RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ DJILLALI LIABES SIDI BEL ABBES

FACULTÉ GÉNIE ÉLECTRIQUE

DÉPARTEMENT D'ÉLECTRONIQUE

THESE DE DOCTORAT

POUR OBTENIR LE GRADE DE

DOCTEUR EN SCIENCE

SPÉCIALITÉ : ÉLECTRONIQUE

OPTION : TRAITEMENT DE SIGNAL ET DE L'IMAGE

PRÉSENTÉE PAR

BENBAKRETI SAMIR

INTITULÉ :

Développement d'un système de reconnaissance en ligne du manuscrit Arabe

SOUTENUE LE : 23/07/2019 DEVANT LE JURY COMPOSÉ DE :

Pr ANANI Macho, (UDL-SBA)	(Professeur)	: Président
Pr BOUKELIF Aoued, (UDL-SBA)	(Professeur)	: Directeur
Dr ADJOUDJ Réda, (UDL-SBA)	(MCA)	: Invité
Dr ELARBI-BOUDIHIR Mohamed (ESI-SBA)	(MCA)	: Examinateur
Dr AMAR BENSABER Djamel, (ESI-SBA)	(MCA)	: Examinateur

2018/2019

Table des matières

Liste des acronymes et abréviations

ANN	Artificial Neural Network
ASCII	American Standard Code for Information Interchange
BP	Back Propagation
CD	Contrastive Divergence
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DBN	Deep Belief Network
DCP	Digital Curve Partitioning
DNN	Deep Neural Networks
DTW	Dynamic Time Warping
DTDNN	Distributed Time Delay Neural Network
EM	Espérance-Maximisation
EQM	Erreur Quadratique Moyenne
FLC	Fuzzy Logic Controller
GPU	Graphics Processing Unit
GRNN	Generalized Regression Neural Network
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
HWR	Handwriting Recognition
IHM	Interface Homme Machine
InkML	Digital Ink Markup Language
J2ME	Java 2 Micro Edition
KL	Kullback-Leibler
KPPV	K Plus Proche Voisin
MCMC	Markov Chain Monte Carlo
MLP	Multi Layer Perceptron
NN	Neural Network

OCR	Optical Character Recognition
OVO	One Over One
PAW	Part of Arabic Word
PC	Personal Computer
PCD	Persistant Contrastive Divergence
PDA	Personal Digital Assistant
PNN	Probabilistic Neural Network
RB	Réseau Bayésien
RBF	Radial Basis Function
RBFe	Radial Basis Function Exact
RBM	Restricted Boltzmann Machine
RNA	Réseau de Neurone Artificiel
RVB	Rouge Vert Bleu
SVM	Support Vector Machine
TDNN	Time Delay Neural Network
UNESCO	United Nations Educational, Scientific and Cultural Organization
WD	Writer Dependent
WI	Writer Independent
W3C	World Wide Web Consortium
XML	Extensible Mark-Up

Liste des tableaux

Liste des figures

Remerciements

Les travaux rapportés dans cette thèse sont le résultat de cinq années de doctorat au laboratoire Réseaux de Communication, Architectures et Multimédia (RCAM) de l'université de Djillali Liabes (UDL) et le laboratoire de recherche appliquée sur les technologies de l'information et de la Communication (LARATIC) de l'Institut National de Télécommunication et des Technologies de l'information et de la communication (INTTIC). Je tiens à remercier toutes les personnes qui ont fait partie de cette expérience, pour leurs conseils et leur soutien.

Je tiens d'abord à remercier et à témoigner ma gratitude à mon directeur de thèse Professeur BOUKELIF Aoued. Je le remercie pour la confiance qu'il m'a accordée dès le début de ma thèse, pour son soutien et pour ses encouragements qui représentaient une grande source de motivation pour moi. Je le remercie pour sa disponibilité malgré un emploi du temps chargé, pour ses conseils et pour son implication et contributions dans ces travaux.

Sur un plan technique, je voudrais remercier tout particulièrement Dr BENBAKRETI Soumia, Mme BENBAKRETI Safia, Dr BENOUIS Mohamed et Dr ROUMANE Ahmed pour leurs collaborations. Ils ont toujours été disponibles, à l'écoute de mes nombreuses questions, et se sont toujours intéressés à l'avancée de mes travaux. Les nombreuses discussions que nous avons eues ainsi que leurs conseils sont pour beaucoup dans le résultat final de ce travail.

Je voudrais remercier aussi le jury en l'occurrence ; Dr ELARBI-BOUDIHIR Mohamed, Dr AMAR BENSABER Djamel et Dr ADJOUDJ Réda d'avoir accepté de rapporter cette thèse et d'en être rapporteurs. La version finale de ce mémoire a bénéficié de leur lecture très attentive et de leurs remarques précieuses. Je tiens à remercier Pr ANANI Maachou d'avoir accepté d'être président du jury. Je remercie également tous les membres du jury d'avoir accepté d'assister à la présentation de ce travail.

Finalement j'adresse un grand merci à ma famille, qui a su me soutenir, me supporter, m'encourager . . . pendant toute la durée de ma thèse et plus particulièrement durant les derniers mois de rédaction qui n'ont pas toujours été des plus agréables.

Chapitre I

Introduction

I.1 Contexte et problématique

L'écriture est une des méthodes fondamentales utilisée dans la communication humaine. Il ne fait aucun doute que l'écriture a pris une place prépondérante dans la vie humaine. Elle a été utilisée pendant longtemps comme une expansion de la mémoire humaine. Grâce à l'écriture, les gens peuvent matérialiser et enregistrer leurs opinions, activités, impressions, etc. Beaucoup de vieilles cultures et civilisations, sciences et religions ont été corroborées par la vertu de l'écriture.

En fait, l'écriture est une collection de marques graphiques artificielles sur une surface ; ces marques sont combinées avec la relation conventionnelle de Mark, normalement définie dans le contexte d'un langage naturel [1]. Plus précisément, ces marques représentent des caractères de langue ou des alphabets. Chaque langue humaine (par exemple, arabe, anglais, etc.) est caractérisée par un ensemble spécifique de caractères. De plus, il y a un groupe de règles qui spécifient les formes de connexion de caractères ; et en fonction de ces règles, les caractères sont placés de manière à former des unités de haut niveau (c'est-à-dire des mots, voir des phrases et des textes entiers).

De nos jours, la reconnaissance de texte est un domaine de recherche très actif, un axe qui s'inscrit dans le domaine de la reconnaissance de formes, dont le but est de développer des systèmes automatisés capables de lire des textes aussi efficacement que les êtres-humains. L'idée de base est de reconnaître le contenu, de fournir un texte de transcription à partir d'images capturées par divers moyens comme un scanner, une caméra, des enregistrements vidéo ou n'importe quel signal d'écriture sauvegardé d'une manière dynamique (pas d'image dans ce cas de figure). Le terme reconnaissance de texte peut inclure des processus associés en plus de la tâche principale qui est la reconnaissance, parmi ces processus on peut citer : la localisation de texte, l'amélioration de la qualité d'image, d'autres traitements qui visent à faciliter la tâche au classifieur et des étapes de post-reconnaissance comme la correction d'orthographe.

La reconnaissance de texte possède de nombreuses applications utiles et intéressantes allant de la numérisation et l'indexation de manuscrits historiques à la lecture automatique des plaques d'immatriculation des voitures, le traitement automatique des chèques bancaires, le tri automatique des enveloppes, le traitement automatique des formulaires et des tâches plus complexe impliquant la reconnaissance de mots. En outre, en raison de l'utilisation généralisée des papiers dans la vie de tous les jours et de la nécessité de les stocker électroniquement pour une sauvegarde et une récupération efficaces, il existe un besoin de systèmes de traitement de documents hautement fiables et robustes. D'autre part, l'avènement des terminaux de petite dimension tels que les smartphones et les tablettes, a permis de zapper cette étape, c.-à-d. qu'on peut actuellement introduire électroniquement un texte et le traiter automatiquement.

Si la reconnaissance de texte est effectuée sur des images de texte imprimées à la machine, elle est communément appelée reconnaissance de texte imprimé ou reconnaissance optique de caractères (OCR). D'autre part, la reconnaissance de texte manuscrite concerne la

reconnaissance de texte à partir d'images contenant du texte manuscrit saisi par des humains. Bien que la reconnaissance du texte imprimé soit beaucoup plus facile que la reconnaissance du texte manuscrit, elle a ses propres défis tels que la reconnaissance de documents dégradés, la reconnaissance de texte imprimé en utilisant des fontes multiples ou inhabituelles et des documents ayant des alignements de texte et des orientations non uniformes. La reconnaissance de texte manuscrit, en revanche, fait face à la plupart des défis ci-dessus, en plus de ses propres problèmes, comme une énorme variation dans les écritures humaines que ça soit entre différents scripteurs ou bien pour le même scripteur. Des spécificités et des défis sur lesquels nous reviendrons ultérieurement.

La reconnaissance de texte imprimé à la machine peut être considérée comme un problème résolu pour de nombreuses applications pratiques comme le système de tri d'adresses postales ; alors que les défis sont loin d'être résolus en matière de reconnaissance de texte manuscrit [2]. Cette dernière est scindée en deux approches : hors ligne et en ligne. Dans la reconnaissance de texte hors ligne, le texte est reconnu de manière statique une fois le processus d'écriture terminé. Alors que la reconnaissance de texte en ligne est effectuée sur des données capturées en temps réel. Ceci est généralement fait lorsque le texte est écrit sur les écrans tactiles des tablettes ou des téléphones intelligents. Dans ce cas de figure, des informations telles que la pression du stylet et les informations temporelles sur l'écriture sont disponibles, ce qui n'est pas le cas dans la reconnaissance de texte hors ligne.

La principale motivation du travail présenté dans cette thèse est le fait que les systèmes de traitement en ligne de la langue Arabe (et les retombés commerciaux qui vont avec) ne sont pas encore arrivés à maturité par rapport aux autres langues étrangères, en raison de l'absence des ressources linguistiques comme les bases de données et le manque d'outils pour la collecte, l'annotation et le prétraitement des ensembles de données.

L'une des principales étapes de la mise en place d'un système de reconnaissance de texte est l'apprentissage du système. Pour bien entrainer un système de reconnaissance de texte, il est important que nous ayons suffisamment d'échantillons d'apprentissage pour chaque classe à reconnaître. Ces classes peuvent représenter des caractères, des mots ou des phrases. Des bases de données de référence sont élaborées pour faciliter la recherche dans ce domaine, de sorte que les systèmes puissent être bien formés et évalués sur la base des données de test. La création de ces bases de données est une activité longue et coûteuse. La quantité de données d'apprentissage qui est nécessaire pour permettre à un dispositif de reconnaissance du texte et d'apprendre correctement est directement liée au nombre de classifications des unités de reconnaissance de base dans un système de reconnaissance, c-a-d, il devrait y avoir suffisamment d'échantillons d'apprentissage pour chaque classe de reconnaissance. Ainsi, pour un script donné, si nous pouvons avoir des représentations alternatives pour les unités de reconnaissance, la représentation qui conduit au nombre minimum de classes semble avoir un net avantage. Pour une quantité donnée de données d'apprentissage, le dispositif de reconnaissance de texte aura plus d'échantillons par classe, ce qui peut conduire à un apprentissage plus robuste que celui d'un système de reconnaissance qui utilise une représentation qui a plus de classes. D'autre part, dans les situations où peu de données d'apprentissage sont disponibles, le système de reconnaissance qui a le jeu de modèles le plus compact devrait fonctionner de manière plus robuste. Ainsi, la modélisation choisie est une décision importante. Pour certains scripts comme le Latin, les caractères sont le choix de modélisation le plus simple et le plus évident. Mais, pour d'autres scripts comme l'Arabe, le choix n'est pas si évident.

La reconnaissance est une tâche difficile à réaliser selon l'application demandée. C'est pour cela que les chercheurs ont développé plusieurs approches. On distingue l'approche statistique et l'approche structurelle. Une différence essentielle réside dans la représentation de la forme : vecteur de caractéristiques dans l'approche statistique, agencement de primitives pour l'approche structurelle. Nous allons rappeler quelques-unes de ces approches couramment utilisées en les classant en 4 groupes : structurelle (eg : KPPV...), statistique (SVM...), neuronale (MLP, TDNN, RBF...) et stochastique (HMM, RB ...). Cette classification qui a été développée par [3] comprend une part d'arbitraire car celle-ci n'est pas unique. Elle est bien résumée dans [4].

I.2 Objectif de la thèse

Le but de cette thèse est d'ouvrir la porte à la reconnaissance en ligne des textes en Arabe, en fournissant dans un premiers temps l'ensemble de données de base qui peuvent former un texte Arabe. D'autre part, ça permettrai aussi aux chercheurs de travailler sur de grands ensembles de données, de réduire leur effort sur le prétraitement des données et les orienter vers les procédures de reconnaissance.

Une fois l'ensemble de données est collecté d'une manière en ligne, nous allons tout d'abord traiter la reconnaissance de caractère arabe manuscrit. Par la suite, nous traiterons le problème de reconnaissance du mot, bien évidemment, ces challenges sont pleins de problématiques : le caractère cursif de l'écriture arabe, l'aspect on-line, le problème de variabilité que ça soit mono ou multi-scripteurs. L'objectif est d'obtenir le meilleur taux de reconnaissance possible en utilisant plusieurs méthodes neuronales.

I.3 Contribution de la thèse

Dans [4], nous avons construit une première base de données de caractères isolés que nous allons développer dans ce travail en rajoutant des échantillons de caractères quelque soient leur emplacement dans le mot (début, milieu et fin), cette nouvelle version de la base de données contient 2800 caractères et 4800 mots représentant les noms de villes algériennes. L'ensemble de données est acquis sans contrainte, faisant intervenir 20 scripteurs différents. La motivation de cette collecte de données est de montrer comment il est vraiment nécessaire et bénéfique pour les éventuels futurs travaux de reconnaissance du manuscrit Arabe, ainsi que pour valider les résultats obtenus sur un ensemble de réseaux de neurones utilisés.

Par la suite, et après avoir réalisé la tâche d'apprentissage mentionnée ci-dessus, nous allons procéder à la classification des caractères et des mots en utilisant un ensemble de méthodes neuronales telles que : MLP, TDNN, RBF, GRNN, PNN et le DBN. En effet, nous allons commencer par utiliser les méthodes peu-profondes pour faire la classification des caractères. Cette partie peut être considérée comme la pierre angulaire des travaux futurs car même la reconnaissance de phrases peut se faire grâce à des méthodes de segmentation qui conduiront vers la reconnaissance de caractères. L'utilisation des méthodes peu-profondes possèdent autant d'atouts que d'inconvénients que nous développerons ultérieurement. Par conséquent, nous allons utiliser le Deep Learning ou ce qu'on appelle réseau de neurones profond pour effectuer la reconnaissance de mots.

Nous aimerions mentionner que de nombreuses techniques développées dans le cadre de ce travail de thèse peuvent aussi bien fonctionner pour la reconnaissance de texte en ligne. En outre, les techniques présentées dans cette thèse peuvent être applicables à d'autres langues, en

particulier les langues qui utilisent l'écriture arabe comme l'ourdou et le persan, mais nous n'avons pas évalué ces techniques sur ces langues.

I.4 Organisation de la thèse :

Nous allons commencer dans un premier temps par décrire un système de reconnaissance de l'écriture manuscrite Arabe en ligne. En effet ce système est composé de blocs usuels qui ne changent pas quel que soit la manière de procéder (en ligne ou hors ligne). Tout d'abord, dans les deux premiers chapitres, nous détaillerons dans l'ordre le bloc d'acquisition de données, celui de l'extraction de caractéristiques, de classification et de poste traitement. Par la suite, étant donné que la thèse traite le manuscrit Arabe, nous nous pencherons sur les caractéristiques de l'écriture Arabe ainsi que les chalenges de celle-ci.

Le chapitre 3 présente un état de l'art des travaux réalisés sur la reconnaissance du manuscrit Arabe et par conséquent toutes les méthodes de classification qui ont été explorées dans le domaine. Etant donné que l'objectif de la thèse est de développer un système de reconnaissance en ligne du manuscrit Arabe à l'aide de réseaux de neurones, l'accent sera mis dans ce chapitre sur les réseaux de neurones peu-profonds (MLP, TDNN, RBF, GRNN et PNN), leurs architectures et leurs méthodes d'apprentissage.

Dans le chapitre 4, nous allons voir en détail le concept du deep learning à travers le réseau profond Deep Belief Network. En effet le fondement théorique est différent dans ce genre de réseau, car le DBN représente un empilement de machines de Boltzmann restreintes (RBM), et l'apprentissage est effectué couche par couche en utilisant l'algorithme de divergence contrastive (CD). L'architecture du DBN est composée d'une partie non-supervisée qui permet de faire une initialisation des poids, et d'une partie supervisée qu'on peut voir comme un MLP classique.

Le chapitre 5 sera consacré à l'étude de l'évaluation des différentes méthodes neuronales développées précédemment dans les chapitres 3 et 4 sur la reconnaissance en ligne du manuscrit Arabe. Mais avant d'attaquer ce volet, nous avons commencé par concevoir notre base de données intitulées NOUN-DATABASE v2, cette base se distingue de la version précédente par le fait qu'elle ne renferme pas les caractères Arabes isolés seulement mais aussi les caractères quelque soient leurs positions dans le mot c.-à-d., au début au milieu et à la fin. De plus elle renferme aussi des mots Arabes représentant les 48 villes d'Algérie. Nous enchainerons par une phase d'extraction de caractéristiques pertinentes du manuscrit Arabe et de classification à travers l'utilisation des méthodes neuronales. Le paramétrage des différents blocs est effectué dans le but d'obtenir les meilleurs résultats.

La réalisation de ce travail a demandé la consultation de plusieurs articles et thèses de référence dans le domaine, qui sont répertoriés dans la partie "Bibliographie" en dernière section du rapport.

Ce manuscrit tend à expliquer de manière plus détaillée les résultats que nous avons publiés dans un article de revue joint en fin de ce rapport.

Chapitre II

Système de reconnaissance et caractéristiques de la langue Arabe

II.1 Introduction

Le système de reconnaissance de l'écriture aide à développer et à faire progresser le processus d'automatisation, et améliorer l'interaction entre l'homme et les machines, dans de nombreuses applications, y compris la bureautique, une grande variété de services bancaires, entreprise, etc... . Peu de recherches se sont intéressées à la reconnaissance de l'écriture arabe en raison de la difficulté de la tâche et de la langue. Dans les sections suivantes, nous allons décrire les concepts liés et utilisés dans la reconnaissance de l'écriture manuscrite.

II.2 La reconnaissance des formes

La reconnaissance des formes est l'une des capacités les plus importantes du cerveau humain. S'appuyant sur cette capacité, les êtres humains extraient des informations utiles de tout ce qui les entourent. Aujourd'hui, comme la technologie informatique numérique a été largement utilisée et développé pour simuler cette capacité unique, la stimulation de cette capacité sur des machines automatisées devient plus réaliste.

II.3 La reconnaissance du manuscrit

Dans le cadre de la communication, en plus de véhiculer des idées via le son et les signaux, la communication par écriture manuscrite est également une tendance depuis longtemps. En utilisant des scripts de sa langue ou parfois si la langue utilisée n'a pas son propre script, en utilisant des scripts d'une autre langue, on peut exprimer ses idées en utilisant des écritures.

La reconnaissance de l'écriture manuscrite peut être définie comme la tâche de transformer le texte représenté dans la forme spatiale des marques graphiques en sa représentation symbolique. Les yeux humains peuvent voir et identifier ce qui est écrit sur un document, qu'il s'agisse d'une écriture humaine pure ou d'une impression à la machine. En fait, non seulement identifier les textes, mais aussi le faire correspondre à la langue afin d'obtenir le sens du document. Dans la reconnaissance de l'écriture manuscrite, la justification de la reconnaissance est d'obtenir un document avec écriture manuscrite et d'amener le contenu du document dans un format lisible par machine [5, 6, 7]. La reconnaissance de l'écriture manuscrite peut être catégorisée de manière générale en reconnaissance d'écriture manuscrite en ligne et hors ligne.

La reconnaissance de l'écriture manuscrite (HWR) a été définie par Plamondon et Srihari [1] comme la tâche de transformer un langage représenté dans sa forme spatiale de marques graphiques en sa représentation symbolique. Contrairement au HWR, la reconnaissance optique de caractères (OCR) cible les applications reconnaissant le texte tapé à la machine. OCR et HWR sont des domaines importants dans le domaine de la reconnaissance de formes. Les deux domaines ont évolué régulièrement au cours de l'histoire et ont toujours été un terrain d'expérimentation favorable pour de nouvelles idées dans la reconnaissance des formes, donnant naissance à un ensemble passionnant de sujets de recherche et produisant de nombreuses applications pratiques puissantes.

Cependant, étant donné que de nombreuses expériences sur de nouvelles idées dans la reconnaissance de formes ont été menées sur des caractères isolés, les résultats ne sont pas toujours immédiatement reflétés dans les applications HWR [8]. Bien que considéré comme un domaine technologique bien développé, HWR reste un domaine de recherche scientifique active et d'ingénierie créative [9]. Outre la reconnaissance, l'écriture manuscrite est associée à d'autres types d'analyse, tels que la vérification de la signature, l'identification de l'auteur, etc.

Il existe différents types de problèmes présentant une complexité variable dans le domaine des HWR. D'une part, il y a le cas de caractères isolés écrits à l'intérieur de boîtes graphiques dans lesquelles le problème de segmentation est déjà résolu. À l'extrême opposé, il y a le cas de l'écriture cursive et sans restriction dans laquelle des mots ou des parties d'un mot sont écrits d'un seul trait à l'aide de ligatures reliant des lettres adjacentes. Les systèmes HWR ont une forte expérience dans l'utilisation de cette difficulté graduée [10].

II.4 Reconnaissance off-line versus on-line du manuscrit

La reconnaissance de l'écriture manuscrite peut être divisée en deux catégories : la reconnaissance d'écriture manuscrite hors ligne et en ligne basée sur le mode de saisie de texte manuscrit. Dans la reconnaissance en ligne, une séquence de coordonnées ordonnées dans le temps, représentant le mouvement de la pointe du stylo, est capturée, tandis qu'en mode hors ligne seule l'image du texte est disponible.

II.4.1 Reconnaissance on-line

L'introduction des appareils numériques sensibles à la pression comme les tablettes PC, a suscité un regain d'intérêt pour les recherches comme la reconnaissance en ligne de l'écriture manuscrite [11]. Dans cet aspect-là, la reconnaissance de textes (caractères ou mots) est effectuée en analysant la chaîne de paires de coordonnées (x, y) à partir de la sortie du dispositif numérique utilisé. Pour l'entrée de texte, une tablette sensible à la pression ou un appareil numérique en général avec un stylet est utilisé à des fins d'écriture. L'affichage sur l'écran des appareils est enregistré dans un fichier séparé avec des détails sur le texte ; ces derniers varient d'un appareil à un autre, comme la paire de coordonnées, le temps, la pression, etc.

La figure 2.1 montre que l'acquisition de données, le prétraitement, la segmentation, l'extraction de caractéristiques, la classification et le post-traitement sont les étapes les plus courantes qui forment la structure du système de reconnaissance en ligne de l'écriture Arabe [12]. L'existence de ces étapes n'est pas obligatoire dans la structure du système (c'est-à-dire que la tâche de reconnaissance peut être accomplie sans l'ensemble complet de ces étapes). Par conséquent, certaines étapes sont intégrées dans d'autres étapes [13].

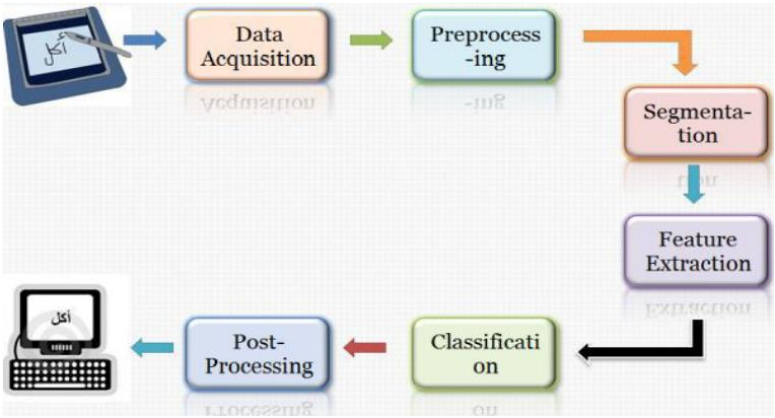


Figure II.1 : Les étapes d'un système de reconnaissance d'écriture manuscrite Arabe en ligne. [12]

II.4.2 Reconnaissance off-line

Dans la reconnaissance de l'écriture hors ligne, les textes sont pris comme une image provenant d'un scanner ou d'un appareil photo numérique. L'image doit être traitée de telle sorte que les textes soient numérisés dans des formats binaires basés sur le modèle de couleur en pixels d'image [6].

Après avoir pris l'image de texte dans des formats binaires, les étapes de reconnaissance ressemblent plus ou moins au cas de la reconnaissance d'écriture manuscrite en ligne avec quelques exceptions. Premièrement, le prétraitement de l'écriture hors ligne nécessite différentes activités de prétraitement comme la détection de la ligne du texte d'écriture manuscrite. En plus de cela, il n'y a pas d'informations temporelles attachées à l'image scannée donc, le classificateur n'a pas la moindre information sur la manière et l'ordre de l'écriture. Ceci limitera en fait la flexibilité du classificateur pour trouver le texte numérisé ou reconnu. Par rapport à la reconnaissance d'écriture manuscrite hors ligne, la reconnaissance d'écriture manuscrite en ligne nécessite que le processus d'écriture manuscrite et la conversion soient simultanés [5].

Le marché mondial favorise la reconnaissance de l'écriture manuscrite en ligne par rapport à la reconnaissance hors ligne. En effet, le système de reconnaissance d'écriture manuscrite en ligne capture les informations temporelles ou dynamiques de l'écriture et améliore la précision hors ligne. De plus, il permet l'interactivité, donc les erreurs de reconnaissance peuvent être corrigées immédiatement avec une série de tests. Cela permettra l'adaptation du dessin des caractères pour l'utilisateur. Dans le même temps, cela donnera également la possibilité d'autoriser l'appareil numérique à s'ajuster en conséquence avec le style d'écriture de l'utilisateur. Ainsi, il y a adaptation de l'utilisateur à la machine et de la machine à l'utilisateur [6].

II.4.2 Structure d'un système de reconnaissance de l'écriture manuscrite on ligne Arabe

Les sections suivantes mettent la lumière sur les étapes qui forment la structure typique de la reconnaissance en ligne de l'écriture manuscrite.

II.4.2.1 Acquisition de données

En fait, les données sont nécessaires pour les phases d'apprentissage et de test des systèmes de reconnaissance de formes. Ces données sont disponibles sous forme de base de données [14].

Par conséquent, les développeurs de systèmes de reconnaissance d'écriture manuscrite Arabe se sont intéressés aux bases de données de l'écriture Arabe en ligne, telle que : La base de données ADAB (Arabic DAtaBase) développée en coopération entre l'Institut des Technologies de communication (*Institute for Communications Technology IfN*), l'Ecole Nationale d'Ingénieurs de Sfax (ENIS) et le groupe de recherche (*Research Group on Intelligent Machines REGIM*) de Sfax en Tunisie.

II.4.2.2 Prétraitement

L'étape de prétraitement suit l'acquisition des données dans le système de reconnaissance. Les données d'entrée sont traitées, à ce stade, afin de faciliter la tâche restante au système de reconnaissance. Habituellement, cette étape aborde les problèmes de réduction des données, d'élimination des imperfections et de normalisation.

La plupart des numériseurs effectuent un échantillonnage temporel uniforme lorsqu'ils sont utilisés par les scripteurs pour saisir un texte à la main. Souvent, cet échantillonnage résulte d'un sur-échantillonnage des régions de mouvement lent du stylet et d'un sous-échantillonnage des régions de mouvement rapide des stylets [15]. De nos jours, les dispositifs de capture en ligne, même les moins chers, produisent des signaux assez fluides ; par conséquent, le prétraitement sur le bruit du signal, tel que le ré-échantillonnage, n'est pas un composant crucial pour les systèmes de reconnaissance. Il existe de nombreuses techniques qui peuvent être utilisées dans la phase de prétraitement telles que :

- Normalisation de la taille : Cette technique caractérise les systèmes de reconnaissance indépendants de la taille. Elle prend en charge les systèmes avec la possibilité de reconnaître l'écriture en ligne si l'écriture est dans des styles différents (petits ou grands).
- Recentrage de la forme : Cette technique calcule le centre de gravité de la forme saisie, puis centre la forme de telle sorte que son origine devienne le centre de gravité.
- Ré-échantillonnage : Cette technique normalise la vitesse d'écriture. Il effectue la normalisation en ré-échantillonnant les séquences de points et en répartissant les points uniformément sur une courbe échantillonnée. Sur la base de la vitesse d'écriture, les points de stylo acquis sont distribués inégalement le long de la trajectoire d'écriture, il y aura moins de points dans le sous-échantillonnage où la vitesse est élevée et plus de points dans le sur-échantillonnage où le mouvement du stylo est lent.
- Elimination du bruit : Les données d'entrée fournies par les tablettes peuvent contenir une quantité considérable de bruits qui compliquent le travail dans les étapes suivantes. Ces bruits sont généralement causés par les numériseurs utilisés ainsi que par les mains tremblantes [6]. Habituellement, la technique est utilisée pour éliminer les points dupliqués en forçant une distance minimale entre les points consécutifs.

II.4.2.3 Segmentation :

La segmentation séparera les symboles d'une entrée donnée. Les données d'entrée non segmentées vont créer des difficultés pour le moteur de reconnaissance, donc la segmentation est vitale pour le processus de reconnaissance [16]. Pour segmenter un mot donné, la première activité consistera à identifier les points de segmentation. Un trait (ou plutôt un stroke) sera extrait entre deux points de segmentation. Les performances d'un système de reconnaissance de caractères dépendent de la façon dont l'entrée est segmentée.

D'après [17], il existe trois types de points de segmentation : les transitions, les extrema locaux et les points de division. Les transitions sont les points situés avant un PENDOWN ou après un PENUP. Un PENDOWN ou un PENUP est le moment qui marque la posée ou la levée du stylo. Les extrema locaux sont les points dont les composantes verticales ont les valeurs extrêmes. Les points de division sont définis comme étant ceux qui divisent un stroke en deux parties égales lorsque celui-ci présente une extension verticale supérieure à un seuil prédéterminé. Une autre façon de segmenter est de calculer la différence d'angle entre les vecteurs qui sont faits à partir de points consécutifs [18]. Un ensemble de vecteurs consécutifs dont les différences d'angle consécutives sont presque les mêmes peut composer un arc.

Une fois que les strokes sont identifiés par un processus de segmentation, ils peuvent être appariés à des ensembles de caractéristiques prédéfinis. Ces caractéristiques peuvent exprimer les strokes identifiés par des lignes et / ou des courbes. Oh, J. dans [19] a décrit la segmentation des caractères comme une approche analytique vs holistique, une segmentation explicite vs implicite et une reconnaissance par segmentation par opposition à une segmentation par reconnaissance.

- Segmentation analytique et holistique

Dans l'approche de segmentation analytique, des points de segmentation hypothétiques seront générés avant le début du processus de reconnaissance, tandis que dans l'approche holistique de la segmentation, seules les caractéristiques globales représentant l'ensemble de l'entrée sont extraites et évaluées.

- Segmentation implicite et explicite

La reconnaissance par segmentation et la segmentation par reconnaissance sont les deux formes extrêmes de l'approche analytique en fonction du choix du point de rupture. Le premier est considéré comme une segmentation explicite alors que le second est implicite.

Dans le cas de la segmentation par reconnaissance ou segmentation implicite, une meilleure segmentation est obtenue en extrayant le chemin qui représente le meilleur candidat en termes de métrique appliquée pour évaluer les chemins après la fin de l'évaluation de reconnaissance. C'est-à-dire que le point de segmentation est décidé sur la base du retour d'un dispositif de reconnaissance pour une reconnaissance optimale à partir d'un trait donné. En fait, cette approche pourrait mener à des hypothèses exhaustives.

D'un autre côté, la segmentation explicite ou la reconnaissance par segmentation détermine les points de segmentation en fonction de certaines caractéristiques comme la cuspide, la fermeture et la largeur de caractère estimée. Les combinaisons ordonnées et légales de ces segments ont été générées en tant que chaînes de caractères possibles.

Une fois les données d'entrée segmentées, les strokes correspondent à un ensemble d'entités prédéfini. Dans le processus de mise en correspondance de ces traits, il existe quatre propriétés à traiter [18]. Ces propriétés sont le type, la direction, la courbure et la longueur relative. La propriété type est exprimée avec trois choix, spécifiquement, ligne, arc et lune où la lune est une arcade et un demi-cercle fortement courbés. Huit directions possibles ont été mises de côté pour les propriétés de direction de sorte que chaque type aura sa propre direction. La direction est mesurée en calculant l'angle direct du premier point du segment à son dernier point et il est étiqueté en conséquence. La propriété de courbure spécifie dans quelle direction réside par

rapport à une ligne joignant le point de départ et de fin d'un stroke donné. Le centre de gravité du segment et l'angle du premier point à ce point sont calculés. Si cet angle se situe du côté gauche de l'angle direct du premier au dernier point, on suppose qu'il s'agit d'une courbure dans le sens des aiguilles d'une montre et sinon, il s'agit d'une courbure dans le sens inverse des aiguilles d'une montre. Finalement, la propriété de longueur relative est utilisée pour décrire le rapport du stroke par rapport à la longueur totale des données d'entrée.

