République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche Scientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE **E**LECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin d'Etudes
De MASTER ACADEMIQUE

Domaine : **Mathématiques et Informatique**
Filière : **Informatique**
Spécialité : **Système Informatique**

Présenté par
SEKLAOUI Wassila
AREZKI Roza

Thème

Reconnaissance de mots manuscrits en utilisant le modèle de Markov Caché : Application aux noms d'auteurs arabes

Mémoire soutenu publiquement le 17/07/ 2016 devant le jury composé de :

Président : M HABET Mohamed-Saïd

Encadreur : M SOUALAH Mohamed Ourabah

Examinateur :M KERBICHE Mohand

Examinateur : YESLI Yasmine

Remerciement

Nous tenons en premier lieu à remercier le Bon Dieu

Pour le courage et la patience qu'il nous a donné afin de mener ce projet à terme.

Nous remercions vivement Monsieur SOUALAH Mohamed Ourabah, pour toute l'aide qu'il nous a apportée.

Nos plus vifs remerciements vont aussi aux membres du jury qui ont bien voulu nous honoré en acceptant d'examiner notre travail.

Nos plus vif remerciement vont aussi à tous ceux qui on contribuer à la réalisation de ce travail et à tous ceux qui nous ont aidé à vaincre les nombreuses difficultés
rencontrer toute au long de la duré de ce mémoire.

Nous exprimons, en fin, notre infinie gratitude à nos chers parents et familles en reconnaissance de leurs sacrifices, aides, soutien et encouragement

Rosa et Wassila

« DEDICACE »

Je dédie ce modeste travail à celle qui m'a donné la vie, le symbole de Tendresse, qui s'est sacrifiée pour mon bonheur et ma réussite, à ma mère

A mon père, qui a veillé tout au long de ma vie

à m'encourager, à me donner l'aide et à me protéger.

Que dieu les gardes et les protège.

A mes frères Massinissa et Jugurtha et son fils ADAM

A mon adorable frère jumeau Ahmed

A ma sœur Karima et ses enfants ANIEL et YALLAS

A mon binôme : Wassila et sa famille.

A mes amis et amies en particuliers

« Kahina, Mohand, Nabila, hichem, adlane, dihia, krimo, nouria, célia, ouiza Katia».

A tous ceux qui me sont chères.

A tous ceux qui m'aiment.

A tous ceux que j'aime.

ROSA

« DEDICACE »

En signe de reconnaissance et de respect, je dédie ce modeste travail :

Aux êtres les plus chers à moi Mes parents qui ont toujours cru en moi et encouragée tout au long de mes études et qui ont fait de moi ce que je suis aujourd'hui que dieu les protège.

A mes chères sœurs radia, Soraya et Lamia.

A mes deux frères Nadir et Azzedine et beaux-frères Samir et Kamel.

A mes amours Lytissia, Moumouh, Cerine et Anais.

A mes cousins et cousines.

A mon binôme adorée HANANE ainsi sa Famille.

A mes amis : Houda, Celia, Sarah, Souad, Celia, Nouria, Ouiza et sans oublié Arave qui m'a aidé a réalisé ce travail.

Wassila

Sommaire

Chapitre II : l'écriture Arabe et le modèle de Markov Caché

Chapitre III : Analyse et conception

Chapitre IV : Implémentation et Réalisation

Liste des figures

INTRODUCTION GÉNÉRALE

INTRODUCTION

La reconnaissance de l'écriture manuscrite arabe est un domaine très demandé dans la recherche, elle est encore aujourd'hui au niveau d'étude et d'expérimentation, la reconnaissance elle peut cependant encore être considéré comme un problème de recherche ouvert en raison de sa variation substantielle en apparence. Avec l'introduction de modèles markoviens sur le terrain, une modélisation et une reconnaissance paradigme prometteur a été créé pour la reconnaissance de l'écriture manuscrite hors ligne.

L'objectif de notre travail est la réalisation d'un système de reconnaissance de mots arabes manuscrit. Pour ce faire, nous proposons un système basé sur une méthode analytique en utilisant le modèle de Markov caché (MMC) avec segmentation.

