture de texte, les noms de rues et les adresses postales, reconnaissance du montant littéral et dates sur les chèques bancaires.
Les méthodes de la reconnaissance de phrases peuvent être distinguées en deux catégories principales : les méthodes basées segmentation et les méthodes indépendantes de la segmentation [M.E. Morita, 2003].
Dans la 1ère catégorie, la méthode la plus utilisée segmente une phrase en ses éléments constituants basée sur l'espacement entre les composants adjacents. Cette méthode montre ses limites si la phrase n'a pas d'espacement uniforme entre ses composants. En plus, les espaces entre les composants ne peuvent pas être estimés facilement par une métrique à une seule dimension [M.E. Morita, 2003].
Un exemple de cette catégorie est le travail présenté dans [J. T. Favata et al, 1998] qui adopte une approche à deux étapes pour la reconnaissance de phrases. Après l'étape de la segmentation, ils groupent les segments en fonction de leurs relations spatiales et de leur réponse au système de reconnaissance. La sortie du système de reconnaissance est un graphe dirigé qui contient de multiples interprétations de la phrase. La seconde étape est la recherche dans ce graphe en se servant d'un processeur linguistique qui calcule le poids de chaque chemin selon des statistiques liées à la langue et certaines caractéristiques statistiques mesurées à partir le système de reconnaissance.
Dans la seconde catégorie, la phrase manuscrite est traitée tout entière comme unité séparée [M.E. Morita, 2003].
Un exemple de cette catégorie est le travail de [U. Marti et al, 2001] qui présente un système pour la lecture non contrainte du texte manuscrit. Le centre du système est un modèle de markov caché (HMM) pour la reconnaissance de l'écriture. Ce HMM est amélioré par un modèle statistique linguistique incorporant des informations linguistiques au niveau mot. Le HMM a une structure hiérarchique avec des modèles de caractères au niveau le plus bas. Ces modèles sont enchaînés en mots et en phrases entières. Dans cette architecture, la segmentation d'une ligne de texte en mots n'est pas nécessaire, elle est obtenue pendant le processus de la reconnaissance.

2.8 La reconnaissance des chaînes de chiffres

La reconnaissance des chiffres numériques diffère de celle des chiffres isolés en incluant la classification des chaînes de chiffres et la segmentation des chiffres chevauchés. Elle est également différente du problème de la reconnaissance des mots manuscrits dans le sens où presque aucune information contextuelle n'est disponible, c à d, n'importe quel chiffre peut suivre n'importe quel autre. La segmentation des chaînes numériques est généralement une tâche difficile parce que les différents numéros dans une chaîne peuvent chevaucher ou se connecter, ou encore un chiffre peut être constitué de plusieurs fragments.
Les stratégies de la reconnaissance des chaînes numériques manuscrites peuvent être classées en segmentation-puis-reconnaissance [S.Ouchtati et al, 2007] et segmentation-basée-reconnaissance [R. Palacios et al, 2003]. Dans la 1ère catégorie, le module de la segmentation fournit une

séquence d'hypothèses où chaque sous-séquence devrait contenir un caractère isolé soumis au système de reconnaissance.
La stratégie segmentation-basée-reconnaissance est fondée sur une hypothèse probabiliste où la décision finale doit exprimer la meilleure segmentation-reconnaissance de l'image d'entrée. Généralement, le système rapporte une liste d'hypothèses issue du module de segmentation, chaque hypothèse est ensuite évaluée par reconnaissance. En conclusion, la liste est posttraitée en tenant compte l'information contextuelle. Bien que cette approche donne une meilleure fiabilité que la précédente, l'inconvénient majeur se situe dans l'effort requis pour comparer toutes les hypothèses produites. D'ailleurs, le module de reconnaissance doit distinguer de diverses configurations telles que les fragments de caractères, les caractères isolés, et les caractères connectés [M.E. Morita, 2003].

2.9 Les techniques de la reconnaissance d'écriture manuscrite

Partant des quatre approches générales de la reconnaissance de forme, de nombreuses techniques de la reconnaissance de l'écriture manuscrite peuvent être trouvées: les techniques d'appariement de modèles (template matching), les techniques statistiques, les techniques structurelles, et les réseaux de neurones. De telles approches ne sont pas nécessairement indépendantes [M.E. Morita, 2003].
Les techniques d'appariement de modèle déterminent le degré de similitude entre deux vecteurs (groupes de Pixel, de formes, de courbures, …etc) dans l'espace de caractéristiques. Ces techniques peuvent être assemblées en trois groupes :

- L'appariement direct
- L'appariement élastique et déformable
- L'appariement par relaxation

Les techniques statistiques utilisent des fonctions de décision statistiques et un ensemble de critères qui déterminent la probabilité d'appartenance du modèle observé à une certaine classe. Plusieurs approches populaires de la reconnaissance d'écriture appartiennent à cette catégorie [M.E. Morita, 2003] :

- *k*- Plus-Proche-Voisin (*k*-NN)
- Le classificateur bayésien
- Le classificateur discriminant polynomial
- Le Modèle de Markov Caché (HMM)
- Les Supports Vecteurs Machine (SVMs)

Les réseaux de neurones (RNs) sont définis comme structure de calcul composée de processeurs neuronaux adaptatifs qui sont interconnectés de manière massivement parallèle. Les RNs ont été largement répandus dans le domaine de la reconnaissance de l'écriture manuscrite et des résultats prometteurs ont été parvenus, particulièrement dans le cas des chiffres manuscrits en utilisant " le Perceptron multicouche" (MLP).

2.10 Le fonctionnement d'un système de reconnaissance

Les systèmes de reconnaissance de caractères manuscrits sont généralement basés sur cinq étapes principales [G. Vamvakas et al, 2010], [G. Vamvakas et al, 2009] :

1- Le prétraitement
2- La segmentation
3- L'extraction de caractéristiques
4- La classification
5- Le Post-traitement

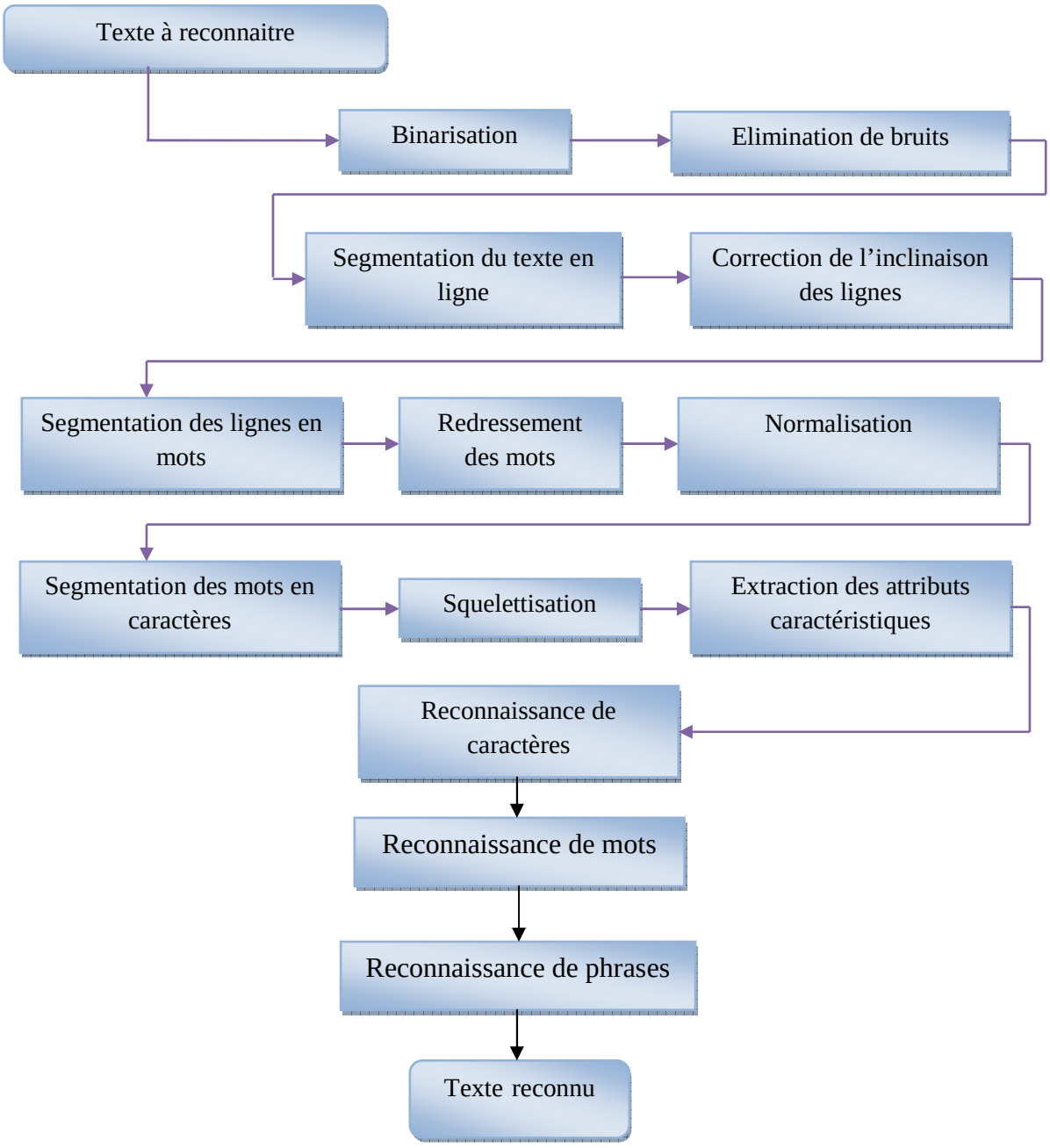


Figure 2.5: Les différentes étapes pour la reconnaissance d'un texte manuscrit

2.10.1 Le Prétraitement

Les images originales doivent être prétraitées afin de fournir la forme adéquate à la reconnaissance. Les principaux objectifs du prétraitement sont : la Binarisation, la réduction de bruit, la normalisation de la largeur de graphèmes, la correction de la pente et de l'inclinaison [G. Vamvakas et al, 2010], [G. Vamvakas et al, 2009]. D'autres opérations sont souvent nécessaires afin d'extraire les champs manuscrits, la suppression de la texture du fond, et la distinction entre l'imprimé et l'écriture manuscrite. D'une façon générale, le résultat du prétraitement doit être une image contenant le champ à reconnaître sans n'importe quel autre élément ennuyeux.

La binarization (thresholding)

Il existe deux catégories de la binarisation (thresholding) : *les méthodes globales* basées sur un seuil global de l'image entière qui est généralement basé sur l'histogramme d'intensité de l'image et *les méthodes locales (adaptatives)* qui emploient différentes valeurs de seuil pour chaque Pixel selon l'information locale de son voisinage.

La réduction de bruit et la normalisation :

La réduction de bruit : est généralement basée sur deux approches principales qui sont :
Les filtres (masques) et les opérations morphologiques (érosion, dilatation, … etc.)

La normalisation : Dans un modèle idéal d'écriture, un mot est écrit horizontalement avec des ascendantes et des descendantes alignées le long de la direction verticale. Dans les données réelles, de telles conditions sont rarement respectées. L'inclinaison (l'angle entre la direction horizontale et la direction de la ligne implicite sur laquelle le mot est aligné) et la pente (l'angle entre la direction verticale et la direction des primitives verticales) sont souvent différentes de 0 et doivent être corrigés (figure 2.6). Les images normalisées sont invariables en termes de pente et inclinaison pour bien servir le processus de la reconnaissance. La normalisation de la pente et l'inclinaison ce qu'on appelle redressement de mot et lettres rend la forme des caractères moins variables, et donc plus faciles à reconnaître avec les techniques de la reconnaissance de formes [A. Ricciarelli, 2002]. Cette étape précède la reconnaissance afin d'augmenter le taux de la reconnaissance.

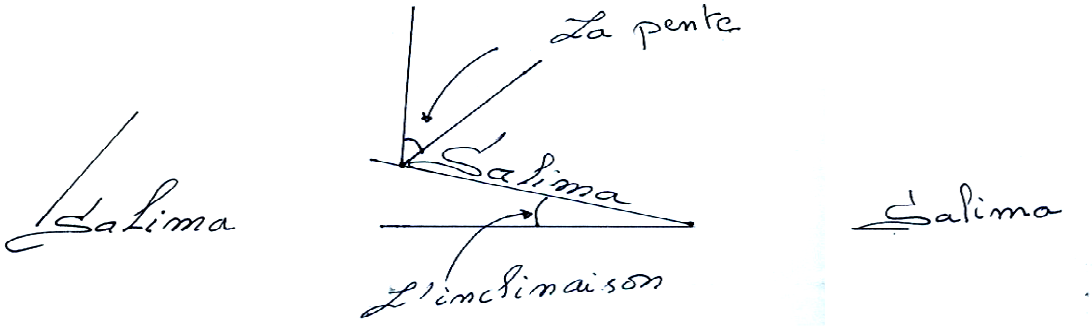


Figure 2.6 : la pente et l'inclinaison d'un mot avant et après le prétraitement.

Redressement des mots :

Cette étape est employée pour aligner l'écriture au même rang du module de balayage. Les approches principales pour le redressement de mots sont la corrélation, projection de profils, et la transformée de Hough [R. Buse et al, 1997].

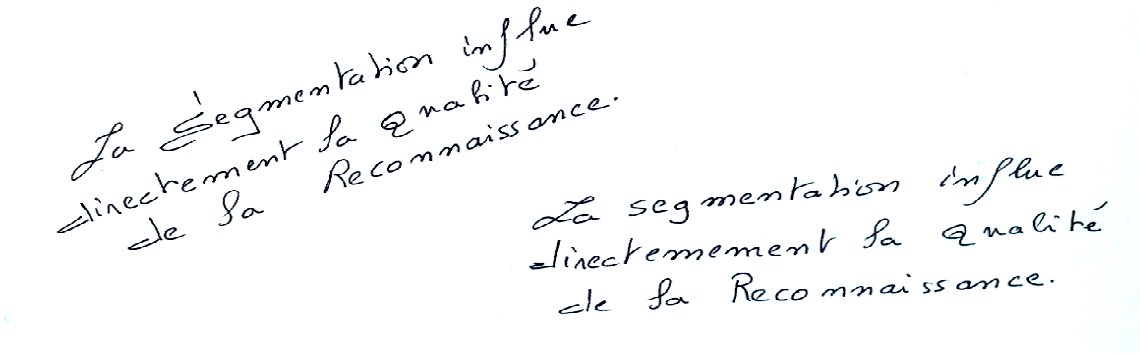


Figure 2.7: redressement des mots inclinés

La plupart des techniques de redressement de mots présentées dans la littérature sont inspirées de la méthode basée projection de profils proposée dans [R.M. Bozinovic et al, 1989], [S.A.Khan, 1998]. celle-ci consiste en une évaluation globale de la région enfermant le corps d'un mot, puis emploie le tracé le plus proche de la ligne horizontale sur laquelle le mot est aligné pour redresser cette ligne. Ainsi, l'image doit être tournée jusqu'à ce que telle ligne soit horizontale (figure 2.7). L'évaluation de la région qui enferme le corps d'un mot est l'étape fondamentale qui consiste à trouver les lignes ayant la densité horizontale la plus élevée. Pour ce faire, l'histogramme horizontal est analysé en examinant ses caractéristiques telles que les maximums et les crêtes de ses 1 ^{ères} dérivées [A. Vinciarelli, 2002].

Redressement des caractères :

Des méthodes de correction de la pente verticale des caractères manuscrits ont été employées pour normaliser les caractères dans un format standard telles que [G. Vamvakas et al, 2007b] :

- calcul de l'angle moyenne des éléments verticaux
- Méthode de Bozinovic – Shrihari (BSM).

La plupart des méthodes de redressement des lettres sont basées sur la technique proposée dans [R.M. Bozinovic et al, 1989], [S.A.Khan, 1998]. celle-ci est fondée sur le choix du tracé le plus proche à la ligne verticale dont la pente est assumée locale pour un caractère donné. La valeur globale est obtenue en calculant la moyenne de toutes les pentes locales [A. Vinciarelli, 2002].

Figure 2.8: redressement des caractères dans un mot

Redressement des lettres par entropie *: (*Slant Correction*)*
La pente dominante d'un mot manuscrit peut être trouvée à partir de la pente des caractères corrigés fournissant la valeur minimale de l'entropie d'un histogramme de projection verticale. L'histogramme de projection verticale est évalué pour un intervalle d'angle de ± R. Généralement R=60, qui semble couvrir tous les modèles d'écriture. La pente α_m du caractère est calculée comme suit [B. Gatos et al, 2006] :

$$\alpha_m = \min_{\alpha \in \mp R} H \qquad H = -\sum_{i=1}^{N} p_i \log p_i \qquad (2.1)$$

Avec N *est* le nombre de bins de l'histogramme de projection verticale qui sont égales à une valeur maximale *xmax et pi la* probabilité d'un Pixel du fond de caractère apparaissant dans le bin *i.*
Chaque caractère est ensuite corrigé en calculant ses nouvelles coordonnées (*x'* et *y'*) basé sur la valeur de α_m [B. Gatos et al, 2006] :

$$x' = x - y\tan(\alpha_m) \qquad y' = y \qquad (2.2)$$

2.10.2 La segmentation

Il existe plusieurs méthodes de la segmentation de l'écriture manuscrite, les plus répandues sont la transformée de Hough et les profils de projection horizontaux pour la détection des lignes du texte manuscrit aussi bien que les profils de projection verticaux et l'analyse de composantes connexes pour l'extraction des mots [G. Vamvakas et al, 2008].
La segmentation d'une image est effectuée en identifiant les ensembles de Pixels connectés et appartenant au même environnement spatial [R.Palacios et al, 2004]. C'est le moyen d'isoler des fragments en mots manuscrits constituant les unités d'information pour le système de reconnaissance.
La segmentation peut être explicite ou implicite selon, que les éléments isolés soient des caractères ou pas [A. Vinciarelli, 2002].

Segmentation implicite et Segmentation explicite **:**
La segmentation est dite explicite, si elle est basée sur l'isolement d'un mot en lettres simples afin de les reconnaître séparément, c'est l'exemple des systèmes de reconnaissance basés sur la programmation dynamique. Tandis que, la segmentation implicite réduit le mot en sous-lettres secondaires et la seule contrainte à respecter est la segmentation finale : le mot doit être divisé en ligatures (liens) réelles entre les caractères manuscrits. En d'autres termes, chaque sous-unité du mot doit appartenir à un seul caractère, c'est l'exemple des systèmes de reconnaissance basés sur le modèle de Markov caché [A.Vinciarelli, 2002].

Segmentation implicite Segmentation explicite

Figure 2.9: exemple : segmentation implicite et segmentation explicite

Les méthodes explicites doivent faire face aux chevauchements de caractères par des algorithmes heuristiques, qui sont lourds et difficiles à définir. Les méthodes de la segmentation implicite sont généralement employées pour éviter ce type de problèmes. Mais en raison de la nature bidimensionnelle des images et le chevauchement de caractères, les méthodes implicites sont moins efficaces dans le cas de la reconnaissance d'écriture en-ligne. En effet, la division verticale du mot manuscrit perd l'aspect séquentiel des caractères chevauchés, qui sont mieux représentés par les méthodes explicites. Les méthodes explicites dans la plupart des cas surmontent les méthodes implicites. Néanmoins, les méthodes implicites complètent les méthodes explicites et se sont particulièrement efficaces en faisant face au chevauchement de caractères, pour lequel il est difficile de trouver la segmentation explicite correcte [M.E. Morita, 2003].

La segmentation explicite est plus difficile : une lettre ne peut être segmentée avant sa reconnaissance et ne peut être reconnue avant l'étape de segmentation. Par contre, la segmentation implicite est plus facile puisque la seule contrainte à respecter est la segmentation finale. La segmentation explicite dans la plupart du temps est effectuée par les systèmes fondés sur la programmation dynamique, alors que la segmentation implicite est employée dans les systèmes appliquant les modèles de Markov cachés. Le problème réel est lorsqu'on applique la programmation dynamique, les morceaux extraits d'un mot doivent être des caractères. Alors que les HMMs peuvent travailler sur une séquence de morceaux ne correspondant pas nécessairement aux lettres, ils peuvent aussi faire face aux variations de styles et de bruits [A. Vinciarelli, 2002].

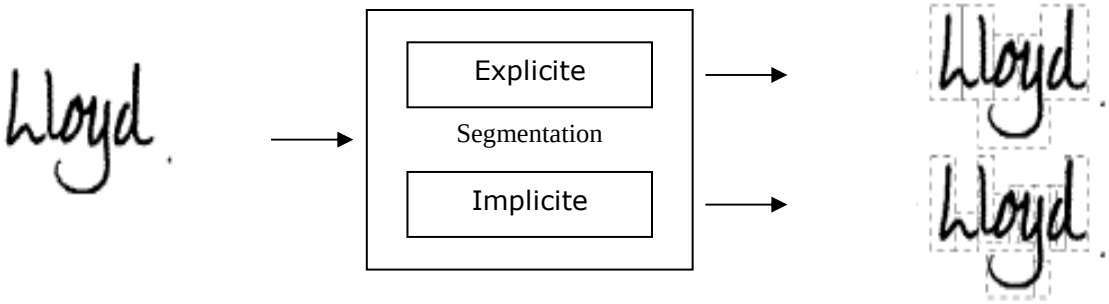


Figure 2.10: la segmentation : explicite et implicite [A. Vinciarelli, 2002]

2.10.3 L'extraction des attributs caractéristiques

Les attributs caractéristiques aussi appelés primitives peuvent être groupés en trois classes selon qu'ils soient extraits à partir du mot entier (attributs de haut niveau), des lettres (attributs de moyen niveau) ou des lettres secondaires (attributs de bas niveau).
Les attributs de bas niveau sont extraits à partir des morceaux de lettres ; leurs formes sont élémentaires telles que de petites lignes, des courbures, des barres..Etc. Ils représentent, en général, des caractéristiques géométriques simples. Dans la plupart des cas, *les primitives de bas niveau* sont des déformations des éléments d'un ensemble de base de primitives où la déformation en elle-même pourra être employée comme primitive. Alors que l*es attributs à moyen niveau* sont les attributs représentant la moyenne des régions locales et qui sont généralement préférables pour la segmentation explicite, tandis que *les attributs de haut niveau* comprennent la détection des éléments structuraux, ils ne dépendent pas du style d'écriture et sont donc stables par rapport à la variabilité cursive. Ils se sont généralement les boucles, les ascendantes et les descendantes d'un caractère [A. Vinciarelli, 2002].

2.10.4 La reconnaissance

Avant l'étape de la reconnaissance de caractères, un processus de la segmentation des données manuscrites est employé suivi d'une étape d'extraction de caractéristiques pour convertir les caractères isolés en vecteurs d'attributs. La reconnaissance consiste à trouver, parmi les lettres disponibles, le plus similaire par des techniques telles que les réseaux de neurones, le classificateur Bayesien, les SVMs (Machine à Vecteur Support)… etc.
Dans un système de la reconnaissance des mots manuscrits, un élément fondamental est le lexique qui doit être construit en réduisant sa taille afin d'améliorer les performances du système. La reconnaissance consiste à trouver, parmi les mots du lexique, le plus similaire aux données manuscrites. Les deux techniques principales de cette tâche sont la programmation dynamique et le modèle de Markov caché. L'étape de la reconnaissance emploie généralement les primitives d'un mot isolé par segmentation pour évaluer les éléments du lexique. Le meilleur mot du lexique est assumé comme interprétation des données manuscrites [A. Vinciarelli, 2002].

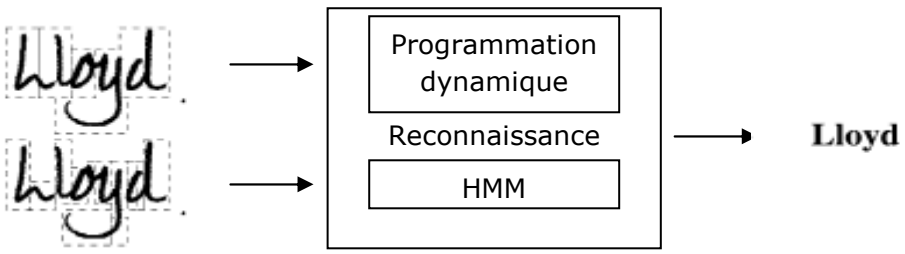


Figure 2.11: La reconnaissance d'un mot par HMM ou Programmation dynamique [A. Vinciarelli, 2002]

2.10.5 La Réduction de la taille du lexique

La taille du lexique est l'une des paramètres les plus importants dans un système de reconnaissance de mots. Plus que le nombre d'interprétations permises augmente, plus que la probabilité d'une fausse classification devient plus haute. Pour cette raison, plusieurs systèmes de réduction de lexique ont été développés. D'une façon générale, un système de réduction de lexique prend comme entrée un lexique et un mot manuscrit et donne un sous-ensemble des entrées du lexique. Les systèmes de réduction de lexique sont toujours basés sur une représentation approximative des données manuscrites permettant d'accepter des entrées de lexique ou de rejeter ceux qui sont incompatibles avec les données à reconnaître. Dans certains cas, ils se sont basés sur d'autres informations que le mot à reconnaître (par exemple dans les applications postales c à d la reconnaissance du code postal, on limite le nombre des interprétations permises d'un nom de ville) [A. Vinciarelli, 2002].

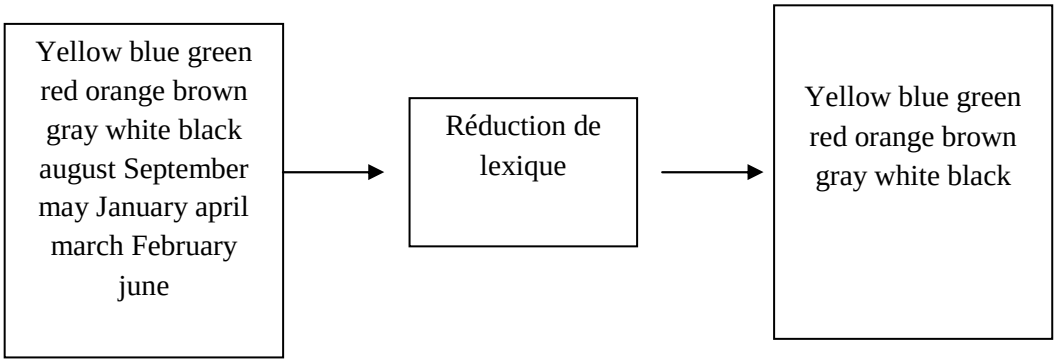


Figure 2.12 : La réduction de lexique

2.10.6 Le post-traitement

L'introduction de l'information du contexte et de forme dans toutes les étapes d'un système de reconnaissance de caractères est nécessaire pour des améliorations significatives du taux de la reconnaissance. Généralement un dictionnaire, un lexique bien développé et un ensemble de règles orthographiques sont utilisés pendant ou après l'étape de reconnaissance pour la vérification et l'amélioration du taux de reconnaissance.
La manière la plus simple est l'utilisation d'un dictionnaire pour corriger les erreurs mineures [G. Vamvakas et al, 2010].

Conclusion

2.11 Conclusion

La reconnaissance de caractères est le problème de classification des caractères pré isolés dans un ensemble donné d'alphabet. Les systèmes de reconnaissance de caractères manuscrits sont généralement basés sur cinq étapes principales :

1- Le prétraitement dont l'objectif est la binarisation, la réduction de bruit et la normalisation, le résultat du prétraitement est l'image des caractères à reconnaître sans bruit ni texture.

2- La segmentation dont l'objectif est la séparation des caractères en commençant par des méthodes comme les profils de projection horizontaux pour la détection des lignes du texte aussi bien que les profils de projection verticaux et l'analyse de composantes connexes pour l'extraction des mots [G. Vamvakas et al, 2008]. La segmentation des mots peut être explicite ou implicite selon que les éléments isolés soient des lettres ou sous lettres. Les méthodes implicites sont préférables dans le cas des caractères chevauchés [A. Vinciarelli, 2002]. La segmentation de l'écriture *cursive* est plus difficile et des méthodes comme le modèle de Markov caché (HMM) sont souvent employées pour la reconnaissance des mots sans leur segmentation.

3- L'extraction des attributs caractéristiques est une étape cruciale dont l'objectif est d'extraire les vecteurs d'attributs permettant la distinction et la classification de caractères.

4- La classification est l'étape finale dont l'objectif est la décision de la classe de chaque caractère, elle se fait par des techniques d'apprentissage ou de classification statistique.

5- Le Post-traitement dont l'objectif est la vérification pour l'amélioration du taux de reconnaissance, généralement un dictionnaire, un lexique ou un ensemble de règles orthographiques sont utilisés.

La reconnaissance de mots est le choix de la meilleure entrée de l'ensemble des mots lexicaux représentant les hypothèses possibles pour ces mots [J. Park, 1999]. La taille du lexique est l'une des paramètres les plus importants. Plus que le nombre d'interprétations permises augmente, plus que la probabilité d'une fausse classification devient plus haute. Il existe deux types de méthodes de la reconnaissance de mots : analytiques et holistiques [J. Park, 1999]. Les méthodes holistiques sont employées dans les applications ayant un lexique de petite taille. Alors que les méthodes analytiques nécessitent la segmentation du mot en segments, elles se sont avantageuses que les approches précédentes dans le sens où il est plus fiable de reconnaître les caractères qu'un mot tout entier.

Les systèmes de la reconnaissance de phrase généralement suivent les étapes de traitement suivantes : segmentation en mots, reconnaissance des mots et une étape de post-traitement pour compléter le procédé du choix de la phrase [J. Park, 1999].

Chapitre 3
SEGMENTATION DE CARACTERES MANUSCRITS

3.1 Introduction

La segmentation de caractères est l'une des problèmes difficiles dans la réalisation d'un système de reconnaissance. Elle influe de manière considérable la performance d'un système de reconnaissance de l'écriture manuscrite. Les stratégies de segmentation de caractères peuvent être distinguées en quatre catégories principales. Les méthodes de segmentation par dissection, qui divisent l'image d'entrée en sous-images ayant les propriétés des caractères de l'image. Les méthodes de segmentation basées reconnaissance [H.Nishida et al, 1994], [O. Matan et al, 1991] qui sont basées sur la reconnaissance pour valider ou rejeter une segmentation, les approches hybrides qui combinent la dissection avec des méthodes de recherche et les approches holistiques qui évitent la segmentation en reconnaissant le mot tout entier [S.Marinai et al, 2005]. Dans la pratique, la segmentation et la reconnaissance de caractères ont de très forte dépendance. La meilleure décision de segmentation en caractères est celle qui satisfait les contraintes contextuelles au niveau mot, ou même les contraintes contextuelles au niveau phrase telles que le nombre de caractères constituant le mot et le nombre de mots constituant la phrase. L'interaction entre la segmentation de caractères et la reconnaissance de caractère est l'aspect le plus important à considérer en concevant une approche efficace de la segmentation de caractères [N. Arıca, 1998]. Le présent chapitre donne un état de l'art des méthodes principales de la segmentation de caractères, la séparation des caractères chevauchés aussi bien que des méthodes de segmentation de caractères tirées de la littérature et qui ont été appliquées sur l'écriture manuscrite.

3.2 Segmentation de caractères

La segmentation de manière générale est la séparation des zones connexes appartenant au même environnement spatial. La segmentation de caractères est le moyen d'isoler des fragments dans un mot manuscrit pour être des unités d'information de base pour la reconnaissance.

Difficultés dans la segmentation de caractères :
La segmentation de caractères manuscrits est une tâche difficile due à la dégradation des images acquises, l'écriture cursive, la grande variété des styles d'écriture, le chevauchement ou la fragmentation de caractères, ou encore l'appartenance des caractères à un fond texturé [T. Saba et al, 2010], [R. J. Rodrigues et al, 2000]. L'un des principaux problèmes qui rendent cette tâche pénible est sa forte dépendance avec la reconnaissance due à la similarité visuelle de caractères. Il est souvent extrêmement difficile de prendre la bonne décision en choisissant un caractère candidat tout en rejetant le reste d'un ensemble d'hypothèses pour un caractère donné. L'existence de beaucoup de caractères visuellement semblables (` h 'et ` b ', ` t 'et ` r ', ` o 'et ` 0 ', ` 1 et l) aussi bien que l'existence des caractères qui replient exactement les parties d'autres caractères (` n' et ` m ', ` c ' et ` d ' ` l ' et ` d ') augmente ce problème, [C. Fang, 1997]. Si, par exemple, la segmentation adoptée accepte plusieurs coupures, certaines lettres peuvent être divisées en sous-lettres. La lettre " m" par exemple, pourrait être interprétée comme une paire de lettres "nn". Ces deux interprétations ne peuvent pas être considérées en même temps, Ceci a comme conséquence un certain nombre de contraintes sur les interprétations possibles des sous lettres [T. Saba et al, 2010], [R. J. Rodrigues et al, 2000].

3.3 Les méthodes de la segmentation de caractères

Les anciennes méthodes de la segmentation de caractères sont principalement basées sur l'analyse du profil de projection et l'estimation de pitch (nombre de caractères par unité de distance horizontale), ces méthodes donnent de bons résultats sur les caractères imprimés ayant des espacements de même taille. L'analyse de composantes connexes est ensuite employée pour traiter les ligatures et la police italique. Les évolutions les plus récentes combinent la segmentation avec la reconnaissance et emploient les résultats de la reconnaissance aussi bien que le contexte linguistique comme feedback pour améliorer les décisions de segmentation [C. Fang, 1997].

Les méthodes de la segmentation peuvent être distinguées en quatre grandes catégories qui sont : *la segmentation par dissection*, qui est l'isolement de diverses unités d'écriture avant la reconnaissance, *la segmentation par reconnaissance*, qui est l'isolement de lettres, particulièrement, dans les mots cursifs en se servant de la reconnaissance, *la segmentation hybride* qui combine la dissection avec des méthodes de recherche et *la segmentation holistique* qui reconnaît le mot tout entier sans faire la segmentation en caractères.

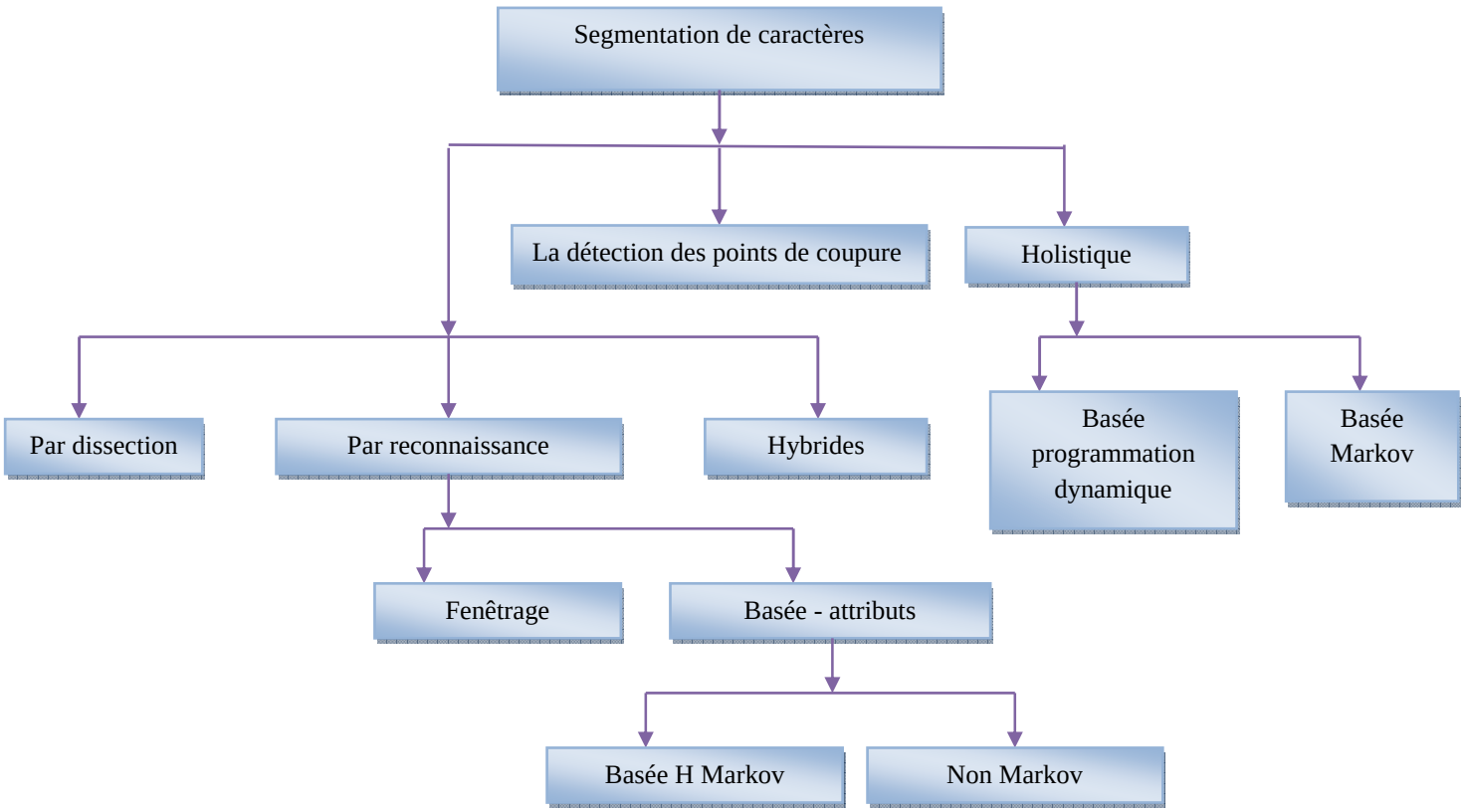


Figure 3.1: Classification des méthodes de segmentation de caractères [M.E. Morita, 2003]

3.3.1 La segmentation par dissection

La segmentation par dissection est un processus indépendant de la reconnaissance qui s'effectue avant la reconnaissance, son but est de trouver l'endroit exact de séparation de caractères. Elle identifie les segments basée sur les propriétés de caractères. Ce processus de coupure de l'image en composants significatifs est appelé "dissection". La dissection est un processus qui analyse une image sans employer des informations structurelles.

Les techniques de la segmentation par dissection incluent :
- L'analyse de composants connexes
- L'espace blanc et l'estimation de pitch
- Landmarks (points de repère)
- Analyse de profils

Nous décrivons par la suite quelques méthodes de cette classe.

3.3.1.1 L'analyse de composants connexes

Une composante connexe est la région maximale de Pixels connexes (non séparés par un contour), la définition de l'ensemble de composantes connexes est la division de l'image en segments [M. Cesar et al, 1990], [R.A. Wilkinson, 1992].

Il existe de différents algorithmes basés sur différentes structures pour déterminer les composants connexes d'images binaires tels que :

- *La diffusion récursive* des composants connexes en balayant récursivement la matrice d'image.
- *Les chaînes de codage* en traçant le contour autour d'un composant dans la matrice d'image
- *analyse de composants connexes basée RLC* (Run-Length-Contour).

Le principe de l'algorithme le plus simple pour déterminer les composants connexes est comme suit :

- Trouver le prochain Pixel non étiqueté dans la matrice d'image
- Diffuser récursivement aux voisins
- Marquer les Pixels visités par l'étiquette courante
- quand le composant est épuisé, sélectionner la prochaine étiquette et répéter

3.3.1.2 L'espace blanc et l'estimation de pitch

La notion de l'espace blanc vertical entre les caractères successifs est un concept important pour séparer les caractères et principalement pour les applications de police imprimée où chaque caractère occupe un bloc de largeur fixe. Le pitch, ou le nombre de caractères par unité de distance horizontale, fournit une base pour estimer les points de segmentation. La combinaison de l'espace blanc et l'estimation de pitch peuvent produire des résultats de segmentation fiables sur des caractères bien espacés avec une largeur fixe. L'estimation de pitch a permis également la segmentation correcte en cas de caractères fragmentés ou chevauchés [C. Fang, 1997].

3.3.1.3 Détection des bornes limites (landmarks)

Les attributs caractéristiques de l'image tels que les ascendantes et les descendantes d'un caractère peuvent servir comme bornes limites pour aider la segmentation des caractères dans un mot manuscrit. La segmentation de caractères basée sur la détection des ascendantes et des descendantes a été appliquée au texte imprimé aussi bien qu'à l'écriture cursive. Un exemple de la détection des bornes limites pour la segmentation de caractères est montré dans [L.D.Harmon, 1972], [C. Fang, 1997], [L.D. Harmon, 1972].

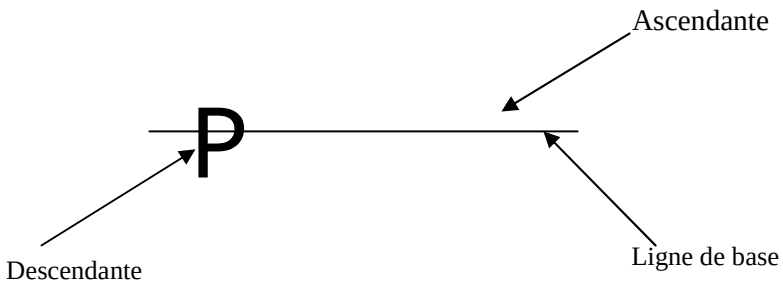


Figure 3.2: Ascendantes et descendantes d'un caractère

3.3.1.4 Les profils de projection

Le profil de projection horizontal est généralement utilisé pour segmenter les lignes d'un texte. Les profils de projection sont caractérisés de basse performance quand ils sont appliqués à l'écriture cursive, les caractères chevauchés ou italiques [A.Zramdini et al, 1993].

Soit S (*N*, M) une image binaire de *N* lignes et M colonnes.
***Le Profil horizontal* :** est la somme de Pixels noirs perpendiculaires à l'axe des *x* ; ceci est représenté par le vecteur p_h *de* taille M *définit* par :

$$P_h[j] = \sum_{i=1}^{N} S[i,j] \qquad (3.1)$$

Les espaces entre mots ou même les caractères aussi bien que les tracés verticales peuvent être détectées par le profil de projection horizontal [A. Kawtrakul et al, 2004].
***Le Profil vertical* :** est la somme de Pixels noirs perpendiculaires à l'axe des *y* ; ceci est représenté par le vecteur P_v *de* taille N *définit* par :

$$P_v[i] = \sum_{j=1}^{M} S[i,j] \qquad (3.2)$$

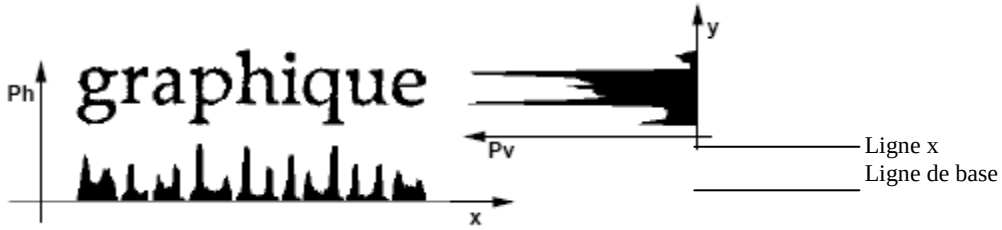


Figure 3.3: profils de projection horizontal et vertical [A.Zramdini et al, 1993]

Dans la figure 3.3, le profil de projection vertical montre deux crêtes à la ligne des x et à la ligne de base. Le profil de projection horizontal montre des zones sans hauteur qui correspondent aux blancs séparant les caractères d'un mot. L'analyse des profils de projection horizontaux et verticaux permet donc la séparation des mots et lignes du texte imprimé basée sur les crêtes et les zones sans hauteur [S. J. Lu et al, 2007].

L'analyse des profils de projection a été appliquée dans [U. Pal et al, 2003] pour la segmentation des caractères manuscrits en se servant de l'analyse de composants connexes. Dans cette approche une ligne de texte est balayée horizontalement de gauche à droite. S'il existe moins de deux Pixels dans une colonne verticale, le balayage retourne "0", sinon il retournera le nombre de Pixels dans cette colonne. Un histogramme de balayage horizontal est alors construit. S'il existe dans l'histogramme une tranche de K "0" consécutifs (la valeur de K est déterminée expérimentalement) alors le point médian de cette tranche est considéré comme la frontière d'un caractère. L'approche considérée combine le profil de projection horizontal avec étiquetage de

composants connexes pour séparer les caractères chevauchés. Si la distance entre deux frontières consécutives est grande (une fausse segmentation est détectée dans cette position) alors cet étiquetage de composants connexes est employé pour une segmentation ultérieure. Au cours de l'étiquetage le recouvrement vertical des composants est vérifié. Si deux composants ou plus sont entièrement recouverts verticalement, les composants représentent donc différentes parties du même caractère. Par contre, si le recouvrement horizontal entre deux boîtes de deux composants consécutifs est moins de 35% les deux composants, font partie de deux caractères différents.

Un autre exemple de la segmentation par profil de projection est le travail présenté dans [S.Ouchtati et al, 2007]. Ce travail est la réalisation d'un système de la lecture automatique des chaînes numériques manuscrites (constituées d'un nombre variable de chiffres) en employant le profil de projection vertical pour segmenter la chaîne numérique en chiffres isolés tout en exploitant les transitions de pixel noir/blanc (ou blanc/noir) par balayage horizontal de l'image à travers le profil de projection vertical. Chaque chiffre est présenté séparément à l'entrée du système. Ce système est un réseau de neurones dont l'apprentissage est réalisé par rétropropagation de gradient et les vecteurs caractéristiques sont extraits à partir d'images binaires des chiffres segmentés par plusieurs méthodes : séquences de distribution et les moments centrés des différents profils et projections.

3.3.2 La segmentation par reconnaissance

La segmentation par reconnaissance est un processus basé reconnaissance, une lettre ne peut être segmentée sans être reconnue et ne peut être reconnue sans être segmentée, c-à-d la segmentation et la reconnaissance sont faites en même temps. La reconnaissance interprétant l'exactitude syntaxique ou sémantique des hypothèses de la segmentation est un processus qui sert généralement pour valider ou rejeter une hypothèse [N. Arıca, 1998], [R.G. Casey et al, 1982], [V.A. Kovalevsky, 1968]. Les méthodes de segmentation par reconnaissance peuvent être, eux-mêmes distinguées en deux classes [T. Saba et al, 2010].

3.3.2.1 Les approches de Fenêtrage

Dans cette catégorie, des hypothèses de segmentation se sont générées en balayant l'image. La détection des points de coupure est confirmée ou validée par reconnaissance [T. Saba et al, 2010]. Le principe de base est d'employer une fenêtre glissante de largeur variable pour fournir des séquences de segmentation expérimentales qui doivent être confirmées par reconnaissance de caractères [N. Arıca, 1998]. Des approches séquentielles et parallèles ont été employées. L'approche séquentielle identifie des mots itérativement en glissant la fenêtre de gauche à droite. L'approche parallèle procède d'une manière plus globale. Elle produit une grille d'attributs aux combinaisons possibles de lettres. La décision finale est trouvée en choisissant le chemin optimal à travers le graphe des attributs possibles [C. Fang, 1997].

3.3.2.2 Les approches basées attributs caractéristiques

Dans ce type d'approche, les attributs caractéristiques de l'image sont extraits, les correspondances possibles entre les attributs et les lettres sont ensuite générées, et puis une recherche de la meilleure combinaison parmi toutes les correspondances est effectuée [T. Saba et

al, 2010]. Ce type d'approche réalise une segmentation implicite par classification des sous-ensembles d'attributs spatiaux de l'image. Cette méthode peut être divisée en deux catégories : *Les approches basées modèle de Markov caché (*HMM) et les approches *non-Markov*. L'objectif des modèles de Markov cachés est de modéliser les variations de l'écriture cursive comme structure probabiliste qui n'est pas directement observable. L'idée fondamentale des modèles de Markov cachés est de modéliser le texte manuscrit comme un processus de Markov. Ce processus permet d'affecter des probabilités de transition aux diverses combinaisons de lettres.

Les modèles de Markov cachés peuvent être employés pour la segmentation des caractères parce que les contraintes contextuelles telles que les fréquences de mots et de lettres aussi bien que des règles syntaxiques, peuvent être formulées par des probabilités de transition entre les différents états d'un caractère [M. Gilloux, 1994].
Des exemples d'utilisation des HMMs pour la segmentation et la reconnaissance du texte manuscrit ont été rapportés dans [L.D.Harmon, 1972], [C. Fang, 1997]. Les HMMs ont été employés dans [R. Nag et al, 1986] pour la reconnaissance des chiffres littéraux et dans [M. Gilloux et al, 1993] pour la reconnaissance du montant des chèques bancaires.

Un exemple de la segmentation par reconnaissance basée sur le modèle de Markov caché est la méthode de la reconnaissance des chaînes numériques proposée dans [A.D.S.B. Jr et al, 2001]. Cette approche suit deux étapes : La première emploie une méthode de segmentation implicite et d'une information contextuelle de la chaîne numérique pour fournir les chemins de segmentation possibles. Les hypothèses produites par cette étape sont ensuite reclassifiées dans une étape de vérification, cette dernière est basée sur un classificateur de chiffres manuscrits.

La 1ère étape de cette approche trouve les "N" meilleurs chemins de segmentation pour une chaîne numérique en utilisant la programmation dynamique afin d'apparier les modèles de Markov cachés (HMMs) contre une chaîne numérique donnée.

Dans ce processus, la segmentation d'une chaîne numérique manuscrite est réalisée lorsque tous ses composants sont identifiés par reconnaissance en tant que chiffres isolés. L'information contextuelle utilisée pendant l'apprentissage du Modèle de Markov caché est la pente des caractères aussi bien que la variation de la taille intra chaîne. La taille intra chaîne est la taille de la fenêtre entourant une chaîne numérique. Cette taille est employée pour effectuer la segmentation implicite. Ainsi, l'extraction des attributs de chaque chaîne d'apprentissage est faite en tenant compte la variation de la taille intra chaîne et la pente estimée de la chaîne originale afin de corriger les chiffres isolés.

Les approches Non_*Markov* telles que la relaxation probabiliste et le test d'hypothèse ont été employés dans la segmentation par reconnaissance. Hayes [K.C.Hayes, 1980] a utilisé une méthode de relaxation probabiliste pour lire des mots manuscrits. Le modèle utilise une description hiérarchique des mots dérivés par relaxation d'une représentation squelettique. La relaxation a été exécutée sur le graphe des primitives (sous lettres) et des lettres où toutes les segmentations possibles sont représentées [C. Fang, 1997].

3.3.3 La Segmentation hybride :

Les méthodes hybrides combinent des méthodes de dissection et les méthodes de recherche des points de coupure par reconnaissance de manière hybride. Un algorithme de dissection est appliqué à l'image, mais l'intention est de couper l'image suffisamment en plusieurs endroits de telle sorte que les frontières correctes de segmentation sont incluses dans les coupures réalisées. Une fois que ceci est assuré, la segmentation optimale est cherchée par évaluation des sous-ensembles des coupures faites. Chaque sous-ensemble implique une hypothèse de segmentation, et la classification est appliquée pour évaluer les différentes hypothèses et ainsi choisir la segmentation la plus prometteuse [H.Fujisawa et al, 1992], [N. Arıca, 1998].

Un exemple de ces méthodes est l'algorithme de segmentation décrit dans [R.Palacios et al, 2003] qui produit une séquence de chiffres et de symboles, et puis reçoit un feed-back avec le module de reconnaissance pour réajuster les partitions si nécessaire. La segmentation est correcte si tous les chiffres sont reconnus avec une confiance proportionnelle. Le système commence la segmentation par séparations évidentes de caractères, ainsi les chiffres qui ne sont pas correctement identifiés sont considérés comme étant composés de caractères connectés. Les blocs rejetés sont encore divisés et reconnus dans une boucle de feed-back jusqu'à ce que la solution finale soit trouvée avec tous les segments reconnus. Le système fusionne également les fragments des chiffres avec les chiffres voisins pour créer de nouveaux segments. Si ces segments ne sont pas encore identifiés, alors différentes stratégies de fusion seront examinées. Dans la pratique, plusieurs chiffres touchent leurs voisins, et constituent donc un composant connexe. Dans ce cas, le système applique un algorithme de division de contour pour trouver les chemins possibles permettant la séparation des caractères recouvrés, tels que l'analyse de contour, l'algorithme hybride Drop Falling et l'algorithme étendu de Drop Falling que nous allons présenter par la suite.

Ce système évite les tentatives inutiles de reconnaissance autant que possibles, il emploie donc une approche heuristique pour choisir un chemin et puis sert de la meilleure information venant du module de reconnaissance dans une boucle de feedback. Des chemins sont choisis de manière heuristique en utilisant des attributs structurels. Plusieurs caractéristiques ont été utilisées pour choisir les meilleurs chemins de coupure telles que :

1. *Nombre de coupures faites pour diviser le segment*
2. *Longueur de la coupure*
3. *Bordure de la coupure*

La segmentation est correcte si les deux chiffres sont reconnus. Si aucun caractère n'a été correctement identifié (comme chiffre) avec un niveau de confiance élevé, alors des chemins alternatifs seront examinés. Si l'une des deux parties est identifiée, alors l'autre partie est composée de caractères connectés et le procédé de segmentation sera répété récursivement pour cette partie jusqu'à ce qu'une solution complète soit trouvée. Pendant la reconnaissance du montant chiffre, le système fait de multiples tentatives de fusion et séparation. Lorsque de petits morceaux de caractères sont identifiés en tant que `` fragments de caractère'', ils seront fusionnés avec le segment voisin en tenant compte des critères tels que la proximité et le chevauchement de caractères.

3.3.4 La segmentation holistique

Les approches holistiques essayent d'identifier les mots tout entiers, évitant de ce fait la nécessité de segmenter un mot en caractères. La reconnaissance est basée sur la comparaison d'un ensemble d'attributs simples extraits à partir du mot entier contre un lexique des chaînes représentant la forme théorique des mots possibles. Des mots sont représentés par une liste d'attributs tels que les ascendantes, les descendantes, les boucles, ...etc. Ces techniques sont généralement fondées sur les chaînes de Markov cachées ou la programmation dynamique [S. Madhvanath et al, 1993], [N. Arıca, 1998].

Un exemple de segmentation holistique basée programmation dynamique est l'algorithme présenté dans [G. R. Ball et al, 2006].

3.4 Les méthodes de détection des points de coupures

La plupart des méthodes de segmentation identifient les blocs isolés de Pixels en tant que composants connexes. Dans l'écriture cursive, plusieurs caractères sont chevauchés avec leurs voisins qui seront ainsi considérés en tant que composants connexes difficilement reconnaissables par un système de reconnaissance. Dans ce cas, des algorithmes de segmentation appliquent une division du contour des composants connexes pour trouver les chemins possibles séparant les caractères chevauchés tels que Hit-and-deflect, l'analyse de contour, l'algorithme hybride drop-and-falling, et l'algorithme étendu drop-and-falling, ...etc.

3.4.1 L'algorithme Hit-and-deflect

Les algorithmes Hit-and-deflect essayent de trouver un chemin optimal de coupure d'un composant connexe en frappant et en guidant un point à travers le composant connexe [M. Shridhar et al, 1986]. Shridar *et al* ont employé l'algorithme Hit-and-deflect afin de segmenter des mots manuscrits. Leur système commence par séparation des caractères à travers un balayage vertical près du milieu de l'image d'entrée. Si le balayage avait comme résultat zéro ou deux transitions de pixel noir à blanc (blanc à noir), alors les caractères sont supposés non connectés ou simplement connectés (connectés à travers un seul point). Dans ce cas, le balayage fournit le chemin approprié de segmentation. D'autre part, si la ligne de balayage traverse plus de deux transitions, les caractères sont supposés chevauchés. A ce moment, l'algorithme Hit-and-deflect est employé. L'algorithme commence à partir d'une crête du contour inférieur du composant connexe. Il essaye alors de se déplacer vers le haut suivant une série de règles. Ces règles sont conçues pour réduire au minimum le nombre de coupures qui doivent être faites à un composant connexe. L'algorithme termine une fois qu'il atteint le niveau supérieur de l'image [S.Khan, 1998].

3.4.2 L'algorithme Drop-and-Falling

L'algorithme Drop and Falling [S.Khan, 1998] est basé sur le concept d'une goutte d'eau placée au niveau supérieur de l'image et glissant vers le bas le long d'une coupure. Si la goutte d'eau est emprisonnée dans un coin, une coupure est considérée qui traverse cette partie de la figure de haut en bas. Les décisions critiques qui doivent être prises pendant le traitement sont le choix du point de départ et les règles régissant le mouvement de la goutte. L'algorithme résulte en quatre

variations suivant la position initiale et la direction de la goutte d'eau (ascendante, descendante, à gauche ou à droite) et les coupures produites peuvent être meilleures ou mauvaises selon les situations considérées pour séparer les caractères chevauchés.

3.4.3 L'algorithme des points de coupure critique

Cet algorithme est basé sur l'observation que dans la plupart des cas où les caractères sont chevauchés, la jonction entre ces caractères forme une ou plusieurs régions concaves le long de la coupure. Par conséquent, si les points d'un coin concave sont identifiés, ceux-ci peuvent être utilisés pour trouver les points corrects de coupure. Cet algorithme commence au centre horizontal du caractère connecté et cherche de haut en bas les points critiques de coupure. Il recherche, à partir du centre horizontal, toutes les paires de points de coupure jusqu'à ce qu'une paire appropriée soit trouvée. La décision de la meilleure paire est prise basée sur des critères tels que leur distance, leur distance par rapport au centre et l'angle de la coupure. Les coupures traversant un espace blanc sont interdites [S.Dey, 1999].

3.4.4 Minima des profils horizontaux et verticaux

Lorsque les caractères voisins sont chevauchés horizontalement, le profil de projection vertical ne donne pas les points de segmentation corrects. Baird [H.S.Baird, 1987] a proposé une méthode dans laquelle le rapport de la dérivée seconde du profil de projection par rapport à sa hauteur est employé comme critère de choix des points de segmentation :

Les minima du profil inférieur sont les lieux de coupure les plus naturels pour les ligatures "naturelles" [S.W. Lee et al, 1996], [K. Koo et al, 2009].

Les minima du profil supérieur sont bien adaptés pour séparer les lettres en contact au niveau supérieur de l'image [S.W. Lee et al, 1996], [K. Koo et al, 2009].
Les minima du profil médian sont utilisés pour trouver les points de coupure cachés par des ascendantes et descendantes de lettres [S.W. Lee et al, 1996], [K. Koo et al, 2009].

L'analyse des ligatures longues permet de placer des points de coupure là où il n'y a pas des minima de profils [S.W. Lee et al, 1996], [K. Koo et al, 2009].

3.5 Exemples de méthodes de la segmentation de caractères

3.5.1 La segmentation par réseau de neurones

Dans le travail de [K. Bhattacharya et al, 2009] une application des réseaux de neurones pour la segmentation des caractères manuscrits de la langue Assamese a été proposée. L'apprentissage du réseau de neurones est effectué par différents caractères manuscrits, ensuite, le texte est segmenté en lignes par une méthode statique. Les différents caractères séparés, à travers une segmentation préliminaire, sont fournis au réseau de neurones pour confirmer la segmentation par reconnaissance.

Les étapes de la segmentation basée réseau de neurones peuvent être résumées comme ci-dessous :

- le réseau de neurones est entraîné par tous les caractères Assamese disponibles.
- Segmentation du texte en phrases par la méthode statique.
- Segmentation préliminaire des phrases obtenues en caractères par intervalles de 2 : 5% de la longueur de chaque phrase.
- Chaque segment obtenu est fourni au réseau de neurones pour sa reconnaissance.
- Si le réseau ne reconnaît pas le segment courant, le prochain segment est considéré.

Ce processus est répété jusqu'à trouver les frontières correctes de la segmentation.

3.5.2 La méthode du réservoir d'eau

Dans le travail de [N.Tripathy et al, 2004], une méthode de segmentation du texte Oriya manuscrit en lignes, mots et puis en caractères, a été proposée. Les projections d'histogramme horizontales ont été employées pour segmenter le texte en ligne. Les lignes sont ensuite segmentées en mots basée sur le profil de projection vertical et des attributs structurels des caractères Oriya, les caractères isolés et chevauchés sont ensuite détectés et segmentés par la méthode du réservoir d'eau en employant des attributs structurels et topologiques.

Principe de la méthode du réservoir d'eau.

Le principe est comme suit : Si l'eau est versée du niveau supérieur (ou le fond) d'un composant, les régions concaves du composant où l'eau est emprisonnée sont considérées en tant que réservoirs supérieurs (ou inférieurs).

Les concepts de base employés dans cette méthode sont [N. Tripathy et al, 2004] :

Réservoir supérieur : le réservoir obtenu quand l'eau est versé du coin supérieur du composant.

Réservoir inférieur : le réservoir obtenu quand l'eau est rempli à partir du fond de composant. Un réservoir inférieur est visualisé comme un réservoir supérieur quand l'eau est versée du coin supérieur après rotation du composant par 180.

Réservoir (droit) gauche : Si l'eau est versée du coin (droit) gauche d'un composant, les régions concaves du composant où l'eau est emprisonnée sont considérées les réservoirs (droits) gauches. Un réservoir (droit) gauche d'un composant est visualisé comme réservoir supérieur quand l'eau est versée à partir du coin supérieur après rotation par 90 dans le sens horaire (dans le sens antihoraire).

Région du réservoir d'eau : est la région concave où l'eau peut être stockée si l'eau est versée d'un coin spécifique du composant. Le nombre de Pixels à l'intérieur du réservoir est calculé, ce nombre est considéré comme étant la région du réservoir.

Niveau d'écoulement de l'eau : Le niveau, à partir lequel, l'eau déborde le réservoir s'appelle le niveau d'écoulement de l'eau.

Ligne de base du réservoir : est la ligne qui passe par le point le plus profond d'un réservoir parallèlement à son niveau d'écoulement d'eau.

La hauteur du réservoir : la profondeur de l'eau dans le réservoir. En d'autres termes, la hauteur d'un réservoir est la distance normale entre la ligne de base et le niveau d'écoulement de l'eau du réservoir.

Largeur du réservoir : la distance normale entre deux frontières extrêmes (perpendiculaires à la ligne de base) du réservoir.

Pour chaque réservoir, sa région est calculée. Les points de la région sont les points de frontière du réservoir ayant une hauteur moins que 2 ×*RL* à partir de la ligne de base du réservoir.
La valeur de *RL est* calculée comme suit : Le composant est balayé horizontalement et verticalement, si pour un composant, on obtient "n" longueurs différentes d'écoulement d'eau : r1, r2, …, rn avec des fréquences f_1, f_2,… f_n respectivement, alors la valeur de RL = ri où *fi* = max (*fj*), j = 1…n.

L'algorithme commence par détection des composants connexes et isolés selon les observations suivantes : deux caractères sont connectés si l'une des situations suivantes est vérifiée :

(a) Deux caractères consécutifs créent un grand réservoir inférieur**.**
(b) le nombre de réservoirs et de boucles dans un composant connexe est plus grand que celui d'un composant isolé.
(c) deux caractères consécutifs créent un petit réservoir supérieur près de la ligne moyenne.
(d) la forme du caractère chevauché est plus complexe que celle des caractères isolés.

Les caractères isolés et chevauchés seront, par la suite, identifiés basés sur des fonctions booléennes appliquées sur les différents attributs obtenus par les observations ci-dessus et des caractéristiques des caractères Oriya, telles que le nombre de boucles, la hauteur, la largeur et le nombre de réservoirs dans un composant connexe.
Pour segmenter un composant connexe, l'algorithme commence par localiser la position du chevauchement (au niveau supérieur, inférieur ou médian). Le composant est ensuite segmenté basé sur la position du chevauchement, les points de la région de base d'un réservoir, et des attributs topologiques et structurels du composant connexe.

Par exemple, la segmentation d'un chevauchement supérieur se fait en connectant le point du fond le plus bas du réservoir supérieur et le point le plus bas d'un réservoir inférieur.

Conclusion

3.6 Conclusion

La segmentation de caractères est une tâche pénible due à la similarité visuelle de caractères aussi bien que sa forte dépendance avec la reconnaissance ; on ne peut pas reconnaître les caractères chevauchés sans faire leur segmentation aussi leur segmentation est difficile sans faire leur reconnaissance.

Les anciennes méthodes de la segmentation basées sur l'analyse du profil de projection et l'estimation de pitch assurent de bons résultats sur les caractères ayant des espacements de même taille. L'analyse de composants connexes a été introduite pour traiter les ligatures et la police italique. Les évolutions récentes combinent la segmentation avec la reconnaissance comme feedback pour améliorer les décisions de segmentation. Les modèles de Markov cachés faisant partie des méthodes basées reconnaissance s'adaptent mieux aux variations de l'écriture cursive.

Dans le cas des caractères chevauchés, des algorithmes tels que Hit-and-deflect, l'analyse de contour, l'algorithme hybride drop falling, l'algorithme étendu drop falling. Minima des profils horizontaux et verticaux.. Sont employés pour diviser le contour des composants connexes afin de trouver les meilleurs chemins séparant les caractères chevauchés

Les méthodes de segmentation hybrides combinent es méthodes de recherche des points de coupure avec la reconnaissance de manière hybride pour réajuster les partitions si nécessaire.
Les approches holistiques essayent d'identifier les mots tout entiers, évitant de ce fait les erreurs de la segmentation. Ces techniques sont généralement basées sur les chaines de Markov cachées ou la programmation dynamique.

Chapitre 4
EXTRACTION DES ATTRIBUTS CARACTERISTIQUES

4.1. Introduction :

L'extraction des attributs caractéristiques est l'une des étapes les plus importantes pour la réussite d'un système de reconnaissance d'écriture manuscrite. C'est à partir les caractéristiques extraites du mot à reconnaître qu'il est possible d'améliorer la robustesse d'un système de reconnaissance. La segmentation de caractères est le processus qui trouve un segment particulier et de l'isoler du reste de l'image. Cet objet doit être identifié par des caractéristiques significatives mises dans son vecteur d'attributs, après la détermination de ces vecteurs d'attributs pour tous les caractères, il est possible de les faire séparer en les affectant à plusieurs classes pré connues qui représentent tous les types possibles de caractères attendus. Les vecteurs d'attributs doivent être invariables pour les objets de la même classe, c.-à-d. ils doivent être indépendants aux : translation, la rotation et le facteur d'échelle (scalling).

Dans la pratique, le choix des attributs caractéristiques pour une certaine application est une tâche difficile où plusieurs méthodes ont été proposées comme solutions au problème du choix des meilleurs attributs caractéristiques, telles que les méthodes de sélection du meilleur sous-ensemble des attributs caractéristiques [A. Jain et al, 1997], [K. Kira et al, 1992], les méthodes qui automatisent la génération des attributs caractéristiques [P.D. Gader et al, 1996] ou encore les méthodes basées clustering pour grouper les attributs caractéristiques. Récemment, les systèmes de reconnaissance emploient des attributs caractéristiques extraits par multi-résolution qui se sont avérés très avantageux dans la classification que d'autres méthodes utilisant des attributs simples. Un des systèmes de la reconnaissance basé multi-résolution est le système décrit dans [J. T. Favata et al, 1996].

Ce chapitre est consacré à la présentation des différents extracteurs d'attributs caractéristiques de la littérature. Les avantages et les inconvénients de quelques méthodes sont également présentés.