II.4.2.4 Extraction des caractéristiques :

Cette phase permet d'extraire les caractéristiques pertinentes des données brutes de l'écriture manuscrite en ligne. Comme il est inutile d'entrer directement ces données dans la phase de reconnaissance, l'extraction des caractéristiques est nécessaire pour obtenir une reconnaissance efficace et précise.

De nombreuses fonctionnalités peuvent être extraites de l'écriture manuscrite en ligne, où la plupart de ces caractéristiques ont été utilisées dans divers systèmes de reconnaissance de l'écriture arabe en ligne [20, 21, 22, 23, 24]. En fait, la reconnaissance de l'écriture manuscrite doit être effectuée en fonction d'une classe sélectionnée de ces caractéristiques, de sorte que cette classe doit être petite et capable d'atteindre efficacement la reconnaissance. Le compromis entre les caractéristiques sélectionnées et leurs exigences de calcul doit être soigneusement traité [23].

Les caractéristiques de base pour la reconnaissance comprennent les coordonnées x et y des points rééchantillonnés, les moments du stroke jusqu'au quatrième ordre, la direction générale et la courbure du trait, la longueur du trait, le rapport d'aspect, la surface du stroke, nombre et direction des points dans différentes sous-fenêtres, histogrammes de projection dans les directions X et Y et coefficients de Fourier des séquences x et y [18].

On va décrire quelques une de ces caractéristiques qui sont utilisées à des fins de reconnaissance :

- Penup et le pendown : C'est une variable booléenne indiquant si oui ou non le stylo touche la surface d'acquisition. Chaque trait de l'écriture en ligne est représenté comme une séquence de points à travers laquelle le stylo passe d'un état down a un état up. Le nombre de points dans le stroke et leurs distances varient également en fonction de la vitesse d'écriture.
- Hat-feature : Indique si un trait retardé a été supprimé à la position horizontale considérée.
- Vitesse : La vitesse à laquelle un point donné est écrit est calculée avant le rééchantillonnage, on calcule aussi l'interpolation / extrapolation.
- Coordonnée X : La position x ou la position horizontale du point après la normalisation.
- Coordonnée Y : La position y ou la position verticale d'un point après la normalisation.
- Direction d'écriture : le cosinus et le sinus de l'angle entre le segment de ligne commençant au point et l'axe des x.
- Courbure : Le cosinus et le sinus de l'angle entre les lignes au point précédent et le point suivant.
- Aspect de proximité : l'aspect d'une trajectoire
- Pente voisine : Cosinus et sinus de l'angle (t) de la droite du premier au dernier point de voisinage.

- Courbure de proximité : La longueur de la trajectoire dans le voisinage est divisée par la longueur maximale.
- Linéarité de voisinage : La distance carrée moyenne de chaque point à proximité de la ligne droite du premier au dernier point de voisinage.
- Ascendants / Descendants : Le nombre de points au-dessus / en dessous du corpus dans le x-voisinage d'un point donné.

II.4.2.5 Classification :

L'étape de classification est l'étape essentielle de la reconnaissance. Elle regroupe les caractéristiques extraites de l'écriture arabe en ligne, dans un ensemble de catégories ou de classes finies. Ces catégories peuvent être des mots, des pseudo-mots ou des lettres en langue Arabe. La reconnaissance des formes a de nombreuses méthodes pour résoudre les problèmes de classification [25]. Voici quelque unes de ces méthodes.

Reconnaissance statistique (par exemple, HMM) : La théorie de la décision bayésienne est une approche statistique fondamentale du problème de la classification des modèles. En utilisant cette approche, le problème est posé en termes probabilistes.

Le modèle de Markov caché est l'une des approches de l'apprentissage automatique. C'est un modèle statistique dans lequel le système en cours de modélisation est supposé être un processus de Markov avec des états (cachés) non observés. C'est une approche dynamique de reconnaissance de formes qui a été initialement introduite pour la reconnaissance de la parole. Cependant, dernièrement, elle a été appliquée avec succès à la reconnaissance de l'écriture manuscrite et a même été utilisée pour la classification des séquences d'images [11, 26].

Un HMM peut être caractérisé par les propriétés suivantes :

i. Le nombre d'états (N) dans le modèle
ii. Le nombre de symboles d'observation distincts (M) par état
iii. Matrice de probabilité de transition d'état $A = \{a_{ij}\}$
iv. Distribution des probabilités des symboles d'observation dans l'état j mentionnée B
v. Probabilité initiale de l'état $\pi = \{\pi_{ij}\}$

En général, une représentation complète d'un HMM nécessite la spécification de deux paramètres de modèle (N et M), la spécification de symboles d'observation et la spécification des trois mesures de probabilité A, B et π. Cela peut être exprimé de manière compacte comme : $\lambda = (A, B, \pi)$.

Avec la spécification de HMM décrite ci-dessus, il existe trois problèmes d'intérêt fondamentaux que le HMM traite dans des applications réelles. Le premier problème est, étant donné la séquence d'observation et un modèle λ, comment calculer efficacement la probabilité de la séquence d'observation donnée ? Le deuxième problème est, étant donné la séquence d'observation et le modèle λ, comment choisir la séquence d'état correspondante qui est optimale dans un sens significatif ou qui explique le mieux les observations ? Enfin, comment ajuster le paramètre du modèle λ de façon à maximiser la probabilité d'observation ? Le premier problème est traité avec l'algorithme Forward-Backward qui est un algorithme d'inférence calculant la probabilité de toutes les variables d'état cachées en fonction d'une séquence d'observations. Le deuxième problème peut être résolu en utilisant l'algorithme de Viterbi. C'est un algorithme de programmation dynamique qui permet de trouver la séquence la plus probable

d'états cachés, parfois aussi appelée chemin de Viterbi, ce qui donne une séquence de l'observation. Par conséquent, il est utilisé pour trouver une seule meilleure séquence d'état pour une séquence d'observation donnée. De la même manière, le troisième problème peut être résolu en utilisant l'algorithme de Baum-Welch. Cet algorithme est utilisé pour trouver les paramètres inconnus de HMM.

L'application des ensembles flous (Fuzzy sets)

La reconnaissance de formes à l'aide d'ensembles flous est une technique permettant de déterminer les fonctions de transfert non linéaires qui mappent les entités en classes [30]. Ces fonctions de transfert permettent au concepteur IHM (interface homme machine) de développer des systèmes capables de trouver des modèles dans les données obtenues à partir des capteurs d'entrée, puis de mapper ces modèles sur des actions spécifiques qui peuvent contrôler le fonctionnement de l'ordinateur.

Les ensembles retenus sont les ensembles qui ont été utilisés fréquemment dans la vie des humains. Dans un ensemble, un élément est soit un membre de l'ensemble ou non. D'autre part, les ensembles flous permettent aux éléments d'être partiellement dans un ensemble. Chaque élément reçoit un degré d'appartenance à un ensemble. Cette valeur d'appartenance peut aller de 0 (pas un élément de l'ensemble) à 1 (un membre de l'ensemble).

Machine à vecteurs de support (SVM)

Les SVM sont également une autre approche d'apprentissage machine qui sont des modèles d'apprentissage supervisé avec des algorithmes d'apprentissage associés qui analysent les données et reconnaissent les modèles utilisés pour la classification et l'analyse de régression. Il est utilisé pour classer les vecteurs de longueur fixe [11]. Un classificateur SVM peut être entraînée en trouvant un hyper-plan de marge maximale en terme de combinaison linéaire de sous-ensembles (vecteurs de support) de l'ensemble d'apprentissage [28]. Cependant, il est indiqué que les SVM ne fournissent pas les probabilités a posteriori de la reconnaissance primitive, ce qui est nécessaire pour coupler les résultats au modèle génératif de l'apprentissage des mots [20].

Les réseaux de neurones

Les réseaux de neurones sont des modèles artificiels qui tentent d'imiter la fonction d'apprentissage du cerveau humain. Ils sont composés d'un ensemble de neurones ou d'unités de traitement reliés entre eux au moyen de poids de connexion. Ces réseaux sont structurés de manière à apprendre par ajustement continu des poids des connexions [30].

La reconnaissance des formes est une application importante des réseaux de neurones. La reconnaissance peut être mise en œuvre en utilisant un réseau de neurones feed-forward pouvant être entraîné. Pendant l'apprentissage, le réseau est formé pour associer les sorties aux schémas d'entrée. Lorsque le réseau est utilisé, il identifie le modèle d'entrée et essaie de l'associer à une sortie (classe) [31]. Nous reviendrons en détails sur ces réseaux dans le chapitre suivant.

II.4.2.6 Post-traitement

Le rôle de l'étape de post-traitement est d'augmenter la précision de la reconnaissance et de produire des résultats plus significatifs. Quelques efforts ont été faits pour corriger les mots Arabes qui ont été obtenus par la classification. Dans la plupart des cas, des dictionnaires de

mots arabes sont utilisés pour renvoyer les mots les plus proches des résultats de la classification.

II.5 Le script arabe

L'Arabe est l'une des langues sémitiques, la quatrième langue la plus parlée au monde et le troisième système d'écriture le plus utilisé au monde. Il est parlé par plus de 400 millions de personnes dans le monde, dont plus de 200 millions de personnes qui parlent l'Arabe comme première langue [32]. C'est la langue officielle de 22 pays dans le monde (UNESCO 2015). Le script Arabe est également utilisé par beaucoup d'autres langues comme l'ourdou, le persan et l'ouïghour.

Dans cette section, nous allons présenter un aperçu du script Arabe. Nous discuterons principalement du système d'écriture arabe sans évoquer les aspects linguistiques de la langue comme sa grammaire et sa prononciation car ils ne sont pas directement liés au sujet de la présente thèse.

II.5.1 Caractères et signes diacritiques

L'écriture Arabe est cursive aussi bien dans sa formes imprimée que manuscrite. L'alphabet arabe contient des lettres qui représentent les consonnes. L'Arabe est écrit de droite à gauche et a 28 caractères de base. Les caractères n'ont pas différentes formes, c.-à-d. la notion de majuscule et minuscule n'existe pas. La figure II.2 montre les caractères du script Arabe.

ا ب ت ث ج ح خ د ذ ر ز س ش ص ض

ط ظ ع غ ف ق ك ل م ن ه و ي

Figure II.2 : Les lettres de l'alphabet Arabe.

Un caractère possède une forme de noyau (connue sous le nom de Rasm) et peut avoir des points au-dessus (comme ت ن) ou au-dessous (comme ي ب) des formes de base. Beaucoup de caractères partagent la même forme de base et ne diffèrent que par le nombre et la position des points. Il y a huit caractères (خ ذ ز ض ظ غ ف ن) ayant un point au-dessus, deux caractères (ت ق) ayant deux points au-dessus, deux caractères (ث ش) ayant trois points au-dessus, deux caractères (ب ج) ayant un point au-dessous, un caractère (ي) ayant deux points au-dessous, et les 13 caractères restants n'ont pas de points au-dessus ou au-dessous des formes de noyau.

Les caractères sont normalement connectés en utilisant un coup horizontal appelé *Kashida*. La figure II.3 montre des exemples de textes imprimés et manuscrits en arabe. On peut observer à partir de la figure que les textes manuscrits ainsi que les textes imprimés à la machine sont cursifs et connectés.

ارحموا من في الارض يرحمكم من في السماء

Figure II.3 : Échantillon de textes arabes manuscrits (ci-dessus) et imprimés (ci-dessous).

En dehors des points, les caractères peuvent avoir d'autres signes diacritiques comme Shadda (ّ), Hamza (ء) et Sukun (ْ). Les voyelles courtes sont également écrites en tant que signes diacritiques. Il y a trois voyelles courtes (Fatha, dama et kasra) en Arabe, ces diacritiques (en dehors des points) sont utilisés pour le guidage phonétique. Dans les textes manuscrits, et dans une plus large mesure même dans les textes imprimés à la machine, la plupart des signes diacritiques (à l'exception des points obligatoires et de Hamza) ne sont pas écrits mais peuvent être déduits par les lecteurs du contexte. Une exception à cette règle est l'écriture du Coran, de documents juridiques et de textes écrits dans le but d'enseigner l'Arabe. La figure II.4 montre un exemple de texte arabe, avec et sans les signes diacritiques optionnels.

أَحِبَّ لِأَخِيكَ مَا تُحِبُّ لِنَفْسِكَ

أحب لأخيك ما تحب لنفسك

Figure II.4 : Exemples de textes en arabe, avec et sans les signes diacritiques optionnels

À titre d'exemple, certains manuscrits historiques contiennent des textes arabes même en l'absence de points. La figure II.5 montre un exemple d'une telle page de manuscrit historique ayant des textes arabes (du Saint Coran) sans les points.

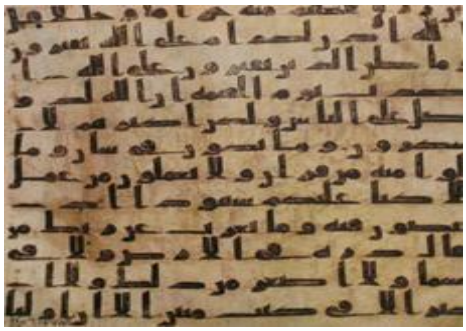


Figure II.5 : Exemple de verset coranique sans la présence de signes diacritiques [34].

II.5.2 La position du caractère, les mots et les pseudo-mots :

Les caractères dans l'écriture Arabe peuvent prendre différentes apparences visuelles en fonction de leur position dans un mot. Comme le script Arabe est cursif, un caractère dans un mot est connecté à ses caractères adjacents. 23 des 28 caractères peuvent prendre jusqu'à quatre formes dépendant de la position. Au début lorsque le caractère est connecté à un caractère après lui mais n'est connecté à aucun caractère avant (comme le premier caractère d'un mot), au milieu quand le caractère est connecté à la fois avant et après à ses caractères adjacents, a la fin lorsque le caractère est connecté à un autre avant lui mais n'est pas connecté à un caractère après (comme le dernier caractère dans un mot), et isolé quand il n'y a aucun caractère connecté. Les cinq caractères restants ne peuvent prendre que deux des quatre formes dépendant de la position, c'est-à-dire, la fin et isolé. La figure II.6 montre les caractères Arabes avec leurs formes dépendantes de la position.

N°	Character name	Position in the word			
		End	Middle	begining	Isolated
1	Alef	أ			أ
2	Ba	لعب	صبر	بحر	ب
3	Ta	كرة	ستر	تمر	ت
4	Tha	نعث	نثر	ثلج	ث
5	Jim	ثلج	خجل	جمل	ج
6	Ha	ملح	بحر	حلم	ح
7	kha	صراخ	نخل	خجل	خ
8	Dal	يد			د
9	Dhal	معاذ			ذ
10	Ra	مصر			ر
11	Zain	معز			ز
12	Sin	فرس	بسر	سهل	س
13	Chin	نعش	نشر	شرق	ش
14	Sad	مص	مصر	صراخ	ص

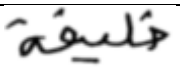
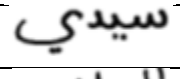
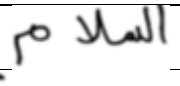
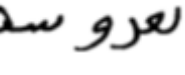
N°	Character name	Position in the word			
		End	Middle	begining	Isolated
15	Dhad	مرض	مضلة	ضمير	ض
16	Ttah	تسليط	مطر	طفل	ط
17	Thah	حظ	نظر	ظهر	ظ
18	Ain	متاع	معطف	عنب	ع
19	Ghain	صبغ	لغم	غمامة	غ
20	Fa	انف	طفل	فم	ف
21	Qaf	ازرق	مقص	قمر	ق
22	Kaf	حراك	مكيال	كم	ك
23	Lam	محل	ملعب	لهو	ل
24	Mim	فم	ضمير	ملعب	م
25	Noun	لبن	عنب	نزل	ن
26	Waw	هو	اسود	وهر	و
27	He	مياه	نهر	هرم	ه
28	Ya	هي	ويل	يسر	ي

Figure II.6 : Caractères Arabes et leurs formes dépendantes de la position.

Ces cinq caractères qui ne prennent que deux formes ne permettent pas aux caractères d'après de se connecter à eux et par conséquent ils n'ont pas de forme de début et de milieu. S'ils viennent au début, ils prennent les formes isolés et s'ils viennent au milieu, ils prennent les formes finales.

Le fait que certains caractères ne permettent pas aux caractères de se connecter après eux, un mot en Arabe peut être divisé en plusieurs composants. Chacune des composantes séparées d'un mot est appelée Part of Arabic Word (PAW), c'est ce qu'on appelle aussi pseudo-mot. Le tableau II.1 montre quelques exemples de mots en Arabe ayant un nombre différent de PAW.

Tableau II.1 : Exemples de mots en Arabe avec différents nombres de pseudo-mots. (Source : base de données IFN / ENIT [35]).

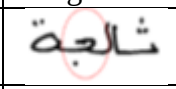
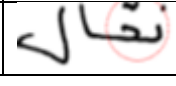
Nombre de PAW	Ecriture machine	Ecriture manuscrite
1	خليفة	
2	سيدي	
3	السلام	
4	العروسة	

II.5.3 Ligature :

Un aspect important du système d'écriture arabe est la présence de ligatures spéciales. Certaines séquences de caractères peuvent être écrites dans des formes compactes spéciales au lieu de les relier simplement en utilisant les Kashidas horizontaux. Les ligatures, lorsqu'elles sont écrites, prennent des apparences visuelles qui sont sensiblement différentes de la simple concaténation des caractères. La séquence de caractères lam-alif est une ligature obligatoire, c'est-à-dire, lorsque l'écriture de lam est suivie par alif, elle est toujours écrite dans la forme ligaturée (لا) au lieu de la forme non-ligaturée (ل ا). Toutes les séquences de caractères ne forment pas de ligatures, mais certaines séquences de caractères sont écrites en tant que ligatures.

Il est important de noter que, seulement lam-alif est une ligature obligatoire. D'autres ligatures ne sont pas obligatoires et donc certains auteurs peuvent les écrire sous forme de ligatures alors que d'autres auteurs peuvent écrire les mêmes séquences de caractères dans les formes non-ligaturées, même dans des contextes similaires. En fait, il est également possible qu'un scripteur écrit une séquence de caractères comme une ligature dans un cas alors qu'il l'écrit dans une forme non-ligaturée dans d'autres cas. Le tableau II.2 illustre certaines séquences de caractères, leurs formes de ligature, et leurs formes non-ligaturées (sauf lam-alif) à la fois dans la machine imprimée et les textes manuscrits. Dans [36], l'auteur a présenté une bonne analyse des ligatures arabes et de leur importance dans la reconnaissance de texte.

Tableau II.2 : La forme ligaturée et non-ligaturée de la même séquence de caractères, (Source : base de données IFN / ENIT [35]).

Séquence de caractère	Ecriture machine		Ecriture manuscrite	
	Ligature	Pas de Ligature	Ligature	Pas de ligature
Lam-jim				
Lam-alif		-		-
Noun-ha				

II.6 Les chalenges de la reconnaissance du manuscrit arabe :

La reconnaissance du texte arabe manuscrit est confrontée à un certain nombre de défis, il s'agit d'un axe de recherche très ouvert. Certains problèmes rencontrés dans la reconnaissance de texte arabe manuscrit sont similaires à ceux rencontrés par d'autres scripts, comme la variabilité de l'écriture chez différents auteurs et même pour un seul auteur, les problèmes liés aux textes et aux chevauchements. Cependant, comme le script arabe a ses propres caractéristiques, la reconnaissance de texte arabe manuscrite est confrontée à des problèmes spécifiques qui doivent être résolus de manière appropriée. Dans la partie suivante, nous présenterons les principaux défis en matière de reconnaissance de texte arabe manuscrit liés aux caractéristiques de l'écriture arabe.

II.6.1 Formes varient en fonction de la position :

L'un des principaux problèmes liés à la reconnaissance du texte en arabe est le fait que les caractères peuvent prendre des formes différentes en fonction de leur position dans un mot. Bien qu'il n'y ait que 28 caractères différents dans le script arabe, les variations basées sur la position conduisent à près de 100 formes différentes de caractères. Pour certains caractères, les variations entre leurs différentes formes en fonction de la position ne sont pas très grandes, tandis que pour les autres caractères, les variations intra-caractère sont assez grandes comme le montre la figure II.7. Nous pouvons observer que les formes de caractères sont visuellement très différentes les unes des autres

Figure II.7 : Différentes formes de caractères dépendantes de la position du caractère « ayn (ع) ». (Source : IFN / ENIT [35]).

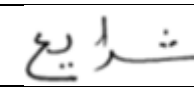
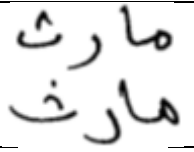
De plus, certaines formes de caractère peuvent sembler très différentes des autres formes de caractères de la même lettre, alors qu'en même temps, elles peuvent ressembler beaucoup aux formes de caractères de certains autres personnages, comme illustré dans la figure II.8.

Figure II.8 : Caractères qui se ressemblent alors qu'ils appartiennent à des classes différentes (Rayn et Fa).

II.6.2 Signes diacritiques :

Un autre problème majeur lié à la reconnaissance du texte arabe manuscrit est lié aux points et autres signes diacritiques présents dans les textes arabes. Comme mentionné précédemment, certains caractères ont des points au-dessus ou au-dessous d'eux. Les scripteurs écrivent ces points de différentes façons. Certains auteurs écrivent clairement ces points, comme c'est le cas avec les textes imprimés à la machine. Mais parfois, les écrivains égarent les points de sorte que les points ne viennent pas directement au-dessus ou au-dessous du caractère pour lequel ils étaient destinés. Parfois, les écrivains joignent deux points ensemble en un seul coup. Certains auteurs écrivent aussi trois points soit d'un seul coup, soit d'un point sur un coup. Une autre variation moins fréquente des points d'écriture consiste à les écrire en petits cercles. De plus, les points peuvent parfois manquer ou coller le corps principal du caractère. Les scripteurs n'écrivent pas d'autres signes diacritiques mais parfois ils peuvent le faire, en particulier le diacritique Shadda. Dans les textes manuscrits, ces diacritiques peuvent être facilement confondus avec des points. De plus, les signes diacritiques comme Shadda, s'ils sont présents sur un caractère, peuvent être considérés comme une autre variation d'écriture pour cette forme de caractère particulière. Le tableau II.3 illustre les problèmes liés aux points et aux signes diacritiques dans les textes arabes manuscrits.

Tableau II.3 : Les problèmes liés aux point et signes diacritiques [35].

Ecriture manuscrite	Ecriture machine	Observation
	الشرايع	Les points bien écrits
	المنزه	Le point est déplacé vers le caractère voisin
	مارث	Trois points saisi en un seul stroke ou deux.
	الخليج	Un point diacritique omis involontairement
	شوّاط	Shadda écrit de la même manière qu'un point diacritique

II.6.3 Présence de ligatures :

Un autre problème important lié à la reconnaissance du texte arabe manuscrit est la présence de ligatures. La ligature Lam-Alif doit faire l'objet d'une attention particulière, dans la plupart des cas elle doit être traitée comme un caractère spécial au lieu de la traiter comme deux caractères distincts (c'est-à-dire, Lam et alif). De plus, d'autres ligatures optionnelles lorsqu'elles sont présentes dans des textes manuscrits nécessitent idéalement une attention particulière. Mais du fait que ces séquences de caractères ne sont pas toujours écrites sous forme de ligatures (tableau II.3), il est difficile de traiter ces séquences de caractères de manière cohérente. Même si les ligatures facultatives sont traitées comme des caractères spéciaux, le problème est que nous pouvons potentiellement nous retrouver avec un grand nombre de caractères spéciaux qui doivent être manipulés de manière appropriée pour les tâches de reconnaissance de texte.

II.6.4 Présence des espaces :

La manipulation des espaces blancs dans les textes arabes manuscrits n'est pas non plus une tâche triviale. En raison du concept de PAW en arabe, les espaces blancs n'apparaissent pas seulement entre les mots dans les textes Arabes mais aussi dans les mots (tableau II.2).

II.7 Méthodes de représentation des ensembles de données d'écriture en ligne :

Il existe deux méthodes pour la structure des données d'écriture en ligne : le format UNIPEN et les représentations basées sur XML [37]. Le format UNIPEN a été le premier de ces méthodes, mais il a été suivi par les représentations basées sur XML. Dans les deux sections suivantes, nous donnons une brève description de ces méthodes.

II.7.1 Format UNIPEN :

Le format UNIPEN a été développé par I. Guyon et al. en collaboration avec un groupe de travail de 14 experts [37]. C'est un format ASCII conçu spécifiquement pour les données collectées avec tous types de dispositifs tactiles, fournissant des informations discrètes sur la trajectoire du stylet. Le nombre minimum de signaux dans le format est de deux (c'est-à-dire, les coordonnées X et Y), mais plus de signaux sont autorisés (par exemple, l'angle de stylet ou

les informations de pression). Il existe des dispositions dans le format UNIPEN pour l'annotation des données sur les conditions d'enregistrement, les auteurs, la segmentation, la disposition des données et la qualité des données.

Le format UNIPEN implique une succession d'instructions composées de mots-clés suivis d'arguments, où :

- Les mots-clés sont des mots réservés commençant par un point dans la première colonne d'une ligne.
- Les arguments sont des chaînes ou des nombres séparés par des espaces, des tabulations ou des nouvelles lignes. Les arguments relatifs à un mot-clé donné commencent après le mot-clé et se terminent par l'apparition du mot-clé suivant ou la fin du fichier.

Les bases de données écrites au format UNIPEN peuvent éventuellement être organisées dans différents fichiers et répertoires, mais toutes les données peuvent également être concaténées dans un seul fichier.

Le format UNIPEN est considéré comme une séquence de coordonnées de stylo annotée par diverses informations. La trajectoire de stylet est codée sous la forme d'une séquence de composants ".PEN_DOWN" et ".PEN_UP", contenant des coordonnées de stylet (par exemple, XY ou XYT comme déclaré dans .COORD). L'instruction ".DT" permet et précise le temps écoulé entre deux composants. Le contenu des bases de données est divisé en une ou plusieurs données commençant par ".START_SET". Les composants dans les ensembles de données sont numérotés implicitement, à partir de zéro.

La segmentation et l'étiquetage sont fournis par l'instruction ".SEGMENT". Les numéros de composants sont utilisés par ".SEGMENT" pour délimiter les phrases, les mots et les caractères. Une hiérarchie de segmentation est déclarée avec ".HIERARCHY". Comme les composants sont référencés par une combinaison unique de nom d'ensemble et de numéro de commande dans cet ensemble, il est possible de séparer le ".SEGMENT" des données elles-mêmes.

UNIPEN est une base de grande taille, très employée dans l'évaluation des performances des différents systèmes de reconnaissance d'écriture latine en ligne [39].
La base UNIPEN est la base de référence pour l'élaboration et la comparaison de systèmes de reconnaissance d'écriture. Cette base contient des tracés de plus de 200 scripteurs. La difficulté de cette base est due principalement au nombre de scripteurs et donc aux nombreux allographes qu'ils emploient. La version R01/V06 est souvent usitée dans le domaine [40]. Cette base de données présente l'avantage de contenir un grand nombre de données dépaillées dans le tableau suivant :

Tableau II.4 : Taille de sous-corpus de la base UNIPEN

Base	Nombre de classes	Nombre total d'exemples	Nombre d'exemples en apprentissage	Nombre d'exemples en test
Chiffres	10	15635	10423	5212
Minuscules	26	52267	34844	17423
Majuscules	26	26605	17736	8869

Le tableau ci-dessous présente l'ensemble des répertoires fournis dans le CD-ROM de la base de données [32].

Tableau II.5 : Description de la base de données UNIPEN

Répertoires	Description	Unité
1a	Chiffres isolés	Char
1b	Lettres majuscules isolées	Char
1c	Lettres minuscules isolées	Char
1d	Symboles isolés (ponctuations etc.)	Char
2	Caractères isolés, cas mixte	Char
3	Caractères isolés dans le contexte des mots et du texte	Char
4	Mots isolés imprimés, non mixités avec les chiffres et les symboles	Mot
5	Mots isolés imprimés, l'ensemble des caractères complet	Mot
6	Le style mixte des mots (sans les chiffres et les symboles)	Mot
7	Mots isolés, aucun style, l'ensemble des caractères complet	Mot
8	Texte (minimum deux mots)	Texte

II.7.2 Représentations basées sur XML

La représentation UNIPEN utilise des fichiers ASCII pour stocker les données d'écriture manuscrite et les annotations associées. Cela apporte des avantages tels que la simplicité, la facilité de visualisation et d'édition en utilisant un simple éditeur de texte, ainsi que la possibilité de définir des mots-clés supplémentaires pour décrire des attributs supplémentaires des données ou des auteurs. Cependant, UNIPEN souffre de quelques lacunes qui sont :

- UNIPEN n'est pas structuré. Il n'y a aucun moyen d'organiser l'information dans des classes sémantiquement bien classées telles que les données, les définitions d'auteur, les sources d'étiquette ou la hiérarchie d'annotations. Au lieu de cela, UNIPEN fournit un certain nombre de mots-clés qui peuvent être spécifiés dans n'importe quel ordre.
- UNIPEN n'est pas strict. De plus, de nombreux aspects pertinents du processus de collecte de données et des données elles-mêmes sont décrits dans les expressions UNIPEN COMMENT. En outre, les mots-clés tels que SETUP contiennent souvent des informations sur les auteurs, les dispositifs d'enregistrement, les logiciels, la mise en forme des formulaires, etc.
- UNIPEN a un problème de portée. Étant donné que l'ordre des mots-clés saisis n'est pas fixe, la portée des expressions UNIPEN est définie comme suit : Toute coordonnée spécifiée dans UNIPEN est décrite par le contexte des balises UNIPEN précédentes.
- L'effort UNIPEN original était axé sur la reconnaissance du texte latin cursif, et le soutien aux scripts non latins, et pour les modalités telles que les dessins et les mathématiques est limitée.

Les représentations basées sur XML sont les successeurs du format UNIPEN, de telle sorte qu'elles tentent de surmonter les insuffisances d'UNIPEN [42]. XML est un choix naturel pour les représentations car il a l'avantage d'être extensible avec la nature hiérarchique. Dans ce qui

suit, nous allons présenter un bref aperçu sur deux exemples de représentations basées sur XML. Ces exemples sont : hwDataset et UPX.

II.7.2.1 hwDataset

hwDataset a été proposé en tant que représentation XML pour l'annotation de données manuscrites inspirées de la norme UNIPEN. Il utilise le langage InkML (Digital Ink Markup Language) pour la représentation de l'encre numérique annotée [43].

L'encre numérique fait référence à une série de positions de stylet et d'attributs facultatifs (liés à time-stamp, à la pression du stylet, à l'inclinaison du stylet, etc.) capturés à partir d'un dispositif d'entrée de stylet approprié.

Il existe littéralement des milliers de périphériques compatibles avec l'encre numérique (allant des tablettes de numérisation autonomes aux PDA, aux Tablet PC et aux téléphones portables, ainsi qu'aux périphériques propriétaires pour différents marchés) prenant en charge différentes représentations propriétaires de l'encre numérique. Le langage Digital Markup Language (InkML), du World Wide Web Consortium (W3C), est une représentation standard émergente de l'encre numérique [43]. Cette représentation est indépendante de la plateforme et du périphérique. InkML est conçu pour prendre en charge la saisie, le stockage et le traitement de l'écriture manuscrite, des gestes, des croquis, de la musique et d'autres langages de notation dans les applications sensibles à l'encre. InkML fournit également un format commun pour l'échange de données d'encre entre des composants tels que l'écriture manuscrite et les outils de reconnaissance de gestes, les vérificateurs de signature et d'autres modules sensibles à l'encre. Bien qu'InkML offre de nombreuses fonctionnalités appropriées en ce qui concerne la spécification de l'encre numérique, il manque certains aspects plus avancés nécessaires pour les annotations.

Heureusement, InkML fournit des moyens pour les extensions spécifiques aux applications. En vertu d'un langage basé sur XML, il permet aux utilisateurs d'ajouter facilement des informations spécifiques aux fichiers d'encre afin de répondre aux besoins de l'application à portée de main. Dans ce sens, hwDataset peut être considéré comme une extension spécifique à l'application de InkML (voir figure II.9).

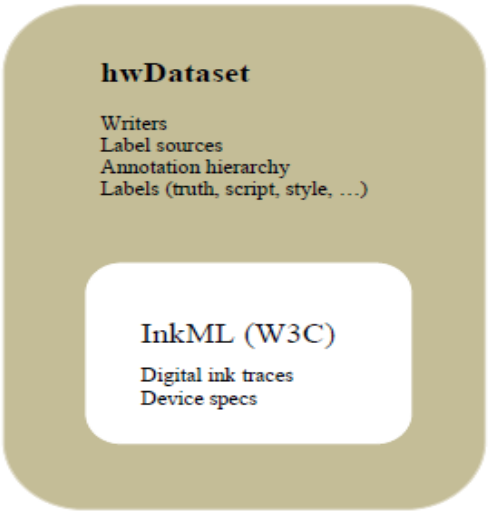


Figure II.9 : La relation conceptuelle entre hwDataset et InkML [42].