Le système que nous allons proposer est composé de deux sous-systèmes : un sous-système d'apprentissage et un sous-système de reconnaissance :

Le sous-système d'apprentissage a la capacité de traitement des images, et se charge aussi d'extraction des caractéristiques sous forme d'un vecteur de description, qui sera destiné à être traité par le modèle de Markov caché. Les résultats obtenus seront sauvegardés dans une base de données d'apprentissage.

Le sous-système de reconnaissance a pour objectif la reconnaissance simple, puis n-gramme du mot en utilisant l'algorithme Viterbi.

INTRODUCTION

Pour bien mené notre travail, nous avons organisé ce dernier en quatres chapitres comme suit :

Dans le premier chapitre, nous présenterons les différents aspects d'un système de reconnaissance de caractères, ensuite nous présenterons le processus de reconnaissance de l'écriture manuscrite ainsi que ces différentes approches.

Dans le deuxième chapitre, nous allons décrire les principales notions de l'écriture arabe et la méthode de reconnaissance de caractères Markov caché.

Dans le troisième chapitre, nous allons présenter la conception de l'application en nous appuyant sur une démarche de modélisation basée sur la méthode UML.et nous allons décrire le déroulement et les étapes de notre système de reconnaissance de mots.

Dans le quatrième chapitre, nous allons présenter dans ce chapitre l'environnement et les outils de développement et d'implémentation de celle-ci, ainsi que les langages de programmations qui nous ont servi d'appui pour sa réalisation. Ensuite on va présenter des différentes fonctionnalités qu'offre notre application à travers diverses interfaces.et par la fin on va donner les résultats expérimentaux des taux de segmentation et de reconnaissance

CHAPITRE I :
L'ÉTAT DE L'ART

CHAPITRE I : ETAT DE L'ART

I. Introduction

La reconnaissance de l'écriture est de transformer un texte écrit en une représentation compréhensible par une machine et facilement reproductible par un logiciel de traitement de texte. Cette tâche n'est pas triviale car les mots possèdent une infinité de représentations parce que chaque personne produit une écriture qui lui est propre, et parce qu'il existe de nombreuses polices de caractères pour l'imprimé avec de nombreux styles (gras, italique, souligné, ombré etc.) et des mises en page différentes et complexes. Suivant le type d'écriture qu'un système doit reconnaître (manuscrit, cursif ou imprimé), les opérations à effectuer et les résultats peuvent varier notablement. Notre projet s'intéresse principalement à la reconnaissance de mots manuscrits.

Nous présenterons dans ce chapitre les différents aspects d'un système de reconnaissance de caractères, ensuite nous présenterons le processus de reconnaissance de l'écriture manuscrite ainsi que ces différentes approches.

II. Définition de système de reconnaissance de caractère manuscrits

La reconnaissance de l'écriture manuscrite fait appel à la reconnaissance de forme, mais également au traitement automatique du langage naturel. Cela veut dire que le système, tout comme le cerveau humain, reconnaît des mots et des phrases existant dans un langage connu plutôt qu'une succession de caractères.

III. Généralité sur le système de reconnaissance de caractère manuscrit

Il existe plusieurs modes de classification des systèmes de reconnaissance parmi lesquels nous pouvons citer: Les systèmes de reconnaissance de caractère «en-ligne » ou « hors-ligne » suivant le mode d'acquisition, y compris Les styles d'écritures, Un système de reconnaissance de caractère Mono-scripteur, Multi-scripteur ou Omni-scripteur selon le nombre de scripteurs et un système de reconnaissance de caractère globale ou analytique selon que l'analyse s'opère sur la totalité du mot ou par segmentation en caractères. [2]

III.1. La reconnaissance En-ligne (on-line)

La reconnaissance dite en ligne s'effectue en même temps que les mots sont écrits, elle concerne les nombreux objets électroniques de poche permettant de saisir du texte sans clavier. La reconnaissance de l'écrit est généralement assurée par un appareil en ligne fonctionne avec un stylo et comprend des affichages-tablettes. Ces tablettes peuvent fournir l'ordre temporel des points qui constituent les lignes de texte.