4.2 L'extraction des attributs caractéristiques

L'extraction de caractéristiques est l'opération d'extraction des attributs les plus appropriés pour classifier un caractère afin d'augmenter le taux de la reconnaissance dans un OCR. Ces attributs constituent les informations pertinentes permettant la distinction des objets appartenant à différentes classes. Dans cette étape chaque caractère est représenté comme étant un vecteur d'attributs, qui devient son identité. Le but principal est d'extraire un ensemble d'attributs, qui maximisent le taux de reconnaissance avec la moindre quantité d'information. L'obtention des attributs est une tâche difficile en raison de la nature très variable de l'écriture [G. Vamvakas et al, 2010], [G. Vamvakas et al, 2009]. Les attributs caractéristiques sont supposés invariables pour les objets d'une même classe, c'est-à-dire ils doivent être indépendants en translation, rotation et le facteur d'échelle en fonction de l'application [L.A.Torres-Méndez et al, 2000]. Généralement cette pertinence est réalisée à partir de la corrélation de plusieurs types d'attributs caractéristiques [Asif Iqbal et al, 2008].

Pour certaines méthodes d'extraction des attributs caractéristiques, les caractères peuvent être reconstruits à partir les vecteurs extraits d'attributs caractéristiques [F.P.Kuhl et al, 1982], [A.Khotanzad et al, 1990]. Cette propriété de reconstruction est utile pour vérifier si l'implémentation est correcte ou pas [Ø.D. Trier et al, 1996].

Avant de choisir une méthode spécifique d'extraction des attributs caractéristiques, on doit considérer le système de reconnaissance tout entier dans lequel cette méthode va être employée et pour quel type de caractères le système est-il conçu ? Caractères imprimés, caractères multi police, caractères manuscrits ou non contraints ? [Ø.D. Trier et al, 1996].

Une méthode d'extraction de caractéristiques toute seule n'est pas suffisante pour aboutir à la meilleure puissance de discrimination. Une solution évidente est de combiner des attributs de différentes méthodes d'extraction de caractéristiques. Si un grand ensemble d'apprentissage est disponible, l'analyse discriminante peut être employée pour choisir les attributs ayant la puissance distinctive la plus élevée. Une autre approche est d'employer de multiples classificateurs [J.Cao et al, 1994], [L.Xu et al, 1992]. Dans ce cas, on peut combiner des classificateurs statistiques, structurels et des réseaux de neurones pour bénéficier de leurs différences inhérentes [Ø.D. Trier et al, 1996].

4.3 Les techniques d'extraction des attributs caractéristiques

Les techniques d'extraction de caractéristiques varient selon la nature d'image analysée et la nature du classificateur utilisé (structurel, statistique ou arbre de décision) [G. Vamvakas et al, 2009].
Les méthodes d'extraction de caractéristiques sont basées sur trois grands types d'attributs [G. Vamvakas et al, 2010], [G. Vamvakas et al, 2009] :

1 statistique
2 structurel
3 - Transformations globales et moments

Les attributs statistiques sont basés sur une distribution statistique pour représenter un caractère. Les plus utilisés sont :

- ✓ le zonage : le caractère est subdivisé en zones et les attributs caractéristiques sont extraits par calcul de la densité de chaque zone [L.S. Oliveira et al, 2001] ou par mesure de la direction du contour de chaque zone en utilisant les histogrammes des chaînes de codage [K. M. Mohiuddin et al, 1994].
- ✓ Les profils de projection [A.L. Koerich et al, 2003]
- ✓ les croisements et les distances [J.H. Kim et al, 2000b].

Les attributs structurels sont basés sur des propriétés topologiques et géométriques du caractère, telles que, les lignes de référence, les ascendantes et les descendantes, les points de croisement, les primitives et leurs directions, inflexion entre deux points, les courbes horizontales, etc... [N. Arica et al, 2001].

Une description détaillée des méthodes d'extraction de caractéristiques peut être trouvé dans [O.D. Trier et al, 1996].

4.3.1 Les attributs statistiques

Les principaux attributs statistiques comme déjà mentionnés sont [G. Vamvakas et al, 2010], [G. Vamvakas et al, 2009] :

1 Le zonage
2 Les projections et les profils
3 Les croisements et les distances

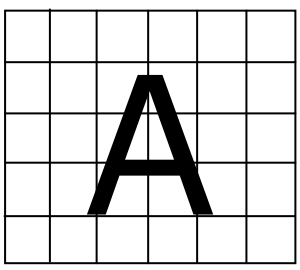
4.3.1.1 Le Zonage

Cette technique est basée sur le principe d'imposer une grille N*M au caractère et donc de diviser l'image du caractère en un nombre de zones où des caractéristiques sont extraites de chaque zone pour former le vecteur d'attribut. Le but du zonage est d'obtenir des caractéristiques locales au lieu des caractéristiques globales [L.S. Oliveira et al, 2001].

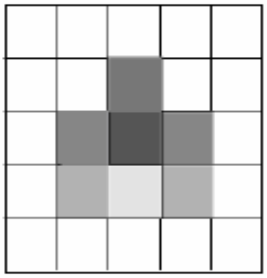
En fait, le Zonage n'est pas toujours employé comme méthode d'extraction de caractéristiques mais comme procédure auxiliaire utilisée en combinaison avec d'autres extracteurs d'attributs caractéristiques. Dans la méthode de zonage, l'image d'entrée est subdivisée en zones comme une grille rectangulaire. Puis une certaine méthode d'extraction de caractéristiques est appliquée séparément à chacune de ces zones. Le vecteur d'attributs final est formé par concaténation des attributs de ces zones [J. Laaksonen, 1997].

Exemples d'attributs extraits par Zonage :

1. Le calcul de la moyenne du niveau de gris ou du niveau binaire de chaque zone



Caractère original



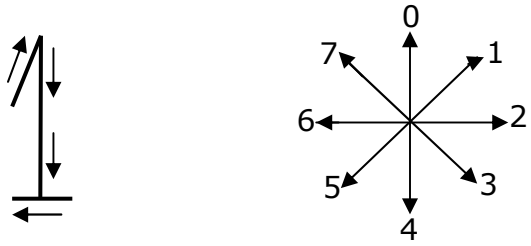
Caractère en niveau de gris

Figure 4.1: exemple de zonage

2. Calcul de la densité en pixels de chaque zone : La densité représente le rapport du nombre de pixels noirs formant le caractère sur la taille totale de la zone [J.R. Parker, 1993].

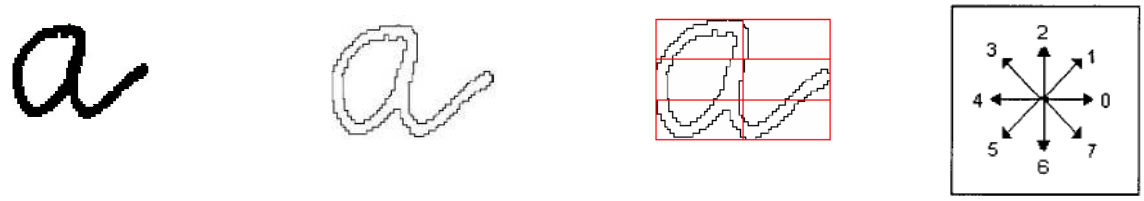
3. Calcul des attributs directionnels, de chaque zone, basé sur l'extraction du contour suivie d'une étape de calcul d'un histogramme directionnel en analysant les Pixels adjacents dans un voisinage de 3x3 [K.M. Mohiuddin et al, 1994]. Pour chacune des huit directions, le vecteur qui porte l'indice de cette direction est utilisé pour compter le nombre de pixels noirs formant le contour du caractère. A chaque fois qu'un pixel noir est rencontré dans une direction, l'élément du vecteur qui porte l'indice de cette direction est incrémenté comme indiqué ci-dessous [J.R. Parker, 1993], [S. Njah et al, 1998] :

L'histogramme des directions du chiffre 1 est représenté par la figure suivante :



	1			2		1	
0	1	2	3	4	5	6	7

Figure 4.2: Histogramme des directions du chiffre 1 [S. Njah et al, 1998]

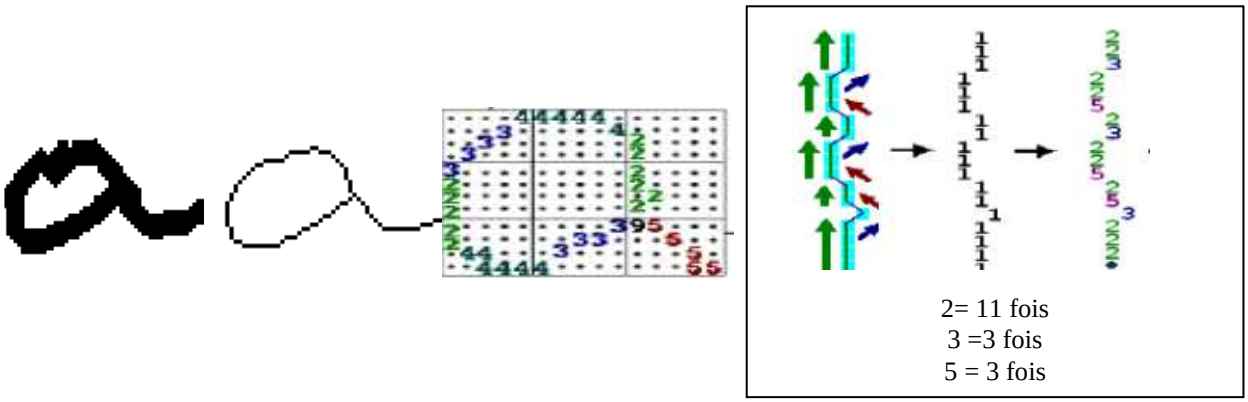


Caractère original | Extraction de contour | Zonage | Attribut directionnel dans un voisinage 3 * 3

Figure 4.3 : calcul des attributs directionnels du caractère « a »

4. Calcul des attributs directionnels en utilisant le codage de Freeman. Cette technique est basée sur la détection de contour par l'algorithme de canny, le calcul du gradient dans les directions X et Y, le calcul des grandeurs de phases du gradient et enfin calcul des valeurs de direction quantifiée selon la direction 8-Freeman. Cette technique est variante en taille et rotation et peut donner de faux contours [A. Shubair, 2007].

5. Calcul des attributs directionnels,de chaque zone, basé sur la squelettisation suivie d'une étape d'étiquetage selon la position du segment dans la zone : (vertical (2), horizontal (4), diagonal droit(3), diagonal gauche (5)) ensuite pour chaque zone et pour chaque direction le nombre de traits et leurs longueurs sont calculés comme montré ci-dessous [M. Blumenstein et al, 2003].



Caractère original | squelettisation | Zonage et étiquetage | Attributs directionnels

Figure 4.4: Attributs directionnels du caractère « A » par étiquetage

4.3.1.2 Les histogrammes de projection

Les histogrammes de projection sont généralement employés pour segmenter des lignes, des mots, et des caractères dans un document manuscrit. Ils transforment une image bidimensionnelle en deux histogrammes unidimensionnels : horizontal et vertical [J. Laaksonen, 1997], [A.L. Koerich, 2003]. Ces attributs sont bien indépendants au bruit et déformation. En raison de leur sensibilité aux petites rotations, les attributs caractéristiques de ce type ne sont pas efficaces tous seuls. Ils se sont généralement combinés avec d'autres types d'attributs pour offrir des informations utiles [J. Laaksonen, 1997].
Le principe de cette méthode est de calculer le nombre de Pixels placés en lignes et colonnes en se servant respectivement de l'histogramme de projection vertical et l'histogramme de projection horizontal. Les caractéristiques extraites sont la position et l'endroit des crêtes (peaks). Les histogrammes de projection peuvent séparer des caractères tels que "m" et "n" mais exigent souvent l'utilisation des histogrammes horizontaux et verticaux en même temps pour aboutir à des meilleurs résultats [A.L. Koerich, 2003].

Figure 4.5: Les histogrammes de projection horizontal et vertical du caractère " a "[G. Vamvakas et al, 2007a]

Histogramme du codage de Freeman

C'est une méthode de projection basée contour qui commence par localiser les points du contour d'un objet d'une image binaire à travers l'utilisation d'une fenêtre 3× 3. Un point est considéré point de contour s'il est connecté à quatre voisins qui sont des points de fond comme montré dans la Figure 4.6

	X	
X	P	X
	X	

Figure 4.6 : Détection d'un point de contour

Le suivi du contour par la méthode Freeman produit une représentation appelée " chaîne de codage" comme montré dans la Figure 4.7 [S. Arora et al, 2009].

pour chaque Pixel du contour est assigné un code indiquant la direction (sens horaire ou antihoraire) du prochain Pixel appartenant au même contour.

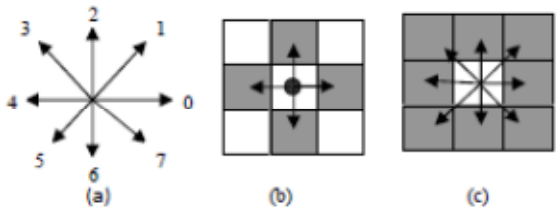


Figure 4.7: chaînes de codage, (a) direction de la connectivité (b), 4-connectivité (c) 8-connectivité.

S'il existe plusieurs connexions dans le même caractère, il peut y avoir plusieurs codages pour représenter le contour. Il est donc préférable de subdiviser l'image en zones de tailles égales d'où un histogramme de chaînes de codage est calculé pour chacune des zones [S. Arora et al, 2009].

4. 3.1.3 Les profils de projections

Le profil estime le nombre de Pixels (distance) entre la boîte qui entoure l'image du caractère et les bords du caractère. Les profils décrivent bien la forme externe du caractère et permettent la distinction d'un grand nombre de lettres, telles que "p" et "q". Comme ils peuvent être employés au contour du caractère [G. Vamvakas et al, 2007a].

Figure 4.8: Les profils du caractère " a " [G. Vamvakas et al, 2007a].

Mais on peut s'attendre à des cas d'ambiguïtés tel que le cas suivant :

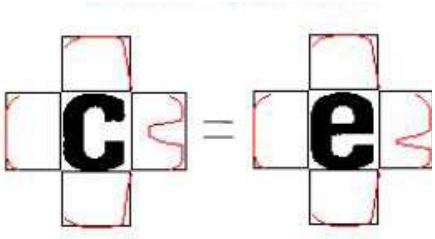


Figure 4.9: Profils de projection : cas d'ambiguïté

4.3.1.4 Croisements et distances

Les croisements estiment le nombre de transitions verticales et horizontales entre les pixels du fond de l'image et les pixels du caractère. *Les distances* par contre estiment les distances verticales et horizontales entre les frontières supérieures et inférieures, gauches et droites du caractère [J. H. Kim et al, 2000b].

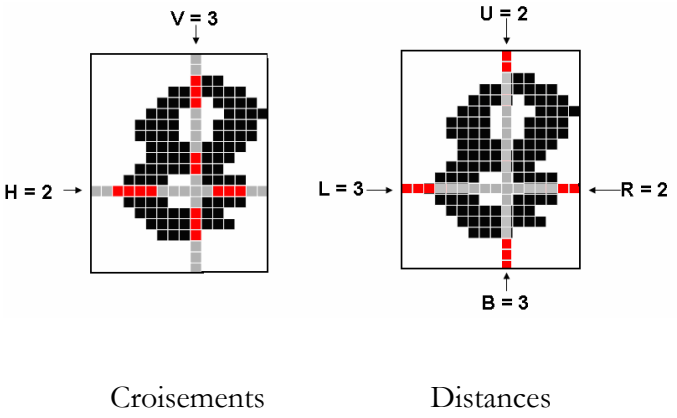


Figure 4.10: Croisements et distances du chiffre 8

4.3.2 Les attributs structurels

Les caractères peuvent être représentés par des attributs structurels ayant une tolérance élevée aux déformations et aux variations de style d'écriture. Ce type de représentation peut coder des connaissances qui concernent la structure de l'objet ou la structure de ses composants. Les attributs structurels sont basés sur des propriétés topologiques et géométriques du caractère, telles que, les points d'intersections, les boucles, les primitifs et leurs directions, l'inflexion entre deux points, les courbes horizontales en haut ou en bas du caractère… [G. Vamvakas et al, 2007b].

D'autres méthodes d'extraction d'attributs structurels sont basées sur des représentations graphiques ; le caractère est d'abord divisé en un ensemble d'attributs topologiques, tels que, les boucles, les points, … etc. Ces attributs sont ensuite représentés par des graphes relationnels attribués [N. Arica et al, 2001]. Il existe deux types de représentation graphiques. La première emploie les coordonnées de la forme du caractère [H. Cheng et al, 1993]. La seconde est une représentation abstraite avec des nœuds correspondant aux attributs et des lignes reliant ces attributs [W. Lu et al, 1991], [G. Vamvakas et al, 2007b].

4.3.2.1 Les intersections

Le principe est de vérifier 2,3 ou 4 connectivités d'un Pixel. Les caractéristiques extraites sont la position et le nombre d'intersections. Cette technique exige la squelettisation et elle est variable aux rotations et translations [E. Yfantis, 2006].

Les modèles de matrices à placer sur un pixel sont :

0	0	1
0	X	1
1	0	0

0	0	0
0	X	0
1	0	1

1	0	0
1	X	0
0	0	1

0	0	0
0	X	1
0	1	0

0	1	0
0	X	1
0	0	0

0	1	0
0	X	1
0	1	0

Figure 4.11: exemples des cas d'un pixel d'intersection

0 pixel blanc
1 pixel noir
X le pixel d'intersection

4.3.2.2 Les Pixels de fin de ligne

Un pixel de fin de ligne est connecté à un seul pixel de ses 8 voisins. Le principe est de placer un masque de taille 3x3 sur les Pixels noirs afin de vérifier leurs connexions avec leurs 8 voisins. Les caractéristiques extraites sont la position et le nombre de pixels de fin de ligne. Cette technique nécessite l'amincissement du caractère, en outre, elle est variable au modèle de police choisi, la rotation et la translation [E. Yfantis, 2006].

0	0	0
0	X	1
0	0	0

0	0	0
0	X	0
0	0	1

0	0	0
0	X	0
0	1	0

0	0	0
0	X	0
1	0	0

0	0	0
1	X	0
0	0	0

1	0	0
0	X	0
0	0	0

0	1	0
0	X	0
0	0	0

0	0	1
0	X	0
0	0	0

Figure 4.12: Les différents cas d'un point de fin de ligne

0 pixel blanc
1 pixel noir
X le pixel de fin de ligne

4.3.3 Les transformations globales et les moments

Cette classe comporte plusieurs types d'attributs comme :
Les moments de Zenrike
Les wavelets
Descripteur de Fourier [F. Kuhl et al, 1982]
La transformé de Hough [G. Louloudis et al, 2007]
La transformé de Gabor.
Walsh Hadamard, … etc.

4.3.3.1 Les Wavelets

Les caractéristiques extraites par wavelet sont la position et l'endroit des contours aussi bien que la moyenne et l'écart type des Pixels de chaque sous bande ou encore les coefficients approximatifs de l'image normalisée au deuxième niveau de la décomposition [B.V. Kumar et al, 2004].

Le principe est de diviser l'image *f(x, y)* en différents niveaux de fréquences L_rL_c, L_rH_c, H_cL_r, H_cH_r (avec c : colonne, r : ligne) en appliquant deux filtres lowpass(L) et highpass (H).
Pour calculer la transformée de wavelet d'une image f (x, y), les filtres H et L sont d'abord appliqués sur les lignes de l'image pour donner deux images résultats $H_r f$ (x,y) (image détaillée) et $L_r f$ (x,y) (image approximative). Les filtres H et L sont ensuite appliqués sur les colonnes de $H_r f$ (x,y) et de $L_r f$ (x,y) ce qui produit comme résultat quatre images : $H_c H_r f$ (x,y), $L_c L_r f$ (x,y), $H_c L_r f$ (x,y) et $L_c H_r f$ (x,y). Ces images seront par la suite échantillonnées en deux et puisque l'image approximative contient la majeure partie d'énergie, la décomposition est continuée sur l'image $L_c L_r f$ (x,y) [B.V. Kumar et al, 2004].

4.3.3.2 Descripteurs de Fourier

Cette technique est basée sur la représentation fréquentielle de l'image du caractère. Les caractéristiques extraites sont les coefficients de Fourier puisque ces coefficients permettent de reconstruire le contour d'un caractère. Les descripteurs de Fourier peuvent être employés pour caractériser le contour externe d'un caractère. Leurs premiers coefficients décrivent la forme la plus proche du caractère et plus que le nombre de coefficients considérés est grand, plus la quantité d'information pour représenter un caractère est augmentée [J. Laaksonen, 1997], [F. Kuhl et al, 1982].

Les descripteurs de Fourier sont basés contour et calculés dans un espace bi-dimensionnel en utilisant la méthode de Fourier [G. G. Rajput et al, 2010a] :
Le contour est d'abord échantillonné en K-points à partir d'un point arbitraire (x0, y0), leurs coordonnées correspondantes (x_0, y_0), (x_1, y_1), (x_2, y_2)...(x_{k-1}, y_{k-1}) sont ensuite déterminées en traversant le contour dans un sens anti-horaire. Ces coordonnées peuvent être exprimées sous la forme suivante : $x(k) = x_k$ et $y(k) = y_k$; ainsi, le contour pourra être noté comme suit :
S(k) = [x(k), y(k)], pour k = 0, 1, 2..... K-1.
Chaque paire de coordonnées pourra être représentée comme suit :
S(k) = x(k) + jy(k), pour k = 0, 1, 2..... K-1 ; Séquence de nombres complexes

C'est-à-dire, l'axe des abscisses est considéré comme axe des réels et l'axe des ordonnées comme axe imaginaire. La transformée discrète de Fourier (DFT) de s(k) est calculée comme suit [G.G. Rajput et al, 2010b] :

$$a(u) = \frac{1}{K}\sum_{k=0}^{K-1} s(k)\, e - \frac{j2\pi uk}{K}, u = 0,1, \ldots., K-1 \qquad (4.1)$$

Les coefficients complexes : a(u) sont les descripteurs de Fourier du contour.

L'inverse des descripteurs de Fourier reconstitue l'image (s(k)) . C'est-à-dire [G.G. Rajput et al, 2010b]

$$s(k) = \sum_{u=0}^{K-1} a(u)\, e^{\frac{j2\pi uk}{K}}, k = 0,1, \ldots., K-1 \qquad (4.2)$$

En résumé, le processus d'extraction des attributs caractéristiques de Fourier commence par tracer le contour d'un objet dans le sens antihoraire, et puis de considérer un échantillonnage uniforme de ce contour. Ensuite, le contour échantillonné est représenté dans un plan complexe. Les descripteurs de Fourier sont invariables et peuvent être calculés en annulant le descripteur de Fourier 0 (c-à-d : invariance à la translation (position)), et en divisant tous les descripteurs de Fourier par la grandeur du 1er descripteur de Fourier (invariance aux tailles) et en considérant seulement la grandeur des descripteurs de Fourier (invariance à la rotation) [G. G. Rajput et al, 2010a], [G.G. Rajput et al, 2010b].

Les descripteurs de Fourier ne peuvent pas être appliqués aux caractères fragmentés puisque cette méthode permet l'extraction des attributs d'un contour fermé. De plus, les méthodes basées contour externe n'emploient pas les informations du contour interne tel que le cas des trous dans les chiffres : 8 et 0, ainsi quelques caractères pouvant être confondus. Une des solutions possibles à ce problème est l'utilisation des classificateurs multi-échelle (multi-stage)[J.Cao et al, 1994], [Ø.D. Trier et al, 1996].

Les invariants de moment, les moments de Zernike et la transformé de Karhunen Loeve peuvent être des solutions alternatives, puisqu'ils ne sont pas sensibles aux dégradations (fragmentations) en même degré que celui des descripteurs de Fourier [Ø.D. Trier et al, 1996].

4.3.3.3 Filtres de Gabor

Le filtre de Gabor est un compromis espace-fréquence efficace pour la représentation du contenu de l'image (Weldon, 1998). Les attributs de Gabor sont invariables à la translation, la rotation, et le facteur d'échelle [K.C. Chung et al, 1999].

Une fonction 2D de Gabor est définie comme une fonction gaussienne modulée par un signal sinusoïdal [X. Wang et al, 2005], [P. Hu et al, 2002]:

$$g(x,y) = a(x,y) * c(x,y) \qquad (4.3)$$

où :
$a(x,y)$: est la composante gaussienne
$c(x,y)$: est la composante sinusoïdale
x, y sont les coordonnées spatiales

$$c(x,y) = \cos(2\pi\ (F_x\, x + F_y\, y)) \qquad (4.4)$$

$$a(x,y) = \frac{1}{(2\pi\ \sigma_x\ \sigma_y)} * e^{-0.5\left[\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right]} \qquad (4.5)$$

$\sigma x, \sigma y$ Détermine la variance gaussienne dans les axes spatiaux et spectraux
F_x , F_y : représente la fréquence de la composante sinusoïdale.

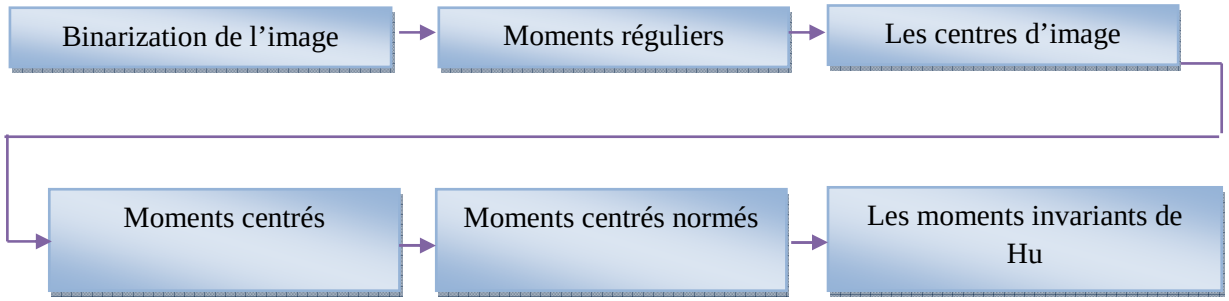
4.3.3.4 Les Moments de Hu et de Zernike

Les moments sont basés sur le calcul du centre d'un caractère tout entier aussi bien que les centres des sous-images et de calculer ensuite la distance entre les centres locaux et le centre global [P.Hu, 1962].

Les moments de Hu

Hu a introduit 7 moments non-linéaires qui sont invariables à la translation, la rotation et le facteur d'échelle. Les moments de Hu sont constitués uniquement de 7 valeurs ce qui permet de réduire le temps de calcul par rapport à d'autres méthodes d'extraction de caractéristiques.

Ci-dessous est une brève description des moments invariants de Hu [J. Villegas-Cortez et al, 2005], [R. S. Kunte et al, 2007] :



Figue 4.13 : Évaluation des moments invariants de Hu

Pour calculer ces attributs ; l'image est premièrement binarisée.

- les *moments réguliers* sont évalués sous le format :

$$m_{pq} = \iint_{-\infty}^{+\infty} x^p y^p f(x,y)\, dx\, dy\, , p,q = 0,1,2, \ldots \ldots . \qquad (4.6)$$

Où m_{pq} est le moment d'ordre (p+q) de l'image *f(x, y)*.

- Les *moments centrés* de *f(x, y)* sont définis comme suit :

$$\mu_{pq} = \iint_{-\infty}^{+\infty} (x - x_c)^p \ (y - y_c)^q \ f(x,y)\, dx\, dy, \qquad p,q = 0,1,2, \ldots \qquad (4.7)$$

Avec $x_c = \frac{m_{10}}{m_{00}}$ et $y_c = \frac{m_{01}}{m_{00}}$ sont les centres de l'image.

Pour des images numériques, les intégrales peuvent être transformées en sommes et donnent ainsi les *moments normés et centrés* respectivement :

$$m_{pq} = \sum_{x}\sum_{y} x^p y^p f(x,y) \ , p,q = 0,1,2,\dots\dots. \qquad (4.8)$$

$$\mu_{pq} = \sum_{x}\sum_{y} (x - x_c)^p (y - y_c)^q f(x,y), \ \ p,q = 0,1,2,\dots\dots.. \qquad (4.9)$$

- les moments *centrés normés* peuvent être définis comme suit :

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{y}}, \qquad y = \frac{(p+q+2)}{2}, p+q = 2,3,\dots\dots. \qquad (4.10)$$

- Les 7 moments de Hu sont définis par les équations suivantes :

$$\emptyset_1 = \eta_{20} + \eta_{02} \qquad (4.11)$$

$$\emptyset_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \qquad (4.12)$$

$$\emptyset_3 = (\eta_{30} - 3\,\eta_{12})^2 + (3\,\eta_{21} - \eta_{03})^2 \qquad (4.13)$$

$$\emptyset_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \qquad (4.14)$$

$$\emptyset_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \qquad (4.15)$$

$$\emptyset_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\,\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \qquad (4.16)$$

$$\emptyset_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{30})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \qquad (4.17)$$

Les moments de Hu sont invariables à la rotation, et quelques invariants de moment sont aussi invariables à l'inclinaison. Les invariants de moment n'ont pas la capacité de reconstruction, d'autres attributs caractéristiques sont nécessaires auxquels la reconstruction est possible [T.H.Reiss, 1993], [Ø.D. Trier et al, 1996].

Les Moments de Zernike

Les moments de Zernike sont issus des polynômes de Zernike présentés par Zernike (1934), qui constituent un ensemble de polynômes complexes formant un ensemble orthogonal à l'intérieur

d'un cercle d'unité. Les moments de Zernike sont des projections de l'image d'entrée dans l'espace traversé par les fonctions orthogonales V [R. S. Kunte et al, 2007]:

$$V_{nm}(x,y) = V_{nm}(r,\theta) = R_{nm}(r).exp(jm\theta) \quad (4.18)$$

Où, n = 0, $|m| = n$, r est le vecteur d'origine au Pixel (x, y) représentant le centre de l'image.
et θ est l'angle entre le vecteur r et l'axe des x dans la direction d'horloge.
Le polynôme radial orthogonal $R_{nm}(r)$ est défini comme suit [R. S. Kunte et al, 2007]:

$$R_{nm}(r) = \sum_{s=0}^{(n-|m|)/2} (-1)^s g \frac{(n-s)!}{s!\left(\frac{n+|m|}{2}-s\right)!\left(\frac{n-|m|}{2}-s\right)} g^{r(n-2s)} \quad (4.19)$$

Le moment de Zernike d'ordre n et de répétition m de la fonction continue d'image $f(x, y)$ est donné par [R. S. Kunte et al, 2007]:

$$Z_{nm} = \frac{n+1}{\pi} \sum_x \sum_y f(x,y)\ [V^*_{nm}(x,y)] \quad (4.20)$$

Pour calculer les moments de Zernike d'une certaine image, le centre de l'image est considéré comme coordonnées d'origine et l'image doit être représentée dans le cercle $x^2+y^2 \leq 1$.
Les Pixels qui se trouvent à l'intérieur de ce cercle sont considérés pour calculer les moments de Zernike.
Si on tourne l'image par un angle α, la fonction permettant de transformer les moments de Zernike Z'_{nm} est donnée ci-dessous [R. S. Kunte et al, 2007] :

$$Z'_{nm} = Z_{nm}.e^{-jm\alpha} \quad (4.21)$$

Les moments de Zernike sont invariables à la rotation, robuste en face du bruit et les variations mineurs, aussi, ils se sont moins redondants en termes d'informations [R. S. Kunte et al, 2007].

Les moments de Zernike semblent être très prometteurs pour la reconnaissance de caractères puisqu'ils possèdent la capacité de reconstruction des images originales [Ø.D. Trier et al, 1996].

4.3.3.5 La transformée de Karhunen-Loève:

La transformée discrète de Karhunen-Loève (KLT) est la transformée la plus utilisée dans les applications de la reconnaissance de caractères, dans cette méthode les masques noyau sont d'abord calculés à travers les valeurs propres et les vecteurs propres de la matrice de covariance des données d'apprentissage. Les premiers vecteurs propres dans l'ordre décroissant de leurs valeurs propres sont employés en tant que masques noyau [J. Laaksonen, 1997].

Considérant un ensemble de P caractères manuscrits disponibles sous forme binaire $u^{(p)}$ et de tailles N*N de Pixels. Le P ème caractère est donc une matrice $u^{(p)}$ tels que ses éléments sont [P.J. Grother, 1992] :

$$\mu_{ij}^{(p)} = f(x) = \begin{cases} +1, & si\ pixel\ noir \\ -1, & si\ pixel\ blanc \end{cases} \qquad (4.22)$$

Une image μ peut être considérée comme un vecteur de longueur N^2 constituée d'une séquence de colonnes :

$$\mu = (\mu_{11}, \mu_{21}, \ldots\ldots, \mu_{N1},\ \mu_{12}, \ldots.\mu_{N2}, \ldots\ldots, \mu_{N^2})$$

On soustrait de μ la moyenne de toutes les images et on met le résultat en colonnes dans la matrice U. La matrice de covariance, R, donne la moyenne de toutes les images de l'ensemble, et ainsi, décrit la variation statistique des images de caractères manuscrits. La matrice R est symétrique et calculée comme suit [P.J. Grother, 1992] :

$$R = UU^T \qquad (4.23)$$

La matrice de covariance R a N^2 vecteurs propres représentant les colonnes de ψ définie dans l'équation suivante [P.J. Grother, 1992] :

$$R\,\psi = \psi A \qquad (4.24)$$

Les éléments de A qui sont différents de zéro et qui se trouvent sur sa diagonale sont les valeurs propres. Les vecteurs propres sont les directions de la variance maximale dans l'espace N^2. N'importe quel vecteur constituant une colonne de la matrice U peut être exprimé comme une combinaison linéaire des vecteurs de base μ [P.J. Grother, 1992] :

$$U = \psi\, V \qquad (4.25)$$

Où l'inversion de cette formule définit la transformée de Karhunen Loève . V. et les vecteurs constituant les colonnes de V représentent les vecteurs transformés de Karhunen Loève qui seront employés comme l'entrée d'un classificateur [P.J. Grother, 1992].

$$V = \psi^T U \qquad (4.26)$$

L'utilisation de KLT dans la reconnaissance de caractères manuscrits a été suggérée par Andrews qu'il l'a conjecturée pour être supérieur par rapport à d'autres transformés comprenant Fourier, Walsh-Hadamard, et Haar. L'inconvénient majeur de cette méthode est que les vecteurs d'apprentissage de toutes les classes doivent être considérés avant que les vecteurs propres soient résolus. Par conséquent, les propriétés de distribution de classes peuvent se chevaucher dans le processus d'extraction des attributs caractéristiques [J. Laaksonen, 1997].

La transformé de Karhunen-Loeve a la meilleure compacité d'information en termes de la somme des erreurs quadratique en employant un classificateur non paramétrique comme le classificateur du K- plus proche voisin. En outre, puisque les attributs de Karhunen-Loeve sont

attachés aux positions des Pixels d'un caractère, ils ne sont pas appropriés aux classificateurs paramétriques [Ø.D. Trier et al, 1996].

4.4 L'analyse par composante principale (PCA)

PCA est une technique célèbre utilisée pour la réduction de la dimension des données sans perte d'informations ; Si un grand nombre d'attributs d'apprentissage est disponible, l'analyse par composante principale peut être employée pour choisir les attributs ayant la plus grande puissance distinctive.
PCA est basée sur l'analyse de la matrice de variance-covariance de l'ensemble des données. Partant de l'idée, s'il existe des variables qui peuvent être exprimées en fonction d'autres, nous n'avons besoin que de *"p* variables *"* qui tiennent compte de la variabilité du système.

Cette technique est basée sur les étapes suivantes [L.I. Smith, 2002]:
Étape 1 : préparer les données en vecteurs
Étape2 : soustraire la moyenne de chaque dimension des données (pour chaque dimension, on soustrait la moyenne de toutes les données).
Étape 4 : Calcule de la matrice de covariance
Étape 5 : Calcule des vecteurs et valeurs propres de la matrice de covariance
(Permettent d'obtenir des informations sur les modèles de l'ensemble des données).
Étape 6 : Choisir les composantes principales pour former le vecteur d'attributs :
d'abord, les vecteurs propres sont triés selon leurs valeurs propres, ensuite, les Premiers vecteurs propres sont choisis en tant que composantes principales.
Le vecteur_d'attribut = (vecteur-propre1, vecteur-propre2....., vecteur-propreP).
Étape 7 : Obtenir de nouvelles données : vecteur_d'attribut ^{T} * donnée_adjustée^{T}
où : le vecteur_d'attribut ^{T} est la transposée du vecteur d'attribut et
Donnée_adjustée^{T} est la transposée des données après soustraction de la moyenne de chaque dimension.

4.5 Exemples des extracteurs d'attributs caractéristiques

Extraction des attributs de caractères

La méthode proposée dans [V .N.Manjunath et al, 2005], permet de haute exactitude dans un temps réduit. Elle est aussi invariable à la rotation, le facteur d'échelle et les transformations de translation. Cette méthode obtient le contour d'un caractère par une technique de détection de contour, le résultat est ensuite utilisé pour trouver les attributs invariables pour différents caractères de différentes langues. Elle permet l'extraction des attributs caractéristiques par calcule de la distance entre le centre d'un caractère et les Pixels de son contour.

Cette méthode suit trois étapes :

Dans la première étape, un masque est employé pour détecter le contour du caractère. *Dans la seconde étape* les attributs caractéristiques sont extraits pour chaque point du contour en calculant la distance entre le centre du caractère et les pixels du contour. Le nombre de Pixels

noirs du contour divisé par la somme de toutes les distances donne un attribut invariable pour chaque point du contour.

Étape 1

- Pour chaque Pixel du caractère, utiliser la règle suivante :

Si

0	0	0
0	0	0
0	0	0

Alors

0	0	0
0	1	0
0	0	0

- Répéter ce procédé jusqu'à ce qu'aucun autre changement ne soit permis.

Étape 2 :

La méthode calcule le centre du caractère en estimant les coordonnées Xmin, Xmax, Ymin et Ymax du contour. Elle estime ensuite la distance (d) entre le centre (c) et les Pixels du contour et puis trouve la somme de toutes les distances et le nombre des Pixel noirs (n). Le rapport carré de (n) sur la somme de toutes les distances est un attribut invariable aux transformations, telles que, la rotation, le facteur d'échelle et la translation [V .N. Manjunath et al, 2005].

Extraction des attributs de mots :

Dans le travail de [S. Madhvanath et al, 1998] une approche pour la reconnaissance des mots manuscrits sans segmentation explicite est présentée.

L'approche proposée utilise des caractéristiques basées sur la représentation des régions concaves et convexes qui existent dans le corps d'un mot. De cette manière, un ensemble d'attributs perceptuels comme les ascendantes, les descendantes et les boucles dans une des trois zones du mot sont extraits. Les attributs choisis contiennent des informations sur le type d'attribut, sa taille et sa position.

Conclusion

4.6 Conclusion

L'extraction de caractéristiques est une étape cruciale dont l'objectif est de représenter chaque caractère par un vecteur d'attributs qui devient son identité. Les attributs caractéristiques doivent être indépendants aux transformations telles que la translation, la rotation et le facteur d'échelle. Une seule méthode d'extraction de caractéristiques n'est pas suffisante. La combinaison de plusieurs extracteurs caractéristique permet la meilleure distinction de caractères et surtout en utilisant de multiples classificateurs. Si un grand ensemble de données d'apprentissage est disponible, l'analyse par composant principal (PCA) permet de choisir les attributs les plus significatifs.

Les techniques d'extraction de caractéristiques varient selon le type du classificateur utilisé. Elles peuvent être classifiées en 3 types :

- *statistique* : basée sur une distribution statistique pour représenter un caractère tels que le zonage, les profils de projection, les croisements et les distances.
- *structurel* : basés sur des propriétés topologiques et géométriques du caractère telles que les intersections, les Pixels de fin de ligne,… etc.
- *Transformations globales et moments* : comporte plusieurs types d'attributs comme : les moments de Zenrike, les wavelets, les descripteurs de Fourier, la transformée de Hough, la transformée de Gabor, Walsh Hadamard….

Le Zonage et l**es histogrammes de projection** sont souvent employés en combinaison avec d'autres extracteurs caractéristiques pour aboutir à des meilleurs résultats.

Les intersections et les Pixels de fin de ligne exigent la squelettisation et se sont variables aux rotations et translations

Les descripteurs de Fourier permettent la reconstruction des caractères, mais ne s'appliquent pas aux caractères fragmentés. Les moments de Hu, les moments de Zernike et la transformée de Karhunen Loeve peuvent être des solutions alternatives, puisqu'ils ne sont pas sensibles aux fragmentations. Les moments de Zernike constituent une solution prometteuse pour la reconnaissance de caractères puisqu'ils possèdent la capacité de reconstruction des caractères.

Les attributs de Gabor sont invariables à la translation, la rotation, et le facteur d'échelle

La transformée de Karhunen-Loève semble être meilleure pour les classificateurs employant la somme des erreurs quadratiques, tels que, le K- plus proche voisin et K-means.

Chapitre 5
LA CLASSIFICATION DE CARACTERES

5.1 Introduction

Après la segmentation de caractères et l'extraction des attributs caractéristiques, une étape de reconnaissance basée classification de caractères est employée. La classification est un algorithme qui consiste à prédire la classe la plus proche, la plus probable ou la plus vraisemblable d'une certaine donnée non apprise par le classificateur. Dans le cas du problème de la reconnaissance de caractères, l'entrée de l'algorithme est l'ensemble des attributs caractéristiques d'un caractère, la sortie est la classe de ce caractère ou la référence d'une classe pré connue. Si l'entrée est constituée de plusieurs données, la sortie de l'algorithme est la liste des classes correspondantes. Les techniques de la classification reposent sur une stratégie de décision qui permet de catégoriser un objet le mieux possible selon certains critères d'optimisation. En fait, les algorithmes de classification peuvent être distingués en deux catégories : supervisés où la sortie correcte doit être fournie à l'avance et non supervisés où la sortie correcte n'est pas exigée à l'avance, elle résulte après une étape d'apprentissage ; ce type d'algorithme est généralement employé pour le clustering. La littérature se réfère généralement à quatre types d'approches de la classification : techniques d'appariement de modèles (template matching), les réseaux de neurones, les techniques statistiques et les techniques structurelles. Ces approches ne sont pas nécessairement indépendantes [M.E. Morita, 2003]. Dans ce chapitre, nous allons présenter quelques méthodes de la classification les plus célèbres et les plus efficaces dans le domaine de la reconnaissance de l'écriture manuscrite.

5.2 Les techniques de classification

La reconnaissance de formes est le domaine d'affectation des classes les plus appropriées aux objets inconnus. Ces objets sont décrits par ensemble de mesures appelées les attributs caractéristiques. Les techniques de classification comme mentionné précédemment peuvent être distinguées en quatre catégories : les techniques d'appariement de modèles basées sur l'estimation d'un degré de similitude entre deux vecteurs (groupes de Pixel, formes ou courbes) dans l'espace des attributs. Ces techniques peuvent être classifiées en trois approches : l'appariement direct [V. Lepetit et al, 2004], appariement élastique et déformable et appariement par relaxation [S. Uchida et al, 2003]. Les techniques statistiques [N. Sharma et al, 2006] et structurelles [U. Bhattacharya et al, 2005] sont basées sur des fonctions de décision statistiques qui déterminent la probabilité d'un modèle observé (modèle statistique, structurel ou autres) pour être un élément d'une classe prédéfinie. Plusieurs approches bien connues font partie de cette catégorie telles que : le classificateur du K-plus-Proche-Voisin (K-NN), le classificateur bayésien, le modèle de Markov caché (HMM) et les machines à vecteur support (SVMs) [M.E. Morita, 2003], [L.S. Oliveira et al, 2004]. Les réseaux de neurones artificiels (ANNs) [M.Fu. Zhang et al, 1998] sont des modèles informatiques inspirés du fonctionnement des neurones d'un cerveau humain. Les neurones sont des fonctions mathématiques groupées dans un réseau utilisé pour l'apprentissage et la classification des données d'entrée. Les réseaux de neurones sont très utilisés dans le domaine de la reconnaissance, et plus particulièrement la reconnaissance de l'écriture manuscrite, dus à leur capacité de généralisation et le temps de réponse très réduit en comparaison à d'autres méthodes.

Généralement, pour évaluer la performance d'un classificateur, on poursuit deux étapes principales qui sont l'apprentissage et le test. Les données d'apprentissage sont les données utilisées pour construire un classificateur capable de reconnaître des formes inconnues, les données de test sont les données inconnues employées pour examiner la performance d'un classificateur. Si toutes les données sont employées pour l'apprentissage et le test en même temps, il est possible que le classificateur soit incapable de reconnaître d'autres formes inconnues. C'est pourquoi il est important d'avoir trois ensembles de données pour améliorer la généralisation d'un classificateur : le premier ensemble est spécifié pour l'apprentissage, le 2ème pour la validation, et le dernier pour le test. Les données de validation sont utilisées pour continuer le processus d'apprentissage jusqu'à ce que les performances sur l'ensemble de validation ne s'améliorent plus. En ce moment, l'apprentissage devrait être arrêté afin d'éviter le sur-apprentissage (overtraining) du classificateur où sa capacité de généralisation commence à diminuer.

5.3 Le classificateur du K plus proches voisins

C'est un classificateur simple basé sur le calcul de distance entre les exemples d'apprentissage et les exemples de tests, généralement la norme euclidienne est souvent employée comme mesure de distance, dans chaque étape de l'apprentissage, l'algorithme mémorise les k meilleurs exemples de l'ensemble d'apprentissage (kppv(x)) qui sont proches à l'exemple de test x. Cet algorithme est souvent performant s'il y a suffisamment d'exemples d'apprentissage, mais demande un temps de prédiction très long pour passer tous les exemples afin de trouver les K meilleures solutions.

L'algorithme [X. Wu et al, 2008]**,** [B.V.Dasarathy, 1991]

Soit L = {(x', c)|$x' \in R^d$, $c \in C$} est un ensemble d'apprentissage.
(x' est un exemple d'apprentissage et c sa classe qui lui est associée.
Soit x l'exemple de test dont on souhaite déterminer sa classe
Début
Pour chaque *(exemple* (x', c) $\in$ L*)* **faire**
Calculer la distance $D(x, x')$
Fin
Pour chaque {$x' \in kppv(x)$} **faire**
Compter le nombre d'occurrences de chaque classe
Fin
Attribuer à x la classe la plus fréquente
fin

5.4 Le classificateur Bayesien

Le classificateur de Bayes est un algorithme d'apprentissage probabiliste basé sur la théorie de Bayes qui donne des résultats performants même s'il y a peu de données [D.Weissenbacher et al, 2007].
Le théorème de Bayes indique comment prévoir la meilleure classe pour un exemple non résolu « e » de la base des exemples d'apprentissage : On choisit pour « e » la classe C_i qui maximise la probabilité suivante [D.Weissenbacher et al, 2007]:

$$P(C_i/e) = \frac{P(e/C_i) * P(C_i)}{P(e)} \qquad (5.1)$$

où $C_i \in$ {ensemble des classes possibles}
$P(e|C_i)$ est estimée à partir *du corpus d'apprentissage.*

5.4.1 Le classificateur Bayesien naïf

Si les attributs " v_i " de l'exemple « e » sont indépendant l'un de l'autre, le classificateur est appelé "classificateur de Bayes naïf " et la probabilité P(e | C_i) est exprimée par le produit $P(v_1|C_i) * ... * P(v_a|C_i)$.

On choisit pour « e » la classe C_i qui maximise la probabilité reformulée comme suit [D.Weissenbacher et al, 2007] :

$$P(C_i/e) = \frac{P(C_i)}{P(e)} \prod_{j=1}^{a} P(v_j/C_i) \qquad (5.2)$$

Étant donné un ensemble d'apprentissage X = (x_1, x_2, · · · , x_n) où x_i = (v_{i1}, v_{i2}, · · · , v_{ip}). Pour classifier un nouvel élément x = (v_1, · · ·, v_p) : Le classificateur de Bayes estime les probabilités P(c) et P (x|c) à partir le corpus d'apprentissage X et puis emploie une règle de décision connue

sous : *maximum a posteriori* (*MAP)* qui sélectionne l'hypothèse la plus probable . La classe correspondante est la fonction C_{MAP} définie comme suit [I. Pop, 2006], [H.Zhang, 2004]:

$$C_{MAP} = argmax_C\, P\left(\frac{C}{x}\right) = argmax_C\, \frac{P\left(\frac{x}{C}\right) * P(C)}{P(x)} = argmax_C\, P\left(\frac{x}{C}\right) P(C) \quad (5.3)$$

5.5 Les Arbres de décision

Les Arbres de décision sont des règles de classification basées sur des tests associés aux attributs organisés de manière arborescente. C'est une méthode d'apprentissage supervisé très répandu et rapide dont le principe est le suivant :

5.5.1 Principe

Un arbre de décision est constitué de nœuds reliés par des branches descendantes du nœud racine jusqu'aux nœuds feuilles. Chaque nœud est étiqueté par un attribut, les branches sont étiquetées par les valeurs correspondant à ces attributs et les feuilles par des classes. Chaque nœud interne ou feuille est repéré (les numéros des arcs qui permettent d'y accéder) par sa position depuis la racine. Une instance est testée par son chemin partant de la racine jusqu'à la feuille. Chaque chemin est une règle de décision. Les conditions d'arrêt sont : le pourcentage des cas appartenant à la classe majoritaire > à un seuil, ou le nombre des cas dans une feuille < à un seuil, ou la combinaison des 2, ... [I.S.I. Abuhaiba, 2005], [J.R.Quinlan, 1986].

Un arbre de décision se compose de tests ou nœuds d'attribut liés à deux ou plusieurs sous-arbres et des feuilles ou nœuds de décision marqués avec des classes qui signifient des décisions. Un nœud de test calcule des conclusions basées sur les valeurs d'attribut d'une instance, où chaque conclusion possible est associée à un des sous-arbres [V. Podgorelec et al, 2002], [J.R.Quinlan, 1993].

Une instance est classifiée en commençant par le nœud racine de l'arbre, les conclusions pour l'instance sont ensuite déterminées et le processus continue d'employer le sous-arbre approprié [V. Podgorelec et al, 2002]. Autrement dit, pour attribuer un cas non résolu, on commence par le nœud racine de l'arbre et puis on se déplace le long des nœuds d'attribut en choisissant les branches ayant les valeurs d'attributs les plus appropriées dans l'arbre de décision jusqu'à ce qu'un nœud feuille soit atteint, ce dernier représente la classe du cas traité [Y. Yohannes et al, 1999].

5.5.2 La construction d'un arbre de décision

La première étape consiste à construire un «petit arbre» consistant avec la plupart des données et puis de diviser récursivement l'échantillon d'apprentissage par des tests définis à l'aide des attributs jusqu'à obtenir des sous-groupes homogènes ne contenant (presque) que des exemples appartenant à une même classe. Pour choisir la variable de coupure en sous- groupes qui peuvent être purs, l'algorithme teste toutes les variables des attributs existants et puis choisit celles qui maximisent un critère donné (ou un gain en pureté). Il existe un grand nombre de critères informationnels, tels que, l'entropie et le coefficient de Gini [Y. Yohannes et al, 1999].

Ci-dessous est la méthode pour construire un arbre de décision expliqué par Quinlan [J.R.Quinlan, 1993], [G. Stempfel, 2008] :

Algorithme de construction d'un arbre de décision

Procédure : construire-arbre(X)
Si tous les points de X appartiennent à la même classe alors
    Créer une feuille portant le nom de cette classe
Sinon
    Choisir le meilleur attribut pour créer un nœud
    Le test associé à ce nœud sépare X en deux parties : X_g et X_d
    construire-arbre (X_g)
    construire-arbre (X_d)
Finsi
Répéter cette procédure pour chacun des sous-arbres

5.5.3 Algorithmes d'apprentissage d'un arbre de décision

L'inférence inductive est le processus de passer des exemples concrets aux modèles généraux. Le but est d'apprendre comment classifier des objets en analysant un ensemble d'instances dont leurs classes sont déjà connues. Les entrées d'apprentissage se composent d'un ensemble de vecteurs attribut-valeur, chacun appartient à une classe connue, et la sortie est un assortissement des valeurs d'attribut aux classes. Cet assortissement devrait exactement classifier les exemples donnés et les exemples inconnus [V. Podgorelec et al, 2002].

L'inférence inductive est l'une des méthodes les plus utilisées pour l'apprentissage d'un arbre de décision. La plupart des algorithmes d'apprentissage qui ont été développés sont des variations d'un algorithme de recherche générique de haut en bas dans l'espace des arbres des décisions possibles [I. S. I. Abuhaiba, 2005], [J. R. Quinlan, 1986]. L'algorithme d'apprentissage est un algorithme de classification supervisé qui détermine un modèle de classification. Ce modèle est un arbre de décision qui servira à classer de nouveaux exemples.

L'idée générale d'un algorithme d'apprentissage est la suivante :
Un nœud est terminal lorsque (presque) tous les exemples correspondant à ce nœud sont dans la même classe, ou encore, s'il n'y a plus d'attributs non utilisés dans la branche correspondante. On attribue à un nœud terminal la classe majoritaire. Lorsque plusieurs classes sont en concurrence, on choisit la classe la plus représentée dans l'ensemble d'apprentissage. On sélectionne le test qui a progressé de plus la classification des données d'apprentissage c-à-d. qui maximise le gain en information. Les principaux algorithmes d'apprentissage par arbres de décision sont : ID3, CART et C4.5.
CART utilise l'indice de Gini et l'algorithme C4.5 utilise la notion d'entropie pour mesurer le gain en information lors de la progression de l'algorithme d'apprentissage [G. Stempfel, 2008] , [J. R. Quinlan, 1986].

Algorithme d'apprentissage générique [V. Podgorelec et al, 2002] **:**

Soient k classes dénotées {C_1, C_2…, C_k}, et un ensemble d'apprentissage " S "
• Si " S " contient un ou plusieurs objets qui appartiennent tous à la même classe Cj, alors l'arbre de décision est une feuille identifiant la classe Cj.

• Si " S " ne contient aucun objet, l'arbre de décision est une feuille déterminée à partir d'une information autre que " S ".

• Si " S " contient des objets qui appartiennent à plusieurs classes, alors un test est choisi, basé sur un attribut simple, qui a un ou plusieurs conclusions (sorties) mutuellement exclusif {O_1, O_2…, O_n}. " S " est divisé en sous-ensembles S_1, S_2…, S_n, où S_i contient tous les objets de " S " qui ont les conclusions O_i du test choisi.
La même méthode est appliquée récursivement à chaque sous-ensemble d'objets d'apprentissage.

L'aspect le plus important de la stratégie d'induction des arbres de décision est la manière dont un ensemble est subdivisé, c-à-d. comment choisir un test d'attribut qui détermine la meilleure subdivision en sous-arbres. L'algorithme d'induction ID3 emploie la théorie de l'information de Shannon pour évaluer la subdivision. L'algorithme C4.5 est une amélioration d'ID3 du point de vue de la simplicité.

Exemple : l'algorithme ID3 choisit l'attribut de coupure qui conduit à une partition en sous-arbres dont le Gain d'information ($Gain_f$) correspondant est maximum. C'est-à-dire l'attribut qui permettra de classer les exemples à un certain niveau (branche) de l'arbre. On appelle ce calcul l'entropie de Shannon dont voici la formule utilisée [G. Stempfel, 2008], [J. R. Quinlan, 1986]:
Soit S un ensemble et S_1,…, S_c la partition de S en C classes cibles

Soit *p* une position, l'entropie de p est définie par :

$$Entropie(p) = -\sum_{k=1}^{C} P(k/p) \times log\big(P(k/p)\big) \quad (5.4)$$

$P(k/p)$ est la probabilité *à priori* de la classe k à la position *p*.

Le gain en information :
Le gain est maximal lorsque le choix d'un attribut permet de classer correctement toutes les données ; il est nul lorsque les données sont mal classées après le test. Soit p la position courante de l'arbre en construction et T un test, si les attributs descriptifs sont N-aires alors le gain en information $Gain_f$ est définit comme suit [G. Stempfel, 2008] :

$$Gain_f(p,T) = f\big(S_p\big) - \sum_{j=1}^{N} P_j * f\left(S_{P_j}\right) \quad (5.5)$$

Où :

f = Entropie

- S_p est l'échantillon associée à p,
- S_{Pi} est l'ensemble des éléments de S_p qui satisfont la i -ème branche de T,
- P_j est la proportion des éléments de S_p satisfont la $j^{ème}$ branche de T.

La représentation de la connaissance par arbres de décision est facilement comprise en comparaison avec les réseaux de neurones ; les arbres de décision apprennent plus vite que les réseaux de neurones qui sont relativement lents ; les arbres de décision ont des difficultés en traitant les données bruitées ce qui n'est pas le cas pour les réseaux de neurones … etc.
Par conséquent, différentes approches hybrides qui combinent les avantages des deux méthodes ont été proposées [M.Zorman et al, 1999].

5.6 Les Modèles de Markov Cachés (MMC)

Les modèles de Markov sont des modèles mathématiques de processus stochastiques qui produisent des séquences de résultats aléatoires selon certaines probabilités. Un modèle de Markov caché est un model dans lequel on observe une séquence des émissions sans savoir la séquence des états qui a produit ces émissions [L.R. Rabiner, 1989], [B. H. Juang et al, 1991].

5.6.1 Les éléments d'un MMC

Un MMC est défini par un quintuplet Λ={Σ,Ω,Π,A,B} où [L.R. Rabiner, 1989]:
L'alphabet Σ = {s_1,...,s_m} décrit les états de la chaîne de Markov
L'alphabet Ω = {o_1,..., ok} sont des symboles émis par les s_i pour un MMC discret
Les probabilités initiales des états Π= {π_i=P (s_i)}
La matrice des probabilités de transitions entre les états A = {a_{ij}= P ($s_j|s_i$)}
Les probabilités d'émission B = {b_i (o_k)=P ($o_k|s_i$)}

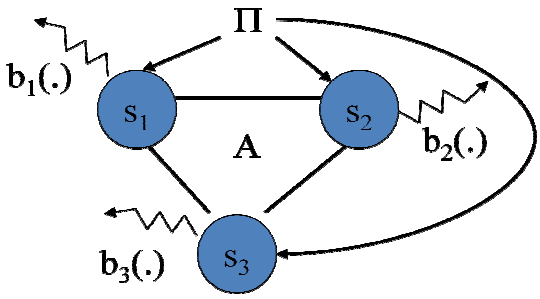


Figure 5.1: Une chaîne de Markov à 3 états avec les processus associés [B. H. Juang et al, 1991]

Les MMCs permettent de répondre aux trois questions suivantes [B. H. Juang et al, 1991], [L. R. Rabiner, 1989] :

- On peut calculer la probabilité d'une séquence d'observation à l'aide de l'algorithme Forward-Backward
- Partant d'un ensemble d'observation *O*, on peut trouver la séquence la plus probable des états qui ont conduit à la génération de cette séquence à l'aide de l'algorithme Viterbi

- Partant d'un ensemble d'observation *O*, on peut régler les paramètres d'un model MMC afin de maximiser la vraisemblance de $P(O|\Lambda)$ (c-à-d l'apprentissage d'un MMC) en utilisant l'algorithme de Baum-Welch ou l'algorithme de Viterbi.

5.6.2 Apprentissage d'un MMC

Les méthodes d'apprentissage d'un MMC les plus utilisées sont : l'algorithme de Viterbi, et l'algorithme Baum-Welch [L. R. Rabiner, 1989].

5.6.2.1 L'algorithme de Viterbi

Idée générale :
L'algorithme de Viterbi est un algorithme de la programmation dynamique basé treillis qui exécute un MMC pour trouver la séquence la plus susceptible des états cachés, appelé le chemin de Viterbi [A. Churbanov et al, 2008]. On détermine, pour chaque sommet du treillis, le meilleur chemin (le chemin le plus probable) [L. R. Rabiner, 1989] menant à une séquence particulière.

Une présentation détaillée des MMCs peut être trouvée dans [L.R. Rabiner, 1989]

L'algorithme [L.R. Rabiner, 1989]**:**

1. Initialisation :
Initialisation des probabilités d'état qui sont le produit des probabilités d'états cachées initiales avec les probabilités d'observation associées.
Pour t=1 et $1 \leq i \leq$ nombre d'états m,

$$\alpha_1(i) = \pi_i * b_i(O_1) \qquad (5.6)$$

$$\psi_1(i) = 0 \qquad (5.7)$$

2. Récurrence :
Détermination du chemin le plus probable au prochain état en enregistrant, comment y arriver. Ceci est réalisé à travers le calcul des produits des probabilités de transition avec les probabilités maximales dérivées dans l'étape précédente.

Pour t = 2,…,n, et $1 \leq i \leq m$,

$$\alpha_{t+1}(i) = max_{j=1\ldots m}\left[\alpha_t(j) * a_{ji}\right] * b_i(O_{t+1}) \qquad (5.8)$$

$$\psi_t(i) = argmax_{j\,(j=1\ldots m)}\left(\alpha_{t-1}(j) * a_{ji}\right) \qquad (5.9)$$

3. Terminaison :
Ceci détermine l'état le plus probable du système (t=T).

$$s(n) = argmax_i\, \alpha_T(i) \qquad (5.10)$$

4. Retour en arrière :

Retour en arrière par le treillis en suivant le chemin le plus probable.
La séquence ***s (1) … s (T)*** est la séquence la plus probable des états cachés pour la séquence des observations traitée.
Pour *t* = n-1,…,1,

$$s(t) = \psi_{t+1}\big(s(t+1)\big) \qquad (5.11)$$

5.7 Les Machines à Vecteur Support (SVMs)

Les SVMs sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de la classification binaire et de la régression. Les SVMs reposent sur deux idées principales : la notion de *la marge maximale* et la notion de *la fonction noyau*. *La marge maximale* est employée pour les problèmes de la classification linéaire. Elle représente la distance entre la frontière de séparation et les échantillons d'apprentissages les plus proches. Ces derniers sont les *vecteurs supports*. *Les fonctions noyau* sont employées dans le cas des problèmes de la classification non-linéaire pour transformer l'espace de représentation des données d'entrées en un espace de plus grande dimension dans lequel il est probable qu'il existe de séparateurs linéaires [A.Borji et al, 2007].

Le classificateur SVM est un algorithme qui maximise la marge entre les classes du problème à résoudre et réduit au minimum l'erreur de classification. L'objectif de la marge maximale est de faire séparer deux classes par un hyperplan de telle sorte la distance par rapport aux vecteurs supports soit maximale [S. B. Kotsiantis, 2007].

Dans la tâche de classification, un SVM construit l'hyperplan optimal de séparation des attributs caractéristiques dans un espace de haute dimension. Le calcul de cet hyperplan est fondé sur la maximisation de la marge entre les exemples d'apprentissages les plus proches qui appartiennent à différentes classes.

5.7.1 Le cas linéairement séparable

L'hyperplan optimal de séparation est un classificateur de marge dont la sortie est donnée par [F. Lauer et al, 2007], [S. B. Kotsiantis, 2007]:

$$f(x) = sign\,(x^T w + b) \qquad (5.13)$$

Avec x un modèle d'entrée, le vecteur des poids W et le biais b sont ajustés par apprentissage en maximisant la marge 1 / ‖ w‖ sous la contrainte de la bonne classification des N modèles d'apprentissage à l'extérieur de la marge [F. Lauer et al, 2007]:

$$\min \frac{1}{2}\,\|w\|^2$$

Sous la contrainte $\quad y_i(x_i^T w + b) \geq 1, \quad i = 1, \ldots\ldots, N \qquad (5.14)$

Avec $y_i \in \{-1,1\}$ représente l'étiquette d'un modèle d'apprentissage x_i. (y_i =1 pour la classe 1, et -1 pour la classe 2).

La solution de ce problème correspond au point extrême du lagrangien primal [F. Lauer et al, 2007]:

$$L_P = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(x_i^T w + b) - 1\right] \qquad (5.15)$$

où αi sont les multiplicateurs de Lagrange.
Ce problème mène à la maximisation du lagrangien dual par rapport aux α_i [F. Lauer et al, 2007]:

$$\max L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j}^{N} \alpha_i \alpha_j \, y_i y_j \left(x_i^T x_j\right) \qquad (5.16)$$

Sous les contraintes : $\alpha_i \geq 0, \;\; i = 1, \ldots N \;\; et \;\; \sum_{i=1}^{N} \alpha_i y_i = 0$

La règle de séparation est donc [F. Lauer et al, 2007], [A.Borji et al, 2007]:

$$f(x) = sign\left(\sum\nolimits_{vecteurs\ supports} y_i \alpha_i (x_i^T\, x) + b\right) \qquad (5.17)$$

Où " x_i " sont les vecteurs supports (SVs) lorsque les multiplicateurs de Lagrange (α_i) sont différent de zéro. Les SVs sont les modèles d'apprentissage qui se trouvent sur les frontières de la marge [F. Lauer et al, 2007]. αi et b sont déterminés en résolvant un problème quadratique [G. G. Rajput et al, 2009].

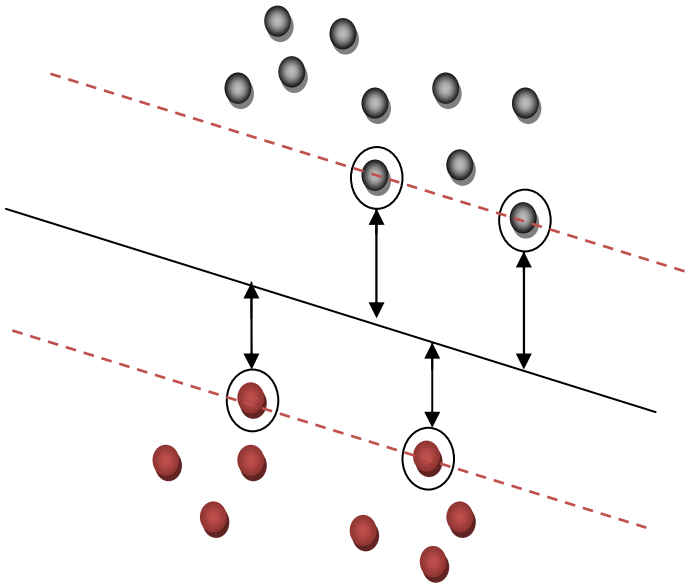


Figure 5.2: Discrimination linéaire

5.7.2 Le cas non-linéairement séparable

Dans le cas d'un espace non-linéaire. Chaque point " x " de l'espace d'entrée est transformé à un point z = Φ (x) dans un espace de plus grande dimension, appelé espace des attributs. La propriété principale de la fonction de transformation Φ (.) est que le produit scalaire de deux

points dans l'espace d'attributs Φ (x). Φ (x) peut-être réécrit par une fonction noyau : K(x, y) [A.Borji et al, 2007].
Ainsi, le classificateur SVM linéaire peut être étendu au cas non-linéaire en remplaçant le produit scalaire entre les vecteurs d'entrée "x" et les vecteurs supports "xi" par une fonction noyau K(xi ,x) [G. G. Rajput et al, 2010b].
La règle de séparation devient donc sous la forme suivante [A.Borji et al, 2007]:

$$f(x) = sign\left(\sum_{i=1}^{N} \alpha_i y_i \, K(x_i, x) + b\right) \qquad (5.18)$$

Les coefficients αi et b sont les solutions d'un problème de programmation quadratique [G. G. Rajput et al, 2009].