II.7.2.2 UPX

La définition UPX est conforme à InkML. Tout comme hwDataset, il a été influencé par UNIPEN [42, 45]. Alors que hwDataset a été créé principalement pour prendre en charge la nouvelle collecte de données, le point de départ de l'effort UPX a été d'évaluer la validité de hwDataset pour le stockage des référentiels de données manuscrites existants.

Un ensemble de données, comprend plusieurs documents InkML et UPX qui s'organisent dans une structure de répertoires, avec peut-être des informations communes stockées dans un document UPX distinct et référencées par d'autres documents UPX. L'élément upx est l'élément racine de chacun des documents UPX [52]. Il contient trois sous-éléments principaux pour spécifier l'encre en ligne annotée. Ces éléments sont :

1. L'élément datasetInfo spécifie les métadonnées liées à l'ensemble de données dans son ensemble. En d'autres termes, il spécifie une caractérisation de haut niveau de l'ensemble de données.

2. Elément datasetDefs, contenant des informations sur les auteurs, les sources d'annotation et les hiérarchies d'annotations référencées dans le document UPX.

3. Elément hwData, contient un étiquetage détaillé des traces d'encre numérique, organisé dans une hiérarchie qu'on peut personnaliser. Gestes et autres notations comme la musique et les mathématiques peuvent également être logés dans la même structure.

Chapitre III

Travaux réalisés sur le script Arabe et les réseaux de neurones

III.1 Introduction :

La reconnaissance d'écriture manuscrite en ligne a été étudiée depuis les années 1960. Jusqu'en 1990, de nombreux travaux de recherche avaient été effectués pour la reconnaissance des caractères anglais pour les modes isolés et cursifs ; et il y avait eu un peu de recherche sur le chinois, l'indien, le coréen et d'autres langues ; mais très peu de travaux se sont penchés sur les caractères Arabes [46].

À la fin des années 1990, A. M. Alimi initié la recherche sur les caractères arabes isolés manuscrits en ligne [47]. On peut citer quelques autres contributions telles que :

- Système IRAC I d'Amin et al [48]. Ce système a été testé par 3 scripteurs sur un ensemble de 73 caractères pour obtenir un taux de reconnaissance de caractères de 95,45%.

- Le système d'El-Wakil et Shoukry [49]. Ce système a été testé par 7 scripteurs sur des ensembles de 60 caractères pour obtenir un taux de reconnaissance de caractères de 93%.

- Le système d'Alimi et Ghorbel [50]. Ce système a été testé par un scripteur sur des ensembles de 28 caractères pour obtenir un taux de reconnaissance de caractères de 96%.

En plus de ces contributions, Alimi a rapporté d'autres recherches de mots arabes cursifs manuscrits en ligne. Ces recherches sont :

1- Les systèmes IRAC II, IRAC III et IV d'Amin et al. [51, 52, 53]. Le système IRAC II a été testé sur 400 mots pour obtenir un taux de reconnaissance de 80%. Le système IRAC III reconnaît les mots sans segmentation, et il a été testé sur 400 mots pour obtenir un taux de reconnaissance de 90%. L'IRAC IV a un taux de reconnaissance de 90% également.
2- Le système de S. Al-Emami et M. Usher [46]. Ce système a été basé sur la méthode structurale pour la reconnaissance de formes.

Deux travaux de recherche, concernant la reconnaissance en ligne de l'écriture arabe, ont été réalisés au cours des sept premières années du troisième millénaire [20, 21]. Le premier est une analyse concernant un système de reconnaissance de caractères arabes en ligne utilisant le réseau Kohonen [20]; tandis que le second est un système de reconnaissance en ligne de l'écriture manuscrite arabe (AraPen) qui a été implémenté par l'utilisation de Java pour les appareils mobiles (J2ME) [21]. Récemment, certaines études ont utilisé les HMM pour la reconnaissance de l'écriture manuscrite en ligne [24].

L'Approche structurelle, le neuro-flou évolutif, le réseau de neurones de Kohonen, la déformation temporelle dynamique suivie de neurones simples, et les HMM sont les approches classificatrices qui ont été utilisées pour ces travaux de recherche. En outre, de nombreux aspects des systèmes de reconnaissance de formes (tels que le volume et la représentation des données d'apprentissage et de test, les caractéristiques extraites et les taux de reconnaissance) ont été impliqués dans ces contributions.

Nous présentons dans ce qui suit un aperçu sur les travaux de référence récents qui ont contribué à la reconnaissance du manuscrit arabe.

III.2 Etat de l'art des travaux récents sur la reconnaissance on ligne du manuscrit arabe :

III.2.1 Réseau de neurone de Kohonen :

N. Mezghani et al [20] ont présenté un système de reconnaissance en ligne des caractères arabes manuscrits dans lesquels le réseau de neurones de Kohonen est utilisé. Le système a été étudié à travers une série de trois expériences. Les résultats de ces expériences montrent que le réseau reconnaît avec succès les caractères clairement et grossièrement écrits avec de bonnes performances.

Le réseau neuronal impliqué dans le système était la mémoire de Kohonen (également appelée carte auto-organisatrice de Kohonen) qui représentait les points d'une source en un nombre inférieur de points dans un espace cible, de sorte que les points voisins auront des valeurs voisines pour préserver l'ordre de la topologie. La mémoire a été organisée comme un tableau bidimensionnel comme le montre la figure III.1. X étant l'entrée de la mémoire. W étant le poids du nœud de mémoire j.

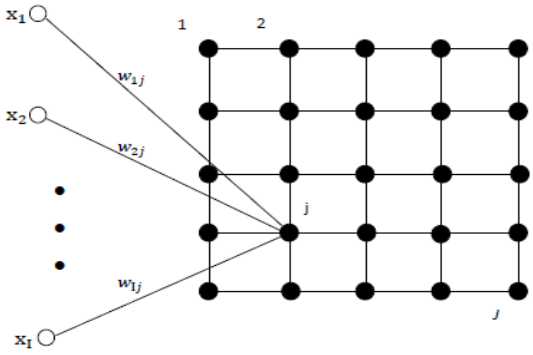


Figure III.1 : Mémoire de Kohonen avec j nœuds.

L'entrée du réseau est un vecteur de caractéristiques extraites de la représentation dynamique d'un caractère manuscrit. Ces caractéristiques sont des coefficients de Fourier pour les coordonnées X (t) et Y (t) des positions du stylo qui représentent l'écriture manuscrite.

Une base de données contenant environ 7400 échantillons de caractères arabes est utilisée pour l'apprentissage et les tests du réseau, de sorte que l'ensemble d'apprentissage contient 5000 échantillons et l'ensemble de test environ 2400 échantillons. La base de données est préparée en utilisant 18 formes pour des caractères arabes isolés qui ont été écrites 24 fois par 17 auteurs (voir la figure III.2).

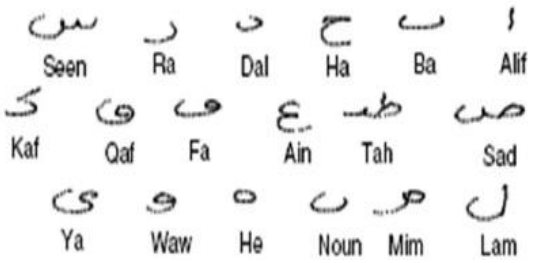


Figure III.2 : les 18 formes de caractères arabes isolés.

L'ensemble de l'alphabet Arabe se compose de 28 caractères isolés. Les 18 formes ont été obtenues par la suppression de toutes les marques diacritiques, qui ne portent pas d'informations importantes sur les formes de caractères. Pour étudier l'effet de différents paramètres sur le taux de reconnaissance du réseau, trois expériences ont été effectuées. Ces expériences sont :

Expérience 1 : étudie l'effet de la dimension du vecteur caractéristique sur le taux de reconnaissance. Cette expérience montre que le meilleur taux de reconnaissance est obtenu avec une dimension du vecteur de caractéristiques égale à 17, ce qui correspond dans les descripteurs de Fourier à n = 5 en omettant les caractéristiques (voir figure III.3).

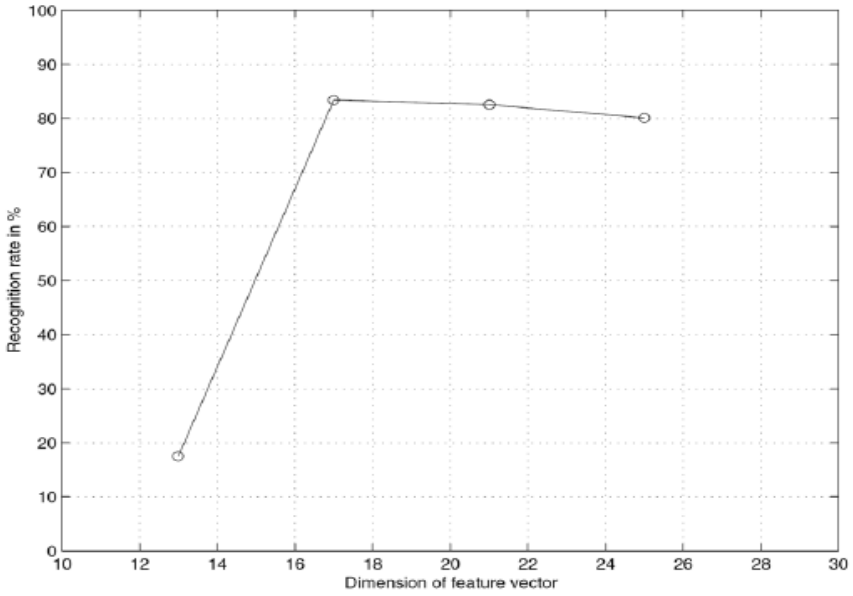


Figure III.3 : Taux de reconnaissance versus dimension du vecteur de caractéristiques [20]

Expérience 2 : étudie l'effet du nombre d'itération de l'algorithme d'apprentissage sur le taux de reconnaissance. Les résultats montrent que le taux de reconnaissance augmente rapidement et atteint un maximum pour rester approximativement constant, où la dimension du vecteur caractéristique est fixée à 17, le nombre de nœuds est fixé à 1600 et 100 itérations sont conservées pour l'algorithme d'apprentissage (voir figure III.4)

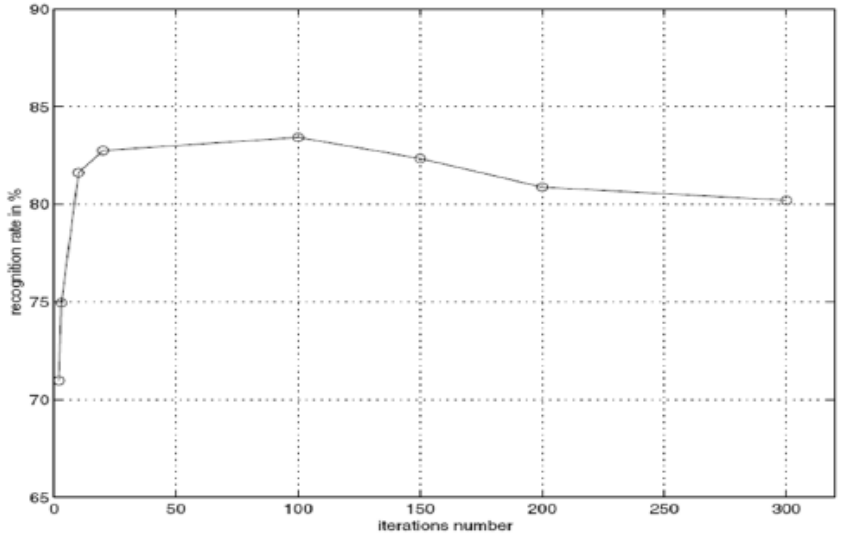


Figure III.4 : Taux de reconnaissance versus nombre d'itérations [20].

Expérience 3 : étudie l'effet du nombre de nœuds dans la mémoire de Kohonen qui se produit sur le taux de reconnaissance. L'expérience montre que le taux de reconnaissance atteint son maximum à 1600 nœuds, où la dimension du vecteur caractéristique est fixée à 17 et le nombre d'itérations à 100 (voir la figure III.5).

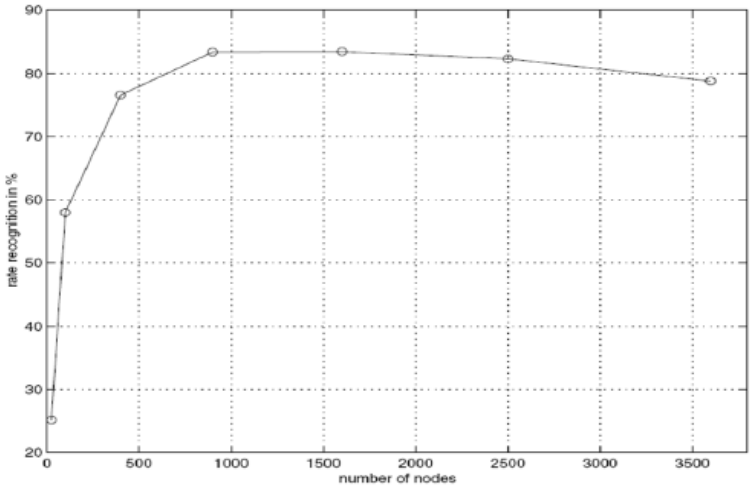


Figure III.5 : Taux de reconnaissance en fonction du nombre de nœuds [20].

III.2.2 Modèle de Markov caché :

Biadsy et al. [23] ont introduit un système de reconnaissance basé sur les HMM. Après prétraitement des séquences de points d'entrée, la détection des strokes retardées est effectuée en fonction de l'emplacement, de la taille des strokes et de leur ordre temporel. L'incorporation du stroke retardé dans le corps de la lettre est effectuée en insérant la séquence de point de coup retardé dans le corps de la lettre par points virtuels nouvellement ajoutés. Le cadre de la reconnaissance utilise des HMM discrets pour représenter chaque forme de lettre.

Pour améliorer la reconnaissance des mots, ces modèles en forme de lettre sont intégrés dans un réseau qui représente un dictionnaire de pseudo-mots. La segmentation des parties de mots en formes de lettres et leur reconnaissance sont effectuées simultanément dans un processus intégré. Le dictionnaire arabe D est subdivisé en un ensemble de sous-dictionnaires {D1, D2, ..., Dn} basé sur le nombre de pseudo- mots dans chaque mot.

Le sous-dictionnaire D_k comprend tous les mots composés de k parties de mots. Nous nous référons au dictionnaire de pseudo-mot $WPD_{k,i}$ comme la liste des pseudo mot situées dans l'index i (commençant à droite dans un mot) des mots dans D_k comme l'exemple montré dans le tableau suivant :

Tableau III.1 : Le système de subdivision du dictionnaire de Biadsy.

D={وسام، هل، معلم، محمود، محمد، فادي، رواية، جامعة، ثقافة، التحدي، انسان}	
Sous-dictionnaires de D	Dictionnaire de pseudo-mots de D_3
D_1={ هل، معلم، محمد }	$WPD_{3,1}$={ ا, فا,و }
D_2={ محمود، جامعة، ثقافة }	$WPD_{3,2}$={ سا، د، لتحد، نسا}
D_3={وسام، فادي، التحدي، انسان}	$WPD_{3,3}$={م،ى،ن}
D_4={رواية }	

Le HMM discret avec le nombre variable d'état a été adopté pour modélisez chaque forme de lettre arabe. Par exemple : 11 états sont assignés à des Sheen, et 5 états à Alef isolé. La forme de lettre est intégrée dans un réseau qui représente le dictionnaire de pseudo-mots $WPD_{k,i}$ (réseau de pseudo-mots).

Les scripteurs sont invités à écrire à l'aide d'une tablette numérique une liste de mots prédéterminés (données d'apprentissage). Ils sont également invités à spécifier manuellement les points de démarcation qui séparent les formes de lettre de sorte que tous les strokes retardés d'une forme de lettre soient horizontales entre les points de démarcation de la lettre.

Quatre scripteurs sont invités à écrire 800 mots sélectionnés. Pour tester, dix scripteurs (les quatre d'apprentissage, en plus de six nouveaux volontaires) sont invités à écrire 280 mots non présents dans les données d'apprentissage. L'ensemble de test comprenait 2358 mots au total (6220 pseudo-mots). Cinq tailles de dictionnaire différentes (5K, 10K, 20K, 30K et 40K mots) sélectionnés à partir de différents sites arabes sont utilisées. Les 280 mots de test sont présents dans toutes les tailles de dictionnaire. The Writer dependent (WD) and writer independent (WI), les taux moyens de reconnaissance des mots et des pseudo-mots sont indiqués dans les tableaux III.2 et III.3. La performance se dégrade quand l'ambiguïté (taille du dictionnaire) augmente.

Tableau III.2 : Les taux moyens de reconnaissance des mots du système de Biadsy

	5K	10K	20K	30K	40K
WD	96.47	95.50	92.86	90.84	89.75
WI	96.28	95.21	92.55	89.68	88.01

Tableau III.3 : Les taux moyens de reconnaissance des pseudo-mots de Biadsy

	5K	10K	20K	30K	40K
WD	98.44	97.94	96.86	95.90	95.44
WI	98.49	97.78	96.54	95.12	94.40

Abdelazeem et al. [54] présentent un système de reconnaissance basé sur le HMM avec un large vocabulaire pour la reconnaissance des noms de personnes arabes. Le système est formé à l'aide de la base de données ADAB. Les strokes retardés sont détectés et supprimés. La base de données de test de 300 noms personnels manuscrits provenant de 10 scripteurs est utilisée pour l'évaluation. La réduction de la taille du lexique est effectuée en fonction du nombre de PAW et des stroke retardé lié à chaque PAW dans le nom du test. La réduction du lexique n'a pas seulement réduit le temps de reconnaissance, mais elle a également un effet remarquable sur le taux de reconnaissance des mots du système. Les résultats sont présentés dans le tableau III.4.

Tableau III.4 : Résultats de recherches d'Abdelazeem et al [54].

Type de test	Taux de reconnaissance de mot		
	Top 1	Top 2	Top 10
Système proposé (Sans réduction du lexique)	55.37%	65.44%	77.52%
Système proposé (Avec réduction du lexique)	92.05%	93.38%	96.03%

En 2011, H. Ahmed et S. A. Azeem ont introduit un système de reconnaissance en ligne de l'écriture manuscrite arabe basé sur HMM [23]. L'apprentissage et les tests du système ont été réalisés grâce à l'utilisation de la base de données ADAB. Le système a été implémenté par le kit d'outils HMM (HTK Engine) pour générer 64 HMM de gauche à droite pour les lettres arabes. La plupart des lettres arabes ont quatre formes différentes, en fonction de leur position dans un mot qui se traduit par plus de 100 modèles HMM. Après avoir supprimé les strokes retardés, le nombre de modèles a diminué à 64 modèles où les lettres similaires ont été regroupées dans une classe ; par exemple, les lettres "ت" ,"ن" ,"ب" et "ث" ont été réduites à une classe, comme le montre la figure suivante :

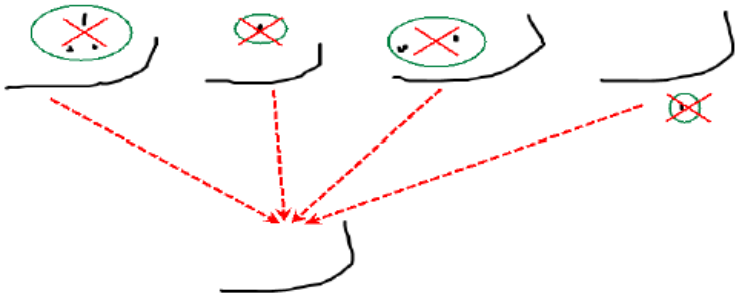


Figure III.6 : Regroupement des lettres arabes dans des classes, après suppression des strokes retardés. [19]

La principale caractéristique de ce système est l'élimination des strokes retardés dans les phases d'apprentissage et de test afin d'éviter la confusion causée par les différences dans l'ordre d'écriture de ces strokes par différents scripteurs. Comme le montrent les tableaux III.5 et III.6, les résultats obtenus par le système ont nettement surpassé les résultats de REGIM-HTK et de REGIM-CV-HTK. REGIM-HTK et REGIM-CV-HTK sont deux systèmes de reconnaissance basés sur HTK qui ont été utilisés dans le concours en ligne de reconnaissance de l'écriture manuscrite arabe 2009 de l'ICDAR [60].

Le problème de la détection des frontières pour les lettres de mots arabes manuscrits a été un problème difficile dans le domaine de la reconnaissance en ligne de l'écriture. Les chercheurs, en particulier ceux qui emploient des HMM, se sont démenés pour atténuer ce problème. On peut citer : G. Al-Habian et K. Assaleh qui utilisent une fenêtre glissante de longueur de 10 échantillons qui va au-dessus des vecteurs caractéristiques des mots [22]. De plus, Biadsy F. et al demandent aux personnes qui saisissent les données d'apprentissage de spécifier manuellement des points de démarcation qui séparent les formes des lettres de sorte que tous les strokes retardés d'une forme de lettre soient horizontalement entre les points de démarcation de la lettre [23].

Tableau III.5 : Taux de reconnaissance du système introduit par H. Ahmed et S. A. Azeem. [22]

Données d'apprentissage	Données de test	Taux de reconnaissance
Ensemble 1 et 2 de la base de données ADAB	Ensemble 3 de la base de données ADAB	95.27%
Ensemble 1 et 3 de la base de données ADAB	Ensemble 2 de la base de données ADAB	89.72%
Ensemble 2 et 3 de la base de données ADAB	Ensemble 1 de la base de données ADAB	92.71%

Tableau III.6 : Les taux de reconnaissance de REGIM-HTK et REGIM-CV-HTK. [19]

Système	Ensemble 1 de la base de données ADAB	Ensemble 2 de la base de données ADAB	Ensemble 3 de la base de données ADAB
REGIM-HTK	57.87%	54.26%	53.75%
REGIM-CV-HTK	28.85%	35.75%	30.60%

III.2.3 L'approche DTW :

Un système de reconnaissance de l'écriture manuscrite arabe en ligne (appelé AraPen) a été développé par B. Alsallakh et H. Safadi en 2006 [21]. Le système a été construit sur la base de DTW qui est un algorithme de correspondance mathématique. La structure du système comprend les cinq étapes des systèmes de reconnaissance : acquisition des données, prétraitement, extraction des caractéristiques, classification et étape de post-traitement. Les entités ponctuelles locales sont utilisées pour représenter l'entrée d'AraPen. Les résultats des tests du système ont montré un taux de reconnaissance élevé pour la reconnaissance de caractères non cursifs.

Le système est déployé avec un ensemble de motifs par défaut, contenant 21 motifs, qui représentent l'ensemble des lettres arabes isolées. Chacun de ces modèles stocke les caractéristiques extraites d'une entrée "standard". L'utilisateur du système peut modifier la manière dont une lettre ou un chiffre est dessiné et les caractéristiques des motifs qui représentent ce caractère sont mises à jour en conséquence. De plus, l'utilisateur peut ajouter de nouvelles caractéristiques à l'ensemble de formes et lui associer certaines sorties. Cette situation édicte l'apprentissage du système. Un exemple pour chaque lettre est suffisant pour entraîner le système et le temps d'apprentissage est négligeable. AraPen peut être entraîné et l'alphabet est extensible, mais il n'est pas indépendant du scripteur.

Pour impliquer la segmentation dans AraPen, le développeur a implémenté deux méthodes de segmentation : la segmentation explicite, la segmentation implicite.

- Segmentation explicite : Habituellement, la connexion entre deux lettres arabes est horizontale. Cela signifie que l'entrée qui forme l'écriture cursive est divisée en segments horizontaux. Cette méthode de segmentation dédie un seuil de longueur pour les segments, de sorte qu'un point de segmentation est placé à la fin du segment horizontal s'il est suffisamment long. Puisque l'écriture est en ligne, le système peut informer l'utilisateur immédiatement lorsqu'un point de segmentation est placé. Cette rétroaction aide l'utilisateur à contrôler la longueur des segments horizontaux et ainsi atteindre un meilleur taux de reconnaissance.
- Segmentation implicite : Tappert a généralisé dans [55] l'approche DTW à la reconnaissance cursive de l'écriture manuscrite en trouvant la meilleure chaîne de motifs qui correspond à l'entrée. Cette méthode place implicitement les points de segmentation dans l'entrée, car elle trouve la partie correspondant à chaque lettre de la meilleure chaîne. Le nombre de chaînes possibles est très prohibitif ; une fois de plus, la programmation dynamique est appliquée pour trouver la meilleure chaîne en un temps raisonnable. La performance de cette approche dépend de la fonction de distance utilisée. Le taux de reconnaissance généré par l'approche était loin de celui du cas non récursif; par conséquent, les développeurs promettent d'étudier certaines techniques qui

peuvent atteindre un taux de reconnaissance satisfaisant en incorporant d'autres caractéristiques dans la fonction de distance et en combinant les résultats d'autres classificateurs.

Le système a été testé en utilisant un petit corpus de données collectées par les développeurs. Le taux de reconnaissance dans le mode non cursif était d'environ 91% avec le jeu de formes par défaut, et de 98% après l'apprentissage du système avec l'écriture manuscrite de l'utilisateur. En mode cursif, en utilisant la fonction de distance simple, la reconnaissance était inférieure à 50%.

III.2.4 Les réseaux de neurones et le SVM :

Abdelazim dans [57] et [56] a étudié deux techniques différentes pour s'attaquer au problème de reconnaissance d'écriture manuscrite en arabe. Les deux sont basées sur le même algorithme de segmentation, un algorithme DCP (Digital Curve Partitioning) basé sur Vertex Finder. Dans cet algorithme, les points de contrôle significatifs sont extraits avec les courbures les plus élevées. Dans [56], les segments de droite sont fuzzifiés en fonction de leurs angles, et un comparateur de logique flou (FLC) est utilisé pour faire correspondre une base de données existante de prototypes. L'étendue du matching représente la base de la décision de segmentation. Les signes diacritiques et les points sont reconnus à l'aide du réseau neuronal de quantification de l'apprentissage du vecteur (LVQ). Dans [57], une technique de reconnaissance / segmentation séquentielle est adoptée. Les segments de droite sont générés à partir du même DCP commun aux deux techniques. Ils sont accumulés séquentiellement et injectés dans un réseau de neurones RBF (Radial Basis Function) connu pour sa puissante classification et ses capacités de rejet plus importantes. La sortie du réseau est utilisée comme mesure pour décider des segments de caractères.

Résultats rapportés, dans [56] : 95% pour un seul auteur, écriture claire «Naskh ». Dans [57], la précision atteint 96% pour 10 écrivains différents, avec aussi l'écriture "Naskh" contrainte.

Alimi dans [47] a mis en place un système complet qui segmentait les lettres selon une compréhension de la façon dont les humains écrivent. C'est l'une des rares recherches dans la reconnaissance d'écriture manuscrite arabe en ligne qui traite le problème global de reconnaissance de script. Étant donné qu'une lettre arabe peut avoir au plus 6 strokes et qu'un stroke est défini comme une fonction asymétrique en forme de cloche de vitesse curviligne avec une vitesse décroissante à la fin du stroke, un système peut automatiquement segmenter une lettre en sous-strokes, qui définissent cette lettre.

Chaque caractère peut être représenté par 6 vecteurs caractéristiques. Si le caractère à moins de 6 strokes, les strokes vides sont remis à zéro. Cet ensemble de vecteurs caractéristiques a été attribué à un réseau de neurones à fonction de base radiale bêta floue pour reconnaître diverses lettres. Les strokes étaient superposés pour donner toutes les combinaisons de strokes possibles en lettres. Ces sorties superposées ont été transmises à un algorithme génétique pour reconnaître de manière robuste les mots. Grâce à une série de mutations et de croisements, les lettres ont été segmentées et reconnues. Le taux de reconnaissance rapporté était de 89% sans information diacritique [58] sur 100 répétitions du mot (عز الدين) écrit par le même scripteur.

Eraqi et al. [59] présentent un système de reconnaissance d'écriture manuscrite arabe en ligne basé sur une nouvelle technique de segmentation de graphèmes en ligne qui dépend de la

direction d'écriture locale. La détection de la ligne de base, la détection des strokes retardés et la construction des caractères à partir des graphèmes de base sont effectuées.

Les caractères sont décidés selon un ensemble de règles qui traitent la séquence des graphèmes reconnus, des types de strokes retardés associés, de la position verticale des storkes retardés associés (au-dessus ou en dessous du trait principal). Les caractères segmentés sont classés à l'aide d'un one over one (OVO) multiclass fuzzy support vector machines (multiclass fuzzy SVM) utilisant le noyau RBF. Le nombre de mots PAW, l'apprentissage de strokes retardés de chaque PAW et le nombre de caractères du mot sont utilisés pour la réduction du lexique. Alors que le lexique est consulté en utilisant l'algorithme de distance d'édition minimum. Les expériences sont effectuées sur la base de données ADAB et les résultats sont présentés dans le tableau III.7.

Tableau III.7 : Les résultats de la recherche d'Eraqi [59]

Ensembles de la base de données ADAB	Taux de reconnaissance de mot		
	Top 1	Top 2	Top 5
Ensemble 1	87%	90.3%	92%
Ensemble 2	86%	89.3%	92.4%
Ensemble 3	87.6%	90.8%	92.5%

III.2.5 Méthodes basées sur les règles et le matching

Elanwar et al. [61] ont développé un autre système de reconnaissance d'écriture manuscrite sans contrainte. Les méthodes basées sur des règles sont utilisées pour effectuer une segmentation et une reconnaissance simultanées de portions de mots à l'aide de la programmation dynamique.

La sortie de ces étapes se présente sous la forme d'une liste classée des décisions de caractère possibles. Dans la phase de prétraitement, l'élimination des points est effectuée. Les motifs sont définis (modélisés) en utilisant trois ensembles d'entités directionnelles représentant les mouvements d'activation / désactivation du stylo. La séquence de code de direction résultante est appelée un modèle de squelette. Pour chaque forme de motif définie rencontrée dans les documents d'apprentissage, tous ses modèles de squelette représentatifs sont regroupés dans un groupe et stockés dans un registre (table de consultation). À l'étape de reconnaissance, le vecteur de caractéristiques de code de direction du mot entier testé est comparé élément par élément par rapport aux structures squelettes dans le registre pour détecter la forme / formes de motif comprenant ce stroke et les emplacements de segmentation. Cette comparaison entre deux squelette est réalisée à l'aide d'une technique de programmation dynamique "Minimum Edit Distance" [62].

Les calculs des coûts d'insertion, de suppression et de substitution sont détaillés dans [61]. La mesure de similarité est représentée par la fonction de coût «Distance», définie être égal à la distance d'édition minimale entre le modèle de squelette de test et le modèle de squelette d'apprentissage multiplié par un facteur représentant la quantité de ressemblance entre eux. Cette quantité de ressemblance est déterminée par chaîne de correspondance pour trouver le nombre de correspondances entre les modèles de squelette du registre et le vecteur de test (équation III.1) :

$$Distance = minimum - edit - distance * \frac{lenght\ skeleton\ pattern}{Number\ of\ matches} \quad \text{(III.1)}$$

Cette comparaison est effectuée par rapport à toutes les formes de modèle d'apprentissage. Ainsi, plus d'une réponse possible peut être trouvée. Les formes de motif probables du premier caractère du trait sont stockées en tant que racines d'arbres individuels. Chaque arbre est complété en comparant la partie non identifiée du vecteur de caractéristiques avec le registre, encore et encore, pour trouver les formes de motif probables (et les points de segmentation suivants) du reste des caractères jusqu'à ce que tout le stroke soit totalement reconnu.

Une liste classée est obtenue. Le rang est calculé en accumulant la valeur de la fonction de coût 'Distance' associée à chaque motif reconnu. La base de données utilisée pour l'apprentissage est composée de 317 mots (1814 caractères), écrits par quatre scripteurs, la base de données de test est composée de 94 mots (435 caractères) écrits par quatre autres scripteurs. Le résumé des résultats est présenté dans le tableau III.8.

Tableau III.8 : Résultats du system Elanwar

	Nombre total	Correctement reconnu	Taux de reconnaissance
Caractères	435	415	95.4%
Strokes	305	279	91.48%
Mots	94	70	74.5%

Dans cette partie du chapitre, nous avons passé en une revue la plupart des études de référence de la reconnaissance en ligne de l'écriture arabe. La revue a été présentée dans un contexte de catégorisation basé sur les approches classificatrices impliquées dans ces études. Ces approches sont : méthode structurale, neuro-flou évolutif, réseau neuronal de Kohonent, DTW suivi d'une approche neuronale simple, et HMMs. De plus, cette revue explore de nombreux aspects des études tels que le volume de données utilisées dans les expériences d'apprentissage et de test, les caractéristiques extraites et les résultats acquis.

III.3 Les réseaux de neurones artificiels :

La classification est une technique d'exploration de données (apprentissage automatique) utilisée pour prédire le groupe pour les instances de données. Pour simplifier les problèmes de prédiction ou de classification, des réseaux de neurones sont introduits. Les réseaux de neurones sont des modèles simplifiés du système de neurones biologiques. C'est un système de traitement distribué massivement parallèle, composé d'éléments de calcul neuronal hautement interconnectés qui ont la capacité d'apprendre, d'acquérir des connaissances et les rendre disponibles pour l'utilisation. Divers mécanismes d'apprentissage existent pour permettre aux NN (neural network) d'acquérir des connaissances. Les architectures NN ont été classées en différents types, en fonction de leurs mécanismes d'apprentissage et d'autres caractéristiques.