Le mode de reconnaissance en-ligne est réservé généralement à l'écriture manuscrite, il présente un avantage majeur qui est la possibilité de correction et de modification de l'écriture de manière interactive. [2]

III.2. La reconnaissance Hors-ligne(off-line)

La reconnaissance dite hors ligne concerne tout document déjà écrit comme des formulaires, des livres, des chèques.la reconnaissance hors ligne est plus difficile, elle se retrouve souvent cantonnée à des problématiques très précises telles que la lecture d'adresses postales, de montants littéraux de chèques. La gamme de ces problèmes s'étend au fur et à mesure que la puissance des ordinateurs s'accroît. [2]

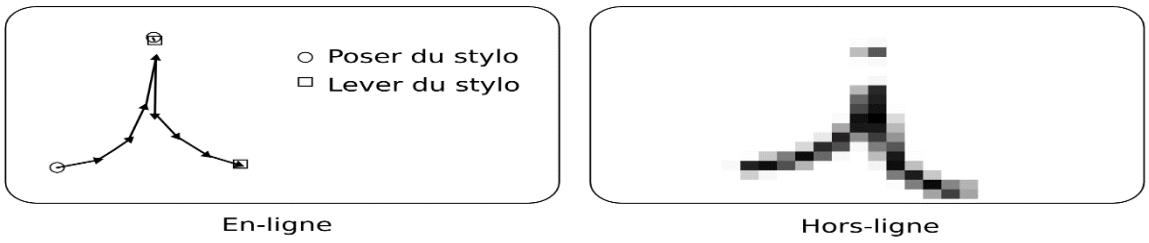


Fig. I.1 : Reconnaissance En-ligne et hors-ligne. [1]

III.3. Styles d'écriture

La difficulté de la reconnaissance est en partie liée au style d'écriture, plus l'écriture est lisible et régulière, plus la résolution est facile. Cette constatation paraît évidente mais si un lecteur humain s'en soucie rarement, les performances obtenues par des logiciels de reconnaissance varient beaucoup avec la clarté des images fournies (lisibilité, bonne résolution de l'image, lignes d'un paragraphe bien espacées, ...). On distingue trois styles d'écriture classés par ordre croissant de difficulté (figure 2.1) bien qu'ils soient parfois emmêlés :

1. L'écriture imprimée,

2. l'écriture manuscrite mais en capitales d'imprimerie ou caractères bâtons,

3. l'écriture manuscrite cursive.

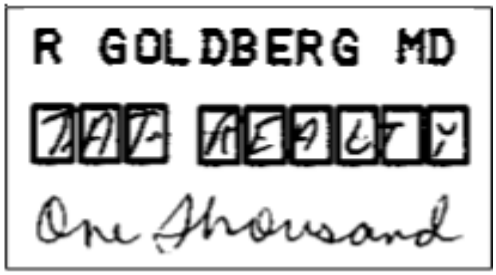


Fig. I. 2 : Trois styles d'écriture, imprimé, bâton, cursif. [2]

III.4. Un système Mono-scripteur, Multi-scripteur et Omni-scripteur

La difficulté de reconnaissance croît avec le nombre de scripteur, divisant l'échelle en trois : mono, multi et omni-scripteurs.

- **Mono-scripteur** Un système est dit mono-scripteur (propres au scripteur) si le système ne peut reconnaître qu'une seule écriture.
- **Multi-scripteur :** Un système est dit multi-scripteur (propres à l'écriture manuscrite) si le système peut identifier et reconnaître l'écriture pour un certain nombre de scripteurs.
- **Omni-scripteur :** Un système est dit omni-scripteur (propres à n'importe quelle écriture manuscrite) si le système doit être capable de généraliser son apprentissage à n'importe quel type d'écriture. [2]

III.5. Approches de reconnaissance

Il existe deux approches pour la reconnaissance de mots manuscrits : l'approche globale et l'approche analytique.