Les fonctions noyau les plus utilisées sont [M.Antkowiak, 2006]:

Polynomial:

$$K(x, x') = \langle x, x' \rangle^d \qquad (5.19)$$

Gaussienne:

$$K(x, x') = exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right) \qquad (5.20)$$

Sigmoïde :

$$K(x, x') = tanh(k\langle x, x' \rangle + \vartheta) \qquad (5.21)$$

Où k > et $\vartheta < 0$ sont des paramètres

5.7.3 Classification Multi-classes

Bien que la méthode SVM soit adaptée aux problèmes de deux classes, elle peut être facilement transformée en un outil très efficace pour la classification N-classes. Il existe deux méthodes pour résoudre un problème multi-classe avec des classificateurs SVMs [F.Lauer et al, 2007] :

1. **Stratégie un contre tout :** Dans cette stratégie, un classificateur est construit pour chaque classe et chargé de la séparation de cette classe des autres classes restantes. L'idée consiste tout simplement à transformer le problème du k classes en k classificateurs binaires. La classification finale est donnée par le classificateur qui répond le mieux.
2. **Stratégie un contre un :** Dans cette stratégie, un classificateur est construit pour chaque paire des classes pour séparer les classes deux à deux. Le problème est remplacé par N*(N−1)/2 tâches de classification entre deux classes. La classification finale est donnée par le vote majoritaire ou un graphe acyclique de décision.

Une autre stratégie,pour la reconnaissance de n chiffres, est d'employer n-classe SVM au lieu des classificateurs binaires avec la stratégie un contre tout résolvant de ce fait un problème d'optimisation contraint [F. Lauer et al, 2007]. Mais cette méthode n'est pas très populaire dans les applications de la reconnaissance de chiffres et n'apporte pas de meilleures performances que

les autres classificateurs [E. J. Bredensteiner et al, 1999]. Une description détaillée des SVMs peut être trouvée dans [C. Burges, 1998].

5.8 Les réseaux de neurones

Un réseau de neurones est un algorithme d'apprentissage basé sur des concepts inspirés du fonctionnement d'un cerveau humain. Un réseau de neurones peut être considéré comme un graphe dirigé reliant un certain nombre de neurones par des liens pondérés, le fonctionnement des neurones est simulé par des fonctions d'activation.

Les réseaux de neurones peuvent être classés en trois catégories basées sur l'architecture adoptée [R.P.W. Duin, 2000] : les réseaux feed forward (FFs) ou Perceptrons multicouche (MLP), les réseaux de foncions à base radiale (RBFs) et les cartes auto-organisatrices (SOMs).
Dans les réseaux MLP, une combinaison linéaire des entrées est calculée en utilisant une fonction qui renvoie le produit scalaire entre les entrées et les poids synaptiques correspondants par contre les RBFs emploient des fonctions de combinaison qui renvoient les distances euclidiennes entre les entrées et les centres de la couche cachée. Les cartes auto-organisatrices (SOMs) possèdent une structure différente où des nœuds (neurones) sont régulièrement placés dans une grille hexagonale ou rectangulaire. Les SOMs sont basés sur le principe de transformer les données d'entrée de grande dimension à des données de dimensions inférieures. La donnée de la carte est le nœud ayant le vecteur du poids le plus proche au vecteur d'entrée que nous assignons les coordonnées de ce nœud [J. Vesanto et al, 2000].
L'apprentissage d'un réseau de neurones implique l'ajustement de ses poids basé sur un processus itératif qui optimise une fonction d'erreur particulière.

Les méthodes de la descente de gradient sont des méthodes d'apprentissage supervisées qui ont prêté une grande attention due à leurs résultats raisonnables dans un temps minimal. Cependant, elles peuvent converger vers des solutions locales dans la plupart des cas et plus particulièrement pour les tâches difficiles, telles que, la reconnaissance de caractères manuscrits.

Les réseaux de neurones, avec leur capacité de généralisation, peuvent reconnaître et détecter des formes imprévues qui sont difficiles à détecter par d'autres méthodes de classification. Un réseau de neurones entraîné peut être considéré comme un "expert" capable de fournir des réponses à des nouvelles circonstances.

5.9 Réseaux de neurones vs SVMs

Les paramètres des classificateurs neuronaux sont généralement ajustés par descente de gradient. Leur temps d'apprentissage est ainsi linéaire avec le nombre des exemples d'apprentissage. Alors que l'apprentissage des SVMs se fait par programmation quadratique, leur temps est généralement proportionnel au carré du nombre des exemples d'apprentissage. La performance de généralisation des classificateurs neuronaux est sensible à la taille et structure du réseau, en outre, la sélection de leur structure la plus appropriée est difficile à déterminer. Aussi, la convergence d'un réseau de neurones souffre des minimax locaux. Alors que, l'apprentissage des SVMs par programmation dynamique garantit de trouver l'optimum global. La performance des SVMs dépend du type de noyau et de leurs paramètres, mais cette dépendance est moins influente. Les SVMs ne peuvent pas être utilisés pour un apprentissage en modes holistiques tel que le cas de la reconnaissance des mots. Les SVMs donnent souvent des résultats supérieurs par rapport aux réseaux de neurones en termes du taux de reconnaissance [C.L.Liu et al, 2008].

Conclusion

5.10 Conclusion

La classification est l'affectation des classes les plus appropriées aux objets inconnus. Les méthodes de classification peuvent être distinguées en quatre types : les techniques d'appariement de modèles basées sur l'estimation d'un degré de similitude entre deux vecteurs, les réseaux de neurones, les techniques statistiques et les techniques structurelles. Ces méthodes peuvent compléter les unes des autres.

Les techniques statistiques basées sur des fonctions de décision statistiques, telles que, le classificateur du K-plus-Proche-Voisin (K - NN), le classificateur bayésien, le modèle de Markov caché (HMM) et les machines à vecteur support (SVMs) présentent l'avantage de la simplicité de modélisation de tous types des objets d'entrées aussi bien que leur efficacité de classification.

Le K plus proches voisins : est souvent performant quand il y a suffisamment d'exemples d'apprentissage, mais demande beaucoup du temps pour passer tous les exemples afin de trouver la meilleure solution.

Le classificateur de Bayes est un algorithme d'apprentissage probabiliste basé sur la théorie de Bayes qui donne des résultats performants même s'il y a peu d'exemples d'apprentissage.

Les arbres de décision : le comportement d'un arbre de décision est facilement interprété en comparaison avec les réseaux de neurones, en outre, ils se sont plus rapides que les réseaux de neurones ; mais présentent des difficultés de généralisation.

Les MMCs permettent de modéliser les variations de l'écriture manuscrite et ainsi permettent de résoudre la segmentation et la reconnaissance en même temps ; il est possible de reconnaître les mots sans leur pré segmentation.

Les SVMs sont efficaces pour les problèmes non linéaires à travers la notion de fonction noyau, ils permettent aussi un apprentissage rapide avec un nombre limité des exemples d'apprentissage et évitent le surapprentissage souvent provoqué par les réseaux de neurones.

Les réseaux de neurones possèdent la capacité de généralisation à partir des exemples simples d'apprentissage, les réseaux de neurones exigent un temps énorme pour l'apprentissage, mais une fois terminé le test est rapide.

Chapitre 6
LA COMBINAISON DE CLASSIFICATEURS

6.1 Introduction

L'objectif de la construction d'un système de la reconnaissance de formes est d'aboutir aux meilleures performances en termes d'exactitude. Cet objectif a mené au développement de plusieurs types de classificateurs qui peuvent produire des résultats dissimilaires. Partant de cette idée, La combinaison de plusieurs classificateurs ayant des informations complémentaires peut offrir des résultats supérieurs. Ainsi, les sorties de différents classificateurs peuvent être combinées pour fournir de meilleurs résultats [S. Arora et al, 2010]. La combinaison de classificateurs est un ensemble de méthodes basées sur la coopération de plusieurs classificateurs en combinant leurs décisions pour résoudre le même problème. Ces méthodes doivent produire des résultats plus fiables que ceux d'un classificateur individuel [W. Wang et al, 2002]. Dans la littérature, un grand nombre de travaux montre que la combinaison de classificateurs est plus performante qu'un classificateur individuel dit de base [K. Chen et al, 1997], [Rafael M. O et al, 2010]. La combinaison de classificateurs prend deux formes : la combinaison des classificateurs similaires entraînés par différents types des données d'entrées et la combinaison de classificateurs dissimilaires ayant des structures différentes [S. Arora et al, 2010]. La construction de ce type d'algorithmes est devenue un domaine de recherche très important [J. Kittler et al, 2000], [R.P.W. Duin, 2002].

Ce chapitre est prévu pour exposer certaines méthodes et algorithmes importants de la combinaison. La majorité de ces méthodes sont bien connues dans le domaine de l'apprentissage et la reconnaissance de formes [L.I. Kuncheva, 2004], telles que le vote majoritaire, le vote par majorité pondéré, le borda Count, BKS, les réseaux de neurones, combinaison des valeurs de confiance …. [T. Ho et al, 1994], [Y. Huang et al, 1995], [J. Kittler et al, 1998]. Les méthodes de combinaison présentées dans ce chapitre sont largement inspirées de [K. H. Zouari, 2004].

6.2 Combinaison de classificateurs

Un classificateur est un outil de reconnaissance qui reçoit un vecteur des attributs d'entrées et produit un vecteur des classes de ces entrées. La combinaison de plusieurs classificateurs permet le plus souvent d'améliorer la qualité de la classification en termes d'exactitude. Cette combinaison est d'autant plus importante lorsque les classificateurs sont complémentaires. Lorsque les classificateurs sont identiques, il n'y aura aucune amélioration de l'exactitude ; il faut, donc créer des experts différents.

Le problème de la combinaison de classificateurs peut être considéré comme problème d'optimisation d'une fonction (de combinaison) qui accepte de vecteurs *N*-dimensionnels représentant les classes issues de *M* classificateurs et produit *N* points d'une classification finale en réduisant par exemple au minimum le coût d'une classification incorrecte [S.Tulyakov et al, 2008].

6.3 Types de sorties d'un classificateur

On peut distinguer trois types de sorties d'un classificateur [I. Gasmi et al, 2005]**:**
Soit un classificateur (e) qui reçoit une entrée (x) et fournit une sortie (Ci) tel que (i=1..m), (m) est le nombre de classes.

• **Sortie de type classe**
C'est le type le plus simple qui apporte le moins d'informations. Le classificateur donne une étiquette (Ci) indiquant la classe de l'entrée traitée.
ej(x) = Ci (i ϵ {1..m}).

• **Sortie de type rang**
Le classificateur donne un ensemble de classes en précisant leurs faveurs par un rang.
ej(x) = [rj 1, rj 2, …, rj m] où rj i est le rang accordé à la classe (i) par le classificateur (j). Les rangs fournis par chaque classificateur peuvent être considérés comme des votes pondérés pour chaque classe.

• **Sortie de type mesure**
Cette sortie est la plus riche en information. Dans ce cas, pour chaque classe est associée une mesure qui indique sa préférence par une probabilité, une vraisemblance, une distance, un score ou une confiance, ….
ej(x)=[Mj 1, Mj 2,….,Mj m] où Mj i est la mesure fournie à la classe (i) par le classificateur (j).

6.4 Les mesures de performances d'un classificateur

Pour une certaine entrée, un classificateur peut produire l'une des réponses suivantes :
•**Un rejet :** si le classificateur n'a pas pu classifier l'entrée.
•**Une reconnaissance :** si le classificateur accorde la classe exacte à l'entrée traitée.
•**Une substitution :** si le classificateur associe une classe incorrecte à l'entrée.

$$taux\ de\ rejet = \frac{nombre\ de\ formes\ rejetées}{nombre\ total\ de\ formes}$$

$$taux\ de\ reconnaissance = \frac{nombre\ de\ formes\ reconnues}{nombre\ total\ de\ formes}$$

$$taux\ de\ substitution = \frac{nombre\ de\ formes\ males\ reconnues}{nombre\ total\ de\ formes}$$

6.5 Les méthodes de combinaison des classificateurs

Il existe trois architectures de la combinaison de classificateurs que sont [H. Zouari et al, 2002] :

- Séquentielle ou série
- Parallèle
- Hybride (mixte séquentielle et parallèle)

6.5.1 Combinaison Séquentielle (ou Série)

Dans cette architecture, la décision se fait successivement tout en réduisant de manière progressive le nombre de classes possibles. En d'autres termes, les classificateurs sont utilisés les uns à la suite des autres, chacun réduisant l'ensemble des classes possibles pour un exemple donné [H. Zouari et al, 2002].

6.5.2 Combinaison Parallèle

Les classificateurs sont entraînés parallèlement de manière indépendante puis les résultats sont fusionnés en utilisant des fonctions de fusion spécifiques. Les méthodes parallèles sont les plus répandues dues à leur simplicité mais aussi leur efficacité en termes d'exactitude [H. Zouari et al, 2002].

6.5.3 Combinaison Hybride

Cette classe est une mixture des deux approches précédentes : séquentielle et parallèle. Elle consiste à réduire de manière successive le nombre de classes possibles toute en permettant un apprentissage parallèle des classificateurs [H. Zouari et al, 2002].

6.6 Catégorisation des méthodes de combinaison parallèle

De nombreux travaux ont démontré que la combinaison des classificateurs (séquentiels, parallèles ou hybrides) améliore nettement les performances du système de reconnaissance par rapport à chacun des classificateurs pris isolément. Cependant, parmi les différentes architectures permettant la combinaison d'un ensemble de classificateurs, l'architecture parallèle est celle qui a donné lieu aux travaux les plus importants. Sa simplicité de mise en œuvre, sa capacité à exploiter les réponses des classificateurs à combiner en prenant compte (ou non) le comportement de chacun des classificateurs et de son efficacité prouvée dans de nombreux

problèmes de classification expliquent son succès, notamment, par rapport à l'approche séquentielle où la connaissance du comportement de chaque classificateur est nécessaire a priori pour pouvoir obtenir un schéma de coopération efficace. Ci-dessous, une présentation détaillée des méthodes de combinaison parallèle.

Les méthodes de combinaison parallèle peuvent être classées en deux familles : *Les méthodes de fusion* où tous les classificateurs participent à la prise de décision et *les méthodes de sélection* où l'ensemble des classificateurs à combiner est sélectionné en fonction des exemples d'apprentissage [L.I. Kuncheva, 2004]. Il existe aussi deux natures des classificateurs parallèles : les Classificateurs différents (hétérogènes) tels que le cas d'un MLP et un classificateur de Bayes ou les classificateurs homogènes ayant la même structure mais entraînés sur des données différentes ou initialisés de manière différente [R. Duin et al, 2000].
On peut aussi distinguer les méthodes de combinaison parallèle *avec apprentissage* et sans apprentissage. Dans le premier cas, les paramètres d'un classificateur sont optimisés en fonction des exemples d'apprentissage ; l'exemple le plus connu un MLP entraîné par les sorties des classificateurs à combiner, ces sorties sont présentés en tant que entrées de ce dernier. Dans le second cas, c-à-d les méthodes de combinaison parallèles *sans apprentissage* ou fixes qui sont basés sur des règles fixes comme par exemple calcul de la somme ou le produit des probabilités associées à chacune des classes.

La figure ci-dessous présente un panorama structuré sous forme de graphe des différentes méthodes de combinaison parallèle qui constituent les méthodes les plus répandues.

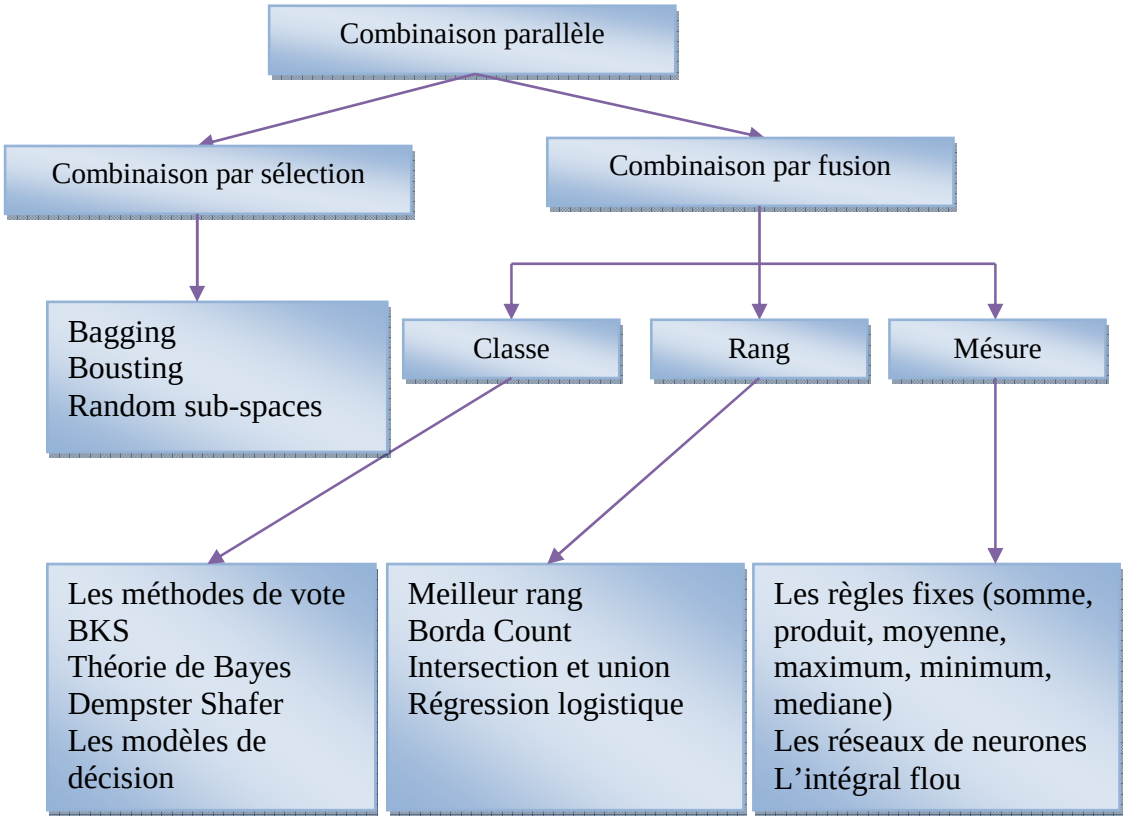


Figure 6.1: Classification des méthodes de combinaison parallèles [H. Zouari et al, 2002]

6.7 Fusion et sélection de classificateurs

Il existe deux stratégies principales de la combinaison de classificateurs : fusion et sélection. La fusion de classificateurs est basée sur un nombre fixe de classificateurs généralement moins que 10 par contre la sélection aussi appelée méthodes ensemblistes est basée sur un grand nombre de classificateurs (potentiellement infinis) à partir de lesquels on sélectionne ou on génère de nouveaux classificateurs [S.Tulyakov et al, 2008].
Il existe également des méthodes hybrides de ces deux stratégies. De telles méthodes prennent par exemple la moyenne des sorties avec les coefficients correspondant à une certaine entrée. Ainsi, les compétences locales d'un classificateur (pour une certaine entrée) sont mesurées par des poids, ensuite, plusieurs classificateurs sont responsables de la même entrée et les sorties de tous les classificateurs responsables sont fusionnées [L.I. Kuncheva, 2004].

6.7.1 Combinaison par fusion

Dans cette catégorie, les classificateurs opèrent sur différents types d'attributs caractéristiques ou différentes entrées. Par conséquent, elle possède l'avantage d'avoir une diversité des modèles d'entrée et chaque classificateur pourrait être un expert sur un certain type de modèles d'entrée [S.Tulyakov et al, 2008]. Dans cette classe, des combinateurs comme les méthodes de vote, le vote pondéré, les règles pondérées, la moyenne de Bayes, BKS et les méthodes de rang, telles que, la somme, le produit et la moyenne peuvent être utilisés.

6.7.1.1 Combinaison de Type Classe

Cette catégorie opère sur les étiquettes de classes des entrées traitées.

6.7.1.1.1 Méthodes de vote

Dans cette méthode, la sortie de chaque classificateur est considérée comme un vote (acceptation/rejet). La plupart des méthodes de vote utilisent un seuil qui représente la proportion des classificateurs devant voter la même classe pour qu'elle soit la décision finale. La classe qui a reçu un nombre de vote (acceptation) supérieur à ce seuil est considérée la décision finale sinon rejet. Par exemple, dans la méthode de vote le plus simple (*le vote à la pluralité*), la classe qui a reçu le plus grand nombre de vote est considérée la classe finale sinon rejet [L.I. Kuncheva, 2004].

Si on suppose que les sorties de L classificateurs sont des vecteurs c - dimensionnels d'étiquettes de classes (le nombre de classes =c) :
$[e_{i,1},\ldots\ldots,e_{i,c}]^T \in \{0,1\}^c$,
i=1,…, L où $e_{i,j}$ = 1 si le classificateur « ei » classifie l'entrée « x » dans sa classe C_i et 0 autrement.
$e_{i,j}$ est la j ^{ème} sortie du classificateur i pour l'entrée x (j=1,….,c).
Les méthodes de vote peuvent être dérivées de la règle,avec seuil, suivante [K. H. Zouari, 2004]:

$$e(x) = \begin{cases} C_i \;\; si \;\; \sum_{j=1}^{L} e_{i,j} = max_{t=1}^{M} \sum_{j=1}^{L} e_{t,j} \;\; \geq \lambda . L \\ sinon \quad rejet \end{cases} \qquad (6.1)$$

L est le nombre de classificateurs, *M* le nombre de classes et λ un seuil de décision.

Majorité unanime :
Si λ = 1, tous les classificateurs doivent voter la même classe pour qu'elle soit la décision finale sinon la décision finale est le rejet. Cette méthode est appelée *majorité unanime* présente l'inconvénient majeur de produire un taux de reconnaissance assez faible [L.I. Kuncheva, 2004], [K. H. Zouari, 2004].

Majorité simple :
Si λ ≥ 0,5, plus de la moitié des classificateurs doivent voter la même classe pour qu'elle soit la décision finale. Il s'agit de la méthode de vote par majorité. Dans un cas de conflit, des informations supplémentaires comme les performances individuelles des classificateurs sont utilisées [L.I. Kuncheva, 2004], [K. H. Zouari, 2004].

Vote à la pluralité :
Si λ =0, la classe qui à reçu le plus grand nombre de vote est considérée la classe finale sinon rejet. Cette règle s'appelle le vote à la pluralité [L.I. Kuncheva, 2004], [K. H. Zouari, 2004].

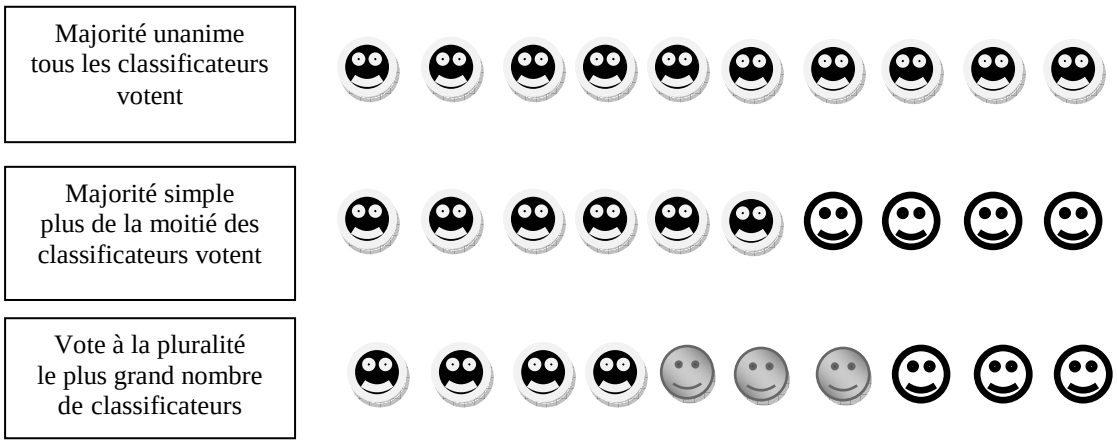


Figure 6.2 : Les méthodes de vote : unanime, majorité simple et vote à la pluralité, la décision finale est celle du groupe noir [L.I. Kuncheva, 2004]

***Le vote pondéré* :**
Cette technique associe des poids w_j aux classificateurs de base e_j. Ces poids varient au fur et à mesure que les classificateurs s'entraînent suivant leur performance de prédiction [K. H. Zouari, 2004].

$$e(x) = \begin{cases} C_i \;\; si \;\; \sum_{j=1}^{L} w_j e_{i,j} = max_{t=1}^{M} \sum_{j=1}^{L} w_j e_{t,j} \\ sinon \quad rejet \end{cases} \qquad (6.2)$$

Par exemple, dans la règle de combinaison, les classificateurs ayant les taux les plus élevés de la reconnaissance sont beaucoup plus importants que les autres classificateurs ayant une faible performance, même si un certain exemple est souvent bien classifié par ces derniers [N. Azizi et al, 2010].

6.7.1.1.2 Méthode du comportement d'espace des connaissances (Behavior Knowledge Space (BKS))

La plupart des méthodes de fusion supposent l'indépendance des décisions prises par différents classificateurs. La méthode (BKS) est une méthode avec apprentissage qui ne respecte pas cette condition et garde les traces de la décision finale en fournissant un espace de connaissance qui rassemble pour chaque exemple d'apprentissage les différents enregistrements de décisions de tous les classificateurs. Si un problème de fusion combine les sorties de K classificateurs : e1…, *eK produisant* chacun M *classes* : *c1…, cM*. La méthode (BKS) crée un espace K–dimensionnel où chaque dimension correspond à la décision d'un classificateur. Une unité de BKS est une intersection des décisions de chaque classificateur [L. Lam et al, 1994], [R.A. Dara, 2007]. Chaque unité BKS contient trois types d'informations : le nombre total d'exemple de l'unité $T_{e1..., eK}$, la meilleure classe représentative de l'unité $R_{e1..., eK}$ et le nombre de tous les exemples d'une classe $n_{e1..., eK}(m)$.

Dans la première étape de la méthode BKS, les données d'apprentissage sont intensivement exploitées pour construire le BKS [N. Azizi et al, 2010]. La décision finale « *e(x)* » pour un exemple d'entrée « *x* » est dérivée de l'unité focale en utilisant la règle décrite ci-dessous. Cette règle donne le résultat d'appartenance à une classe *c*, . c'est-à-dire, la classe dont la probabilité est maximale [C. Meurie et al, 2003].

$$e(x) = \begin{cases} R_{e(1)\ldots\ldots e(K)} \;\; si \;\; T_{e(1)\ldots\ldots e(K)} > 0 \; et \; \dfrac{n_{e(1)\ldots\ldots e(K)}\,(R_{e(1)\ldots\ldots e(K)})}{T_{e(1)\ldots\ldots e(K)}} > \lambda \\ sinon \qquad M+1 \end{cases} \qquad (6.3)$$

Où λ est un seuil qui contrôle la fiabilité de la décision finale ($0 < \lambda < 1$)

6.7.1.1.2 Les modèles de decision (Decision templates)

Les modèles de décision constituent une méthode de fusion dont le principe est comme suit :

Soit $\{D_1, \ldots\ldots, D_L\}$ un ensemble de L classificateurs utilisés pour la classification des exemples en C classes. Les sorties de ces classificateurs pour l'entrée "x" est une matrice L*C appelée le profile de décision.

La structure de cette matrice est comme suit [R. Ebrahimpour et al, 2009] :

$$DP(x) = \begin{pmatrix} D11\,(x) & \ldots. & D1j\,(x) & \ldots. & D1C(x) \\ \ldots.. & \ldots. & \ldots & \ldots. & \ldots \\ Di1\,(x) & \ldots. & Dij\,(x) & \ldots. & DiC\,(x) \\ \ldots. & \ldots. & \ldots. & \ldots. & \ldots \\ DL1\,(x) & \ldots. & DLj\,(x) & \ldots. & DLC\,(x) \end{pmatrix}$$

Le modèle de décision (DT) d'une certaine classe est la décision moyenne des profiles de décision des exemples d'apprentissage de cette classe. L'élément (k,s) de la matrice DT pour la classe i est calculé par l'équation suivante [R. Ebrahimpour et al, 2009]:

$$Dti\,(k,s)\,(Z) = \frac{\sum_{j=1}^{N} Ind(Z_j,\omega_i)\; d_{k,s}\,(Z_j)}{\sum_{j=1}^{N} Ind\,(Z_j,\omega_i)} \qquad (6.4)$$

K = 1, ……., L

S = 1, ……., C

Où, Z_j est le j ème élément des données d'apprentissage, $Ind\,(Z_j,\omega_i)$ est une fonction indicatrice, sa valeur est à 1 si Z_j appartient à la classe i et 0 sinon.

Dans la phase du test, les modèles de décision comparent la matrice profile de décision DP d'un exemple d'entrée avec tous les DTs et proposent pour chaque classe une sortie égale à la similarité entre DT et DP. Il existe plusieurs mesures de similarité tels que [R. Ebrahimpour et al, 2009]:

$$S = \left(DT, DP\ (x)\right) = 1 - \frac{1}{L.C} \sum_{k=1}^{L} \sum_{s=1}^{C} \sqrt[4]{\left|DT_i(k,s) - d_{k,s}(x)\right|} \qquad (6.5)$$

6.7.1.2 Les combinaisons de type rang

Un vote pondéré est le rang associé à une classe par un classificateur. Pour un problème à N classes, chaque classificateur fournit un rang N à la 1ère classe de la liste des classes possibles, un poids N-1 à la 2ème, …, un poids 1 à la dernière [H. Zouari et al, 2002].

Ces étapes sont exécutées pour chacun des L classificateurs, ensuite, on additionne les votes obtenus pour tous les classificateurs par les différentes classes. On obtient à la fin un tableau qui donne le nombre total de votes obtenus pour chacune des classes par les différents classificateurs [K. H. Zouari, 2004]. La décision finale est effectuée par les techniques de Borda count comme la somme, la moyenne, la médiane… [H. Zouari et al, 2002]

La somme :
La décision (Ci) = somme de tous les rangs proposés par tous les classificateurs pour la classe Ci puis tri décroissant selon le rang obtenu.

La moyenne :
La décision (Ci) = (somme des rangs proposés pour la classe Ci) / (le nombre de classificateurs), cette technique n'a pas d'intérêt si les listes à combiner ont la même taille.

La médiane :
La décision (Ci) = rang du milieu de la liste des rangs proposés pour la classe Ci.

Borda count itératif :
Pour chaque itération, la classe ayant le borda count le plus faible est éliminée.

Meilleur rang *:*
Cette technique ne considère que le meilleur rang (le rang le plus élevé) pour chaque classe, puis tri par ordre décroissant de la liste des rangs obtenus. Cette technique est préférable pour les problèmes ayant un grand nombre de classes et peu de classificateurs [H. Zouari et al, 2002].

Vote par somme pondérée : Dans cette technique, les rangs $r_{i,j}$ associés aux classes C_i sont pondérés par des coefficients w_j indiquant la vraisemblance associée à chaque classificateur ej de la façon suivante [I. Gasmi et al, 2005], [K. H. Zouari, 2004]:

$$\text{La décision } (C_i) = \sum_{j=1}^{L} w_j r_{i,j} \qquad L \text{ est le nb de classificateurs} \qquad (6.6)$$

6.7.1.3 Les combinaisons de type mesure

Ces méthodes permettent de combiner des probabilités a posteriori. La qualité de ce type de méthodes dépend de la façon dont sont estimées les probabilités a posteriori.

Une fois un ensemble de probabilités $\{p_{ij}(x), i = 1\ldots L ; j = 1\ldots.c\}$ pour L classificateurs et c classes est calculé pour une entrée $\boldsymbol{x}$, ces probabilités doivent être combinés dans un nouveau ensemble $qj(x)$ qui peut être le maximum de tout les classificateurs.
Il existe deux types de règles : les règles fixes et les règles avec apprentissage [R. Duin et al, 2000], [H. Zouari et al, 2002].

6.7.1.3.1 Les Règles de combinaison fixes

La nouvelle confiance $q_j(x)$ pour la classe j *est* calculée par :

$$q_j^{'(x)} = règle_i\left(P_{ij}(x)\right) \qquad (6.7)$$

$$q_j(x) = \frac{q_j'(x)}{\sum_i q_j'(x)} \qquad (6.8)$$

Les combinateurs suivant peuvent être utilisés comme règle dans l'expression (6.7) : **Maximum, Médiane, Moyenne, Minimum** et **Produit** [R. Duin et al, 2000], [H. Zouari et al, 2002]. La règle *maximum*, par exemple, choisit le classificateur produisant la confiance la plus élevée. En revanche, la règle minimum choisit le classificateur ayant la moindre valeur de confiance. La moyenne et médiane calculent la moyenne de la probabilité postérieure réduisant de ce fait les erreurs d'évaluation.

La décision finale se fait de la manière suivante [R. Duin et al, 2000]:
$Ci(x) = \max_j(q_j(x))$ sinon rejet

La moyenne de Bayes :
Si $P_j(C_i/x)$ est la probabilité a posteriori d'une entrée x associée à la classe C_i par un classificateur e_j, la probabilité a posteriori moyenne de tous les classificateurs est calculée par [K. H. Zouari, 2004]:

$$P_{moy}(C_i/x) = \frac{1}{L}\sum_{j=1}^{L} P_j\,(C_i/x) \qquad (6.9)$$

la forme x est classée dans la classe C_i si $P_{moy}(C_i/x)$ est maximale.
La probabilité,a posteriori $P_j(C_i/x)$, est directement fournie par un classificateur bayesien. Pour les autres types de classificateurs, cette probabilité peut être estimée de différentes manières. Par exemple, pour un classificateur e_j de type k-ppv, la transformation est de la forme suivante [K. H. Zouari, 2004]:

$$P_j(x\epsilon\, C_i/x) = \frac{k_i}{k_{nn}} \qquad (6.10)$$

Où k_i est le nombre d'exemples de la classe C_i ($k_i \geq 0$) et k_{nn} le nombre total d'exemples les plus proches reconnus par le classificateur e_j avec [K. H. Zouari, 2004]:

$$k_{nn} = \sum_{j=1}^{N} k_j \qquad (6.11)$$

Les règles pondérées :

Dans ce type de règles, les mesures $m_{i,j}$ associées aux classes C_i sont pondérées par des coefficients w_j indiquant la vraisemblance associée à chaque classificateur e_j. Chaque probabilité a posteriori P_i d'une classe C_i peut être obtenue par l'une des règles suivantes [K. H. Zouari, 2004]:

$$P_i = \lambda \sum_{j=1}^{L} w_j m_{i,j} \qquad (6.12)$$

$$P_i = \prod_{j=1}^{L} m_{i,j}^{w_j} \qquad (6.13)$$

Si λ =1, la 1 ère règle est nommée la somme pondérée, si λ =1/L, cette règle est nommée la moyenne pondérée. La 2ème règle est nommée le produit pondérée.

6.7.1.3.2 Les réseaux de neurones

Dans cette technique, chaque exemple de la base de test est classé par l'ensemble des classificateurs et puis les sorties sont combinées par le réseau de neurones pour fournir la décision finale. Les sorties des classificateurs après le test sont donc considérées pour le réseau de neurones comme un nouvel ensemble d'attributs caractéristiques.
Il a été montré dans [G.Giacinto, 1998], que, lorsque les classificateurs de base sont dépendants, la fusion par réseau de neurones est plus efficace que celle par le vote majoritaire et la moyenne de Bayes [K. H. Zouari, 2004].

Les méthodes de fusion de classificateurs se distinguent par le type d'information en sortie. La sortie de type classe est une étiquette. La sortie de type rang est une liste triée de propositions. La sortie de type mesure est une mesure de confiance associée aux classes reflétant leurs préférences les unes par rapport aux autres. Il a été montré dans [J. Parker, 2001] que la fusion des sorties de type rang est peut être plus efficace que celle des sorties de type classe et mesure. La méthode BKS n'est pas souhaitable si la base de données est de petite taille [K. H. Zouari, 2004].

6.7.2 Combinaison par sélection de classificateurs

Cette classe est basée sur un grand nombre de classificateurs généralement obtenus par sélection des sous-ensembles des données d'apprentissage à partir d'un grand ensemble des données d'entrées, ou par sélection des sous-ensembles d'attributs caractéristiques à partir de tout l'ensemble des attributs disponibles, ces classificateurs sont ensuite entraînés sur les sous-ensembles sélectionnés [S.Tulyakov et al, 2008], [M. Last et al, 2002]. Dans l'approche de sélection, un classificateur est généralement sélectionné pour classer une certaine entrée [L.I. Kuncheva, 2004].

Ce type de méthodes est basé sur un grand nombre de classificateurs générés par des méthodes spécifiques tels que : Le Bootstrapping, le boosting, le bagging et le stacking. D'autres méthodes appliquent différentes conditions d'apprentissage telles que le choix aléatoire des poids initiaux pour l'entraînement d'un réseau de neurones ou le choix des dimensions des arbres de décision [T.K. Ho, 1998], d'autres méthodes génèrent des classificateurs basées sur la séparation de l'espace des attributs caractéristiques en différentes classes [S.Tulyakov et al, 2008]. [E.M. Kleinberg, 2000].

Les méthodes les plus simples utilisées pour la combinaison sont des règles fixes appliquées aux sorties de tous les classificateurs générés tel que le vote majoritaire. Il existe d'autres méthodes plus complexes basées seulement sur la sélection des classificateurs contribuant à la combinaison telles que le boosting et le stacking [S.Tulyakov et al, 2008].

6.7.2.1 Génération des ensembles d'apprentissage

Afin d'aboutir à divers ensembles d'apprentissage qui seront utilisés pour construire les classificateurs de base, plusieurs techniques peuvent être appliquées, parmi lesquelles nous présentons les techniques de : Bootstrapping, *le* bagging et le *boosting*.

Le Bootstrapping :
Le Bootstrapping est une procédure de division de l'ensemble des exemples d'apprentissage afin d'obtenir une multitude des sous-ensembles pour entraîner chaque classificateur sur chacun de ces sous-ensembles [M. Last et al, 2002].

Le boosting :
L'idée générale de boosting est de développer de manière progressive l'ensemble *des* classificateurs, en ajoutant un seul classificateur à la fois. Chaque nouveau classificateur s'occupe des cas «mal classés» (c'est-à-dire les cas difficiles) par les autres classificateurs. La sélection des exemples d'apprentissage commence uniformément, et progresse en incrémentant la probabilité des exemples difficiles [L.I. Kuncheva, 2004].

Le Boosting associe un poids à chaque exemple de l'ensemble d'apprentissage [Y. Freund et al, 1999]. Après chaque cycle d'apprentissage, les poids des exemples sont mis à jour en fonction de la performance du classificateur sur l'exemple correspondant. Dans la 1ère étape de l'algorithme, tous les poids sont identiques, ensuite et pour chaque cycle, les poids des exemples bien classifiés seront décrémentés et les poids des exemples inexactement classifiés seront augmentés en obligeant le système de se concentrer aux exemples durs de l'ensemble d'apprentissage [Y. Freund et al, 1999], [S.Tulyakov et al, 2008].

Un type très populaire du Boosting est l'AdaBoost (Le Boosting adaptatif), l'algorithme AdaBoost génère un ensemble de classificateurs et puis exécute un vote pondéré. La pondération de chaque classificateur est déterminée selon sa performance de prédiction. Les poids des exemples d'apprentissage sont mis à jour basé sur les classificateurs de l'étape précédente. Le but du Boosting est de forcer les classificateurs finaux de réduire au minimum l'erreur sur les différentes distributions des entrées. La classification finale est obtenue par le vote pondéré. Le Boosting a été appliqué avec succès à un grand nombre d'applications [S.Tulyakov et al, 2008].

Le Bagging :

Le Bagging est une méthode simple permettant de générer de multiples classificateurs et d'exécuter plusieurs sessions d'apprentissage avec le même classificateur sur différents sous-ensembles de l'ensemble d'apprentissage, ou avec un classificateur ayant des paramètres légèrement modifiés. Chaque session d'apprentissage crée un nouveau classificateur. L'approche basée sur cette idée a été proposée dans [L. Breiman, 1996] et puis devenue populaire sous le nom "bagging". Cette méthode engendre des ensembles d'apprentissage avec remplacement de l'ensemble d'apprentissage original, et chaque ensemble ayant comme résultat un classificateur légèrement différent après l'étape d'apprentissage. La technique utilisée pour la génération des sous ensembles d'apprentissage est connue sous le nom "bootstraping". Le Bagging a montré de grandes capacités d'apprentissage sauf dans le cas des classificateurs faibles homogènes où il peut donner de légères améliorations. Dans ce cas, le Boosting est préférable [S.Tulyakov et al, 2008].

L'algorithme Bagging [L.I. Kuncheva, 2004]:

Z est l'ensemble des exemples d'apprentissage.

L'étape d'apprentissage

1. Initialisation des paramètres
   D = ø, l'ensemble vide des classificateurs
   L, le nombre de classificateurs à entraîner
2. Pour k =1,…, L
   Prendre un bootstrap d'exemples S_K à partir de Z
   Construire un classificateur D_K pour S_K comme ensemble d'apprentissage
   Ajouter ce classificateur à l'ensemble courant D= DUD_K
3. Renvoyer D

L'étape de classification

Exécuter D_1,…., D_L sur l'entrée x
La classe ayant le nombre maximum de vote est la classe finale de x

---

6.7.2.2 Le clustering

Le clustering est une méthode de sélection de classificateurs qui a été proposée dans [L.I. Kuncheva, 2000], [L. I. Kuncheva, 2002]. Cette méthode commence par une étape d'apprentissage, et puis procède par une étape de classification comme montré ci-dessous. Des régions R_1. . . , R_K peuvent être estimés à priori pour l'apprentissage de tous les classificateurs. Un classificateur peut être entraîné sur les données d'une certaine région R_j pour devenir un expert $D_{i(j)}$. Les performances de cette méthode dépendent fortement de la manière d'affectation des régions aux classificateurs [L.I. Kuncheva, 2004].

L'algorithme du Clustering [L.I. Kuncheva, 2004]

L'apprentissage

1. Créer un ensemble de classificateurs $D_1, \ldots, D_L$ en utilisant les données de l'ensemble d'apprentissage Z.

Sélectionner le nombre de régions K :

2. grouper Z en K groupes, $C_1, \ldots, C_K$ en utilisant, par exemple, la méthode K-means. Trouver les centres de ces groupes $v_1 ; \ldots ; v_K$

3. Pour chaque groupe C_j ; (définir sa région Rj), et estimer l'exactitude de classification par les classificateurs $D_1, \ldots, D_L$: Sélectionner le classificateur le plus performant pour la région R_j et dénoter celui-ci par $D_{i(j)}$.

4. Renvoyer $v_1, \ldots, v_K$ et $D_{i(1)}, \ldots, D_{i(K)}$.

La classification

Pour une certaine entrée « x », trouver le centre du groupe le plus proche parmi : $V_1, \ldots, V_K$. soit V_j
Utiliser le classificateur $D_{i(j)}$ pour classifier x

6.8 Quelques applications de la combinaison de classificateurs :

Dans cette section, nous présentons quelques exemples multi classificateurs tirés de la littérature pour la reconnaissance de caractères manuscrits.

- **Combinaison de classificateurs pour la reconnaissance de chiffres manuscrits** : Dans l'approche décrite dans [S. Arora et al, 2009] deux classificateurs à base de réseaux de neurones ont été employés, ces réseaux sont entraînés sur des attributs caractéristiques différents. Les sorties de ces deux réseaux de neurones sont les confiances associées à chacune des classes possibles. Une fonction d'agrégation basée sur le vote majoritaire pondéré a été employée pour combiner les sorties de ces deux classificateurs, cette combinaison a permis de meilleurs résultats en comparaison à des classificateurs simples.

- **Combinaison de classificateurs pour la reconnaissance des caractères tamil :** La méthode proposée dans [R. J. Kannan et al, 2009, est basée sur la fusion de deux algorithmes pour la reconnaissance de caractères manuscrits Tamil. Ces deux algorithmes sont un SVM et un modèle de markov caché (HMM). Lorsque la sortie des deux algorithmes est dissimilaire pour un caractère donné, un combinateur final est employé afin de prévoir le caractère correct. Ce combinateur est un réseau de neurones à base radial (RBF) qui a comme entrées : les poids des deux algorithmes et le caractère

actuel. L'utilisation de ce dernier a permis une meilleure exactitude en termes du taux de reconnaissance [R. J. Kannan et al, 2009].

- **Trois réseaux MLPs entraînés par différents extracteurs caractéristiques :** Dans le travail présenté de [K. M. Kim et al, 2004] trois ensembles d'attributs caractéristiques ont été utilisés pour représenter les chiffres manuscrits. Le résultat de la reconnaissance montre que les attributs proposés représentent efficacement les variations d'un caractère telles que la pente, la taille et l'épaisseur, … etc. Trois perceptrons multicouches (MLPs) ont été proposés en tant que classificateurs simples, et l'intégrale floue a été utilisé pour combiner les sorties des trois réseaux MLPs.

- **Six MLPs entraînés par six types différents d'attributs caractéristiques** : Ce système [Rafael M. O et al, *2010*] comprend six MLPs ayant la même architecture mais entraînés par six types différents d'attributs caractéristiques et un module de combinaison. Chaque réseau MLP calcule la probabilité à posteriori de chaque chiffre et les envoient au module de combinaison. Deux types de combinateurs ont été employés : des règles fixes comme : la somme, le produit, le maximum, la médiane et le vote. Le deuxième combinateur est un réseau de neurones MLP avec une couche cachée. Ce combinateur a été entraîné en choisissant (5000 images d'apprentissage par chiffre) et (1000 images de validation par chiffre). Pour chaque image, la probabilité à posteriori est estimée par chaque méthode d'extraction d'attributs caractéristiques et puis employée comme attributs du réseau de neurones. Il a été montré que toutes les règles de combinaison donnent de grandes améliorations par rapport à un classificateur d'attributs caractéristiques tout seul. Le système utilisant un réseau MLP comme combinateur a donné un taux d'erreur de 0.32% comme meilleur résultat. Il a été montré que les réseaux de neurones permettent d'effectuer la meilleure combinaison. La règle maximum a également donné de bon résultat.

- **Combinaison de classificateurs pour la reconnaissance des caractères arabe :** Dans l'article [N. Azizi et al, 2010], deux approches pour la reconnaissance des caractères arabes manuscrits en utilisant un système multi-classificateurs ont été présentées. La première investit sur *la coopération de plusieurs attributs caractéristiques* tout en proposant une méthode de choix des attributs les plus discriminants qui seront par la suite l'entrée du système. La seconde propose *un algorithme de sélection de c*lassificateurs. Trois méthodes de combinaison ont été examinées dans les deux approches proposées qui sont : le vote, le vote pondéré, et le comportement de l'espace de connaissance (BKS) : Behavior Knowledge Space [N. Azizi et al, 2010].

  L'approche de coopération des attributs caractéristiques [N. Azizi et al, 2010] **:** Dans cette approche, trois types d'attributs caractéristiques ont été employés.
  - chaque ensemble des attributs caractéristiques est l'entrée d'un classificateur neuronal ; ce qui nous mène a un système de trois classificateurs.
  - les mêmes ensembles des attributs caractéristiques ont été appliqués à un système constitué de trois modèles de Markov cachés HMMs, où le premier est un HMM

discret utilisant des attributs structurels comme observations discrètes et les deux autres sont des HMMs continus.
- les classificateurs sont entraînés avec différents échantillons de la base d'apprentissage.
- un autre système remplaçant les deux modèles HMM par des RNs (réseaux de neurones) a été examiné.

L'approche de sélection des classificateurs [N. Azizi et al, 2010] **:** Cette approche choisit un nombre « m » de L classificateurs de base.
1) elle commence par un ensemble contenant un seul classificateur qui est le meilleur classificateur en termes d'exactitude pendant la phase d'apprentissage.
2) pour chaque itération, elle choisit parmi tous les classificateurs possibles celui qui a mieux contribué à l'amélioration de la performance globale du système une fois ajouté à l'ensemble courant. Une fois l'ensemble de tous les classificateurs est sélectionné, des méthodes de combinaison comme, le vote, le vote pondéré, et BKS sont appliquées pour obtenir la décision finale.

Conclusion

6.9 Conclusion

La combinaison de classificateurs complémentaires permet de meilleures performances.
La combinaison de classificateurs prend trois formes principales qui sont :

- Séquentielle : où les classificateurs sont employés les uns à la suite des autres pour réduire l'ensemble des classes possibles d'un classificateur à un autre.
- Parallèle : Les classificateurs sont entraînés indépendamment et puis les résultats trouvés sont fusionnés.
- Hybride : cette classe est une mixture des deux approches précédentes

Les méthodes de combinaison parallèles sont les plus répandues dues à leur efficacité et à leur simplicité inhérente.

Les méthodes parallèles peuvent être distinguées en deux classes :

- ***Les méthodes de fusion*** où tous les classificateurs participent à la prise de décision, ces méthodes se distinguent par le type d'information en sortie. La sortie de type classe est une étiquette. La sortie de type rang est une liste triée de propositions. La sortie de type mesure est une mesure de confiance. La fusion des sorties de type rang est plus efficace que celle des sorties de type classe et mesure. La méthode BKS n'est pas souhaitable si la base de données est de petite taille [K. H. Zouari, 2004].
  Le type Classe *inclus* : les méthodes de vote, BKS, Théorie de Bayes, Dempster Shafer.
  Le type Rang *inclus* : Meilleur rang, Borda Count, Intersection et union, Régression logistique.
  Le type Mesure *inclus* : Les règles fixes (somme, produit, moyenne, maximum, minimum, médiane), les réseaux de neurones, l'intégral flou.

- ***les méthodes de sélection*** aussi appelées les méthodes ensemblistes où l'ensemble des classificateurs à combiner est sélectionné en fonction des exemples d'apprentissage (Bagging, Bousting, stacking Random sub-spaces..).

Les méthodes de vote telles que *Majorité unanime, majorité simple, vote à la pluralité, le vote pondéré sont basées sur* le vote de chaque classificateur (acceptation/rejet). La classe qui a reçu un nombre de vote (acceptation) supérieur à un seuil constitue la décision finale sinon rejet.

La méthode (BKS) crée un espace K–dimensionnel où chaque dimension correspond à la décision d'un classificateur K, Une unité de BKS est une intersection des décisions de chaque classificateur. La décision finale pour un exemple d'entrée est dérivée de l'unité, la classe dont la probabilité est maximum constitue la décision finale.

La combinaison de type rang : Pour un problème à N classes, chaque classificateur fournit un rang N à la 1ère classe de la liste des classes, un poids N-1 à la 2ème, …, un poids 1 à la dernière. La décision finale est effectuée par les techniques de Borda count comme la somme, la moyenne, la médiane…

Les combinaisons de type mesure : Ces méthodes permettent de combiner des probabilités a posteriori. Par exemple, la règle maximum choisit le classificateur produisant la confiance la plus élevée. En revanche, la règle minimum choisit le classificateur ayant la moindre valeur de confiance...

Les réseaux de neurones : les sorties des classificateurs après le test sont considérées comme un ensemble des entrées du réseau de neurones.

Combinaison par sélection de classificateurs

Le Bootstrapping : est une procédure de division de l'ensemble des exemples d'apprentissage afin d'obtenir une multitude des sous-ensembles pour entraîner chaque classificateur sur chacun de ces sous-ensembles.

Le boosting : associe un poids à chaque exemple d'apprentissage, ces poids au départ sont identiques, ensuite et pour chaque itération, les poids des exemples correctement classifiés seront décrémentés et les poids des exemples inexactement classifiés seront augmentés en obligeant le système de se concentrer aux exemples compliqués.

L'AdaBoost : Un type populaire du Boosting est l'AdaBoost (Le Boosting adaptatif), l'AdaBoost associe un poids pour chaque classificateur. La pondération est déterminée selon sa performance de prédiction et puis exécute un vote pondéré.

Le bagging : Cette méthode engendre des sous-ensembles d'apprentissage avec remplacement de l'ensemble original, et chaque ensemble ayant comme résultat un classificateur légèrement différent après l'étape d'apprentissage.

Deuxième partie

L’intelligence collective et l’intelligence en essaim

Chapitre 7
L'INTELLIGENCE COLLECTIVE ET L'INTELLIGENCE EN ESSAIM

7.1 Introduction

L'intelligence artificielle (IA) est un domaine de recherche dont le but est d'imiter l'intelligence humaine dans le domaine des machines tels que les robots, les cellules électroniques, les ordinateurs, …etc. [R.A. Brooks, 1991]. Bien que les ordinateurs puissent réaliser beaucoup de tâches difficiles, ils restent incapables de penser comme un être humain. Le but de l'intelligence artificielle est de permettre à une machine de réagir d'une façon intelligente. C'est une tâche très difficile mais facile à comprendre ses principes de base imitant le comportement collectif des créatures. L'étude du comportement intelligent des créatures peut servir d'aide pour réaliser des systèmes caractérisés d'intelligence comme les interactions collective ou coopérative entre plusieurs ordinateurs ou plusieurs réseaux, les virus et les antivirus d'ordinateurs, la robotique collective imitant les mouvements des insectes sociaux et bien d'autres.
La vie artificielle est implicitement basée sur la théorie d'évolution puisqu'elle est inspirée de l'étude des phénomènes biologiques fondamentaux. Plus particulièrement, elle essaye de voir comment des propriétés émergentes peuvent être extraites à partir des règles d'évolution simples [C.W. Reynolds, 1987]. Les propriétés émergentes sont inhérentes à la vie c-à-d un comportement complexe émerge des règles simples [C.W. Reynolds, 1987]. Au cours ces dernières années, les biologistes révèlent plusieurs mystères entourant les insectes sociaux dans les domaines désignés sous le nom de « l'intelligence collective », de « l'intelligence en essaims » et du « comportement émergent ». Plus récemment, le paradigme d'essaim qui a été intensivement étudié dans une large gamme des applications, ouvrant de nouvelles façons de penser aux problèmes réels tels que la biologie, l'informatique, la science économique, …etc. Un essaim est tout simplement un ensemble d'agents auto organisationnels capables d'exécuter une résolution distribuée de problèmes [M. Dorigo et al, 2004].
Les systèmes biologiquement inspirés ont gagné une très grande importance ces dernières années pour développer de nouvelles technologies tirant profit des exemples offerts de la nature. Les phénomènes sociaux bien qu'ils se composent des individus simples avec des capacités limitées présentent un comportement collectif intelligent qui émerge par auto-organisation tel que le cas des insectes sociaux. Ces systèmes fournissent des techniques très importantes pouvant être servies dans le développement des systèmes de l'intelligence artificielle distribuée [S. Hassas et al, 2006], [G.D.M. Serugendo et al, 2006].

Dans ce chapitre, nous introduisons le domaine de l'intelligence collective, ainsi que les notions de base qui peuvent nous servir à comprendre ce domaine très vaste telles que l'intelligence en essaim, l'intelligence artificielle distribuée, les systèmes multi-agent, l'auto-organisation et plus particulièrement l'intelligence en essaim aussi bien que les méthodes d'optimisation les plus récentes qui font partie de ce domaine.

7.2 L'intelligence collective

L'intelligence collective détermine l'achèvement de tâches très complexes au sein d'une société d'individus (agents) simples grâce à leurs interactions multiples. Ces agents n'ont qu'une vision partielle de leurs environnement et n'ont pas de cognition de l'objectif du système mais peuvent aboutir à des tâches difficiles grâce à un mécanisme fondamental appelé synergie. La synergie créée par les interactions multiples et répétées des agents fait émerger des tâches supérieures à celles d'un individu isolé. Les formes d'intelligence collective sont très diverses selon les types de communauté et les membres qu'elles réunissent.
La résolution collective de problèmes s'intéresse aux interactions entre des entités autonomes et à l'émergence au niveau supérieur de l'ensemble d'un système complexe. Cette technique porte sur la manière de diviser un problème entre des entités relativement simples mais capables de s'auto organiser afin d'accomplir d'une façon collective un but commun. Cette vision distribuée de problèmes a suscité l'attention croissante des chercheurs pour la qualité de ses applications et ses avantages tels que le parallélisme, la vitesse, flexibilité,…etc.
Les comportements collectifs d'oiseaux, des bancs de poissons, des colonies d'insectes sociaux et des abeilles étant auto organisés, adaptatifs, et qualifiés d'intelligents sont porteurs de capacités d'optimisation suggérant ainsi des heuristiques que l'on peut exploiter pour résoudre des problèmes.

7.3 L'intelligence en Essaim

L'intelligence en essaim est un champ naissant de l'intelligence artificielle inspirée des modèles comportementaux des insectes sociaux tels que les fourmis, les abeilles, les guêpes et les termites [E. Bonabeau et al, 1999]. Cette approche utilise des agents simples et flexibles qui forment en groupe une intelligence collective. L'intelligence en essaim est une approche alternative aux modèles traditionnels d'intelligence artificielle, montrant des propriétés d'autonomie, d'émergence, de la robustesse et de l'auto-organisation.
L'exemple classique d'un essaim est l'essaimage d'abeilles autour de leur ruche ; il existe aussi d'autres systèmes avec une architecture semblable comme la colonie de fourmis qui peut être considérée comme un essaim dont les différents agents sont des fourmis. De même un groupe d'oiseaux est un essaim des oiseaux. Un système immunitaire [L.N. De Castro et al, 1999] est un essaim de cellules et de molécules [J. Vesterstrøm et al, 2002]. L'algorithme d'optimisation d'essaim de particules (PSO) modélise le comportement social d'assemblement d'oiseaux ou de poissons [J. Kennedy et al, 1995].
Les méthodes d'optimisation basées sur l'essaimage sont des algorithmes inspirés de la nature dirigeant la recherche vers des solutions optimales. Les algorithmes les plus connus de cette catégorie incluent les algorithmes d'optimisation par colonie de fourmis (ACO) [M. Dorigo et al, 2004], les algorithmes génétiques (GAs) [D.E. Goldberg, 1989] et l'algorithme d'optimisation par essaims de particules (PSO) [J. Kennedy et al, 2001], [D. T. Pham et al, 2007].
Les algorithmes génétiques fournissent une technique d'optimisation adaptative basée sur le processus de la sélection naturelle et la recombinaison génétique. Cet algorithme fonctionne par sélection des solutions courantes et de leurs appliquer des opérateurs génétiques tels que la mutation et le croisement afin de créer de nouvelles populations [D. T. Pham et al, 2007]. Une population des individus représentant les solutions possibles pour un problème donné est

soumise au principe darwinien de la "survie des plus convenables" au cours des générations consécutives. Les individus les plus réussis propageront leurs caractéristiques par des gènes qui seront recombinés pour créer de nouveaux individus. Des résultats prometteurs ont été obtenus par les algorithmes génétiques tout en évoluant une population afin de s'adapter à son environnement par mutation et sélection, [D. Whitley, 1994].

Une des meilleures méthodes d'optimisation par intelligence collective est la méthode d'optimisation par essaim de particules (PSO), développée, en 1995 par James Kennedy, et Russell Eberhart. Le but initial des auteurs était de modéliser l'interaction sociale humaine basée sur des comportements émergents tels que l'assemblage des oiseaux basé sur le modèle des boids de Craig Reynolds. Reynolds a créé une bande d'oiseaux virtuels où chaque oiseau évite la collision avec ses voisins, se déplace dans la direction de la bande, et reste près de ses voisins [C.W. Reynolds, 1987]. Les boids volent dans un seul groupe, se repartirent en sous-groupes quand ils observent des obstacles et se regroupent après. Ce comportement d'assemblage est un phénomène émergent au niveau supérieur issu d'interactions relativement simples entre des éléments fondamentalement simples (non intelligent) [C.W. Reynolds, 1987].

Le PSO est un procédé d'optimisation basé sur le comportement social des groupes d'oiseaux ou de poissons [J. Kennedy et al, 2001]. Des solutions individuelles sont considérées en tant que "particules" qui évoluent ou changent leurs positions au cours du temps. Chaque particule modifie sa position dans l'espace de recherche selon sa propre expérience et celle de ses particules voisines tout en enregistrant sa meilleure position visitée par elle-même et ses voisins, combinant de ce fait une recherche locale avec une recherche globale [J. Kennedy et al, 2001], [D. T. Pham et al, 2007].

Les fourmis sont des insectes qui ne peuvent effectuer que des actions simples. Cependant, une colonie de fourmis manifeste un comportement collectif complexe fournissant des solutions intelligentes telles que la formation des ponts ou la recherche des chemins les plus courts du nid aux sources de nourriture [R. Schoonderwoerd et al, 1996]. Une fourmi n'a aucune connaissance globale sur la tâche qu'elle accomplit. Les actions de la fourmi sont basées sur des décisions locales et sont habituellement imprévisibles. Le comportement intelligent émerge par la suite de l'auto-organisation et d'une communication indirecte entre les fourmis. C'est ce qu'on appelle ***Emergent Behavior*** [G.D.M. Serugendo et al, 2006]. Les fourmis emploient un système de communication de signalisation basé sur le dépôt de la phéromone au-dessus du chemin qu'elles traversent. La phéromone est une hormone produite par les fourmis permettant d'établir une sorte de communication indirecte entre elles. Fondamentalement, une fourmi isolée se déplace au hasard, mais quand elle trouve un chemin de phéromone il y a une grande probabilité que cette fourmi poursuivra ce chemin [R. Schoonderwoerd et al, 1996].

Il existe beaucoup d'autres exemples naturels de comportements émergents tels que les colonies d'abeilles, que nous allons montrer en détaille par la suite puisqu'elles font partie de nos essais.

7.4 L'intelligence artificielle distribuée

L'intelligence Artificielle s'intéresse à l'étude du comportement d'un seul système et ainsi un seul agent [L. Gasser, 1992]. L'une des applications séduisantes de l'intelligence artificielle est l'émotion des machines. Une machine émotionnelle peut être définit comme un logiciel ou matériel pouvant reconnaître, exprimer ou même produire ses propres émotions. Son but est de comprendre le comportement de l'utilisateur pour interagir avec lui d'une façon plus intuitive. Particulièrement, c'est un robot capable de comprendre l'état mental d'un utilisateur pour réagir avec lui. Cependant, il n'est pas facile de comprendre le comportement humain et d'établir des opérations comme [I.P. Wright, 1997] :
Si (utilisateur) == triste alors offrir (une fleur) et jouer (un jeu) sinon demander ("comment allez vous").

L'intelligence artificielle distribuée (L'I.A.D) s'intéresse à l'étude du comportement des systèmes multi-agents caractérisés par plusieurs interactions entre plusieurs agents autonomes. Cette approche est justifiée par le fait que l'interaction de plusieurs expertises peut mener à une expertise plus sure et plus robuste. L'IAD s'adapte mieux aux problèmes réels que l'IA classique [L. Gasser, 1992].

L'intelligence artificielle distribuée essaye de comprendre les principes des comportements de sociétés appelées agents, et leurs interactions afin de produire un comportement global significatif. Dans ce type de société, chaque agent est une intelligence artificielle individuelle capable d'une certaine activité utile, mais plongée dans une société artificielle en communiquant et coopérant avec les autres agents qui peuvent augmenter sa performance [A.H. Bond et al, 1988], [F. Gandon, 2002].

7.5 Les systèmes Multi-Agent

Les systèmes multiagents offrent une manière d'enlever les contraintes du contrôle centralisé, planifié ou séquentiel. Un système multi-agent est ainsi décentralisé plutôt que centralisé, émergent plutôt que planifié, et concurrent plutôt que séquentiel. L'approche autonome d'agent remplace une base de données et contrôle un ordinateur centralisé avec un réseau d'agents, chacun doté d'une vue locale de son environnement et de la capacité à y répondre localement. L'exécution du système global émerge par interaction dynamique des agents dans un temps réel. Ainsi, le système n'alterne pas entre les cycles d'exécution du programme. Les systèmes d'agents autonomes sont inspirés des modèles de la biologie (les écosystèmes) et de la science économique [V.D. Parunak, 1996]. L'approche Multi-agents apporte plusieurs avantages que sont : L'adaptation, la coopération, la modularité, l'efficacité, la fiabilité et la réutilisation [L. Gasser, 1992].

Un agent est un système informatique situé dans un environnement, et qui est capable de l'action autonome dans cet environnement afin de répondre à ses objectifs [G.Weiss, 2000].

L'apprentissage dans un système distribué est fondé sur l'auto-organisation du futur comportement basée sur une expérience antécédente. "Distribué" signifie que plusieurs entités

(agents) contribuent à la solution de la tâche d'apprentissage globale selon ses différentes capacités ou préférences [G.Weiss, 2000].

Les systèmes Multi-agent (SMA) sont des sociétés d'entités autonomes agissantes l'un sur l'autre. Le comportement d'un SMA est un résultat de l'action et des interactions asynchrones répétées des agents. La notion d'auto-organisation est ainsi centrale dans les applications multi-agents. Les simulations Multi-agent peuvent également être employées pour étudier le comportement émergent des systèmes réels [S. K. Mostefaoui et al, 2003].

7.6 L'auto-organisation

Le comportement d'auto-organisation est la formation spontanée de structures bien organisées, des modèles ou des comportements à partir des conditions initiales aléatoires. La biologie, la chimie, la géologie, la sociologie ou encore les systèmes d'intelligence artificielle distribuée (DAI) sont des domaines où l'auto-organisation est souvent produite [S. K. Mostefaoui et al, 2003]. L'auto-organisation dans les insectes sociaux se fonde sur le mécanisme de la stigmergie.

L'auto-organisation a trois caractéristiques. D'abord, le système auto-organisationnel peut accomplir des tâches complexes à partir des comportements simples de ses constituants. Deuxièmement, un changement de l'environnement peut influencer le même système pour produire une tâche différente sans n'importe quel changement des caractéristiques comportementales de ses constituants. Enfin tout petit changement d'un comportement individuel peut influencer le comportement collectif du système [S. K. Mostefaoui et al, 2003].

Un système auto-organisationnel est plus important qu'un système déterministe ou préprogrammée ; un système planifié a un caractère non adaptatif ; une petite défaillance peut provoquer la panne du système puisque qu'il est basé sur la programmation déterministe des agents et de leurs communications qui est aussi une tâche difficile. Par contre un système auto-organisationnel exige un grand nombre d'individus qui se coordonnent à travers un environnement de manière collective. Ce comportement collectif possède un caractère adaptatif. Un tel système est donc simple, fiable et adaptatif basé seulement sur des règles simples définissants les différents comportements et interactions. En plus, la panne d'un agent n'affectera pas l'activité du système tout entière [S. K. Mostefaoui et al, 2003].

Dans un système déterministe, les programmes sont fortement complexes, car il est nécessaire de définir un comportement répondant à chaque situation possible, de plus, il est impossible de prévoir toutes ses conditions. Cependant, dans un système auto-organisationnel, les programmes sont simples et fonctionnent dans des situations imprévues et ainsi permettre au système de s'adapter aux conditions indéterminées. Pour ces raisons, les algorithmes auto-organisationnels sont employés pour contrôler les réseaux informatiques d'un grand nombre d'utilisateurs [S. K. Mostefaoui et al, 2003].

a)l'auto-organisation peut être définie comme un ensemble de mécanismes dynamiques, qui ont comme conséquence des structures au niveau global du système au moyen d'interactions entre ses composants de bas niveau. Ces mécanismes établissent des règles d'interactions de base entre les composants du système. Les règles s'assurent que les interactions sont exécutées au niveau

local sans n'importe quelle relation au modèle global. Bonabeau et autres ont caractérisé quatre propriétés de base sur lesquelles l'auto-organisation est fondée :
La Rétroaction (feedback) positive, rétroaction négative, fluctuations et interactions multiples [E. Bonabeau et al, 1999], [D. Karaboga, 2005] :

- La rétroaction positive est un "principe de base" qui favorise la création des structures convenables. Le recrutement et le renforcement comme la suivie des traces dans quelques espèces de fourmis ou les danses des abeilles qui peuvent être considérés comme exemples de la rétroaction positive.