Les NN sont des imitations simplifiées du système nerveux central [63], et par conséquent, ont été inspirés par le type de calcul effectué par le cerveau humain. Les constituants structurels d'un cerveau humain appelé les neurones sont les entités qui effectuent des calculs tels que la cognition, l'inférence logique, la reconnaissance de formes et ainsi de suite. Cette imitation du

cerveau humain a été appelé réseaux de neurones artificiels (RNA) ou simplement les réseaux de neurones.

III.3.1 Propriétés des NN :

- Les capacités de mappage, c'est-à-dire, ils peuvent mapper les modèles d'entrée à leurs modèles de sortie associés. [63]
- Les NN apprennent par des échantillons. Ainsi, les architectures NN peuvent être entraînées avec des exemples connus d'un problème avant d'être testées pour leur capacité d'inférence sur des instances inconnues du problème. Ils peuvent donc identifier de nouveaux objets auparavant non appris.
- Les NN ont la capacité de généraliser. Ainsi, ils peuvent prédire les nouveaux résultats des tendances passées.
- Les NN sont des systèmes robustes et tolérants aux pannes. Ils peuvent, par conséquent, rappeler des motifs complets de motifs incomplets, partiels ou bruyants.
- Les NN peuvent traiter l'information en parallèle, à grande vitesse et de manière distribuée.

III.3.2 Caractéristiques des RNA :

III.3.2.1 Neurone artificiel :

Le comportement complexe des neurones biologiques a été simplifié pour créer un modèle mathématique de neurones artificiels, également appelés unités. L'unité reçoit ses entrées via des connexions d'entrée provenant des sorties d'autres unités, appelées activations. Ensuite, il calcule une somme pondérée des entrées, appelée potentiel. Enfin, l'activation de l'unité est calculée à partir du potentiel et envoyée aux autres unités. Les poids des connexions entre les unités sont stockés dans une matrice W, où W_{ij} indique le poids de la connexion de l'unité i à l'unité j. Chaque unité j a un potentiel p_j qui est calculé comme somme pondérée de l'ensemble de ses N unités d'entrée et de biais.

$$p_j = \sum_{i=1}^{N+1} W_{ij} a_i \qquad \text{III.1}$$

Le terme de biais, également appelé unité de seuil, est généralement représenté comme une unité d'entrée supplémentaire dont l'activation est toujours égale à un, donc $a_{N+1} = 1$. La présence d'un biais permet de déplacer la fonction d'activation tout au long de l'axis x en changeant le poids de la connexion de l'unité de seuil (threshold).

L'activation de l'unité a_j est ensuite calculée en transformant son potentiel p_j par une fonction d'activation non linéaire « act ».

$$a_j = act(p_j) \qquad \text{III.2}$$

La fonction d'activation non linéaire couramment utilisée allant de 0 à 1 est la fonction sigmoïde grâce à sa dérivée facilement calculable qui est utilisée par les algorithmes d'apprentissage.

Les réseaux de neurones artificiels se réfèrent à l'interconnexion entre les neurones présents dans les différentes couches d'un système. Chaque système est fondamentalement un système à 3 couches, qui sont la couche d'entrée, la couche cachée et la couche de sortie. La couche

d'entrée à des neurones d'entrée qui transfèrent des données via des synapses à la couche cachée, et de même la couche cachée transfère ces données à la couche de sortie via plus de synapses. Les synapses stockent des valeurs appelées poids qui les aident à manipuler l'entrée et la sortie de différentes couches. Un RNA peut être défini en fonction des trois caractéristiques suivantes :

- L'architecture : Le nombre de couches et le nombre de nœuds dans chacune des couches

- Le mécanisme d'apprentissage qui a été appliqué pour mettre à jour les poids des liaisons.

- Les fonctions d'activation utilisées dans les différentes couches.

III.3.2.2 L'architecture :

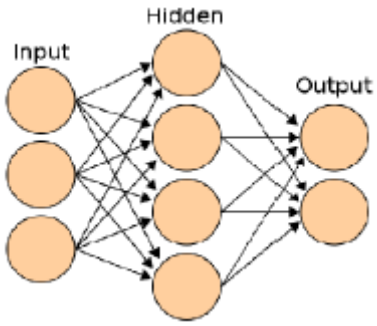


Figure III.7 : Architecture de base d'un RNA

Couche d'entrée : Cette couche est chargée de recevoir des informations (données), des signaux, des mesures de l'environnement externe. Ces entrées (échantillons ou modèles) sont généralement normalisées dans les valeurs limites produites par l'activation des fonctions. Cette normalisation entraîne une meilleure précision numérique pour les opérations mathématiques effectuées par le réseau.

Caches cachés, intermédiaires ou invisibles : Ces couches sont composées de neurones qui sont chargés d'extraire des modèles associés au processus ou au système analysé. Ces couches exécutent la majeure partie du traitement interne à partir d'un réseau.

Couche de sortie : Cette couche est également composée de neurones, et est donc responsable de la production et de la présentation des sorties du réseau final, qui résultent du traitement effectué par les neurones dans les couches précédentes. Voici les architectures de base des RNA :

i- Réseau monocouche :

Ce type de réseau est constitué de deux couches, à savoir la couche d'entrée et la couche de sortie. Les neurones de la couche d'entrée reçoivent les signaux d'entrée et les neurones de la couche de sortie reçoivent les signaux de sortie. Les liens synaptiques portant les poids connectent chaque neurone d'entrée au neurone de sortie, mais pas l'inverse. Un tel réseau est dit feedforward. Malgré les deux couches, le réseau est appelé monocouche puisque c'est la couche de sortie, seule qui effectue le calcul. La couche d'entrée transmet simplement les signaux à la couche de sortie. D'où le nom réseau feedforward monocouche (Figure III.8).

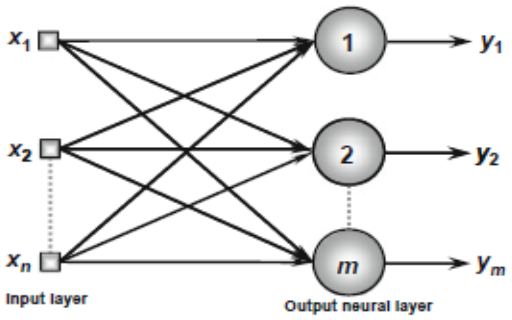


Figure III.8 : Réseau feedforward monocouche

ii- Réseau multi-couches :

Ce réseau est composé de plusieurs couches. Il possède une couche d'entrée et une couche de sortie et possède également une ou plusieurs couches intermédiaires appelées couches cachées comme indiqué dans la figure III.9. Les unités de calcul de la couche cachée sont connues comme les neurones cachés ou les unités cachées. La couche cachée aide à effectuer des calculs intermédiaires utiles avant de diriger l'entrée vers la couche de sortie. Les neurones de la couche d'entrée sont liés aux neurones de la couche cachée et les poids sur ces liens sont appelés poids de couche cachés en entrée. De même, les neurones de la couche cachée sont liés aux neurones de la couche de sortie et les poids correspondants sont appelés des poids de couche de sortie cachée.

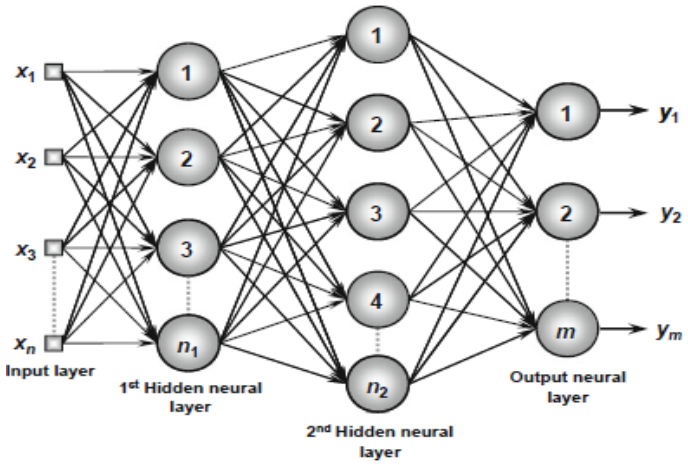


Figure III.9 : Réseau feedforward multicouches.

iii- Réseaux récurrents :

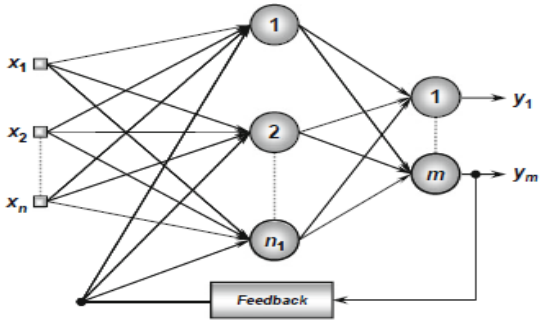


Figure III.10 : Réseau récurrent.

Ces réseaux se différencient des architectures de réseau feedforward car il y a au moins une boucle de rétroaction. Ainsi, dans ces réseaux, il pourrait exister une couche avec des connexions de rétroaction. Il pourrait également y avoir des neurones avec des liens d'auto-retour, c'est-à-dire que la sortie d'un neurone est réinjectée en elle-même en entrée. La figure III.10 illustre un exemple de réseau Perceptron avec retour d'expérience, où l'un de ses signaux de sortie est renvoyé à la couche intermédiaire. Ainsi, en utilisant le processus de rétroaction, les réseaux avec cette architecture produisent des sorties de courant prenant également en compte les valeurs de sortie précédentes.

III.3.2.3 Méthodes d'apprentissage :

L'une des caractéristiques les plus importantes des réseaux de neurones artificiels est leur capacité d'apprentissage à partir de la présentation d'échantillons, qui exprime le comportement du système. Ainsi, une fois que le réseau a appris la relation entre les entrées et les sorties, il peut généraliser les solutions, ce qui signifie que le réseau peut produire une sortie proche de la sortie souhaitée des valeurs d'entrée données. Par conséquent, le processus d'apprentissage d'un réseau de neurones consiste à appliquer les étapes ordonnées nécessaires pour ajuster les poids synaptiques et les seuils de ses neurones, afin de généraliser les solutions produites par ses sorties.

L'ensemble des étapes coordonnées utilisées pour former le réseau est appelé algorithme d'apprentissage. Lors de son exécution, le réseau pourra ainsi extraire des caractéristiques discriminantes sur le système à partir d'échantillons acquis.

Habituellement, l'ensemble complet contenant tous les échantillons disponibles du comportement du système est divisé en deux sous-ensembles, appelés sous-ensemble d'apprentissage et sous-ensemble de test. Le sous-ensemble d'apprentissage, composé de 60 à 90% d'échantillons aléatoires de l'ensemble complet, sera utilisé essentiellement dans le processus d'apprentissage. D'autre part, le sous-ensemble de test, composé de 10 à 40% de l'ensemble complet d'échantillons, sera utilisé pour vérifier si les capacités de généralisation du réseau sont acceptables, permettant ainsi la validation d'une topologie donnée. Néanmoins, lors du dimensionnement de ces sous-ensembles, les caractéristiques statistiques des données doivent également être prises en compte.

Au cours du processus d'apprentissage des réseaux de neurones artificiels, chaque présentation complète de tous les échantillons appartenant à l'ensemble d'apprentissage, afin d'ajuster les poids synaptiques et les seuils, sera appelée itération d'apprentissage.

i- Apprentissage supervisé :

La stratégie d'apprentissage supervisé consiste à disposer des sorties désirées pour un ensemble donné de signaux d'entrée ; en d'autres termes, chaque échantillon d'apprentissage est composé des signaux d'entrée et de leurs sorties correspondantes. Désormais, il nécessite une table avec des données d'entrée / sortie, également appelée table attribut / valeur, qui représente le processus et son comportement. C'est à partir de cette information que les structures neuronales formuleront des «hypothèses» sur le système appris. Dans ce cas, l'application de l'apprentissage supervisé dépend uniquement de la disponibilité de ce tableau attribut / valeur, et il se comporte comme si un «coach» enseignait au réseau quelle est la réponse correcte pour chaque échantillon présenté pour son entrée.

Les poids et seuils synaptiques du réseau sont continuellement ajustés par l'application d'actions comparatives, exécutées par l'algorithme d'apprentissage lui-même, qui supervisent l'écart entre les sorties produites par rapport aux sorties désirées, en utilisant cette différence sur la procédure d'ajustement. Le réseau est considéré comme "formé" lorsque cette divergence se situe dans une plage de valeurs acceptables, en tenant compte des objectifs de généralisation des solutions.

En fait, l'apprentissage supervisé est un cas typique d'inférence inductive pure, où les variables libres du réseau sont ajustées en connaissant a priori les sorties désirées pour le système étudié. Donald Hebb a proposé la première stratégie d'apprentissage supervisé en 1949, inspirée par des observations neurophysiologiques [64].

ii- Apprentissage non-supervisé :

Différent de l'apprentissage supervisé, l'application d'un algorithme basé sur l'apprentissage non supervisé ne nécessite aucune connaissance des sorties. Ainsi, le réseau doit s'organiser lorsqu'il existe des particularités entre les éléments qui composent l'ensemble des échantillons, en identifiant les sous-ensembles (ou groupes) présentant des similitudes. L'algorithme d'apprentissage ajuste les poids synaptiques et les seuils du réseau afin de refléter ces groupes au sein même du réseau. Alternativement, le concepteur de réseau peut spécifier (a priori) la quantité maximale de ces grappes possibles, en utilisant ses connaissances sur le problème.

iii- Apprentissage par renforcement :

Les méthodes basées sur l'apprentissage par renforcement sont considérées comme une variante des techniques d'apprentissage supervisé, car elles analysent en permanence la différence entre la réponse produite par le réseau et la sortie correspondante. Les algorithmes d'apprentissage utilisés ajustent les paramètres neuronaux internes en s'appuyant sur toute l'information qualitative ou quantitative acquise par l'interaction avec le système (environnement) en cours de cartographie, en utilisant cette information pour évaluer les performances d'apprentissage.

Le processus d'apprentissage en réseau se fait généralement par test et erreurs, car la seule réponse disponible pour un intrant donné est de savoir s'il est satisfaisant ou insatisfaisant. Si cela est satisfaisant, les poids synaptiques et les seuils sont graduellement incrémentés pour renforcer (récompenser) cette condition comportementale impliquée avec le système.

Plusieurs algorithmes d'apprentissage utilisés par l'apprentissage par renforcement sont basés sur des méthodes stochastiques qui sélectionnent de manière probabiliste les actions d'ajustement, en considérant un ensemble fini de solutions possibles qui peuvent être récompensées si elles ont des chances de générer des résultats satisfaisants. Pendant le processus d'apprentissage, les probabilités associées à l'ajustement de l'action sont modifiées pour améliorer la performance du réseau [65]. Cette stratégie d'ajustement présente certaines similitudes avec certaines techniques de programmation dynamique [66].

III.3.2.4 Fonctions d'activation

Elles sont utilisées pour limiter la sortie d'un neurone dans un réseau neuronal à une certaine valeur. La sortie peut être dans la plage [-1,1] ou [0,1]. Une fonction d'activation pour un réseau de rétropropagation doit être continue, différentiable et monotone non décroissante [67]. Les différents types de fonctions d'activation sont :

Linear : $f_i(\mathrm{S}) = \mathrm{cS}$

Threshold or Step : $f_i(\mathrm{S}) = \begin{cases} 1 & if \quad S>T \\ 0 & \sin on \end{cases}$

Ramp : $f_i(\mathrm{S}) = \begin{cases} 1 & if \quad S>T \\ S-T_1 \,/\, T_1-T_2 & if \quad T_1<S<T_2 \\ 0 & \sin on \end{cases}$

Sigmoid : $f_i(\mathrm{S}) = \dfrac{1}{(1+\exp^{-cS})}$

Hyperbolic Tangent : $f_i(\mathrm{S}) = \dfrac{(1-\exp^{-S})}{(1+\exp^{-cS})}$

Gaussian : $f_i(\mathrm{S}) = \exp^{\frac{-S^2}{v}}$

III.3.2.5 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 récurrents possèdent un comportement plus complexe, avec des boucles ou connexions bouclées. Dans les sections suivantes nous allons présenter quelques réseaux des plus utilisés dans la littérature.

III.4 MLP (Multi Layer Perceptron) :

Le perceptron multicouche (MLP) est une classe de réseaux feedforward composés de trois couches ou plus (figure III.11). La couche est un groupe d'unités recevant des connexions provenant des mêmes unités. Les unités à l'intérieur d'une couche ne sont pas connectées entre elles. Le MLP se compose de trois types de couches : couche d'entrée, une ou plusieurs couches cachées et couche de sortie. La couche d'entrée est la première couche du réseau et elle ne reçoit aucune connexion d'autres unités, mais conserve le vecteur d'entrée du réseau en tant qu'activation de ses unités. La couche d'entrée est entièrement connectée à la première couche cachée. La couche cachée i est alors entièrement connectée à la couche cachée i + 1. La dernière couche cachée est entièrement connectée à la couche de sortie. L'activation des unités de sortie est considérée comme une sortie du réseau. Celle-ci est calculée dans un processus appelé propagation vers l'avant en trois étapes :

- L'entrée du réseau est copiée sur les activations des unités d'entrée.

- Les couches cachées calculent leurs activations dans l'ordre topologique.

- La couche de sortie calcule son activation et la copie dans la sortie du réseau.

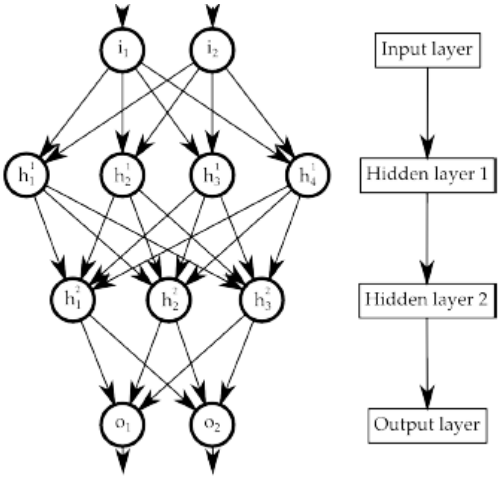


Figure III.11 : MLP composé d'une couche d'entrée avec deux unités, deux couches cachées avec respectivement quatre et trois unités, et une couche de sortie avec deux unités.

Les MLP sont souvent utilisés pour approximer les fonctions inconnues de leurs entrées aux sorties. La capacité du MLP d'approximer n'importe quelle fonction continue avec seulement une seule couche cachée de fonction d'activation de sigmoïde a été prouvée par George Cybenko [68].

III.4.1 Rétropropagation

La rétropropagation, ou propagation vers l'arrière des erreurs, est l'algorithme d'apprentissage supervisé le plus utilisé pour adapter les poids de connexion des ANNs feedforward. Les poids du réseau sont ajustés de manière à minimiser l'erreur quadratique.

$$E = \frac{1}{2}\sum_{i=1}^{N}(\text{target}_{i} - output_i)^2 \qquad \text{III.3}$$

Où la cible (target) indique la sortie désirée fournie par l'enseignant et la sortie (output) est la prédiction du réseau de la sortie pour l'entrée correspondante, les deux de taille N.

En considérant l'erreur E comme une fonction des poids du réseau w, la rétropropagation peut être considérée comme un problème d'optimisation et la méthode de descente de gradient standard peut être appliquée. Le minimum local est approché en changeant les poids selon la direction du gradient d'erreur négatif.

$$-\frac{\partial E}{\partial W} \qquad \text{III.4}$$

En fonction du poids, ΔW_{ij} change proportionnellement à α, ce qui est une valeur positive constante appelée taux d'apprentissage. La fraction de la variation de poids précédente appelée vitesse de mouvement β peut être ajoutée au changement de poids actuel, ce qui accélère souvent le processus d'apprentissage [11].

$$new\Delta W_{ij} = \beta\Delta W_{ij} - \alpha\frac{\partial E}{\partial W_{ij}} \qquad \text{III.5}$$

$$newW_{ij} = W_{ij} + \Delta W_{ij} \qquad \text{III.6}$$

La partie centrale de l'algorithme est de trouver le gradient d'erreur. Soit un MLP avec des couches L dans l'ordre topologique, la première étant l'entrée et la dernière étant la couche de sortie. La couche k a des unités (neurones) et détient une matrice de poids $W^k{}_{ij}$ représentant les poids des connexions de l'unité i dans la couche k-1 à l'unité j dans la couche k. La couche d'entrée n'a aucune connexion entrante. Le calcul peut ensuite être divisé en trois étapes :

1. Propagation vers l'avant. Le vecteur d'entrée est copié sur les activations a_i^1 des unités de couche d'entrée i. Pour chaque couche cachée ou de sortie k dans l'ordre topologique, calculer pour chaque unité i son potentiel (entrée pondérée) p_i^k et son activation a_i^k.

2. Propagation arrière. Calculez Δ_i^L c'est-à-dire la dérivée de l'erreur E activation a_i^L de l'unité de couche de sortie i comme suit :

$$\Delta_i^L = (\text{target}_i - a_i^L)\frac{\partial act(p_i^L)}{\partial p_i^L} \qquad \text{III.7}$$

Pour la couche cachée h dans l'ordre topologique inverse à partir de la dernière couche cachée h= L-1 jusqu'à la première couche d'entrée h = 2 et ses unités i calculent le terme d'erreur comme

$$\Delta_i^h = \sum_{j=1}^{U_{h+1}} \Delta_j^{h+1} W_{ij}^{h+1} \frac{\partial act(p_i^h)}{\partial p_i^h} \qquad \text{III.8}$$

3. Mise à jour des poids : changer les poids dans la couche k selon :

$$new\Delta W_{ij}^k = \beta \Delta W_{ij}^k + \alpha \Delta_i^{k+1} a_j^k \qquad \text{III.9}$$

$$newW_{ij}^k = W_{ij}^k + \Delta W_{ij}^k \qquad \text{III.10}$$

III.5 TDNN (Time Delay Neural Network) :

Dans plusieurs applications le temps est une donnée indissociable des signaux d'entrée. La reconnaissance de l'écriture (et de la parole) est l'un des domaines le plus étudié dans le cadre des réseaux neuronaux où le temps joue un rôle très important. Ce genre de problème nécessite un modèle capable de représenter des relations entre les différents signaux dans le temps.

Les réseaux de neurones, les plus adaptés pour de telles applications sont ceux qui utilisent les connexions à délais du type TDNN introduites par Alexander Waibel et Geoffrey Hinton [70]. Il s'agit, en fait, d'une utilisation particulière de connexions à poids partagés afin de propager l'information du temps à travers le réseau. Il apprend par des exemples et adapte ses poids selon la rétro-propagation du gradient.

III.5.1 L'unité temps-retard :

L'unité de base, utilisée dans de nombreux réseaux neuronaux, calcule la somme pondérée de ses entrées avant de lui appliquer une fonction non linéaire *f* :

$$y_j = f(\sum_{i=1}^{N} W_{ij}.x_i) \qquad \text{III.11}$$

On suppose que l'unité j est connectée aux N unités de la couche précédente.

Dans le TDNN, cette unité de base est modifiée en lui ajoutant une dimension temporelle par introduction de la notion de retard, comme indiqué sur la figure III.12 :

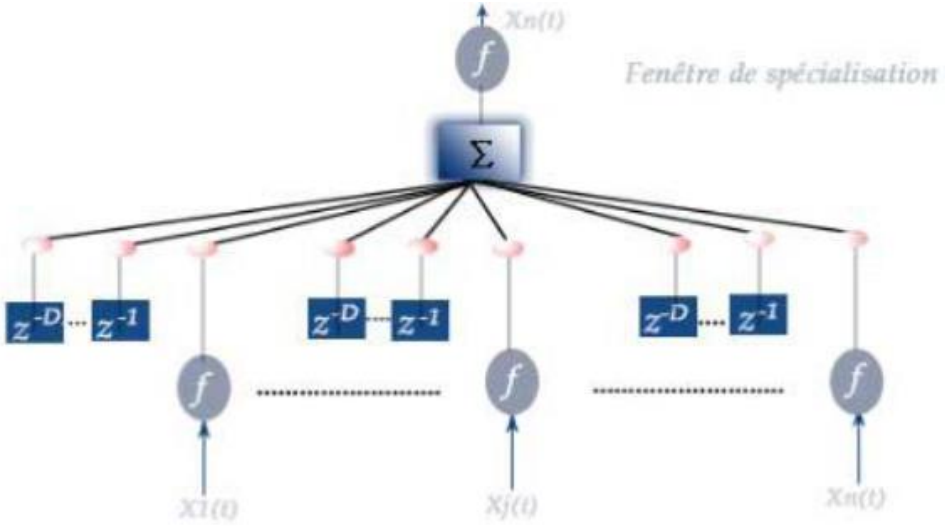


Figure III.12 : Chaque cellule du TDNN comprend une structure d'entrée qui teste les valeurs du vecteur de données courantes $x_i(t)$ directement, ainsi que D vecteurs retardés $x_i(t-D)$. z^{-D} indique la quantité du temps retard dans l'entrée.

Les entrées à l'unité vont être maintenant multipliées par plusieurs poids, un pour chaque retard et un pour l'entrée non retardée. L'équation III.11 devient alors :

$$y_j(t) = f(\sum_{i=1}^{N}\sum_{d=0}^{D} W_{ij}^{d}.x_i(\mathrm{t-d})) \qquad \text{III.12}$$

L'équation précédente montre une manière de déplier le temps dans l'espace, cela permet de traiter statiquement l'architecture dynamique des réseaux TDNN avec deux contraintes sur la matrice des poids :

- Elle n'est pas entièrement connectée : $y_j(t)$ ne dépend pas des unités avant *t- D* ou après *t*.

- Certains poids sont partagés, par exemple, les unités calculant $y_j(t)$ et $y_j(t+1)$ partagent entièrement leur poids.

III.5.2 Notion des poids partagés :

Waibel a proposé un réseau MLP avec des poids partagés afin de mieux cerner l'aspect dynamique du signal. Cela revient à contraindre le réseau à avoir des poids similaires pour plusieurs unités.

En effet, le comportement présumé du cerveau humain utilise le concept des poids partagés, car dans le cerveau humain des neurones détectent certains traits dans des petites régions de la rétine essentiellement de la même manière dans toutes ces régions, ce qui fait qu'on a plusieurs neurones qui calculent la même fonction sur des entrées différentes.

Conséquence directe de cette contrainte, le réseau est de moindre dimension sans pour autant perdre de son pouvoir discriminant, et même qu'il en résulte souvent un pouvoir de généralisation plus accru par la suite, car le niveau d'abstraction est plus élevé. Ces réseaux permettent de mieux prendre en compte l'aspect temporel du signal.

III.5.3 Architecture d'un TDNN :

La première couche du réseau acquiert les caractéristiques du signal. Une ou plusieurs couches cachées du réseau de neurones (phase d'extraction) transforment une séquence de vecteurs caractéristiques en une autre séquence de vecteurs caractéristiques d'ordre supérieur. Le champ de vision du neurone est restreint à une fenêtre temporelle limitée. Avec la contrainte des poids partagés, le même neurone est dupliqué dans la direction temps (soit la même matrice des poids dupliquée) pour détecter la présence ou l'absence de la même caractéristique à différentes places le long de la trajectoire du signal. En utilisant plusieurs neurones (nombre de pas) à chaque position temporelle, le réseau de neurones effectue la détection de caractéristiques différentes : les sorties des différents neurones produisent un nouveau vecteur caractéristique pour la couche supérieure. La composante temporelle de la représentation du signal d'origine est éliminée au fur et à mesure. Pour compenser cette perte d'informations, le nombre de caractéristiques est multiplié. Nous avons donc une architecture qui convertit progressivement les informations temporelles en informations caractéristiques (figure III.13).

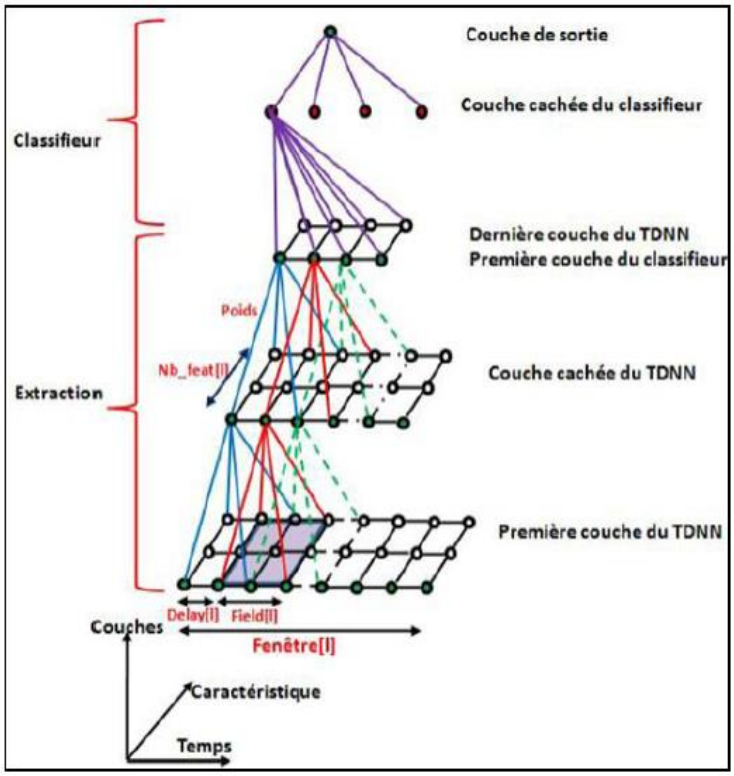


Figure III.13 : Architecture d'un TDNN

Cette architecture comprend deux parties ; une partie extraction permettant de transformer progressivement les caractéristiques en grandeurs de plus en plus signifiantes vis-à-vis du problème et la seconde correspond à un MLP, il reçoit en entrée l'ensemble des sorties de la partie TDNN.

Ces deux blocs sont complètement paramétrables, ils sont décrits par les grandeurs présentées ci-après. La partie extraction se caractérise par :

- le nombre de couches, (chaque couche comprend deux directions : une direction temporelle et une direction caractéristique), nb_layer

- le nombre de neurones de chaque couche selon la direction temporelle, window_T.

- le nombre de neurones de chaque couche selon la direction caractéristique, nb_feat.

- la taille de la fenêtre temporelle vue par chaque couche (sauf celle d'entrée) soit le nombre de neurones de la couche vue par un neurone de la couche l +1, field_T

- le délai temporel (nombre de neurones) entre chaque fenêtre, delay.

La partie classification se caractérise par le nombre de couches et le nombre de neurones de chaque couche.

III.6 Les réseaux RBFs (Radial Basis Function) :

Les réseaux à fonction de base radiale (RBF) sont des réseaux feed-forward formés à l'aide d'un algorithme d'apprentissage supervisé. Ils sont généralement configurés avec une seule couche cachée de neurones dont la fonction d'activation est sélectionnée parmi une classe de fonctions appelées fonctions de base. Bien que similaires aux réseaux back-propagation, les réseaux de fonctions de base radiale présentent plusieurs avantages. Ils apprennent généralement beaucoup plus vite que les réseaux de back-propagation. Ils sont moins sensibles aux problèmes d'entrées non stationnaires en raison du comportement des unités cachées de la fonction de base radiale.

Popularisés par Moody et Darken [71], les réseaux RBF se sont avérés être une architecture de réseau neuronal utile. La différence majeure entre les réseaux RBF et les réseaux de propagation arrière (c'est-à-dire, le perceptron multi-couche MLP) est le comportement de la couche cachée unique. Plutôt que d'utiliser la fonction d'activation sigmoïdale, les unités cachées dans les réseaux RBF utilisent une fonction gaussienne ou une autre fonction noyau de base. Chaque unité cachée agit comme un processeur réglé localement qui calcule un score pour la correspondance entre le vecteur d'entrée et ses poids ou centres de connexion. En effet, les unités de base sont des détecteurs de motifs hautement spécialisés. Les poids reliant les unités de base aux sorties sont utilisés pour prendre des combinaisons linéaires des unités cachées pour produire la classification ou la sortie finale.

III.6.1 Architecture des réseaux RBF :

Les fonctions de base radiale sont d'abord introduites dans la solution des vrais problèmes d'interpolation multi variable. Broomhead et Lowe [72], et Moody et Darken [71] ont été les premiers à exploiter l'utilisation de fonctions de base radiales dans la conception de réseaux de neurones. La structure d'un réseau RBF dans sa forme la plus basique implique trois couches entièrement différentes (Figure III.14).

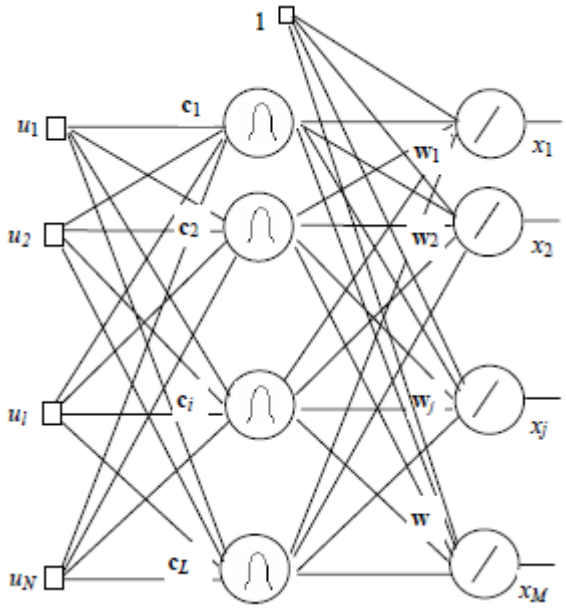


Figure III.14 : Structure d'un RBF standard.