III.5.1 L'approche globale (ou approche holistique) :

Elle consiste à apprendre et reconnaître un mot dans son ensemble. Souvent utilisée pour des tâches de reconnaissance à vocabulaire restreint, elle est peu robuste face à des tâches plus difficiles où le vocabulaire contient plus d'une centaine de mots.

III.5.2 L'approche analytique

Quant à elle propose de segmenter le mot en sous-parties (caractères ou graphèmes) à modéliser. Les mots sont reconstruits ensuite par concaténation des modèles qui les composent : cela permet l'utilisation de lexiques libres, à condition qu'ils soient basés sur l'alphabet (ou les graphèmes) appris. Pour segmenter les images de mots, il est possible de procéder à un découpage explicite ou à une segmentation implicite. C'est cette dernière approche que nous avons choisie, illustrée par l'utilisation de fenêtres glissantes parcourant les images de mots dans le sens de lecture afin de les transformer en séquences. **[**4]

III.7. Processus de reconnaissance

Un système de reconnaissance fait appel généralement aux étapes suivantes : acquisition, prétraitements, segmentation, extraction des caractéristiques, classification, suivis éventuellement d'une phase de post-traitement.

III.7.1. Phase d'acquisition

L'acquisition permettant la conversion du document papier sous la forme d'une image numérique (bitmap). Cette étape est importante car elle se préoccupe de la préparation des documents à saisir, du choix et du paramétrage du matériel de saisie (scanner), ainsi que du format de stockage des images. La numérisation du document est opérée par balayage optique. Le résultat est rangé dans un fichier de points, appelés pixels, dont la taille dépend de la résolution.

III.7.2. Prétraitements

Les prétraitements appliqués sur l'image du mot permettent, d'une part, d'éliminer ou de réduire le bruit dans l'image, et d'autre part, de simplifier les traitements ultérieurs. Dans notre système nous avons utilisé la binarisation, le lissage, la normalisation, le cadrage, la squelettisation, la correction de la déformation des caractères et l'estimation de la ligne de base. [5]

III.7.2.1 Binarisation

Cette opération consiste à transformer une image avec niveau de gris en une image bi-modale (blanc et noir) composé de deux valeurs 0 et 1. [5]

III.7.2.2 Lissage

L'image des caractères peut être entachées de bruits dû aux artefacts de l'acquisition et à la qualité du document, conduisant soit à une absence de points ou à une surcharge de points. L'opération de lissage est destinée à rendre plus homogène les différentes parties de l'image, et préparer ainsi la détection des contours par l'élimination des fortes variations d'intensité lumineuse ponctuelles non significatives. Généralement, un filtre passe bas est utilisé, mais il a comme effet secondaire la diminution du contraste (rendre l'image plus floue). [5]

III.7.2.3 La normalisation

Étant donné que la taille des caractère arabes est très variable, la normalisation de la taille est souvent utilisée pour échelonner le caractère a une taille fixe et pour centrer les caractères avant leur reconnaissance. Cette opération est très utile dans les méthodes de reconnaissance qui sont sensible a petite variation dans la taille et la position. [5]

III.7.2.4 Le cadrage

L'opération de cadrage consiste à chercher la première et la dernière ligne / colonne signifiante (pixel <> de l'arrière-plan), ensuite créer une nouvelle image cadrée à partir de l'image mère. [5]

III.7.2.5 La squelettisation

La squelettisation est l'une des techniques les plus utilisées dans la reconnaissance des formes. Elle permet de diminuer l'information utile en ne gardant que le squelette de la forme. Le principe est de ramener l'image du mot à une écriture linéaire d'une épaisseur égale à un pixel, en préservant la forme, la connexité et la topologie du tracé. Nous désirons de cette étape la correction de la déformation des caractères. En effet après la squelettisation du corps des caractères, plusieurs déformations de type concave et convexe ont apparu au niveau des lignes continues (supposées sans concave ni convexe). La fig. montre quelques déformations détectées après l'étape de squelettisation. [5]