- La rétroaction négative équilibre la rétroaction positive et aide à stabiliser le modèle collectif. Afin d'éviter la saturation qui pourrait se produire en termes d'explorateurs, l'épuisement de la source de nourriture, la concurrence aux sources disponibles de nourriture, un mécanisme de la rétroaction négative est nécessaire.

- Les fluctuations telles que les explorations aléatoires, les erreurs, commutation aléatoire de tâches entre les individus d'essaim sont essentielles pour la créativité et l'innovation. L'aspect aléatoire est souvent crucial pour des structures émergentes puisqu'il permet la découverte de nouvelles solutions.

- En général, l'auto-organisation exige une densité minimale des individus mutuellement tolérants, en leur permettant de se servir des résultats de leurs propres activités aussi bien que des autres.

7 .7 Exemples des méthodes d'intelligence en essaim

7.7.1 L'optimisation par essaim de particules

PSO est une technique évolutionnaire coopérative développée par Kennedy et Eberhart [J. Kennedy et al, 1995] inspirée du mouvement collectif des groupes d'animaux. Ces animaux peuvent être vus comme des agents qui ont la capacité à prendre leur propre décision en imitant le comportement réussi de leur voisinage tout en apportant leur expérience personnelle. En faite, PSO résulte d'une analogie avec le comportement collectif du déplacement d'animaux. Dans les groupes d'animaux, chaque individu se déplace selon sa propre connaissance qui peut être atteinte par le déplacement de ses voisins [J. Kennedy, 1997], [R.C. Eberhart et al, 2000].

Principe :

Cette méthode est basée sur la coopération entre des individus simples souvent appelés des agents. Ces agents possèdent une capacité de la mémorisation dans le sens où chaque agent se rappelle de sa meilleure position visitée et la meilleure position trouvée dans son voisinage. Chaque particule se déplace alors selon une trajectoire perturbée entre ses deux positions et de sa vitesse.

PSO est un algorithme simple et efficace qui commence par un certain nombre de particules placées de manière aléatoire dans l'espace de recherche et initialisées avec des vitesses aléatoires.

Formalisation :

Dans PSO, chaque particule est caractérisée par son propre vecteur de position et son vecteur de vitesse.

Pour un certain nombre d'itérations ou jusqu'à ce qu'une condition soit satisfaite les particules se déplacent dans l'espace de recherche selon les règles récursives suivantes :

$$\begin{cases} V_{id} = wV_{id} + c_1 \times r_1 \times (p_{id} - x_{id}) + c_2 \times r_2 \times (p_{gd} - x_{id}) \\ x_{id} = x_{id} + v_{id} \end{cases} \quad (7.1)$$

Où:

X est la position de la particule.

V: représente sa vitesse.

i est son index.

d est sa d ème *dimension* dans l'espace de recherche.

Pi et *Pg* sont respectivement sa meilleure position personnelle et la meilleure position trouvée dans son voisinage.

C_1 et C_2 : sont respectivement le facteur cognitif et le facteur social permettant le contrôle du comportement individuel et collectif de chaque particule.

r_1 et r_2 : sont des valeurs qui changent uniformément dans l'intervalle [0.1] pour permettre une meilleure exploration de l'espace de recherche

W : s'appelle le poids d'inertie, il sert à contrôler l'équilibre entre l'exploration et l'exploitation de l'espace de recherche.

PSO en pseudo-code [R.C. Eberhart et al, 2000]

Initialiser un nombre de particules (xi)
Pour un nombre d'itérations faire
Pour *i* = 1 jusqu'à le nombre de particules faire
Si fitness (*xi*) < Pbest alors
    Maj Pbest = *xi* % Pbest est la meilleure position personnelle
Fin si

Si fitness (*xi*) < gbest alors
    Maj gbest = *xi* % gbest est la meilleure position trouvée
Fin si

Pour *d* = 1 jusqu'au nombre de dimensions faire
Maj de vitesses et positions en employant l'équation. (7.1)
Fin pour
Fin pour

Figure7.1: L'algorithme PSO

7.7.2 L'algorithme des abeilles (BA)

La modélisation du comportement des insectes sociaux a attiré l'attention des chercheurs depuis longtemps et plus particulièrement le comportement des abeilles du miel caractérisé d'intelligence et riche des propriétés encourageant son utilisation comme modèle telles que la division de travail, la communication individuelle et social aussi bien que le comportement collectif et coopératif des abeilles [J.Teo et al, 2001].

Le fourragement des abeilles dans la nature :
Pendant la saison de récolte, la colonie d'abeilles emploie une partie de sa population pour procéder une recherche des sources de nourriture dans les champs qui entourent la ruche [V. Tereshko et al, 2005]. En particulier, les abeilles scoutes recherchent les endroits envahis de fleurs où le nectar est abondant et facile à extraire. Quand elles reviennent à la ruche, les abeilles scoutes déposent le nectar (ou pollen) rassemblé pendant le processus de la recherche. Les abeilles qui ont trouvé des ressources de la nourriture de haute qualité signalent la position de leur découverte aux abeilles restantes par danse connue sous le nom "waggle dance" [T.D. Seeley, 1996]. Cette danse est exécutée dans un secteur particulier de la ruche, et communique trois informations de base concernant l'endroit trouvé de fleurs : la direction où il est localisé, sa distance par rapport à la ruche, et sa qualité en nectar [S. Camazine et al, 2003]. Après la danse, l'abeille danseuse revient à l'endroit trouvé suivie des autres abeilles recrutées de la ruche. Le nombre d'abeilles recrutées dépend de la qualité de l'endroit trouvé. Les sites Fleurit qui sont riches en nectar attirent le plus grand nombre d'abeilles exploratrices [T.D. Seeley, 1996]. Une fois qu'une abeille recrutée retourne à la ruche, elle danse à son tour pour diriger d'autres abeilles vers la source de nourriture. Grâce à ce mécanisme, la colonie d'abeilles optimise l'efficacité du processus d'exploration de la nourriture [V. Tereshko et al, 2002], [D.T. Pham et al, 2008].

Le processus de fourragement des abeilles artificielles :
L'algorithme commence avec « n » abeilles (scoutes) aléatoirement dispersées à travers l'espace des solutions. Ensuite, l'algorithme entame quatre phases principales qui sont [D.T. Pham et al, 2008] :

La phase d'évaluation de fitness
La phase de recherche locale
La phase de recherche globale
La phase de création d'une nouvelle population d'abeilles

Dans *la phase d'évaluation* de fitness, la fitness de chaque site (endroit) visité est évaluée.

Dans *la phase de recherche locale*, les meilleures abeilles placées dans les *« m »* sites effectuent une recherche locale en recrutant leurs voisins. Plus d'abeilles sont recrutées dans le voisinage des « e » meilleurs sites. Ce recrutement différentiel est l'idée principale de l'algorithme des abeilles qui favorise une meilleure exploitation de l'espace de recherche.

Pour chaque site, la fitness des positions visitées par les abeilles recrutées est évaluée. Si une des abeilles recrutées localise une position d'une fitness plus élevée que celle de l'abeille site, l'abeille recrutée devient la nouvelle représentative du site [D.T. Pham et al, 2008].

Dans *la phase de recherche globale*, un nombre d'abeilles sont placées d'une façon aléatoire dans l'espace de recherche pour localiser de nouveaux sites. La recherche aléatoire représente l'effort d'exploration de l'algorithme d'abeilles [D.T. Pham et al, 2008].

À la fin de chaque itération, la nouvelle population d'abeilles est formée de deux groupes. Le premier groupe se compose de meilleures abeilles représentatives de chaque site qui est le résultat d'une recherche exploitante locale. Le deuxième groupe se compose d'abeilles scoutes liées aux solutions aléatoires, et représente le résultat d'une recherche exploratrice globale [D.T. Pham et al, 2008].

Un nouveau cycle d'optimisation est alors entrepris en choisissant les m meilleurs sites du cycle précédant pour lancer une exploitation locale. L'algorithme s'arrête si un critère d'arrêt est atteint [D.T. Pham et al, 2008].

7.7.3 L'optimisation par les abeilles artificielles (ABC):

Le modèle naturel :

Le modèle de fourragement minimal des abeilles se compose de trois sous-populations essentielles : les sources de nourriture, les explorateurs employés et les explorateurs non employés (les spectateurs et les scouts). On suppose qu'il y a une seule abeille employée pour chaque source de nourriture. Le modèle définit aussi deux modes principaux du comportement : le recrutement à une source de nectar et l'abandon d'une source de nectar [D. Karaboga, 2005].

- **Les sources de nourriture :** La qualité d'une source de nourriture dépend de beaucoup de facteurs tels que sa proximité au nid, sa richesse ou sa concentration d'énergie, et la facilité d'extraire cette énergie [D. Karaboga, 2005].

- **les explorateurs employés :** Ils se sont les abeilles associées à une source particulière de nourriture à exploiter. Ils diffusent des informations sur cette ressource, sa distance et sa direction par rapport au nid, la rentabilité de la ressource et partagent cette information avec une certaine probabilité [D. Karaboga, 2005].

- **les explorateurs non employés :** Ils se sont continuellement en recherche d'une source de nourriture. Il y a deux types de ces explorateurs : les scoutes, recherchant de nouvelles sources dans l'environnement qui entoure le nid et les spectateurs de nourriture attendant dans le nid et établissant une source de nourriture par partage d'information avec les explorateurs employés. Le nombre des scouts est environ 5-10% de la population. L'échange d'informations entre les abeilles est l'opération la plus importante pour la formation de la connaissance collective. Après avoir identifié la ressource de nourriture, l'abeille utilise ses propres possibilités pour mémoriser un emplacement et puis commence à l'exploiter. Par conséquent, l'abeille deviendra "un explorateur employé". L'abeille exploratrice prend une quantité de nectar et revient à la ruche pour la déposer [D. Karaboga, 2005].

Le modèle artificiel :

Dans le modèle artificiel, la colonie des abeilles artificielles est composée de trois groupes : les abeilles découvreuses associées aux sources de nourriture, les abeilles observatrices observant la

danse des abeilles découvreuses pour choisir une source et les abeilles exploratrices recherchant de manière aléatoire des sources de nourriture. Les abeilles exploratrices et observatrices sont également appelées les abeilles non employées. Dans l'algorithme ABC, la première moitié de la colonie se compose des abeilles employées (abeilles découvreuses) tandis que la deuxième moitié est constituée des abeilles observatrices. L'abeille employée dont la source de nourriture est épuisée devient une abeille exploratrice (scoute). La position d'une source de nourriture représente une solution possible au problème et sa quantité de nectar correspond à sa fitness. Le nombre d'abeilles employées est égal au nombre de sources de nourriture (solutions) puisque chaque abeille employée est associée à une seule source de nourriture [D. Karaboga, 2010].

La structure générale de l'algorithme ABC est comme suit [D. Karaboga, 2010] :

Phase d'initialisation
Répéter
Phase des abeilles employées
Phase des abeilles observatrices
Phase des abeilles exploratrices
Mémoriser la meilleure solution trouvée
Jusqu'à Nombre Maximum d'itérations

Phase d'initialisation :
Initialisation de la population des abeilles employées et non employées x_m ($m = 1….S_N$) S_N est la taille de la population. Chaque employée est un vecteur (une solution) constitué des paramètres x_{mi} du problème [D. Karaboga, 2010] :

$$x_{mi} = l_i + rand\,(0,1) * (u_i - l_i) \qquad (7.2)$$

Où li et u_i sont respectivement les limites inférieures et supérieures d'un paramètre x_{mi}. i est la dimension du problème.

Phase des abeilles employées :
Chaque abeille découvreuse x_m détermine une nouvelle source de nourriture voisine v_m qui peut etre mieux que x_m en utilisant la formule donnée par l'équation (7.3) [D. Karaboga, 2010].

$$v_{mi} = x_{mi} + \emptyset_{mi}\,(x_{mi} - x_{ki}) \qquad (7.3)$$

Où x_k est une source de nourriture aléatoirement choisie et $\emptyset_{mi}$ est un nombre aléatoirement choisi dans l'intervalle $[-a, a]$. Après la production de (v_m), sa fitness est calculée par l'équation (7.4) et une sélection par l'algorithme glouton (greedy) est appliquée entre v_m et x_m.

$$fit_m(x_m) = \begin{cases} \dfrac{1}{1 + f_m\,(x_m)} & si\ f_m\,(x_m) \geq 0 \\ 1 + abs\,(f_m(x_m)) & si\ f_m\,(x_m) < 0 \end{cases} \qquad (7.4)$$

Où $f_m\,(x_m)$ est la valeur de fitness de la solution x_m .

Phase des abeilles observatrices :
Une abeille observatrice choisit une source de nourriture selon une probabilité calculée basée sur les valeurs de fitness fournies par les abeilles employées. Pour ce faire, une technique de sélection par roulette est employée.

La probabilité P_m de choisir x_m par une abeille observatrice peut être calculée par l'expression suivante (7. 5) [D. Karaboga, 2010] :

$$P_m = \frac{fit_m(x_m)}{\sum_{m=1}^{SN} fit_m(x_m)} \qquad (7.5)$$

Après avoir choisi une source de nourriture x_m par une abeille observatrice, une nouvelle source dans son voisinage v_m peut être déterminée par l'équation (7.3), et sa valeur de fitness est calculée selon l'équation (7.4), une sélection glouton (greedy) est ensuite appliquée entre v_m et x_m.

Phase des abeilles exploratrices :
Les abeilles employées dont leurs solutions ne peuvent pas être améliorées pour un nombre d'itérations deviennent des exploratrices et leurs ressources seront abandonnées. Par exemple, si la solution x_m est abandonnée, la nouvelle solution peut être définie par (7.2). [D. Karaboga, 2010].

7.7.4 L'optimisation par mariage des abeilles de miel (MBO)

L'optimisation par mariage des abeilles de miel (MBO : Marriage in Honey Bees Optimization) est une nouvelle technique de l'intelligence en essaim inspirée du processus de mariage des abeilles du miel.

Le comportement naturel des abeilles de miel :

Structure de la colonie :
La colonie naturelle des abeilles de miel est composée : des reines, des bourdons (mâles), des ouvrières et des couvains. La source principale reproductrice de nouveaux individus est la reine. Les bourdons sont les individus haploïdes représentant les pères de la colonie. Les ouvrières sont consacrées au soin des couvains, mais peuvent parfois pondre des œufs. Les couvains proviennent des œufs fertilisés ou non fertilisés, les œufs fertilisés lors de la maturation deviennent des reines ou des ouvrières tandis que les œufs non fertilisés deviennent des bourdons [J.Teo et al, 2001].

Le processus de reproduction (le Mating –Flight) :
La reproduction commence par la danse des reines décolérant avec les bourdons compagnons. La reproduction a lieu dans le ciel où une reine joindra de 7 jusqu'à 20 bourdons pour chaque vol. Le sperme des différents bourdons sera déposé et accumulé dans le spermatheca des reines pour former le réservoir génétique des couvains. Pour chaque œuf fertilisé pondu par une reine, le

sperme correspondant est recherché de manière aléatoire dans son spermatheca [J.Teo et al, 2001].

Le modèle artificiel correspondant.

L'optimisation par mariage des abeilles de miel (MBO) est un algorithme très récent d'intelligence en essaim développé par Abbass [H.A. Abbass, 2001a], [H.A. Abbass, 2001b] pour résoudre des problèmes d'optimisation combinatoire. Cet algorithme est basé sur l'opération de reproduction génétique haploïde-diploïde des abeilles de miel.

Les principales étapes de cet algorithme sont [J.Teo et al, 2001]:

(1) le mating-flight des reines avec les bourdons
(2) la création de nouveaux couvains par les reines
(3) l'amélioration de la fitness des couvains par les ouvrières
(4) l'adaptation de la fitness des ouvrières
(5) remplacement des reines les moins convenables avec les couvains les plus convenables.

Les étapes détaillées de l'algorithme [J.Teo et al, 2001] **:**

Initialiser les ouvrières
Générer aléatoirement des reines
Appliquer une recherche locale pour obtenir la meilleure reine
Pour un nombre maximal de vols (flight) faire
Pour chaque reine de la liste faire
Initialiser l'énergie, la vitesse et la position correspondante
Tant que le spermatheca de la reine n'est pas encore plein et énergie > 0
La reine se déplace entre des états et choisit selon la valeur d'une probabilité des bourdons
Si un bourdon est sélectionné alors
Ajouter son sperme au spermatheca de la reine
Fin si
Mettre à jour l'énergie et la vitesse de la reine
Fin tant que
Fin pour chaque
Produire des couvains par croisement et mutation
Employer les ouvrières pour améliorer les couvains
Mettre à jour la fitness des ouvrières
Tant que le meilleur couvain est mieux que la plus mauvaise reine
Remplacer la mauvaise reine par le meilleur couvain
Enlever le meilleur couvain de la liste des couvains
Fin tant que
Tuer tous les couvains
Fin pour

Figure 7.2: L'optimisation par mariage des abeilles de miel (MBO)

Dans le modèle artificiel MBO, il y a cinq étapes principales : (1) le vol des reines avec des bourdons, (2) la création de nouveaux couvains par les reines, (3) l'amélioration de la fitness des couvains par les ouvrières, (4) l'adaptation de la fitness des ouvrières, et (5) le remplacement des mauvaises reines avec les meilleurs couvains [J.Teo et al, 2001].

L'algorithme commence par l'initialisation aléatoire d'un ensemble des ouvrières. Le génotype de la reine est initialisé au hasard et amélioré par une ouvrière choisie au hasard. Puis, un ensemble de vols est effectués par les reines. Un vol peut être traduit par un ensemble de transitions d'états dans un voisinage du problème d'optimisation. Au début du vol, la reine est initialisée avec une certaine valeur d'énergie et de vitesse, et sa trajectoire est initialisée de manière aléatoire, de nouvelles sous-trajectoires sont ensuite créées selon la valeur d'une probabilité dépendante de la vitesse de la reine et seront considérées en tant que bourdons potentiels. En d'autres termes, la probabilité d'accepter une nouvelle transition d'état est égale à 1. À chacune de ces trajectoires, la reine choisit un bourdon selon une probabilité dépendante de sa vitesse et la différence de fitness entre la reine et le bourdon [J.Teo et al, 2001] :

$$prob(Q,D) = e^{-\frac{\Delta(f)}{S(t)}} \qquad (7.6)$$

Où *prob* (*Q*, *D*) représente la probabilité d'un assemblage réussi entre reine et bourdon, c'est-à-dire la probabilité d'ajouter le sperme du bourdon D au spermatheca de la Reine Q. Δ(f) est la différence absolue entre la fitness du bourdon et la fitness de la reine, et S(t) la vitesse de la reine au temps *t*. Ainsi, la probabilité d'un assemblage réussi est élevée quand la fitness du bourdon est supérieure que la fitness de la reine ou encore quand la vitesse de la reine est élevée au début du vol. Après chaque transition d'état, l'énergie et la vitesse de la reine seront diminuées suivant les équations ci-dessous [J.Teo et al, 2001] :

$$E(t+1) = E(t) - \gamma \qquad (7.7)$$
$$S(t+1) = \alpha * S(t) \qquad (7.8)$$

Où α est un facteur entre 0 et 1 et γ la quantité réduite de l'énergie pour chaque transition. Le vol termine quand l'énergie de la reine est près du zéro ou quand son spermatheca est plein, dans ce cas, la reine revient au nid pour commencer la reproduction. Le sperme d'un bourdon est aléatoirement choisi de son spermatheca et un nouveau couvain est créé par croisement en utilisant le génotype haploïde du bourdon et sa complémentation par le génotype diploïde de la reine. La mutation est ensuite appliquée sur le nouveau couvain pour compléter le procédé de sa création. Les ouvrières sont habituées à améliorer les couvains. Les fitness respectives des ouvrières sont alors triées selon la qualité d'amélioration effectuée. La mauvaise reine est remplacée par le couvain le plus convenable jusqu'à ce qu'aucune des couvains ne soit meilleure qu'aucune de ces reines. Les couvains restants sont alors tués et un nouveau vol est initialisé. Ceci est répété jusqu'à ce que tous les vols d'assemblage soient accomplis ou jusqu'à ce que l'état d'arrêt soit atteint [J.Teo et al, 2001].

7.7.5 L'optimisation par colonie de fourmi

Un des modèles les plus réussis de l'intelligence en essaim est la classe des algorithmes d'optimisation combinatoire inspirée des fourmis appelée l'optimisation par colonie de fourmi (ACO). Elle s'est avérée efficace en résolvant des problèmes combinatoires classiques tels que le problème de voyageur de commerce [M.Dorigo et al, 1999]. Le routage des télécommunications [M.Dorigo et al, 1999] et les problèmes de routage de véhicules [L.M. Gambardella et al, 1999], [J.Teo et al, 2001].

L'optimisation par colonie de fourmis (ACO) est une technique d'optimisation de problèmes par recherche guidée dans l'espace des solutions. L'algorithme ACO est inspiré de la suivie des traces de phéromones du déplacement des fourmis. Lorsque les fourmis se déplacent dans certains chemins, elles laissent des traces de phéromone qui influencent alors le choix de ces chemins par les autres fourmis. Le dépôt de phéromone par les fourmis fonctionne comme une rétroaction positive qui renforce les meilleurs chemins [A. Malisia et al, 2006].
L'ACO est une méta-heuristique basée graphe où un nombre de fourmis traversent l'espace des solutions tout en accumulant des solutions partielles jusqu'à atteindre une solution complète. Le choix des solutions partielles dépend de la quantité en phéromone des chemins et d'une évaluation heuristique. À chaque étape de la construction d'une solution partielle, la fourmi « *k* »

choisit le prochain nœud basé sur une règle de choix probabiliste qui décide la probabilité avec laquelle cette fourmi choisira d'aller du nœud courant *i au* prochain nœud *j* [A. Malisia et al, 2006] :

$$P_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l=N_j^k}[\tau_{il}]^{\alpha}\ [\eta_{il}]^{\beta}} \quad si\ j\ \in\ N_i^k \qquad (7.9)$$

Où τ_{ij} est la quantité en phéromone de l'arc du nœud *i* au nœud *j*, N_i^k sont les nœuds voisins de la fourmi *k* si elle est sur le nœud *i*. Les nœuds voisins sont les nœuds qui n'ont pas été visités par la fourmi *k*. Les constantes α *et* β représentent l'influence de la quantité de phéromone et de l'heuristique respectivement. Les résultats expérimentaux recommandent d'initialiser α = 1 et β de 2 à 5 [A. Malisia et al, 2006].

η_{ij} est la désirabilité d'aller du nœud *i* au nœud *j*. la désirabilité est une mesure du coût pour augmenter la solution partielle courante. Une fois qu'une solution est construite, elle est évaluée et une quantité de la phéromone sera déposée relativement à sa qualité. Les fourmis déposent de la phéromone sur les arcs qu'elles visitent suivant la relation ci-dessous [A. Malisia et al, 2006] :

$$\tau_{ij}^{nouveau} = \tau_{ij}^{courant} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \qquad (7.10)$$

Où $\Delta\tau_{ij}^{k}$ est la quantité de phéromone de la fourmi *k à* ajouter à l'arc acheminant du nœud *i* au nœud *j*, et *m* le nombre de fourmis. La quantité ajoutée de phéromone est définie par [A. Malisia et al, 2006]:

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{C^k} & si\ l'arc\ est\ dans\ le\ chemin\ de\ la\ fourmi\ k \\ 0 & sinon \end{cases} \qquad (7.11)$$

C^k est le coût total de la solution d'un chemin. Tous les arcs du même chemin ont cette valeur du coût. L'évaporation de phéromone est également appliquée à tous ces arcs suivant la relation ci-dessous [A. Malisia et al, 2006]:

$$\tau_{ij} = (1-p)\tau_{ij} \quad 0 < p \leq 1 \qquad (7.12)$$

L'ACO peut être résumé dans les étapes suivantes [N. Monmarché, 2007] :

Placer aléatoirement les « m » fourmis sur les sommets du graphe
Pour chaque arc Initialiser la quantité en phéromone
Pour chaque itération faire
    Pour chaque fourmi k faire
        Pour chaque arc faire
        Construire un chemin S^k (itération) en tenant compte la quantité en
        Phéromone (τ_{ij}) et la désirabilité (η_{ij})
        Calculer le fitness de la solution S^k (itération)
    Fin pour
    Mettre à jour la meilleure solution trouvée
Pour chaque arc faire
    Mettre à jour la quantité de phéromone selon la formule (7.10)
Fin pour

Figure 7.3: L'optimisation par colonies de fourmis

Conclusion

7.8 Conclusion

La résolution collective de problèmes s'intéresse à la manière de distribuer un problème entre des agents simples capables d'achever collectivement une tâche complexe. Elle est ainsi décentralisée plutôt que centralisée, émergente plutôt que planifiée, et concurrente plutôt que séquentielle.

L'intelligence en essaim est un champ naissant de l'intelligence collective inspirée des modèles comportementaux des insectes sociaux. Les algorithmes les plus connus de l'intelligence en essaim incluent les algorithmes d'optimisation par colonie de fourmis (ACO), l'algorithme d'optimisation par essaims de particules (PSO) et les algorithmes d'abeilles.

Le PSO est une méthode coopérative basée sur l'interaction entre des individus simples appelés particules. Chaque particule se rappelle de sa meilleure position visitée et la meilleure position trouvée dans son voisinage. Chaque particule se déplace alors selon une trajectoire perturbée entre ses deux positions et sa vitesse.

L'algorithme d'abeilles (BA) est une métaheuristique qui imite le comportement de fourragement des abeilles de miel en recherchant les endroits de la nourriture appelés sites. Cet algorithme est basé sur une recherche par voisinage associée à une recherche aléatoire ; une proportion de la population d'abeilles recherche dans le voisinage des meilleurs sites trouvés et les autres abeilles recherchent de manière aléatoire d'un site à un autre.

L'algorithme ABC : la population dans l'algorithme ABC est organisée en trois groupes : les abeilles découvreuses, les observatrices et les exploratrices. Les découvreuses recherchent les sources de nourriture et reviennent avec danse autour de la ruche. Les observatrices choisissent les sources de nourriture selon les fitness des abeilles découvreuses. Les abeilles dont les sources de nourriture restent inchangées deviennent exploratrices pour rechercher de nouvelles sources de nourriture qui pourront être supérieures.

L'optimisation par mariage des abeilles de miel (MBO) est une technique d'intelligence en essaim inspirée du processus de mariage des abeilles du miel. La colonie des abeilles est composée : des reines, des bourdons (mâles), des ouvrières et des couvains. La source principale reproductrice de nouveaux individus est la reine. Les bourdons sont les individus haploïdes représentant les pères de la colonie. Les ouvrières sont consacrées au soin des couvains. La reine choisi un bourdon selon une probabilité dépendante de sa vitesse et la différence de fitness entre la reine et le bourdon. Produit des couvains par croisement et mutation, emploie les ouvrières pour améliorer les couvains et puis l'algorithme met à jour la fitness des ouvrières, remplace ensuite les mauvaises reines par les meilleurs couvains jusqu'à un nombre maximum d'itérations.

L'algorithme ACO est inspiré du suivie des traces de phéromones du déplacement des fourmis. Lorsque les fourmis se déplacent dans certains chemins, elles laissent des traces de phéromone qui influencent alors le choix de ces chemins par les autres fourmis. Le dépôt de phéromone par les fourmis fonctionne comme une rétroaction positive qui renforce les meilleurs chemins

Chapitre 8
LES RESEAUX DE NEURONES

8.1 Introduction

Un réseau de neurones est un outil puissant de la modélisation des relations complexes entre les données d'entrée et de sortie. Le développement de cette technologie a été motivé par l'ambition d'avoir un système artificiel capable d'accomplir des tâches "intelligentes" de manière semblable à celles qui sont exécutées par le cerveau humain. Un réseau de neurones ressemble au cerveau humain dans le sens où il apprend des connaissances par apprentissage, et ses connaissances sont stockées dans des connexions inter-neurones connues sous le nom « poids synaptiques ». L'avantage des réseaux de neurones se situe dans leur capacité d'apprendre à résoudre des problèmes complexes à partir la modélisation des exemples d'apprentissage réels [D. Xiao Ni, 2007].

Les réseaux de neurones, avec leur capacité remarquable de dériver la signification des données incompréhensibles ou imprécises, peuvent reconnaître et détecter des formes complexes qui sont difficiles à détecter par d'autres techniques. Un réseau de neurones entraîné peut être vu comme un "expert" capable de catégoriser l'information à analyser et de fournir des réponses à des nouvelles informations ou situations.
Les réseaux de neurones ont été largement utilisés pour la reconnaissance de caractères grâce à leur capacité de généralisation et ceci selon leurs différents types d'architecture tels que les travaux suivant avec les taux de reconnaissance correspondants : La reconnaissance de caractères manuscrits (les lettres latines) par le perceptron multi-couche [D.Singh et al, 2011].
La reconnaissance de chiffres par réseau convolutionnel (taux d'erreur = 0.27%) [D.C. Cireset al, 2011], LeNet5 sur la base MNIST (taux de reconnaissance = 0.95%) [D.Bouchain, 2007].
La reconnaissance de caractères par réseau convolutionnel basée sur la déformation élastique des caractères (un taux d'erreur = 0.4%) [P. Y. Simard et al, 2003]. La reconnaissance des chiffres Devanagari par un réseau de neurones à base radiale [P.Singh et al, 2011]. La reconnaissance de caractères par RBF basé sur la rétropropagation sélective comme méthode d'apprentissage [M.Taghi et al, 2009]. La reconnaissance du texte imprimé canadien par un RBF en utilisant les wavelet de Haar comme attributs caractéristiques (un taux de reconnaissance = 99.1%) [B. VijayKumar et al, 2004]. La reconnaissance des symboles manuscrits par un réseau de neurones récurrent en utilisant des attributs extraits des règles floues (taux de reconnaissance = 97.98%) [J.A. Fitzgerald et al, 2005]. La reconnaissance des caractères Tamil par SOM [J.Venkatesh, C et al, 2009]. La reconnaissance des caractères Tamil par SOM (taux de reconnaissance= 98.5%) [R.I.Gandhi et al, 2009]. Une étude comparative entre les RN feed forward, SVM et SOM sur les caractères Tamil. Les résultats ont montré la supériorité des SVM et SOM par rapport au RN feed forward [C.Sureshkumar et al, 2010]. La reconnaissance des lettres arabes par un réseau de neurones probabiliste en utilisant les moments de Hu comme attributs caractéristiques (taux de reconnaissance = 96%) [L.M.ALZoubaidy et al, 2006]. La reconnaissance de l'écriture en-ligne par réseaux TDNN [K.Jung at al, 2000], [N.P. Matic et al, 2002].
Ce chapitre présente quelques modèles d'architecture des réseaux de neurones artificiels et leur principe de fonctionnement notamment les réseaux de neurones convolutionnels, les cartes auto-organisatrices, les réseaux de neurones probabilistes, les réseaux de neurones récurrents et les réseaux de neurones temporels.

8.2 Définitions

Les réseaux de neurones sont des algorithmes d'apprentissage et d'optimisation basés sur des concepts inspirés du fonctionnement d'un cerveau humain. Ils se sont généralement constitué des cinq composants suivants :

1. Un graphe dirigé qui représente la topologie du réseau.
2. Une variable d'état associée à chaque nœud.
3. Un poids associé à chaque connexion.
4. Un biais associé à chaque nœud.
5. Une fonction de transfert pour chaque nœud qui détermine l'état d'un nœud en fonction des poids de ses liens entrants, son biais et les états des nœuds reliés à ce nœud. Cette fonction est habituellement une fonction sigmoïde [D. Montana et al, 1989].

Un réseau de neurones possède les avantages suivants :
• un apprentissage adaptatif : qui est sa capacité d'apprendre comment résoudre de nouvelles tâches basée sur l'expérience initiale ou les données d'apprentissage.
• l'auto-Organisation : qui est la capacité d'un réseau de neurones à créer sa propre architecture ou sa propre représentation d'information qu'il reçoit au cours de la phase d'apprentissage.
• exécution en temps réel : les calculs d'un réseau de neurones se font en temps réel comme ils peuvent être effectués en parallèle à l'aide des dispositifs particuliers [D. Xiao Ni, 2007].

Le réseau de neurones doit apprendre à travers des exemples connus pour savoir exécuter des tâches imperceptibles. Dans la pratique : Les réseaux de neurones ne sont pas une solution-miracle : le choix de la meilleure structure et taille peut avoir un impact énorme sur la solution. Les résultats fournis sont approximatifs, et peuvent diverger beaucoup dans certains cas.

8.2.1 Le neurone artificiel

Un neurone artificiel est une unité élémentaire qui reçoit un nombre d'entrées ou des sorties en provenance des autres neurones du réseau, un poids est associé à chacune de ces entrées qui représente la puissance de sa connexion avec un neurone.
La figure 8.1 montre le modèle du neurone artificiel.

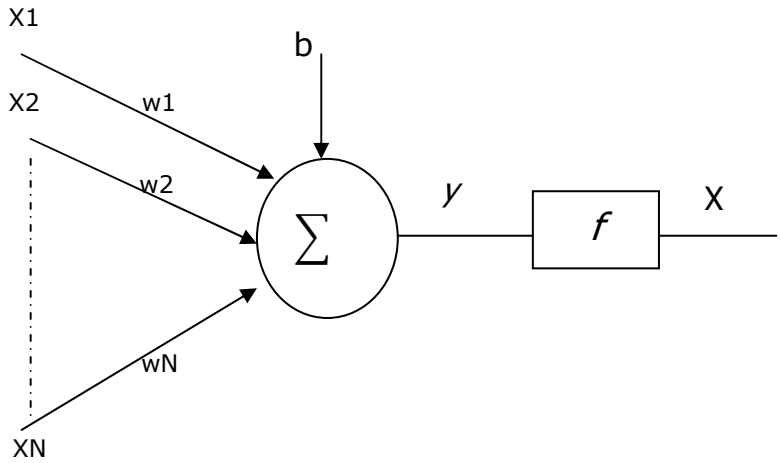


Figure 8.1: Le neurone artificiel

La sortie du neurone est une somme pondérée, de ses entrées plus un biais, propagée par une fonction d'activation qui peut être (sigmoïde, tangente ou hyperbolique, etc.) [F. Lauer et al, 2007].

$$y = \sum_{j=1}^{N} W_j \; x_j + \; b \qquad (8.1)$$

$$X = f(y) \qquad (8.2)$$

xj est la valeur de la jeme entrée du neurone, Wj est le poids synaptique correspondant au neurone j, b désigne le biais et f(.) la fonction d'activation. La fonction la plus utilisée est du type sigmoïde [J. Moody et al, 1989]. Elle est définie par :

$$f(y) = \frac{1}{1 + \; e^{-\delta y}} \qquad (8.3)$$

Où δ dénote le paramètre de la sigmoïde qui définit le degré de non-linéarité.

8.2.2 Topologies

Il existe de diverses topologies possibles pour un nombre spécifique de neurones. Certains sont des représentations réalistes des `réseaux de neurones biologiques, et les autres ont des avantages pour résoudre des problèmes spécifiques en informatique. Les réseaux Feed-forward sont simples, n'ayant pas de retour (connexion) en arrière alors que les réseaux Récurrent possèdent un comportement plus complexe, avec des boucles ou connexions bouclées.

8.3 Le perceptron multicouche(MLP)

Le perceptron multicouche (MLP) est l'exemple le plus répandu des classificateurs neuronaux employés dans le domaine de la reconnaissance. Un MLP est composé d'une couche d'entrée dont la taille est égale à la taille des données d'entrée, une ou plusieurs couches cachées dont la taille est déterminée par essai et la couche de sortie dont la taille est égale au nombre de classes du problème à résoudre (fig. 8.2).

Dans les réseaux MLP, chaque neurone est relié à un certain nombre d'entrées qui peuvent être les données d'entrée ou les sorties des couches précédentes (fig 8.1).
Chacune de ces entrées est pondérée par un poids synaptique ; le poids total d'un neurone est le produit scalaire entre les entrées et leurs poids correspondant avec addition d'un biais (b) :
Le Poids total = X_1W_1 + ……+ X_NW_N + b

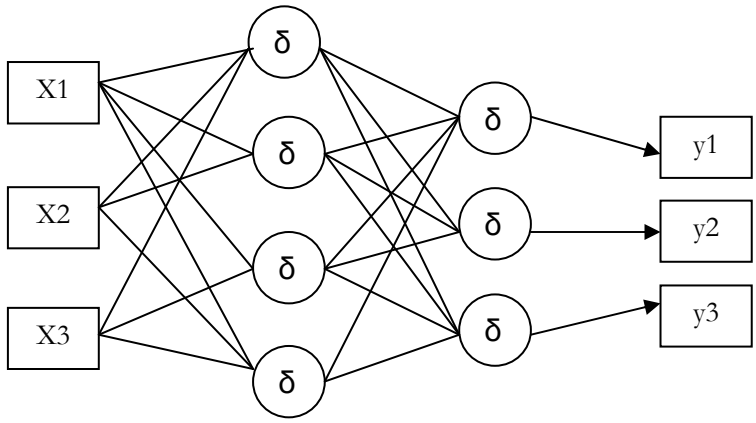


Figure8.2: Structure d'un MLP

Avec δ est une fonction d'activation employée pour propager l'information à la couche suivante, habituellement, la fonction sigmoïde qui renvoie un résultat "y" entre [0, 1].

Un algorithme d'apprentissage tel que la rétropropagation de gradient est appliqué pour ajuster les poids en réduisant au minimum une fonction d'erreur qui est, en général, la somme des erreurs quadratiques entre la sortie du réseau de neurones et la sortie attendue.

8.3.1 La rétropropagation par descente de gradient

Cet algorithme évalue l'erreur de gradient pour chaque neurone dans le réseau depuis la dernière couche jusqu'à la première. Les poids produisant une erreur significative seront modifiés en grande valeur par rapport aux poids produisant une erreur marginale. La rétropropagation par descente de gradient a été efficacement employée pour l'apprentissage d'un MLP ; son principe est de converger de manière itérative vers un ensemble optimal des poids synaptiques. Dans le cas de l'apprentissage supervisé, les sorties sont fournies à l'avance ; la différence entre ces sorties et les sorties du réseau (après propagation de l'information par une fonction d'activation) constitue l'erreur à réduire au minimum par rétropropagation tout en ajustant les poids [R.A.Rojas, 1996], [D.F.A.Santiago et al, 2002].
Cette étape est répétée plusieurs fois jusqu'à ce que le réseau puisse offrir la meilleure prévision. Dans certains cas l'algorithme de la rétropropagation de gradient ne peut pas échapper des optimums locaux, pour cette raison, un poids momentum a été ajouté à la formule de rétropropagation permettant à l'algorithme d'échapper aux optimums locaux.

Pour des raisons de simplicité, nous avons choisi de schématiser ce processus de recherche à travers la figure suivante qui résume les étapes principales de l'algorithme de la rétropropagation de gradient :

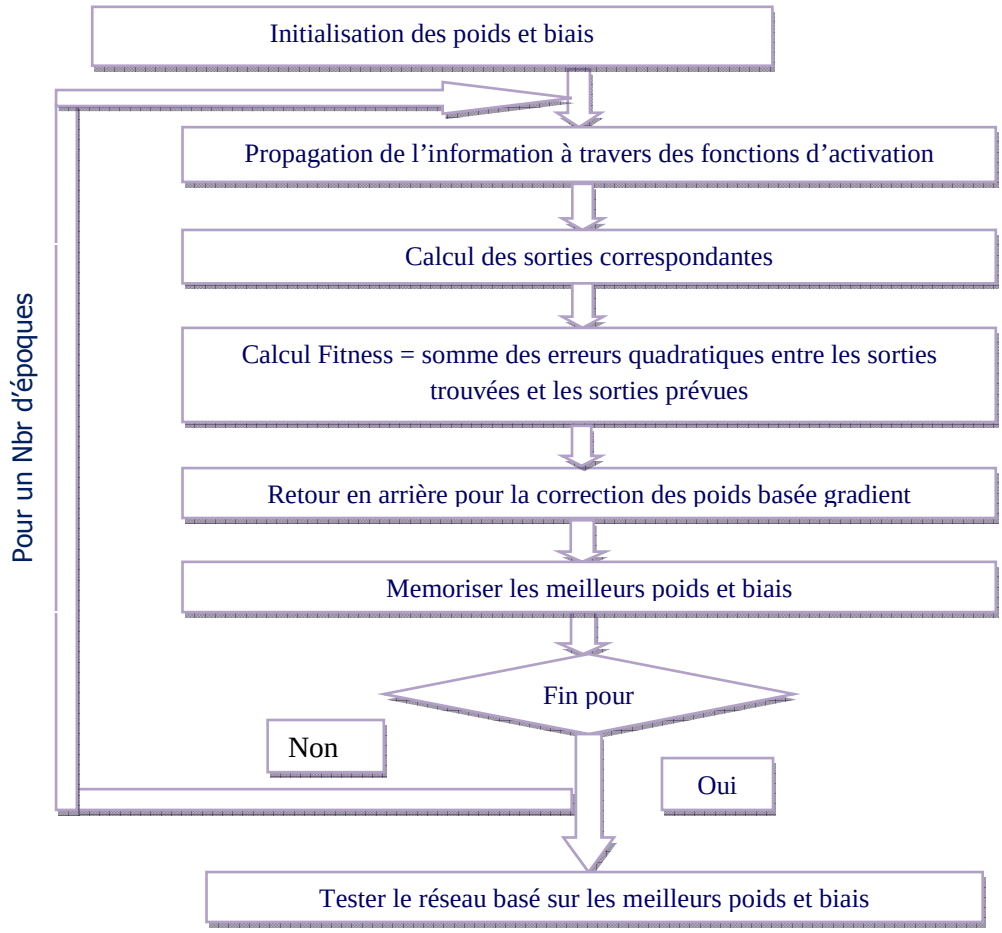


Figure8.3 : L'algorithme de la rétro-propagation de gradient

L'algorithme [R.A.Rojas, 1996]*:*

L'algorithme de rétropropagation par descente de gradient suit les étapes suivantes :

- De chaque neurone, l'information est propagée à la couche suivante en utilisant une fonction d'activation (habituellement une fonction sigmoïde) appliquée au produit scalaire des entrées et les poids correspondants :

$$net_j = b + \sum_k O_{pk}\left(w_{jk}\right) \qquad (5.22)$$

$$O_{pj}\left(net_j\right) = 1 \,/\, (1 + \ e\text{^}(-\lambda net_j)\,) \qquad (5.23)$$

Où :

j = le neurone courant

p = le modèle courant

W_{Jk} = le poids synaptique de l'entrée k au neurone "j"
O_{Pj} = la sortie du neurone "j" pour le modèle "p" employant une fonction d'activation sigmoïde
net_j = le poids total du neurone j

- A la fin de la propagation, la sortie du réseau est obtenue et l'erreur entre la sortie obtenue et la sortie attendue du sous-ensemble d'entrée est calculée.
- Pour chaque neurone "j" appartenant à la couche de sortie, l'erreur correspondante est calculée par la formule suivante :

$$Dpj = (Tpj - Opj)Opj\,(1 - Opj) \qquad (5.24)$$

T_{pj} = la sortie attendue du neurone j pour le modèle p
O_{pj} = la sortie obtenue du neurone j pour le modèle p

- Ces erreurs seront propagées en arrière vers la dernière couche cachée par l'équation suivante :

$$\delta_{pj} = 0_{pj}\left(1 - O_{pj}\right)\sum_{k}\delta_{pk}w_{kj} \qquad (5.25)$$

δ_{pk} = l'erreur d'un neurone de sortie k
W_{kj} = le poids du neurone caché "j" au neurone de sortie k

- Le poids est ainsi ajusté par :

$$\Delta w_{ji}(t) = \lambda\delta p_j O_{pi} \qquad (5.26)$$

λ est le taux d'apprentissage choisi entre]0, 1[.
- ajustement des poids selon :

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}(t) \qquad (5.27)$$

8.3.1.1 Les modes d'apprentissage par rétropropagation de gradient

Il existe trois principaux modes d'apprentissage par descente de gradient, selon la manière dans laquelle les poids synaptiques sont mis à jour [A. Abraham, 2004]:

- Le premier est l'apprentissage **"en ligne" :** il consiste à mettre à jour les poids synaptiques après chaque présentation d'un caractère au hasard. Dans ce cas, le gradient instantané de la fonction de performance est employé pour l'adaptation des poids. L'apprentissage en ligne rend la recherche stochastique, il est donc moins probable de tomber dans des minima locaux.

- Le deuxième mode dit **"hors ligne"** consiste à accumuler les gradients consécutifs et puis de mettre à jour les poids après que tous les caractères d'apprentissage sont présentés.
- Le troisième mode d'apprentissage connu **"semi-ligne"** semble être comme solution de compromis entre les deux modes précédents en ligne et hors-ligne. Il consiste à présenter consécutivement les exemples de chaque classe, pour accumuler les gradients instantanés, et puis de mettre à jour les poids. Ce procédé est ainsi répété pour d'autres exemples d'une autre classe et ainsi de suite.

Des expérimentations sur le perceptron multicouche pour la classification de 26 lettres latines ont montré que le mode en ligne et le mode **"semi-ligne"** sont plus efficaces que le mode hors-ligne [B. Gosselin, 1996].

8.3.1.2 Adaptations de la méthode de rétropropagation de gradient :

Il existe d'autres modifications de l'algorithme de la rétropropagation de gradient qui ont été proposées pour accélérer l'apprentissage du perceptron multicouche telles que les méthodes de la régularisation ou de la minimisation comme *newton* et *quasi-newton*.

Les méthodes de la régularisation contrôlent la valeur des poids pendant l'apprentissage. Plusieurs méthodes de régularisation ont été employées comme le "*early stopping"* qui consiste à stopper l'apprentissage avant la convergence ou les méthodes de pénalisation qui ajoutent un terme additionnel à la fonction de performance [W. Gerstner, 1988]. Ce terme est une fonction de pénalisation qui est souvent la fonction *"weight decay"* [P. Gallinari et al, 1999].

Quand les poids du réseau sont devenus très grands (en valeur absolue), le sigmoïde des neurones cachés est ainsi saturé et les fonctions à optimiser peuvent avoir des variations brusques. Pour obtenir des fonctions régulières, il est nécessaire de contrôler la partie linéaire de la fonction sigmoïde pour obtenir de petites valeurs de poids.

La méthode de régularisation de poids *" weight decay ou technique* de la dégradation de poids *"* limite *la* valeur absolue des poids en utilisant le terme de pénalité suivant [Y. Jin et al, 2004], [A. Vesely, 2003] :

$\Omega = \frac{\alpha}{2}\sum_{i=1}^{P} w_i^2$

L'apprentissage est exécuté par minimisation de :

la nouvelle fonction $= ancienne\ fonction + \frac{\alpha}{2}\sum_{i=1}^{P} w_i^2$

Où : *p est* le nombre de poids w_i. "α" est un paramètre qui détermine l'importance relative des deux termes dans la nouvelle fonction de performance. Si "α" est trop grand, les poids tendent vers zéro, si "α" est à très petit, le terme de la régularisation perd son importance et le réseau pourra être saturé. Dans le cas intermédiaire, les poids après l'apprentissage posséderont des valeurs modérées.

Le principe des méthodes de minimisation est de réitérer une procédure de minimisation jusqu'à une satisfaction d'un certain critère. Ces algorithmes sont plus puissants que les méthodes simples de gradient.

La méthode *newton* emploie la dérivée seconde de la fonction de performance pour atteindre un minimum. La direction de la descente est l'inverse hessienne de la fonction de performance. C'est un algorithme qui est inefficace loin du minimum et très efficace près du minimum.
La méthode *quasi-newton* consiste à approximer l'inverse hessienne plutôt qu'à calculer sa valeur exacte. Cette méthode a une plus grande vitesse de convergence par rapport à la méthode de gradient.
La méthode *Levenberg-Marquardt* (LM) est l'un des algorithmes quasi-newton indiquée pour l'optimisation de l'erreur quadratique. Le LM est rapide et assure de meilleure performance pour les problèmes d'approximation de fonction où le nombre de poids est inférieur à 100 autrement ses performances diminues. Le LM est un mauvais choix pour les problèmes de classification [P. Wu et al, 2006].
L'algorithme de la rétropropagation assure la meilleure performance pour les problèmes de classification et n'exige pas de grand espace dans la mémoire de stockage, mais il n'est pas un bon choix pour les problèmes d'approximation de fonction.
Les algorithmes de gradient conjugué, en particulier le SCG (Scaled Conjugate Gradient) est un algorithme puissant pour un grand nombre de problèmes et plus particulièrement pour les réseaux de neurones de grande taille. L'algorithme SCG est aussi rapide que Levenberg-Marquardt dans les problèmes d'approximation de fonction et aussi rapide que l'algorithme de la rétropropagation pour les problèmes de la classification. En outre, le SCG (traingdx en matlab) n'exige pas de grand espace dans la mémoire de stockage [T. Gagnon et al, 2003], [A. Abraham et al, 2001].

8.4 Le réseau de neurones convolutionel

Le perceptron multicouche est bien adapté aux problèmes de classification mais présente des inconvénients quand il est appliqué aux problèmes plus complexes, dans un MLP, le nombre de paramètres d'apprentissage devient extrêmement large quand la taille ou le nombre des données d'apprentissage est augmenté. De plus, un MLP n'est pas invariant au décalage des formes à reconnaître, à leur facteur d'échelle et autres aspects de déformation, aussi, la topologie des données d'entrée est complètement ignorée, menant de ce fait à des résultats similaires pour toutes les déformations du vecteur d'entrée. Pour surmonter ce type de problème, Yann LeCun et Yoshua Bengio ont introduit le concept des réseaux de neurones convolutionels, ils ont conçu un réseau de neurones capable d'extraire d'une façon implicite les attributs appropriés, tout en limitant les poids neuronaux d'une couche à un champ réceptif local de la couche précédente. Ainsi, une carte d'attributs est obtenue en deuxième couche. En réduisant la résolution spatiale de la carte des attributs, un certain degré d'invariance au décalage et aux déformations est achevé aussi le nombre de paramètres est considérablement diminué en employant les mêmes poids pour tous les attributs d'une carte d'attributs [Y. LeCun et al, 1998b].

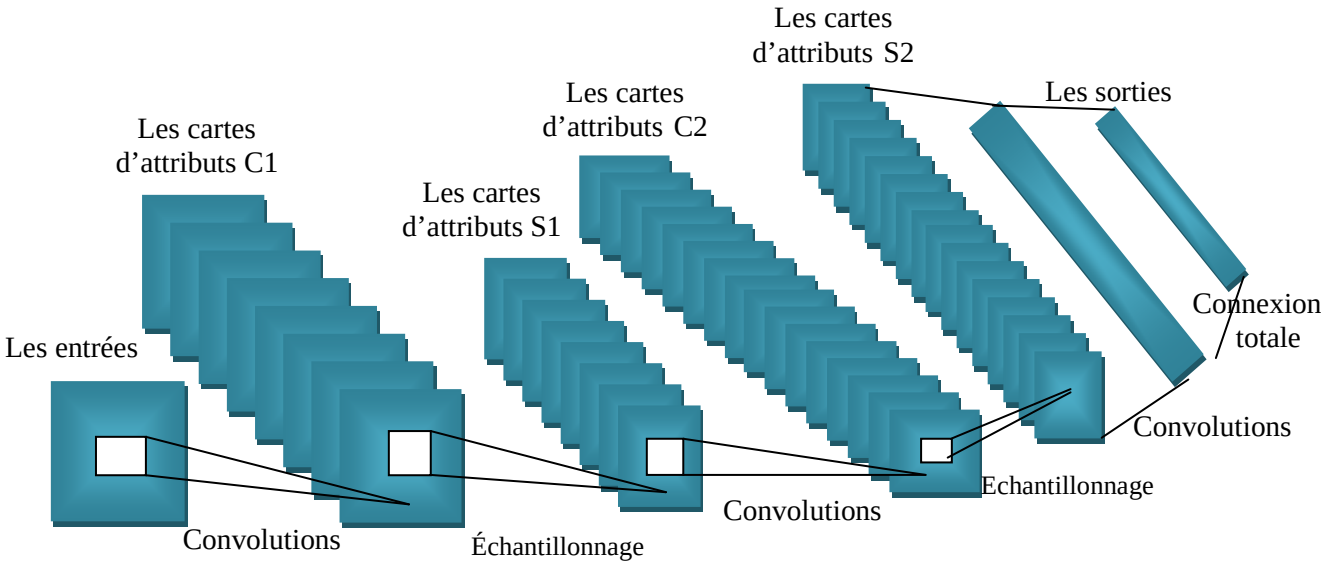


Figure8.4 : architecture d'un réseau de neurones convolutionnel [X. Yao, 1999]

Le CNN est un réseau feed-forward capable d'extraire les attributs topologiques d'une image. Sa première couche cachée est utilisée pour extraire les attributs caractéristiques de l'image d'entrée et sa dernière couche est utilisée pour la classification. Les deux premières couches du réseau peuvent être considérées comme des extracteurs d'attributs caractéristiques [D. Montana et al, 1989]. Le nombre de neurones cachés est variable à partir de lequel nous pouvons contrôler la capacité d'apprentissage et la capacité de généralisation du réseau *CNN [*C.Enăchescu et al**,** 2010].

Les unités d'une couche sont organisées dans des surfaces dans lesquelles toutes les unités partagent le même ensemble de poids. L'ensemble des sorties des unités d'une telle surface s'appelle une carte d'attributs. Les unités d'une carte d'attributs effectuent la même opération sur les différentes parties de l'image. Une couche convolutionnel se compose de plusieurs cartes d'attributs (avec différents vecteurs de poids) [Y. Lecun et al, 1998b]. Les sorties des unités cachées entrent dans un plus petit ensemble d'unités cachées appelées couches du sous-échantillonnage.

En général, les couches convolutionnels sont entrelacées avec les couches du sous-échantillonnage pour réduire le temps de calcul et pour permettre d'une façon graduelle l'invariance spatiale [J. Bouvrie, 2006], [Y. Lecun et al, 2010].

De ce fait, les réseaux convolutionnels combinent des idées architecturales pour assurer un certain degré d'invariance au décalage et au facteur d'échelle qui sont [Y. Lecun et al, 1998b] :
1) les champs réceptifs locaux.
2) le partage des poids.
3) le sous-échantillonnage spatial ou temporel.

A l'aide des champs réceptifs locaux les neurones peuvent extraire les attributs visuels élémentaires tels que les contours, les points finaux et les coins. Ces attributs sont ensuite combinés par les couches suivantes afin de détecter des attributs d'ordre supérieur. Les détecteurs élémentaires d'attributs importants sur un endroit de l'image sont susceptibles d'être utiles à travers l'image entière. Cette connaissance peut être appliquée en affectant des poids identiques à l'ensemble des unités ayant des champs réceptifs situés à différents endroits de l'image [D.E.Rumelhart et al, 1986], [Y. Lecun et al, 1998b].

La stratégie générale d'*un CNN* est d'extraire des attributs simples à une plus haute résolution, et puis de les convertir en attributs complexes à une résolution élémentaire. Chaque couche convolutionnelle peut réduire la résolution initiale de l'image d'un facteur (*n* - 3)/2 *:* un neurone d'une couche convolutionnelle traite une seule région de la carte issue d'une couche précédente. Ainsi, un neurone fonctionne comme un noyau indépendamment des autres neurones, appelé le *noyeau convolutionnel [*P.Y. Simard et al, 2003]. Ce noyeau contient comme entrées la région correspondante de la carte précédente [C.Enăchescu et al**,** 2010].

Un réseau convolutionnel typique a été conçu pour la reconnaissance de chiffres manuscrits, appelé LeNet-5. LeNet5 traite une image d'entrée de 32 * 32 Pixels. Il se compose de 7 couches : trois couches convolutionnelles (C1, C3 et C5), deux couches du sous-échantillonnage (S2 et S4), une couche totalement connectée (F6) et la couche de sortie. Les couches convolutionnelles et celles du sous-échantillonnage sont entrelacées**.** La première couche est une couche convolutionnelle (C1) composée de 6 cartes d'attributs de 28 * 28 unités. La couche suivante du sous-échantillonnage (S2) réduit en facteur 2 la résolution, alors que la couche convolutionnelle suivante (C3) augmente le nombre de cartes des attributs à 16. Chaque unité de C3 est reliée à plusieurs champs réceptifs dans des endroits identiques du sous-ensemble des cartes d'attributs de S2. Ces combinaisons sont arbitraires et forcent les cartes d'attributs d'extraire de différents attributs, car elles obtiennent de différentes entrées. La couche S4 du sous-échantillonnage agit comme S2 et réduit la taille des cartes d'attributs à 5 * 5. La dernière couche convolutionnelle C5 est différente de C3. Chacune de ses 120 cartes d'attributs est connectée à un champ réceptif de toutes les cartes d'attributs de S4. La taille des cartes d'attributs de C5 est 1*1 puisque les cartes d'attributs de S4 sont de taille 5 * 5. Ainsi C5 est une couche totalement connectée. Elle est aussi considérée comme une couche convolutionnelle parce que si l'image d'entrée est plus grande, la dimension des cartes d'attributs serait plus grande. La couche (F6) contient 84 unités reliées aux 120 unités de C5 [F. Lauer et al, 2007].

Il a été montré dans [Y.LeCun et al, 1998], que LeNet-5 non seulement fournit une meilleure exactitude par rapport à d'autres techniques, mais également fournit une meilleure classification avec grande vitesse. En plus, LeNet-5 est plus résistant au bruit et invariant aux déformations, les réseaux de neurones convolutionnels s'avèrent le meilleur choix pour la reconnaissance de chiffres manuscrits.

8.4.1 L'apprentissage d'un CNN

Comme une forme particulière du perceptron multicouche, un réseau *CNN* peut être entraîné par rétropropagation [Y. Lecun et al, 1998b] et comme tous les neurones d'une carte d'attributs de la même couche partagent les mêmes poids et biais, le nombre de paramètres est considérablement réduit par rapport à ceux d'un perceptron multicouche totalement connecté.
Les couches sous-échantillonnage ont un seul poids et un seul biais [Y. Lecun et al, 1998b], ainsi le nombre de paramètres dans les couches sous-échantillonnage est inférieur que celui des couches convolutionnelles. L'entraînement des réseaux convolutionnels ne demande pas beaucoup du temps comme les perceptrons multicouche puisque le nombre de paramètres est considérablement réduit, aussi bien que, l'invariance aux déformations est augmentée à un certain degré, font des réseaux de neurones convolutionnels un bon candidat pour les tâches de classification et plus particulièrement la reconnaissance de caractères manuscrits [Y. Lecun et al, 1998], [D. Bouchain, 2007], [Y. LeCun et al, 1998b].

Une réduction de la résolution spatiale des cartes d'attributs est effectuée par les couches du sous-échantillonnage. Une telle couche comporte le même nombre de cartes d'attributs de la couche convolutionnelle précédente, mais avec la moitié du nombre de lignes et de colonnes. Chaque unité *j* d'une certaine couche convolutionnelle est connectée à un champ réceptif 2*2, calcule la moyenne de ses quatre entrées *yi* (qui sont les sorties de la carte d'attributs correspondante de la couche précédente), la multiplie par un poids *wj* et ajoute un biais *bj* pour obtenir le niveau d'activité *vj* [F. Lauer et al, 2007]:

$$v_j = w_j \frac{\sum_{i=1}^{4} y_i}{4} + b_j \qquad (8.4)$$

Toutes les unités de toutes les couches jusqu'à F6 ont une fonction d'activation sigmoïde φ du type :

$$y_j = \varphi(y_j) = A\, tanh\,(S\, v_j) \qquad (8.5)$$

Où *vj* est le niveau d'activité de l'unité. A et S sont deux paramètres constants de la fonction sigmoïde.
La couche de sortie est une couche RBF euclidienne de 10 unités (pour les 10 classes de chiffres : 0 à 9) dont les sorties *yj* sont calculées par [F. Lauer et al, 2007]:

$$y_j = \sum_{i=1}^{84} (y_i - w_{ij})^2 \qquad j = 0, \ldots\ldots, 9 \qquad (8.6)$$

Où *yi* est la sortie de l'i ^{ème} unité de la couche F6. Pour chaque neurone RBF, *yj* est un terme de pénalité mesurant la fitness de ses entrées *yi* par rapport à leurs paramètres *Wij*. Ces paramètres sont fixés et initialisés à - 1 ou +1 pour représenter des images binaires (caractères) de taille 7 * 12 qui sont les sorties de la couche précédente (par conséquent la taille de la couche F6 est 7 * 12 = 84). La sortie minimale donne la classe du modèle d'entrée [F. Lauer et al, 2007].

8.5 Le réseau de neurones de fonctions à base radial

Le réseau de fonctions à base radial (RBF) est un réseau de neurones feed-forward composé d'une couche d'entrée, une seule couche cachée composée de centres radiaux et une couche de sortie. Le réseau RBF est conçu pour effectuer une combinaison non-linéaire de l'espace d'entrée à l'espace caché, suivi d'une combinaison linéaire de l'espace caché à l'espace de sortie [D. Dumitrescu et al, 2005].

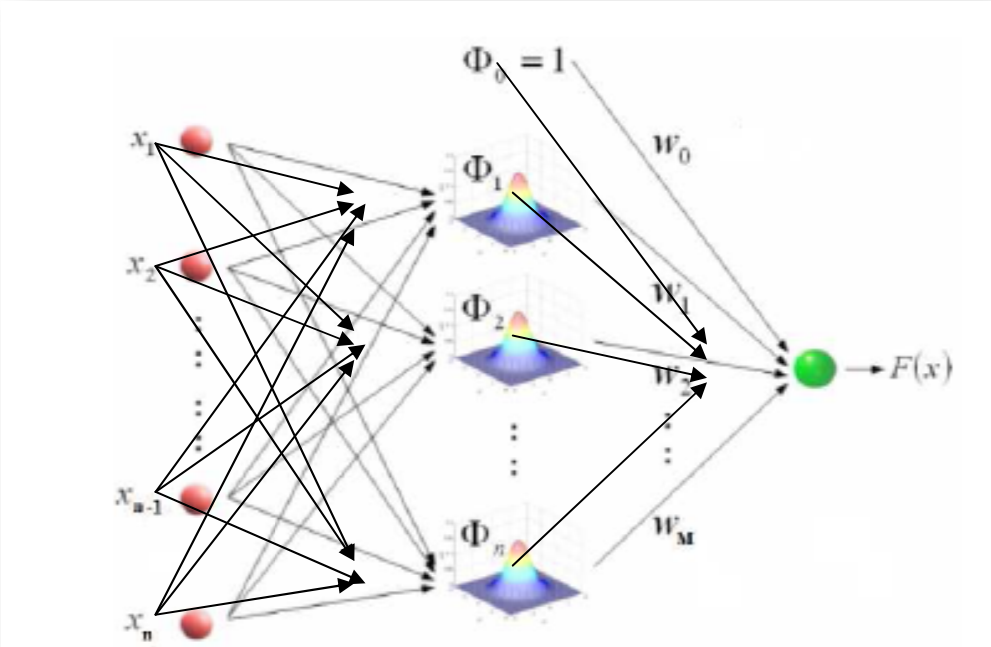


Figure 8.5: structure d'un RBF

Les réseaux (RBF) peuvent être employés pour un grand nombre d'applications parce qu'ils peuvent optimiser n'importe quelle fonction régulière et leur apprentissage est plus rapide qu'un perceptron multicouche. Cette vitesse vient du fait qu'un RBF a seulement deux couches de poids qui peuvent être évalués d'une façon déterministe.

La performance d'un réseau RBF dépend du nombre et position des fonctions à base radiales, de leur forme et de la méthode employée pour son apprentissage [C. Wu et al, 2003].
Les stratégies de sélection des centres d'un RBF peuvent être classifiées comme suit :

- les stratégies choisissant de manière aléatoire les centres RBF à partir les exemples d'apprentissage
- les stratégies non supervisées comme les algorithmes génétiques [B.A. Whitcheas et al, 1994],
- Les stratégies supervisées comme le calcul des moindres carrés [J.B. Gomm et al, 2000], affectation de ressources [J.Platt, 1991] et la descente de gradient [E.S. Chan et al, 1996] ou ceux qui emploient des algorithmes du clustering comme K-means et les cartes auto-organisatrices [Y. Hwang et al, 1997].

Autre méthode heuristique sur l'ensemble d'apprentissage qui commence à partir d'un sous-ensemble vide des fonctions à base radiale (les centres de la couche cachée) avec des exemples tirés de la base d'apprentissage. Pour chaque étape, l'ensemble vide s'accroît en incluant les fonctions permettant de diminuer la valeur de la fonction coût sur l'ensemble d'apprentissage, telles que, la somme des erreurs quadratiques. De cette manière, un réseau RBF est construit où la construction continue jusqu'à ce que la qualité du critère est réduit au minimum [P.Venkatesan et al, 2006].

Une propriété de la méthode ci-dessus est qu'elle mène souvent à des solutions sous optimales. Pour surmonter ce problème, la complexité du réseau est souvent réduite par des méthodes d'approximation [T. Poggio et al, 1990]. [S. Song et al, 2005] ou des méthodes du clustering en groupant les données d'apprentissage par des techniques du clustering telle que *k* –means [S. Z. Selim et al, 1986] ou des techniques plus efficaces permettant de spécifier d'une façon dynamique la topologie optimale d'un réseau RBF telle que les algorithmes évolutionnaires [K. Simon, 2003], [D. Dumitrescu et al, 2005] ou encore l'optimisation de l'ensemble des exemples d'apprentissage par les algorithmes génétiques (GA) [R. G. Kamp et al, 2006].

Apprentissage d'un RBF

Il existe deux méthodes communes pour déterminer les poids de la couche de sortie. La méthode de régularisation [S. Haykin, 1994] et la méthode de la descente de gradient [J. Moody et al, 1989], [Y. Hwang et al, 1997].