La couche d'entrée est constituée de nœuds sources dont le nombre est égal à la dimension p du vecteur d'entrée u.

III.6.1.1 Couche cachée :

La deuxième couche est la couche cachée qui est composée d'unités non linéaires qui sont connectées directement à tous les nœuds de la couche d'entrée. Elle est de dimension assez élevée.

Chaque unité cachée prend son entrée de tous les nœuds composants la couche d'entrée. Comme mentionné ci-dessus, les unités cachées contiennent une fonction de base, qui possède un centre et une largeur comme paramètres. Le centre de la fonction de base pour un nœud i à la couche cachée est un vecteur c_i dont la taille est le vecteur d'entrée u et il y a normalement un centre différent pour chaque unité dans le réseau.

Premièrement, la distance radiale d_i, entre le vecteur d'entrée u et le centre de la fonction de base c_i est calculée pour chaque unité i dans la couche cachée en utilisant la distance Euclidienne comme suit :

$$\mathrm{d}_i = \|u - c_i\| \qquad \text{III.13}$$

La sortie h_i de chaque unité cachée i est ensuite calculée en appliquant la fonction de base G à cette distance :

$$h_i = G(d_i, \sigma_i) \qquad \text{III.14}$$

Comme le montre la figure III.15, la fonction de base est une courbe (typiquement une fonction gaussienne, la largeur correspondant à la variance, σ_i) qui a un pic à une distance nulle et qui diminue à mesure que la distance du centre augmente.

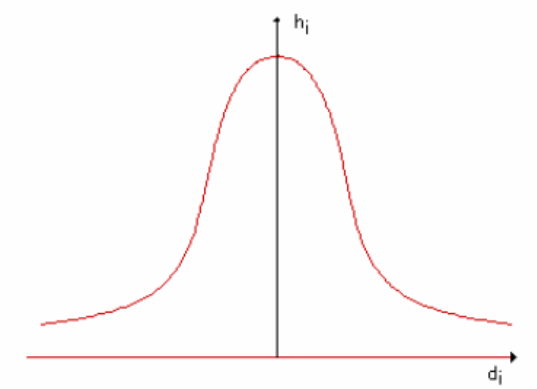


Figure III.15 : La zone de réponse d'un nœud caché RBF autour de son centre en fonction de la distance de ce centre.

Pour un espace d'entrée u∈R2, c'est-à-dire M = 2, cela correspond au centre gaussien bidimensionnel centré sur c_i sur l'espace d'entrée, où également $c_i \in R2$, comme le montre la figure III.16.

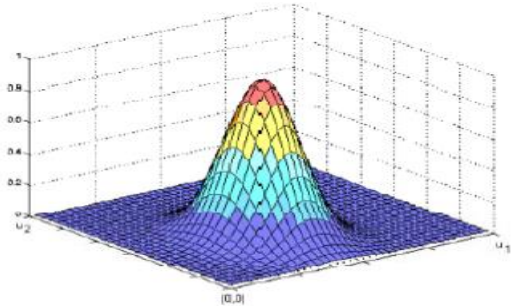


Figure III.16 : Réponse d'une unité cachée sur l'espace d'entrée pour u∈R2

III.6.1.2 Couche de sortie :

La transformation de l'espace d'entrée vers l'espace unité caché est non linéaire, alors que la transformation vers l'espace unité caché vers l'espace de sortie est linéaire. La sortie j est calculée comme :

$$x_j = f_j(u) = W_{0j} + \sum_{i=1}^{L} W_{ij} h_i \qquad \text{Ou : } j = 1,2,...,M \qquad \text{III.15}$$

En résumé, le modèle mathématique de l'équation (III.15) du réseau RBF peut être exprimé comme suit :

$$x = f(u), \qquad f : R^N \to R^M$$

$$x_j = f_j(u) = W_{0j} + \sum_{i=1}^{L} W_{ij} G(\|u - c_i\|) \qquad j = 1,2,...,M$$

III.6.2 Apprentissage d'un réseau RBF :

L'apprentissage d'un réseau RBF peut être formulé comme un problème d'optimisation non-linéaire non-contrainte donné ci-dessous :

Étant donné les modèles d'apprentissage d'entrée sortie (u^k , y^k), $k = 1, 2, .., K$, choisissez w_i, j, c_i, $i = 1, 2, ..., L$, $j = 1, 2, ..., M$ de sorte à minimiser :

$$J(w, c) = \sum_{k=1}^{K} \left\| y^k - f(u^k) \right\|^2 \qquad \text{III.16}$$

Notez que le problème d'apprentissage devient quadratique une fois les centres de fonction de base radiale connus.

III.6.2.1 Ajuster les largeurs

Dans sa forme la plus simple, toutes les unités cachées dans le réseau RBF ont la même largeur ou le même degré de sensibilité aux entrées. Cependant, dans certaines parties de l'espace d'entrée où il y a peu de motifs, il est parfois souhaitable d'avoir des unités cachées avec une large zone de réception. De même, dans des parties de l'espace d'entrée qui sont encombrées, il peut être souhaitable d'avoir des processeurs très fortement accordés avec des champs de réception étroits. Le calcul de ces largeurs individuelles augmente les performances du réseau RBF au détriment d'un processus d'apprentissage plus compliqué.

III.6.2.2 Ajuster les centres

Rappelez-vous que dans un réseau de back-propagation, tous les poids de toutes les couches sont ajustés en même temps. Cependant dans les réseaux de fonctions de base radiale, les poids dans les unités de base de couche cachée sont habituellement établis avant que la deuxième couche de poids soit ajustée. Lorsque l'entrée s'éloigne des poids de connexion, la valeur d'activation diminue. Ce comportement conduit à l'utilisation du terme "centre" pour les poids de première couche. Ces poids centraux peuvent être calculés en utilisant des cartes de Kohonen, des méthodes statistiques telles que les K-Means ou d'autres moyens. Dans tous les cas, ils sont ensuite utilisés pour définir les zones de sensibilité des unités cachées du réseau RBF, qui restent alors fixes.

III.6.2.3 Ajuster les poids

Une fois les poids de couche cachés définis, une seconde phase d'apprentissage est utilisée pour ajuster les poids de sortie. Ce processus utilise généralement l'algorithme de descente le plus raide standard. Notez que le problème d'apprentissage devient quadratique une fois que les centres de fonction de base radiale sont connus.

III.7 Les PNN (Probabilistic neural network)

Le réseau neuronal probabiliste (PNN) a été proposé pour la première fois par Donald Specht en 1990 [73]. Il s'agit d'un réseau de neurones artificiels pour le calcul non linéaire qui se rapproche des limites de décision optimales de Bayes. Ceci est fait en estimant la fonction de densité de probabilité de l'ensemble de données d'apprentissage en utilisant l'estimateur non paramétrique de Parzen. Les stratégies bayésiennes sont des stratégies de décision qui minimisent le risque attendu d'une classification. La théorie de la décision bayésienne est à la base de nombreux schémas d'apprentissage importants tels que le classificateur naïf de Bayes, les réseaux bayésiens et l'algorithme EM (Espérance-Maximisation). Un PNN a la capacité de s'entraîner sur des ensembles de données clairsemés. De plus, il est capable de classer les données dans des catégories de sortie spécifiques [74, 75]. L'utilisation du PNN pour la

classification présente un certain nombre d'avantages. Par exemple, le temps de calcul de PNN est plus rapide, il est aussi robuste au bruit. De plus, le mode d'entraînement du PNN est simple et instantané [76,77].

Dans un contexte de classification des formes, chaque vecteur observé x (x est vecteur de dimension d) est placé dans une des classes de cluster C_i , $i = 1, 2, ..., \mathrm{m}$ prédéfini ; où *m* est le nombre de classes possibles, dans lequel *x* peut appartenir. L'efficacité du classificateur est limitée par la dimension du vecteur x d'entrée et le nombre de classes possibles m. Le classificateur de Bayes met en œuvre le modèle de la règle de Bayes de probabilité conditionnelle que la probabilité $p(C_i/x)$ pour x d'appartenir à la classe C_i . Cette probabilité est donnée par :

$$p(C_i/x) = \frac{p(x/C_i)\, p(\mathrm{C}_i)}{\sum_{j=1}^{m} p(x/C_j)\, p(\mathrm{C}_j)} \qquad \text{III.17}$$

Où $p(C_i/x)$ est la fonction de densité de probabilité conditionnelle de x sachant C_i et $p(\mathrm{C}_j)$ est la probabilité de choisir un échantillon de la classe C_j . Un vecteur en entrée *x* est classifié comme appartenant à C_i si :

$$p(C_i/x) > p(C_j/x); \qquad \forall\, \mathrm{j} = 1, 2, ..., \mathrm{m}; \qquad \mathrm{j} \neq \mathrm{i}$$

L'estimation de ces probabilités à partir de l'ensemble d'apprentissage est réalisée en utilisant la technique de fenêtrage de Parzen pour l'estimation des fonctions de densités [78]. L'estimateur utilisé pour les réseaux PNN est :

$$\mathrm{f(x)} = \frac{1}{2\pi^{p/2}\sigma^{p}} \frac{1}{m} \sum_{i=1}^{m} \exp\left[-\frac{(X - X_{ai})^{t}(X - X_{ai})}{2\sigma^{2}} \right] \qquad \text{III.18}$$

Où X_{ai} est le i ème échantillon appartenant à la classe C_a et σ est un paramètre de lissage.

III.8 Les GRNN (Generalized Regression Neural Network)

Le réseau de neurones de régression générale (GRNN) est l'un des réseaux de neurones les plus populaires. C'est un réseau à anticipation pour les données supervisées. Il utilise des fonctions de régression non linéaires pour l'approximation. GRNN utilise le mappage direct pour lier la couche d'entrée à la couche cachée. Le réseau utilise aussi le facteur de lissage comme paramètre en phase d'apprentissage. Le facteur de lissage unique est sélectionné pour optimiser la fonction de transfert pour tous les nœuds. Pour réduire le temps de calcul, GRNN effectue un apprentissage en un passage à travers le réseau [89]. Ce réseau ne nécessite pas une procédure d'entraînement itératif qui est nécessaire dans la méthode de rétro-propagation. Ils utilisaient quatre couches : la couche d'entrée, la couche de motif (cachée), la couche de sommation et la couche de sortie comme le montre la figure III.17 :

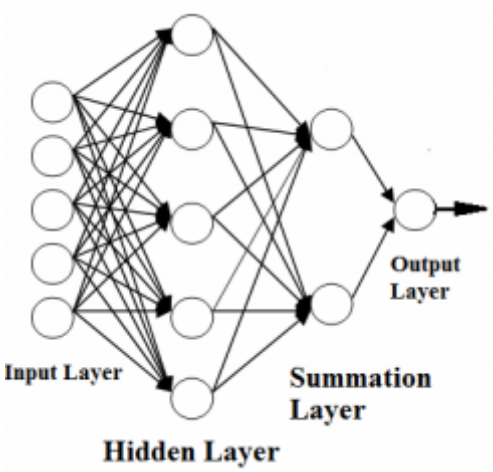


Figure III.17 : Structure d'un GRNN

III.9 Conclusion

Dans ce chapitre, nous avons effectué un état de l'art des travaux réalisés dans la reconnaissance du script arabe en ligne. Cette revue a mis en avant les résultats plus ou moins satisfaisants des différentes méthodes explorées. Par la suite, et étant donné notre choix de classification, nous avons développé les réseaux de neurones peu-profonds, leurs fondements théoriques, leurs architectures et les paramètres influant sur leur capacité à classifier un signal d'entrée. Dans la suite, nous nous pencherons sur les réseaux de neurones à apprentissage profond et l'avantage qu'ils procurent par rapport aux réseaux précédemment cités.

Chapitre IV

Apprentissage profond et DBN

IV.1 Le concept du Deep Learning :

L'un des aspects clés de la plupart des méthodes d'apprentissage automatique est la façon dont les données sont représentées, c'est-à-dire les caractéristiques à utiliser. Si les caractéristiques utilisées sont mal choisies, la méthode échouera quelle que soit sa qualité. De plus, cette sélection affecte la connaissance avec laquelle la méthode peut fonctionner : si vous avez formé votre algorithme d'analyse de marché avec des valeurs numériques, il ne pourra pas avoir de sens à partir d'un rapport écrit, peu importe sa qualité. Par conséquent, il n'est pas surprenant qu'il y ait eu un intérêt historique à trouver les caractéristiques appropriées. Cela devient particulièrement pertinent dans le cas de problèmes de vision par ordinateur. La raison en est que, face à une image, il y a généralement trop de fonctionnalités - une simple image RVB de 640 × 480 a près de 1 million de pixels - et la plupart d'entre elles ne sont pas pertinentes. Pour cette raison, il est important de trouver un moyen de condenser cette information de manière plus compacte.

Il y a deux façons principales d'obtenir des caractéristiques, en les choisissant manuellement - comme les valeurs physiologiques dans les applications médicales - ou en les générant automatiquement, une approche connue sous le nom d'apprentissage de la représentation. Ce dernier s'est avéré être plus efficace dans des problèmes tels que la vision par ordinateur, car il est très difficile pour nous, humains, de savoir ce qui distingue une image. Au lieu de cela, dans de nombreux cas, les machines ont été en mesure de déterminer quelles caractéristiques étaient pertinentes pour elles, ce qui a donné lieu à des résultats encourageants dans la littérature. Le cas le plus démonstratif de l'apprentissage de la représentation, sont les auto-encodeurs. Ils effectuent un processus en deux étapes, d'abord ils codent les informations qu'ils reçoivent dans une représentation compressée, puis ils essaient de décoder ou de reconstruire l'entrée originale à partir de cette représentation réduite.

En ce qui concerne les caractéristiques extraites, les gens peuvent avoir des idées claires sur ce qui rend un objet, comme une voiture, reconnaissable. Ayant 4 roues, des portes dans le latéral, un verre à l'avant, etc. Cependant, ce sont des caractéristiques de haut niveau, qui ne sont pas faciles à trouver pour une machine dans une image. De plus, chaque type d'objet dans le monde a ses caractéristiques particulières, habituellement avec une grande variabilité intra-classe. Pour cette raison, le développement d'une application générale de reconnaissance d'objets serait impossible, car nous aurions besoin de fonctionnalités sélectionnées manuellement pour chacune d'entre elles. Par conséquent, si les machines sont capables de déterminer par elles-mêmes ce qui est représentatif d'un objet, elles auront le potentiel d'apprendre à représenter n'importe quel objet avec lequel elles sont entraînées.

Cependant, il existe une difficulté supplémentaire pour ce genre de problèmes, c'est-à-dire la variabilité en fonction des conditions de chaque image. Nous n'avons pas seulement à faire face à la variabilité intra-classe, mais aussi à la même variabilité d'objet. En effet, la même voiture peut être représentée de façon presque infinie, en fonction de la position de la voiture, des conditions de luminosité, de la qualité de l'image, etc. Les humains sont capables de se

débarrasser de cette variation en extrayant ce que nous pourrions considérer comme des traits abstraits. Ces caractéristiques peuvent inclure celles que nous avons mentionnées précédemment, comme le nombre de roues, mais aussi d'autres dont nous ne sommes pas au courant, comme le fait qu'elles sont généralement sur une route ou que leurs roues doivent être en contact avec le sol. Afin de développer une méthode d'apprentissage de la représentation réussie, il devrait être capable d'extraire ce type de fonctionnalités de haut niveau, indépendamment de leur variation. Le problème est que ce processus peut être extrêmement difficile à développer dans une machine, ce qui peut conduire à penser que cela n'a aucun sens de faire l'effort de le faire. C'est précisément là que le Deep Learning s'est révélé extrêmement utile.

La caractéristique principale du Deep Learning est qu'il est capable de faire des abstractions, en construisant des concepts complexes à partir de concepts plus simples. Étant donné une image, il est capable d'apprendre des concepts tels que les voitures, les chats ou les humains en combinant des ensembles de caractéristiques de base, tels que des coins ou des bords. Ce processus est fait à travers des "couches" successives qui augmentent la complexité des concepts appris. L'idée de profondeur dans le Deep Learning vient précisément de ces niveaux d'abstraction.

Chaque couche reçoit en entrée la sortie du précédent et l'utilise pour apprendre des fonctionnalités de plus haut niveau, comme le montre la Figure IV.1. Fait intéressant, dans certains cas, le propre réseau est celui qui utilise ces fonctionnalités pour produire une sortie, et parfois il les génère simplement pour d'autres méthodes à utiliser.

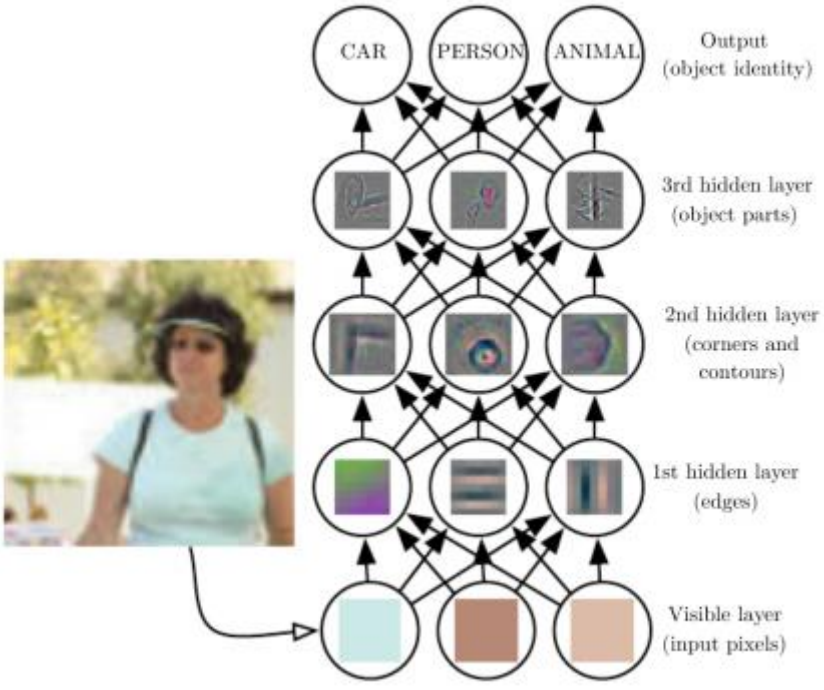


Figure IV.1 : Deep Neural Network (Réseau neuronal profond) : Plus la couche est haute, plus les caractéristiques des concepts sont abstraites, jusqu'à ce qu'elles soient capables d'apprendre à reconnaître des concepts complexes tels que les voitures ou les personnes. [80]

IV.2 Deep Neural networks :

Bien qu'il existe différentes approches de Deep Learning, telles que Deep Kernel Methods [81], celle qui a été la plus utilisée, de loin, utilise des réseaux neuronaux, et elle est connue sous le nom de Deep Neural Networks (DNN).

Plus concrètement, ils peuvent être grossièrement compris comme ANN avec de nombreuses couches cachées. L'une des approches ANN les plus couramment utilisées pour le DNN est le MLP. Comme déjà expliqué dans le chapitre précédent, les réseaux de neurones sont composés de couches de neurones interconnectés. En principe, il n'y a pas de limite concernant le nombre de couches ou de neurones par couche, mais, en pratique, il a été pratiquement impossible d'entraîner avec succès plus d'une poignée de couches cachées. Comme déjà expliqué, le nombre de poids dans un réseau peut facilement atteindre les milliers, voire des millions dans les plus grands, ce qui signifie un grand nombre de paramètres à apprendre. Cela nécessite à la fois des temps de calcul extrêmement importants et des données à fournir aux étapes d'apprentissage. Il y a eu des tentatives de le faire depuis des décennies, mais ce n'est qu'à la fin des années 2000 que les moyens de le faire ont été disponibles.

Il y a plusieurs facteurs qui ont permis de former ce genre de réseaux. Le premier de tous est l'augmentation du pouvoir de calcul que les ordinateurs ont connu. Non seulement les ordinateurs d'aujourd'hui sont beaucoup plus puissants que ceux d'il y a une dizaine d'années, mais l'apparition de cartes graphiques a grandement accéléré la vitesse de ces méthodes. Les unités de traitement graphique, ou GPU, ont d'abord été conçues pour permettre aux ordinateurs d'exécuter des programmes graphiques exigeants, principalement des jeux vidéo. Pour ce faire, ils excellaient à effectuer rapidement de grandes quantités d'opérations simples. Voyant cela, il est devenu évident qu'ils pourraient être utilisés pour d'autres types d'applications avec des besoins similaires, tels que, précisément, DNN. De nos jours, la plupart des chercheurs et des utilisateurs de DNN utilisent des GPU pour exécuter les leurs, car ils peuvent réduire le temps de fonctionnement en ordres de grandeur. Cela a popularisé l'utilisation de DNN, car il n'est plus nécessaire d'utiliser des superordinateurs coûteux pour former les réseaux dans un délai raisonnable.

L'autre facteur qui a contribué à la formation de DNN a été la nouvelle culture axée sur les données qui a vu le jour au cours des années 2000. Comme l'exploration de données et l'apprentissage automatique ont permis d'analyser rapidement et de manière fiable toutes sortes de données, de nombreuses entités ont voulu s'en servir. Pour ce faire, ils ont commencé à rassembler de grandes quantités de données et à les convertir en ensembles de données utilisables. Ceux-ci couvrent un large éventail de disciplines, telles que la santé, l'économie, le comportement social, etc.

Bien que certains de ces ensembles de données fussent d'usage privé, beaucoup d'entre eux ont été rendus publics. Il est devenu un cercle d'auto-alimentation, parce que plus de données étaient disponibles pour étudier, meilleures étaient les techniques d'analyse, ce qui a attiré plus de personnes dans l'utilisation. Cela a permis la création de grands ensembles de données, avec des millions d'instances, qui pourraient être utilisés pour former ANN avec un grand nombre de paramètres à apprendre, sans risque de sur-apprentissage (overfitting).

Le dernier facteur qui a permis la popularisation de DNN a été l'apparition de nouvelles méthodes pour les former. Bien que les deux faits précédents aient aidé, sans algorithmes

d'apprentissage avancés, nous n'aurions pas pu les utiliser. On considère généralement que c'est Hinton qui a établi la base de l'apprentissage profond [82]. Dans cette publication, il a proposé une manière de former les réseaux neuronaux profonds de manière rapide et réussie. Ceci a été réalisé en traitant chaque couche comme une machine de Botzmann restreinte, et en les entraînant une par une, pré-entraînant ainsi les poids du réseau. Après cela, le réseau a été affiné dans son ensemble. Cette percée a permis de former plusieurs réseaux profonds en couches qui n'auraient pas pu être formés auparavant, car ils auraient fini par être surentrainé. Après cela, de nombreuses autres méthodes ont été développées afin de former des réseaux profonds, tels que les couches ReLU (Unité linéaire rectifiée) ou la régularisation d'abandon (dropout regularization).

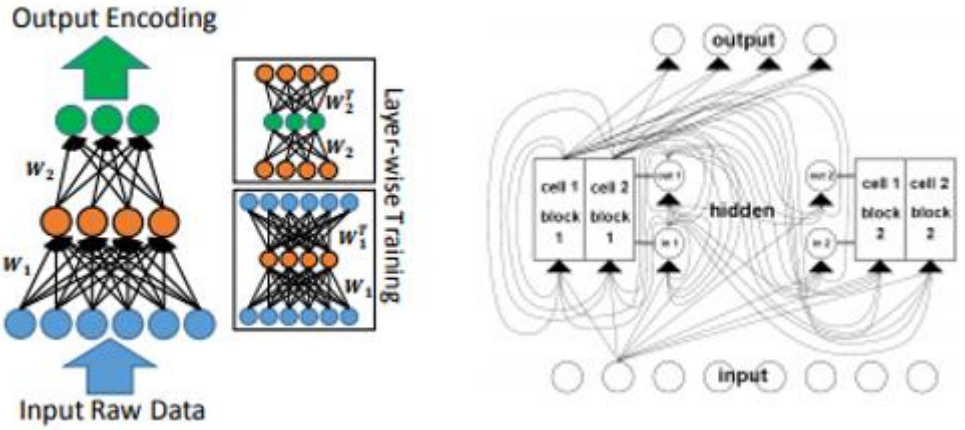
IV.3 Les architectures profondes :

Dans cette section, nous allons décrire brièvement certaines des architectures de réseau neuronal profond, avec leurs principales caractéristiques.

IV.3.1 Auto-encodeurs empilés

Les auto-encodeurs sont des réseaux de neurones artificiels qui sont utiles pour apprendre des représentations efficaces à partir de données «brutes». Pour un ensemble donné d'entrées, un auto-encodeur tente d'apprendre une transformation en une représentation compressée et distribuée. Par conséquent, les auto-encodeurs peuvent être considérés comme effectuant une réduction de dimension (ou une compression avec perte). Un seul auto-encodeur consiste essentiellement en une couche d'entrée correspondant à la représentation d'entrée brute, et une couche cachée qui forme le codage. Un Auto-encodeur empilé est un réseau de neurones constitué de plusieurs couches d'auto-encodeurs, de sorte que les sorties de chaque couche sont câblées aux entrées de la couche successive.

Dans un auto-encodeur, l'apprentissage du codage est essentiellement fait de manière rétroactive où le réseau tente de minimiser une mesure de dissimilarité (l'erreur de reconstruction) entre l'entrée et une reconstruction de l'entrée de la couche cachée (en utilisant des poids inverses). C'est l'un des avantages les plus utiles des auto-encodeurs empilés : leur capacité à s'entraîner couche par couche [83], grâce à laquelle ils sont capables d'apprendre l'encodage à partir de données non supervisées.



(A) (B)

Figure IV.2 : (A) Auto-encodeurs empilés (B) Réseau de neurones récurrents [32]

IV.3.2 Réseaux de neurones récurrents :

Les réseaux neuronaux récurrents (comme la mémoire à long terme [32]) sont des réseaux conçus pour prendre en compte l'information temporelle contenue dans les données afin de générer des codages. Ceci est réalisé grâce à la façon dont les données sont stockées dans le réseau au fil du temps : les fonctions d'activation des neurones dépendent de l'histoire récente des données d'entrée et agissent donc comme une mémoire à court terme, tandis que les poids entre les neurones peuvent être considéré comme une sorte de mémoire à long terme car leurs valeurs sont affectées par les entrées rencontrées par le réseau jusque-là.

Bien sûr, tout comme avec d'autres types de réseaux neuronaux artificiels, ces poids changent lentement (en fonction du taux d'apprentissage), en particulier par rapport à la mémoire à court terme des activations.

IV.3.3 Réseaux de neurones convolutionnels (CNN) :

Les réseaux neuronaux convolutionnels (CNN) sont des réseaux feef-forward multicouches spécialement conçus pour reconnaître des entités dans des données d'image bidimensionnelles. L'architecture des CNN est inspirée de l'étude de Hubel et Wiesel sur le traitement des signaux neurobiologiques dans le cortex visuel du chat [84]. Une application typique de CNN consiste en la reconnaissance de divers objets dans des images. Cependant, les réseaux convolutifs ont été utilisés avec succès pour diverses tâches [85] [86].

Les neurones dans les CNN fonctionnent en considérant une petite partie de l'image, appelons-la sous-image. Les sous-images sont ensuite inspectées pour rechercher les fonctionnalités pouvant être reconnues par le réseau. À titre d'exemple simple, une caractéristique peut être une ligne verticale, une arche ou un cercle. Ces caractéristiques sont ensuite capturées par les cartes d'entités respectives du réseau. Une combinaison de fonctionnalités est ensuite utilisée pour classer l'image. En outre, de multiples cartes de caractéristiques différentes sont utilisées pour rendre le réseau robuste à différents niveaux de contraste, de luminosité, de niveaux de saturation des couleurs, de bruit, etc.

Il existe deux types de couches, tous deux composés de cartes de caractéristiques. Le but de la couche convolutionnelle est de reconnaître les caractéristiques de l'image d'entrée.

Le réseau se compose généralement de plusieurs cartes de caractéristiques, chaque carte reconnaissant certaines caractéristiques. Les sous-images couvrent intentionnellement les régions qui se chevauchent de l'image originale. Une telle conception est importante pour tolérer des distorsions d'image, comme la translation, la rotation, l'inclinaison, etc.

L'autre type de couche, appelé couche de sous-échantillonnage, suit toujours une couche convolutionnelle. Elle se compose du même nombre de mappages d'entités, chaque carte de la couche convolutionnelle étant utilisée comme entrée pour la carte d'entités correspondante dans la couche de sous-échantillonnage suivante. Les sous-images couvertes par les neurones de la couche de sous-échantillonnage ne se chevauchent généralement pas.

Selon la profondeur du réseau, les couches de convolution et de sous-échantillonnage alternent jusqu'à ce que la dernière couche de sous-échantillonnage soit atteinte. Après la dernière couche de sous-échantillonnage, il peut y avoir un nombre quelconque de couches entièrement connectées, la dernière d'entre elles étant la couche de sortie.

IV.3.4 Deep Belief Network :

La première description modélisant un Deep Belief Network (DBN) a été publiée en 1986 par Smolensky [87]. À l'époque, le modèle était appelé «harmonium». C'est un type de réseau neuronal profond composé de plusieurs couches, chaque couche étant constituée de neurones visibles représentant l'entrée de la couche et de neurones cachés représentant la sortie de la couche. Dans ce texte, nous allons considérer une couche pour posséder les neurones cachés. Les neurones visibles appartiendront à la couche précédente, pour laquelle ces neurones sont cachés. Les neurones visibles sont entièrement interconnectés avec ceux qui sont cachés. La caractéristique distinctive d'un DBN est qu'il n'y a pas de connexions entre les neurones visibles, et pas de connexions entre les neurones cachés. Les connexions sont symétriques et sont exclusivement entre les neurones visibles et les neurones cachés.

Dans ce qui suit, nous décrirons l'architecture des machines Boltzmann restreintes, qui représentent le bloc de construction principal des DBN. On expliquera aussi l'architecture du réseau et le processus d'apprentissage. Nous verrons ça en détail, car c'est l'architecture profonde qu'on a choisi pour notre application.

IV.4 Introduction sur la machine de Boltzmann :

Au début des années 1980, Geoffrey Hinton et Terry Sejnowski ont développé le concept original d'une machine Boltzmann. Empruntant au domaine de la thermodynamique, leur recherche s'est étendue aux systèmes d'apprentissage de l'époque pour ajouter une approche stochastique qui permettait d'échapper aux minima locaux durant le processus d'apprentissage [88].

IV.4.1 Structure d'une machine de Boltzmann :

La machine de Boltzmann est «un réseau d'unités binaires stochastiques couplées symétriquement» [89]. Le système comporte une couche d'unités binaires visibles et une couche d'unités binaires cachées. Chaque unité a une connexion bidirectionnelle à chaque autre unité du système.

Le but de cette machine est d'être capable de se stabiliser dans une distribution proche de l'équilibre en utilisant à la fois l'initialisation aléatoire et les unités visibles pilotées par les données. L'idée générale est d'ajuster les poids pour que les initialisations aléatoires se stabilisent dans des états similaires à ceux qui existent dans l'environnement. "On dira que le réseau aura un modèle parfait de l'environnement s'il obtient exactement la même distribution de probabilité sur ces états 2^v quand il fonctionne librement à l'équilibre thermique sans apport environnemental" [88]. Bien que le système ne puisse jamais être un modèle parfait, les régularités dans les données peuvent être capturées pour donner une estimation proche de la vraie probabilité.