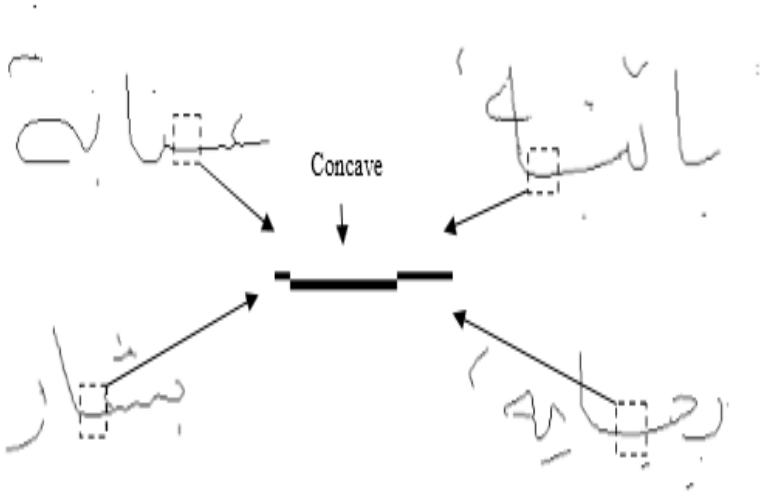


Fig.1.3 : Déformation des mots après la squelettisation (Création de concave et convexe non désirée). [5]

III.7.2.6 La correction de la déformation des caractères

Pour remédier à ce problème, nous avons proposé un algorithme qui permet de redresser les lignes. Cet algorithme se base sur le principe de continuité c.-à-d. si deux segments horizontaux ou verticaux qui se trouvent sur la même ligne respectivement colonne et s'il existe un autre segment horizontal respectivement vertical et ce dernier contient des pixels voisins avec les deux segments alors c'est une déformation horizontale respectivement verticale. La correction s'effectue par le déplacement du segment. L'algorithme peut se résumer dans les instructions suivantes : [5]

Algorithme

Répéter

Pour chaque ligne / colonne faire

1) Détecter les bords du segment A

2) Détecter les bords du segment B

3) S'il existe un troisième segment C dont les pixels des bords sont voisins avec le premier et le deuxième segment alors :

a. Créer un nouveau segment entre les segments A et B

b. Supprimer le segment C

Fin si

Fin pour

Jusqu'à « aucune modification n'est possible sur l'image ». [5]

L'extraction de certaines caractéristiques (c.-à-d. points diacritiques) demande l'estimation de la ligne de base d'écriture du mot. La méthode utilisée donne une bonne estimation de la ligne de base. Elle est basée sur l'analyse de l'histogramme de projection horizontale.

Fig.I.4 : Résultats de l'opération de correction de la déformation sur des mots manuscrits arabe. [5]

III. 7.2.7 L'estimation de la ligne de base

La ligne de base dans les texte porte une information assez pertinente sur l'orientation du texte et sur les positions des points de connexion .la méthode la plus utilisé pour trouver la ligne de base est fondée sur la projection horizontale. L'histogramme de la projection horizontale est tout simplement un vecteur ayant autant d'entrés qu'il y a de ligne dans l'image. Chaque entée contient le nombre de pixel noirs dans la ligne correspondante.

III.7.3 Segmentation explicite-implicite :

La littérature distingue parfois deux types de segmentations : implicite et explicite.

- **Explicite**

Lorsque la segmentation en graphèmes a pour objectif le découpage de l'image d'un mot directement en lettres, celle-ci est explicite. La reconnaissance de ce mot est alors réduite à une reconnaissance de caractères. [2]

- **Implicite**

En revanche, lorsque la segmentation en graphèmes découpe cette image en lettres ou en morceaux de lettres, la reconnaissance statistique doit intégrer le fait qu'une lettre est constituée d'un ou plusieurs morceaux. Par conséquent, la segmentation est dite implicite car les lettres n'apparaissent jamais de manière explicite. [2]

III.7.4 Extraction des caractéristiques

Etant donné que les graphèmes peuvent avoir des tailles variables, l'étape suivante consiste à décrire ces images par un vecteur de caractéristiques de dimension fixe. Ces nombres sont réels et chacun d'eux est censé décrire un aspect de l'image comme :