La méthode de la descente de gradient :

Soit x = (x_1....., x_n) un vecteur d'entrée, g_i () une fonction d'activation et w_i le poids synaptique correspondant au i^{ème} neurone caché, la sortie du réseau RBF est ainsi calculée par l'équation suivante [D. Dumitrescu et al, 2005]:

$$y = \sum_{i=1}^{K} w_i \, g_i(x) \qquad (8.7)$$

La fonction gaussienne est souvent employée comme fonctions d'activation c-à-d la fonction à base radial (RBF) [D. Dumitrescu et al, 2005] :

$$g_i\,(x) = e^{-\frac{\|x - c_i\|^2}{2\sigma_i}} \qquad (8.8)$$

Où c_i est le centre et σ_i la largeur du centre *de* la fonction correspondante au neurone *i*. La couche cachée d'un réseau RBF peut être entrainé par un algorithme d'apprentissage supervisé.
Pour trouver les meilleurs poids synaptiques du réseau, un algorithme de la descente de gradient est considéré.
Soit :

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \quad i = 1, \ldots\ldots, K \qquad (8.9)$$

Où η est le *taux d'apprentissage* et E *l'erreur totale*, l'erreur totale est calculée selon la formule ci-dessous [D. Dumitrescu et al, 2005]:

$$E = \frac{1}{N}\sum_{i=1}^{N}(z_i - y_i)^2 \qquad (8.10)$$

Où N *est* le nombre des exemples d'apprentissage, z_i est la sortie prévue, y_i est la sortie trouvée par le réseau. Les poids du réseau sont modifiés par la règle suivante [D. Dumitrescu et al, 2005]:

$$w_i = w_i + \Delta w_i, \qquad i = 1, \ldots\ldots, K \qquad (8.11)$$

8.6 Comparaison MLP et RBF

La différence principale d'un RBF par rapport à un MLP est l'absence des poids de la -couche cachée. Aussi, les sorties cachées ne sont pas calculées en utilisant une fonction d'activation sigmoïde, elles se sont basées sur des fonctions à base radiale. L'apprentissage d'un réseau RBF est radicalement différent de l'apprentissage classique d'un réseau MLP. Dans ce cas, il n'y a aucun poids caché à changer par la méthode du gradient. Dans les réseaux RBF, l'apprentissage est basé sur le choix des centres et leurs largeurs aussi bien que les poids des neurones de sortie.

Table 8.1: MLP versus RBF

	Le perceptron multi couche	**Les réseaux de neurones à base radiale**
Les unités cachées	Fonctions hyper-plan	Fonctions hyper-sphère
Les couches cachées	Plusieurs	Une seule couche cachée
apprentissage	supervisé	non-supervisé + combinaison linéaire
	lent	rapide

Les réseaux de neurones du type MLP et RBF sont capables de converger avec n'importe quel type de problème ; la principale différence porte sur l'architecture qui consiste de deux couches seulement pour les RBF et peut comporter plus pour les MLP. Le réseau RBF soufre du problème d'identification de ces paramètres de contrôle qui sont nombreux notamment, la détermination des centres des fonctions radiales. Leurs rayons (largeurs) et les poids de la couche de sortie. Si l'un de ces paramètres n'est pas identifié précisément, Le réseau RBF peut produire des résultats néfastes. En outre, la taille du réseau RBF augmente considérablement avec le nombre des exemples d'apprentissage, augmentant ainsi le temps de calcul. Cependant un réseau MLP avec Une seule couche cachée peut avoir une grande capacité de généralisation sur un grand nombre d'exemples d'apprentissage.

8.7 Les cartes auto organisatrices

Les cartes auto-organisatrices (SOMs) sont des réseaux de neurones populaires basés sur un apprentissage non supervisé ayant les propriétés de la quantification de vecteur et de la projection de vecteur. Les SOMs, à travers un processus auto-organisateur, configure les unités de sortie dans une représentation topologique des données originales, positionnant les vecteurs prototypes sur une grille bi-dimensionnelle, faisant du SOM un outil puissant de la visualisation. Les neurones sont connectés aux neurones adjacents par des relations de voisinage. Pour chaque neurone « i » est associé un vecteur prototype d-dimensionnel (codebook) : wi= [wi1, wi2. . . , wid]. La dimension d est identique à la dimension du vecteur d'entrée [D. Moreno et al, 2006]. La carte est habituellement une grille bidimensionnelle avec une certaine organisation pouvant être appliquée (comme la structure hexagonale).

8.7.1 Apprentissage des cartes auto-organisatrices

L'apprentissage d'une carte auto-organisatrice suit les étapes suivantes :
• Choisir aléatoirement un vecteur d'apprentissage $x = (x1, x2, \dots, xn)$
• Trouver la meilleure unité de sortie «w_c » par une métrique de distance appropriée
• Modifier tous les poids de toutes les unités de « i » en utilisant la règle suivante [S. K. Mostefaoui et al, 2003] :

$$wi(t + 1) = wi(t) + \alpha(t)[x(t) - wi(t)] \qquad (8.12)$$

Où « t » dénote le nombre d'itérations, α est le taux d'apprentissage.

La carte auto-organisatrice de Kohonen

Chaque unité « i » *a* son propre vecteur prototype w_i (vecteur de poids) qui représente une mémoire locale d'un vecteur d'entrée présenté au système. Les vecteurs prototypes sont initialisés aléatoirement avec une dimension « n » de l'espace d'entrée. Lorsqu'une nouvelle entrée est présentée au système, ses vecteurs prototypes sont modifiés selon l'équation suivante [S. K. Mostefaoui et al, 2003] :

$$w_i = w_i + \alpha.\eta\,(gagnant).(x_i - w_i), \qquad \forall\, i \in \{1, \dots, n\} \qquad (8.13)$$

Où α le taux d'apprentissage, η (*gagnant*) est une fonction de voisinage qui a ses valeurs entre 0 et 1, le *gagnant* est l'unité de la carte qui a le vecteur prototype le plus proche du vecteur d'entrée courant selon la valeur de la distance euclidienne entre l'unité courante et le gagnant [S. K. Mostefaoui et al, 2003] :

$$gagnant = \arg\, min_j \sqrt{\sum_{j=1}^{n} (x_j - w_j)^2} \qquad (8.14)$$

La fonction de voisinage est généralement une fonction gaussienne [S. K. Mostefaoui et al, 2003] :

$$\eta(gagnant) = \frac{1}{\sqrt{2\pi}\, nb}\; e^{-0.5\,\frac{(gagnant\,-courant\,)^2}{nb^2}} \qquad (8.15)$$

Où nb est un paramètre indiquant la largeur de la fonction gaussienne c-à-d la taille du voisinage (radius) dans lequel les unités voisines de l'unité gagnante peuvent mettre à jour leurs prototypes de manière significative. Après avoir présenté une quantité suffisante des données d'entrée à la carte SOM, l'auto-organisation aura comme conséquence une carte topographique, où les données similaires sont placées sur les unités d'une région particulière de la carte, et les unités voisines seront activées (c à d. elles deviennent gagnantes) pour les données d'entrée qui sont similaires [S. K. Mostefaoui et al, 2003].

8.8 Le Réseau de neurones probabiliste (PNN)

Le réseau de neurones probabiliste introduit par Donald Specht en 1988, est un réseau feed-forward à 3 couches utilisé pour la classification des données [D. F. Specht, 1990]. À l'inverse des autres réseaux de neurones qui sont basés sur la rétro-propagation, le réseau de neurones probabiliste est basé sur des principes statistiques dérivés de la stratégie de décision de Bayes et des fonctions estimateurs de la densité de probabilité.

Le PNN emploie des fonctions gaussiennes à base radiale et sphériques centrées à chaque vecteur d'apprentissage. La probabilité d'appartenance d'un vecteur à une certaine classe peut être exprimée comme suit [W. Tomasz et al, 1997], [M. Berthold et al, 1998]:

$$f_i\,(x) \;=\; \frac{1}{2\pi^{P/2}\;\sigma^{P}\;M_i}\;\sum_{j=1}^{M} exp\;\frac{-\,(X\,-\,X_{ij})^{T}\;(X-\,X_{ij})}{2\,\sigma^{2}} \qquad (8.16)$$

Où i est le nombre de classes, j est le nombre des formes à reconnaitre , x_{ij} est le j ème vecteur d'apprentissage de la classe i, x est un vecteur de test, M_i est le nombre de vecteurs d'apprentissage de la classe i, p est la dimension du vecteur x, σ est le facteur de lissage (l'écart type) et $f_i(x)$ est la somme des Gaussiens sphériques multi-variables centré à chacun des vecteurs d'apprentissage x_{ij} utilisée pour l'évaluation de la fonction de densité de probabilité de la classe i. Les décisions de classification sont prises selon la règle de décision de Bayes [W. Tomasz et al, 1997]:

$d(\boldsymbol{x}) = Ci$, si: $f_i\;(x) > \;f_k\,(x) \quad pour\; k \;\neq\; i$
Avec Ci est la classe de i.

8.9 Le réseau de neurone récurrent (RNN)

Les réseaux de neurones récurrents (RNNs) constituent une classe des réseaux de neurones où les connexions entre les unités peuvent former des cycles dirigés. Ceci permet de créer un état interne du réseau lui permettant un comportement temporel et dynamique. Inversement aux réseaux de neurones feed-forward, Les réseaux de neurones récurrents emploient une mémoire interne pour traiter des séquences arbitraires des entrées. Ceci les rend applicables aux tâches difficiles comme la reconnaissance de l'écriture manuscrite non segmentée, où les RNNs ont achevés de très bons résultats [H.T. Siegelmann, 1993]. Il existe plusieurs types de réseaux de neurones récurrents tels que le Perceptron Multicouche Récurrent, le réseau de Hopfield et le réseau récurrent simple" (SRN) d'Elman et Jordan.

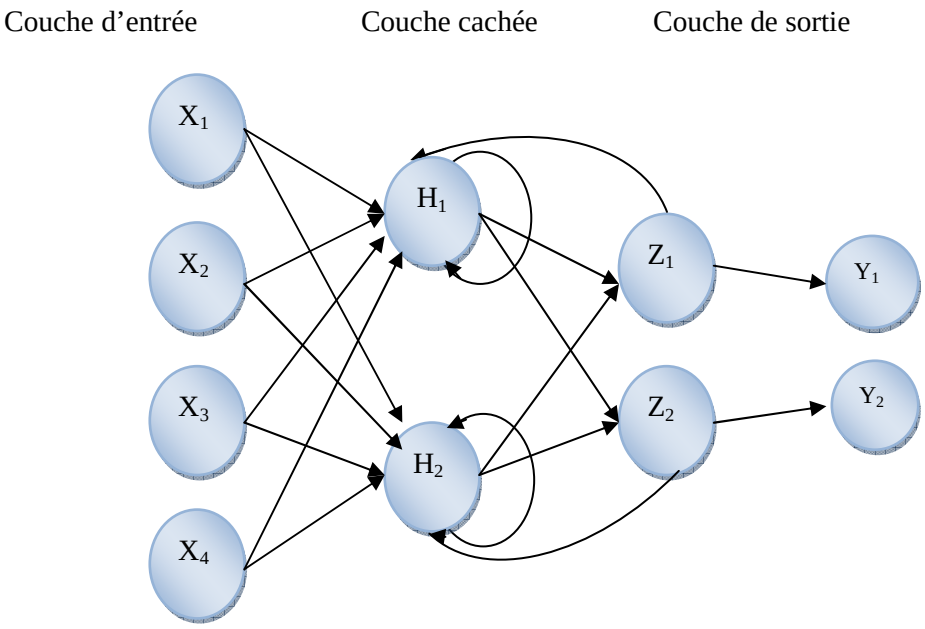


Figure 8.6 : structure d'un simple RNN

8.9.1 Apprentissage d'un RNN

La méthode d'apprentissage d'un RNN est " la rétro-propagation au cours du temps "qui représente une généralisation de la rétro-propagation des réseaux feed-forward [P.J.Werbos, 1988].

Les réseaux de neurones récurrents sont fondamentalement différents des architectures feed-forward dans le sens qu'ils opèrent non seulement sur un espace d'entrée mais également sur un espace d'état interne [M. Bodn, 2002].

Considérant un réseau constitué de : deux couches où chaque couche a son index *k*, des nœuds d'entrée *i*, des nœuds cachés *h* et des nœuds de sortie *j*.

Dans un réseau feed-forward, un vecteur d'entrée *x* est propagé à travers la première couche de poids *v* comme suit [M. Bodn, 2002] :

$$y_j\,(t) = g\left(net_j\,(t)\right) \qquad (8.17)$$

$$net_j\,(t) = \sum_i^n x_i\,(t)\;v_{ji} + \;\theta_j \qquad (8.18)$$

Où *n est* le nombre d'entrées, θj est le biais et g une fonction d'activation.

Dans un réseau récurrent (simple), le vecteur d'entrée est similairement propagé mais combiné avec un état d'activation précédent par une couche *récurrente* additionnelle *u* [M. Bodn, 2002] :

$$y_j\,(t) = g\,\left(net_j\,(t)\right) \qquad (8.19)$$

$$net_j\,(t) = \sum_i v_{ji}\,x_i\,(t) + \;\sum_k u_{jk}\,y_k\,(t-1)\;+\;\theta_j \qquad (8.20)$$

Où k indique un nœud ' état '.

Dans les deux cas (récurrent ou non récurrent), la sortie du réseau est déterminée par un ensemble de poids de sortie w comme ci-dessous [M. Bodn, 2002]:

$$y_k\,(t) = g\left(net_k(t)\right) \qquad (8.21)$$

$$net_k\,(t) = \sum_j^m y_j\,(t)\,w_{kj} + \;\theta_k \qquad (8.22)$$

L'apprentissage d'un réseau non-récurrent est montré dans la **figure 8.7** [M. Bodn, 2002] :

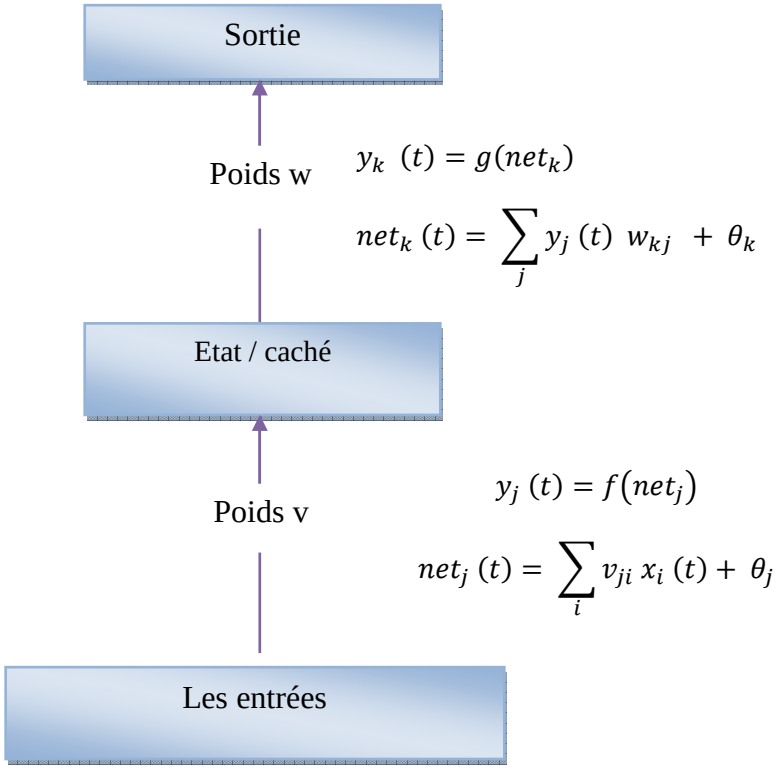


Figure 8.7: l'apprentissage d'un réseau de neurones non récurrent (feed-forward)

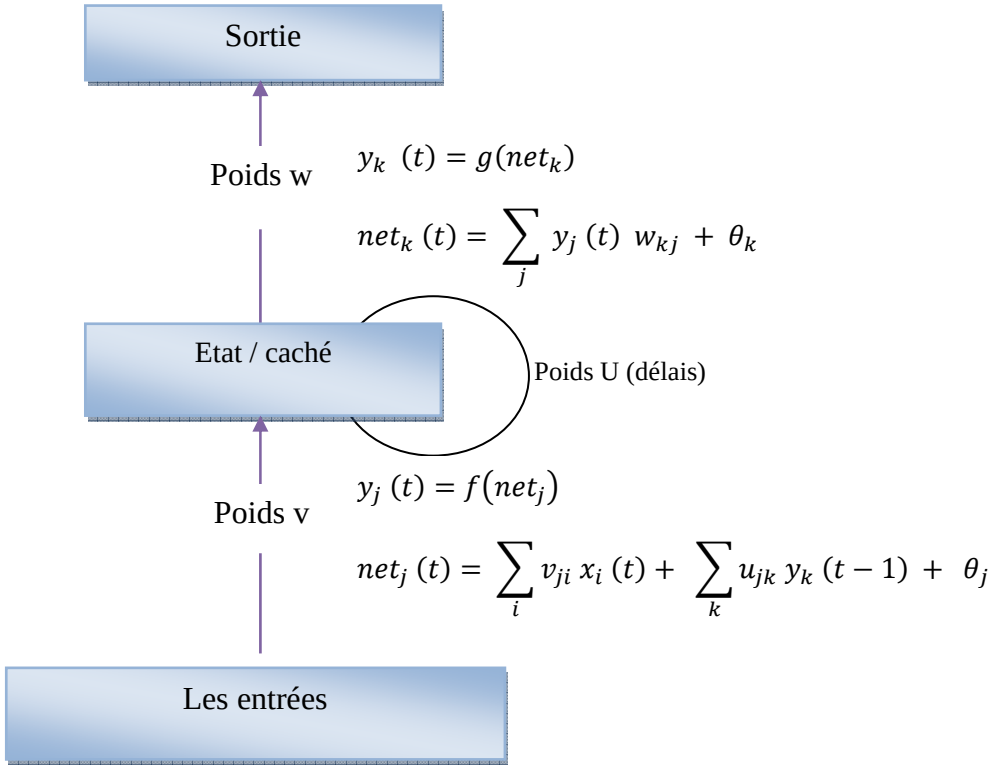


Figure 8.8 : l'apprentissage d'un simple réseau de neurones récurrent

Le principe du rétro- propagation

La rétropropagation de gradient vise à minimiser la fonction d'évaluation de la performance du réseau de neurones qui est le plus souvent la somme des erreurs quadratique [M. Bodn, 2002]:

$$C = \frac{1}{2}\sum_{p}^{n}\sum_{k}^{m}\left(d_{pk} - y_{pk}\right)^2 \qquad (8.23)$$

Où *d* est la sortie prévue, *n* est le nombre d'exemples d'apprentissage disponibles et *m* le nombre de nœuds de sortie.
Ainsi, l'erreur pour les nœuds de sortie est [M. Bodn, 2002]:

$$\delta_{pk} = \left(d_{pk} - y_{pk}\right) g'\left(y_{pk}\right) \qquad (8.24)$$

et pour les nœuds cachés

$$\delta_{pj} = \sum_{k}^{m} \delta_{pk}\, w_{kj}\; g'\left(y_{pj}\right) \qquad (8.25)$$

et le changement de poids se fait selon :

$$\Delta w_{kj} = \eta \sum_{p}^{n} \delta_{pk}\, y_{pj} \qquad (8.26)$$

Pour les poids de sortie, et

$$\Delta\, v_{ji} = \eta \sum_{p}^{n} \delta_{pj}\, x_{pi} \qquad (8.27)$$

Pour les poids d'entrée, en ajoutant un terme du temps, les poids récurrents peuvent être modifiés selon [M. Bodn, 2002]:

$$\Delta u_{jh} = \eta \sum_{p}^{n} \delta_{pj}\,(t)\, y_{ph}\,(t-1) \qquad (8.28)$$

Un choix commun de la fonction d'activation est la fonction sigmoïde :

$$g(net) = \frac{1}{1 + e^{-net}} \qquad (8.29)$$

Le dérivé de la fonction logistique peut être écrit comme suit

$$g'(y) = y(1-y) \qquad (8.30)$$

8.10 Le réseau de neurones à délais temporels

Les réseaux de neurones à délais temporels sont un cas particulier des réseaux de neurones récurrents. La réponse de ces réseaux au temps *t* est basée sur les entrées en temps. L'apprentissage de ce type de réseaux de neurone se fait suivant une méthode similaire au rétropropagation, mais modifié selon les séquences des entrées utilisées.

La notion de délai :
Un attribut : est un composant du vecteur de l'exemple à apprendre.
Unité d'attribut : L'unité du réseau associée à l'attribut à apprendre. Il y a autant d'unités que des attributs dans la couche d'entrée d'un TDNN.
Le délai : Afin de pouvoir reconnaître les exemples avec leur temps invariant, les anciennes valeurs d'activation des unités d'attributs aussi bien que leur valeur de connexions doivent être mémorisées. Ceci est réalisé en mémorisant les unités d'attributs avec toutes leurs connexions de sortie pour chaque pas du temps et avant de mettre à jour les unités originales. Le nombre des étapes de temps mémorisées par ce procédé s'appelle « délai ».

L'activation d'une unité est habituellement calculée en passant la somme pondérée de ses entrées à une fonction d'activation (un seuil ou une fonction sigmoïde). Pour TDNNs ce comportement est modifié en introduisant la notion du délai.

Cette propriété du temps des TDNNs les rend indépendants des erreurs d'algorithmes de prétraitement. Leur inconvénient majeur est la longue durée de la phase d'apprentissage intensive.

Conclusion

8.11 Conclusion

Les réseaux de neurones sont des algorithmes d'apprentissage inspirés du fonctionnement d'un cerveau humain. Un réseau de neurones est adaptatif, auto-organisationnel et peut répondre dans un temps réel.

Les réseaux de neurones peuvent être distingués en quatre classes basées sur l'architecture adoptée : les réseaux feed-forward, les réseaux récurrent, les réseaux de fonctions à base radiale (RBFs) et les cartes auto-organisatrices (SOMs).

Les réseaux Feed-forward n'ayant pas de retour (connexion) en arrière alors que les réseaux récurrents emploient une mémoire interne à travers des connexions bouclées. Ceci les rend applicables aux tâches difficiles comme la reconnaissance de l'écriture manuscrite non segmentée. Les réseaux de neurones à délais temporels sont un cas particulier des réseaux de neurones récurrents. La réponse de ces réseaux au temps *t* est basée sur les entrées en temps, ils se sont essentiellement adaptés à la reconnaissance de l'écriture en-ligne.

Dans les réseaux feed-forward tels que MLP, une combinaison linéaire des données d'entrée est calculée à travers une fonction qui renvoie le produit scalaire des données d'entrée et leurs poids synaptiques correspondants, contrairement aux réseaux RBFs où la fonction de combinaison renvoie la distance euclidienne entre les données d'entrée et les centres des fonctions radiales. La différence principale d'un RBF par rapport à un MLP est l'absence des poids de la -couche cachée. Aussi, les sorties cachées sont basées sur des fonctions à base radiale. L'apprentissage d'un RBF est basé sur le choix des centres et leurs largeurs aussi bien que les poids des neurones de sortie. Le réseau RBF soufre du problème d'identification de ces paramètres de contrôle qui sont nombreux. Les SOMs possèdent une structure différente où les neurones sont régulièrement placés sur une grille hexagonale ou rectangulaire. Les SOMs permettent la compression des données multidimensionnelles. L'apprentissage d'un SOM est l'adaptation des poids *de connexion* de telle sorte les exemples voisins dans l'espace d'entrée sont associés au même neurone ou à des neurones proches dans la carte.

Le MLP n'est pas invariant au décalage des formes à reconnaitre, à leur facteur d'échelle et autres aspects de déformation, aussi, la topologie des données d'entrée est complètement ignorée durant l'apprentissage, ainsi les résultats pour toutes les déformations sont similaires. Pour résoudre ce problème, les réseaux de neurones convolutionels *CNN*s ont été introduits. Un neurone d'une couche d'un perceptron est connecté à tous les neurones de la couche précédente tandis que pour un *CNN* le neurone est connecté à un sous-ensemble de neurones de la couche précédente. Ainsi, chaque neurone peut être vu comme unité de détection d'une caractéristique locale. Les *CNN*s combinent des idées architecturales pour assurer un certain degré d'invariance au décalage et au facteur d'échelle qui sont : les champs réceptifs locaux, le partage des poids et le sous-échantillonnage spatial ou temporel. Les *CNNs* s'avèrent le meilleur choix pour la reconnaissance de chiffres manuscrits. L'entrainement des *CNNs* ne demande pas beaucoup du temps comme les MLPs puisque le nombre de paramètres est considérablement réduit [Y. LeCun et al, 1998b].

À l'inverse de ces types de réseaux de neurones qui sont souvent basés sur la rétro-propagation comme méthode d'apprentissage, le réseau de neurones probabiliste est basé sur des principes statistiques dérivés de la stratégie de décision de Bayes et des fonctions estimateurs de la densité de probabilité.

Chapitre 9
LES RESEAUX DE NEURONES EVOLUTIONNAIRES

9.1 Introduction

Les réseaux de neurones artificiels et les algorithmes évolutionnaires (AEs) sont inspirés des processus naturels et formulés en tant que modèles informatiques adaptatifs de la vie artificielle. Pour réaliser un réseau de neurones optimal, l'algorithme d'apprentissage doit être dynamiquement adapté avec son architecture et le problème à résoudre. Les premières méthodes de la conception automatique des réseaux de neurones ont employé de divers algorithmes constructifs et destructifs. Un algorithme constructif ajoute de nouveaux nœuds et de connexions d'une façon incrémental, tandis qu'un algorithme destructif enlève graduellement des nœuds et l'information inutile des connexions. Cependant, ces deux méthodes sont sujettes au problème des minimums locaux ; il est difficile de trouver une structure optimale du réseau de neurones parce qu'elles recherchent seulement dans l'espace des solutions restreint et prédéterminé par le problème donné [K.J. Kim, 2008]. Les AEs avec leur nature d'échange d'information peuvent être utile dans ce cas. Par conséquent, un AE est le meilleur candidat en comparaison aux algorithmes qui commencent par un réseau (minimal) maximal et puis supprime (ajoute) des couches, des nœuds ou des connexions si nécessaires pendant l'apprentissage.
Les réseaux de neurones évolutionnaires sont une classe des réseaux de neurones artificiels dans lesquels l'évolution est une autre forme fondamentale d'adaptation en plus de l'apprentissage [A. Ajith, 2002]. Un AE est une méthode universelle de recherche globale qui n'est pas sensible aux configurations initiales [K.J. Kim, 2008]. Les algorithmes évolutionnaires peuvent être employés pour adapter les poids de connexion, la structure du réseau ou ses règles d'apprentissage [A. Ajith, 2002]. Les poids de connexion peuvent être représentés par des chaînes binaires ou réelles et le réseau entier peut être représenté par la concaténation de toutes ces chaînes dans un chromosome. Aussi, les règles d'apprentissage peuvent être adaptées par des algorithmes évolutionnaires ; pour décider par exemple le meilleur taux d'apprentissage ou le meilleur momentum de l'algorithme de rétropropagation de gradient ; ou encore le meilleur algorithme d'apprentissage, ces différents paramètres peuvent être codés par des valeurs réelles [A. Abraham et al, 2000].
L'avantage principal des réseaux de neurones évolutionnaires, par rapport aux RNs traditionnels, est leur capacité d'apprendre dans des espaces de recherche où aucune information de gradient n'est disponible. C'est le cas du paysage des architectures [A. Sierra et al, 2001]. Pour ces raisons, la combinaison des AEs et les RNs deviennent de plus en plus prometteuses. Les algorithmes évolutionnaires, bien qu'ils soient préférables dans la recherche globale, ils se sont inefficaces dans la recherche locale. L'efficacité de l'approche évolutionnaire est considérablement améliorée en incorporant un algorithme de recherche local dans l'évolution [A. Ajith, 2002].

Ce chapitre a pour but d'exhiber les différentes techniques évolutionnaires introduites aux réseaux de neurones pour augmenter leur performance. Les techniques d'adaptation de la structure et poids d'un réseau de neurones précisément le MLP sont exposées, ainsi qu'un aperçu

sur des exemples réels de la littérature est arboré. L'objectif est de comprendre le principe de cette voie intéressante afin d'apporter une solution efficace au problème de la reconnaissance de caractères.

9.2 Les Algorithmes Évolutionnaires (AEs)

Les algorithmes évolutionnaires est une classe des algorithmes d'adaptation probabiliste inspirée du principe de l'évolution naturelle. Ce type d'algorithme suit une stratégie de recherche stochastique sur une population d'individus où chaque individu représente une solution possible au problème à résoudre. Les algorithmes évolutionnaires commencent avec un ensemble de solutions aléatoires. Pour chacune de ces solutions une *valeur de fitness* est évaluée qui peut être la valeur de la fonction à optimiser.

Les algorithmes évolutionnaires peuvent être distingués en trois formes principales :

Les *stratégies évolutionnaires*
Les *algorithmes génétiques*
La *programmation évolutionnaire*

9.2.1 La programmation génétique

La programmation génétique est une méthode permettant de conduire la recherche dans un ensemble d'arbres, qui sont généralement employés pour représenter des fonctions. Les arbres se composent de nœuds, qui correspondent aux *fonctions*, et des feuilles, qui correspondent aux *termes*. Un nœud correspondant à une fonction donnée a toujours un nombre prédéterminé d'enfants, qui représentent les arguments de la fonction. Les termes sont réellement des fonctions sans arguments (qui peuvent être des constantes, ou renvoient un certain état) [C.H. Gros, 2003].
Le but est de trouver un arbre qui optimise une *mesure de fitness*. La recherche est conduite selon la stratégie suivante : un nombre prédéterminé *des individus sont* aléatoirement créés, ces individus forment la *première génération* à évaluer selon la mesure de fitness employée. Une nouvelle génération, comprenant le même nombre d'individus est ensuite créée par sélection des meilleurs individus tout en échangeant les sous-arbres compatibles par *croisement* et *mutation*. Ce processus est répété pour un nombre de générations, ou jusqu'à un individu optimal est trouvé. Il existe plusieurs méthodes de sélection des meilleurs individus telle que la sélection proportionnelle aux valeurs de fitness [C.H. Gros, 2003].

9.2.2 Les stratégies d'évolution

Les stratégies d'évolution ont été employées pour des problèmes d'optimisation avec des représentations en valeurs réelles. À l'inverse des algorithmes génétiques, la recherche est fondamentalement focalisée sur la mutation des gènes. Il est nécessaire de représenter chaque solution (individu) par un vecteur réel. Pour aboutir à des meilleures solutions. Deux parents sont sélectionnés pour produire de nouveaux enfants par recombinaison, qui seront ensuite perturbés par mutation. Le meilleur individu (μ+1 sélection) ou les meilleurs individus ($\mu+\lambda$ sélection) de l'ensemble composé des parents et des enfants seront choisi pour former la prochaine génération [A. Berlanga et al, 1999].

Avec μ est la taille de population, λ est le nombre d'enfants générés par application des opérateurs génétiques.

Les stratégies d'évolution (ES) possèdent deux sélections : *plus* et *virgule*. Dans la sélection (μ ,λ)-ES, les meilleurs μ enfants deviennent les parents de la génération suivante, alors que dans la sélection (μ +λ)-ES, les meilleurs des μ + λ parents plus enfants sont les parents de la nouvelle génération.

Un individu est représenté comme suit [A. Berlanga et al, 1999] :

$$a = (x_1, \dots\dots, x_n, \quad \sigma_1, \dots . \sigma_n)$$

(x_i) sont les valeurs réelles et σ_i leurs écarts type correspondants utilisés pour le processus de mutation dans les stratégies d'évolution$(\mu + \lambda)$.

La mutation est effectuée par les équations suivantes [A. Berlanga et al, 1999] :

$$\sigma'_i = \sigma_i . \exp\big(N(0, \Delta\sigma)\big) \quad (9.1)$$

$$x'_i = x_i + N(0, \sigma'_i) \quad (9.2)$$

Où $x'_i \ et \ \sigma'_i$ sont les valeurs mutées, suivant une distribution normale $\big(N(0, \sigma)\big)$.
La recombinaison suit l'approche canonique des algorithmes génétique.

9.3 L'approche neuro-évolutionnaire

Les applications des algorithmes évolutionnaires aux réseaux de neurones sont souvent concentrées autour les topologies appropriées du réseau et de son apprentissage. Un algorithme évolutionnaire peut rapidement localiser des solutions de haute qualité quand l'espace du domaine est très large ou complexe. Ils se sont très importants en conception et apprentissage des réseaux de neurones où l'espace de recherche est infini, de grande dimension ou multimodal [X. Yao, 1999].

L'algorithme évolutionnaire est souvent employé pour réduire au minimum la somme des erreurs quadratiques entre les sorties prévues et les sorties du réseau après chaque étape d'apprentissage ou pour maximiser l'inverse de cette somme. Une fois que toutes les valeurs de fitness des individus de la population sont évaluées, un procédé de *sélection* est effectué où les meilleurs individus portent une plus grande chance d'être choisi pour la prochaine génération. Les individus sélectionnés subissent la *recombinaison* et la *mutation* pour produire de nouveaux individus. Les solutions les plus mauvaises (individus) sont rejetées de la population et les meilleures sont incluses. Ce processus entier est répété avec la nouvelle population jusqu'à ce qu'un critère d'arrêt soit satisfait [A. Abraham et al, 2000].

Les algorithmes évolutionnaires peuvent aider à déterminer une architecture optimale du réseau de neurones, à fournir des mécanismes efficaces de son apprentissage et aussi des modifications

ou reformulations des objectifs du réseau sans effort [A. Abraham et al, 2000]. Cependant, ils se sont très gourmands en termes du temps d'apprentissage. Ils se sont plus appropriés aux problèmes n'ayant pas toutes les informations nécessaires ou la performance requise doit être élevée. Ils se sont surtout plus efficaces quand ils sont employés pour commencer la conception et une fois qu'une connaissance globale est obtenue, d'autres algorithmes de recherche locale peuvent être employés pour fournir la solution d'une façon plus rapide. Les implémentations parallèles de ces algorithmes sont également de plus en plus utiles pour concevoir des applications réelles [X. Yao, 1999].

Pour réaliser un réseau optimal, la fonction de transfert de chaque nœud (gaussienne, sigmoïde, etc…) peut être aussi formulé comme un problème de recherche global, en évoluant simultanément les fonctions d'activation avec la structure du réseau. Une expérience réussie sur l'évolution des règles d'apprentissage est fournie dans [D. J. Chalmers, 1990] qui ont évolué des règles d'apprentissage pour les réseaux d'une seule couche quand ils sont appliqués à une variété de tâches linéairement séparables [A. Sierra et al, 2001].

9.3.1 Évolution des poids de connexion :

L'évolution des poids de connexion présente une approche d'apprentissage adaptative globale. À l'inverse des méthodes d'apprentissage *basées gradient*, c'est-à-dire la rétropropagation de gradient, les algorithmes évolutionnaires se fondent sur des techniques de recherche probabilistes et ainsi, même si l'espace de recherche est très grand, ils peuvent capturer les meilleures solutions [X. Yao, 1999].

9.3.1.1 Codage des Génotypes

N'importe quel algorithme évolutionnaire nécessite une représentation consistante des paramètres de la fonction d'évaluation de fitness. Les opérateurs génétiques (croisement et mutation) doivent être alors décidés en fonction de la représentation.
Le codage (représentation *de génotypes*) peut être fait de deux manières : binaire ou réel. Dans les deux cas, un individu est une concaténation des poids du réseau sous forme d'une chaîne binaire ou réelle, chaque poids de connexion est ainsi représenté par une chaîne binaire ou réelle. Ces chaînes sont enchaînées pour former *un chromosome*. Chaque chromosome est un individu de la population qui est alors soumis aux opérateurs génétiques.

Les représentations de génotype peuvent être distinguées en deux types de codage :

- *Le codage direct* : Un exemple de tel codage est une matrice de connexions qui indique avec précision et directement l'architecture du réseau correspondant. Le codage direct est simple à décoder c -à-d. la transformation du génotype au phénotype est facile mais souffre souvent des problèmes comme l'implémentation de l'opérateur croisement [A. Ajith, 2002]. Aussi, le codage binaire direct peut avoir des chromosomes de très grande taille ayant pour résultat une évolution inefficace. Le codage réel est souvent employé en concevant les opérations génétiques convenables à de tels chromosomes [J.D. Montana et al, 1989]. Le codage direct augmente la longueur de génotypes avec la croissance de la taille du réseau. Ainsi, l'espace topologique maximum doit être limité par l'utilisateur.

Ceci peut exclure la structure la plus convenable parmi les solutions possibles [P. Eggenbergen-Hotz, 2004].

- *Le codage indirect* exige un effort considérable pour le décodage d'un génotype, ce type de codage a été utilisé par plusieurs chercheurs pour réduire la taille de génotype [H. Kitano, 1990], [S.A. Harp, 1989]. Un exemple de tel codage est l'emploie d'un ensemble de règles de réécriture permettant la construction du réseau de neurones de manière récursive [K. Balakrishnan et al, 1995a]. Les codages indirects utilisés dans la littérature sont deux grandes catégories : les codages grammaticaux permettant de représenter un graphe par des grammaires génétiques, biologiques ou géométriques etc [P. Eggenbergen-Hotz, 2004]. c'est un ensemble de règles suggérant une grammaire contextuelle d'un graphe selon lequel le réseau peut être généré [H. Kitano, 1990], [E. Mjolsness et al, 1989], [X. Yao et al, 1995], [E. Vonk et al, 1995]. L'autre catégorie est le paradigme de programmation génétique telle que l'élaboration de programmes LISP comme modèle d'une certaine structure.

Il existe plus de détail dans [K. Balakrishnan et al, 1995b] concernant les représentations génétiques des architectures neuronales.

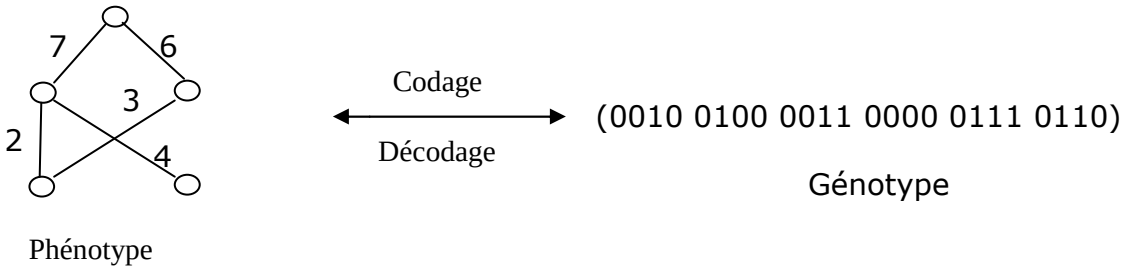


Figure 9.1: Représentation binaire des poids de connexion [X. Yao, 1999].

9.3.1.2 Le codage réel dans un algorithme évolutionnaire

Le codage réel des poids peut être fait par un vecteur n-dimensionnel. Le vecteur des poids de connexion peut être représenté comme suit : $u = (u_1, u_2, u_3..., u_n)$ où $u_1, u_2,..., u_n$ sont les poids du réseau de neurones.
Un autre *vecteur de variance* : $S = (s_1, s_2, s_3..., s_n)$ est généralement employé. Chaque individu de la population est un réseau de neurones représenté par une paire de ces deux vecteurs - (U_i, S_i). La valeur de fitness de chaque individu est la moyenne des erreurs calculées entre la sortie calculée et la sortie prévue de chaque nœud de sortie [A. Berlanga et al, 1999].

$$Erreur = \frac{\sum(sortie_{réelle} - sortie_{prévue})^2}{nombre\ de\ noeuds\ de\ sortie} \qquad (9.3)$$

L'objectif de l'algorithme est de réduire au minimum la valeur de fitness.Pour chaque individu de la population, une progéniture est créée par mutation comme suit [A. Berlanga et al, 1999], [X. Yao, 1999] :

$$S_i' = S_i \exp(\tau' N(0,1) + \tau N_i(0,1)) \qquad (9.4)$$
$$\mu_i' = \mu_i + S_i' N(0,1) \quad pour \quad i = 1,2,\dots\dots,n$$

n est le nombre d'individus.

Cette formule permet une mutation gaussienne. N(0,1) est une valeur aléatoire avec distribution normale de moyenne 0 et variance 1. N_i (0.1) est une valeur aléatoire similaire à N(0,1) mais produite pour chaque individu de la population. Les valeurs de τ' et τ sont généralement calculées comme suit [A. Berlanga et al, 1999]:

$$\tau' = \frac{1}{sqrt\,(2\,sqrt\,(n))} \qquad (9.5)$$

$$\tau = \frac{1}{sqrt(2n)} \qquad (9.6)$$

Le vecteur de variance S_i est employé pour décider la valeur de décalage (changement) des poids. Les poids qui ont bien contribué à la convergence aux sorties prévues, leur valeur ne subissent pas de grand changement [A. Berlanga et al, 1999].

La prochaine étape évalue la fitness de la progéniture pour produire la prochaine génération. La prochaine population peut être produite par sélection de tournoi.
De manière générale, l'algorithme d'apprentissage neuro-évolutionnaire suit les étapes ci-dessous :

1) Créer aléatoirement une population de chromosomes
2) Construire le réseau de chaque chromosome
    3) Entrainer le réseau
    4) Valider le réseau
    5) tester le réseau
6) Mesurer la fitness après apprentissage pour un nombre d'époques
7) Éliminer les chromosomes (réseaux) qui ne respectent pas les limites de fitness
8) Appliquer le croisement génétique aux chromosomes non-éliminés
9) appliquer la mutation génétique à la population
10) Si la population ne satisfait pas le critère d'arrêt revient à l'étape 2

Bien que les AEs offrent une manière intéressante d'optimisation des poids d'un RN, ils se sont relativement lents par rapport aux méthodes de gradient. Une solution évidente est de combiner un AE avec une méthode de recherche locale telle que la méthode de gradient. L'algorithme évolutionnaire est ainsi employé pour localiser la meilleure région de l'espace de recherche et puis un procédé de recherche local est employé pour localiser le point optimal de cette région. A l'inverse des méthodes de gradient, les AEs ne sont pas sensibles aux conditions initiales. L'AE

découvre le meilleur ensemble de poids initiaux pour initier l'algorithme de recherche local. Un tel algorithme *hybride* est tout à fait concurrentiel en termes de convergence et vitesse [X. Yao, 1999]. C'est le cas par exemple de l'algorithme ci-dessous [A. Ajith, 2002].

L'algorithme de base de l'approche métaheuristique proposée dans [A. Ajith, 2002] est comme suit :

1. Générer la première population des réseaux de neurones avec des architectures, des fonctions de transfert, des nœuds et des poids de connexion initialisés au hasard.
2. Faire entraîner en parallèle chaque réseau par les méthodes de BP/SCG/QNA et LM. (BP : backpropagation, SCG (scaled conjugate gradient) l'algorithme de descente de gradient conjugué, QNA : quasi-Newton algorithm et LM : Levenberg-Marquardt)
3. Sélectionner des parents pour reproduire la prochaine population
4. Appliquer la mutation aux parents pour produire une progéniture de la prochaine génération.
5. Revient à l'étape 2
6. Arrêt si la solution est trouvée ou le nombre maximal d'itérations est atteint.

9.3.1.3 Comparaison entre l'apprentissage évolutionnaire et l'apprentissage basé rétro-propagation de gradient

Les approches évolutionnaires sont attractives parce qu'elles peuvent mieux traiter les problèmes de recherche global qui possèdent un espace de recherche très vaste, complexe et multimodal. Elles ne dépendent pas de l'information de gradient de la fonction objective et se sont particulièrement utiles quand cette information est indisponible ou très difficile à estimer. De plus, le même algorithme évolutionnaire peut être employé pour l'apprentissage de différents types de réseaux de neurones : feedforward, récurrent,…etc. La complexité d'un RN peut être diminuée et sa généralisation peut être aussi améliorée en incluant un terme de régularisation dans la fonction de fitness [X. Yao, 1999].

L'apprentissage évolutionnaire est très lourd pour certains problèmes par rapport à des variantes rapides des algorithmes de gradient [E. M. Johansson et al, 1991]. Cependant, les AEs sont généralement moins sensibles aux poids initiaux. Ils recherchent une solution globale optimale, alors qu'un algorithme de la descente de gradient peut trouver un optimum local dans le voisinage de la solution initiale [X. Yao, 1999].

La méthode GA–Bp est une technique d'apprentissage qui commence par un algorithme génétique et puis exécute l'algorithme de la rétropropagation de gradient. Cette méthode peut être mieux ou équivalente aux variantes les plus rapides de la rétropropagation de gradient dans les réseaux de neurones de petite taille, mais beaucoup moins efficace dans les réseaux de grande taille. Cependant, il y avait beaucoup d'autres travaux qui rapportent d'excellents résultats en utilisant les algorithmes évolutionnaires hybrides et de la descente de gradient [W. Kinnebrock, 1994], [B. Deo et al, 1994]. En général, les algorithmes hybrides tendent à réaliser mieux que d'autres pour un grand nombre d'applications [X. Yao, 1999].

9.3.2 L'évolution d'architecture des réseaux de neurones

La détermination de la topologie optimale d'un réseau de neurones était toujours un problème, il est même impossible de dire qu'une certaine architecture est optimale pour un problème donné. Pour trouver l'architecture la plus convenable d'un réseau de neurones, on doit l'évoluer pour s'adapter à une tâche bien précise. Dans un espace de recherche très large ; des architectures similaires peuvent avoir de différentes performances et de différentes architectures peuvent avoir des performances similaires [H. Kwasnicka et al, 2005].

Le but de l'apprentissage d'une topologie réseau est de déterminer l'ensemble optimal des nœuds et de leurs connexions. Ceci est basé sur la minimisation d'une fonction d'erreur tout en ajustant de manière itérative les poids. L'erreur moyenne entre les sorties prévues et les sorties des nœuds de sortie sert comme fitness de la configuration correspondante. La représentation des données et les opérateurs de l'algorithme évolutionnaire sont les deux facteurs les plus importants dans la conception évolutionnaire d'un réseau de neurones [H. Kwasnicka et al, 2005].

L'adaptation de l'architecture d'un réseau de neurones peut être réalisée par des algorithmes constructifs ou destructifs. Les algorithmes constructifs, ajoutent la complexité au réseau à partir d'une architecture très simple jusqu'à ce que le réseau entier puisse bien apprendre la tâche. Les algorithmes destructifs commencent par des grandes architectures et enlèvent des nœuds et des connexions jusqu'à ce que le RN ne puisse pas accomplir sa tâche [A. Ajith, 2002]. Cependant, comme mentionné dans l'introduction ce type d'algorithme peut converger prématurément à une seule solution locale. Un algorithme évolutionnaire est une façon simple et prometteur pour surmonter cette anomalie.

Dans un AE, la représentation de génotypes est critique au fonctionnement correct d'un système de conception d'architecture neuronale. La représentation doit prendre en considération les structures utiles et omettre les structures sans signification, elle doit également contraindre le processus de décodage. Par exemple, un réseau de neurones exigeant une structure récurrente devrait avoir une représentation assez expressive pour décrire les réseaux récurrents. En plus le mécanisme de décodage devrait pouvoir lire cette représentation et la transformer en réseau récurrent le plus approprié [H. Kwasnicka et al, 2005].

La performance d'une certaine architecture ne peut être jugée qu'après son apprentissage. Généralement, le réseau est entraîné pour un nombre d'époques et sa généralisation est mesurée sur un ensemble de validation et de test. Bien évidemment, le meilleur réseau de neurones est celui qui a la meilleure capacité de généralisation [H. Kwasnicka et al, 2005].

Le processus principal de l'approche de conception neuro-évolutionnaire d'architecture est montré dans la Figure 9.2 [K. Balakrishnan et al, 1995a].

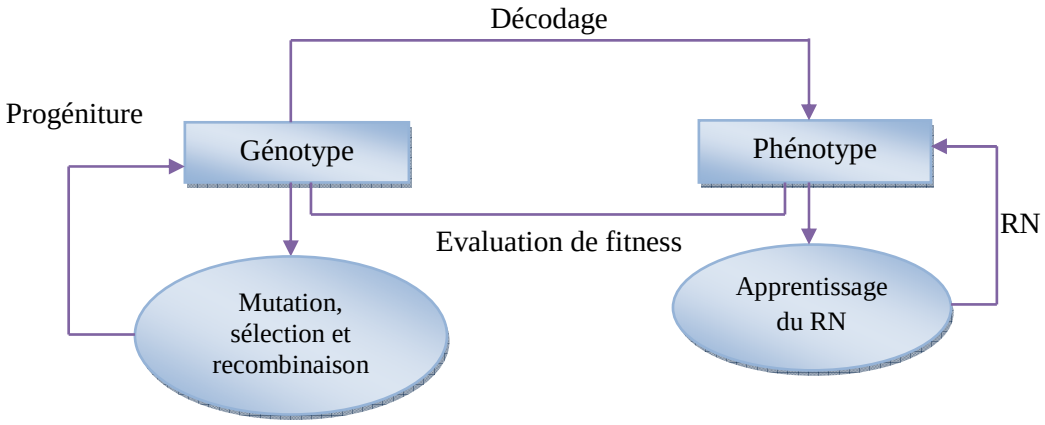


Figure 9.2: processus d'évolution de l'architecture d'un réseau de neurones (RN) [K. Balakrishnan et al, 1995a]

La performance du réseau de neurones dépend fortement de sa structure puisque l'interaction des informations des divers nœuds est déterminée par cette structure. La taille d'un réseau de neurones est également très importante ; un réseau de petite taille ne peut pas apprendre la liaison entre les différentes entrées et les sorties désirées ; un réseau de très grande taille ne peut pas mémoriser correctement les entrées précédemment vues et ainsi perdra la capacité de généralisation [H. Kwasnicka et al, 2005], [H. Kwasnicka et al, 2005].

9.3.2.1 La représentation de génotype

Cette section démontre par un exemple, la conception évolutionnaire d'une architecture neuronale. L'exemple est tiré de [G.F. Miller et al, 1989]. Dans cet exemple l'architecture du réseau est constituée de N unités représentées par une matrice C de dimension N * (N + 1) où N colonnes indique les contraintes sur les connexions entre les N unités et la colonne finale indique la contrainte sur le biais de chaque unité [K. Balakrishnan et al, 1995a].

Chaque entrée C_{ij} de la matrice indique la nature de la connexion de l'unité i à l'unité j (ou le seuil du biais de l'unité i si j = N +1). Dans cet exemple chaque entrée de la matrice peut prendre une des deux valeurs possibles 0 dénote l'absence et 1 indique la présence d'une connexion entre les unités correspondantes [K. Balakrishnan et al, 1995a]. Un génotype est construit par concaténation des lignes de la matrice de connectivité pour donner une chaîne de longueur N * (N +1). Ceci est illustré dans la figure 9.3 [K. Balakrishnan et al, 1995a].

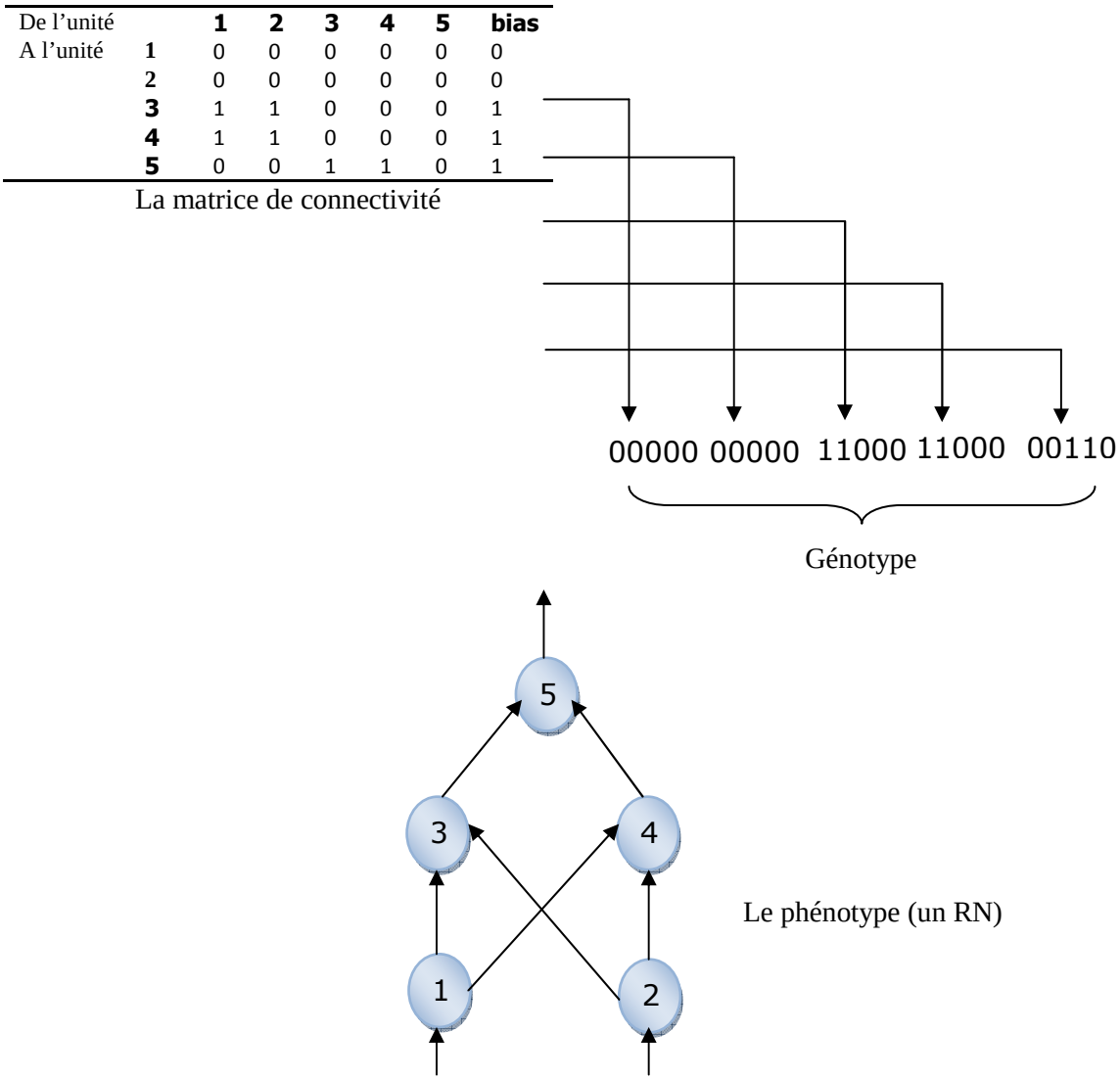


Figure 9.3: La conception évolutionnaire d'un réseau de neurones [K. Balakrishnan et al, 1995a]

L'algorithme génétique évolue une population de telles chaînes binaires (chacune de longueur N*(N +1). Chaque chaîne représente une architecture possible d'un réseau de neurones (RN). Une sélection proportionnelle aux valeurs de fitness est employée pour la reproduction ; les opérateurs de croisement et mutation sont ensuite appliqués [K. Balakrishnan et al, 1995a].

Pour déterminer la fitness d'un génotype, l'architecture qu'il représente est premièrement construite à partir les connexions correspondantes aux entrées qui sont égale à 1 de la matrice de connectivité. Ces connexions de réseau sont ensuite initialisées par des valeurs aléatoires et le réseau subit un apprentissage pour un nombre d'époques à l'aide de l'algorithme de la rétropropagation. A la fin de l'apprentissage, la somme des erreurs quadratiques du réseau est utilisée comme mesure de fitness [K. Balakrishnan et al, 1995a].

Dans l'approche neuro--évolutionnaire, la pression évolutionnaire guide les réseaux vers les meilleures solutions. Les réseaux de neurones qui achèvent les meilleures décisions reçoivent de

plus hautes fitness, permettant à leurs gènes de survivre et de propager aux futures générations [D. E. Moriarty et al, 1994].

9.3.3 L'évolution des exemples d'apprentissage

Un grand nombre des exemples d'entrées d'un RN augmente sa taille et son temps d'apprentissage et peut même dégrader sa capacité de généralisation. Dans un ensemble d'apprentissage, il existe souvent une certaine redondance pouvant être diminuée par diverses techniques de réduction de la base d'apprentissage y compris l'analyse de composant principal employé pour la réduction de dimensions des vecteurs d'entrées [X. Yao, 1999].

Etant donné un ensemble de grande taille des entrées potentielles d'un RN, un AE peut trouver un sous-ensemble optimal d'attributs caractéristiques permettant d'améliorer la performance du RN par rapport à celle d'un RN utilisant l'ensemble entier des attributs caractéristiques. Des algorithmes évolutionnaires ont été employés de manière efficace pour exécuter une telle recherche [J. F. Fontanari et al, 1991], [Z. Guo et al, 1992] [A. D. Brown et al, 1997]. Une meilleure performance avec peu d'attributs d'entrées a été rapportée dans ces travaux [X. Yao, 1999].

Dans la méthode d'évolution des attributs caractéristiques, chaque individu de la population représente un sous-ensemble de l'ensemble entier des entrées possibles. Ceci est réalisable à travers l'utilisation d'un chromosome binaire dont la longueur est identique à tout le nombre des vecteurs d'attributs caractéristiques. Chaque bit du chromosome correspond à un vecteur d'attributs. "1" indique la présence d'un attribut, et "0" indique son absence. L'évaluation d'un individu est effectuée en faisant l'apprentissage du RN avec ces entrées et en employant le résultat pour calculer sa valeur de fitness [X. Yao, 1999].

9.3.4 L'évolution des ensembles de RNs:

L'apprentissage est souvent formulé comme un problème d'optimisation. Cependant, l'apprentissage est différent de l'optimisation : l'objectif de l'apprentissage est d'aboutir à une meilleure généralisation alors que l'objectif de l'optimisation est de réduire au minimum un certain critère d'erreur sur l'ensemble d'apprentissage (un ensemble issu de nichage ou spéciation). Un RN avec une erreur minimale peut ne pas avoir la meilleure généralisation à moins qu'il y ait une équivalence entre la généralisation et la minimisation d'erreur sur les exemples d'apprentissage [X. Yao, 1999]. Un exemple de l'approche ensembliste est présenté dans la section 9.7.1.

En revanche, la mesure exacte de la généralisation est presque impossible dans les problèmes réels bien qu'il y a beaucoup de théories et critères de la généralisation, tels que, la longueur de description au minimum (*minimum description length* :MDL) [J. Rissanen , 1978], le critère de l'information d'Akaike **(***Akaike information criteria* **:** AIC**)** [H. Akaike, 1974] , et la longueur de message minimum (MML) [C. S. Wallace et al, 1991]. Dans la pratique, ces critères ont été souvent employés pour réduire au minimum les fonctions d'erreur pour maximiser la généralisation. Néanmoins il n'y a pas de garantie de convergence vers la solution optimale [X. Yao, 1999].

La combinaison de différents individus d'une population des RNs pour les intégrer dans un système est l'une des solutions permettant de meilleurs résultats. Une telle population de RNs s'appelle l'ensemble des RNs. Il existe certains travaux très réussis qui montrent l'avantage des algorithmes évolutionnaires pour l'évolution des ensembles de RNs [X. Yao et al, 1996], [S. Wesolkowski et al, 1997], [X. Yao et al, 1996], [X. Yao, 1999].

9.4 Difficultés de l'approche neuro-évolutionnaire :

Une des difficultés principales en évoluant un RN est le choix des opérateurs génétiques employés dans les AEs. Des AEs basés croisement et des AEs basés mutation ont été employés. Cependant, l'utilisation du croisement semble contredire l'idée fondamentale derrière les réseaux de neurones, parce que le croisement fonctionne mieux quand il y a des "bloque d'information", mais la notion du bloque n'existe pas dans un RN puisque le RN est une connaissance distribuée [D. E. Rumelhart et al, 1986], [X. Yao, 1999].

La connaissance dans un RN est distribuée entre tous les poids du RN. La Recombinaison d'une partie d'un RN avec une autre partie d'un autre RN est susceptible de détruire les deux RNs qui ont été déjà entraînés. Cependant, si le RN n'emploie pas une représentation distribuée mais plutôt localisée, telle que le cas des réseaux de fonction à base radiale (RBF), le croisement pourrait être un opérateur très utile. Il existe certains travaux de ce type où de bons résultats ont été rapportés [P. J. Angeline, 1997], [E. P. Maillard, 1997]. Cependant, la représentation distribuée d'un RN possède de meilleures possibilités de généralisation pour un grand nombre de problèmes réels [X. Yao, 1999].

Il est donc prouvé que l'opérateur croisement dans l'approche neuro-évolutionnaire produit plus d'inconvénient que d'avantage dans l'évolution d'une représentation distribuée telle que le perceptron multi-couche parce qu'il détruit la connaissance apprise. Le croisement est plus approprié aux RNs localisés tels que les réseaux de neurones de fonctions à base radiale RBF [X. Yao, 1999].

Les sources principales de bruit dans l'approche neuro-évolutionnaire sont [X. Yao, 1999] :

- l'initialisation aléatoire des poids de connexion : de différents poids initiaux peuvent produire différents résultats d'apprentissage.
- l'algorithme d'apprentissage : de différents algorithmes d'apprentissage peuvent produire différents résultats d'apprentissage même avec le même ensemble des poids initiaux. Ceci est particulièrement vrai pour des fonctions d'erreur multimodale.
- La séparation de l'évolution d'architecture et l'évolution des poids de connexion peut mener à une évaluation incorrecte de fitness et une évolution souvent trompeuse. L'évolution simultanée des architectures de RN et des poids de connexion produit généralement de meilleurs résultats.

L'évolution d'un RN est une tâche difficile et peut avoir comme conséquence une évaluation *incontrôlable de fitness* ; les différents poids initiaux peuvent conduire à différents résultats d'apprentissage ; les différentes structures peuvent conduire à différents résultats sur le même

ensemble de poids initiaux, la représentation de génotypes de deux RNs, bien qu'ils définissent la même fonction pouvant être structurellement différents. Les AEs ne peuvent pas détecter ces différences de structures, dans ce cas, un croisement produira une progéniture inviable. L'espace de recherche est aussi rigoureusement augmenté et l'efficacité des opérateurs est sévèrement affectée. En plus, dans les réseaux de neurones où plusieurs tâches doivent être apprises, il y a le problème de confusion des rôles pouvant mener à des solutions similaires. Une des solutions simples à ce type de problème est de faire d'abord évoluer la structure du réseau avant que n'importe quel opérateur génétique soit appliqué (tel que la mutation et le croisement), faire évoluer la structure et puis une phase finale d'apprentissage est appliquée une fois que la meilleure structure est décidée [X. Yao, 1999]. Une autre solution aux problèmes mentionnés ci-dessus est de limiter l'opérateur de sélection à des sous-populations, et d'introduire des mesures intuitives de seuil aux opérateurs de croisement et de mutation [X. Yao, 1999], [H. Kwasnicka et al, 2005].

D'une façon générale, le bruit est provoqué par les un-à-plusieurs mapping de génotypes aux phénotypes. Angeline et *al.* [P. J. Angeline et al, 1994] et [D. B. Fogel, 1995] ont fourni une discussion détaillée sur le mapping entre les génotypes et les phénotypes. Il est clair que l'évolution d'une architecture sans n'importe quelle information sur les poids provoque des difficultés dans l'évaluation exacte de la fitness. En conséquence, l'évolution serait inefficace [X. Yao, 1999].

Afin de réduire de tels bruits, l'architecture du réseau doit être entraînée plusieurs fois avec différents poids initiaux. Le résultat moyen est ainsi utilisé pour estimer la fitness du réseau. Cette méthode augmente considérablement le temps de calcul. C'est l'une des raisons principales pour lesquelles seuls les réseaux de neurones de petite taille sont évolués par les algorithmes évolutionnaires dans des travaux précédents [X. Yao, 1999].
Une autre solution pour résoudre ce problème est de co-évoluer l'architecture d'un RN et les poids de connexion d'une façon simultané [M. Srinivas et al, 1991], [J. R. Koza et al, 1991], [S. Oliker et al, 1992], [X. Yao, 1999]. La co-évolution des solutions et leurs représentations peuvent être une manière efficace pour résoudre des problèmes difficiles où peu d'information est disponible.

9.5 Exemples de modèles neuro-évolutionnaire

9.5.1 L'évolution des ensembles de RNs

La plupart des méthodes neuro- évolutionnières opèrent une population des réseaux de neurones où la fitness de chaque réseau est déterminée indépendamment des autres réseaux de la population. Cependant, cette approche peut souvent converger prématurément à une seule solution simple dominante [D. E. Moriarty et al, 1994]. Dans cette section, nous allons présenter une approche ensembliste permettant d'empêcher la convergence prématurée grâce à la notion de nichage (spéciation) où plusieurs solutions sont possibles pour un problème donné.

Les approches ensemblistes peuvent être classées par catégories en deux types : la génération de différents réseaux artificiels et la combinaison de différentes prévisions. Le Boosting et le

Bagging sont des méthodes populaires employées pour générer différents réseaux de neurones [K.J. Kim, 2008].
Le travail de [K.J. Kim, 2008] présente une méthode de génération des poids de réseaux de neurones par partage de fitness (fitness sharing) et la combinaison de ces réseaux par la méthode (BKS : Behavior Knowledge Space). Le partage de fitness est une méthode de spéciation permettant de partager les mêmes ressources entre les individus ayant une distance inférieure du rayon de partage. Contrairement aux algorithmes évolutionnaires standards qui convergent à une seule solution, cette méthode conduit à plusieurs résultats. Plus particulièrement, cette méthode calcule la distance entre les individus en utilisant la sortie moyenne, la corrélation de Pearson et l'entropie de Kullback Leibler pour augmenter les performances de fitness sharing. Des expérimentations sur différentes bases de données ont montré la supériorité de cette méthode par rapport aux méthodes simples sans spéciation [K.J. Kim, 2008].

La méthode initialise de manière aléatoire chaque RN totalement connecté avec des poids initiaux, et conduit un apprentissage partiel avec l'algorithme de la rétro-propagation pour un nombre limité d'époques afin d'ajuster les poids des architectures initiaux [K.J. Kim, 2008].

La fitness d'un RN est le taux de reconnaissance sur un ensemble de validation. La fitness de partage utilise la mesure de distance pour susciter la diversité. Une fois que la fitness est calculée, l'algorithme génétique choisit les 50% meilleurs individus sur lesquels il applique les opérateurs de croisement et mutation. Après application des opérateurs génétiques, le nouvel ensemble d'individus forme une nouvelle génération. L'algorithme termine quand le critère d'arrêt est satisfait (le nombre de générations dépasse le nombre maximal ou la fitness est égale 1.0). L'étape finale consiste à faire un apprentissage de chaque RN de la dernière génération par rétro-propagation de gradient suivi d'une étape de clustering pour choisir les individus représentatifs de chaque groupe et enfin de les combiner avec de diverses méthodes de combinaison telles que le vote, le vote pondéré etc… [K.J. Kim, 2008].

La **figure 9.4** montre l'algorithme évolutionnaire des poids d'un réseau de neurones [K.J. Kim, 2008].