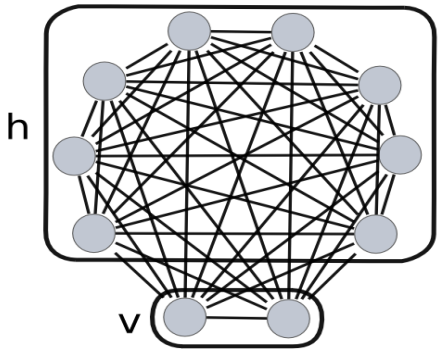


Figure IV.3 : Machine de Boltzmann

IV.4.2 Apprentissage :

Les machines de Boltzmann entrent dans la catégorie de l'apprentissage non supervisé. Contrairement aux méthodes supervisées, il n'y a pas de réponse correcte à comparer et à ajuster. Au lieu de cela, une fonction énergétique [89] de l'état actuel est utilisée :

$$E(v,h;\theta) = -\frac{1}{2}v^T Lv - \frac{1}{2}h^T Jh - \frac{1}{2}v^T Wh \qquad \text{IV.1}$$

L, J et W sont respectivement les poids visible-visible, caché-caché et visible-caché des paramètres du système θ. Cette fonction d'énergie peut ensuite être utilisée pour calculer la probabilité que le système assigne à un ensemble donné d'unités visibles :

$$p(v;\theta) = \frac{1}{Z(\theta)}\sum_h e^{-E(v,h;\theta)} \qquad \text{IV.2}$$

Z (θ) est la fonction de partition qui normalise la probabilité et est donnée par :

$$Z(\theta) = \sum_v \sum_h e^{-E(v,h;\theta)} \qquad \text{IV.3}$$

Notez que la fonction de partition implique le calcul de tous les états possibles du système et des changements si les paramètres du système sont modifiés. Les probabilités conditionnelles ne nécessitent pas la fonction de partition et sont données comme suit :

$$p(h_j = 1 \setminus v, h_{-j}) = \sigma(\sum_{i=1}^{D} W_{ij} v_i + \sum_{m=1/j}^{P} J_{jm} h_j) \qquad \text{IV.4}$$

$$p(v_i = 1 \setminus h, v_{-i}) = \sigma(\sum_{j=1}^{P} W_{ij} h_j + \sum_{k=1/i}^{D} L_{ik} v_i) \qquad \text{IV.5}$$

Le but de l'apprentissage est de mettre à jour les paramètres du système afin qu'il modélise la réalité, aussi précisément que possible. Cela signifierait que les données et le modèle devraient avoir la même probabilité que deux unités soient allumées en même temps. Essentiellement, l'objectif est que l'attente du modèle soit la même que l'attente dépendante des données :

$$E_{p_{model}} = E_{p_{data}} \qquad \text{IV.6}$$

Ep_{model}, plus précisément, l'attente par rapport à la distribution empirique [89] :

$$P_{data}(h,v;\theta) = p(h \setminus v;\theta)P_{data}(v) \quad \text{IV.7}$$

Par conséquent, pour mettre à jour chaque paramètre de sorte que le système monte le long du gradient de log-vraisemblance, les paramètres de poids sont mis à jour en fonction des différences dans les unités qu'ils relient [88] :

$$\Delta W = E_{P_{data}}[vh^T] - E_{P_{model}}[vh^T] \quad \text{IV.8}$$

$$\Delta L = E_{P_{data}}[vv^T] - E_{P_{model}}[vv^T] \quad \text{IV.9}$$

$$\Delta J = E_{P_{data}}[hh^T] - E_{P_{model}}[hh^T] \quad \text{IV.10}$$

IV.5 Machine de Boltzmann restreinte (RBM)

Au milieu des années 1980, le scientifique cognitif Paul Smolensky a développé une théorie du traitement de l'information appelée Harmony Theory. Elle était similaire à l'idée de la machine de Boltzmann, en utilisant des méthodes statistiques, le système a tenté de prédire des variables latentes (caractéristiques environnementales) pour devenir en harmonie avec le véritable environnement. En fait, dans un article publié en 1986, Smolensky a montré comment une machine de Boltzmann modifiée et parallélisée pouvait avoir une inférence exacte avec ce que l'on appelle maintenant une machine de Boltzmann restreinte [87].

IV.5.1 Structure d'un RBM :

La machine de Boltzmann restreinte est une machine de Boltzmann dans laquelle J et L sont mis à 0. En termes simples, il n'y a pas de connexions entre les unités de la même couche, mais chaque unité est encore entièrement connectée aux unités de l'autre couche (figure IV.4). L'avantage de cette configuration peut être vu avec les probabilités conditionnelles réduites. (Les termes de biais ont été omis des probabilités conditionnelles et de l'image de la machine de Boltzmann restreinte).

$$p(h_j = 1 \setminus v) = \sigma(\sum_{i=1}^{D} W_{ij} v_i) \quad \text{IV.11}$$

$$p(v_i = 1 \setminus h) = \sigma(\sum_{j=1}^{P} W_{ij} h_j) \quad \text{IV.12}$$

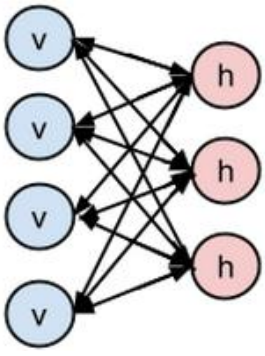


Figure IV.4 : Machine de Boltzmann restreinte

La fonction d'énergie (Equation IV.1) pour une configuration donnée est maintenant simplifiée pour :

$$\mathrm{E}(v,h;W) = -\frac{1}{2}v^T W h$$

Cette simplification signifie que l'état caché peut être calculé en un seul passage parallèle, ascendant, à partir de n'importe quel vecteur visible. Si les vecteurs visibles sont échantillonnés de façon aléatoire à partir des données d'apprentissage, l'attente basée sur les données devient triviale à trouver. Cependant, l'attente des modèles ne dispose pas d'échantillons à partir desquels une méthode d'approximation doit être utilisée [89].

IV.5.2 Apprentissage d'une RBM :

La base pour l'apprentissage avec RBM vient de la méthode de Chaine de Markov de Monte Carlo (MCMC) connue sous le nom d'un échantillonneur de Gibbs. Les chaînes de Markov sont une séquence d'états de variables aléatoires qui sont transférés en utilisant des probabilités conditionnelles connues. Une exigence est que le résultat de chaque transition doit uniquement dépendre de l'état précédent. Pour démarrer une chaîne de Markov, un état initial doit être choisi. Après un petit nombre de transitions, l'état du système est encore fortement influencé par la sélection initiale. Cependant, après suffisamment de transitions, l'état sera indépendant de la sélection initiale et aura atteint une distribution stationnaire. L'échantillonnage à partir de cette distribution stationnaire est considéré comme équivalent à l'échantillonnage à partir des probabilités réelles et une approche de Monte Carlo peut être utilisée pour approximer les attentes du modèle. Un échantillonneur de Gibbs est un cas particulier où les transitions impliquent seulement une variable aléatoire. On peut facilement voir à partir des probabilités conditionnelles du RBM que chaque passage vers le haut ou vers le bas répond à ce critère. Puisque la chaîne de Markov doit alors être construite en demi-étapes, cette méthode est appelée échantillonnage en alternance de Gibbs [90].

IV.5.2.1 Contrastive divergence :

Dans [91], GE. Hinton a proposé un algorithme d'apprentissage efficace en terme de calcul et de temps appelé Divergence Contrastive (Contrastive Divergence : CD). Cet algorithme d'apprentissage est basé sur le fait de minimiser l'énergie du réseau, ça revient à minimiser la distance entre les données originales et les données statistiques générées.

L'apprentissage des paramètres par maximum de vraisemblance standard serait effectué par une ascension en gradient le long de la log-vraisemblance moyenne :

$$\Delta W = \frac{\partial L(W \setminus x)}{\partial W} \qquad \text{IV.13}$$

Où :

$$\frac{\partial L(W \setminus x)}{\partial W} = -\left\langle \frac{\partial E(x \setminus W)}{\partial W} \right\rangle_0 + \left\langle \frac{\partial E(x \setminus W)}{\partial W} \right\rangle_\infty \qquad \text{IV.14}$$

La première partie du côté droit peut facilement être calculée en utilisant des données d'apprentissage. Comme mentionné dans la discussion des machines générales de Boltzmann,

le calcul du second terme implique l'utilisation de la fonction de partition et ne peut pas être fait efficacement. MCMC est un excellent moyen d'échantillonner à partir de distributions complexes et inconnues. Cependant, faire fonctionner la chaîne suffisamment longtemps pour obtenir un bon échantillon est très lent et l'équilibre est difficile à garantir.

La divergence contrastive tente de suivre le gradient d'une fonction entièrement différente. La divergence de Kullback-Leibler est la suivante :

$$KL(\mathrm{p}_0 \backslash\backslash p_\infty) = \sum_x \mathrm{p}_0(x) \log \frac{\mathrm{p}_0(x)}{p(x \backslash W)} \qquad \text{IV.15}$$

Il est possible, selon le système, de minimiser cette divergence elle-même, cependant, afin de laisser la distribution initiale, p_0, inchangée, il est plus logique d'utiliser la différence entre deux divergences. Si le calcul de p_∞ est intraitable en utilisant la différence, cela permet également d'annuler les attentes de la distribution d'équilibre [91].

$$CD_n = KL(\mathrm{p}_0 \backslash\backslash p_\infty) - KL(\mathrm{p}_n \backslash\backslash p_\infty) \qquad \text{IV.16}$$

La minimisation de la divergence contrastive (Equation IV.13) devient donc :

$$\Delta W = \left\langle \frac{\partial E(x \backslash W)}{\partial W} \right\rangle_0 - \left\langle \frac{\partial E(x \backslash W)}{\partial W} \right\rangle_n + \partial \varphi$$

Le troisième terme est le gradient de la distribution de reconstruction par rapport aux paramètres multipliés par le gradient de la divergence KL par rapport à la distribution de reconstruction. Il est "problématique de le calculer, mais des simulations étendues montrent qu'il peut être ignoré en toute sécurité car il est petit et il s'oppose rarement à la résultante des deux autres termes." [91]

Il convient de noter que la divergence de Kullback-Leibler n'est pas symétrique ($KL(\mathrm{p}_0 \backslash\backslash p_\infty) \neq KL(\mathrm{p}_\infty \backslash\backslash p_0)$) et fournit une estimation biaisée de l'apprentissage du maximum de vraisemblance. Cependant, le biais a été montré pour être petit et un certain nombre d'applications réussies ont prouvé qu'il est un signal d'apprentissage efficace et efficient. Il convient également de noter que puisque le troisième terme de la différence n'est pas utilisé, l'algorithme d'apprentissage ne suit réellement aucune fonction. Pour une machine de Boltzmann restreinte, les mises à jour de paramètres utilisant la divergence contrastive deviennent :

$$\Delta \mathrm{W} = < vh >_0 - < vh >_n \qquad \text{IV.17}$$

L'apprentissage fonctionne même si la reconstruction utilisée pour estimer l'attente des modèles se fait en une seule étape. Ceci est noté CD_1. Le compromis pour plusieurs étapes (CD_n) est une amélioration de l'estimation des attentes des modèles (et donc une meilleure approximation du gradient) pour le temps requis pour une autre passe ascendante. Une grande partie de la littérature commune utilisant la divergence contrastive comme méthode d'apprentissage utilise CD_5 [92].

IV.5.2.1 Persistant CD :

À l'heure actuelle, la divergence contrastive est l'une des approximations les plus populaires des RBM. Cependant, il y a plusieurs modifications à l'algorithme standard du CD, et il n'est pas évident de savoir lequel est le meilleur. Une alternative très courante est le CD persistant, abrégé en PCD. Certaines recherches affirment que les PCD produisent des détecteurs de caractéristiques plus significatifs et surpassent les autres variantes des algorithmes CD [93].

L'algorithme PCD utilise une approximation différente pour les états d'échantillonnage que CD. Il élimine l'étape d'initialisation des états cachés en fonction du modèle d'entrée. Par conséquent, la chaîne d'échantillonnage est en cours de redémarrage pour chaque motif observé. Au lieu de cela, PCD initialise les états cachés une seule fois au début de l'entraînement. Cela provoque une chaîne unique d'échantillonnage tout au long de l'apprentissage, ce qui permet de se déplacer plus rapidement vers le modèle de distribution plutôt que vers les données. Plus le taux d'apprentissage est petit, plus le PCD fonctionne [93]. C'est parce que les mises à jour de paramètres plus petites sont alors assez petites comparées aux changements dans la chaîne d'échantillonnage, et la chaîne peut plus facilement rattraper les changements dans le modèle.

IV.6 Deep Belief Netwrok (DBN) :

Les machines de Boltzmann restreintes sont capables de détecter et d'extraire des caractéristiques à partir de données d'entrée. Plusieurs couches de RBM peuvent être empilées les unes sur les autres pour former un réseau multicouche appelé DBN [94]. Chaque couche RBM utilise les neurones cachés de la couche RBM précédente comme entrée (voir Figure IV.5). L'architecture profonde avec plusieurs couches peut alors extraire une représentation hiérarchique profonde des données d'apprentissage.

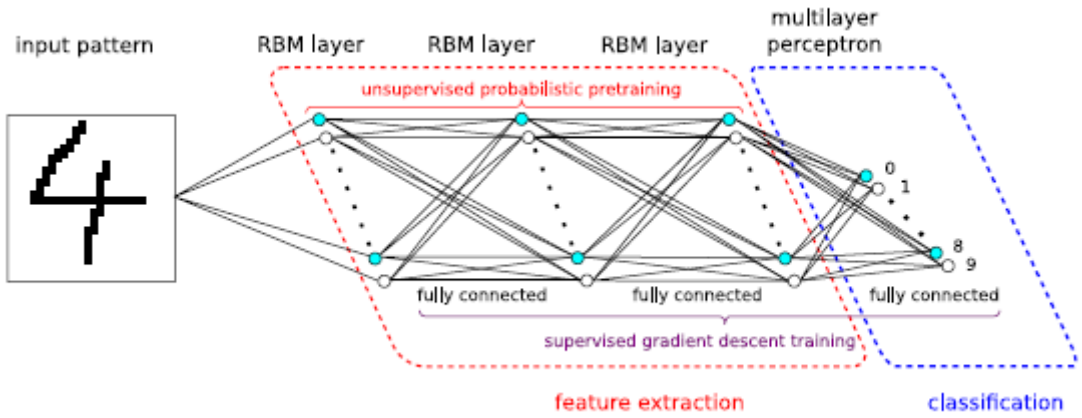


Figure IV.5 : Le réseau Deep Belief Network composé d'empilement de RBMs.

Un ensemble de couches RBM effectue la tâche de détection des entités, tandis que la tâche de classification est effectuée en utilisant un perceptron multicouche comme dernière couche. La dernière couche MLP utilise à nouveau les neurones cachés de la couche RBM précédente comme entrée. L'architecture qui en résulte est un mélange de neurones probabilistes en phase d'extraction de caractéristiques et de neurones déterministes en phase de classification.

IV.6.1 L'apprentissage du Deep Belief Network :

Comme expliqué dans le paragraphe précédent, l'architecture d'un DBN est constituée de couches RBM et d'un MLP. Cependant, les RBM utilisent l'apprentissage non supervisé, tandis

que les MLP utilisent un apprentissage supervisé. Cela se traduit par un processus d'apprentissage en deux phases.

La première phase, appelée pré-entraînement, effectue l'entraînement non supervisé de chaque couche RBM séparément [94, 83]. La seconde phase utilise la méthode du gradient descendant pour l'entraînement supervisé des couches MLP. Il peut être implémenté à l'aide, par exemple, de l'algorithme de propagation standard décrit dans le chapitre précédent.

Le processus de pré-entraînement et d'apprentissage peut être résumé comme suit :

1. Le réseau est initialisé avec de petites valeurs de poids, biais et autres paramètres aléatoires.

2. La première couche RBM est initialisée avec des données d'entrée représentant des potentiels dans ses neurones visibles. Ensuite, l'apprentissage non supervisé est effectué sur cette couche en itération sur l'ensemble de données d'apprentissage pour un nombre prédéfini d'itérations.

3. La couche suivante obtient son entrée en échantillonnant les potentiels générés dans les neurones cachés de la couche précédente. Ensuite, l'apprentissage non supervisé est effectué sur cette couche en itération sur l'ensemble de données d'apprentissage.

4. Répéter l'étape précédente pour le nombre de couches souhaité. Dans chaque itération, les échantillons sont propagés vers le haut dans le réseau. La phase de préapprentissage est terminée lorsque la première couche MLP est atteinte.

5. L'apprentissage supervisé commence et s'arrêté après avoir atteint un nombre prédéfini d'itérations ou après avoir atteint le taux d'erreur ciblé.

Notez que structurellement, les neurones du DBN sont interconnectés de la même manière que le réseau MLP (voir figure IV.5). Cela permet d'effectuer la deuxième phase d'apprentissage exactement comme si la formation du réseau MLP était en cours. Par conséquent, toute la procédure d'apprentissage équivaut à initialiser les poids et les biais d'un réseau MLP profond avec les valeurs obtenues dans le pré-test probabiliste non supervisé. Une fois le réseau formé, la classification des données d'entrée présentées est effectuée exactement comme dans le cas du réseau MLP.

IV.6.2 Pré-apprentissage gourmand (Greedy pre-training) :

Nous avons vu comment une machine Boltzmann restreinte peut apprendre des fonctionnalités. Dans un modèle hiérarchique, nous voulons maintenant apprendre les caractéristiques de ces fonctionnalités. Nous pouvons le faire en créant un autre RBM, pour lequel l'entrée est le premier ensemble de caractéristiques apprises des données (l'état des unités cachées du premier RBM, lorsque l'entrée est un vecteur de données). Ce processus peut être répété plusieurs fois, permettant l'apprentissage de couches de fonctionnalités de plus en plus élevées. On peut prouver que chaque fois que nous ajoutons une autre couche, nous améliorons la limite inférieure variationnelle de la probabilité de générer les données [82].

Après avoir créé ces RBM, nous devons les combiner. La façon dont les RBM sont combinées n'est pas complètement évidente : pour la couche supérieure RBM, nous conservons les connexions non dirigées (telles qu'elles forment une mémoire associative), mais pour les niveaux inférieurs, nous ne conservons que les pondérations génératives. Nous gardons toujours les poids de reconnaissance minimaux, mais ils ne font pas partie du modèle. Les poids de

reconnaissance sont la transposition des poids génératifs et seront utilisés pour l'inférence. La figure IV.6 illustre la différence entre les poids de reconnaissance et les poids génératifs.

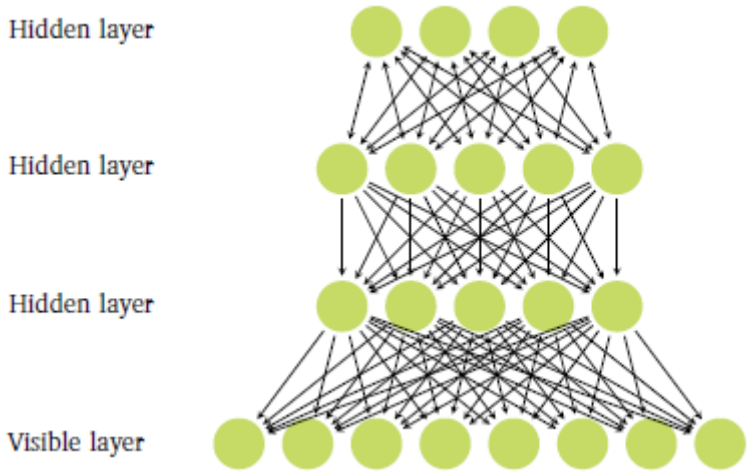


Figure IV.6 : Un réseau DBN en tant que modèle génératif.

Le modèle qui en résulte est un modèle génératif. Nous devons donc nous poser la question suivante : quelle distribution modélise-t-il et comment générons-nous les données ? Les DBN modélisent la distribution conjointe entre les données observables (v) et les variables cachées latentes (vecteurs de caractéristiques h_k) :

$$p(v,h_1,h_2,...h_n) = p(h_n,h_{n-1}).\prod_{k=0}^{n-2} p(h_k \mid h_{k+1}) \qquad \text{IV.18}$$

Comme observation intéressante, nous notons que les DBN sont récursifs : supprimer la couche visible d'un réseau DBN avec plus de 3 couches aboutira à un autre réseau DBN, avec une couche de moins.

IV.6.3 Amélioration de l'apprentissage gourmand :

Pour pouvoir empiler deux RBM les unes sur les autres, le nombre d'unités cachées du premier RBM doit être égal au nombre d'unités visibles du second RBM. Cela est nécessaire pour pouvoir propager les activations cachées du premier RBM en tant que données d'apprentissage pour le deuxième RBM.

Que faire si nous savons également que le nombre d'unités visibles du premier RBM est égal au nombre d'unités cachées du second RBM ? La figure IV.7 montre un tel exemple. Si le modèle RBM est symétrique et que les activations cachées du premier RBM sont les entrées du second, les deux réseaux tentent de modéliser des corrélations similaires, de sorte que les poids appris par le premier peuvent être utilisés pour initialiser les poids du second RBM.

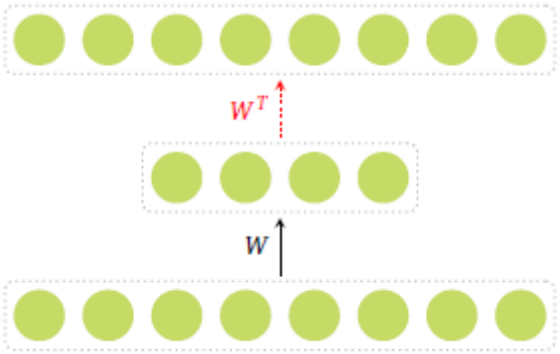


Figure IV.7 : Architecture réseau requise pour initialiser les poids d'un réseau à la transposition des poids du réseau formé précédent. Les poids en noir sont ceux après l'entraînement du RBM correspondante et ceux en rouge avant l'entraînement du RBM.

Comme indiqué ci-dessus, cette initialisation est faite uniquement si les RBM sont symétriques. Dans leur formulation initiale, les RBM sont symétriques car les unités tant cachées que visibles utilisent la même fonction d'activation, à savoir la sigmoïde. Cependant, des travaux récents [93] ont montré que l'utilisation d'unités gaussiennes visibles et d'unités linéaires rectifiées bruyantes pouvait améliorer les performances des RBM. Ce type de RBM n'est pas symétrique, en raison des différentes fonctions d'activation. De plus, il est courant de mettre à l'échelle les données d'entrée vers une RBM avec des unités visibles gaussiennes pour avoir une moyenne et une variance d'unité nulles afin de ne pas apprendre la variance des unités visibles à l'aide du CD. Dans ce cas, les données d'entrée pour le deuxième RBM ne sont pas données par les activations cachées du premier RBM, mais par des activations cachées mises à l'échelle. Ainsi, le deuxième modèle RBM modélise différentes corrélations, ce qui signifie qu'il ne faut pas initialiser les poids du deuxième RBM à ceux résultant de la formation du premier réseau, même si la forme des deux réseaux le permet.

IV.6.4 Génération des données du DBN

Étant donné que les DBN sont des modèles génératifs, nous aimerions pouvoir échantillonner à partir de la distribution qu'ils définissent. Cela se fait comme suit :

1) Obtenir un échantillon d'équilibre du RBM de niveau supérieur (en effectuant un échantillonnage de Gibbs alterné entre les deux couches)

2) À partir des réseaux cachés de la couche supérieure RBM, utilisez les poids génératifs descendants pour effectuer un passage dans le réseau.

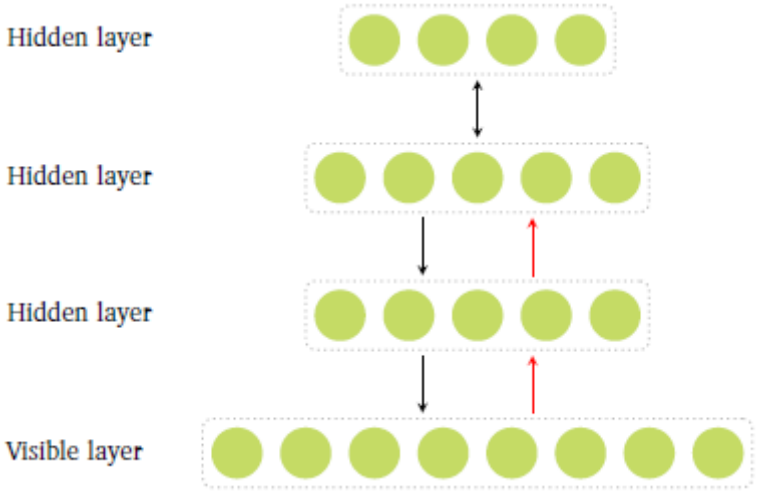


Figure IV.8 : Poids générés par rapport aux poids de reconnaissance dans un DBN. Les poids de reconnaissance sont représentés en rouge.

IV.6.5 Classification à l'aide des DBN :

Jusqu'à présent, nous avons discuté des réseaux DBN en tant que modèles génératifs. Ils peuvent être adaptés et utilisés pour la classification et la régression. Pour utiliser des réseaux profonds pour la classification, une autre couche doit être ajoutée au-dessus du réseau. Une autre option consiste à alimenter les dernières valeurs de la couche en entrée dans un autre classificateur, tel qu'une machine à vecteurs de support [10].

Pour former le réseau, nous effectuons d'abord l'apprentissage gourmand (Greedy) couche par couche. Ensuite, nous appliquons une rétro-propagation à l'ensemble du réseau afin d'apprendre à distinguer les étiquettes de classes.

Cette approche élimine une grande partie des problèmes généralement rencontrés avec la rétro-propagation :

• La rétro-propagation ne doit pas apprendre les caractéristiques des données. La tâche a été faite par l'apprentissage gourmand. Cela résout le problème du gradient de disparition : le but principal de la rétro-propagation est d'apprendre les poids de la couche supérieure (discriminante), car les poids des premières couches ont déjà des valeurs sensibles. Si le gradient est trop petit pour affecter les premières couches, l'impact sur l'apprentissage n'est pas aussi radical.

• L'algorithme est moins susceptible de rester bloqué dans un minimum local de la fonction énergie, en raison de l'initialisation sensible des poids.

• Des données moins étiquetées sont nécessaires. Le pré-apprentissage gourmand ne nécessite pas de données étiquetées, car il est non supervisée. Les données étiquetées sont une ressource rare, car leur obtention implique un travail manuel. Exiger moins de données étiquetées est un plus pour tout l'algorithme, car il peut être donné en tant que jeux de données plus volumineux.

• Le pré-apprentissage gourmand entraîne moins de sur-ajustement que la simple utilisation de la rétro-propagande standard, car les données d'entrée fournissent beaucoup plus d'informations (à savoir les caractéristiques de niveau supérieur acquises lors de la première phase de l'apprentissage).

IV.7 Conclusion

Dans ce chapitre, nous avons exploré le concept du Deep Learning et comment ce volet a permis de résoudre plusieurs problématiques. Nous avons aussi examiné les variantes du Deep Learning, qui sont les Autoencodeurs, le CNN et le DBN, en mettant l'accent sur ce dernier. Nous avons présenté son architecture composée essentiellement d'un empilement de machines de Boltzmann restreintes et d'un MLP classique, son apprentissage non supervisé dans la partie features extraction et supervisé dans la partie MLP. Dans le chapitre suivant, nous allons utiliser les différentes architectures peu-profondes (MLP, TDNN, RBF... etc) et profonde (DBN) pour faire la classification du manuscrit arabe dans ces deux formes (caractères et mots).

Chapitre V

Système proposé, expériences et résultats

5.1 Motivation du travail :

L'apprentissage d'un dispositif de reconnaissance est l'une des étapes les plus importantes de toute tâche de reconnaissance de manuscrit. La présence de suffisamment d'échantillons d'apprentissage de chaque classe est très importante pour bien former le dispositif. Les classes peuvent représenter des caractères, des formes de caractères, des mots, des traits ou toute autre représentation appropriée, comme celles présentées dans les sections précédentes. Pour faciliter la recherche dans ce domaine, des bases de données de référence sont développées pour fournir des données pour l'apprentissage et le test du système de reconnaissance, seul bémol le volet en ligne est un peu négligé.

Afin de garantir une base de données d'apprentissage adéquate, de grandes quantités de données sont collectées et étiquetées. La collecte, l'étiquetage et la vérification des données sont des activités à forte intensité de main-d'œuvre et de temps. Les chercheurs conviennent généralement que la quantité et la qualité des données d'apprentissage sont importantes [94]. De plus, comme les données collectées dans un environnement ne conviennent généralement pas à une tâche de reconnaissance de texte dans un environnement différent, il est nécessaire de préparer de nouveaux jeux de données pour différents scénarios de reconnaissance de texte.

Ce chapitre présente une large utilisation des réseaux de neurones et la construction d'une nouvelle approche (DBN) pour la reconnaissance de l'écriture manuscrite en arabe. De plus, il décrit les résultats substantiels obtenus par ces méthodes neuronales.

Récemment, des chercheurs ont montré un intérêt accru dans le domaine de la reconnaissance de l'écriture manuscrite arabe [95,96]. Cet intérêt nous motive à développer cette approche systémique.

5.2 Système proposé :

Le système de reconnaissance du manuscrit arabe acquiert les données provenant d'un équipement en ligne (tablette dans notre cas). En premier lieu, nous procédons à un prétraitement nécessaire pour la tache de reconnaissance, car les données brutes ne permettent pas de récupérer de bons résultats. Par la suite, nous effectuons une extraction des caractéristiques pertinentes du signal, plus le nombre de caractéristiques est élevé, plus les taux de reconnaissance sont meilleurs. Enfin, la matrice des caractéristiques est présentée à l'entrée des différents classifieurs utilisés afin d'attribuer une étiquette à la forme d'entrée. Néanmoins, un réglage des paramètres est primordial pour avoir des résultats plus au moins satisfaisants. La figure suivante représente le schéma global du système de reconnaissance proposé :

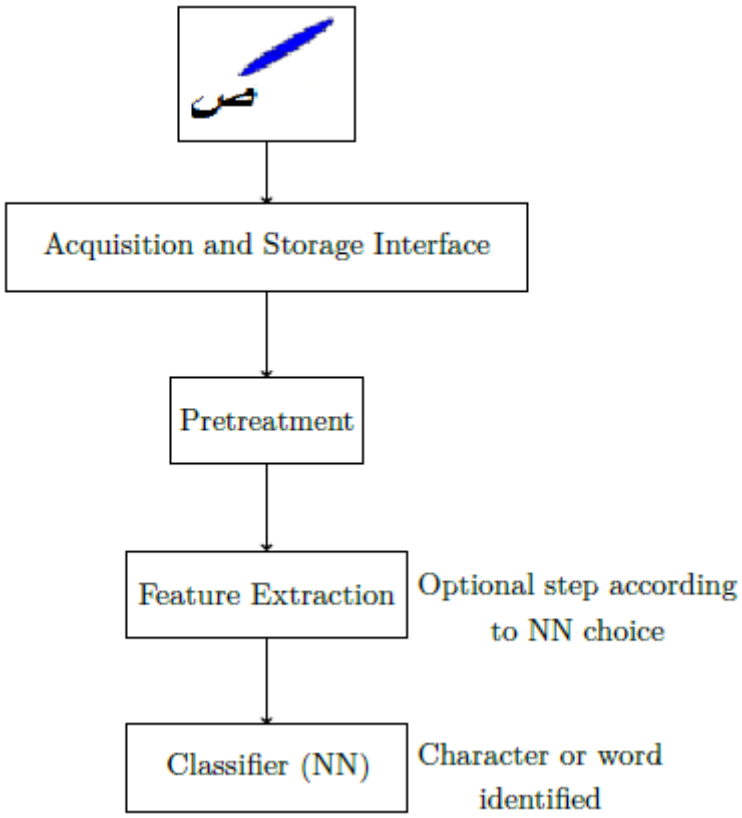


Figure V.1 : Le système de reconnaissance du manuscrit arabe proposé

5.3 DISPOSITIFS MATÉRIELS

5.3.1 Ordinateur

Pour la réalisation des expériences, un ordinateur portable a été utilisé avec :

- Un processeur Intel (R) Core (TM) i7-2670QM CPU avec vitesse d'horloge 2.20 GHz

- Mémoire vive RAM estimée à 6 Go.

- Une carte graphique NVIDIA GEFORCE with CUDA de 1Go.

- Le système d'exploitation utilisé est le Windows seven (64 bits) édition Intégrale.

Toutes les expériences sont développées sous MATLAB version (R2016b).

5.3.2 Tablette PC

Pour l'acquisition du signal d'écriture et la construction de la base de données « NOUN » version 2 (la version 1 contient les caractères arabes isolés seulement), nous avons utilisés une tablette WACOM BAMBOO version 5.08-6 ainsi qu'une interface graphique conçu spécialement pour cette tache dont laquelle nous pourrions garder des informations sur l'acquisition telles que la date, le nom du scripteur... . Les deux figures V.2 et V.3 illustrent la tablette et l'interface d'acquisition respectivement.

Figure V.2 : Tablette WACOM BAMBOO version 5.08-6 utilisée pour l'acquisition du signal d'écriture

Il est conseillé de tenir le stylet comme un crayon classique. Le stylet peut être incliné pour travailler. Les boutons de stylet sont positionnés à pouvoir appuyer facilement dessus avec le pouce ou l'index. Avant l'utilisation de la tablette, il faut organiser l'espace de travail de manière à travailler confortablement. Il faut placer la tablette, le stylet et le clavier de façon qu'ils soient aisément accessibles. Il faut aussi placer le moniteur de manière à le voir avec aisance et un minimum de fatigue oculaire. Pour une utilisation optimale, la tablette doit être orientée de sorte que le curseur de l'écran se déplace dans le même sens de la main du scripteur.

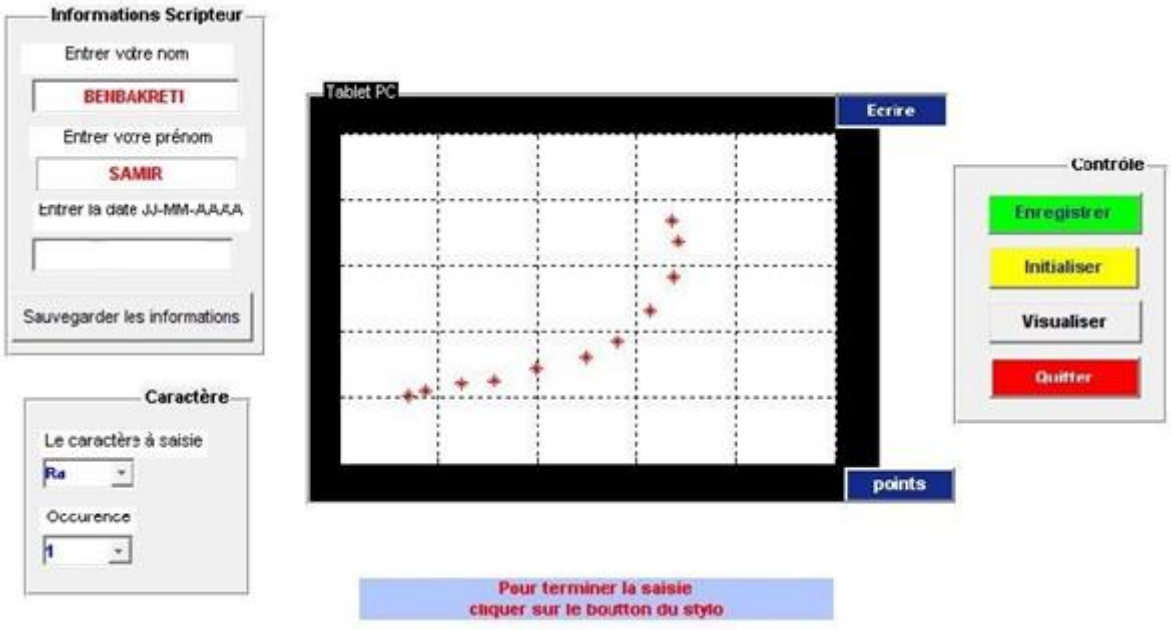


Figure V.3 : Interface d'acquisition du signal d'écriture

Pour l'interface graphique, le scripteur est prié d'inscrire son *nom, prénom et la date de saisie* dans l'onglet **« Information Scripteur »** mentionné à gauche dans l'interface.

En bleu, les deux boutons : *« Écrire »* et *« points »* sont réservé pour saisir le signal d'écriture (caractère ou mot) et les points diacritiques respectivement. Le scripteur doit faire une sélection du caractère à saisir et quelle occurrence dans l'onglet **« caractère »**.

L'onglet **« contrôle »** à droite sert à appliquer quelques contrôles utiles pour la construction des échantillons. En vert, le bouton *« enregistrer »* pour la sauvegarde, en jaune le bouton *« initialiser »* pour la remise à zéro de toutes les informations sur l'interface. En gris pour la visualisation du caractère sauvegardé et en rouge pour quitter l'interface. (Voir figure V.3)

5.4 Prétraitement du signal d'écriture :

La tâche de reconnaissance des caractères/mots est difficile en raison de la grande variabilité inhérente au style d'écriture manuscrite et de la variation de la position du caractère dans le mot.

Lors de la construction de notre base de données nous avons fait appel à la tablette WACOM, les données brutes du signal d'écriture contiennent d'importantes variations selon la vitesse d'écriture se traduisant par un nombre différent des points formant la lettre. Entraîner un réseau de neurones avec de telles données peut engendrer de très faibles performances pouvant atteindre les 70% de taux d'erreurs, raison pour laquelle l'étape de prétraitement est primordiale.

L'étape de prétraitement décrite dans le système de reconnaissance mentionné précédemment (figure V.1) a pour objectif, la normalisation de la taille des caractères et l'échantillonnage des tracés en un nombre fixe de points équidistants.

L'étape suivante consiste à extraire les caractéristiques du tracé précédemment échantillonnée, ce qui prépare l'entrée pour le classifieur c.-à-d. le réseau de neurones. Après la phase d'apprentissage, la classe du caractère est fournie au niveau de la couche de sortie. Cette procédure est respectée par presque tous les réseaux de neurones. Cependant, les réseaux de neurones profonds ne nécessitent aucune phase d'extraction de caractéristiques.

Pour la plupart des lettres arabes, des similitudes sont notées entre les formes au début / au milieu d'une part et la fin / à l'isolement de l'autre. La présence d'une ligature avec une lettre précédente / suivante ne modifie pas de manière significative la forme de la lettre (pas plus que dans le cas de l'écriture cursive en latin). Les ligatures arabes se produisent lorsque deux lettres sont écrites l'une sur l'autre. De plus, l'écriture arabe est riche en signes diacritiques (ou signes secondaires) tels que voyelles, points, chaddah, maddah, hamzah ... etc. Dans notre travail, nous définissons une marque diacritique comme une composante secondaire d'une lettre, qui peut la compléter ou même modifier tout le sens du mot. Mais la notion des voyelles telles que : el damah, el kasrah, el fathah et el soukoun ne sont pas prises en compte.

Ces spécificités de la langue arabe rendent la tâche de reconnaissance plus complexe, d'où la nécessité de procéder à un prétraitement comprenant les étapes suivantes :

5.4.1 Echantillonnage spatial :

Il permet de conserver les informations utiles du signal et d'exclure les informations redondantes résultant de la répétition de points. En effet, la durée de formation du caractère/mot ainsi que sa forme peuvent varier considérablement d'un scripteur à un autre et d'un moment à

un autre. L'échantillonnage spatial transforme un signal d'écriture manuscrite en une séquence de points de coordonnées [X (n), Y (n)] (n est l'indice de points) en fonction de la longueur du tracé en un nombre fixe de points. Voici l'algorithme de l'échantillonnage spatial :

. Définition de la ligne entre un Penup et PenDown

N : nombre de points d'échantillonnage

Réservation d'espace mémoire pour les vecteurs suivants : X, Y, PenUp / Down

. Calcul de la longueur totale du signal

Si un seul trait :

Longueur = longueur + distance entre les points

Si plusieurs traits :

Longueur = longueur + longueur entre les traits

. Calcul de la distance d'échantillonnage

Dist-samp = Longueur / (N-1)

Détermination de N-1 points par interpolation

5.4.2 Normalisation et centrage des caractères/mots :

Ça permet d'obtenir une représentation invariante vis-à-vis des translations et des distorsions spatiales. Le caractère/mot est recentré sur (X0, Y0) puis normalisé sur la taille maximale du caractère/mot. Les étapes de recentrage de la forme sont détaillées dans la première partie de l'algorithme 1 de la figure V.5.

Un exemple de résultat de ces deux étapes de prétraitement est illustré à la figure suivante :

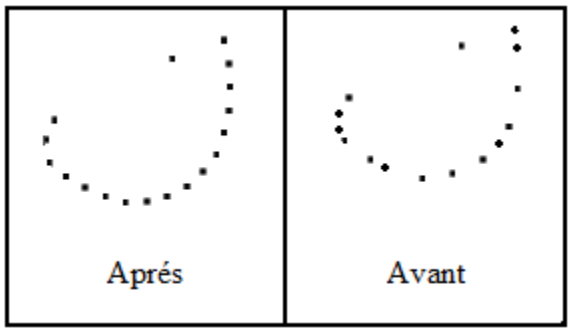


Figure V.4 : Caractère isolé "Noun" avant et après la phase de prétraitement

5.5 Extraction des caractéristiques :

L'étape de l'extraction des fonctionnalités consiste à extraire les fonctionnalités pertinentes des données brutes de l'écriture manuscrite en ligne. Comme il est inutile d'entrer ces données directement dans la phase de reconnaissance, il est nécessaire d'extraire les caractéristiques pour obtenir une reconnaissance efficace et précise.

Cette étape consistait à transformer la série ordonnée de points de l'algorithme de prétraitement du tracé (caractère ou mot), en un vecteur de caractéristiques. Ceci peut ensuite être utilisé pour identifier la classe de test par comparaison aux échantillons du jeu d'apprentissage.

Étant donné que chaque tracé peut avoir une longueur différente, le nombre de points peut varier. Cela produirait des vecteurs caractéristiques de différentes tailles. Comme ils ne peuvent pas être comparés directement, le nombre de points est normalisé à une valeur standard par interpolation. Les vecteurs caractéristiques formés auront donc tous, une longueur standard et sont donc directement comparables. Lors de l'inspection visuelle des points interpolés, 20 ont été choisis comme longueur standard.

Les caractéristiques que nous avons prélevées de chaque point du tracé sont :

- Les coordonnées en x et y
- Les cosinus directeurs de la direction du mouvement,
- Les cosinus directeurs de la courbure de la trajectoire
- la position du poser\lever du stylet.

L'algorithme suivant décrit en détails les étapes d'extraction (deuxième partie).

Algorithm 1

$xmax \leftarrow x(0)$
$xmin \leftarrow x(0)$
$ymax \leftarrow y(0)$
$ymin \leftarrow y(0)$
{Calculation of the center of the character $[x0, y0]$}
$n \leftarrow 0$
while $n \leq N-1$ **do**
  if $x(n) > xmax$ **then** $xmax \leftarrow x(n)$ **end if**
  if $x(n) < xmin$ **then** $xmin \leftarrow x(n)$ **end if**
  if $y(n) > ymax$ **then** $ymax \leftarrow y(n)$ **end if**
  if $y(n) < ymin$ **then** $ymin \leftarrow y(n)$ **end if**
end while
$x0 \leftarrow (xmin + xmax)/2$
$y0 \leftarrow (ymax + ymin)/2$
{$Delta$ scaling factor calculation}
$Delta_y \leftarrow (ymax - ymin)$
$Delta_x \leftarrow (xmax - xmin)$
if $Delta_x < Delta_y$ **then**
  $Delta \leftarrow Delta_y$
else
  $Delta \leftarrow Delta_x$
end if
{Calculation of the new coordinates $x[n][1]$ and $x[n][2]$: $x[n][1]$ the x coordinates $x[n][2]$ the y coordiantes. Scanning of (X and Y) points}
$x[n][1] \leftarrow (X[n] - x0)/Delta$
$x[n][2] \leftarrow (Y[n] - y0)/Delta$
$x[n][7] \leftarrow penUpDown[n]$
{Calculation of the direction $x[n][3]$ and $x[n][4]$: cosine directions Scanning points Calculation of the length of the dS string}
$dx \leftarrow X[i+1] * X[i-1]$
$dy \leftarrow Y[i+1] * Y[i-1]$
$dS \leftarrow \text{sqrt}(dx * dx + dy * dy)$
if $dS = 0$ **then**
  $x[i][3] \leftarrow 0$
  $x[i][4] \leftarrow 0$
else
  $x[i][3] \leftarrow arc_x/dS$
  $x[i][4] \leftarrow arc_y/dS$
end if
{Special points: initial and last point}
$Initial\ point \leftarrow next\ point$
$Last\ point \leftarrow previous\ point$
{Curvature calculation $x[n][5]$ and $x[n][6]$: cosine directions Scanning points}
$x[i][5] \leftarrow x[i+1][3] * x[i-1][3] + x[i+1][4] * x[i-1][4]$
$x[i][6] \leftarrow x[i+1][3] * x[i-1][4] + x[i+1][4] * x[i-1][3]$
{Special points: initial and last point}
$Initialpoint \leftarrow nextpoint$
$Lastpoint \leftarrow previouspoint$

Figure V.5 : Algorithme de recentrage de la forme du signal acquise et d'extraction de caractéristiques.

Alors pour résumer, l'extraction des caractéristiques de chaque point du tracé nous permet de construire une matrice de 7*20, les caractéristiques étant : les coordonnées en x, les cordonnées

en y, les cosinus directeurs de la direction (cos θ et sin θ), les cosinus directeurs de la courbure (cos Φ et sin Φ) et la position poser\lever du stylet. C'est la matrice que nous propagerons dans les architectures neuronales décrites dans le chapitre précédent.

5.6 Classification du manuscrit arabe :

L'étape de classification est l'étape essentielle de la reconnaissance. Elle mappe les caractéristiques extraites de l'écriture arabe en ligne, en un ensemble de catégories ou de classes finies. Ces catégories peuvent être des mots, des sous-mots ou des lettres en langue arabe. La reconnaissance de formes utilise de nombreuses méthodes neuronales pour résoudre les problèmes de classification comme décrit dans le chapitre 3. Ces méthodes ainsi que les principaux paramètres de chaque réseau de neurones utilisé sont décrits dans la suite :

5.6.1 MLP :

Le premier modèle testé est le Perceptron multicouche. Comme cela a déjà été expliqué dans les chapitres précédents, il s'agit d'un réseau à feed-forward entièrement connecté. La connectivité totale implique un grand nombre de poids de réseau. Pour cette raison, le modèle est assez robuste pour un grand nombre de neurones cachés.

Nous avons utilisé le MLP avec une rétro-propagation d'erreur avec une seule couche cachée. Les sept caractéristiques extraites des caractères Arabes (caractères sous leurs formes isolées, au début, au milieu et à la fin), générées par le module précédent, constituent les entrées du réseau. La couche cachée comprend 80 neurones. Les classes à distinguer sont les 28 caractères de l'alphabet arabe d'où le choix de 28 neurones pour la couche de sortie. Le sigmoïde unipolaire a été utilisé comme fonction d'activation des neurones. La figure montre comment le MLP traite une lettre en l'occurrence « Mim ».

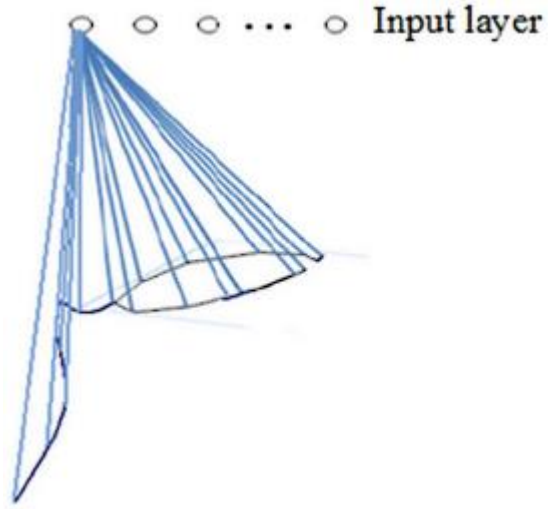


Figure V.6 : Connexion de type MLP - Fenêtre de vue globale -

Cependant, l'apprentissage de la couche la plus basse (la plus proche de la couche en sortie) est moins efficace dans un MLP profond à plusieurs couches cachées [97,100]. Il semble que la mise à jour des paramètres soit de moins en moins pertinente puisque nous propageons dans les couches les plus basses. Pour cette raison, nous utilisons une seule couche cachée pour notre MLP.

5.6.2 TDNN :

L'architecture testée du TDNN est un réseau à 3 couches, la première étant la couche d'entrée du réseau, la seconde est la couche cachée de la partie extraction ainsi que la première couche de la partie classification (la fonction d'activation étant une sigmoïde), la dernière est la couche de sortie (la fonction d'activation étant une fonction linéaire).

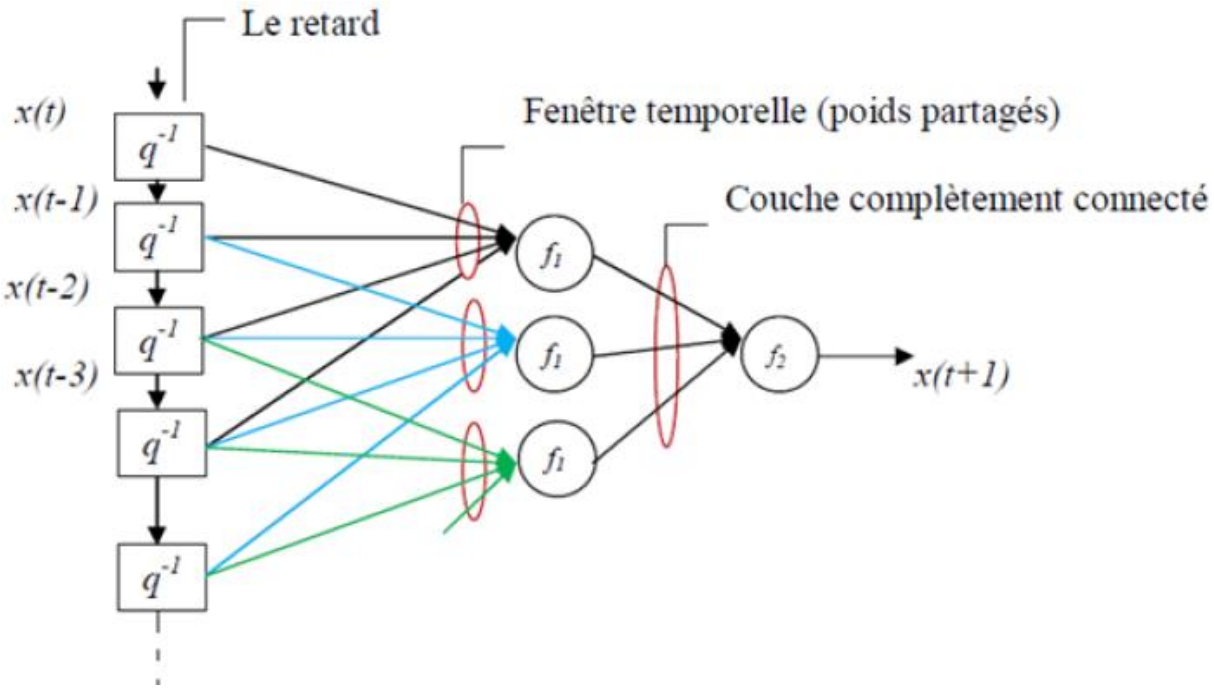


Figure V.7 : TDNN avec une fenêtre temporelle de taille 4 reliée à chaque neurones de la couche caché.

Le TDNN se distingue d'un perceptron multicouche par le fait qu'il prend en compte une notion de temps. Par conséquent, il traite tous les neurones de la couche d'entrée en même temps, avant de procéder à un balayage temporel. C'est la notion de fenêtre temporelle, celle-ci a été fixé à 4 neurones et le délai entres les fenêtres est estimé à 3 neurones selon l'axe temporel.

5.6.2.1 DTDNN :

Le TDNN distribué a été introduit pour la première fois pour la reconnaissance de phonèmes par Waibel [70]. L'architecture d'origine était très spécialisée pour ce problème particulier.

La seule différence entre le réseau TDNN et la DTDNN est que dans le premier cité, nous spécifions le retard dans la couche d'entrée uniquement, tandis que dans le second, nous introduisons les retards dans toutes les couches cachées. Dans notre réseau, des retards de 6 neurones sont utilisés dans la couche d'entrée et de 3 neurones dans la couche cachée, le nombre de neurones dans la couche cachée est de 200 neurones. La fonction d'apprentissage est la même que celle utilisée dans TDNN, une rétro-propagation de gradient conjugué.

5.6.3 RBF :

Le réseau de fonctions de base radiale (RBF) utilise des fonctions radiales plutôt que sigmoïdes (cas du MLP) pour créer une fonction de décision locale centrée sur un sous-ensemble à partir des entrées. La somme de toutes les fonctions locales représente la fonction de décision globale

pour résoudre le problème des minima locaux. Le réseau RBF est constitué de 3 couches. Chaque nœud caché applique une fonction du noyau aux données en entrée, puis la couche en sortie effectue une somme pondérée de ces fonctions du noyau. Chaque nœud est caractérisé par deux paramètres, qui sont la largeur et le centre de la fonction radiale. Si les données d'entrée d'un nœud sont proches du centre (estimées à l'aide de la distance euclidienne), sa valeur de sortie sera élevée. Dans ce travail, nous avons utilisé une fonction du noyau gaussien (Gaussian Kernel Function).

La configuration complète du réseau est obtenue après détermination du centre et de la largeur associés à chaque nœud ainsi que du poids des connexions entre la couche cachée et la couche en sortie, qui contient les 28 caractères arabes. Dans ce travail, nous avons expérimenté plusieurs variantes de réseaux RBF, une brève description de chacune d'elles est donnée dans ce qui suit :

5.6.3.1 RBFe (exact) :

La conception d'un RBFe peut être réalisée à l'aide d'une fonction prenant en entrée : les matrices de vecteurs d'entrée P, les vecteurs cibles T et le facteur d'étalement de la couche de base radiale. Puis renvoie un réseau avec des pondérations et des valeurs de biais telle que la sortie est exactement T quand l'entrée est P. La même fonction crée autant de neurones de base radiaux que de vecteurs d'entrée dans P. Nous avons donc une couche de neurones de base radiale dans laquelle chaque neurone agit en tant que détecteur pour un vecteur d'entrée différent.

Ensuite, le seul paramètre dont dispose RBFe est la propagation des fonctions de base radiales de la première couche. Dans ce réseau, nous avons fixés la valeur du facteur de propagation à 0,3.

5.6.3.2 RBF :

Le second type crée de manière itérative un neurone radial à chaque instant. La différence est que le RBF crée un seul neurone à la fois. À chaque itération, le vecteur d'entrée, qui réussira à réduire la plupart des erreurs de réseau, est utilisé pour créer un neurone à base radiale. L'erreur du nouveau réseau est alors vérifiée si elle est suffisamment petite alors la création est terminée, sinon un neurone suivant est ajouté. Cette procédure est répétée jusqu'à ce que l'erreur quadratique moyenne ciblée soit atteinte (10e-6) ou que le nombre maximum de neurones soit atteint (300). Le nombre de neurones ajoutés entre chaque évaluation est fixé, après plusieurs expériences, à 25.

5.6.4 GRNN :

Cette variante est un autre type de NN proposé par Speckt [98]. Ce réseau basé sur la régression estime la valeur moyenne attendue de la variable de sortie en utilisant une technique bayésienne. Toute fonction continue sera approximée par une combinaison linéaire de fonctions gaussiennes. L'objectif est de faire une régression, c'est-à-dire de construire une bonne approximation d'une fonction, connue par un nombre limité de points expérimentaux.

Ce réseau peut être utilisé pour résoudre le problème de classification. Pour chaque entrée, la première couche calcule les distances entre le vecteur d'entrée et le vecteur de pondération, puis produit un vecteur qui sera multiplié par le biais.

5.6.5 PNN :

Le réseau de neurones probabiliste introduit par Donald Specht en 1988 est un réseau à anticipation à trois couches, utilisé pour la classification des données. Contrairement aux autres réseaux de neurones, basés sur la méthode de la rétropropagation, le PNN est basé sur la stratégie de décision de Bayes et la probabilité d'estimation de la densité. Le PNN utilise des fonctions gaussiennes sphériques à base radiale centrées dans chaque vecteur d'apprentissage. La probabilité d'appartenance d'un vecteur dans une classe donnée est exprimée comme suit [99] :

$$\mathrm{f}_i(\mathrm{x}) = \frac{1}{2\pi^{p/2}\sigma^p}\frac{1}{m_i}\sum_{j=1}^{m}\exp\left[-\frac{(X - X_{ij})^t(X - X_{ij})}{2\sigma^2}\right] \qquad \text{V.1}$$

Où :

i est le nombre de classes (28 ou 48 dans notre cas),

j le nombre de formes à reconnaître,

m_i le numéro du vecteur d'apprentissage de la i-eme classe,

x est un vecteur de test,

X_{ij} est le j-eme vecteur d'apprentissage de la i-eme classe,

p est la dimension du vecteur x, σ l'écart-type, et $\mathrm{f}_i(\mathrm{x})$ est la somme des fonctions multi-variables sphériques gaussiennes, centrées sur chaque vecteur d'apprentissage X_{ij} utilisé pour l'évaluation de la fonction de densité de probabilité de la i-eme classe.

Les décisions de classification sont prises conformément à la décision de la règle Bayes [99] :

$$d(x) = C_i \quad if \quad \mathrm{f}_i(\mathrm{x}) > \mathrm{f}_k(\mathrm{x}) \quad \text{for} \quad \mathrm{k} \neq \mathrm{i}$$

Où C_i est la i-eme classe. Lorsqu'une entrée est présentée ;

• La première couche calcule les distances entre le vecteur d'entrée et tous les vecteurs d'apprentissage et produit un vecteur avec des éléments indiquant comment ce vecteur d'entrée est proche de chaque vecteur d'apprentissage.

• La deuxième couche ajoute ces contributions pour chaque classe d'entrées afin de produire un vecteur de probabilité à la sortie du réseau. Enfin, une fonction de transfert à la sortie de la deuxième couche sélectionne le maximum de ces probabilités, et affecte "1" à la classe correspondante et "0" aux autres.

5.6.6 Deep Belief Network :

Les machines Boltzmann restreintes empilées de manière générative représentent un type particulier de réseau de neurones profonds, raison pour laquelle nous avons précédemment détaillé la structure RBM (chapitre 4). La topologie d'un tel réseau est présentée à la figure V.8.

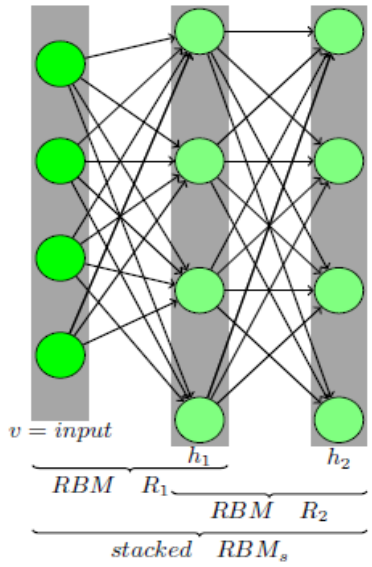


Figure V.8 : Topologie d'un réseau DBN : Empilement de RBM restreintes.

L'apprentissage du RBM empilé se fait couche par couche, de manière gourmande (gloutonne). Un premier RBM est formé sur l'ensemble de données d'échantillons afin de minimiser la contrastive divergence (CD). Ensuite, chacun des RBM suivants effectue son apprentissage sur les représentations cachées du RBM précédent. Le processus d'apprentissage comprend deux phases :

(1.) Pré-apprentissage non supervisé : le réseau DBN se distingue des autres réseaux de neurones par l'apprentissage couche par couche. L'idée est de former chaque couche en tant que RBM. L'apprentissage commence à partir des couches cachées les plus basses (proches du vecteur d'entrée visible) et progresse vers le vecteur de sortie. Ainsi, un DBN apprend à reconstruire de façon probabiliste ses entrées. Les couches font alors office de détecteurs de caractéristiques.

L'algorithme d'apprentissage DBN pour chaque couche RBM est présenté comme suit :

Étape 1 : créez un réseau multicouche à partir des machines Boltzmann restreintes (RBM), puis entraînez les couches à l'aide de l'algorithme de Greedy-wise. Naturellement, le vecteur d'entrée est le vecteur visible v.

Étape 2 : encodez X en tant que RBM pour produire un nouvel échantillon v_0, en utilisant la méthode d'échantillonnage de Gibbs.

Étape 3 : répétez la deuxième étape avec v - v_0 pour chacune des deux couches successives jusqu'à atteindre les deux couches supérieures du réseau.

(2.) Apprentissage supervisé : en utilisant d'un algorithme supervisé, tel que l'algorithme de rétro-propagation dans MLP, qui permet de rechercher les paramètres optimaux du réseau DBN.

5.7 Description de la base de données :

L'objectif des architectures décrites précédemment est la classification et la reconnaissance des caractères / mots arabes issus de notre base de données NOUN-DATABASE v2, construite à partir d'une acquisition en ligne au moyen d'une tablette d'acquisition. Ce choix est motivé par le fait que la plupart des bases de données existantes sont hors ligne, c'est-à-dire contenant des

images. Dans ce travail, nous voulions gérer le signal d'écriture en ligne plutôt que les images. L'alphabet arabe se compose de 28 lettres de formes variables en fonction de sa position dans le mot. Notre base de données contient 2800 caractères arabes écrits par 20 scripteurs différents, chaque auteur insère 5 fois l'alphabet (une fois pour chaque position dans le mot : sous sa forme isolée, au début, au milieu, à la fin et une autre fois selon le choix de scripteur). La base de données construite comprend aussi 4800 mots, saisis par les mêmes scripteurs (20). Les mots représentent les 48 villes Algériennes (wilayas). Chaque auteur effectue cinq insertions des 48 wilayas. L'écriture manuscrite peut être influencée par plusieurs paramètres tels que l'âge, le sexe et l'état du scripteur (chaque auteur possède un style d'écriture approprié). Pour préserver la variabilité des signaux d'écriture, nous avons pris soin de faire l'acquisition des données (la base de données des échantillons) par 20 personnes de sexe et d'âge différents. Le tableau V.1 donne un aperçu sur la base de données NOUN v2.

Tableau V.1 : Description de la base de données utilisée

Base de données	Nombre de classes	Nombre de scripteurs	Echantillons d'apprentissage/test
Classe des caractères	28	20	1400/1400
Classe des mots	48		2400/2400

5.8 Méthodologie du test :

L'apprentissage est stoppé lorsque l'erreur (EQM) devient minimum (10^{-6}), ou bien le nombre d'itération fixé atteint le maximum. Ensuite le réseau est évalué à partir des données différentes de celles utilisées durant l'apprentissage. Par définition ce dernier jeu de données est appelé ensemble de test.

Le pas du gradient ainsi que le pas d'apprentissage sont des facteurs déterminant dans la vitesse de convergence du réseau de neurones. Le temps d'apprentissage croissant très vite avec la complexité du réseau, il faut donc trouver un pas optimal. Plus le pas est petit, plus le nombre d'itérations de la base d'apprentissage sera important. Tandis que plus le pas est grand, plus le nombre d'itérations nécessaire sera faible mais le réseau risque de diverger. Dans le cadre de notre expérience, nous avons fixé le pas d'apprentissage à 0,01 et le nombre maximum d'itérations à 1000.

Nous allons dans ce qui suit explorer les taches de reconnaissance du manuscrit arabe en commençant par les caractères avant de passer aux mots. La variété des formes des traits compliquent la tâche de reconnaissance. Comme expliqué dans la section 5.7, la base de données regroupes les caractères arabes quelques soit leur position dans le mot c-a-d au début, milieu, fin et isolé. D'autre part, elle regroupe aussi les mots qui représentent les 48 wilayas de l'Algérie. Les architectures utilisées sont dans l'ordre : MLP, TDNN, RBFe, RBF, PNN, GRNN et DBN. Les paramètres de chaque réseau de neurones (mentionnés précédemment), sont ceux qui nous ont donnés les meilleurs résultats. Nous noterons aussi que nous avons utilisés deux principales métriques :

- Temps nécessaires à la classification estimés en secondes
- Taux de reconnaissance (%)

Expérience 1 :

Dans la première expérience, nous avons utilisé les architectures classiques (peu profondes) décrites précédemment pour la reconnaissance de caractères arabes en ligne, quelle que soit leur position dans le mot, c'est-à-dire sous sa forme isolée, au début, au milieu et à la fin. Les meilleures configurations de notre système ont donné les résultats illustrés dans le tableau V.2.

Tableau V.2 : Taux de reconnaissance des caractères Arabes

Réseau de neurones (classique)	Caractères Arabes		Temps nécessaire à la classification (sec)
	% correct	% erreur	
MLP	**97.73**	**2.27**	48.21
TDNN	49.46	50.54	2358.00
DTDNN	55.29	44.71	1256.00
RBF	86.09	13.91	23.09
RBFe	75.39	24.61	8.68
GRNN	86.36	13.64	**8.14**
PNN	84.92	15.05	8.73

Discussion : Nous remarquons que les architectures classiques utilisées donnent des résultats plus ou moins satisfaisants. Cela dit, les meilleurs résultats de classification des caractères arabes ont été obtenus en utilisant le perceptron multicouche MLP avec un taux de reconnaissance de 97,73%, ceci dit le temps nécessaire à la classification est 6 fois plus important que celui réalisé dans le GRNN dont les résultats sont aussi très satisfaisants.

Expérience 2 :

Dans la deuxième expérience, nous allons utiliser les mêmes architectures précédentes pour la reconnaissance de mots arabes en ligne, représentée par les 48 wilayas d'Algérie. Les meilleures configurations de notre système ont donné les résultats illustrés dans le tableau V.3.

Tableau V.3 : Taux de reconnaissance des mots Arabes

Réseau de neurones (classique)	Mots Arabes		Temps nécessaire à la classification (sec)
	% correct	% erreur	
MLP	**85.31**	**14.69**	27.01
TDNN	46.76	53.24	32017.00
DTDNN	49.71	51.29	25209.00
RBF	81.85	18.15	37.57
RBFe	62.58	37.42	**16.38**
GRNN	78.44	21.56	16.74
PNN	78.44	21.56	17.77

Discussion :

Nous constatons que les taux de reconnaissance de mots ont considérablement chuté par rapport au tableau V.2, ceci peut être expliqué par de l'augmentation du nombre de classes (qui passe de 28 à 48) d'une part. D'autre part, Cela s'explique aussi par la complexité des échantillons de données. En effet, un mot peut être composé de plusieurs pseudo-mots (comme expliqué dans le chapitre 2 section II.5.2) pouvant contenir un ou plusieurs caractères. Cependant, la

variabilité du signal d'écriture est plus riche dans le cas de l'écriture d'un mot. De plus, la dégradation des taux de reconnaissance peut s'expliquer par la principale limitation des architectures classiques. Si l'architecture est trop profonde, l'optimisation des paramètres conduit très souvent à un des minimas locaux non optimaux.

L'augmentation du nombre de couches n'a pas amélioré les résultats (elle les a parfois dégradés). Pour cette raison, nous avions précédemment souligné que les résultats mentionnés étaient ceux de l'architecture idéale (avec les meilleurs paramètres pour chaque réseau de neurones, y compris le nombre de couches). Comme expliqué dans [97, 100], la surface de l'erreur n'est pas convexe et est de plus en plus irrégulière quand le réseau est profond. Cependant, le MLP donne toujours les meilleurs résultats, soit 85,31% avec un temps de classification acceptable de 27,01 secondes.