// Initialisation
Pour chaque individu faire
{Initialiser le triangle haut de la matrice des poids avec des 1
Initialiser le triangle bas de la matrice des poids avec des valeurs entre 0 et 1}

// Apprentissage partiel :
Pour chaque individu faire
{Apprentissage du réseau correspondant pour un nombre d'itérations}
// Evolution
Pour un nombre maximum de générations faire {

1. Pour chaque individu {calculer sa fitness = taux de reconnaissance sur un ensemble de validation}
2. Pour chaque individu {calculer sa distance par rapport aux autres}
3. Pour chaque individu {calculer
   fitness sharing = $\sum_{j}^{taille\ de\ population}(\frac{1-(distance\ par\ rapport\ \text{à}\ individu\ j)}{(\ le\ rayon\ de\ partage)}$}
4. Pour chaque individu {fitness (individu) = (fitness (individu) / sharing (individu))
5. Trier la population selon les valeurs de fitness
6. Selectionner 50 % de la population
7. Appliquer le croisement et la mutation
8. Appliquer un apprentissage partiel
   }
   // Apprentissage total
9. {Pour chaque individu {faire son apprentissage + Validation pour un nombre d'époques}

// Combinaison
{Classification selon la distance et sélection des meilleures solutions de chaque groupe selon fitness sharing et puis faire leur combinaison}

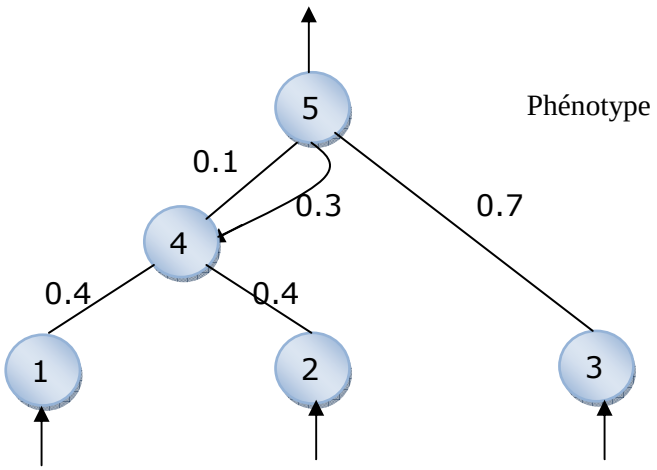
Figure 9.4 : Évolution des poids de RNs par spéciation

9.5.2 L'évolution de l'architecture

Un des principaux défis de l'évolution des RNs est comment trouver une méthode significative de croisement de diverses topologies. Les opérateurs génétiques habituels ne préservent pas la structure topologique du réseau. Un certain type de spéciation est généralement employée de telle sorte que les individus concurrencent dans leurs propres niches mais pas avec la population dans son ensemble. Nous décrivons par la suite une méthode, **N**euro **E**volutionnaire d'**A**ugmentation des **T**opologies (NEAT), qui emploie une méthode de spéciation, et de la croissance incrémentale d'architectures [K.O. Stanley, 2004].

9.5.2.1 NeuroEvolution pour l'Augmentation des Topologies (NEAT)

Un génome dans **NEAT** se compose de plusieurs gènes de connexions. Un gène de connexion contient l'information de deux nœuds connectés, le poids correspondant, et d'un attribut spécial appelé le *nombre d'innovation*. L'information de la couche des nœuds (d'entrée, de sortie, ou cachée) fait également partie du génome. Le nombre d'innovations est employé pour maintenir l'origine historique de chaque gène de telle sorte que les gènes n'ayant pas le même ancêtre ne concurrencent pas les uns avec les autres ; ce nombre est incrémenté à chaque fois qu'un nouveau gène apparaît. Pour que l'information historique passe d'une génération à une autre dans tout le processus évolutionnaire, la progéniture de chaque gène hérite aussi le même nombre d'innovations [K. O. Stanley, 2004].
Le nombre d'innovations sert aussi pour compter le nombre de compétitions entre les différentes topologies. Pendant le processus de la recombinaison, les gènes ayant le même nombre d'innovations peuvent concurrencer les uns contre les autres ; ceux des parents les plus adaptés sont hérités. Les gènes n'ayant pas un nombre d'innovations (disjoint) seront passés à la progéniture [K. O. Stanley, 2004].



Génotype correspondant de la figure précédente

Les gênes nœuds	Noeud1 Couche d'entrée	Noeud2 Entrée	Noeud3 Entrée	Noeud4 Cachée	Noeud5 Sortie
Les gênes de connexions	1< >4 W14 = 0.4 IN = 1	2< > 4 W24 = 0.4 IN = 3	3< >5 W35 = 0.7 IN = 5	4< > 5 W45 = 0.1 IN = 6	5< > 4 W54= 0.3 IN = 10

Figure 9.5: Représentation du génome NEAT [K. O. Stanley, 2004].

Mutation dans NEAT :

La mutation dans une architecture neuronale peut se produire de deux manières - en ajoutant un gène de connexion, ou en ajoutant un gène de nœud. Deux nœuds non attachés peuvent être reliés en ajoutant un nouveau gène de connexion au génome. On peut également diviser une connexion existante en deux et un nouveau nœud peut être introduit entre ces deux nœuds. L'ancienne connexion est désactivée et deux nouveaux gènes de connexion sont ajoutés. De nouveaux nœuds et connexions peuvent être incluses dans la structure suivant cette méthodologie [K. O. Stanley, 2004].

Croisement dans NEAT

La sélection des parents pour la reproduction détermine les attributs topologiques transférés à la progéniture. Si la sélection est aléatoire, il est possible que les innovations structurales soient pertes pendant le processus de reproduction. NEAT traite ce problème en divisant la population en espèces selon leurs similitudes topologiques, et puis en assemblant les meilleurs individus de chaque espèce pour produire la nouvelle progéniture. La division de la population en différentes espèces est basée sur une mesure de distance. Si la distance d'un individu aléatoirement choisi est moins qu'une valeur- seuil, alors les deux individus seront placés dans la même espèce. La spéciation permet la croissance des solutions les plus performantes et ne permet à aucune structure particulière de dominer la population entière [K. O. Stanley, 2004].

9.5.2.2 L'évolution symbiotique : SANE

La structure d'un RN n'est pas unique à un problème donné, et peut exister de différentes méthodes pour la déterminer. Généralement l'architecture d'un réseau de neurones est évaluée par test ; différentes combinaisons des nœuds et des connexions doivent être essayées afin de donner la meilleure réponse au problème à résoudre. De telles méthodes sont fondées sur la performance globale du réseau, ainsi : les parties du réseau qui ont bien contribué à son amélioration sont difficiles à identifier et à les exploiter. Le travail ci-dessous décrit une méthode dénommée *SANE* permettant d'évaluer les parties constructives d'un réseau de neurones d'une façon partielle en se basant sur le principe de la coévolution.

SANE évolue une population de neurones où la tâche de chaque neurone est d'établir des relations avec d'autres neurones de la population pour former un nouveau réseau de neurones. La fitness d'un neurone est déterminée par sa contribution d'amélioration de la fitness d'un réseau en coopérant avec un ensemble de neurones de la population. L'évolution conduit donc la population vers divers neurones symbiotiques pour éviter la convergence prématurée, avec cette nouvelle stratégie symbiotique, la population peut découvrir de meilleures solutions à des problèmes difficiles [D. E. Moriarty et al, 1994].

Les étapes de base d'une itération de *SANE* peuvent être résumées comme suit (figure 9.6) : Pendant l'étape d'évaluation, des sous-populations aléatoires sont choisies et combinées pour former un réseau de neurones. Le réseau est évalué et assigné un score. Les scores sont ajoutés à la valeur de fitness de chaque neurone sélectionné. Le processus continu jusqu'à ce que chaque neurone ait participé à un nombre suffisant de réseaux. La valeur moyenne de fitness de chaque neurone est alors calculée en divisant la somme de ses valeurs de fitness par le nombre de réseaux qu'il a participés avec. Les neurones qui ont une valeur élevée du fitness moyenne ont bien coopéré avec les autres neurones de la population. Les neurones qui ne coopèrent pas et sont nuisibles aux réseaux reçoivent de faibles valeurs de fitness. Une fois que chaque neurone a une valeur moyenne de fitness, des opérations de croisement sont employées pour combiner les chromosomes des neurones ayant les meilleures valeurs de fitness moyenne. La mutation est également employée pour empêcher la perte du matériel génétique [D. E. Moriarty et al, 1994].

Les étapes de base de SANE [D. E. Moriarty et al, 1994]

---

1. Choisir aléatoirement N neurones de la population.
2. Créer un réseau de neurones à partir les neurones choisis.
3. Évaluer le réseau avec la tâche à résoudre.
4. Ajouter le score du réseau à la valeur de fitness de chaque neurone choisi.
5. Répéter les étapes 2-4 pour un nombre suffisant.
6. Calculer la valeur moyenne de fitness de chaque neurone en divisant sa valeur totale de fitness par le nombre de réseaux qu'il a participés avec.
7. Effectuer les opérations génétiques sur la population

Figure 9.6: L'évolution de la structure d'un RN par SANE

Conclusion

9.6 Conclusion

Les réseaux de neurones évolutionnaires représentent une classe des réseaux de neurones dont leur adaptation est basée sur les algorithmes évolutionnaires [A. Ajith, 2002]. L'algorithme évolutionnaire est souvent employé pour l'adaptation de la topologie du réseau et de son apprentissage à une tâche bien précise. Un AE peut aussi découvrir le sous-ensemble d'attributs caractéristiques permettant une meilleure performance du RN par rapport à celle d'un RN employant l'ensemble entier des attributs caractéristiques.

Les approches ensemblistes permettant la génération des réseaux artificiels et ainsi la combinaison des résultats complémentaires offrent de meilleure performance. Le Boosting et le Bagging sont les exemples les plus connus pour la génération des réseaux de neurones [K.J. Kim, 2008].

Bien que les AEs offrent d'intéressantes capacités d'optimisation des RNs, ils se sont très lents par rapport aux méthodes de gradient. Une des solutions est de combiner un AE avec une méthode de recherche locale pour localiser le point optimal de la bonne région trouvée par l'AE [X. Yao, 1999].

La séparation de l'évolution d'architecture et de leurs poids correspondants peut avoir comme conséquence une évaluation *incontrôlable de fitness* ; les différents poids initiaux peuvent conduire à différents résultats d'apprentissage ; les différentes structures peuvent conduire à différents résultats sur le même ensemble de poids initiaux, la représentation de génotypes de deux RNs, bien qu'ils définissent la même fonction pouvant être structurellement différents. Une des solutions à ce type de problème est d'évoluer d'abord la structure du réseau avec différents poids initiaux. Le résultat moyen est ainsi utilisé pour estimer la fitness du réseau et puis une phase finale d'apprentissage est appliquée une fois la meilleure structure est décidée. Cette vision prend un temps énorme d'exécution. C'est l'une des raisons pour lesquelles seuls les réseaux de petite taille sont évolués par les algorithmes évolutionnaires [X. Yao, 1999]. Une autre manière efficace est de co-évoluer l'architecture d'un RN avec leur poids correspondant d'une façon simultané [M. Srinivas et al, 1991], [J. R. Koza et al, 1991], [X. Yao, 1999].

L'opérateur de croisement dans l'approche neuro-évolutionnaire produit plus d'inconvénient que d'avantage dans l'évolution d'une représentation distribuée telle que le perceptron multi-couche parce qu'il détruit la connaissance apprise. Le croisement est plus approprié aux RNs localisés tels que les réseaux de neurones de fonctions à base radiale RBF [X. Yao, 1999].

Troisième partie :

La reconnaissance de caractères manuscrits :

Les approches développées

Chapitre 10

LA RECONNAISSANCE DE CARACTERES PAR DES CLASSIFICATEURS STATISTIQUES BASES SUR L'INTELLIGENCE EN ESSAIM

10.1 Introduction

L'intelligence collective se manifeste dans la capacité des groupes d'entités simples à résoudre des problèmes complexes. Les insectes sociaux tels que les abeilles et les fourmis réalisent collectivement des performances remarquables en exploitant des principes très simples. Ils offrent une preuve spectaculaire de la capacité d'un groupe d'agents simples d'accomplir des tâches complexes de manière rapide et exacte que quelques agents complexes. Néanmoins, l'utilisation des méthodes d'intelligence en essaim n'est plus répandue dans le domaine de la reconnaissance de caractères. Il existe quelques travaux tels que [L. S. Oliveira, 2005], [K.Sankhuangaw et al, 2005], [S. Mathur et al, 2010], [H.Aljuaid et al, 2010].

Dans ce travail, nous avons proposé la résolution du problème de la reconnaissance de caractères notamment les chiffres arabes manuscrits et les lettres latines manuscrites à travers des métaheuristiques principalement basées sur l'intelligence collective à savoir : la méthode d'optimisation d'essaims de particules (PSO), l'algorithme d'abeilles (BA), et l'optimisation par colonies d'abeilles (ABC). L'intérêt de telles approches est de permettre une qualité supérieure des résultats en termes d'exactitude. Les aboutissements sont satisfaisants en s'appuyant sur une étude comparative basée sur le taux de reconnaissance de chaque méthode.

Après avoir donné brièvement les bases théoriques de chaque méthode d'optimisation par intelligence collective que nous avons détaillées dans le 7ème chapitre soient : PSO, BA et ABC, nous essayerons de les adapter à la tâche d'optimisation de la reconnaissance de caractères manuscrits. Nous allons démontrer par la suite l'importance de l'intelligence en essaim dans le cadre de la reconnaissance en illustrant leur potentiel par expérimental sur plusieurs variantes d'images tirées de la base connue de chiffres manuscrits (MNIST) et des exemples de modèles des lettres anglaises. Ces classificateurs s'attachent à grouper les exemples de la base d'apprentissage MNIST en réduisant la variabilité intra classes et en maximisant la variabilité inter-classes ceci est effectué par minimisation de la somme des erreurs quadratiques de manière itérative à travers les méthodes d'optimisation d'intelligence collective.

10.2 Formulation du problème

Comme nous l'avons évoqué au premier chapitre, un système de la reconnaissance de caractères se compose de trois sous-étapes : le prétraitement, l'extraction de caractéristiques et la classification.
L'approche décrite se résume dans les étapes suivantes :

10.2.1 L'étape de prétraitement

Nous exécutons les étapes suivantes avant d'entamer la reconnaissance :

- Binarisation en utilisant la méthode d'Otsu
- Élimination des espaces blancs de chaque image caractère
- Squelettisation (amincissement des caractères)

10.2.2 L'étape d'extraction de caractéristiques

Nous avons considéré trois extracteurs d'attributs caractéristiques qui sont : zonage, les moments de Hu et des attributs « structurels & géométriques ».

Zonage *:* L'image d'un caractère est divisée en 8*8 zones, puis la densité de Pixels de chaque zone est considérée comme élément du vecteur d'attributs, c-à-d, le nombre de pixels du caractère sur le nombre de pixels du fond de l'image. Ainsi le vecteur caractéristique de chaque caractère à une taille de 64 attributs (8*8).

Les Moments de Hu *:* Les sept (7) premiers moments de Hu ont été employés ; ces attributs ont été choisis puisqu'ils se sont connus par leur invariance à la translation, le facteur d'échelle et à la rotation et permettent de minimiser le temps de calcul avec 7 valeurs uniquement.

Les attributs structurels et géométriques :
L'algorithme employé pour l'extraction des attributs structurels & géométriques dont le détail ci-dessous est tiré de [D. Dileep Gaurav, 2009] commence par des opérations de prétraitement telles que :

- La binarisation
- La squelettisation
- Élimination des espaces blancs
- La normalisation
- Effectue un zonage du caractère en 9 zones

Après et pour chaque zone, l'algorithme calcule les attributs structurels et géométriques suivants : Les intersections, les pixels de fin de ligne mineurs, les pixels de fin de ligne, les lignes segment et leurs directions, nombre de lignes horizontales, nombre de lignes verticales, nombre de lignes diagonales droites, nombre de lignes diagonales gauches, la longueur normalisée de toutes les lignes horizontales, la longueur normalisée de toutes les lignes verticales, la longueur normalisée de toutes les lignes diagonales droites, la longueur normalisée de toutes les lignes diagonales gauches, région normalisée du caractère.

Ensuite l'algorithme calcule les attributs suivants sur l'image entière du caractère :

Nombre d'Euler basé sur le nombre de cavités et la région du caractère basé sur le nombre de Pixels du caractère et l'excentricité de l'ellipse encadrant le caractère.

Les intersections et les pixels de fin de ligne : sont décrits auparavant dans les sections 4.3.2.1, 4.3.2.2 du chapitre 4.

Les pixels de fin de ligne mineurs : sont créés quand le Pixel a plus de deux voisins. Il y a deux états possibles :

- Le cas des intersections : quand le Pixel est une intersection. la ligne segment finira à ce pixel et tous ses voisins qui ne sont pas encore visités seront ajoutés à la liste des pixels de fin de ligne mineurs.
- Le cas non-intersection : quand le Pixel n'est pas une intersection, la direction de la ligne segment est trouvée en employant la position du Pixel précédent. Si un des Pixels non visités dans cette direction, alors on le considère comme le prochain Pixel et tous les autres Pixels sont ajoutés à la liste des pixels de fins de lignes mineures. S'il n'existe pas de Pixels non visités dans la direction courante, alors la ligne segment est accomplie et tous les Pixels dans ce voisinage seront ajoutés à la liste des pixels de fins de lignes mineurs.

Direction des lignes segment

Une ligne segment est une ligne traversant du pixel de fin de ligne jusqu'à une intersection. Après l'extraction des lignes segments, elles doivent être classifiées dans l'un des types suivants :

- Ligne horizontale.
- Ligne verticale.
- Ligne diagonale droite.
- Ligne diagonale gauche.

Les attributs :

Chaque zone est ainsi identifiée par un vecteur d'attributs de longueur 9 qui contient les attributs caractéristiques suivants :

1) nombre de lignes horizontales.
2) nombre de lignes verticales.
3) nombre de lignes diagonales droites.
4) nombre de lignes diagonales gauches.
5) La longueur normalisée de toutes les lignes horizontales.
6) La longueur normalisée de toutes les lignes verticales.
7) La longueur normalisée de toutes les lignes diagonales droites.
8) La longueur normalisée de toutes les lignes diagonales gauches.
9) Région normalisée du caractère.

En outre, l'image originale sera ensuite divisée en 3 zones dans la direction horizontale. Ensuite, on refait l'extraction des attributs précédemment décrits sur chacune des zones. Certains attributs sont extraits de l'image entière en utilisant des propriétés régionales notamment :

- Le nombre d'Euler : il est défini comme étant la différence entre le nombre d'objets et le nombre de cavités dans l'image. Par exemple, dans le caractère 'A' parfaitement dessiné le nombre d'Euler est zéro, puisque le nombre d'objets est 1 et le nombre de cavités est 1, tandis que le ` B 'aurait le nombre d'Euler -1, puisqu'il a deux cavités.
- Région : Elle est définie comme le rapport entre le nombre de Pixel du caractère après squelettisation et le nombre total de Pixels de l'image.
- Excentricité : Elle définit l'excentricité de la plus petite ellipse qui entoure le squelette du caractère.

10.2.3 L'étape de la classification

Après la détermination des vecteurs caractéristiques, il est possible de les affecter à des classes prédéterminées de caractères (ici dix classes : de 0 à 9). Dans ce travail, chaque particule (abeille) est constituée d'un nombre de vecteurs d'attributs de caractères choisis aléatoirement à partir de la base d'apprentissage, ces attributs caractéristiques sont ensuite considérés comme étant des centres de groupes. Un groupe est un ensemble de caractères ayant la même classe. Les centres sont des caractères permettant de donner la valeur minimale de la somme des erreurs quadratiques (SSE). Le but de PSO (où l'algorithme d'abeilles) est de réduire au minimum la SSE entre les données d'apprentissage et l'ensemble de caractères manuscrits à reconnaître et puis de mémoriser les meilleurs caractères de la base qui ont donné la fitness réduite au minimum (c'est à dire la somme des erreurs quadratiques).
Les approches proposées emploient PSO ou les dynamiques d'optimisation par colonies d'abeilles pour rechercher la bonne classification de caractères. La colonie d'abeilles ou la colonie de particules choisit collectivement la solution la plus proche en s'orientant vers la minimisation d'une fonction qui calcule la somme des distances euclidiennes sur l'ensemble de tests.

10.3 L'optimiseur d'essaim de particules pour la reconnaissance de caractères

Principe de la méthode PSO :
PSO résulte d'une analogie avec le comportement collectif du déplacement des groupes d'animaux ou le vol des oiseaux. Dans les groupes d'animaux, chaque individu se déplace selon sa propre connaissance qui peut être atteinte par le déplacement de ses voisins [J. Kennedy, 1997], [R.C. Eberhart et al, 2000].

Cette méthode est basée sur la coopération entre des individus simples souvent appelés des agents. Ces agents possèdent une capacité de la mémorisation dans le sens où chaque agent se rappelle de sa meilleure position visitée et la meilleure position trouvée dans son voisinage. Chaque particule se déplace alors selon une trajectoire perturbée entre ces deux positions et sa vitesse.

PSO est un algorithme simple et efficace qui commence par un certain nombre de particules placées de manière aléatoire dans l'espace de recherche et initialisées avec des vitesses aléatoires.

Formalisation :
Dans PSO, chaque particule est caractérisée par son propre vecteur de position et son vecteur de vitesse.
Pour un certain nombre d'itérations ou jusqu'à une condition soit satisfaite, les particules se déplacent dans l'espace de recherche selon les règles récursives suivantes :

$$\begin{cases} V_{id} = {}^{w}V_{id} + c_1 \times r_1 \times (p_{id} - x_{id}) + c_2 \times r_2 \times (p_{gd} - x_{id}) \\ x_{id} = x_{id} + v_{id} \end{cases} \quad (10.1)$$

Où :
X : est la position de la particule.
V : représente sa vitesse.
i : est son index.
d : est sa d ème *dimension* dans l'espace de recherche.
Pi et *Pg* sont respectivement sa meilleure position personnelle et la meilleure position trouvée dans son voisinage.
C_1 et C_2 *:* sont respectivement le facteur cognitif et le facteur social permettant le contrôle du comportement individuel et collectif de chaque particule.
r_1 et r_2 *:* sont des valeurs qui changent uniformément dans l'intervalle [0.1] pour permettre une meilleure exploration de l'espace de recherche
W : s'appelle le poids d'inertie, il sert à contrôler l'équilibre entre l'exploration et l'exploitation de l'espace de recherche.

PSO en pseudo-code [R.C. Eberhart et al, 2000]

Initialiser un nombre de particules (xi)
Pour un nombre d'itérations faire
Pour *i*= 1 au nombre de particules faire
Si fitness (*xi*) < Pbest alors
Maj Pbest = *xi* % Pbest est la meilleure position personnelle
Fin si
Si fitness (*xi*) < gbest alors
Maj gbest = *xi* % gbest est la meilleure position trouvée
Fin si
Pour *d* = 1 au nombre de dimensions faire
Maj de vitesses et positions en employant l'équation. (10.1)
Fin pour
Fin pour

Ci-dessous est l'adaptation de l'algorithme PSO pour la reconnaissance de caractères.

Algorithme 1 : L'optimiseur d'essaim de particules pour la reconnaissance de caractères

Transformer l'ensemble de caractères à reconnaître en vecteur d'attributs
Initialiser une population de particules (où chaque particule inclut les attributs caractéristiques des caractères aléatoirement choisis de la base d'apprentissage et chaque particule possède une taille égale à la taille du vecteur des caractères à reconnaître).

{**Pour** chaque caractère faire// traitement d'un caractère dans une particule
{**pour** chaque particule faire
{**pour** un nombre d'itérations **faire**
calculer la distance euclidienne entre le vecteur d'attributs de ce caractère et le caractère qui lui correspond dans le vecteur des caractères à reconnaître.

Mise à jour de P_{best} et G_{best}
Mettre à jour la position et la vitesse suivant les équations (10.1)
}}}
Employer les attributs trouvés comme centres de groupes pour les étiqueter en choisissant les caractères de la base d'apprentissage qui donnent la valeur minimale de la somme des erreurs quadratiques (SSE).
Les étiquettes des caractères qui donnent la valeur minimale du SSE
Constituent la solution recherchée.

10.4 L'algorithme d'abeilles pour la reconnaissance de caractères

Les abeilles dans la nature emploient la langue de la danse pour indiquer les endroits de l'eau ou de la nourriture. Lorsqu'une abeille trouve un champ de fleurs, elle revient à la ruche, dépose son nectar et commence une danse d'une façon ronde. Cette danse est le langage de communication des colonies d'abeilles, elle indique la direction, la distance par rapport à la ruche et la qualité du nectar des fleurs. La connaissance de chaque individu est recueillie à partir de cette danse. Après la danse, l'abeille revient à l'endroit des fleurs accompagnée des abeilles informées qui ont été dans la ruche. Le plus grand nombre d'abeilles sera mobilisé vers les meilleurs endroits de fleurs. Dans la prochaine danse c'est à dire après le retour à la ruche, les abeilles vont informer s'il y a d'autres sources de nourriture qui sont meilleures [DT. Pham et al, 2007].

Principe :

L'algorithme d'abeilles est une métaheuristique récente qui imite le comportement de fourragement des abeilles de miel en recherchant les endroits (sites) de nourritures. Cet algorithme réalise une recherche par voisinage associée à une recherche aléatoire pour l'optimisation combinatoire.[DT. Pham et al, 2006]. Pendant l'étape d'exploration, une proportion de la population d'abeilles recherche les meilleurs sites et les autres abeilles recherchent de manière aléatoire d'un site à un autre. [DT. Pham et al, 2005].

Formalisation :

Cet algorithme exige un certain nombre de paramètres, à savoir [DT. Pham et al, 2005]:

- Un nombre d'abeilles exploratrices (appelés scouts) (n)
- Un nombre de sites (m) parmi (n) sites visités.
- Un nombre de sites élite (e) parmi les (m) sites choisis.
- Un nombre d'abeilles recrutées pour les meilleurs sites élite :(nep)
- Un nombre d'abeilles recrutées pour les autres (m-e) sites choisis (nsp)
- La taille initiale (ngh) d'un site (pour une recherche en voisinage).
- Le critère d'arrêt.

L'algorithme des abeilles en pseudo-code [DT. Pham et al, 2005], [DT. Pham et al, 2006]

1. Initialiser de manière aléatoire une population de solutions (abeilles exploratrices).
2. Évaluer la fitness de chaque individu.
3. Tandis que le critère d'arrêt n'est pas encore satisfait faire
4. Sélectionner des sites pour une recherche en voisinage.
5. Recruter un nombre d'abeilles pour les sites choisis
6. Recruter plus d'abeilles autour les meilleurs sites (élite).
7. Évaluer les fitnesses des abeilles recrutées.
8. Remplacer chaque site (les meilleurs emplacements) par la meilleure abeille trouvée dans son voisinage.
9. initialiser les abeilles restantes pour rechercher de manière aléatoire en évaluant leurs fitnesses.
8. Fin Tandis que.

Dans la première étape, l'algorithme commence par localiser les abeilles exploratrices de manière aléatoire dans l'espace de recherche. Dans la deuxième étape, les fitnesses des sites visités sont calculées. Dans la quatrième étape, les sites qui ont les meilleures valeurs de fitness sont choisis pour focaliser la recherche dans leurs voisinages et puis, dans les étapes 5.6 et 7 l'algorithme recrute des abeilles autour tous les sites en incluant les sites élites. Le recrutement est une recherche focalisée au voisinage de chaque site choisi, la taille de chaque site est définie dès le départ comme elle peut être variable au cours du temps. Un certain nombre d'abeilles qui appartiennent au voisinage d'un site sont choisies pour leur évaluation. Plus d'abeilles sont choisies au voisinage des meilleurs sites (élites). Le choix peut être directement basé sur les fitnesss des sites ou alternativement, en utilisant leurs valeurs de fitness pour déterminer la probabilité du choix des abeilles. Dans l'étape 8, remplacement des sites par les meilleures abeilles trouvées dans leurs voisinages pour former la prochaine population. Les abeilles restantes sont placées aléatoirement dans l'espace de recherche pour trouver de nouvelles solutions qui peuvent être de qualité supérieure. Ce recrutement différentiel (voisinage + aléatoire) est l'idée principale de l'algorithme d'abeilles. Ces étapes seront répétées jusqu'à ce qu'un critère d'arrêt soit satisfait. [DT. Pham et al, 2006].

Ci-dessous est l'adaptation de l'algorithme BA pour la reconnaissance de caractères.

Algorithme 2 : L'optimiseur d'abeilles pour la reconnaissance de caractères

Transformer l'ensemble de caractères à reconnaître en vecteurs d'attributs caractéristiques.
Initialiser une population des abeilles scoutes (où chaque abeille comporte les attributs caractéristiques des caractères aléatoirement choisis de la base d'apprentissage et chaque abeille possède une taille égale à la taille des vecteurs de caractères à reconnaître).

Initialiser le nombre d'abeilles élites
Initialiser le nombre d'abeilles en voisinage de chaque abeille élite
Initialiser le nombre d'abeilles sites
Initialiser le nombre d'abeilles en voisinage de chaque abeille site

{**Pour** un nombre d'itérations **faire**
{**Pour** chaque caractère à reconnaître **faire**
{**Pour** P=1 : nombre-abeilles-élite **faire**
calculer Fitness élite = la distance euclidienne entre le vecteur d'attributs de ce caractère d'abeille élite et le caractère qui lui correspond dans le vecteur des caractères à reconnaître

{**pour** F=1 : nombre-voisines-élite **faire**
Créer une abeille dans le voisinage de l'abeille élite
Calculer sa fitness
Si sa fitness et mieux que la fitness de élite
Remplacer l'abeille élite par l'abeille voisine}}

{**Pour** P=1 : (nombre-abeilles-elite + 1) : nombre-abeille-site **faire**
calculer fitness site = la distance euclidienne entre le vecteur d'attributs de ce caractère d'abeille site et le caractère qui lui correspond dans le vecteur des caractères à reconnaître

{**pour** F=1 : nombre-voisines-site **faire**
Créer une abeille dans le voisinage de l'abeille site
Calculer sa fitness
Si sa fitness et mieux que la fitness de site
Remplacer l'abeille site par l'abeille voisine}}

- Assigner les abeilles restantes de la population des abeilles scoutes de manière aléatoire et calculer leurs fitnesses
- Trier l'ensemble des abeilles selon leurs valeurs de fitnesses}}

- Employer les attributs trouvés comme centres de groupes pour les étiqueter en choisissant les caractères de la base d'apprentissage qui donnent la valeur minimale de la somme des erreurs quadratiques (SSE).
- Les étiquettes des caractères qui donnent la valeur minimale de la SSE constituent la solution recherchée.

10.5 L'Optimisation par Colonie d'Abeilles Artificielles pour la reconnaissance de caractères

L'Optimisation par Colonie d'Abeilles Artificielles (ABC) est un algorithme basé population proposé récemment par Karaboga [D.Karaboga, 2005] inspiré du comportement intelligent des abeilles de miel en recherchant des sources de nourriture.

Principe :
La population dans l'algorithme ABC est organisée en trois groupes : les abeilles découvreuses, les observatrices et les exploratrices. Les découvreuses recherchent les sources de nourriture et reviennent avec danse autour de la ruche. Les observatrices choisissent les sources de nourriture selon les danses des abeilles découvreuses. Une abeille découvreuse est considérée pour chaque source de nourriture. Les abeilles dont la source de nourriture est abandonnée deviennent des exploratrices pour rechercher de nouvelles sources de nourriture qui pourront être meilleures [D.Karaboga et al, 2007a].

Formalisation
Cet algorithme se résume dans les étapes suivantes [D.Karaboga et al, 2007b] :
Le nombre des sources de nourriture est égal au nombre d'abeilles découvreuses qui sont initialisées au départ de manière aléatoire.

- Répéter :
    - Chaque abeille découvreuse détermine une source voisine de sa source de nourriture, puis évalue sa fitness (sa quantité de nectar) et commence à danser dans la ruche
    - Chaque observatrice choisit une des ressources de nourriture selon les danses des abeilles découvreuses, et puis détermine une source voisine autour de sa source de nourriture, et ensuite évalue sa fitness.
    - Les sources abandonnées (non optimisées pour un certain Nbr d'itérations) sont remplacées de manière aléatoire par des abeilles exploratrices.
    - La meilleure source trouvée sera mémorisée.
- Jusqu'au nombre maximal d'itérations

Dans l'algorithme ABC, les sources de nourriture sont les solutions possibles à un problème donné et leurs quantités de nectar correspondent à leurs valeurs de fitnesses. Cet algorithme commence par une population de solutions aléatoires. Une abeille découvreuse produit une nouvelle source de nourriture (solution) basée sur sa source initiale de nourriture. Si la nouvelle solution est meilleure que celle de la source initiale alors l'abeille découvreuse la garde en tant que source de nourriture. Chaque observatrice choisit une source de nourriture basée sur la danse (selon la fitness) des abeilles découvreuses. Alors chacun d'eux crée une nouvelle source de nourriture et vérifie sa fitness pour la garder ou la rejeter. Les sources de nourriture dont la fitness demeure inchangée pour un certain nombre d'itérations seront remplacées de manière aléatoire par des abeilles exploratrices [D.Karaboga, 2005].

Ci-dessous est l'adaptation de l'algorithme ABC pour la reconnaissance de caractères.

Algorithme 3 : L'optimiseur de la Colonie d'Abeilles Artificielles (ABC) pour la reconnaissance de caractères

Transformer l'ensemble de caractères à reconnaître en vecteurs d'attributs
Initialiser une population d'abeilles (où chaque abeille comporte les attributs caractéristiques des caractères aléatoirement choisis de la base d'apprentissage et chaque abeille possède une taille égale à la taille du vecteur des caractères à reconnaître).

Diviser la population en deux sous populations : les abeilles découvreuses et les abeilles observatrices
{**Pour** chaque caractère faire// traitement d'un caractère d'une abeille
{**pour** un nombre d'itérations faire
{**pour** chaque abeille découvreuse faire
Créer une nouvelle abeille dans son voisinage
Évaluer sa fitness comme mentionné précédemment
{**Si** la nouvelle abeille est mieux que l'abeille découvreuse alors
remplacer l'abeille découvreuse par sa voisine
Sinon
incrémenter sa valeur d'exploitation}}

{**pour** chaque abeille observatrice **faire**
Basé sur un calcul de probabilités choisir une source de nourriture

Créer une nouvelle abeille dans le voisinage de cette ressource

Évaluer sa fitness
{**Si** la nouvelle abeille est mieux que l'abeille observatrice alors
remplacer l'abeille observatrice par sa voisine
Sinon
incrémenter sa valeur d'exploitation}}

{**Si** la valeur d'exploitation d'une découvreuse ou d'une observatrice est > valeur-limite abandonner leurs ressources de nourriture en les initialisant de manière aléatoire}}}

Employer les attributs trouvés comme centres de groupes pour les étiqueter en choisissant les caractères de la base d'apprentissage qui donnent la valeur minimale de la somme des erreurs quadratiques (SSE).

Les étiquettes des caractères qui donnent la valeur minimale de la SSE constituent la solution recherchée.

10.6 Résultats expérimentaux

Pour évaluer les performances des algorithmes proposés, différents types de chiffres manuscrits ont été employés comme les chiffres manuscrits obtenus à partir de la base de caractères MNIST ou des échantillons de lettres manuscrites extraites de [D. Dileep Gaurav, 2009].

Les chiffres et les lettres manuscrites :

La reconnaissance de chiffres manuscrits est au même niveau de difficulté de la reconnaissance de caractères manuscrits. La seule distinction entre eux réside dans la taille du vocabulaire ; les chiffres manuscrits ont un vocabulaire très restreint (10 chiffres).

La base notée « MNIST » de chiffres manuscrits présente l'avantage d'être une base de référence dans la communauté scientifique [Y.LeCun et al, 1998].

Les images montrées ci-dessous tirées de la base MNIST indiquent une idée sur la base d'apprentissage employée.

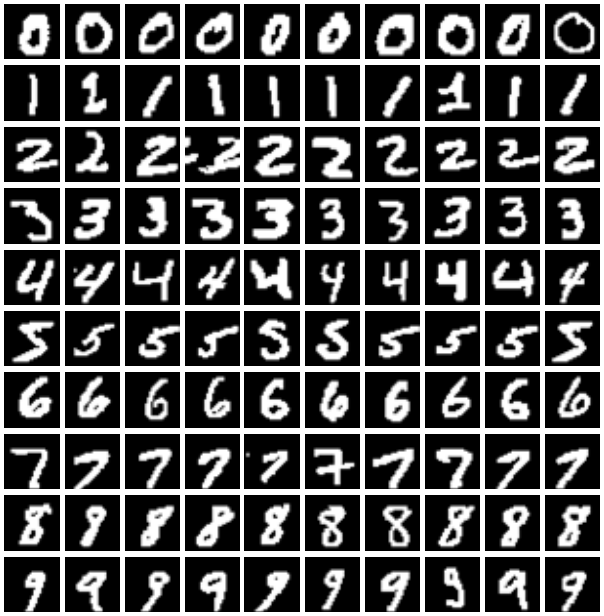


Figure 10.1 Un échantillon des chiffres de la base MNIST [Y. LeCun,1998a]

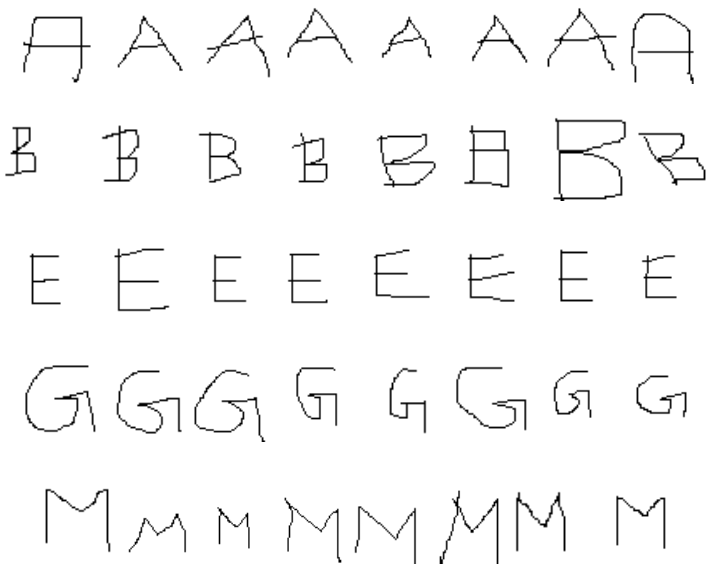


Figure 10.2 Un échantillon des lettres manuscrites [D. Dileep Gaurav, 2009]

Les paramètres initiaux pendant nos expériences sont résumés dans la table ci-dessous :

Table 10.1: Les paramètres initiaux des classificateurs statistiques cas des chiffres

La classification par PSO	La classification par BA	La classification par ABC
Nombre de particules : 200	Nombre des abeilles : 200	Nombre limite d'itérations pour lancer la recherche des exploratrices : 20
Le poids d'inertie : varie de 0.78 à 0.1	Nombre de sites : 40 Nombre de sites élites : 20	
Les facteurs cognitif et social : 1.49	Nombre des abeilles autour chaque site : 50 Nombre des abeilles autour chaque site élite : 80	Nombre des abeilles observatrices : 100
	La taille d'un site : 0.0234	Nombre des abeilles découvreuses : 100
8000 chiffres pour l'apprentissage et 2000 autres pour le test		

Table 10.2 : Les paramètres initiaux des classificateurs statistiques cas des lettres

La classification par PSO	La classification par BA	La classification par ABC
Nombre de particules : 200	Nombre des abeilles : 100	Nombre limite d'itérations pour lancer la recherche des exploratrices : 10
Le poids d'inertie: varie de 0.78 à0.1	Nombre de sites : 20 Nombre de sites élites : 10	
Les facteurs cognitif et social : 1.49	Nombre des abeilles autour chaque site : 25 Nombre des abeilles autour chaque site élite: 40	Nombre des abeilles observatrices : 40
	La taille d'un site : 0.0234	Nombre des abeilles découvreuses : 40
260 lettres pour l'apprentissage et 234 autres pour le test		

L'approche ABC a été adaptée avec quelques modifications ; nous avons employé la notion de voisinage de l'algorithme BA au lieu de choisir aléatoirement des solutions possibles en tant que sources voisines. De plus, nous avons employé la somme des erreurs quadratiques entre les caractères à reconnaître et les caractères d'apprentissage comme fitness sans changement tel qu'il est apporté dans l'algorithme ABC initial. La probabilité des sources préférées par les abeilles observatrices est : (la fitness d'une abeille découvreuse/Somme de toutes les fitness des abeilles découvreuses). Avec ces modifications, nous avons trouvé l'algorithme ABC plus robuste (stable) quand il est appliqué à la reconnaissance de caractères manuscrits.

Dans la table 10.3, nous présentons les résultats obtenus après 5 essais pour chaque algorithme.

Table 10.3: Résultats des classificateurs statistiques

	Le taux de reconnaissance		
La méthode de classification	Zonage	Moment de Hu	Attributs structurels et géométriques
PSO (100 itérations)	80.428 %	92.2%	81.20%
ABC (100 itérations)	80 %	84.49%	70%
BA (100 itérations)	92.857 %	99.8%	82.48%
K-nn	90%	98%	82.48%
		Les chiffres	**Les lettres d'anglais**

On observe que le classificateur BA donne de meilleurs résultats par rapport à PSO. Par ailleurs, on voit que les moments de Hu contribue à une meilleure classification par rapport au zonage dans la classification des chiffres manuscrits. Les caractéristiques structurelles et géométriques sont les meilleures pour la distinction des lettres.

Les graphes ci-dessous exposent le comportement de ces classificateurs c-à-d. la minimisation de la somme des erreurs quadratiques (SSE) durant la phase d'apprentissage pendant un nombre d'itérations où les moments de Hu sont employés en tant que vecteurs d'attributs caractéristiques.

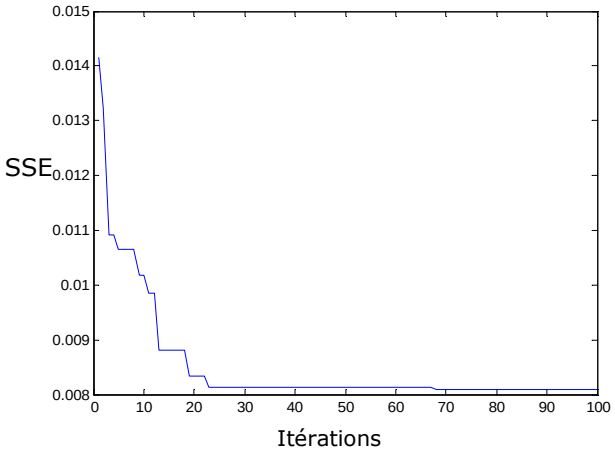


Figure 10.3. Le comportement de l'algorithme ABC: Le taux de reconnaissance 80.9%

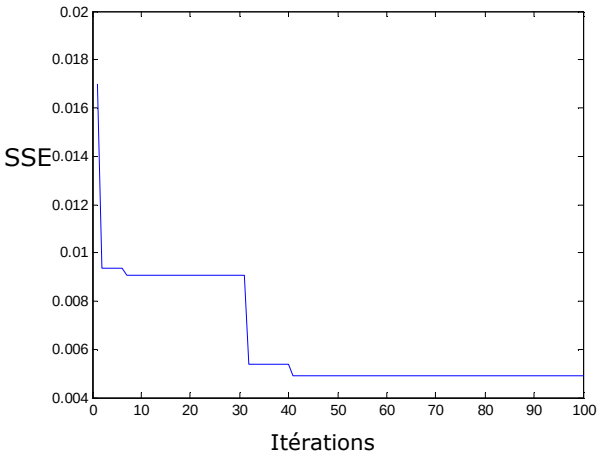


Figure 10.4. Le comportement de l'algorithme BA : taux de reconnaissance 99.85%

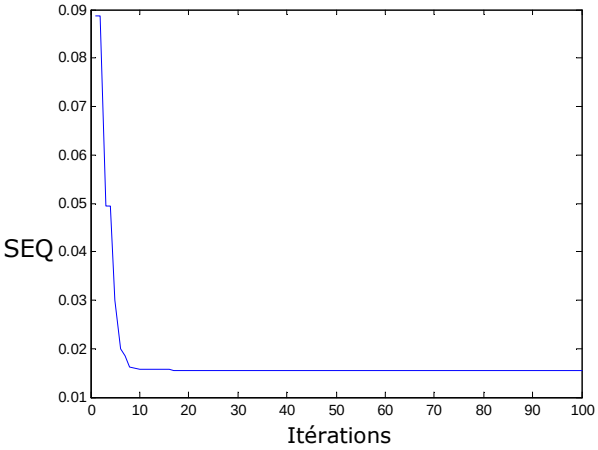


Figure 10.5. Le comportement de l'algorithme PSO : taux de reconnaissance 90.65%

A partir de ces résultats, on voit que l'approche basée sur l'algorithme d'abeilles (BA) donne les meilleurs résultats et l'algorithme PSO donne des résultats qui sont mieux que ceux de l'algorithme ABC en traitant le problème de la reconnaissance de chiffres manuscrits. En outre, on a constaté que les moments de Hu contribuent à une meilleure classification que celle obtenue par les attributs zonage.

Les figures illustrées ci-dessus montrent que PSO atteint sa valeur minimale après 20 itérations seulement alors qu'ABC atteint presque le même niveau au bout de 30 itérations. Mais l'algorithme BA dépasse 40 itérations pour sa convergence avec un gain d'environ 10% dans le taux de reconnaissance.

Dans l'algorithme des abeilles, le nombre d'abeilles impliqué accélère nettement cette convergence, mais augmente le temps de calcul tandis qu'un petit nombre d'abeilles diffusées dans l'espace de recherche peuvent ne pas converger l'algorithme vers une solution satisfaisante. De plus, il est important de dire que nous avons trouvé dans la base MNIST un certain nombre de chiffres très similaires appartenant à différentes classes (c'est-à-dire, ayant le même vecteur d'attributs mais étiquetées de manière différente ce qui pose le problème de la confusion entre les différentes classes et ainsi diminuer le taux de reconnaissance)

Par exemple, nous avons trouvé :

 qui sont similaires au chiffre 6 mais étiquetés en tant que chiffre 5.

 est similaire au chiffre 4 mais étiqueté en tant que chiffre 8.

 est similaire au chiffre 8 mais étiqueté en tant que chiffre 3... et bien d'autres.

Après l'élimination de ces chiffres ambigus, un taux de reconnaissance de 100% a été parvenu par l'approche basée BA en utilisant 7500 chiffres pour l'apprentissage et 2000 chiffres pour le test. On constate aussi que la méthode BA est supérieure dans le cas des lettres manuscrites que ce soit en termes de vitesse ou d'exactitude.

Conclusion

10.7 Conclusion

Le présent travail propose des méthodes d'optimisation inspirées de la nature pour traiter le problème de la reconnaissance de chiffres arabes manuscrits et de lettres anglaises manuscrites. En incorporant des idées d'interaction collective observée dans la nature telle que le vol des oiseaux et le comportement collectif des abeilles quand elles cherchent du nectar ou de la nourriture. Ces algorithmes ont été appliqués afin de maintenir la diversité et permettre de ce fait une meilleure exactitude en termes du taux de la reconnaissance. L'étude comparative a montré que l'algorithme d'abeilles BA offre des capacités intéressantes et significatives en traitant le problème de la reconnaissance de chiffres et lettres manuscrites, et les autres deux méthodes donnent des résultats comparables mais aussi satisfaisants. Néanmoins, l'inconvénient majeur de ces algorithmes de la classification statistique est qu'ils prennent de très longue durée pendant la phase du test, ce qui rend leur application réelle impossible. Pour résoudre ce problème nous avons choisi de travailler dans les chapitres suivants avec les réseaux de neurones.

Chapitre 11

UNE APPROCHE COOPERATIVE COEVOLUTIONAIRE POUR LA RECONNAISSANCE DE CARACTERES MANUSCRITS

11.1 Introduction

Les algorithmes Co-évolutionnaires constituent une classe des métaheuristiques de recherche adaptatives qui sont inspirées du mécanisme des bénéfices réciproques entre les espèces dans la nature. Le travail de ce chapitre propose un algorithme Coopératif Co-évolutionnaire pour améliorer les performances de deux types de réseaux de neurones notamment le perceptron multi couche (MLP) et le réseau de neurones de fonctions à base radiale (RBF) quand ils se sont appliqués au problème de la reconnaissance de caractères manuscrits. Ce travail est en fait une combinaison de dix réseaux de neurones MLP (RBFs) où chaque réseau est considéré en tant que classificateur expert permettant la distinction d'une classe parmi les 9 (25) autres classes; chaque classificateur MLP (RBF) adapte les attributs caractéristiques de son entrée et sa structure comprenant le nombre des centres, leurs positions et les poids d'un RBF ou les poids et biais d'un MLP à travers l'algorithme symbiotique. L'ensemble des attributs caractéristiques et les centres d'un RBF ont été considérés en tant qu'espèces différentes où chacune d'elles peut tirer bénéfice de l'autre, imitant de manière simplifiée l'interaction symbiotique des espèces dans la nature. La Coévolution est fondée sur le principe de sauvegarde des meilleurs poids et centres d'un RBF ou poids et biais d'un MLP qui ont fourni le maximum d'amélioration de la somme des erreurs quadratiques après un temps d'apprentissage. La qualité des résultats a été estimée et comparée à d'autres expérimentations de MLP et RBF standard. Les résultats sur les chiffres de la base MNIST et des lettres anglaises démontrent la supériorité de l'approche Co-évolutionnaire.

Ce chapitre expose deux approches co-évolutionnaires à base MLP et RBF ainsi que les expérimentations examinant leur efficacité. Les performances de ces algorithmes sont aussi évaluées et comparées à d'autres méthodes standard bien connues.

11.2 Le perceptron multi couche

Le perceptron multi couche (MLP) est un classificateur neuronal supervisé qui se compose d'une couche d'entrée dont la taille est égale à la taille d'une donnée d'entrée, une ou plusieurs couches cachées dont la taille est déterminée par expérimentation et la couche de sortie dont la taille est égale au nombre de classes du problème. Dans les réseaux MLP, chaque neurone est connecté à un certain nombre d'entrées qui peuvent être les données d'entrée ou les sorties des couches précédentes.

De chaque neurone, l'information se propage à la prochaine couche à travers une fonction d'activation (généralement une fonction sigmoïde) appliquée au produit scalaire des entrées et les poids correspondants.

$$net_j = biais * w_{biais} + \sum_k O_{pk}\left(w_{jk}\right) \qquad (11.1)$$

$$O_{pj}\left(net_j\right) = \frac{1}{1 + e^{-\lambda net_j}} \qquad (11.2)$$

"O_{pj}" : est la sortie du neurone "j" pour l'exemple d'apprentissage "p" utilisant la fonction d'activation sigmoïde
"j" : le neurone courant
"p" : indique l'exemple courant
"w_{jk}" : Le poids synaptique de l'entrée k au neurone "j"
"net_j : Total des poids du neurone j

11.3 Réseaux de neurones de fonctions à base radiale

Un réseau de neurones RBF est constitué de trois couches : La couche d'entrée transmet les entrées à la couche cachée, la couche cachée contient des neurones RBF et la couche de sortie applique une combinaison linéaire. Un réseau RBF contient des neurones RBFs qui sont très sensibles aux entrées proches de leurs centres.
Les neurones de la couche cachée d'un réseau de neurones RBF sont des fonctions à base radiales. Au lieu de calculer une somme pondérée de ses entrées comme dans le cas des réseaux de neurones MLP, un neurone RBF compare chaque entrée à une valeur centrale. lorsque la valeur d'entrée est très proche de la valeur centrale, il produit la plus grande valeur de sortie [Z. Yang et al, 2007].
Chaque entrée xi est ainsi associée à une valeur centrale Ci. La comparaison entre les entrées et les centres est généralement faite à travers le calcul de distances euclidiennes. Le neurone transforme ainsi la valeur obtenue à travers une fonction d'activation. La fonction d'activation est dite radiale parce qu'elle a une symétrie radiale autour d'un point central : Autrement dite, sa

sortie demeure inchangée au vu d'une rotation autour de ce point. Dans la pratique, la fonction gaussienne est souvent employée comme fonction d'activation.
Un réseau de neurones RBF avec m sorties et n_h neurones cachés peut être exprimé par la formule ci-dessous :

$$y_i(t) = w_{i0} + \sum_{j=1}^{n_h} w_{ij}\,\delta\left(\left\|x(t) - C_j(t)\right\|\right),\quad i = 1, \ldots, m \qquad (11.3)$$

Où w_{ij}, w_{i0}, $C_i(t)$ sont respectivement les poids, les biais et les centres RBF, $x(t)$ est la donnée d'entrée. δ est une fonction à base radiale. $\|\ \|$ est une mesure de distance telle que la norme euclidienne [Z. Yang et al, 2007].

11.4 Formulation du problème

Dans la plupart des systèmes de la reconnaissance de caractères utilisant les réseaux de neurones, les étapes de la binarisation, de la normalisation suivie de la squelettisation sont nécessaires avant l'étape d'extraction de primitives pour ne garder que la structure de base caractérisant un caractère. La squelettisation est une technique adoptée dans l'étape de prétraitement pour corriger les effets d'épaisseur des caractères causés par les moyens d'écriture et les différents styles d'écriture. La squelettisation que nous avons employée est basée sur l'opération d'amincissement morphologique Matlab suivante :

```
bwmorph(image_caractère,'skel',inf);
```

« `bwmorph` » enlève les pixels du contour d'un caractère sans permettre sa fragmentation et jusqu'à aucun changement d'épaisseur n'est permis.

Nous devons d'abord choisir une représentation appropriée permettant la conception du problème en termes de l'algorithme co-évolutionnaire.

11.4.1 La représentation à base RBF

Nous avons combiné dix réseaux de neurones RBFs, chacun est responsable de la distinction d'une classe parmi les 9 (25) autres classes, ainsi, chaque RBF est entraîné de manière indépendante à travers l'algorithme symbiotique Co-évolutionnaire (expliqué ci-dessous) pour trouver ses meilleurs centres et poids, ces dix (26) RBFs sont combinés pour identifier les dix classes (de 0 à 9) ou 26 (de A…Z). En d'autres termes, dans l'étape du test, un caractère est assigné à la classe du classificateur qui est capable de le distinguer, c'est-à-dire la sortie de celui-ci est égale 1 pour cette classe et 0 pour toutes les autres classes.

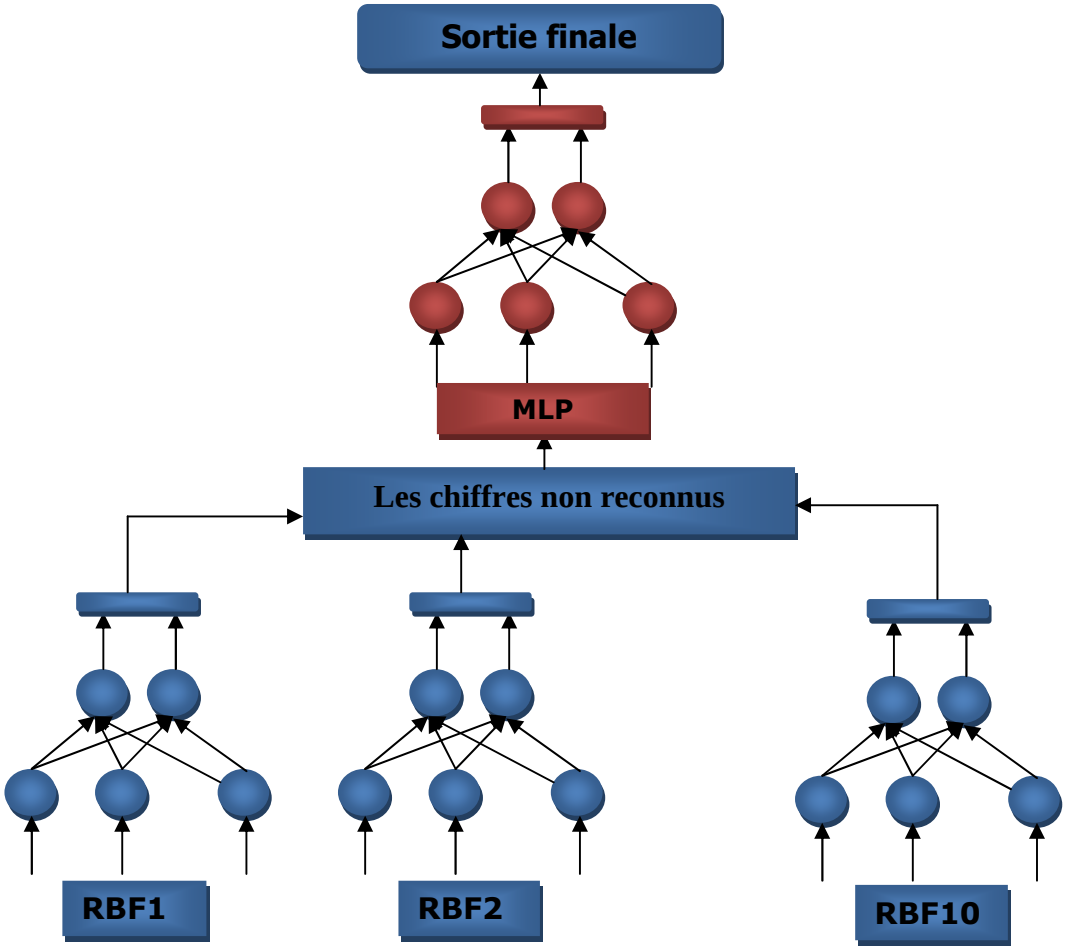


Figure 11.1: schématisation du système coopératif co-évolutionnaire pour les chiffres

Par exemple, la sortie du premier RBF permettant la distinction du chiffre 0 est :

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
.....

La sortie du deuxième RBF permettant la distinction du chiffre 1 est :

0 0 0 0 0 0 0 0 0 0
0 **1** 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
.....

La sortie du troisième RBF permettant la distinction du chiffre 2 est :

0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 **1** 0 0 0 0 0 0
.....
Et ainsi de suite.

Nous avons considéré les sorties sous cette forme afin de pouvoir identifier les chiffres inconnus pendant l'étape du test. (Un chiffre inconnu est celui qui n'a pas pu être reconnu par l'un des dix (26) classificateurs RBF).
En outre, nous avons considéré un autre réseau de neurones de type MLP qui doit produire la valeur de sortie égale à 1 pour chaque classe comme indiqué ci-dessous :

1 0 0 0 0 0 0 0 0 0
0 **1** 0 0 0 0 0 0 0 0
0 0 **1** 0 0 0 0 0 0 0
0 0 0 **1** 0 0 0 0 0 0
.....

Un réseau de type MLP a été choisi pour effectuer une classification finale afin d'améliorer le taux de reconnaissance en classifiant les chiffres qui n'ont pas été reconnus par tous les réseaux RBFs. Ce choix est basé sur l'idée que les MLPs sont plus efficaces que les RBFs en termes du taux de reconnaissance, mais ils prennent plus de temps pour l'apprentissage.

Lalgorithme symbiotique : est basé sur la co-évolution des deux espèces suivantes :
La première espèce : est un essaim de particules employé pour grouper les caractères d'entrée dans un ensemble de groupes cohérents.

Une particule est un vecteur d'un nombre variable de positions des centres de groupes comme montré ci-dessous :

c_1	c_2	$\cdots$	c_M

Où : c_1 ..... c_M sont M centres utilisés en tant que centres de la couche cachée du réseau RBF compagnon. La sélection des attributs caractéristiques dans le contexte du clustering est effectuée par l'algorithme PSO. L'algorithme du clustering que nous avons employé est un algorithme itératif basé sur la minimisation du critère indiqué par l'équation suivante

$$SSE = \sum_{i=1}^{M} \sum_{x \in C_i}^{N} \left\| x - \overline{C_i} \right\|^2 \qquad (11.4).$$

Où M : est le nombre de groupes
N : est le nombre de caractères par groupe
$\overline{C_i}$: sont les centres des classes C_i

L'algorithme divise les vecteurs d'attributs en un nombre de groupes tout en minimisant le critère d'homogénéité (11.4). Chaque vecteur d'attributs (un caractère) est affecté au groupe le plus proche qui a le centre le plus proche.

La deuxième espèce : est une population de réseaux de neurones RBF (différent nombre de centres avec différentes positions) coévoluant avec l'ensemble des attributs caractéristiques (caractères) pour trouver la meilleure structure c-à-d les centres des fonctions à base radiales (leurs positions et nombre), les poids et les biais qui ont causé la meilleure performance.

L'objectif de la première espèce est de diviser les caractères d'apprentissage en groupes cohérents ayant le même style d'écriture tout en réduisant le nombre de caractères d'apprentissage par sélection des centres de groupes issus du clustering comme étant des centres de la couche caché d'un RBF.

L'objectif principal de la coévolution est d'étudier l'effet d'adaptation de l'architecture d'un RBF avec leurs attributs caractéristiques à travers l'algorithme Co-évolutionnaire décrit dans [A.F. Silva et al, 2003]. La coévolution a lieu quand un RBF obtient sa fitness de leurs attributs compagnons, permettant à l'architecture d'un RBF de s'adapter à leurs attributs compagnons et ceci à travers la mise à jour du nombre et positions des centres d'un RBF par PSO.

La fitness d'un RBF (structure d'un RBF) est ainsi déterminée basée sur son interaction avec l'ensemble des attributs correspondants ; cette fitness est l'amélioration produite à la somme des erreurs quadratiques par le meilleur ensemble des centres, poids et biais correspondants comme montré ci-dessous :

Fitness (d'un RBF) = (l'ancienne valeur de fitness – sa nouvelle valeur de fitness).

La nouvelle valeur de fitness est calculée après apprentissage pour un nombre d'itérations. Le meilleur ensemble des centres et poids est l'ensemble qui a produit la plus grande valeur d'amélioration et l'action d'affectation de fitness à une certaine structure RBF se produit après un nombre d'itérations (ici après 30 itérations), sinon le nombre des centres et attributs restent inchangés pour améliorer la qualité des poids de connexion.

Cette approche peut être récapitulée comme montré ci-dessous

L'approche co-évolutionnaire à base RBF

Initialiser un ensemble de particules par des attributs caractéristiques choisis de manière aléatoire.

Pour chaque itération
Pour chaque particule de la 1ère espèce
Faire le clustering des caractères basé sur PSO
Si itération mod 30 == 0 (après 30 itérations du clustering)
- construire un nouveau RBF (un individu de la 2ème espèce) basé sur la meilleure particule (meilleurs centres) de la 1ère espèce.
- entraîner le nouveau réseau RBF
- évaluer sa performance
  Si ce nouveau RBF améliore la classification alors
    Sauvegarder les poids et les biais correspondants
    Sauvegarder les centres des fonctions radiales (leur nombre et leurs positions).
  Fin si
Fin % si *itération mod 30 == 0*
Fin % pour chaque particule

Pour chaque particule
  En utilisant les équations de PSO (10.1)
  Mettre à jour le nombre des centres
Fin % pour chaque particule
Fin % pour chaque itération

Combiner les meilleurs réseaux RBF (poids, biais et centres) trouvés pour le test
Déterminer les chiffres inconnus durant l'étape du test
Envoyer les chiffres inconnus au classificateur final (Le MLP) pour améliorer le taux de reconnaissance

11.4.2 La représentation à base MLP

La représentation de chaque MLP est très similaire à celle d'un RBF, La seule différence est que ici on ne cherche pas le nombre et positions optimales des centres des fonctions à base radiale, on cherche uniquement les meilleurs poids et biais pour chaque réseau MLP sans qu'il ya un classificateur MLP final pour améliorer le taux de reconnaissance.

Cette approche peut être résumée comme indiqué ci-dessous :

L'approche co-évolutionnaire à base MLP

Initialiser un ensemble de particules (de tailles variantes ou la taille est le nombre des centres) par des attributs caractéristiques.

Pour chaque itération
Pour chaque particule de la 1ère espèce
Faire le clustering des caractères par la méthode PSO
Si itération mod 30 == 0
Pour chaque espèce (MLP)
- faire l'apprentissage du MLP en utilisant les attributs sélectionnés comme une base d'apprentissage
- évaluer sa performance

Si le nouvel ensemble de caractères d'entrée améliore la classification alors
Sauvegarder les poids et les biais correspondants
Fin % si
Fin %Si *itération mod 30 == 0*
Fin % *pour chaque particule*
Fin % *pour chaque MLP*

Pour chaque particule
En utilisant les équations de PSO (10.1)
Mettre à jour le nombre et positions des attributs caractéristiques
Fin % *pour chaque particule*
Fin % pour chaque itération

Combiner les 10 meilleurs réseaux MLP (poids et biais) pour trouver le résultat du test

11.5 Résultats expérimentaux

Pour évaluer les performances des algorithmes proposés, différents types de chiffres manuscrits extraits de la base de chiffres MNIST disponible dans [Yann lecun, 1998] et de lettres manuscrites proposées par Dinesh Dileep Gaurav [D. Dileep Gaurav, 2009] ont été employés.

Les paramètres initiaux utilisés pendant nos expérimentations étaient :

Table 11.1: Initialisation des paramètres de l'approche co-évolutionnaire

Les paramètres	**Leur valeur**
Nombre d'itérations	500
Nombre de particules	20
Le poids d'inertie W	poids décroissant de [0.78 à 0.1]
C1 et C2	valeur aléatoire dans l'intervalle [0, 1.49].
Le nombre de centres	[50, 100]
Le nombre des attributs sélectionnés	[50, 100]
Le nombre des exemples d'apprentissage	8000 caractères (800 par classe).
Le nombre des exemples de test	autres 2000 caractères (200 par classe)
La fonction à base radiale	"poly harmonique"
L'extracteur de caractéristique	le zonage basé densité de pixels (chaque caractère a été subdivisé en 8 * 8 zones = 64 attributs de densité de Pixel dans chaque zone)

Pour les résultats suivants, nous avons fixé le nombre des centres à 80 pour réduire le temps de convergence.
Nous présentons ci-dessous les valeurs expérimentales obtenues sur le même ensemble de caractères de test :

Table 11.2 : Résultats de l'approche co-évolutionnaire

La Méthode d'apprentissage	Taux de reconnaissance
Le RBF pur (newrb de matlab)	71.80 %
L'approche symbiotique coopérative	88.79 %
Le MLP (Traingdx)	80 %
L'approche co-évolutionnaire à base MLP	86%

La table.11.2 est la table des résultats associés aux meilleures performances de chaque algorithme

A partir les résultats obtenus, on remarque que l'approche coopérative coévolutionnaire à base RBF donne des résultats prometteurs une fois comparés aux résultats obtenus par la fonction matlab : newrb en termes du taux de reconnaissance et la vitesse due à une meilleure exploration basée sur un petit sous-ensemble d'attributs et de centres à travers la notion de la coévolution.

En observant le pourcentage des caractères correctement classifiés, il est possible de remarquer que l'algorithme à base RBF et plus efficace comparativement à l'algorithme à base MLP. On peut dire que les deux modèles sont relativement similaires bien que l'algorithme RBF coévolutionnaire paraisse meilleur due à la stratégie localisée que suivent les RBF.

Conclusion

11.6 Conclusion

Le travail actuel propose une approche coopérative Co-évolutionnaire pour traiter le problème de la reconnaissance de chiffres manuscrits basée sur deux types de réseaux de neurones. Symbiose a été proposée pour trouver les meilleurs centres et poids d'un RBF ou les meilleurs poids et biais d'un MLP. Chaque réseau RBF (MLP) a été considéré comme un classificateur expert qui évolue de manière indépendante pour identifier une classe spécifique des chiffres puis coopèrent avec les autres RBF (MLP) pour la classification finale. La qualité des résultats a été estimée basée sur le taux de reconnaissance. Les résultats démontrent que les algorithmes Co-évolutionnaires offrent des capacités intéressantes comparativement à l'algorithme newrb et l'algorithme de la rétropropagation de gradient Traingdx implémentés en matlab.

En outre, les résultats de l'approche symbiotique à base RBF sont les meilleurs comparativement à ceux de l'approche à base MLP due à la stratégie localisée que suivent les RBFs basés sur le calcul de distance par rapport aux centres des fonctions radiales, alors que l'information apprise par un MLP est distribuée en tant que poids de connexions. Cette information pourra ainsi être détruite quand on fait la mise à jour de ces poids par la méthode PSO.

Chapitre 12

LA RECONNAISSANCE DE CARACTERES MANUSCRITS BASEE RNS ET L'INTELLIGENCE EN ESSAIM

12.1 Introduction

Les méthodes de la descente de gradient sont des méthodes supervisées qui sont largement utilisées pour l'apprentissage du perceptron multi couche due à leur convergence rapide avec un taux de reconnaissance raisonnable. Cependant, elles peuvent converger vers une solution sous optimale plus particulièrement si la tâche à résoudre est difficile telle que la reconnaissance de caractères manuscrits. L'utilisation du calcul évolutionnaire peut apporter plus d'efficacité aux réseaux de neurones incapables d'adapter leur structure et poids avec le problème à résoudre et ainsi incapable d'échapper des optimums locaux. En outre, les réseaux de neurones évolutionnaires ont été justifiés efficaces dans la reconnaissance de caractères. Plus particulièrement, pour faire l'apprentissage des réseaux de neurones [R.Kaur et al, 2011], [M.Mangal et al, 2006], [R.Ghosh et al, 2005], l'optimisation de leurs paramètres [L. S. Oliveira, 2005], l'optimisation de leurs structures [Y. Li et al, 2006], [R. Kala et al, 2010], l'optimisation de leurs poids [S.Guangmin et al, 2010], l'optimisation de poids et structures [T. Binos, 2002], [H.El Fadili et al, 2005] ou la sélection des attributs caractéristiques [C.Gagné et al, 2006], [L. Cordella et al, 2008]. Un système neuro-évolutionnaire fourni un taux de reconnaissance plus élevé comparé aux anciennes méthodes basées sur la rétro-propagation de gradient.

Les algorithmes d'optimisation globale tels que PSO ont gagné plus d'attention dans le domaine des réseaux de neurones artificiels. Les raisons derrière leur utilisation consistent principalement en leur capacité d'explorer de grands espaces de recherche avec une connaissance minimum au sujet de la fonction objective à optimiser. Dans cette issue, les travaux de ce chapitre décrivent des hybridations des algorithmes d'intelligence en essaim et les réseaux de neurones MLPs et RBFs pour traiter la reconnaissance de caractères manuscrits. Nous allons, dans un premier temps, employer PSO comme étape du clustering des exemples de la base de caractères. Ensuite, nous proposons des techniques de combinaison de classificateurs notamment une combinaison série d'un classificateur à base MLP suivie d'un classificateur statistique basé sur l'algorithme d'abeille (BA), et la combinaison de trois réseaux de neurones dont les entrées sont différentes issues de différents types d'extracteurs d'attributs caractéristiques.

12.2 Formulation du problème :

Les systèmes de reconnaissance de ce chapitre se résument dans les étapes suivantes que nous allons décrire par la suite :

- Le prétraitement
- L'extraction de caractéristiques
- La classification

L'étape du prétraitement

Nous avons employé les étapes du prétraitement suivantes :

- La binarisation à travers la méthode Otsu
- Extraction de chaque caractère de son image en éliminant les espaces blancs.
- La squelettisation
- La normalisation de la taille des images traitées.

L'étape d'extraction de caractéristiques

Cette opération consiste à rassembler dans un vecteur plusieurs attributs qui caractérisent de manière unique chaque classe des caractères de la base d'apprentissage. Le but de cette étape est d'extraire les attributs qui permettent de maximiser le taux de reconnaissance avec la moindre quantité d'information. Après la détermination des vecteurs d'attributs, il est possible de les affecter aux classes possibles des caractères.
Dans ce travail, nous avons choisi trois types d'attributs caractéristiques : le zonage, les moments de Hu et les attributs structurels & géométriques.

L'étape de la classification

Nous présentons par la suite les approches développées pour effectuer cette étape. Notamment : le clustering de caractères pour l'apprentissage du MLP et RBF, la combinaison série MLPS (RBFs)-BA et la combinaison de plusieurs réseaux de neurones basée sur différents types des extracteurs caractéristiques.

12.2.1 Le Clustering de caractères

Le clustering (classification) de caractères est une tâche principale dans la reconnaissance de caractères qui vise à diviser les caractères d'une base d'apprentissage en différentes classes tels que les caractères d'une même classe sont assez similaires que possible et les caractères de différentes classes sont assez dissimilaires que possible.

Dans cette section, nous explorons un mécanisme d'adaptation de réseaux de neurones avec des lois d'évolution différentes à celles que nous avons décrites précédemment. Plus précisément, le mécanisme d'adaptation mis en œuvre ici consiste à faire progressivement décroître la somme des erreurs quadratique (SSE) sur l'ensemble sélectionné des caractères d'apprentissage. La minimisation du SSE se fait à travers le clustering via la méthode PSO (ou BA) pour faire l'apprentissage des réseaux de neurones (MLP ou RBF).

12.2.1.1 Le Principe de l'approche Clustering-MLP

La tâche du clustering que nous avons employée consiste à trouver un nombre prédéterminé de groupes avec la moindre valeur de la variabilité intra-cluster et la plus grande valeur de la variabilité inter-cluster. Ce travail vise à trouver l'ensemble des centres de clusters qui réduisent au minimum le critère de la somme des erreurs quadratique (SSE). L'idée principale est que les centres de clusters sont des vecteurs d'attributs caractéristiques qui se déplacent dans l'espace de la base d'apprentissage selon une nouvelle stratégie (la dynamique de la méthode PSO ou BA).

$$SSE = \sum_{i=1}^{M} \sum_{x \in C_i} \|x - \bar{C}_i\|^2 \qquad (12.1)$$

où M: est le nombre de groupes
N: est le nombre de caractères par groupe.
Ci: sont les centres de groupes

Le principe de base de cette approche est très similaire aux algorithmes du 10 ème chapitre, la seule différence réside dans la fitness employée pour trouver ou sélectionner des caractères.

Ci-dessous est un pseudo code de l'approche clustering MLP basée PSO:

Convertir la base d'apprentissage en un ensemble d'attributs caractéristiques
Initialiser la population des particules avec des centres de classes choisis aléatoirement de la base d'apprentissage
{**Pour** chaque caractère faire
{**Pour** chaque particule faire
{**Pour** un nombre d'itérations faire
Calcul de la distance de chaque caractère par rapport à chacun des centres de classes
Affectation de chaque caractère au centre le plus proche
Évaluation de la performance de la particule courante selon (12.1)
Mise à jour de P_{best} et G_{best}
Mise à jour de la position et de la vitesse}}}

Les meilleurs centres trouvés seront employés en tant qu'ensemble d'apprentissage
Apprentissage du réseau MLP avec l'ensemble d'apprentissage trouvé

Ci-dessous est l'adaptation de l'algorithme clustering basé BA pour l'apprentissage d'un MLP :

Convertir la base d'apprentissage en un ensemble d'attributs caractéristiques
Initialiser les particules avec des centres de classes choisis aléatoirement de la base d'apprentissage

{**Pour** un nombre d'itérations **faire**
{**Pour** chaque caractère à reconnaitre **faire**
{**Pour** P=1 : nombre-abeilles-élite **faire**
Calcul de la distance de chaque caractère par rapport à chacun des centres de l'abeille élite
Affectation de chaque caractère au centre le plus proche
Évaluation de la performance de l'abeille élite courante selon (12.1)

{**Pour** F=1 : nombre-voisines-élite **faire**
Créer une abeille dans le voisinage de l'abeille élite
Calculer sa fitness
Si sa fitness est mieux que la fitness d'élite
Remplacer l'abeille élite par l'abeille voisine}}

{**Pour** P= (nombre-abeilles-elite + 1) : nombre-abeilles-site **faire**
Calcul de la distance de chaque caractère par rapport à chacun des centres de l'abeille site
Affectation de chaque caractère au centre le plus proche
Évaluation de la performance de l'abeille site courante selon (12.1)

{**Pour** F=1 : nombre-voisines-site **faire**
Créer une abeille dans le voisinage de l'abeille site
Calculer sa fitness
Si sa fitness est mieux que la fitness de l'abeille site
Remplacer l'abeille site par l'abeille voisine}}
Assigner les abeilles restantes de la population d'abeilles scoutes de manière aléatoire et calculer leurs valeurs de fitness
Trier l'ensemble des abeilles selon leurs valeurs de fitness}}

Les meilleurs centres trouvés seront employés en tant qu'ensemble d'apprentissage
Apprentissage du réseau MLP avec l'ensemble d'apprentissage trouvé

12.2.1.2 Principe de l'approche Clustering-RBF

L'algorithme procède globalement de la même manière du perceptron multi couche. La différence tient au fait que l'ensemble des centres de groupes issus du clustering sont employés en tant que centres des fonctions à base radiale du réseau RBF.

Ci-dessous est un pseudo-code de l'approche proposée :

Convertir la base d'apprentissage en un ensemble d'attributs caractéristiques
Initialiser la population des particules avec des centres de classes choisis aléatoirement de la base d'apprentissage
{**Pour** chaque caractère faire
{**Pour** chaque particule faire
{**Pour** un nombre d'itérations faire
Calcul de la distance de chaque caractère par rapport à chacun des centres de classes
Affectation de chaque caractère au centre le plus proche
Évaluation de la performance de la particule courante selon (12.1)
Mise à jour de P_{best} et G_{best}
Mise à jour de la position et vitesse}}}
Employer les centres trouvés en tant que centres de la couche cachée du réseau RBF
Apprentissage du réseau RBF

12.2.2 La combinaison série MLPs(RBFs) + BA

Dans cette section, le modèle basé sur l'algorithme des abeilles est établi. Le modèle est utilisé pour estimer les performances sur deux types de caractères notamment les chiffres et les lettres. Cette approche est une combinaison série de deux classificateurs le premier est un MLP (RBF) dont l'entraînement se fait par l'algorithme Traingdx (Newrb) et le second est un classificateur statistique basé sur l'algorithme des abeilles (BA) ce qui a permis d'augmenter le taux de reconnaissance. Pour ce faire, nous avons considéré dix MLPs (RBFs) dans le cas des chiffres et 26 réseaux dans le cas des lettres anglaises. La sortie de chaque MLP (RBF) est 1 pour la classe du chiffre (lettre) qui lui est associée et 0 pour toutes les autres classes des autres chiffres (lettre),telle qui'il est décrit dans l'algorithme co-évolutionnaire.

Par exemple, la sortie du premier MLP (RBF) qui ne reconnaît que le chiffre 0 est :

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
.....

La sortie du deuxième MLP(RBF) qui reconnaît le chiffre 1 est :

0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
.....

La sortie du troisième MLP (RBF) qui reconnaît le chiffre 2 est :

0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0
.....

Et ainsi de suite.

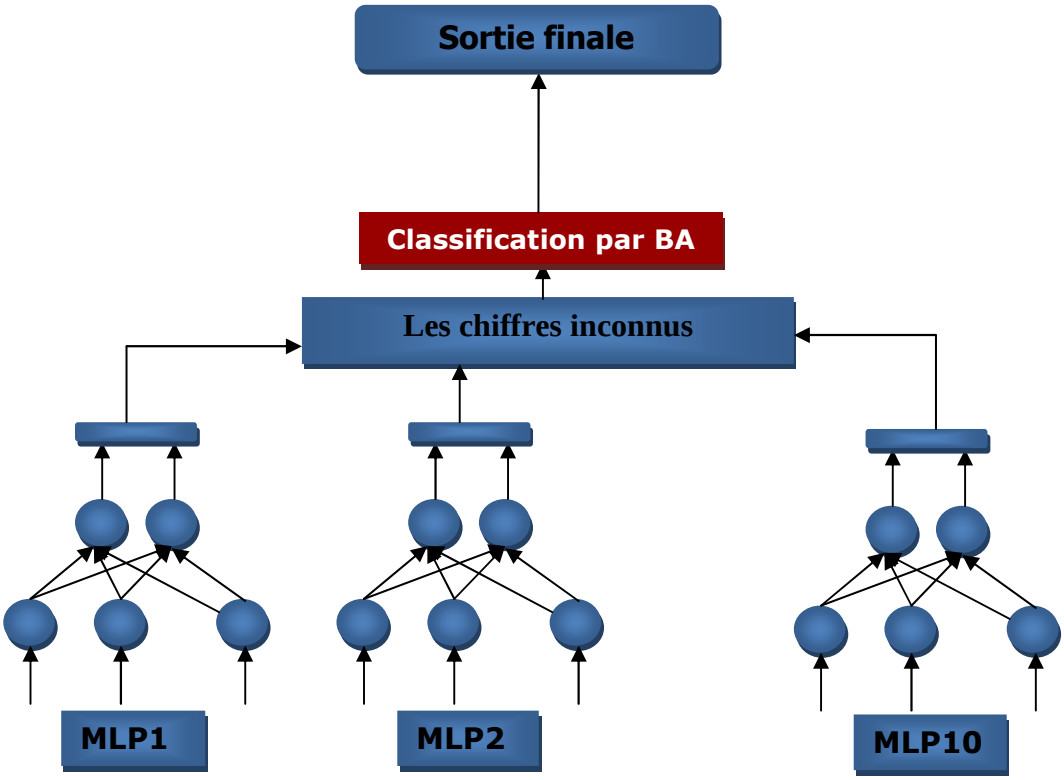


Figure 12.1 : Schématisation de l'approche MLP + BA

Nous avons considéré cette structure afin de pouvoir déterminer les chiffres inconnus pendant l'étape du test (le caractère inconnu est le caractère non classifié par tous les réseaux MLP (RBF)).

Les premières avancées dans ce type d'architectures modulaires sont établies dans l'article [S.Muthuraman, 2005] qui montre qu'un réseau de neurones modulaire adopte une meilleure stratégie de classification qu'un simple réseau de neurones.

En résumé, l'approche développée suit les étapes ci-dessous :

Algorithme : La combinaison série MLP(RBF) + BA
1. Transformer l'ensemble des caractères (apprentissage +test) en vecteurs d'attributs caractéristiques 2. Faire l'apprentissage des réseaux de neurones MLP (RBF) 3. Tester les dix (26) réseaux et déterminer les caractères inconnus 4. Envoyer ces caractères comme entrées au classificateur BA que nous avons décrit dans le 10 ème chapitre.

12.2.3 La combinaison de plusieurs classificateurs

La combinaison de plusieurs classificateurs où chaque classificateur traite un type spécifique des attributs caractéristiques a donné des résultats prometteurs comme nous allons réaffirmer à travers les résultats de ce travail. Nous avons employé trois types d'extracteurs caractéristiques ; chaque extracteur représente l'entrée d'un classificateur MLP (RBF). Le premier est un extracteur d'attributs structurels et géométriques, le second permet l'extraction des attributs par zonage et le dernier ne fait que la normalisation de la taille des caractères.

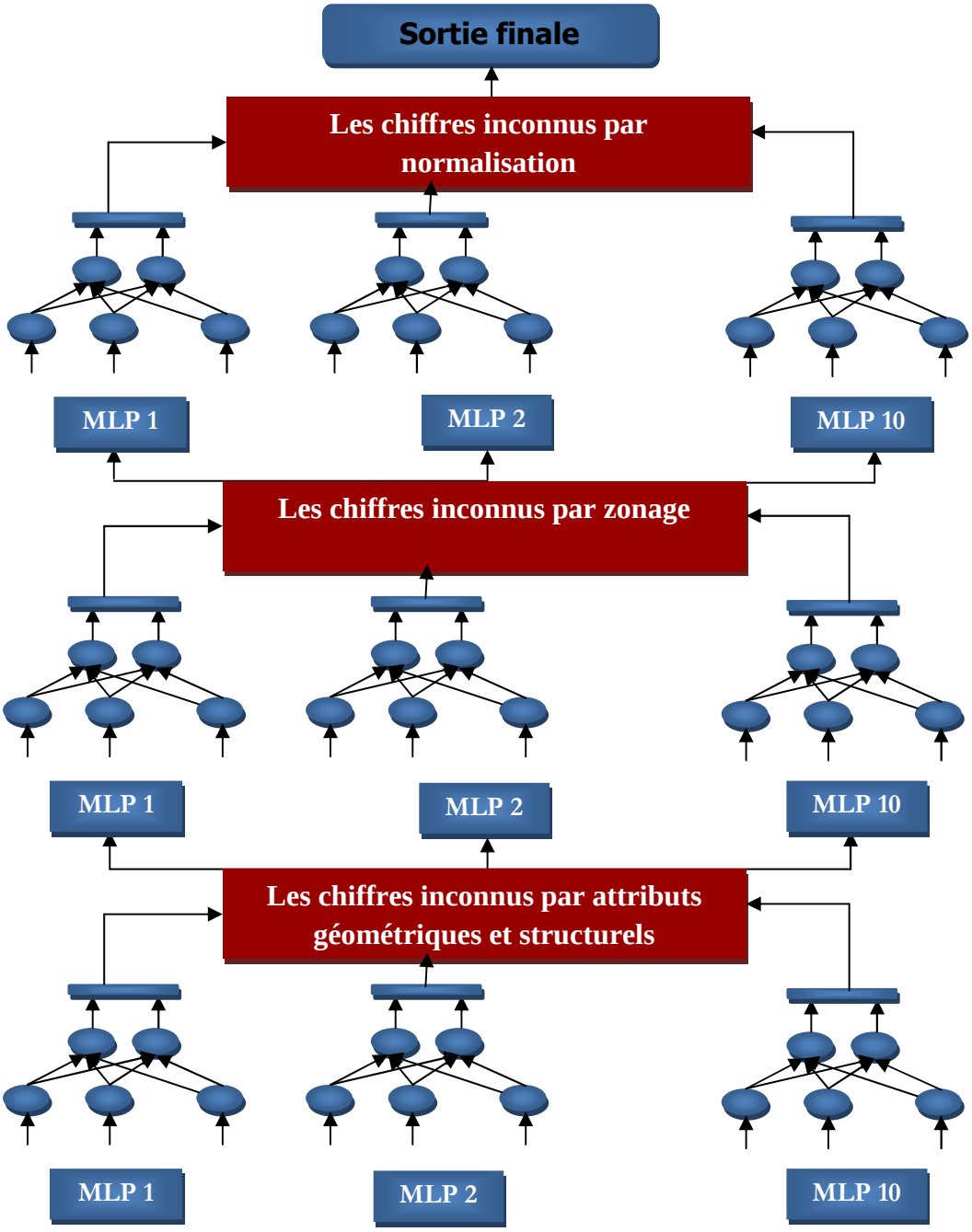


Figure 12.2 : schématisation de la combinaison de trois classificateurs MLP modulaires, taux de reconnaissance = **99.65%**

Le principe de la combinaison des sorties des différents réseaux de neurones est simple. Il est basé sur l'idée qu'un caractère doit être classifié par un et un seul réseau de neurones sinon il sera ajouté à la liste des caractères inconnus qui seront ensuite envoyés au prochain classificateur et ainsi de suite.

Cette approche suit les étapes suivantes :

Algorithme : Combination de trois classificateurs

1. Faire l'apprentissage du 1 er classificateur c-à-d les 10 (26) MLPs (RBFs) en utilisant traingdx (newrb) comme méthodes d'apprentissage et les caractères normalisés comme vecteurs d'attributs.
2. Faire l'apprentissage du 2 ème classificateur de la même manière mais avec les attributs de zonage.
3. Faire l'apprentissage du 3 ème classificateur avec les attributs structurels et géométriques.
4. Tester le 1 er classificateur et déterminer les caractères inconnus.
5. Envoyer ces caractères inconnus au 2 ème classificateur.
6. Test le 2 ème classificateur et déterminer le reste de caractères inconnus.
7. Envoyer le reste des caractères inconnus au 3 ème classificateur pour la décision finale.

12.3 Résultats expérimentaux
Résultats du clustering de caractères :

Nous avons employé les mêmes paramètres indiqués dans le chapitre 10. Les graphes ci-dessous montrent le comportement de chaque algorithme indiqué ci-dessous.

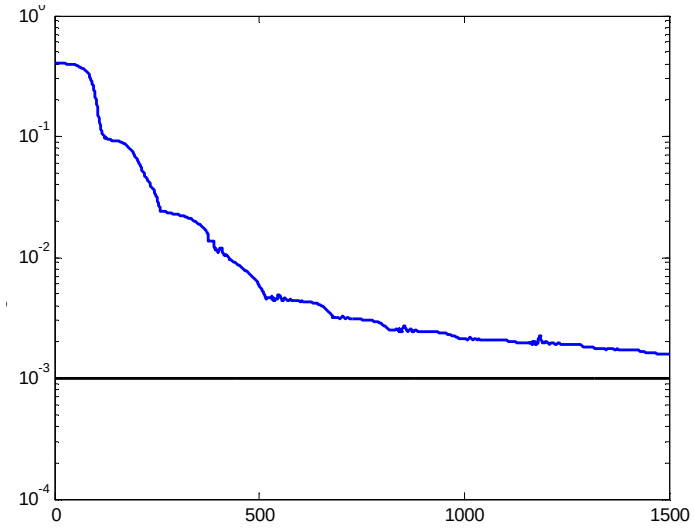


Figure 12.3 : comportement de l'algo du clustering PSO+ MLP, taux de reconnaissance = 79.20%, Nbr d'itération = 1500

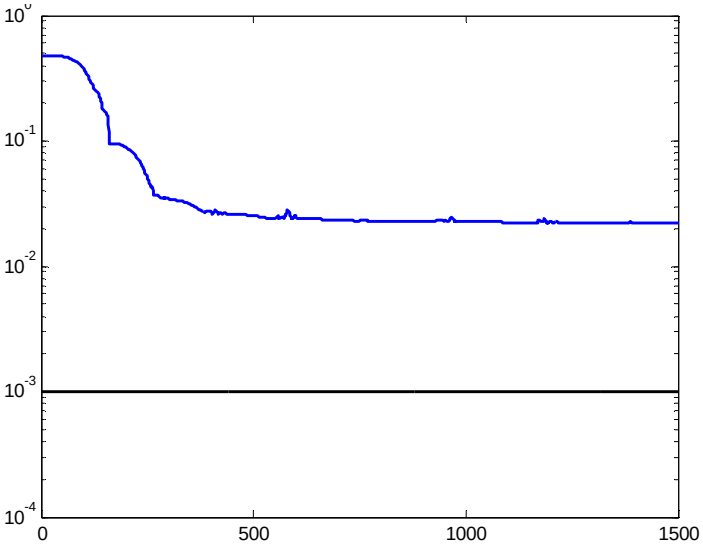


Figure 12.4 : comportement de MLP pur, taux de reconnaissance = 73.45%, Nbr d'itération = 1500

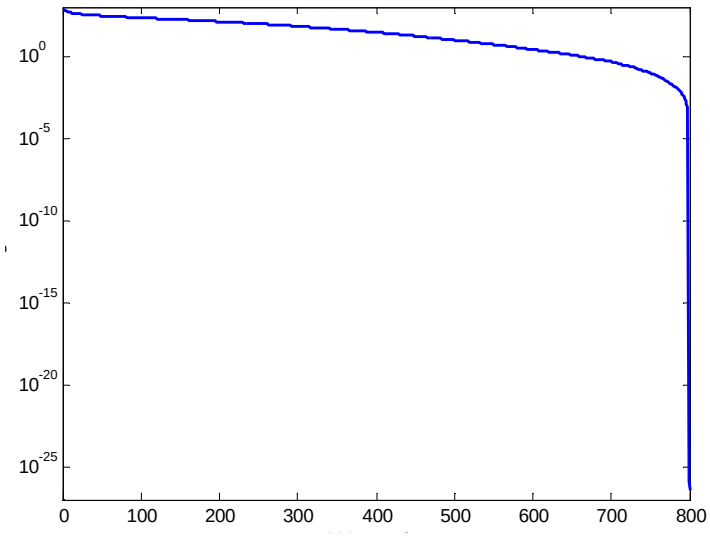


Figure 12.5: comportement de l'algo newrb, taux de reconnaissance = 71. 80% , Nbr d'itartions = 800

Table 12.1: Résultats du clustering par PSO/BA sur les chiffres

Méthode d'apprentissage	Taux de reconnaissance (Zonage)
MLP (traingdx)	80%
MLP - clustering (PSO)	79.20%
MLP - clustering (BA)	79.40%
RBF (newrb)	71.80%
RBF - clustering (PSO)	79.80%
RBF - clustering (BA)	87.70 %

Nous remarquons la supériorité de l'approche clustering basée RBF, d'après ces résultats, on peut dire que la sélection évolutionnaire des caractères d'apprentissage apporte une amélioration du taux de reconnaissance comparativement à Newrb et traingdx.

Résultats de la combinaison série MLP (RBF) + BA

8000 chiffres ont été employés pour l'apprentissage et 2000 pour le test.
260 lettres pour l'apprentissage et 260 pour le test.

Dans ce travail, les attributs de zonage aussi bien que structurels et géométriques ont été employés pour l'apprentissage des réseaux de neurones MLPs et RBFs tandis que le classificateur BA utilise les moments de Hu en tant que attributs caractéristiques car ils se sont sont efficaces dans la classification statistique et permettent aussi de réduire le temps de calcul.

Table 12.2: Résultats de la combinaison série RN +BA

Méthode d'apprentissage		**Taux de reconnaissance**
L'extracteur caractéristique	Zonage	Attributs structurels et géométriques
MLP (traingdx)	80%	69.63%
RBF (newrb)	71.80%	50%
MLPs - BA	99.82%	92.20%
RBFs - BA	72.50%	94.02%
	Les chiffres	**Les lettres**

A partir de ces résultats, nous pouvons dire que la combinaison séquentielle offre les meilleurs résultats dans les deux cas chiffres et lettres ; cela est dû à la correction des erreurs de classification du réseau par le classificateur BA. Aussi, nous observons l'inefficacité des MLP et RBF standard en reconnaissance des lettres.

Notant que, lorsque les chiffres ambigus de la base MNIST sont éliminés comme indiqués dans le premier chapitre de cette partie, un taux de reconnaissance de 100% a été abouti par l'algorithme de la combinaison série MLP - BA, en employant 7500 chiffres pour l'apprentissage et 2000 chiffres pour le test.

Résultats de la combinaison de plusieurs classificateurs

Le taux de reconnaissance du premier classificateur sur les chiffres est 73%
Le taux de reconnaissance du deuxième classificateur est devenu 99.65%
Le taux de reconnaissance du troisième classificateur reste 99.65**%** qui représente le taux final.

Table 12.3: Résultats combinaison de trois réseaux de neurones

Le classificateur employé	**Les chiffres**	**Les lettres**
Combinaison de MLPs modulaires	99.65%	85.95%
Combinaison de RBFs modulaires	95%	92.13 %

12.4 Comparaison avec autres méthodes

Les résultats des chapitres précédents ont démontré un taux d'erreur de 0.18% pour l'approche de la combinaison série MLPs(RBFs) – BA et un taux de 0.35% dans le cas de la combinaison de plusieurs MLPs ayant différents extracteurs caractéristiques. Ces deux classificateurs sont plus performants que le MLP et le RBF standard basé sur les mêmes données d'apprentissage et de test. En outre, nos résultats sont mieux que [Y. LeCun et al, 1992] où un taux d'erreur de 1.1% avec un rejet de 3.4% et ils se sont aussi mieux que [Y. LeCun et al, 1990] où un taux d'erreur de 3.4% a été achevé.

La table suivante montre les résultats obtenus basés sur les règles les plus communes pour la combinaison de classificateurs en parallèle. Notamment, le vote majoritaire, les règles de fusion fixes : min, max, somme, moyenne, le produit, la moyenne de bayes, les modèles de décision et l'espace de connaissance de comportement (BKS). Le détail de ces méthodes existe dans le chapitre 6.
La table présente le taux de reconnaissance de la combinaison de quatre classificateurs. Notamment, la combinaison de quatre MLPs ayant différents attributs caractéristiques, la combinaison de quatre SVMS avec différents attributs ou la combinaison des classificateurs dissimilaires (MLPs et SVMs) avec les attributs mentionnés ci-dessous dans le même ordre.

Table 12.4: combinaison de classificateurs pour la reconnaissance de chiffres

Les attributs structurels et géométriques, des attributs de zonage, la normalisation de caractères et les moments de Hu sont employés dans cet ordre pour chacun des classificateurs. Chaque algorithme a été exécuté 2 fois				
Méthode de combinaison	**Quatre MLPs**	**Quatre SVMs**	**2 MLPs & 2 SVMs**	**Mlp-svm-mlp-svm**
Vote majoritaire	77.75%	76%	74%	74.75%
Règle maximum	41.25%	86%	77.75%	76%
Règle somme	83.25%	86 %	86.75%	83.50%
Règle minimum	69.25%	86.50%	79%	79.75%
Règle moyenne	86.25%	86%	79.75 %	83.50%
Règle produit	77.50%	88.50%	87%	73.25%
Règle de Bayes	75.25%	82.50%	72.50%	79.75%
Les modèles de décision	87. 25%	87%	86.75%	84.25%
BKS	82%	75.50%	75.75%	77.00%

Ces résultats montrent l'efficacité des classificateurs SVMs par rapport aux classificateurs MLPs en termes d'exactitude particulièrement quand la règle produit est employée. En outre, nous avons constaté que les SVMs sont plus rapides en phase d'apprentissage comparés aux MLPs.

En comparant ces résultats avec ceux des chapitres précédents, il est clair que nos structures de combinaison série soient la meilleure particulièrement la combinaison série MLP-BA (99.82%) et la combinaison de plusieurs MLPs basé sur différent extracteurs caractéristiques (99.65%) dû à la correction des erreurs de classification de manière séquentielle.

Conclusion

12.5 Conclusion

Les réseaux de neurones sont conçus de façon à être ajustables pour s'adapter aux variations du style d'écriture, cependant et après une série d'expérimentations, nous avons remarqué que les valeurs des entrées initiales aussi bien que les poids initiaux influent largement le comportement du réseau et ainsi les résultats attendus. Le travail actuel présente une étude comparative de plusieurs algorithmes hybrides où PSO a été employé comme méthode d'initialisation des poids du perceptron multi couche. D'autres techniques sont détaillées telles que : l'utilisation de PSO (BA) comme un algorithme de clustering pour minimiser la taille des exemples d'apprentissage et ainsi réduire le temps de convergence, enfin, nous nous somme concentré à la combinaison de plusieurs classificateurs. La qualité des résultats a été estimée et comparée à d'autres épreuves basées sur différents types d'extracteurs caractéristiques et des méthodes standard tels que le MLP et le RBF conventionnels. La combinaison de classificateurs a permis les meilleurs résultats et plus particulièrement la combinaison série d'un MLP avec un classificateur statistique à base BA. Ceci est dû à la correction des erreurs du MLP par un classificateur efficace basé sur l'algorithme des abeilles. D'autres part, les approches de la combinaison que nous avons développées ont été comparées aux méthodes de combinaison bien connues de la littérature telles que : le vote majoritaire, les règles de la fusion fixes (somme, max, min, médiane, moyenne, produit et Bayes), les modèles de décision et BKS.

DISCUSSIONS ET CONCLUSION GENERALE

Discussions :

Pour que des conclusions fiables soient tirées, cette thèse, s'est appuyée sur une série d'expérimentation et ce, sans perdre de vue, les résultats fournis par les autres méthodes existantes.
Notre premier constat porté sur l'évaluation des performances de nos approches est que l'utilisation d'un ensemble de validation est préférable puisque la fitness calculée sur les mêmes données d'apprentissage n'exprime pas la capacité du pouvoir de généralisation d'un algorithme.

Vu, les nombreuses précautions nécessaires pour l'obtention d'une haute précision. Nous signalons l'importance de l'étape du prétraitement : les images de caractères doivent être normalisées, binarisées, et squelettisées pour extraire éventuellement, les informations pertinentes relatives aux attributs caractéristiques. C'est à l'étape d'extraction des attributs caractéristiques que revient la première responsabilité dans la qualité des résultats (chapitre 3).

Après un nombre énorme de tests et d'expérimentation avec le réseau MLP, nous confirmons les difficultés suivantes : Les réseaux MLPs sont très sensibles à l'initialisation ; pour limiter ce problème, les données d'apprentissage sont généralement centrées et réduites, de même, les sorties désirées peuvent être centrées et réduites. Le nombre d'entrées peut également être réduit pour accélérer l'apprentissage en employant l'analyse de composant principal (PCA). Les poids doivent être initialisés au hasard avec de petites valeurs. Le taux d'apprentissage influe de manière significative la performance du réseau, si la valeur du taux est petite l'algorithme converge graduellement, si elle est par contre une grande valeur, l'algorithme est susceptible de diverger. Généralement, la valeur du taux doit être comprise dans l'intervalle [0, 1] pour assurer la convergence vers une solution optimale [A. Abraham, 2004], [J. Aschenwald et al, 2004]. En outre, il n'y a aucune règle qui détermine de manière significative le nombre de couches cachées ou le nombre de neurones de chaque couche. Une grande variété de méthodes pour traiter ces problèmes ont été proposés ; parmi lesquels : l'addition dynamique de neurones ; un neurone est ajouté à chaque fois que l'erreur est stagnée sur une valeur indésirable. Le momentum est une autre alternative efficace qui accélère l'apprentissage et échappe également des minimums locaux [A. Abraham, 2004].

La mesure de l'impact des divers algorithmes, suite à une série d'expérimentations au nombre de cinq, a été effectuée et mise en relief par des graphiques et des tables, représentant ainsi le taux de reconnaissance moyen. Les graphiques confirment parfaitement bien la convergence qualitative de chaque algorithme.

Le classificateur statistique à base des abeilles est vu comme étant le plus lent d'entre eux, à trouver la bonne solution. Ceci est dû à la technique de recherche par voisinage spatial qu'il suit.
Le phénomène de la généralisation est central dans les interprétations associées aux réseaux de neurones. La généralisation se traduit par la capacité de reconnaître des instances imprévues.
En effet, de nombreuses expériences ont montré l'apparition du phénomène de surapprentissage dans le cas d'un apprentissage du MLP par PSO. Autrement dit, les résultats tendent à se dégrader pendant le processus d'apprentissage. C'est ce qu'on appelle le manque de la généralisation, l'origine de cet effet est l'augmentation énorme des valeurs de poids indépendamment du gradient.

Les méthodes classiques à base des réseaux de neurones, le MLP et le RBF, ont donné des performances sous optimales, montrant leur inefficacité à classifier les caractères manuscrits. Dans nos expérimentations sur les approches neuronales, nous avons rencontré deux difficultés majeures, l'utilisation d'un algorithme de recherche locale tel que la rétropropagation de gradient qui tombe souvent dans un minimum local et l'utilisation d'un algorithme de recherche globale tel que PSO, qui mène souvent à des résultats néfastes, liés à la perte de la généralisation. Pour résoudre ce problème, nous avons considéré l'approche Bp_pso combinant une recherche locale avec une recherche globale, ce qui a généré de meilleurs résultats.

Bien que le MLP standard surpasse le RBF standard, ces deux classificateurs du point de vue co-évolutionnaire donnent des résultats de classification, soit des taux de reconnaissance assez proches : dus à la stratégie localisée que suivent les RBFs. En outre, les positions des centres d'un RBF influent de manière significative son comportement.

Nous avons remarqué que les approches neuronales basées sur la combinaison de classificateurs exhibent une grande puissance de généralisation. Ceci est dû, à la coopération de plusieurs réseaux de neurones dont les entrées sont issues de différents extracteurs caractéristiques. Les techniques de combinaison de classificateurs ont permis d'obtenir des résultats intéressants, avec l'avantage, de ne pas être très sensibles à l'initialisation : les erreurs du premier classificateur sont corrigées par le classificateur suivant et ainsi de suite. Ces méthodes arrivent à reconstituer avec haute précision la solution recherchée.

Il ressort de cette étude, qu'il existe des caractères ambigus dans la base MNIST. Nous avons montré que lorsque les caractères ambigus sont enlevés, nous pouvons atteindre un taux de reconnaissance de 100%, plus particulièrement avec les méthodes de combinaison de classificateurs.

Conclusions :

Cette thèse est consacrée aux améliorations apportées par des méthodes d'intelligence collective au perceptron multicouche et le réseau de neurones de fonctions à base radiale afin de résoudre le problème de la reconnaissance de caractères manuscrits de manière efficace.

Les approches développées ont été étudiées, dans le but, d'améliorer les performances du système de reconnaissance de caractères en termes de vitesse et d'exactitude. Les résultats obtenus sont prometteurs, suite, à la mise en œuvre de différents classificateurs proposés. Nous avons examiné les résultats trouvés selon différents types de caractères manuscrits et différents types d'extracteurs pour montrer l'efficacité des algorithmes proposés. Les performances des algorithmes présentés sont aussi comparées et évaluées, selon une méthodologie comparative sur le plan des performances, suivie pour chacun des modèles proposés.

La 3ème partie de cette thèse dédiée à nos contributions est constituée de trois chapitres :
Le premier chapitre expose le potentiel des méthodes statistiques de la reconnaissance de caractères manuscrits basés sur les algorithmes d'intelligence en essaim PSO, BA et ABC selon différents types d'extracteurs caractéristiques ; notamment les moments de Hu, le zonage et les attributs structurels et géométriques. Nous avons décrit comment des méthodes d'intelligence en essaim (PSO, BA et ABC), peuvent être employées pour résoudre le problème de la reconnaissance de caractères manuscrits tout en examinant leurs performances. Ce chapitre montre la supériorité du classificateur à base BA comparativement à un classificateur bien connu du K-plus proches voisins (K-nn). Bien que ces méthodes statistiques, soient une réussite en termes d'exactitude, leur inconvénient majeur réside dans le fait qu'elles prennent un temps de réponse très large, puisqu'elles procèdent à une opération de recherche globale. A cet effet, les réseaux de neurones ont été proposés comme solution dans le 2ème et 3ème chapitre.

A partir de nos expérimentations sur les réseaux de neurones évolutionnaires, nous avons trouvé que l'apprentissage évolutionnaire des poids du MLP ou RBF engendre souvent le phénomène du surapprentissage dû à la perte de généralisation à cause du changement des poids indépendamment du gradient. Pour résoudre ce problème, nous nous sommes intéressés au *deuxième chapitre* à l'application d'une approche coopérative co-évolutionnaire sur deux types de réseaux de neurones (le MLP et le RBF).Cette approche permet l'évolution simultanée des architectures des RNs avec leurs ensembles d'apprentissage pour estimer les meilleurs poids d'un MLP et meilleurs centres et poids d'un RBF dans le cadre de la reconnaissance de chiffres manuscrits. La coévolution a apporté des améliorations des performances en termes du taux de reconnaissance relativement aux MLP et RBF de base. Particulièrement, l'algorithme co-évolutionnaire à base RBF fournit les meilleures performances, face à l'algorithme de base du RBF. La valeur du taux de la reconnaissance d'une approche co-évolutionnaire est nettement supérieure à celle issue d'une approche évolutionnaire.

Le dernier chapitre est consacré à des hybridations des réseaux de neurones avec les algorithmes d'intelligence en essaim selon différentes sources d'inspiration. Il commence par le clustering de la base d'apprentissage pour améliorer la qualité et le temps de calcul du MLP et RBF. Les résultats obtenus selon différents extracteurs caractéristiques aussi bien que les résultats issus d'autres méthodes existantes ont été présentés afin de mieux cerner la situation de nos travaux par rapport aux autres méthodes. La sélection des attributs significatifs par clustering à travers la méthode PSO a permis de réduire la taille de la base d'apprentissage et ainsi, une amélioration en termes de vitesse et d'exactitude.

Ensuite nous avons présenté la combinaison série du MLP (RBF) et un classificateur statistique inspiré du comportement collectif des abeilles suivie d'une combinaison de trois réseaux modulaires MLPs (RBFs) ayant des extracteurs d'attributs différents aussi bien qu'une comparaison des résultats obtenus à ceux des algorithmes célèbres. Le principe de la combinaison série que nous avons mis en œuvre, consiste à employer une architecture modulaire du MLP (RBF) où chaque MLP (RBF) a été spécifié pour distinguer une classe des autres classes possibles afin de pouvoir déterminer les caractères inconnus par chaque MLP (RBF). Les caractères inconnus sont ensuite envoyés à un classificateur statistique basé sur l'algorithme des abeilles pour augmenter le taux de reconnaissance. Cette combinaison série de classificateurs modulaires suivis d'une classification à base de l'algorithme des abeilles, exhibe une grande capacité d'adaptation et fournit les meilleures performances en terme du taux de reconnaissance sur les chiffres et les lettres manuscrits. En outre, elle a montré sa supériorité par un taux de reconnaissance élevé de 99.82 % sur les chiffres.

La combinaison de classificateurs de même type MLP (RBF), avec différents extracteurs d'attributs caractéristiques a montré la plus grande capacité de généralisation. Elle a permis d'obtenir les meilleures performances en termes du temps de réponse. Néanmoins, cette méthode est relativement longue durant la phase d'apprentissage, elle dure plus de huit heures sur un ensemble de 8000 caractères (800 caractères pour chaque chiffre).

La reconnaissance de caractères à base des réseaux de neurones, offre une solution plus avancée par rapport aux méthodes statistiques que nous avons exposées. Ils offrent des améliorations en termes du temps de réponse, voire, de qualité de classification quand elles sont acquises à partir des méthodes d'apprentissages efficaces. Néanmoins, ils prennent un temps d'apprentissage considérable mais une fois l'apprentissage terminé, les meilleurs poids trouvés seront enregistrés pour un usage rapide.

Les algorithmes proposés sont vérifiés sur les chiffres arabes manuscrits, et sur les lettres de la langue anglaise. A travers l'étude présentée, qui a mis en valeur, les capacités intéressantes des approches proposées, nous concluons, que le travail présenté offre une solution très satisfaisante au problème de la reconnaissance des chiffres arabes manuscrits.

Perspectives

Les travaux accomplis, dans cette thèse, ouvrent plusieurs perspectives de futurs travaux suivant trois voies d'amélioration de nos approches :

- Utilisation des extracteurs caractéristiques et des méthodes de combinaison de classificateurs vus dans les chapitres précédents mais qui ne sont pas exploités dans ce document actuel. L'investigation des algorithmes plus efficaces de la combinaison de classificateurs peut produire des résultats plus performants.
- Le second point focalise sur l'aspect des méthodes de classification et d'apprentissage.
- Certains aspects pourraient être étendus ; il est possible de travailler avec les approches développées sur d'autres applications telles que la reconnaissance de mots et de phrases.

Concernant l'extraction des attributs caractéristiques, il y a deux aspects à approfondir. Tout d'abord il faudrait étudier l'influence des extracteurs plus strictes inexploités dans ce document sur la qualité de la reconnaissance et de savoir comment les fusionner ou les combiner afin d'augmenter la qualité de la reconnaissance. C'est l'idée qui nous a orientés vers les méthodes de la combinaison de classificateurs vus dans le chapitre 5 de la première partie.

Le deuxième aspect à approfondir concerne les méthodes de classification. Les approches développées sont basées sur deux types de réseaux de neurones qui sont les MLP et les RBF, mais il nous semble après lecture de la littérature qu'il serait intéressant d'utiliser d'autres types comme les réseaux de neurones convolutionnels. Les SDNNs et les réseaux de neurones récurrents. Aussi, Nous avons suivi la démarche de modélisation avec des algorithmes existants tels que PSO et les algorithmes d'abeille, mais il est nécessaire d'augmenter leurs performances. Ceci figure parmi les extensions futures de nos applications.

Actuellement, nous avons réalisé un système de la reconnaissance de chiffres arabes manuscrits. Or, comme les applications réelles sont basées sur la reconnaissance des chaines de caractères, nous pensons de choisir un domaine d'application réel tel que la reconnaissance de la somme numérique des chèques bancaires. Dans un deuxième temps, nous allons travailler sur l'automatisation de la reconnaissance du montant littéral des chèques bancaires en se basant sur les approches élaborées. Pour cela nous proposons de développer des techniques de segmentation automatiques des chiffres connectés. Les réseaux de neurones récurrents mériteraient d'être explorés.

ANNEXE

A. Résultats comparatifs :

Nous présentons ci-dessous un tableau récapitulatif des résultats aboutis de chaque algorithme (chaque algorithme a été testé au moins 3 fois).

les classificateurs statistiques (taux de reconnaissance)			
La méthode de classification	Zonage	Moment de Hu	Structurels et géométriques
PSO (100 itérations)	80.428 %	92.2%	81.20%
ABC (100 itérations)	80 %	84.49%	70%
BA (100 itérations)	92.857 %	99.8%	82.48%
K-nn	90%	98%	82.48%
		Les chiffres	**Les lettres d'anglais**
l'approche symbiotique co-évolutionnaire (les chiffres avec zonage)			
La Méthode d'apprentissage			**Taux de reconnaissance**
Le RBF pur (newrb de matlab)			71.80 %
L'approche symbiotique coopérative			88.79 %
Le MLP (Traingdx)			80 %
L'approche co-évolutionnaire à base MLP			86%
Le clustering de caractères (les chiffres avec Zonage)			
Méthode d'apprentissage			**Taux de reconnaissance**
MLP (traingdx)			80%
MLP - clustering (PSO)			79.20%
MLP - clustering (BA)			79.40%
RBF (newrb)			71.80%
RBF - clustering (PSO)			79.80%
RBF - clustering (BA)			87.70 %
La combinaison série RN + BA			
Méthode d'apprentissage			**Taux de reconnaissance**
L'extracteur caractéristique		Zonage	Attributs structurels et géométriques
MLP (traingdx)		80%	69.63%
RBF (newrb)		71.80%	50%
MLPs - BA		99.82%	92.20%
RBFs - BA		72.50%	94.02%
		Les chiffres	**Les lettres**
La combinaison de trois réseaux modulaires			
Le classificateur employé		**Les chiffres**	**Les lettres**
Combinaison de MLPs modulaires		99.65%	85.95%
Combinaison de RBFs modulaires		95%	92.13 %
MLP (traingdx)		80%	69.63%

RBF (newrb)	71.80%	50%

La combinaison parallèle de 4 classificateurs de meme ou de différents types pour la reconnaissance des chiffres manuscrits.				
Les attributs structurels et géométriques, les attributs de zonage, la normalisation de caractères et les moments de Hu sont employés dans cet ordre pour chacun des classificateurs.				
Méthode de combinaison	**Quatre MLPs**	**Quatre SVMs**	**2 MLPs & 2 SVMs**	**Mlp-svm-mlp-svm**
Vote majoritaire	77.75%	76%	74%	74.75%
Règle maximum	41.25%	86%	77.75%	76%
Règle somme	83.25%	86 %	86.75%	83.50%
Règle minimum	69.25%	86.50%	79%	79.75%
Règle moyenne	86.25%	86%	79.75 %	83.50%
Règle produit	77.50%	88.50%	87%	73.25%
Règle de Bayes	75.25%	82.50%	72.50%	79.75%
Les modèles de décision	87. 25%	87%	86.75%	84.25%
BKS	82%	75.50%	75.75%	77.00%

En résumé :

- Le classificateur statistique BA donne les meilleurs résultats par rapport à PSO, ABC et K-nn.

- L'approche coopérative coévolutionnaire à base RBF donne les meilleurs résultats une fois comparés à newrb, Traingdx et l'approche co-évolutionnaire à base MLP. Ceci est dû à la stratégie localisée que suivent les RBFs.

- On remarque la supériorité de l'approche clustering de caractères basée RBF et plus particulièrement quand l'algorithme des abeilles est employé pour le clustering et ceci comparativement à Newrb et traingdx.

- la combinaison série MLP et BA offre les meilleurs résultats dans le cas des chiffres ; cela est dû à la correction des erreurs de classification du réseau par le classificateur BA. Aussi, le MLP est plus efficace que le RBF dans le cas d'un grand nombre de caractères.

- La combinaison série des réseaux MLPs modulaires a permis les meilleurs résultats. Ceci démontre la capacité de généralisation d'un MLP modulaire par rapport à un RBF modulaire.

- Nous avons employé les attributs zonages dans la plupart de nos expérimentations basées RNs puisqu'ils donnent des résultats proches à ceux des attributs structurels et géométriques qui demandent un temps de calcul presque 3 fois plus. Alors que les moments de Hu sont les plus appropriés aux classificateurs statistiques.

- Les SVMs sont plus efficaces par rapport aux MLPs en termes d'exactitude quand la règle produit est employée.

- Les SVMs sont plus rapides en phase d'apprentissage comparés aux MLPs.

- Nos structures de combinaison série sont les meilleures particulièrement la combinaison série MLP-BA et (99.82%) et la combinaison de plusieurs MLPs basé sur différent extracteurs caractéristiques (99.65%) dû à la correction des erreurs de classification de manière séquentielle.

B. Analyse des résultats :

Autres expérimentations ont été effectuées sur des bases de données UCI [A.Frank et al, 2010]

taille	Nom de la base	Type des attributs	N° d'exemples	N° d'attributs	N° de classes
petite	Iris	Réel	150	4	2
	Heart	Catégorique, entier, réel	270	13	2
	Sonar	Real	208	60	2
	Dermatology	Catégorique, entier	358	34	6
moyenne	Vowel	Réel	528	10	11
	Vehicle	Catégorique, entier	846	18	3
	Ionosphere	Entier, réel	351	34	2
	BreastCancer	Catégorique, entier	683	9	2
Large	ImageSegmentation	Réel	2310	19	7
	Yeast	Réel	1484	8	10
	Statlog (satellite images)	entier	6435	36	7

La table suivante montre le taux de reconnaissance des classificateurs statistiques à base PSO et BA par rapport à la classification (SVM). Des bases de données de petite, moyenne et grande taille ont été utilisées. SVM est l'un des meilleurs classificateurs en termes de vitesse et de précision.

Le taux de reconnaissance des classificateurs SVM, PSO et BA pour la reconnaissance de formes			
	SVM	**BA**	**PSO**
Iris	91.11	94.07	96.30
Heart	71.60	55.56	58.02
Sonar	69.84	78.57	66.67
dermatology	91.66	83.80	67.90
BreastCancer	95.12	95.12	94.96
Ionosphere	91.50	85.58	84.44
vehicle	63.77	69.29	59.27
vowel	88.05	96.86	72.02
Segmentation	59.59	93.51	83.40
satimage	23.71	90.01	83.48
yeast	54.93	49.89	46.64

Les résultats montrent que les classificateurs PSO et SVM fournissent de meilleures performances sur les petites bases de données. Le classificateur BA surpasse PSO dans les ensembles de données de taille grande ou moyenne tandis que SVM montre son inefficacité pour la reconnaissance des grands ensembles de données.

En outre, d'autres expérimentations ont été menées basées sur la combinaison parallèle de classificateurs à savoir: (MLP et SVM), (MLP & BA) et (MLP & SVM & BA) en parallèle où la décision finale est calculée par vote majoritaire, les résultats correspondants sont présentés dans le tableau suivant.

Le taux de reconnaissance des méthodes de combinaison parallèle pour la classification des données			
Classificateur / **Base de données**	combinaison Parallèle (MLP&SVM)	combinaison Parallèle (MLP & BA)	combinaison Parallèle (MLP& SVM&BA)
Iris	93.33	94.44	93 .33
Heart	64.82	70.99	75.93
Sonar	61.11	68.25	77.78
dermatology	85.18	84.72	92.59
BreastCancer	96.10	96.10	96.10
Ionosphere	88.68	84.91	90.57
vehicle	56.30	73.32	75.59
vowel	60.38	85.32	79.25
Segmentation	72.87	92.85	89.98
satimage	86.19	86.28	86.75
yeast	54.93	51.46	56.05

A partir de ces résultats, la combinaison parallèle de MLP, SVM et BA fournie les meilleurs résultats dans tous les types des bases de données (petite, moyenne et grande). La combinaison parallèle de MLP et BA surpasse la combinaison parallèle de MLP et SVM. La combinaison de MLP et SVM montre son inefficacité dans la classification des ensembles de données de grande taille.

Bien que l'approche de la combinaison est motivée pour produire les meilleurs résultats, le classificateur BA obtient des résultats similaires à ceux de la combinaison série MLP & BA en moyenne taille de données aussi le classificateur BA surpasse l'approche de la combinaison dans le cas des grande taille de données. Ceci prouve le potentiel des classificateurs basés sur l'intelligence en essaim dans la reconnaissance des formes.

En outre, il est possible de faire la transition du classificateur d co-évolutionnaire et la combinaison série : MLP & BA de la reconnaissance des caractères à d'autres applications. Cependant, le facteur le plus important qui influe sur la qualité des résultats, est essentiellement les attributs caractéristiques employés. Dans le cas des bases de données UCI mentionnées ci-dessus, nous n'avons pas obtenu de bons résultats en particulier sur les petits ensembles de données.

Bibliographie :

[A. Abraham et al, 2000]	A. Abraham, B. Nath, "Optimal Design of Neural Nets Using Hybrid Algorithms", In proceedings of 6th Pacific Rim International Conference on Artificial Intelligence, Springer Verlag, Germany, pp. 510-520, 2000.
[A. Abraham et al, 2001]	A. Abraham. D. Steinberg, "Is Neural Network a Reliable Forecaster on Earth? A MARS Query!", *Lecture Notes in Computer Science,* 2085, 679–686 .2001.
[A. Abraham, 2004]	A. Abraham,"Meta-Learning Evolutionary Artificial Neural Networks", *Neurocomputing Journal, Elsevier Science*, Netherlands, Vol. 56c, pp. 1-38 .2004.
[A. Ajith, 2002]	A. Ajith, "Optimization of Evolutionary Neural Networks Using Hybrid Learning Algorithms", IEEE 2002 Joint international conference on neural networks, IEEE Press, pp: 2797-2802. 2002.
[A. Berlanga et al, 1999]	A. Berlanga, A.Sanchis, P. Isasi, J. M. Molina, "Neural networks robot controller trained with evolution strategies", in: Proc. of 1999 Congress on Evolutionary Computation, CEC99, 1999.
[A. Borji et al, 2007]	A.Borji, et M. Hamidi. Support Vector Machines for Persian Font recognition. In: International Conference on Computer, Electrical, Systems Science, and Engineering (CESSE 2007), Prague, Czech Republic July 27-29, 2007.
[A. Brakensiek et al, 2003]	A. Brakensiek, J. Rottland, G. Rigoll,"Confidence measures for an address reading system". Seventh international conference on document analysis and recognition, ICDAR2003, pp 294–298. 2003.
[A. Churbanov et al, 2008]	A. Churbanov, S. Winters-Hilt, Implementing EM and Viterbi algorithms for Hidden Markov Model in linear memory, BMC Bioinformatics.;9:224. 2008.
[A. D. Brown et al, 1997]	A. D. Brown and H. C. Card, "Evolutionary artificial neural networks," in *Proc. 1997 Canadian Conf. Electrical and Computer Engineering, CCECE'97. Part 1 (of 2)*, pp. 313–317. *1997*
[A. D.S.B. Jr et al, 2001]	A.D.S.B. Jr, R.Sabourin, , F. Bortolozzi, C.Y. Suen, "A Two-Stage HMM-Based System for Recognizing Handwritten Numeral Strings". ICDAR 2001: 396-400. 2001.

[A. Drogoul, 1993]	A.Drogoul, "De la simulation multi-agents à la résolution collective de problèmes", PhD Thesis, University Paris 6, November 1993.
[A.Frank et al, 2010]	A.Frank, A. Asuncion, "UCI Machine Learning Repository" ,http://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science. (2010).
[A. F. Silva et al, 2003]	A.F. Silva, A.P. Silva, E.Costa, "SAPPO: A Simple, Adaptive, Predator Prey Optimiser", in Proc. of the 11th Portuguese Conference on Artificial Intelligence, Workshop on Artificial Life and Evolutionary Algorithms (ALEA), EPIA'03,Beja, Portugal, December-2003
[A. H. Bond et al, 1988]	A.H. Bond, L. Gasser, "Readings in Distributed Artificial Intelligence", San Mateo, CA: Morgan Kaufmann, 1988.
[A. Jain et al, 1997]	A. Jain, D. Zongker. "Feature Selection: Evaluation, Application and Small Sample Performance". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):153–197, 1997.
[A. Kawtrakul et al, 2004]	A. Kawtrakul, W. Peerawit, W. Yingsaeree, "The utilization of closing Algorithm and Heuristic Information for Broken Character Segmentation", 2004 IEEE Conference on Cybernetics and Intelligen Systems & Robotics, Automation and Mechatronics , December 1-3. 2004.
[A. Khotanzad et al, 1990]	A.Khotanzad, Y.H.Hong, "Invariant image recognition by Zernike moments", IEEE Trans, Pattern Analysis and Machine Intelligence. Vol. 12, no. 5, pp. 489-497, 1990.
[A. L. Arnaud et al, 2005]	A. L. Arnaud, P. J. L. Adeodato, G. C. Vasconcelos, R. F. O. Neto, R. Ferreira," MLP Neural Networks Optimization through Simulated Annealing in a Hybrid Approach for Time Series Prediction". *In: SBC ENIA 2005 - V Encontro Nacional de Inteligência Artificial,* pp. 1110-1113. 2005.
[A. L. Koerich et al, 2003]	A.L. Koerich, "Unconstrained Handwritten Character Recognition Using Different Classification Strategies", *International Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR)*, 2003.
[A. L. Koerich, 2003]	A.L. Koerich, "Unconstrained Handwritten Character Recognition Using Different Classification Strategies", *International Workshop on Artificial Neural Networks inPattern Recognition (ANNPR)*, 2003.

[A. Malisia et al, 2006]	A. Malisia, H.R.Tizhoosh, "Image Thresholding Using Ant Colony Optimization", Proceedings of the 13the IEEE International Conference on Image Processing (ICIP-2006), Atlanta, GA, USA, pp. 2409-2412. 2006.
[A. Pinkus, 1999]	A. Pinkus, "Approximation theory of the MLP model in neural networks", volume 8, pages: 143-195. 1999.
[A. Shubair, 2007]	A. Shubair, *"*Off-Line Handwritten Arabic Characters Segmentation Using Slant-Tolerant Segment Features (STSF)*",* Masters thesis, Universiti Sains Malaysia. 2007.
[A. Sierra et al, 2001]	A. Sierra, J. A. Macías, F. Corbacho, "Evolution of functional link networks". IEEE Transactions on Evolutionary Computation. v5 i1. Pp: 54-65.2001.
[A. T. Nghiem et al, 2005]	A.T. Nghiem, A.Boucher, "Reconnaissance d'écriture manuscrite", Techenical report, 2005.
[A. Vesely, 2003]	A. Vesely, "Neural networks in data mining". *AGRIC. ECON. – CZECH*, *49*, (9): 427–431. 2003.
[A. Vinciarelli, 2002]	A.Vinciarelli: "A survey on off-line Cursive Word Recognition". Pattern Recognition 35(7): 1433-1446 .2002.
[A. Zramdini et al, 1993]	A.Zramdini, R.Ingold, "Optical Font Recognition from Projection Profils". Electronic Publishing 6(3): 249-260 .1993.
[Asif Iqbal et al, 2008]	Asif Iqbal, A. B. M. Musa, Anindya Tahsin, Md. Abdus Sattar, Md. Monirul Islam , and K. Murase, "A Novel Algorithm for Translation, Rotation and Scale Invariant Character Recognition," *Proceedings of Joint 4th International Conference on Soft Computing and Intelligent Systems and 9th International Symposium on Advanced Intelligent Systems, (SCIS&ISIS2008) Nagoya, Japan*, Sept 17-21, pp. 1367-1374. 2008
[B. A. Whitcheas et al, 1994]	B.A. Whitcheas, T.D. Choate, "Evolving Space-Filling Curves to Distribute Radial Basis Functions Over an Input Space". IEEE Transactions on Neural Networks, 1: pp: 15-23, 1994.
[B. Deo et al, 1994]	B. Deo, A. Datta, B. Kukreja, R. Rastogi, and K. Deb, "Optimization of back propagation algorithm and GAS-assisted ANN models for hot metal desulphurization," *Steel Res.*, vol. 65, no. 12, pp. 528–533, 1994.
[B. Gatos et al, 2006]	B. Gatos, I. Pratikakis, A. Kesidis and S. Perantonis (2006), Efficient Off-Line Cursive Handwriting Word Recognition, International Workshop on Frontiers in Handwriting Recognition (IWFHR'06), ISBN: 2-9527630-0-3/978-2-9527630-0-4, Baule, France, Oct. 23-26, pp.121-126. 2006.

[B. Gosselin, 1996]	B. Gosselin, "Applications de réseaux de neurones artificiels à la reconnaissance automatique de caractères manuscrits", Ph.D. thesis, Faculté Polytechnique de Mons, Belgium .1996.
[B. H. Juang et al, 1991]	B. H. Juang, L. R. Rabiner, "Hidden Markov Models for Speech Recognition", Technometrics, Vol. 33, No. 3. pp. 251-272. 1991.
[B. VijayKumar et al, 2004]	B. VijayKumar, A. G. Ramakrishnan, Radial Basis Function and Subspace Approach for Printed Kannada Text Recognition, International Conference on Acoustics, Speech, and Signal Processing,. Proceedings. (ICASSP '04). On page(s) 321-4 vol.5. 2004.
[B. V. Kumar et al, 2004]	B.V. Kumar, A. G. Ramakrishnan, "Radial Basis Function and subspace approach for Printed Kannada Text Recognition", ICASSP-2004, pp. 321-324. 2004.
[B. V.Dasarathy, 1991]	B. V. Dasarathy, "Nearest neighbor pattern classification techniques", IEEE Computer Society Press, New York 1991.
[C. Burges, 1998]	C. Burges,"A Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and. Knowledge Discovery, vol. 2, no. 2, pp. 1-47, 1998.
[C. Enăchescu et al, 2010]	C.Enăchescu, C.D.Miron, "Handwritten Digits Recognition Using Neural Computing", Studia Universitatis Babes-Bolyai : Series Informatica ; Vol.LV No.1 .Pages: 95-104. 2010.
[C. Fang, 1997]	C. Fang, "Deciphering Algorithms for Degraded Document Recognition", Ph.D thesis. 1997.
[C.Gagné et al, 2006]	C.Gagné , M.Parizeau, Genetic Engineering of Hierarchical Fuzzy Regional Representations for Handwritten Character Recognition, International Journal on Document Analysis and Recognition, Volume 8 Issue 4, pp. 223-231, August 2006.
[C. H. Gros, 2003]	C.H. Gros, "Genetic evolution of neural networks", Genetic Algorithms and Genetic Programming at Stanford 2003, "Stanford Bookstore", pp: 68-74, 2003.
[C.L.Liu et al, 2008]	C.L.Liu, H.Fujisawa, "Classification and Learning Methods for Character Recognition: Comparison of Methods and Remaining Problems", In Machine Learning in Document Analysis and Recognition, Vol. 90, pp. 139-161.2008.
[C. Meurie et al, 2003]	C. Meurie, O. Lezoray, A. Elmoataz, "Segmentation d'images couleur par fusion de classification de pixels. Application en imagerie biomédicale", 16ème journées de l'Association Française d'Informatique Graphique, pp. 104-110, Paris, 2003.

[C. S. Wallace et al, 1991]	C. S. Wallace, J. D. Patrick, "Coding decision trees," Dep. Comput. Sci., Monash University, Clayton, Victoria, Australia, Tech. Rep. 91/153, Aug. 1991.
[C. Scagliola, 2000]	C. Scagliola, G. Nicchiotti, F. Camastra, "Enhancing Cursive Word Recognition Performance by the Integration of all the Available Information". *7th IWFHR*, Amsterdam, September 11-13, 2000.
[C.Sureshkumar et al,2010]	C.Sureshkumar, T.Ravichandran, Character Recognition using RCS with Neural Network, International Journal of Computer Science Issues, Vol: 7 Issue: 5 Pages: 289-295.2010.
[C. W. Reynolds, 1987]	C.W. Reynolds, "Flocks, herds and schools: A distributed behavioral model.", SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques (Association for Computing Machinery): 25–34, 1987.
[C. Wu et al, 2003]	C. Wu, Y. Liang, "Center Selection for RBF Neural Network in Prediction of Nonlinear Time Series". Proceedings of the Second International Conference on Machine Learning and Cybernetics, pp: 1355-1359, 2003.
[D. B. Fogel, 1995]	D. B. Fogel, "Phenotypes, genotypes, and operators in evolutionary computation," in *Proc. 1995 IEEE Int. Conf. Evolutionary Computation (ICEC'95)*, Perth, Australia, pp. 193–198. *1995*.
[D. Bouchain, 2007]	D. Bouchain, "Character Recognition Using Convolutional Neural Networks", International Joint Conference on Neural Networks. Volume: 16802, Issue: 1, Publisher: Ieee, Pages: 53-58. 2007.
[D.C.Cireset et al, 2011]	D.C.Cires¸ U.Meier, L.M.Gambardella, J.Schmidhuber, Convolutional Neural Network Committees For Handwritten Character Classification, ICDAR IEEE, p. 1135-1139. 2011.
[D. Dileep Gaurav, 2009]	D. Dileep Gaurav "FEATURE EXTRACTION FOR CHARACTER RECOGNITION", http://www.mathworks.com/matlabcentral/fileexchange/24624-feature-extraction-for-character-recognition
[D. Dumitrescu et al, 2005]	D. Dumitrescu, K. Simon, "A new dynamic evolutionary techniques. Applicaion in designing RBF neural network topologies". II. Numerical Experiments, Studia Univ. Babes-Bolyai, Ser. Informatica, 50, pp: 59-69. 2005.
[D. E. Goldberg, 1989]	D.E. Goldberg, "Genetic Algorithms in Search Optimization and Machine Learning". Addison Wesley. pp. 41. 1989.

[D. E. Moriarty et al, 1994]	D. E. Moriarty, R. Miikkulainen, "Evolutionary Neural Networks for Value Ordering in Constraint Satisfaction Problems", Technical Report AI94-218, Department of Computer Sciences, The University of Texas at Austin, 1994.
[D. E.Rumelhart et al, 1986]	D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. I. Cambridge, MA: Bradford Books, pp. 318–362, 1986.
[D. F. Specht, 1990]	D. F. Specht, "Probabilistic Neural Networks," Neural Networks, Vol. 3, pp: 109-18. 1990.
[D. F.A.Santiago et al, 2002]	D.F.A.Santiago, R. Pederiva, "Comparison of optimization techniques of neural networks training for faults diagnostic of rotating machinery". Proceeding of the 1st South-American Congress on Computational Mechanics, MECOM, pp: 1912-1921. 2002.
[D. J. Chalmers, 1990]	D. J. Chalmers, "The evolution of learning: An experiment in genetic connectionism," in Proc. Connectionist Models Summer School, pp. 81–90. 1990.
[D. Karaboga et al, 2007a]	D.Karaboga., B.Basturk., "Artificial Bee Colony Algorithm on Training Artificial Neural Networks", Signal Processing and Communications Applications, SIU 2007, IEEE 15th. 11-13, Page(s):1 - 4, doi: 10.1109/SIU.2007.4298679 .2007.
[D. Karaboga et al, 2007b]	D.Karaboga, B.Basturk, C.Ozturk, "Artificial Bee Colony (ABC) Optimization Algorithm for Training Feed-Forward Neural Networks", LNCS: Modeling Decisions for Artificial Intelligence, Vol: 4617/2007, pp:318-319, Springer-Verlag, 2007, MDAI 2007. doi: 10.1007/978-3-540-73729-2_30 .2007.
[D. Karaboga, 2005]	D.Karaboga, "An Idea Based On Honey Bee Swarm for Numerical Optimization", Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department. 2005.
[D. Karaboga, 2010]	D. Karaboga, "Artificial bee colony algorithm". Scholarpedia, Vol 5; No 3. Page 6915. 2010.
[D. Montana et al, 1989]	D. Montana, L. Davis, "Training feedforward neural networks using genetic algorithms", in: Proceedings of Eleventh International Joint Conference on Artifical Intelligence, N.S. Sridharan (Ed.), Morgan Kaufman Publishers, 1989.