Malgré la notion de poids partagés qui permet de réduire le nombre de paramètres du réseau et d'optimiser l'espace mémoire, les réseaux de neurones convolutifs (TDNN) nécessitent plus de temps de généralisation car leur architecture est beaucoup plus complexe que celle du MLP du fait qu'ils contiennent une dimension temporelle. Nous pensons que plus un réseau est simple, plus ses performances sont élevées. Avec le TDNN et le DTDNN, le caractère temporel des données est exploité par le système de reconnaissance, ce qui permet souvent de lever les ambiguïtés et d'identifier plus facilement certains caractères. Inversement, certaines planifications temporelles sont perturbatrices, en particulier les signes diacritiques ou les retouches effectuées sur un tracé, ce qui explique pourquoi la performance des deux réseaux de neurones à convolution est inférieure à celle du réseau MLP.

Pour remédier à cette dégradation des performances, nous allons utiliser une architecture Deep learning à travers une de ces variétés qui est le Deep Belief Network. Ce réseau a été détaillé dans le chapitre 4, mais nous allons le reprendre dans le paragraphe suivant pour les besoins de l'expérience suivante.

Expérience 3 :

Nous allons utilisé une architecture neuronale profonde du type Deep Belief Network (DBN), qui représente un empilement de machines Boltzmann restreintes. Il convient de noter que le système d'architecture de reconnaissance est légèrement modifié car nous avons supprimé le bloc «Extraction de caractéristiques» (voir la figure V.1). Cela est dû à la capacité d'abstraction du réseau deep learning. En fait, cela permet d'apprendre automatiquement les différentes représentations de données par un niveau d'abstraction à partir des données brutes.

Phase 1 : Pré-apprentissage (Apprentissage non supervisée sans rétro-propagation,)

Dans cette phase, l'algorithme utilise une concaténation des couches RBM pour apprendre la distribution des données d'entrée X (signal d'écriture manuscrite) sans prendre en compte les étiquettes Y. Chaque couche RBM (les deux premières couches dans ce cas) a une représentation plus abstraite que la précédente. Hinton et al. [82] ont prouvé qu'une méthode d'apprentissage gloutonne pour un apprentissage non supervisée était efficace. Cette méthode s'appelle aussi contrastive divergence (CD). Cette phase, appelée parfois pré-apprentissage, peut être considérée comme une initialisation efficace du DBN.

Phase 2 : Fine-tuning (Apprentissage supervisé avec rétro-propagation) Le rôle de la seconde partie du DBN est de convertir la représentation abstraite X en étiquettes Y utilisées dans le cas

de l'apprentissage supervisé (la dernière couche RBM avec 2000 unités). L'algorithme de rétro-propagation est utilisé pour réajuster les paramètres du réseau et, finalement, le réseau optimal global pourrait être obtenu. La couche de sortie a le nombre de classes (48 dans ce cas). La phase d'apprentissage est considérée comme accomplie une fois que l'erreur entre la sortie et celle désirée est stabilisée.

Ensuite, nous testons d'autres échantillons de la base de données (voir tableau V.1). Cette phase, appelée fine-tuning, prend plus de temps que la précédente. Les paramètres du réseau DBN utilisé pour chaque couche (les deux phases) sont mentionnés dans le tableau suivant :

Tableau V.4 : Paramètres du DBN.

Paramètre	Valeur
Nombre de couches cachées	2
Unité de la première couche cachée	100
Unité de la seconde couche cachée	100
Unité de la troisième couche cachée	2000
Batch size	100
Nombre d'itérations du RBM	50

L'architecture du réseau DBN décrite ci-dessus a donné les résultats suivants mentionnés dans le tableau suivant :

Tableau V.5 : Taux de reconnaissances des mots arabes en utilisant le DBN.

Réseau de neurones (profond)	Mots Arabes			Temps nécessaire à la classification (sec)
	% erreur avant BP	% erreur après BP	% correct	
DBN	33.75	2.92	**97.08**	252.21

Discussion :

Nous pouvons constater que les résultats de la classification de mots se sont considérablement améliorés avec l'utilisation du DBN. En effet, dans le tableau précédent, le meilleur taux de classification était de 85,31% avec le MLP. Il atteint un taux de 97,08% avec le réseau DBN à empilement de RBMs, ce qui donne un gain de 11,77%. En outre, nous avons noté que la deuxième étape, la partie supervisée (après la rétro-propagation), appelée fine-tuning, est très importante en termes de réduction du taux d'erreur (2,92% de taux d'erreur), elle s'est avérée être beaucoup plus efficace que l'étape de pré-apprentissage (33,75% de taux d'erreur). Cela s'explique par le fait que nous ne partons pas d'une solution initiale aléatoire.

La figure V.9 représente un résumé des taux de classification obtenus pour chaque réseau de neurones utilisé.

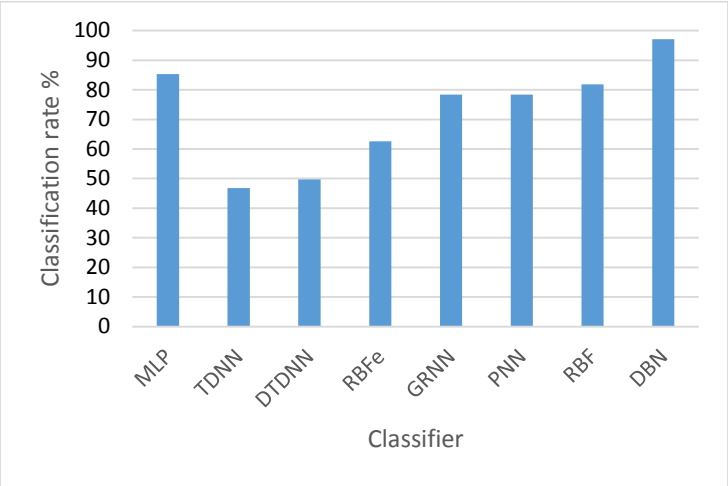


Figure V.9 : Récapitulatif des taux de reconnaissance des mots arabes.

V.9 Conclusion

Dans ce chapitre, nous avons proposé une approche neuronale visant à développer une solution basée sur les réseaux de neurones pour la reconnaissance en ligne des manuscrits arabes acquis dynamiquement. Dans le but de réaliser la reconnaissance en ligne, nous avons construit notre propre base de données, contenant 2800 caractères et 4800 mots. Nous avons divisé notre travail en deux parties ; la première présente une approche neuronale classique utilisant ces différents réseaux de neurones : MLP, TDNN, DTDNN, RBFe, RBF, GRNN et PNN.

Nos expériences sur la reconnaissance des caractères ont donné des résultats intéressants, tels que 97,73% de taux de réussite pour le MLP et 86,36% pour le GRNN. Néanmoins, l'application des mêmes architectures sur la reconnaissance des mots détériore considérablement les résultats, qui ne dépassaient pas 85,31% pour le MLP et 81,85% pour le RBF. Cela s'explique par la complexité du signal d'entrée et le problème des minima locaux.

La seconde approche utilise un apprentissage en profondeur, en l'occurrence le DBN, qui représente un empilement de machines de Boltzmann restreintes et qui est caractérisé par une méthode d'apprentissage couche par couche. Ce réseau améliore considérablement les performances de la base de données des mots et donne un taux de classification de 97,08%.

En fait, le DBN a démontré sa capacité à réaliser d'excellentes performances pour les tâches de classification et de réduction de dimensions. Nous constatons ainsi la supériorité du réseau Deep learning par rapport aux réseaux de neurones classiques. Nous notons également que les différents paramètres mentionnés dans toutes les architectures de neurones précédentes nous ont permis d'obtenir les meilleurs taux de classification.

Chapitre VI
Conclusion générale

VI.1 Bilan

La thèse décrit et explique l'importance de la reconnaissance de l'écriture manuscrite et présente des recherches étroitement liées au domaine. Cette recherche a été entreprise pour concevoir un système de reconnaissance de mots arabes saisis d'une manière non-contrainte et évaluer les performances du système prototype mis au point.

Etant donnée la non-disponibilité des bases de données arabes en ligne, nous nous sommes focalisé dans un premier temps à collecter un ensemble d'échantillons de caractères et de mots pour pouvoir valider notre étude. La collecte a été faite auprès de 20 scripteurs qui ont contribué aux 2800 caractères et 4800 mots de la base de données.

Cependant, les données brutes du signal d'écriture contiennent d'importantes variations selon la vitesse d'écriture se traduisant par un nombre différent des points formant le signal. Un apprentissage sur de telles données peut engendrer de très faibles performances, raison pour laquelle nous avons intégré un bloc de prétraitement qui comprend un échantillonnage spatial du signal d'écriture dans le but de normaliser la distance entre les points ainsi qu'un recentrage de la forme obtenue. Une fois le prétraitement effectué, nous allons extraire un ensemble de caractéristiques pertinentes de la même forme afin que notre système de reconnaissance puisse apprendre les caractéristiques de tous les échantillons de la base de données du training. Généralement la partie consacrée à l'apprentissage contient entre 60 et 80 % de l'ensemble des échantillons, nous avons diminué cette valeur pour augmenter la complexité au système de reconnaissance.

Les résultats de cette étude indiquent que l'utilisation de la base de données de caractères donne des résultats satisfaisants lorsqu'on fait appel aux réseaux de neurones classiques peu profonds (MLP, TDNN, RBF...) développés dans le chapitre 3. Par conséquent, nous pouvons conclure que les caractères arabes manuscrits en ligne peuvent être reconnus pour être lisibles par la machine. Ceci dit, l'application des mêmes méthodes pour la reconnaissance des mots altère sensiblement les taux de reconnaissance.

L'approche neuronale profonde est réalisée à l'aide du Deep Belief Network. En effet pour faire face à cette dégradation des performances nous avons pensé à l'apprentissage profond. Le réseau DBN détaillé dans le chapitre 4, par son architecture composé de deux parties, la première non supervisée qui représente une initialisation des poids du réseau permettant (supprimer l'aspect aléatoire) de minimiser l'intervention manuelle de l'utilisateur en ne spécifiant pas le nombre de classes d'une part, comme ça permet aussi de faire l'impasse sur le bloc d'extraction des caractéristiques car c'est le réseau lui-même qui effectue cette tâche. L'implémentation du DBN donne des résultats très intéressants avoisinant les 97% pour la reconnaissance des mots arabes.

VI.2 Perspectives

Étudier une problématique de reconnaissance du manuscrit arabe est un défi particulier surtout via la technologie en-ligne, sachant que peu de travaux de recherches ni d'applications commercialisées dans ce sens ont vu le jour et que le traitement du manuscrit arabe reste un problème pas du tout résolu.

L'engouement actuel pour le deep learning nous pousse à persévérer encore dans ces travaux, beaucoup d'expériences peuvent être effectuées pour comprendre encore l'impact des différents paramètres sur les performances du système. De plus, nous espérons aussi explorer les autres architectures profondes : les CNN et les auto-encodeurs, par conséquent nous pensons qu'une étude comparative s'impose sur une base de données plus importante.

Enfin, nous comptons aussi intégrer cette partie (reconnaissance de mots) dans des thèmes plus élargis et plus complexes tels que la reconnaissance de phrases en utilisant les contraintes contextuelles et lexicales de la langue arabe.

Bibliographie

[1] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition : A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, 2000.

[2] Plötz, Thomas, and Gernot A. Fink. Markov Models for Offline Handwriting Recognition : A Survey. *International Journal on Document Analysis and Recognition (IJDAR)* 12 (4) : 269-298, 2009. doi:10.1007/s10032-009-0098-4.

[3]K. Hallouli. Reconnaissance de caractères par méthodes markoviennes et réseaux bayésiens, *Thèse de l'École nationale supérieure des télécommunications*, Paris mai 2004.

[4] S.Benbakreti. Extraction des Caractéristiques et Reconnaissance en Ligne de Caractères Arabes Isolés. *Thèse de l'Institut National des Télécommunications et des Technologies de l'Information et de la communication,* Oran juin 2012.

[5] Qian, J., W. Wang, and D. Wang. A novel approach for online handwriting recognition of Tibetan characters. *In Proceedings of the International MultiConference of Engineers and Computer Scientists*. Vol I, 2010.

[6] Santosh, K. and C. Nattee. A comprehensive survey on on-line handwriting recognition technology and its real application to the Nepalese natural handwriting. *Journal of Science, Engineering, and Technology*, **5**(I) : p. 31-55, 2009.

[7] Read, J.C., S. MacFarlane, and C. Casey. A comparison of two on-line handwriting recognition methods for unconstrained text entry by children. *Proceedings of BCS British-HCI*, vol 2, p. 29-32. 2003

[8] Peter Burrow. Arabic handwriting recognition. *Report of Master of Science School of Informatics, University of Edinburgh*, 2004.

[9] Eugene Borovikov. A survey of modern optical character recognition techniques. 2004.

[10] Claus Bahlmann. Advanced Sequence Classification Techniques Applied to Online Handwriting Recognition. *Thesis of University of Freiburg*, 2005.

[11] Kumara, K., R. Agrawal, and C. Bhattacharyya, A large margin approach for writer independent online handwriting classification. *Pattern Recognition Letters*. 29(7) : p. 933-937, 2008.

[12] M. Al-Ammar, R. Al-Majed, and H. Aboalsamh. Online Handwriting Recognition for the Arabic Letter Set. *In Proceedings of the 5th WSEAS international conference on Communications and information technology*, Corfu Island, Greece, 14-17 July 2011.

[13] J. Sternby, J. Morwing, J. Andersson, and C. Friberg. On-line Arabic handwriting recognition with templates. *Pattern Recognition*, vol. 42, no. 12, December 2009.

[14] M. Nakagawa and K. Matsumoto. Collection of On-line Handwritten Japanese Character Pattern Databases and their Analysis. *International Journal on Document Analysis and Recognition*, vol. 7, no. 1, 2004.

[15] H. Shu. On-line Handwriting Recognition Using Hidden Markov Models. *Master's thesis of Massachusetts Institute of Technology*, 1996.

[16] Liwicki, M., et al. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. *In Proceedings 9th International Conference on Document Analysis and Recognition*. 2007.

[17] Chen, Q., et al. A Medical Knowledge Based Postprocessing Approach for Doctor's Handwriting Recognition. *In International Conference Frontiers in Handwriting Recognition (ICFHR)*, 2010.

[18] Halavati, R., M. Jamzad, and M. Soleymani. A novel approach to persian online hand writing recognition. *Trans. Engin. Technol*. Vol 6 : p. 1305-5313, 2005.

[19] Oh, J. An On-Line Handwriting Recognizer with Fisher Matching, Hypotheses Propagation Network and Context Constraint Models. *Thesis in New York University*, 2001.

[20] N. Mezghani, A. Mitiche, and M. Cheriet. On-line recognition of handwritten Arabic characters using a kohonen neural network. *In Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition*, page 490, Washington DC, 2002.

[21] B. Alsallakh and H. Safadi. AraPen : An Arabic Online Handwriting Recognition System. *In Proceedings of 2nd IEEE International Conference on Information & Communication Technologies: from Theory to Applications (ICTTA)*, vol. 1, pp. 1844–1849, Damascus, 24 - 28 April 2006.

[22] G. Al-Habian and K. Assaleh. Online Arabic Handwriting Recognitionusing Continuous Gaussian Mixture HMMS. *In Proceeding of International Conference on Intelligent Advanced System (ICIAS)*, vol. 1, pp. 1183 – 1186, Kuala Lumpur, Malaysia, 25-28 Nov 2007.

[23] F. Biadsy, J. El-sana, and N. Habash. Online Arabic handwriting recognition using hidden markov models. *In Proceeding of 10th International Workshop on Frontiers of Handwriting Recognition*, La Baule, France, 23-26 Oct 2006.

[24] H. Ahmed and S. A. Azeem. On-line Arabic Handwriting Recognition System based on HMM. *In Proceeding of 11th International Conference on Document Analysis and Recognition (ICDAR)*, pp 1324 – 1328, Beijing, China, 18-21 September 2011.

[25] C. M. Bishop. Pattern Recognition and Machine Learning. Springer, 2007.

[26] Rigoll, G., et al. A comparison between continuous and discrete density hidden Markov models for cursive handwriting recognition. In Proceedings of the 13th International Conference of Pattern Recognition, vol 2, 1996.

[27] G. F. Luger. Artificial Intelligence : Structures and Strategies for Complex Problem Solving. Pearson Education Limited, Harlow, England, Fifth Edition, 2005.

[28] Huang, B. and M. Kechadi. A Structural Analysis Approach for Online Handwritten Mathematical Expressions. *International Journal of Computer Science and Network Security (JCSNS)*, vol.7 No.7, pp 47-56, July 2007.

[29] Arora, A. and A.M. Namboodiri. A hybrid model for recognition of online handwriting in indian scripts. *In International Conference of Frontiers in Handwriting Recognition (ICFHR)*, 2010.

[30] A. K. Jain, J. Mao, and K. M. Mohiuddin. Artificial Neural Networks: A Tutorial. *IEEE Computer*, vol. 29, no. 3, pp. 31-44, Mar 1996.

[31] X. Jiang and A. H. K. S.Wah. Constructing and training feed-forward neural networks for pattern classification. *Pattern Recognition*, vol. 36, no. 4, pp. 853–867, April 2002.

[32] Lewis, M. Paul, Gary F. Simons, and Charles D. Fennig (eds.). n.d. Ethnologue : Languages of the World, Dallas, Texas: SIL International. Online Version : Http://www.ethnologue.com [Accessed:03-Dec-2015].

[33] UNESCO. 2015. World Arabic Language Day. Accessed December 3. http://www.unesco.org/new/en/unesco/events/prizes-and-celebrations/celebrations/international-days/world-arabic-language-day/.

[34] Abulhab, Saad D. Roots of Modern Arabic Script : From Musnad to Jazm. Sawt Dahesh. *Published in two parts by the New York based quarterly Journal Dahesh Voice*, pp 50-51, 2007-2009.

[35] M.Pechwitz, S.S.Maddouri, V.Märgner, N.Ellouze, and H.Amiri. IFN/ENIT - Database of Handwritten Arabic Words. *In 7th, Colloque International Francophone sur l'Ecrit et le Document*, (CIFED), pp : 129--136. Hammamet, Tunis, 2002.

[36] Elarian, Yousef S., Irfan Ahmad, Sameh M. Awaida, Wasfi G. Al-Khatib, and Abdelmalek Zidouri. Arabic Ligatures : Analysis and Application in Text Recognition. *In Proceedings of the 13th International Conference on Document Analysis and Recognition* (ICDAR), pp 896-900, 2015.

[37] International Unipen Foundation. The UNIPEN Project. http://www.unipen.org, 1994.

[38] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. UNIPEN project of on-line data exchange and recognizer benchmarks. *In Proceedings of International Conference on Pattern Recognition* (ICPR), vol. 2, pp. 29–33, Jerusalem, Israel, 9-13 Oct 1994.

[39] E. H. Ratzlaff. Methods, report and survey for the comparison of diverse isolated character recognition results on the unipen database. *Document Analysis and Recognition journal*, vol. 1, n°. 2, p. 623-628, 2003.

[40] M. Parizeau, A. Lemieux, et C. Gagné, "Character recognition experiments using UNIPEN data," in Proceedings. Sixth International Conference on Document Analysis and Recognition, Seatle, p. 481–485, 2001.

[41] R. Tlemsani et A. Benyettou. On Line Isolated Characters Recognition Using Dynamic Bayesian Networks. *International Arab Journal of Information Technology*, vol. 08, n°. 04, Oct. 2011

[42] M. Agrawal, K. Bali, S. Madhvanath, and L. Vuurpijl. UPX: a new XML representation for annotated datasets of online handwriting data. *In Proceedings of the Eighth International Conference on Document Analysis and Recognition*, vol. 2, Pages 1161–1165, 29 Aug–1 Sep 2005.

[43] A. S. Bhaskarabhatla, M. P. Kumar, A. Balasubramanian, C. Jawahar, and S. Madhvanath. Representation and Annotation of Online Handwritten Data. *In Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition* (IWFHR-9), Tokyo, Japan, October 2004.

[44] W3C Multi-modal Interaction Working Group. Ink Markup Language (InkML). http://www.w3.org/2002/mmi/ink, 2003

[45] A. S. Bhaskarabhatla and S. Madhvanath. An XML Representation for Annotated Handwriting Datasets for Online Handwriting Recognition. *In Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal, May 2004.

[46] S. Al-Emami and M. Usher. On-line recognition of handwritten Arabic characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 704–710, July 1990.

[47] A. M. Alimi. An evolutionary neuro-fuzzy approach to recognize on-line Arabic handwriting. *In Proceedings of the 4th International Conference Document Analysis and Recognition*, pages 382–386, 1997.

[48] A. Amin, A. Kaced, P. J. Haton, and R. Mohr. Handwritten Arabic Character Recognition by the IRAC System. *In Proceedings of the International Conference of Pattern Recognition*, Miami, Florida, USA, pp. 729-731, 1980.

[49] M. S. El-Wakil and A. A. Shoukry. On-Line Recognition of Handwritten Isolated Arabic Characters. *Pattern Recognition*, vol. 22, no. 2, pp. 97-105, 1989.

[50] A. M. Alimi and A. O. Ghorbel. Error Analysis in an On-Line Recognition System of Arabic Handwritten Characters. *In Proceedings of the International Conference of Document Analysis and Recognition* (ICDAR), Montreal, Canada, pp. 671-674, Aug 1995.

[51] A. Amin. Machine Recognition of Handwritten Arabic Words by the IRAC I1 System. *In Proceedings of the 6th International Joint Conference on Pattern Recognition*, Munich, Germany, pp. 34-36, Oct 1982.

[52] A. Amin and G. Masini. Machine Recognition of Cursive Arabic Words. *Application of Digital Image Processing IV*, SPIE's vol. 359, San Diego, pp. 286-292, Aug 1982.

[53] A. Amin, G. Masini, and P. J. Haton. Recognition of Handwritten Arabic Words and Sentences. *In Proceedings of the 7th International Joint Conference on Pattern Recognition*, Montréal, Canada, pp. 1055-1057, Oct 1984.

[54] Abdelazeem S. and Eraqi H. On-line Arabic Handwritten Personal Names Recognition System based on HMM. In *Proceedings of the 11th international conference on document analysis and recognition* (ICDAR), Beijing, China, pp 1304-1308, September 2011.

[55] C. C. Tappert. Cursive Script Recognition by Elastic Matching. *IBM Journal of Research and Development*, vol. 26, no. 6, 1982.

[56] Abdelazim H.Y. A Hybrid Fuzzy Neural Approach to the recognition of Arabic Script. *In Proceedings of 5th International Conference and Exhibition on Multilingual Computing*, Cambridge, April 1996.

[57] Abdelazim H.Y. On Line Recognition of Arabic Cursive Scripts Using Radial Basis Function Networks. *In Proceedings of the third Conference on Language Engineering*, Cairo, EGYPT, October. 2002.

[58] Abdelazim H.Y. Recent trends in Arabic OCR. *In Proceedings of the 5th Conference of Engineering Language*, Ain Shams University, Cairo, Egypt, 2005.

[59] Eraqi H., and Abdelazeem S. An On-Line Arabic Handwriting Recognition System Based on a new Graphemes Segmentation Technique. *In Proceedings of the 11th international conference on document analysis and recognition* (ICDAR), Beijing, China, September 18-21, pp 409-413, 2011.

[60] H. El-Abed, V. Margner, M. Kherallah, and A. M. Alimi. Online Arabic handwriting recognition competition. *In Proceedings of 10th International Conference on Document Analysis and Recognition* (ICDAR), Barcelona, pp. 1383 – 1387, 26-29 July 2009.

[61] Elanwar R. I., Rashwan M. A., and Mashali S. A. Simultaneous segmentation and recognition of Arabic characters in an unconstrained on-line cursive handwritten document. *In Proceedings of International conference on Machine learning and Pattern Recognition* (MLPR)2007, Vol. 23, pp. 288-291, August 2007.

[62] Jurafski D., and Martin J. H. Speech and Language Processing : An introduction to Natural Language processing, computational Linguistic and Speech recognition, *Prentice-Hall*, pp. 136-156, New Jersey 2000.

[63] S.Rajasekaran, G.A.Vijayalakshmi Pai. Neural Network, Fuzzy Logic and Genetic Algorithm. Prentice Hall, pp-13-20, India 2003.

[64] Hebb. Apprentissage supervisé, 1949.

[65] Tsoukalas et Uhrig. Apprentissage par renforcement, 1997.

[66] Bertsekas et Tsitsiklis. Apprentissage par renforcement, 1996

[67] Khalaf khatatneh, Ibrahiem M.M El Emary and Basem Al- Rifai. Probabilistic Artificial Neural Network For Recognizing the Arabic Hand Written Characters. *Journal of Computer Science* vol 2, N 12, pp 879-884, 2006.

[68] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems* (MCSS), vol 2 N 4, pp 303–314, 1989.

[69] Ilya Sutskever. Training Recurrent Neural Networks. PhD thesis, University of Toronto, 2012.

[70] Alexander Waibel, Toshiyuki Hanazawa, Geofrey Hinton, Kiyohiro Shikano, and Kevin J. Lang. Readings in speech recognition. chapter Phoneme Recognition Using Time-delay Neural Networks, pages 393–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[71] Moody, J. and Darken, C. Learning with Localized Receptive Fields. *In Proceedings of the 1988 Connectionist Models Summer School*, pp 133-143, San Mateo, 1989.

[72] Broomhead, D.S., Lowe, D. Multivariable functional interpolation and adaptive networks. *Complex Systems*, vol. 2, pp. 321-355, 1988.

[73] D. F. Specht. Probabilistic neural networks. *Neural networks*, vol. 3, no. 1, p. 109–118, 1990.

[74] D.F Specht. Probabilistic neural networks and the polynomial adaline as complementary techniques for classification. *Neural Networks*, vol 1 N 1, pp111-121, 1990.

[75] Oliveira E, Ciarelli PM, Souza A, Badue C, editors. Using a probabilistic neural network for a large multi-label problem. *Neural Networks*, the 10th Brazilian Symposium, 2008.

[76] Jeatrakul P, Wong K, editors. Comparing the performance of different neural networks for binary classification problems. Natural Language Processing. *The Eighth International Symposium* ,2009.

[77] Wu SG, Bao FS, Xu EY, Wang Y-X, Chang Y-F, Xiang Q-L. A leaf recognition algorithm for plant classification using probabilistic neural network. *Signal Processing and Information Technology*, IEEE International Symposium, 2007.

[78] M. W. Kim et M. Arozullah. Generalized probabilistic neural network based classifiers. *In International Joint Conference Neural Networks* (IJCNN), p. 648–653, 2002.

[79] Fung CC, Iyer V, Brown W, Wong KW. Comparing the performance of different neural networks architectures for the prediction of mineral prospectivity. *In proceedings of International Conference of Machine Learning and Cybernetics*, 2005.

[80] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. Deep Learning. http://www.deeplearningbook.org. MIT Press, 2016.

[81] Cho, Youngmin and Lawrence K Saul. Kernel methods for deep learning. *In Advances in neural information processing systems*, pp. 342–350, 2009.

[82] Hinton, Geoffrey E, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *In: Neural computation*, vol 18, N 7, pp. 1527– 1554, 2006.

[83] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[84] Hubel, D. and Wiesel, T. Receptive fields and functional architecture of monkey striate cortex. *In: Journal of Physiology*, vol 195, Issue 1, pp 215-243, 1968.

[85] Clark, Christopher and Storkey, Amos. Teaching Deep Convolutional Neural Networks to Play Go, 2014.

[86] Collobert, Ronan and Weston, Jason. A unified architecture for natural language processing: Deep neural networks with multitask learning. *In : Proceedings of the 25th international conference on Machine learning*, pp. 160–167, 2008.

[87] Smolensky, Paul, Rumelhart, David E. and McLelland, James L. Chapter 6: Information Processing in Dynamical Systems: Foundations of Harmony Theory. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations, 1986. MIT Press. pp. 194–281.

[88] Hinton, G. E., & Sejnowski, T. J. Optimal perceptual inference. *InProceedings of the IEEE conference on Computer Vision and Pattern Recognition*, Piscataway, pp. 448-453, 1983.

[89] Salakhutdinov, R., & Hinton, G. E. Deep boltzmann machines. *In Proceedings of the international conference on artificial intelligence and statistics*, Vol. 5, No. 2, pp. 448-455. Cambridge, MA: MIT Press, 2009.

[90] Walsh, B. Markov chain Monte Carlo and Gibbs Sampling, 2004.

[91] Hinton, G. E. Training products of experts by minimizing contrastive divergence. *Neural computation*, vol 14, N 8, pp 1771-1800, 2002.

[92] Carreira-Perpinan, M. A., & Hinton, G. E. On contrastive divergence learning. *In Artificial Intelligence and Statistics*, vol. 2005, p. 17, 2005.

[93] V Nair and G. Hinton. Rectified linear units improve restricted boltzmann machines. *In Proceedings of the 27 th International Conference on Machine Learning* (ICML), Haifa, Israel, no. 3, pp 807-814, 2010.

[94] Varga, Tamás, and Horst Bunke. Machine Learning in Document Analysis and Recognition. Edited by Simone Marinai and Hiromichi Fujisawa. Vol. 90. Studies in Computational Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008..

[95] H. Boubaker, A. Chaabouni, M. Kherallah, A. M. Alimi, and H. El-Abed. Fuzzy Segmentation and Graphemes Modeling for Online Arabic Handwriting Recognition. *In Proceeding of 12th International Conference on Frontiers in Handwriting Recognition*, Kolkata, India, 16-18 Nov. 2010.

[96] H. Boubaker, A. El-Baati, M. Kherallah, and A. M. Alimi. Online Arabic Handwriting Modeling System based on the Graphemes Segmentation. *In Proceeding of 20th International Conference on Pattern Recognition* (ICPR), vol. 20, pp. 2061–2064, Istanbul, Turkey, 23-26 August 2010.

[97]E. Dumitru, M. Pierre-Antoine, B. Yoshua, et al. The difficulty of training deep architectures and the effect of unsupervised pre-training. *In International Conference on artificial intelligence and statistics*, pp. 153–160. 2009.

[98] D. Speckt. A generalized regression neural network. *IEEE Transactions on Neural Network*s, vol 2, N 6, pp : 568–576, 1991.

[99] W. Tomasz, J. Jacek, M. Jacek. Probabilistic neural network for direction estimation. *In Proceedings of the Third Conference Neural Networks and their Applications and Summer School on Neural Networks Applications to Signal Processing*, Kule, pp. 173–178. Eds R. Tadeusiewicz, L. Rutkowski, J. Chojcan. Cz¸estochowa: Pol. Neural Netw. Soc. s, 1997.

[100] I. Guyon, P. Albrecht, I. Le Cun, al. Design of a neural network character recognizer for a touch terminal. *Pattern Recognition*, vol 24, N 2, pp :105–119, 1991.

Résumé :

La principale motivation de cette thèse est d'étudier la langue arabe vis à vis la technologie en ligne dans le domaine de la reconnaissance de l'écriture manuscrite des caractères et des mots. En effet, le système proposé s'appuie sur les réseaux de neurones pour différencier les classes saisies par le scripteur. Ceci-dit, il est nécessaire d'effectuer un prétraitement (algorithme d'échantillonnage et de recentrage) pour optimiser les taux de reconnaissance. Par la suite, nous faisons une extraction des caractéristiques du manuscrit arabe afin que les différents réseaux de neurones proposés puissent effectuer l'apprentissage. En raison de l'indisponibilité des bases de données arabes à caractère en ligne, nous avons conçu notre propre base de données intitulée NOUN-DATABSE v2. La phase de test a montré que le système proposé jumelé au réseau de neurones à apprentissage profond (DBN) permet de récupérer d'excellents résultats.

Abstract :

The main motivation of this thesis is to study the online technology of Arabic language handwriting recognition of both characters and words. The proposed system relies on neural networks to distinguish between the classes that are provided by the writer. This means that it is necessary to perform a preprocessing (sampling algorithm and recentring) to optimize the recognition rates. Then, we should extract the characteristics of the Arabic manuscript so that the proposed neural networks can perform their learning. Due to the lack of Arabic on-line database, we have designed our own database called NOUN-DATABSE v2. The experiments showed that the proposed system combined with the deep learning neural network (DBN) gave the excellent results.

ملخص:

يتمثل الدافع الرئيسي لهذه الأطروحة في دراسة اللغة العربية فيما يتعلق بالتكنولوجيا الديناميكية في مجال التعرف على الكتابة اليدوية من الأحرف والكلمات. في الواقع، يعتمد النظام المقترح على الشبكات العصبية للتمييز بين الطبقات التي يدخلها الكاتب. ومع ذلك، فمن الضروري إجراء معالجة مسبقة (خوارزمية أخذ العينات وإعادة التمركز) لتحسين معدلات التعرف. بعد ذلك، نستخرج خصائص المخطوطة العربية حتى تتمكن الشبكات العصبية المختلفة المقترحة من إجراء التعلم. نظرًا لعدم توافر قواعد بيانات الخط العربي، فقد صممنا قاعدة البيانات الخاصة بنا باسم NOUN-DATABSE v2 . أظهرت مرحلة التجارب أن النظام المقترح المقترن بشبكة العصبونات العميقة للتعلم (DBN) يسمح بتحقيق نتائج ممتازة.