[D. Moreno et al, 2006]	D. Moreno, P. Marco, I. Olmeda, "Self-organizing maps could improve the classification of Spanish mutual funds", European Journal of Operational Research. Volume 174, Issue 2, pp: 1039-1054, 16 October 2006.
[D.Singh et al, 2011]	D.Singh, N.Mehta, P.Purohit, "Text Based Image Recognition using Multilayer Perceptron", Special issues on IP Multimedia Communications, pp: 143-146, 2011.
[D. Weissenbacher et al, 2007]	D.Weissenbacher, A.Nazarenko. "A bayesian classifier for the recognition of the impersonal occurrences of the it pronoun". In Proceedings of the 6th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC'07), pp.145-150. 2007.
[D. Whitley, 1994]	D. Whitley, "A genetic algorithm tutorial". Statistics and Computing, vol.4 (2), 65–85. 1994.
[D. Xiao Ni, 2007]	Dong Xiao Ni, "Application of Neural Networks to Character Recognition", Proceedings of Students/Faculty Research Day, CSIS, Pace University, May 4th, 2007.
[DT. Pham et al, 2005]	DT. Pham, A.Ghanbarzadeh, E.Koc, S.Otri, S.Rahim, M.Zaidi, "The Bees Algorithm". Technical Note, Manufacturing Engineering Centre, Cardiff University, UK .2005.
[DT. Pham et al, 2006]	DT. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, M. Zaidi. "The Bees Algorithm – A Novel Tool for Complex Optimisation Problems", Proceedings of *IPROMS 2006 Conference*, pp.454-461 .2006.
[DT. Pham et al, 2007]	DT. Pham, E. Koç, J.Y. Lee, J. Phrueksanant, "Using the Bees Algorithm to schedule jobs for a machine", Proc Eighth International Conference on Laser Metrology, CMM and Machine Tool Performance, LAMDAMAP, Euspen, UK, Cardiff, p. 430-439. 2007.
[DT. Pham et al, 2008]	DT. Pham, M. Castellani, and A. A. Fahmy, "Learning the Inverse Kinematics of a Robot Manipulator using the Bees Algorithm," in 17th IFAC World Congress COEX, South Korea, pp. 493-498. 2008.
[E. Bonabeau et al, 1999]	E. Bonabeau, M. Dorigo, G. Theraulaz, "Swarm Intelligence: From Natural to Artificial Systems", New York, NY: Oxford University Press, 1999.
[E. Izquierdo-Torres, 2004]	E. Izquierdo-Torres. 'Collective intelligence in multi-agent robotics: Stigmergy, self-organization and evolution'. University of Sussex, 2004.
[E. J. Bredensteiner et al, 1999]	E. J. Bredensteiner, K. P. Bennett, Multicategory classification by support vector machines, Computational Optimization and Applications, 12 (1999). pp 53-79. 1999.

[E. M. Johansson et al, 1991]	E. M. Johansson, F. U. Dowla, D. M. Goodman, "Backpropagation learning for multilayer feed-forward neural networks using the conjugate gradient method," *Int. J. Neural Syst.*, vol. 2, no. 4, pp. 291–301, 1991.
[E. M. Kleinberg, 2000]	E.M. Kleinberg, "On the algorithmic implementation of stochastic discrimination". IEEE Transactions on Pattern Analysis and Machine Intelligence. 22(5) (2000) 473–490. 2000.
[E. Mjolsness et al, 1989]	E. Mjolsness, D. H. Sharp, and B. K. Alpert, "Scaling, machine learning, and genetic neural nets," Advances in Applied Math., vol. 10, pp. 137–163, 1989.
[E. P. Maillard, 1997]	E. P. Maillard and D. Gueriot, "RBF neural network, basis functions and genetic algorithms," in *Proc. 1997 IEEE Int. Conf. Neural Networks. Part 4 (of 4)*, pp. 2187–2190. 1997.
[E. S. Chan et al, 1996]	E.S. Chan, S.Chen, B.Mulgrew, "Gradient Radial Basis Function Networks for Nonlinear and Nonstationary Time Series Prediction". IEEE Transactions on Neural Networks, 1: pp: 190-194, 1996.
[E. Vonk et al, 1995]	E. Vonk, L. C. Jain, R. Johnson, "Using genetic algorithms with grammar encoding to generate neural networks," in Proc. 1995 IEEE Int. Conf. Neural Networks, Part 4 (of 6), pp. 1928–1931. 1995.
[E. Yfantis, 2006]	E. Yfantis, "Performance Evaluation of feature extraction Algorithms for character Recognition". 2006.
[F. Gandon, 2002]	F. Gandon, "Distributed Artificial Intelligence And Knowledge Management: Ontologies And Multi-Agent Systems For A Corporate Semantic Web", scientific philosopher doctorate thesis in informatics INRIA and University of Nice - Sophia Antipolis. 2002.
[F. Kuhl et al, 1982]	F. Kuhl, P. Giardina, "Elliptic Fourier features of a closed contour", *Comput. Vis. Graphics Image Process.* 18, pp 236-258. 1982.
[F. Lauer et al, 2007]	F. Lauer, C. Y. Suen and G. Bloch, "A trainable feature extractor for handwritten digit recognition," Pattern Recognition, vol. 40, 1816-1824, 2007.
[F. P.Kuhl et al, 1982]	F.P.Kuhl, C.R. Giardina, "Elliptic Fourier features of a closed contour", computer vision graphics and image processing, Vol. 18. pp. 236-258, 1982.

[G. Abandah et al, 2008]	G.Abandah, K.Younis, and M.Khedher, "Handwritten Arabic Character Recognition Using Multiple Classifiers Based on Letter Form," *In Proc. 5th IASTED Int'l Conf. on Signal Processing, Pattern Recognition, & Applications (SPPRA 2008)*, Feb 13-15, pp. 128-133, 2008.
[G. D.M. Serugendo et al, 2006]	G.D.M. Serugendo, M.-P. Gleizes, A. Karageorgos, "Self-organisation and emergence in MAS: an overview", Informatica 30(1). Pages: 45-54, Slovene Society Informatika, Ljubljana, Slovenia, 2006.
[G. F. Miller et al, 1989]	G.F. Miller, Todd, P.M., S.U. Hegde, "Designing neural networks using genetic algorithms". In J.D. Schaffer (Ed.), Proceedings of the Third International Conference on Genetic Algorithms. CA: Morgan Kaufmann. San Mateo, (pp. 379-384). 1989.
[G. G. Rajput et al, 2009]	G. G. Rajput, S. M. Mali, "Handwritten Marathi Numeral Recognition with Fourier Descriptors using Five Fold Cross Validation ", Computer Vision and Information Technology: Advances and Applications, I.K. Int. Pub. India (2009), pp: 528-534. 2009.
[G. G. Rajput et al, 2010a]	G.G. Rajput, S. M.Mali, "Isolated Handwritten Marathi Numerals Recognition Based upon Fourier Descriptors and Freeman Chain Code", *International Journal of Computational Intelligence Research*, Volume: 6, Issue: 2, Print ISSN: 0973-1873. 2010.
[G. G. Rajput et al, 2010b]	G.G. Rajput, S. M. Mali, "Fourier Descriptor based Isolated Marathi Handwritten Numeral Recognition", *International Journal of Computer Applications* (0975 – 8887) Volume 3 – No.4. 2010.
[G. Giacinto, 1998]	G.Giacinto, "Design of multiple classifier systems". PhD thesis, Salerno university, Italy. 1998.
[G. Louloudis et al, 2007]	G. Louloudis, B. Gatos and C. Halatsis, "Text Line Detection in Unconstrained Handwritten Documents Using a Block-Based Hough Transform Approach", *9th International Conference on Document Analysis and Recognition (ICDAR'07)*, pp. 599-603, Curitiba, Brazil, 2007.
[G. R. Ball et al, 2006]	G. R. Ball, S. N. Srihari, H. Srinivasan, "Segmentation-Based And Segmentation-Free Methods for Spotting Handwritten Arabic Words", Tenth International Workshop on Frontiers in Handwriting Recognition, .2006.
[G. Stempfel, 2008]	G. Stempfel, Arbres de Décision, Cours www.lif.univ-mrs.fr/~stempfel/Cours_2.pdf

[G. Vamvakas et al, 2007a]	G. Vamvakas, B. Gatos, S. Petridis and N. Stamatopoulos, "An Efficient Feature Extraction and Dimensionality Reduction Scheme for Isolated Greek Handwritten Character Recognition", *9th International Conference on Document Analysis and Recognition (ICDAR'07),* pp. 1073-1077*,* Curitiba, Brazil, 2007.
[G. Vamvakas et al, 2007b]	G. Vamvakas, B. Gatos, I. Pratikakis, N. Stamatopoulos, A. Roniotis and S.J. Perantonis, "Hybrid Off-Line OCR for Isolated Handwritten Greek Characters", *The Fourth IASTED International Conference on Signal Processing, Pattern Recognition, and Applications* (SPPRA 2007), ISBN: 978-0-88986-646-1, pp. 197-202, Innsbruck, Austria, 2007
[G. Vamvakas et al, 2008]	G. Vamvakas, B. Gatos, N. Stamatopoulos, and S.J.Perantonis, "A Complete Optical Character Recognition Methodology for Historical Documents", *8th International Workshop on Document Analysis Systems (DAS'08)*, pp. 525-532, Nara, Japan, September 2008.
[G. Vamvakas et al, 2009]	G. Vamvakas, B. Gatos, S. J. Perantonis, "A Novel Feature Extraction and Classification Methodology for the Recognition of Historical Documents", *10th International Conference on Document Analysis and Recognition (ICDAR'09)*, pp. 491-495, Barcelona, Spain, July 2009.
[G. Vamvakas et al, 2010]	G. Vamvakas, B. Gatos, S. J. Perantonis, "Handwritten character recognition through two-stage foreground sub-sampling", *Pattern Recognition*, Vol. 43, Issue 8, pp. 2807-2816, 2010.
[G. Weiss, 2000]	G.Weiss, "Multiagent Systems: A Modern Approach to Distributed Modern Approach to Artificial Intelligence", The MIT Press, ISBN-10: 0262731312. 2000.
[H. A. Abbass, 2001a]	H.A. Abbass, "A Single Queen Single Worker Honey-Bees Approach to 3-SAT". Proceedings of the Genetic and Evolutionary Computation Conference, GECCO2001, pages 807--814, San Francisco, USA, 2001.
[H. A. Abbass, 2001b]	H.A. Abbass, "Marriage in Honey Bees Optimization (MBO): A Haplometrosis Polygynous Swarming Approach". Congress on Evolutionary Computation, CEC2001, pages 207--214, Seoul, Korea, 2001.
[H. Akaike, 1974]	H. Akaike, "A new look at the statistical model identification", *IEEE Trans. Automat. Contr.*, vol. AC-19, pp. 716–723, Dec.1974.
[H.Aljuaid et al, 2010]	H.Aljuaid, Z. Muhammad, M.Sarfraz, A Tool to Develop Arabic Handwriting Recognition System Using Genetic Approach, Journal of Computer Science 6 (6): 619-624, 2010.

[H. Cheng et al, 1993]	H. Cheng, W. H. Hsu, M. C. Kuo, "Recognition of handprinted Chinese characters via stroke relaxation", *Pattern Recognition*, 26(4), pp 579 – 593, 1993.
[H.El Fadili et al, 2005]	H.El Fadili, K. Zenkouar, H. Qjidaa, "Evolving Neural Networks using Moment Method for Handwritten Digit Recognition", in Engineering and Technology.2005
[H. Fujisawa et al, 1992]	H.Fujisawa, Y.Nakano and K.Kurino, "Segmentation Methods for character recognition from segmentation to document structure analysis", Proc.IEEE, vol.80, no.7, pp.1079-1092,1992.
[H. Kitano, 1990]	H. Kitano, "Designing neural networks using genetic algorithms with graph generation system," Complex Syst., vol. 4, no. 4, pp. 461–476, 1990.
[H. Kwasnicka et al, 2005]	H. Kwasnicka, M. Paradowski, "Efficiency aspects of neural network architecture evolution using direct and indirect encoding". W: Adaptive and natural computing algorithms. Proceedings of the international conference. Eds Bernardette Ribeiro [i.in.]. Coimbra, Portugal, [March 21-23], 2005. Wien; New York: s. 405-408, Springer 2005.
[H. Lipson et al, 2005]	H. Lipson and V. Zykov," Co-Evolutionary Methods in System Design and Analysis", *15th International CIRP Design Seminar, Shanghai, China*, 2005.
[H. Nishida et al, 1994]	H.Nishida, S.Mori. A model-based split-and merge method for character string recognition. In P.S.P Wang, editor, Document Image Analysis, pages 209–226.World Scientific, 1994.
[H. S.Baird, 1987]	H.S.Baird, "Accurate skew estimation and the top-down analysis of document images". In Proceedings of the first Int, Conf, on computer vision.1987.
[H. T. Siegelmann, 1993]	H.T. Siegelmann, "Foundations of Recurrent Neural Networks. PhD Thesis", Rutgers University, 1993.
[H. Wang et al, 2009]	H. Wang, F. Ji, G. Wei, C. Leung., P. Sum, "Regularization Parameter Selection for Faulty Neural Networks". *International Journal of Intelligent Systems and Technologies* 4:1 .2009.
[H. Zhang, 2004]	H.Zhang, "The optimality of Naïve Bayes". In: Proc. 17th Internat. FLAIRS Conf., Florida, USA. 2004.
[H. Zouari et al, 2002]	H. Zouari, L. Heutte, Y. Lecourtier, A. Alimi. "Un panorama des méthodes de combinaison de classificateurs en reconnaissance de formes". RFIA'2002, Angers, France, vol. 2, pp. 499-508, 2002.

[I. Gasmi et al, 2005]	I. Gasmi, H. Merouani, L. Souici-Meslati, "Combinaison de classificateurs". SETIT 2005. 3rd International Conference: Sciences of Electronic, Technologies of Information and Telecommunications, TUNISIA, March 27-31. 2005.
[I. P. Wright, 1997]	I.P. Wright, "Emotional Agents". PhD thesis, School of Computer Science, The University of Birmingham.PhD thesis, 1997.
[I. Pop, 2006]	I. Pop, "An approach of the Naive Bayes classifier for the document classification", General Mathematics Vol. 14, No. 4 (2006), 135–138. 2006.
[I. S. I. Abuhaiba, 2005]	I. S. I. Abuhaiba, "Arabic Font Recognition using Decision Trees Built from Common Words". CIT 13(3): 211-224 .2005.
[J.A. Fitzgerald et al, 2005]	J.A. Fitzgerald, B.Q.Huang, T.Kechadi, "An Efficient Hybrid Approach for Online Recognition of Handwritten Symbols", LNCS, Mexican international conference on artificial intelligence N°4, vol. 3789, pp. 843-853.2005.
[J. Aschenwald et al, 2004]	J. Aschenwald, G. Tappeiner, U. Tappeiner, E. Tasser, .H. W. Holub, "Statistical Aspects of Multilayer Perceptrons Under Data Limitations", *Computational Statistics & Data Analysis,* Vol. 46. No. 1, pp. 173-188 .2004.
[J. B. Gomm et al, 2000]	J.B. Gomm, D.L. Yu, "Selecting Radial Basis Function Network Centers with Recursive Orthogonal Least Squares Training", IEEE Transactions on Neural Networks, 2: pp: 306-314, 2000.
[J. Bouvrie, 2006]	J. Bouvrie, "Notes on Convolutional Neural Networks", tutorial, 2006.
[J. Branke, 1995]	J. Branke, "Evolutionary algorithms for neural network design and training". Technical Report No. 322, Institute AIFB, University Karlsruhe .1995.
[J. Cao et al, 1994]	J.Cao, M.Ahmadi, M.Shridhar, "Handwritten numerical recognition with multiple features and multistage classifiers", in IEEE International Symposium on circuits and systems, Vol. 6. pp. 323-326, 1994.
[J. D. Montana et al, 1989]	J.D. Montana, L. Davis, "Training Feedforward Neural Networks Using Genetic Algorithms", IJCAI-89, Proceedings of The Eleventh International Joint Conference on Artificial Intelligence, vol. 1, Aug. 20-25, 1989.
[J. F. Fontanari et al, 1991]	J. F. Fontanari, R. Meir, "Evolving a learning algorithm for the binary perceptron," *Network*, vol. 2, no. 4, pp. 353–359, Nov. 1991.

[J. H. Kim et al, 2000a]	J. H. Kim, K. K. Kim, and C. Y. Suen. "An HMM-MLP hybrid model for cursive script recognition". *Pattern Analysis and Applications*, 3:314_324, 2000.
[J. H. Kim et al, 2000b]	J. H. Kim, K. K. Kim, C. Y. Suen, "Hybrid Schemes Of Homogeneous and Heterogeneous Classifiers for Cursive Word Recognition", *Proceedings of the 7th International Workshop on Frontiers in Handwritten Recognition*, Amsterdam, pp 433 - 442. 2000.
[J. Kennedy et al, 1995]	J. Kennedy, R. C. Eberhart, "Particle swarm optimization", In Proceedings of the 1995 IEEE International Conference on Neural Networks", Vol. 4, pp. 1942–1948, 1995.
[J. Kennedy et al, 2001]	J. Kennedy, R.C. Eberhart, "Swarm Intelligence". Morgan Kaufmann. ISBN 1-55860-595-9. 2001.
[J. Kennedy, 1997]	J. Kennedy, " The Particle Swarm: Social Adaptation of knowledge", Proceedings of the IEEE International Conference on Evolutionary Computation, Indianapolis, Indiana, USA, pp. 303-308.1997.
[J. Kittler et al, 1998]	J. Kittler, M. Hatef, R. Duin, and J. Matas. "On combining classifiers". IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 20(3):226–239, Mar. 1998.
[J. Kittler et al, 2000]	J. Kittler, F. Roli, "Multiple Classifier Systems", Proc. First Int. Workshop MCS 2000, Cagliari, Italy, Lecture Notes in Computer Science, vol. 1857, Springer Verlag, Berlin, 2000.
[J. Koh et al, 1995]	J.Koh, M. Suk, , S. M.Bhandarkar, "A Multilayer Self-Organizing Feature Map for Range Image Segmentation," *Neural Networks*, Vol. 8, pp. 67-86.1995.
[J. Laaksonen, 1997]	J. Laaksonen, "Subspace classifiers in recognition of handwritten digits", Ph.D. Thesis, Helsinki University of Technology, 1997.
[J. Moody et al, 1989]	J. Moody, C. J. Darken, "Fast learning in networks of locally-tuned processing units". Neural Computation, 1, pp: 281-294. 1989.
[J. P.Cartlidge, 2004]	J.P.Cartlidge, "Rules of Engagement: Competitive Coevolutionary Dynamics in Computational Systems", Ph.D thesis. 2004.
[J. Paredis et al, 1997]	J.Paredis, R.Westra,, "Coevolutionary computation for Path Planning", *Proceedings 5th European Congress on Intelligent Techniques and Soft Computing*; H.-J. Zimmermann; Verlag Mainz. 1997.

[J. Park, 1999]	J. Park," Hierarchical Character Recognition and its use in handwritten word/phrase recognition". Ph.D thesis. 1999.
[J. Parker, 2001]	J. Parker, "Rank and response combination from confusion matrix data". Information Fusion, 2: pp. 113-120. 2001.
[J. Platt, 1991]	J.Platt, "A Resource Allocating Network for Function Interpolation". Neural computation, 2: pp: 213-215, 1991.
[J. R. Koza et al, 1991]	J. R. Koza and J. P. Rice, "Genetic generation of both the weights and architecture for a neural network," in *Proc. IEEE Int. Joint Conf. Neural Networks (IJCNN'91 Seattle)*, vol. 2, pp. 397–404. 1991.
[J. R. Parker, 1993]	J.R. Parker, " Practical computer vision using C", *John Wiley Sons*, Pages: 476.1993
[J. R. Quinlan, 1986]	J. R. Quinlan, "Induction of decision trees ", Machine Learning, pp. 81-106. 1986.
[J. R.Quinlan, 1993]	J.R.Quinlan, "C4.5: Programs for Machine Learning", Morgan Kaufmann, San Francisco, 1993.
[J. Rissanen , 1978]	J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, Sept. 1978.
[J. T. Favata et al, 1996]	J. T. Favata, G. Srikantan. "A Multiple Feature/Resolution Approach to Handprinted Digit and Character Recognition". *International Journal of Imaging Systems and Technology*, 7:304–311, 1996.
[J. T. Favata et al, 1998]	J. T. Favata, S. N. Srihari, and V. Govindaraju. "Off-line handwritten sentence Recognition". *In Proc. 6th International Workshop on Frontiers of Handwriting Recognition (IWFHR)*, pages 171_176, Taegon-Korea, 1998.
[J. Teo et al, 2001]	J.Teo, H.A. Abbass, "An Annealing Approach to the Mating-Flight Trajectories in the Marriage in Honey Bees Optimization Algorithm". Computer Science Monograph 1: Proceedings of the Post-Graduate Conference on Computer Science (PACCS'01), pages 39-46, ISBN 0-7317-0507-6, Canberra, Australia, 2001.
[J.Venkatesh, C et al, 2009]	J.Venkatesh, C. Sureshkumar, Tamil Handwritten Character Recognition Using Kohonon's Self Organizing Map", IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.12, December 2009.
[J. Vesanto et al, 2000]	J.Vesanto, Esa. Alhoniemi, "Clustering of the Self-Organizing Map". *IEEE Transactions on Neural Networks*, 11(3) :586-600, 2000.

[J. Vesterstrøm et al, 2002]	J. Vesterstrøm, J. Riget, Particle Swarms Extensions for improved local, multi-modal, and dynamic search in numerical optimization, MSc.Thesis, May 2002.
[J. Villegas-Cortez et al, 2005]	J. Villegas-Cortez, C.Avilés-Cruz," Font recognition by invariant moments of global textures". *In Proceedings of International Workshop VLBV05 (Very Low Bit-Rate Video-Coding 2005)*. 15-16 September 2005.
[K. Balakrishnan et al, 1995a]	K. Balakrishnan, V. Honavar, "Evolutionary Design of Neural Architecture-A Preliminary Taxonomy and Guide to Literature", Artificial Intelligence Research group, Iowa State University, CS Technical Report #95-01, 1995.
[K. Balakrishnan et al, 1995b]	K. Balakrishnan, V. Honavar, "Properties of genetic representations of neural architectures", in: Proceedings of the World Congress on Neural Networks, pp. 807-813. 1995.
[K. Bhattacharyya et al, 2009]	K. Bhattacharyya, K. K. Sarma, "ANN-based Innovative Segmentation Method for Handwritten text in Assamese", International Journal of Computer Science Issues, Volume 5, pp9-16, October 2009.
[K. C. Chung et al, 1999]	K.C. Chung, S.C.Kee and SR.Kim, Face Recognition Using Principal Component Analysis of Gabor Filter Responses," ratfg-rts, pp.53, *International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*.1999.
[K. C.Hayes, 1980]	K.C.Hayes. "Reading handwritten words using hierarchical relaxation". Computer Graphics and Image Processing, 14 :344-364,1980.
[K.Chellapilla et al,2006]	K.Chellapilla, S.Puri, P.Simard, "High Performance Convolutional Neural Networks for Document Processing", Tenth International Workshop on Frontiers in Handwriting Recognition (2006).
[K. Chen et al, 1997]	K. Chen, L. Wang and H.S. Chi, Methods of combining multiple classifiers with different features and their applications to text-independent speaker identification, International Journal of Pattern Recognition and Artificial Intelligence, vol. 11, no. 3, pp: 417-445. 1997.
[K. H. Zouari, 2004]	K. H. Zouari "Contribution à l'évaluation des méthodes de combinaison parallèle de classificateurs par simulation". Ph.D. Thesis, Rouen university, 2004.
[K. J. Kim, 2008]	K.J. Kim, S.B. Cho, "Evolutionary ensemble of diverse artificial neural networks using speciation". *Neurocomputing* 71(7-9): pp: 1604-1618 .2008.
[K.Jung at al, 2000]	K.Jung, H.J.Kim, "On-line recognition of cursive Korean characters using graph representation". Pattern Recognition, pp: 399-412.2000.

[K. Kira et al, 1992]	K. Kira, L. A. Rendell, "Feature Selection Problem: Traditional Methods and a New Algorithm". In *Proceedings of Tenth National conference on Artificial Intelligence, San Jose, California*, pages 129–134, 1992.
[K. Koo et al, 2009]	K. Koo, J.P. Yun, S. Choi, J. Choi, D. C. Choi, S. W.Kim, "Character Segmentation and Recognition Algorithm of Text Region in Steel Images", ISPRA'09 Proceedings of the 8th WSEAS international conference on Signal processing, robotics and automation, ISSN: 1790-5117, ISBN: 978-960-474-054-3. 2009.
[K. M. Kim et al, 2004]	K. M. Kim, J. J. Park, Y. G. Song, I. Kim, C. Y. Suen, "Recognition of Handwritten Numerals Using a Combined Classifier with Hybrid Features". SSPR/SPR 2004: pp. 992-1000. 2004.
[K. M. Mohiuddin et al, 1994]	K. M. Mohiuddin, J. Mao, "A Comprehensive Study of Different Classifiers for Handprinted Character Recognition", *Pattern Recognition*, Practice IV, pp. 437- 448. 1994.
[K. O. Stanley, 2004]	K. O. Stanley, "Efficient Evolution of Neural Networks through Complexification, Ph.D Thesis, Texas at Austin University, 2004.
[K.Sankhuangaw et al, 2005]	K.Sankhuangaw, S.Phaiboon, S.Somkuarnpanit, and C.Kimpan "Off-Line Hand Written. Thai Character Recognition Using Ant-Miner Algorithm", Enfomatika, Vol. 8, pp. 276-281, 2005.
[K. Simon, 2003]	K. Simon, "Evolutionary Clustering for Designing RBF Neural Networks", Babeş- Bolyai University, MSc. Thesis, 2003.
[L. A.Torres-Méndez et al, 2000]	L.A. Torres-Méndez, J. C. Ruiz-Suárez, Luis E. Sucar, and G. Gómez, "Translation, Rotation, and Scale-Invariant Object Recognition", IEEE Transactions on systems, man, and cybernetics-part C: Applications and reviews, Vol. 30, No. 1, 2000.
[L. Breiman, 1996]	L. Breiman, "Bagging predictors". Machine Learning. 24(2) (1996). pp.123–140. 1996.
[L. Cordella et al, 2008]	L. Cordella, C. De Stefano, F. Fontanella, C. Marrocco, A Feature Selection Algorithm for Handwritten Character Recognition, International Conference on Pattern Recognition - ICPR , pp. 1-4, 2008.
[L. D. Harmon, 1972]	L.D. Harmon, "Automatic Recognition of Print and Script", Proceedings of the IEEE, vol. 60, no. 10, pp. 1165-1177, 1972.

[L. Gasser, 1992]	L. Gasser, "DAI Approaches to Coordination" in Distributed Artificial Intelligence: Theory and Praxis (eds. N. M. Avouris and L. Gasser) Kluwer Academic Publishers pp 31-51. 1992.
[L. I. Kuncheva, 2000]	L.I. Kuncheva, "Clustering-and-selection model for classifier combination". In Proc. Knowledge-Based Intelligent Engineering Systems and Allied Technologies, Brighton, UK, pp. 185–188. 2000.
[L. I. Kuncheva, 2002]	L. I. Kuncheva. "Switching between selection and fusion in combining classifiers: An experiment". IEEE Transactions on Systems, Man, and Cybernetics, 32(2): pp. 146–156, 2002.
[L. I. Kuncheva, 2004]	L.I. Kuncheva. "Combining Pattern Classifiers. Methods and Algorithms". John Wiley & Sons, New Jersey, 2004.
[L. I. Smith, 2002]	L.I. Smith, "A tutorial on Principal Components Analysis", 2002. http://csnet.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf
[L. Jin et al, 1995]	L.Jin, N.Nikiforuk, Mr. Mr. Gupta, "Uniform Approximation of Nonlinear Dynamic Systems Using Dynamic Neural Networks", *International Conference one Artificial Neural Networks,* p. 191-196, Paris, France, 1995.
[L. Lam et al, 1994]	L. Lam, C. Y. Suen. Increasing experts for majority vote in ocr: Theoretical considerations and strategies. In Proceedings of the 4th International Workshop on Frontiersin Handwriting Recognition, pages 245–254, 1994.
[L.M.ALZoubaidy et al, 2006]	L.M.ALZoubaidy, B.A.AlRifai, Probabilistic Artificial Neural Network for Recognizing the Arabic Handwritten Arabic Characters, Journal of Computer Science, vol(3), pp: 881-886. 2006.
[L. M. Gambardella et al, 1999]	L. M. Gambardella, E. Taillard, G. Agazzi, "MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows," in New Ideas in Optimization, D. Corne, M. Dorigo and F. Glover (eds), 63-76, McGraw-Hill, London, 1999.
[L. N. De Castro et al, 1999]	L.N. De Castro, F.J. Von Zuben, "Artificial Immune Systems. Part I. Basic Theory And Applications", Technical Report No. Rt Dca 01/99, Feec/Unicamp, Brazil, 1999.
[L. Pagie et al, 1997]	L.Pagie and P. Hogeweg "Evolutionary consequences of coevolving targets". *Evolutionary Computation 5*, pp.401-418. 1997.

[L. R. Rabiner, 1989]	L.R. Rabiner "A tutorial on hidden Markov models and select applications in speech recognition", Proceeding of IEEE, Vol 77, N° 2, pp: 257-286. 1989.
[L. S. Oliveira et al, 2001]	L.S. Oliveira, F. Bortolozzi, C.Y.Suen, "Automatic Recognition of Handwritten Numerical Strings: A Recognition and Verification Strategy", *IEEE Transactions on Pattern Recognition and Machine Intelligence*, Vol. 24, No. 11, pp. 1448-1456. 2001.
[L. S. Oliveira et al, 2004]	L.S. Oliveira, R. Sabourin, "Support Vector Machines for Handwritten Numerical String Recognition". 9th international workshop on Frontiers in handwriting Recognition, Tokyo, Japan (Oct. 2004) pp: 39-44. 2004.
[L. S. Oliveira et al, 2005]	L. S. Oliveira, A. S.Britto Jr, R. Sabourin. Improving Cascading Classifiers with Particle Swarm Optimization. In Proceedings of ICDAR'2005. pp.570~574. 2005
[L. Xu et al, 1992]	L.Xu, A.Krzyzak, C.Y.Suen, "Methods of combining multiple classifiers and their applications to handwriting recognition", IEEE Trans, Systems, Man, and Cybernetics. Vol. 22. No. 3. pp. 418-435, 1992.
[M. A. Grônroos, 1998]	M.A. Grônroos, "Evolutionary Design of Neural Networks", Msc thesis, University of Turku, 1998.
[M. A.Potter et al, 1994]	M. A.Potter, K. A.De Jong,. "A cooperative coevolutionary approach to function optimization". In Davidor, Y., Schwefel, H.-P., & M¨anner, R. (Eds.), *Parallel Problem Solving from Nature III*, pp. 249.257. Springer-Verlag, Berlin, Germany. 1994.
[M. Antkowiak, 2006]	M.Antkowiak, "Artificial Neural Networks vs. Support Vector Machines for Skin Diseases Recognition". Master's Thesis, Umeå University. 2006.
[M. Berthold et al, 1998]	M. Berthold, J. Diamond, "Constructive training of probabilistic neural networks". Neurocomputing. v19. 167-183.1998.
[M. Blumenstein et al, 2003]	M. Blumenstein, B. Verma and H. Basli, "A novel feature extraction technique for the recognition of segmented handwritten characters", *ICDAR03. pp: 137*-141. 2003.
[M. Bodn, 2002]	M. Bodn, "A guide to recurrent neural networks and backpropagation", Report from the NUTEK-supported project AIS-8: Application of Data Analysis with Learning Systems, 1999-2001. Holst, A. (ed.), SICS Technical Report T2002:03, SICS, Kista, Sweden. 2002.
[M. Cesar et al, 1990]	M. Cesar, R. Shinghal, "Algorithm for segmenting handwritten postal codes", Int. J. Man Mach Stud., vol. 33, no. 1, pp. 63-80, Jul. 1990.

[M. Dorigo et al, 1999]	M.Dorigo, G. Di Caro, " The ant colony optimization meta-heuristic: New ideas in optimization". McGraw-Hill Ltd., UK, Maidenhead, UK, England. 1999.
[M. Dorigo et al, 2004]	M. Dorigo, V. Trianni, E. Sahin, R. Gross, T.H. Labella, G. Baldassarre, S. Nolfi, J.-L. Deneubourg, F. Mondada, D. Floreano, L.M. Gambardella "Evolving self-organizing behaviors for a swarm-bot," Autonomous Robots, 17(2-3):223-245, 2004.
[M. E. Morita, 2003]	M.E. Morita, "Automatic Recognition of Handwritten Dates on Brazilian Bank Cheques", Quebec University, Montreal, Ph.D thesis. 2003.
[M. Fu. Zhang et al, 1998]	M.Fu. Zhang, , H. Yan ,"A Modulator Classification Scheme with Elastic Net models for handwritten digit Recognition", Vol. 31, No. 12 (1998), pp: 1849-1864. 1998.
[M. Gilloux et al, 1993]	M. Gilloux, J.M. Bertille and M. Leroux, "Recognition of Handwritten Words in a Limited Dynamic Vocabulary", Pre-Proceedings IWFHR III, Buffalo, page 417, 1993.
[M. Gilloux, 1994]	M. Gilloux, "Hidden Markov Models in Handwriting Recognition", Fundamentals in Handwriting Recognition, S. Impedovo (Ed.), NATO ASI Series F: Computer and Systems Sciences, vol. 124, Springer Verlag, 1994.
[M. Kirley, 2002]	**M.Kirley, "Ecological Algorithms: An investigation of adaptation, diversity and spatial patterns in complex optimisation problems", Ph.D thesis.2002.**
[M. Last et al, 2002]	M. Last, H. Bunke, A. Kandel, "A feature-based serial approach to classifier combination". Pattern Analysis and Applications 5(4) (2002). pp. 385–398. 2002.
[M.Mangal et al, 2006]	M.Mangal,M.Pratap Singh," Handwritten English Vowels Recognition Using Hybrid Evolutionary Feed-forward Neural Network", Malaysian Journal of Computer Science (ISSN 0127-9084), vol 19, No 2, 2006.
[M. Shridhar et al, 1986]	M. Shridhar, A. Badreldin, "Recognition of Isolated and Simply Connected Handwritten Numerals", Pattern Recognition, vol. 19, no. 1, page 1, 1986.
[M. Srinivas et al, 1991]	M. Srinivas, L. M. Patnaik, "Learning neural network weights using genetic algorithms—Improving performance by search-space reduction," in *Proc. 1991 IEEE Int. Joint Conf. Neural Networks (IJCNN'91 Singapore)*, vol. 3, pp. 2331–2336. *1991*.

[M.Taghi et al, 2009]	M.Taghi, V.Baghmisheh, N.Pavesic, "Training RBF networks with selective backpropagation", IEEE Congress on Evolutionary Computation,. PP: 3149 - 3156. 2009.
[M. Zorman et al, 1999]	M.Zorman, S.Hleb, M.Sprogar, "Advanced Tool for Building Decision Trees MtDeciT 2.0". IC-AI 1999: 315-318. 1999.
[N. Arica et al, 2001]	N. Arica, F. Yarman-Vural, "An Overview of Character Recognition Focused on Off-line Handwriting", IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 31(2), pp. 216 - 233. 2001.
[N. Arıca, 1998]	N. Arıca, "An off-line Character Recognition System for Free Style Handwriting", MS Thesis, METU 1998.
[N. Azizi et al, 2010]	N. Azizi, N. Farah, M. Sellami, "Off-line handwritten word recognition using ensemble of classifier selection and features fusion", Journal of Theoretical and Applied Information Technology, Vol 14. No. 2 – 2010.
[N. Gorski, 1999]	N. Gorski, V. Anisimov, E. Augustin, O. Baret, D. Price, JC. Simon, "A2iA check reader: families of bank check recognition systems". *Proc. fifth int'l conf. document analysis and recognition,* pp 523–526.1999.
[N. Monmarché, 2007]	N. Monmarché, "Optimisation par Colonies de Fourmis Artificielles : Tour d'horizon", Tutoriel, Laboratoire d'Informatique Université François Rabelais-Tours Yravals. 2007.
[N.P. Matic et al, 2002]	N.P. Matic J.C.Platt, T.Wang, "QuickStroke: An Incremental On-line Chinese Handwriting Recognition System", 16th International Conference on Pattern Recognition, pp: 435 - 439. 2002.
[N. Sharma et al, 2006]	N. Sharma, U. Pal , F. Kimura and S. Pal, "Recognition of Off-Line Handwritten Devnagari Characters Using Quadratic Classifier", Springer LNCS, Volume 4338(2006). Pp: 805-816. 2006.
[N. Tripathy et al, 2004]	N. Tripathy, U. Pal., "Handwriting Segmentation of unconstrained Oriya Text" International Workshop on Frontiers in Handwriting Recognition, pp. 306-311,2004.
[O. D. Trier et al, 1996]	O.D. Trier, A. K. Jain, T.Taxt, "Feature Extraction Methods for Character Recognition – A Survey", *Pattern Recognition*, Vol.29, No.4, pp. 641-662. 1996.
[O. Matan et al, 1991]	O. Matan, C.J.C.Burges. "Recognizing overlapping hand-printed characters by centered-objects integrated segmentation and recognition". In IJCNN, pages 504–511, 1991.

[P. D. Gader et al, 1996]	P.D. Gader, M. A. Khabou. "Automatic Feature Generation for Handwritten Digit Recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1256–1261, 1996.
[P. Eggenbergen-Hotz, 2004]	P. Eggenbergen-Hotz, "Comparing direct and developmental encoding schemes in artificial evolution: A case study in evolving lens shapes". In Proceedings of the Congress on Evolutionary Computation, CEC 2004. pp. 752-757. 2004.
[P. Gallinari et al, 1999]	P. Gallinari, T. Cibas,"Practical complexity control in multilayer perceptrons". *Signal Processing*. Vol. 74. No. 1, pp. 29–46. 1999.
[P. Hu et al, 2002]	P. Hu, Y. Zhao, Z. Yang, J. Wang, "Recognition of Gray Character Using Gabor Filters", *Proceedings of the Fifth International Conference on Information Fusion*, vol.1, pages: 419 – 424. 2002.
[P. Hu, 1962]	P.Hu, "Visual pattern recognition by moment invariants". *In IRE Trans. Inf. Theory* 8, pp. 179-187. 1962.
[P. J. Angeline et al, 1994]	P. J. Angeline, G. M. Sauders, and J. B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," *IEEE Trans. Neural Networks*, vol. 5, pp. 54–65, Jan. 1994.
[P. J. Angeline, 1997]	P. J. Angeline, "Evolving basis functions with dynamic receptive fields," in *Proc. 1997 IEEE Int. Conf. Systems, Man, and Cybernetics, Part 5 (of 5)*, pp. 4109–4114. 1997.
[P. J. Grother, 1992]	P.J. Grother, "Karhunen Loève Feature Extraction For Neural Handwritten Character Recognition". *In proceedings: Applications of Artificial Neural Networks III*. Vol 1709. pp. 155-166. SPIE. Orlando. 1992.
[P. J. Werbos, 1988]	P.J. Werbos. "Generalization of backpropagation with application to a recurrent gas market model". Neural Networks, 1:339-356, 1988.
[P. M. Lallican, 1999]	P.M. Lallican, "Reconnaissance de l'écriture manuscrite hors-ligne : utilisation de la chronologie restaurée du tracé", thèse de l'Université de Nantes. 1999.
[P.Singh et al, 2011]	P.Singh, N.Tyagi, "Radial Basis Function For Handwritten Devanagari Numeral Recognition", IJACSA - International Journal of Advanced Computer Science and Applications, Vol. 2, Nr. 5, p. 126—129. 2011.

[P. Venkatesan et al, 2006]	P.Venkatesan, S. Anitha, "Application of a radial basis function neural network for diagnosis of diabetes mellitus". Curr. Sci., 91: 1195-1199. 2006.
[P. Wu et al, 2006]	P. Wu., C.S. Shieh., J.H. Kao, "The Development of Neural Network Models by Revised Particle Swarm Optimization". *JCIS-2006*. P. 1951-6851.2006.
[P. Y. Simard et al, 2003]	P.Y. Simard, D.Steinkraus, J.Platt, "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis," International Conference on Document Analysis and Recognition (ICDAR), IEEE Computer Society, Los Alamitos, pp. 958-962, 2003.
[Q. Xu et al, 2001]	Q. Xu, L.Lam, CY. Suen, "A knowledge-based segmentation system for handwritten dates on bank cheques". *Sixth international conference on document analysis and recognition*, ICDAR2001, pp 384–388. 2001.
[R. A. Brooks, 1991]	R.A. Brooks, "Intelligence without representation", Artificial Intelligence 47. Pages: 139–159. 1991.
[R. A. Dara, 2007]	R. A. Dara, "Cooperative Training in Multiple Classifier Systems", PhD thesis; Waterloo, Ontario, Canada, 2007.
[R. A. Wilkinson, 1992]	R.A. Wilkinson, "Comparison of massively parallel segmenters", National Inst. of Standards and Technology Technical Report, Gaithersburg, MD, Sept. 1992.
[R. A.Rojas, 1996]	R.A.Rojas, "Graph labeling proof of the backpropagation algorithm".Commun. ACM, 39: 207-215. 1996.
[R. Buse et al, 1997]	R. Buse, Z.Q. Liu, and T. Caelli, A Structural and Relational Approach to Handwritten Word Recognition, IEEE Trans. Systems, Man, and Cybernetics, Part B, 27(5), pp. 847-861, Oct. 1997.
[R. C. Eberhart et al, 2000]	R.C. Eberhart. y. Shi. "Comparing Inertia Weights and Constriction Factors in Particle Swarm optimization". Proceedings of the Congress on Evolutionary Computation. pp. 84-88 .2000.
[R. Duin et al, 2000]	R. Duin, D. Tax. "Experiments with classifier combining rules. Multiple Classificateur Systems", LNCS 1857, Springer, pp. 16-29, 2000.
[R. Ebrahimpour et al, 2009]	R. Ebrahimpour, H.Sarmadi, "A New Approach to Combine Classifiers Trained by NCL". JDCTA 3(4): 80-84.2009.

[R. G. Casey et al, 1982]	R.G. Casey, G. Nagy, "Recursive Segmentation and Classification of Composite Patterns", Proc. 6th Int. Conf. on Pattern Recognition, page 1023, 1982.
[R. G. Kamp et al, 2006]	R. G. Kamp, H. H. G. Savenije, "Optimising training data for ANNs with Genetic Algorithms", Hydrol. Earth Syst. Sci., 10, pp: 603–608, 2006.
[R.Ghosh et al, 2005]	R.Ghosh, M.Ghosh: An Intelligent Offline Handwriting Recognition System Using Evolutionary Neural Learning Algorithm and Rule Based Over Segmented Data Points. Journal of Research and Practice in Information Technology 37(1): 2005.
[R.I.Gandhi et al, 2009]	R.I.Gandhi, K.Iyakutti," An Attempt to Recognize Handwritten Tamil Character Using Kohonen SOM", Int. J. of Advance d Networking and Applications Vol (1) Issue: 03 Pages: 188-192, 2009.
[R. J. Kannan et al, 2009]	R. J. Kannan, R. Prabhakar, "A Comparative Study of Optical Character Recognition for Tamil Script", European Journal of Scientific Research, ISSN 1450-216 X Vol.35 No.4 (2009), pp. 570-582. 2009.
[R. J. Rodrigues et al, 2000]	R. J. Rodrigues, A. C. G. Thomé, "Cursive Character Recognition - A Character Segmentation Method using Projection Profil-based Technique", Proceedings of The 4th World Multi-conference on Systemics, Cybernetics and Informatics SCI 2000 and The 6th International Conference on Information Systems, Analysis and Synthesis ISAS 2000 - Orlando, USA, 2000.
[R. Kala et al, 2010]	R. Kala, H. Vazirani, N. Khawalkar, M. Bhattacharya , "Evolutionary Radial Basis Function Network for Classificatory Problems", *International Journal of Computer Science Applications* 7(4) 34-49. 2010.
[R.Kaur et al, 2011]	R.Kaur, B. Singh, "A Hybrid Neural Approach For Character Recognition System", International Journal of Computer Science and Information Technologies, Vol. 2 (2) ,pp : 721-726, 2011.
[R. M. Bozinovic et al, 1989]	R.M. Bozinovic, S.N. Srihari, "Off-line cursive script word recognition", *IEEE Trans. Pattern Anal. Mach. Intell.* 11 (1) 69–83. 1989.
[R. Nag et al, 1986]	R. Nag, K.H. Wong, F. Fallside, "Script Recognition Using Hidden Markov Models", IEEE ICASSP, Tokyo, pp. 2071-2074, 1986.
[R. P. Wiegand, 2004]	R.P. Wiegand, "Analysis of Cooperative Coevolutionary Algorithms." Ph.D. thesis, George Mason University. 2004.

[R. P.W. Duin, 2000]	R.P.W. Duin "Learned from neural networks". Proceeding of the 5ASCI 2000, 6th Annual Conference of the Advanced School for Computing and Imaging, pp: 9-3.2000.
[R. P.W. Duin, 2002]	R.P.W. Duin,"The Combining Classifier: To Train Or Not To Train?", in: R. Kasturi, D. Laurendeau, C. Suen (eds.), ICPR16, Proceedings 16th International Conference on Pattern Recognition (August 11-15, 2002, Quebec City, Canada), vol. II, IEEE Computer Society Press, Los Alamitos, 765-770. 2002.
[R. Palacios et al, 2003]	R. Palacios, A. Gupta, P. S.P.Wang, "Feedback Based Architecture for Reading Courtesy Amounts on Checks", *Journal of Electronic Imaging*, 12(1), pp. 194-202. 2003.
[R. Palacios et al, 2004]	R.Palacios, A.Gupta, S.P. Wang, "Handwritten Bank Check Recognition of Courtesy Amounts", (January 2004). MIT Sloan Working Paper No. 4461-04; Eller College Working. 2004.
[R. S. Kunte et al, 2007]	R. S. Kunte, R. D. Sudhaker Samuel, "A simple and efficient optical character recognition system for basic symbols in Printed Kannada Text", Sadhana, vol. 32, Part 5, pp. 521–533. 2007.
[R. Schoonderwoerd et al, 1996]	R. Schoonderwoerd, O. Holland, J. Bruten, L. Rothkrantz, "Ant-based load balancing in telecommunications networks", Adaptive Behavior, Vol. 5 No 2, pages :169-207.1996.
[R.M.O et al, 2010]	R.M.O. Cruz, G.D.C. Gavalcanti and T.I. Ren, "Handwritten Digit Recognition Using Multiple Feature Extraction Techniques and Classifier Ensemble", IWSSIP 2010 - 17th International Conference on Systems, Signals and Image Processing. 2010.
[S. J. Lu et al, 2007]	S. J. Lu, L. Li, Chew Lim Tan, "Identification of Latin-Based Languages through Character Stroke Categorization". ICDAR 2007 : 352-356. 2007.
[S. A. Harp, 1989]	S.A. Harp, T. Samad, and A. Guha, "Toward the genetic synthesis of neural networks," in Proc. 3rd Int. Conf. Genetic Algorithms and Their Applications, J. D. Schaffer, Ed. San Mateo, CA: Morgan Kaufmann, pp. 360–369. 1989.
[S. A.Khan, 1998]	S.A.Khan, "Character Segmentation Heuristics for Check Amount Verification", master thesis, 1998.

[S. Arora et al, 2009]	S. Arora, D. Bhattacharjee, M. Nasipuri, M.Kundu, D.K. Basu, "Application of Statistical Features in Handwritten Devnagari Character Recognition", International Journal of Recent Trends in Engineering. ISSN 1797-9617. IJRTE 2009.
[S. Arora et al, 2010]	S. Arora, D. Bhattacharjee, M. Nasipuri, D. Kumar Basu, M. Kundu. "Multiple Classifier Combination for Off-line Handwritten Devnagari Character Recognition", CoRR abs/1006.5913. 2010.
[S. B. Kotsiantis, 2007]	S. B. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques", Informatica 31 (2007) 249-268. 2007.
[S. Camazine et al, 2003]	S. Camazine, J.L Deneubourg, N.R. Franks, J. Sneyd, G. Theraulaz, E. Bonabeau, "Self-Organization in Biological Systems", Princeton: Princeton University Press, 2003.
[S. Dey, 1999]	S.Dey, "Adding Feedback to Improve Segmentation and Recognition of Handwritten Numerals". MIT Thesis. 1999.
[S.Guangmin et al, 2010]	S.Guangmin, C. Zhang, Z. Weiwei, Y.Guangyu, "A New Recognition Method of Vehicle License Plate Based on Genetic Neural Network", 5th IEEE Conference on Industrial Electronics and Applications, Pages: 1662-1666, 2010.
[S. Hassas et al, 2006]	S. Hassas, G. D. Marzo-Serugendo, A. Karageorgos, C. Castelfranchi, "On Self-Organising Mechanisms from Social, Business and Economic Domains", Informatica 30 (2006). Pages: 63–71. 2006.
[S. Haykin, 1994]	S. Haykin, "Neural Networks: A Comprehensive Foundation". Macmillan. 1994.
[S. K. Mostefaoui et al, 2003]	S. K. Mostefaoui, O.F Rana, N. Foukia and al, "Self-Organising Applications: A Survey", 1st Workshop on Engineering Self-Organising Applications (ESOA'03), Melbourne, Australia, 2003.
[S. Khan, 1998]	S.Khan, "Character Segmentation Heuristics for Check Amount Verification". Maters thesis. Massachusetts Institute of Technology .1998.
[S. Madhvanath et al, 1993]	S.Madhvanath, V.Govindaraju, "Holistic Lexicon Reduction", Proc.IWFHR III,Buffalo, pp.71,1993.
[S. Madhvanath et al, 1998]	S. Madhvanath, V. Govindaraju, "Perceptual features for offline handwritten word recognition: a framework for heuristic prediction, representation and matching," Lecture Notes in Computer Science, LNCS 1451, pp. 524-531, 1998.

[S. Marinai et al, 2005]	S.Marinai, M.Gori, G.Soda, "Artificial Neural Networks for Document Analysis and Recognition". IEEE Trans. Pattern Anal. Mach. Intell. 27(1): 23-35 .2005.
[S. Mathur et al, 2010]	**S. Mathur, V. Aggarwal, H. Joshi, A. Ahlawat, "Offline Handwriting Recognition Using Genetic Algorithm," IJCSI, Volume 7, Issue 2, March 2010.**
[S. Muthuraman, 2005]	S.Muthuraman, "The Evolution of Modular Artificial Neural Networks", PhD thesis, The Robert Gordon University, Aberdeen, Scotland, 2005.
[S. Njah et al, 1998]	S. Njah, A.Triki, A.M.Alimi, Système de reconnaissance de code postal, 18 éme Conférence Tunisienne d'Electrotechnique et d'Automatique, Tunisie. 1998.
[S. Oliker et al, 1992]	S. Oliker, M. Furst, and O. Maimon, "A distributed genetic algorithm for neural network design and training," *Complex Syst.*, vol. 6, no. 5, pp. 459–477, 1992.
[S.Ouchtati et al, 2007]	S.Ouchtati, M.Bedda, A.Lachouri, "Segmentation and Recognition of Handwritten Numeric Chains", *International journal of information technology*. Vol 4. No 1, ISSN 1305-2403. 2007.
[S. Song et al, 2005]	S. Song, Z. Yu, X. Chen, S. Song, "A Novel Radial Basis Function Neural Network For Approximation", International Journal of Information Technology, Vol. 11 No. 5. 2005.
[S. Tulyakov et al, 2008]	S.Tulyakov, S.Jaeger, V.Govindaraju, D.Doermann, "Review of classifier combination methods". In: Simone Marinai, H.F. (Ed.), Studies in Computational Intelligence: Machine Learning in Document Analysis and Recognition, pp. 361-386. Springer, 2008
[S. Uchida et al, 2003]	S. Uchida, H. Sakoe, "Handwritten character recognition using elastic matching based on a class-dependent deformation model". ICDAR 2003: 163-167. 2003.
[S. W. Lee et al, 1996]	S.W. Lee, D.J. Lee, H.S. Park, "A New Methodology for Gray-Scale Character Segmentation and Recognition", IEEE Transactions on pattern analysis and machine intelligence, Vol. 18, No. IO. 1996.
[S. Wesolkowski et al, 1997]	S. Wesolkowski, K. Hassanein, "Comparative study of combination schemes for an ensemble of digit recognition neural networks," in *Proc. 1997 IEEE Int. Conf. Systems, Man, and Cybernetics. Part 4 (of 5)*, pp. 3534–3539, 1997.
[S. Z. Selim et al, 1986]	S. Z. Selim, M. A. Ismail, "k-means Type Algorithms: A Generalized Convergence Theorem and Characterization of Local Optimality", IEEE Tran. Pattern Anal. Mach. Intelligence, PAMI-6, 1 (1986), pp: 81-87. 1986.

[T. Binos, 2002]	T. Binos, "Evolving Neural Network Architecture and Weights Using An Evolutionary Algorithm", Minor Thesis RMIT Dept of Comp Sci (2002).
[T. D. Seeley, 1996]	T.D. Seeley, "The Wisdom of the Hive: the Social Physiology of Honey Bee Colonies", Cambridge, Mass., Harvard University Press, 1996.
[T. Gagnon et al, 2003]	T. Gagnon, R. Lefebvre, "A neural-network approach for pre-classification in musical chords recognition", *Proc. 37th Asilomar Conference on Signals, Systems and Computers*, Asilomar, Pacific Grove, November 9-12 .2003.
[T. H.Reiss, 1993]	T.H.Reiss, "Recognizing Planar Objects Using Invariant Image Features", vol. 676 of Lecture Notes in Computer Science, Springer-Verlag, 1993.
[T. Hirano et al, 2001]	T. Hirano, Y. Okada, F. Yoda, "Field extraction method from existing forms transmitted by facsimile. Sixth international conference on document analysis and recognition", *ICDAR2001*, pp 738–742. 2001.
[T. Ho et al, 1994]	T. Ho, J. Hull, and S. Srihari. "Decision combination in multiple classifier systems". IEEE Trans. Pattern Analysis and Machine Intelligence, 16(1):66–75, 1994.
[T. K. Ho, 1998]	T.K. Ho, "The random subspace method for constructing decision forests". Pattern Analysis and Machine Intelligence, IEEE Transactions on 20(8) (1998) pp. 832–844. 1998.
[T. Poggio et al, 1990]	T. Poggio, F.Girosi, "Networks for Approximation and Learning". Proceedings of IEEE, 9: pp: 1481-1495, 1990.
[T. Saba et al, 2010]	T. Saba, G. Sulong, A. Rehman, "A Survey on Methods and Strategies on Touched Characters Segmentation", International Journal of Research and Reviews in Computer Science (IJRRCS), Vol. 1, No. 2, June 2010.
[T. Su et al, 2008]	T. Su, J. Jhang, C. Hou, "A hybrid artificial neural networks and particle swarm optimization for function approximation", *International Journal of Innovative Computing, Information and Control*, Vol 4, N 9, ISSN 1349-4198, pp. 2363—2374 .2008.
[U. Bhattacharya et al, 2005]	U. Bhattacharya, B.B. Chaudhari, R.Ghosh, and M.Ghosh, "On recognition of Handwritten Devnagari Numerals", In proc. Of the workshop on Learning algorithms for pattern recognition (in conjunction with the 18th Australian Joint Conference on Artificial Intelligence), Sydney (2005) 1-7. 2005.

[U. Marti et al, 2001]	U. Marti, H. Bunke. "Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system". *International Journal of Pattern Recognition and Artificial Intelligence*, 15(1):65_90, 2001.
[U. Pal et al, 2003]	U. Pal, A.Sarkar: "Recognition of Printed Urdu Script". *ICDAR 2003*: 1183-1187. 2003.
[V. A. Kovalevsky, 1968]	V.A. Kovalevsky, "Character readers and Pattern Recognition", Spartan Books, Washington D.C.,1968.
[V. D. Parunak, 1996]	V.D. Parunak, " chapter4: Applications of Distributed Artificial Intelligence in Industry", Industrial Technology Institute, 1996.
[V. Lepetit et al, 2004]	V. Lepetit, J. Pilet, , P. Fua, "Point Matching as a Classification Problem for Fast and Robust Object Pose Estimation". In Proceedings of CVPR (2). 244-250. 2004.
[V. N. Manjunath et al, 2005]	V .N. Manjunath Aradhya, G. Hemantha Kumar, P. Shivakumara, S. Noushath, "A New Contour Based Invariant Feature Extraction Approach for the Recognition of Multi-lingual Documents". *Proceedings of International Conference on Multi-lingual Computing and Information Management in Networked Digital Environment,* February 2-4, pp 94 -108. 2005
[V. Podgorelec et al, 2002]	V. Podgorelec, P. Kokol, B. Stiglic, I. Rozman, "Decision trees: an overview and their use in medicine", Journal of Medical Systems, Kluwer Academic/Plenum Press, Vol. 26, Num. 5, pp. 445-463. 2002.
[V. Tereshko et al, 2002]	V. Tereshko, T. Lee, "How information mapping patterns determine foraging behaviour of a honey bee colony", Open Systems & Information Dynamics, vol. 9, pp. 181-193. 2002.
[V. Tereshko et al, 2005]	V. Tereshko, A. Loengarov, "Collective Decision-Making in Honey Bee Foraging Dynamics", Computing and Information Systems Journal, vol. 9, no. 3, 2005.
[W. D. Hillis, 1990]	W. D. Hillis, "Co-evolving parasites improve simulated evolution as an optimization procedure." *Physica* D, *42*, 228.234. 1990.
[W. Gerstner, 1988]	W. Gerstner, "Supervised learning for neural networks", a tutorial with JAVA-exercises, Technical Report .1988. {http://diwww.epfl.ch/~gerstner/wg_pub.html}.

[W. Kinnebrock, 1994]	W. Kinnebrock, "Accelerating the standard backpropagation method using a genetic approach," *Neurocomput.*, vol. 6, no. 5–6, pp. 583–588, 1994.
[W. Lu et al, 1991]	W. Lu, Y. Ren, C. Y. Suen, Hierarchical attributed graph representation and recognition of handwritten Chinese characters, Pattern Recognition, 24(7) pp. 617 – 632, 1991.
[W. Tomasz et al, 1997]	W. Tomasz, J. Jacek, M. Jacek, "Probabilistic neural network for direction estimation". Proceedings of the Third Conference Neural Networks and their Applications and Summer School on Neural Networks Applications to Signal Processing, Kule, 14 X-18 X 97 /Eds R. Tadeusiewicz, L. Rutkowski, J. Chojcan. Częstochowa: Pol. Neural Netw. Soc. s. 173-178. 1997.
[W. Wang et al, 2002]	W. Wang , A. Brakensiek , G. Rigoll, "Combination of Multiple Classifiers for Handwritten Word Recognition", Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), p.117, August 06-08, 2002.
[X. Wang et al, 2005]	X. Wang, X. Ding, C. Liu, "Gabor filters- based feature extraction for character recognition", *pattern recognition*, Vol.38, pages: 369 -369. 2005.
[X. Wu et al, 2008]	X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu et al, "Top 10 algorithms in data mining", Knowl Inf Syst (2008) vol.14. pp:1–37. 2008.
[X. Yao et al, 1996]	X. Yao, Y. Liu, and P. Darwen, "How to make best use of evolutionary learning," in *Complex Systems: From Local Interactions to Global Phenomena*, R. Stocker, H. Jelinek, and B. Durnota, Eds. Amsterdam, The Netherlands: IOS, pp. 229–242. 1996
[X. Yao et al, 1995]	X. Yao, Y. Shi, "A preliminary study on designing artificial neural networks using co-evolution," in Proc. IEEE Singapore Int. Conf. Intelligent Control and Instrumentation, Singapore, pp. 149–154. 1995.
[X. Yao et al, 1996]	X. Yao, Y. Liu, "Ensemble structure of evolutionary artificial neural networks," in *Proc. 1996 IEEE Int. Conf. Evolutionary Computation (ICEC'96)*, Nagoya, Japan, pp. 659–664. *1996*
[X. Yao, 1999]	X. Yao, ``Evolving artificial neural networks," *Proceedings of the IEEE*, 87(9):1423-1447, September 1999.

[Y. Freund et al, 1999]	Y. Freund, R. Schapire, "A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence", 14(5) (1999) 771–780. 1999.
[Y. Huang et al, 1995]	Y. Huang, C. Suen. "A method of combining multiple experts for the recognition of unconstrained handwritten numerals". IEEE Trans. Pattern Analysis and Machine Intelligence, 17(1):90–94, 1995.
[Y. Hwang et al, 1997]	Y. Hwang, S. Bang, "An Efficient Method to Construct a Radial Basis Function Neural Network Classifier". Neural Networks, 10(8): pp.1495-1503. 1997.
[Y. Hwang et al, 1997]	Y. Hwang, S. Bang, "An Efficient Method to Construct a Radial Basis Function Neural Network Classifier". Neural Networks, 10(8):1495-1503. 1997.
[Y. Jin et al, 2004]	Y. Jin, T. Okabe, B. Sendhoff, "Neural network regularization and ensembling using multi-objective evolutionary algorithms", *Proc. of Congress on Evolutionary Computation*, 1-8 .2004.
[Y. LeCun et al, 1990]	Y. LeCun, O. Matan, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel and H. S. Baird: Handwritten Zip Code Recognition with Multilayer Networks, in IAPR (Eds), *Proc. of the International Conference on Pattern Recognition, II:35-40, IEEE, Atlantic City, invited paper, 1990.*
[Y. LeCun et al, 1992]	Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel: Handwritten digit recognition with a back-propagation network, in Lisboa P.G.J. (Eds), *Neural Netwotks, current applications, Chappman and Hall, 1992.*
[Y. Lecun,1998a]	http://yann.lecun.com/exdb/mnist/
[Y. Lecun et al, 1998b]	Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-Based Learning Applied to Document Recognition", Proceedings of the IEEE, Vol. 86, No. 11. (1998), pp. 2278-2324. 1998.
[Y. Lecun et al, 2010]	Y. Lecun, K. Kavukcuoglu, and C. Farabet, "Convolutional Networks and Applications in Vision," in Proc. International Symposium on Circuits and Systems (ISCAS'10), 2010.
[Y. Li et al, 2006]	Y. Li, J. Li, L.Meng. Character Recognition Based on Hierarchical RBF Neural Networks. In Proceedings of ISDA (1)'2006. pp.127~132. 2006.

[Y. Lu et al, 2002]	Y. Lu, CL.Tan, "Combination of multiple classifiers using probabilistic dictionary and its application to postcode recognition". *Pattern Recognition, vol:* 35. Pages: 2823–2832. 2002.
[Y. Yohannes et al, 1999]	Y. Yohannes, J. Hoddinott «Classification and Regression Trees- An Introduction" International Food Policy Research Institute, 1999.
[Z. Guo et al, 1992]	Z. Guo, R. E. Uhrig, "Using genetic algorithms to select inputs for neural networks," in *Proc. Int. Workshop Combinations of Genetic Algorithms and Neural Networks (COGANN-92)*, pp. 223–234. 1992
[Z. Yang et al, 2007]	Z. Yang, Y. Che, K.W. Cheng, "Genetic algorithm-based RBF neural network load forecasting model", in: Proceedings of the IEEE General Meeting on Power Engineering Society, pp. 1-6. 2007.

Résumé

La problématique générale qui oriente nos travaux de cette thèse s'intéresse aux interactions collectives entre des entités autonomes et à l'émergence au niveau supérieur de l'ensemble d'un système complexe. Cette technique porte sur la manière de diviser un problème entre des entités relativement simples mais capables de s'auto organiser afin d'accomplir d'une façon collective un but commun. Elle s'insère dans la problématique plus vaste de l'Intelligence collective. Cette vision distribuée de problèmes a suscité l'attention croissante des chercheurs pour la qualité de ses applications et ses avantages tels que le parallélisme, la vitesse, flexibilité…etc. L'intelligence en essaim constitue un domaine à part entière de l'intelligence collective. Les travaux décrits dans cette thèse donnent lieu à des applications principalement inspirées des comportements collectifs d'oiseaux et d'abeilles pour la reconnaissance de caractères et plus particulièrement la reconnaissance de chiffres arabes manuscrits et de lettres anglaises. Cette thèse présente ainsi quatre contributions principales à base de deux types de réseaux de neurones à savoir le MLP et le RBF. Dans le cadre de la pensée distribuée, nous avons essayé de centrer nos efforts sur l'application de nouvelles méthodes de l'intelligence en essaim à savoir PSO, l'algorithme des abeilles et l'optimisation par colonies d'abeilles artificielles en leur appliquant en premier lieu en tant que classificateurs statistiques et puis comme optimiseurs de deux types de réseaux de neurones ou encore comme espèces dans un système coopératif co-évolutionnaire. En outre, nous avons employé la combinaison de pluieurs classificateurs afin d'améliorer le taux de reconnaissance. L'intérêt et l'originalité des approches présentées pour la reconnaissance de caractères résident dans l'aspect de la résolution collective ou la coopération de plusieurs réseaux MLP ou RBF en offrant de bien meilleurs résultats que d'autres approches employées dans la littérature. Les approches développées ont été étudiées en montrant leur intérêt en termes du taux de reconnaissance.

Mots clès : reconnaissance de caractères manuscrits, l'intelligence en essaim,réseaux de neurones.

Abstract

The general problematic that directs our work of this thesis focuses on the collective interactions between autonomous entities and on emergency of a complex system. This technique relates to the manner of dividing a problem between entities relatively simple but able to self-organize in order to achieve in a collective way a common goal. It forms a part of the vaster problematic of collective Intelligence. This distributed vision caused an increasing attention of researchers for the quality of its applications and its advantages such as parallelism, speed, flexibility… etc. Swarm intelligence constitutes a sub-field of collective intelligence. The work described in this thesis gives place to applications mainly inspired of the collective behaviors of birds and bees for characters recognition and particularly for recognition of handwritten Arabic numerals and English letters. This thesis thus presents four contributions based on two types of neural networks namely MLP and RBF neural networks. Within the framework of distributed vision, we tried to focus our efforts on the application of new "swarm intelligence" methods such as PSO, the bees algorithm and artificial bees colony optimization applying them as statistical classifiers and then as optimizers of the two types of neural networks or as species in a cooperative co-evolutionnary system. We have also used classifiers combination to improve accuracy. The Interest and originality of the presented approaches for characters recognition exist in the aspect of collective intelligence or the cooperation of several MLP or RBF neural networks by offering enhanced results than other approaches employed in literature. The developed approaches were studied by showing their interest in terms of recognition rate.

Key words: handwritten characters recognition, swarm intelligence, evolutionary neural networks.