- la taille du graphème (largeur, hauteur),

- l'aire occupée par les pixels noirs,

- le barycentre des pixels noirs,

- l'inclinaison,

- l'épaisseur moyenne des traits verticaux et horizontaux, ces chiffres peuvent être calculés sur l'image entière ou une des bandes verticales ou horizontales, l'image est découpée en quatre ou cinq, selon la résolution,

- le nombre moyen de traits verticaux et horizontaux (transition pixel noir - pixel blanc sur une colonne de l'image),

- la position relative du précédent graphème, du suivant.

III.7.4.1 L'extraction des primitives

L'extraction de primitives consiste à transformer une image (caractère, graphème, bande verticale, ...) en un vecteur de primitives de taille fixe. Cette transformation revient à changer l'espace de représentation des données, du plan de l'image vers un espace à N dimensions.

Le choix des primitives est critique, et influence nettement le résultat de reconnaissance. Ces primitives doivent avoir deux propriétés :

• être discriminantes : permettre une bonne différentiation entre les classes de symboles à reconnaitre.

• maintenir un nombre de dimensions limité, afin d'éviter le phénomène de malédiction de la dimensionnalité.

Les types de caractéristiques peuvent être classés en quatre groupes principaux :

Caractéristiques structurelles, caractéristiques statistiques, transformations globales, et superposition des modèles et corrélation.

III.7.4.1.1 Caractéristiques structurelles

Les caractéristiques structurelles décrivent une forme en termes de sa topologie et de sa géométrie en donnant ses propriétés globales et locales. Parmi ces caractéristiques on peut citer : les traits et les anses dans les différentes directions ainsi que leurs tailles, les points terminaux, les points d'intersections, les boucles, le nombre de points diacritiques et leur position par rapport à la ligne de base, les voyellations et les zigzags (hamza), la hauteur et la largeur du caractère, la catégorie de la forme (partie primaire ou point diacritique, etc.)

Plusieurs autres caractéristiques peuvent être tirées, suivant qu'elles soient extraites d'une courbe, d'un trait ou d'un segment de contour. [6]

III.7.4.1.2 Les caractéristiques statistiques

Les caractéristiques statistiques décrivent une forme en termes d'un ensemble de mesures extraites à partir de cette forme. Les caractéristiques utilisées pour la reconnaissance de textes arabes sont : le zonage (zoning) et les caractéristiques de lieu géométrique (Loci). [3]

- **b.1. Le zonage** : consiste à superposer une grille n×m sur l'image du caractère et pour chacune des régions résultantes, calculer la moyenne ou le pourcentage de points en niveaux de gris, donnant ainsi un vecteur de taille n×m de caractéristiques. [6]

- **b.2 La méthode Loci** : est basée sur le calcul du nombre de segments blancs et de segments noirs le long d'une ligne verticale traversant la forme, ainsi que leurs longueurs. [3]

III.7.4.1.3 Les transformations globales

La transformation consiste à convertir la représentation en pixels en une représentation plus abstraite pour réduire la dimension des caractères, tout en conservant le maximum d'informations sur la forme à reconnaître [6].

Une des transformations les plus simples est celle qui représente le squelette ou le contour d'un caractère sous forme d'une chaîne de codes de directions. La chaîne de code obtenue est souvent simplifiée pour réduire les redondances et les changements brusques de direction.

III.7.4.1.4 Superposition des modèles (Template matching) et corrélation

La méthode de Template Matching appliquée à une image binaire (en niveaux de gris ou squelettes), consiste à utiliser l'image de la forme comme vecteur de caractéristiques pour être comparé à un modèle (Template) pixel par pixel dans la phase de reconnaissance et une mesure de similarité est calculée. [7]

III.7.5 La classification

C'est l'étape principale. On va aborder cette étape au chapitre suivant où nous présenterons un ensemble de classificateurs couramment utilisés en reconnaissance de l'écriture.la classification dans un SRC regroupe deux taches : d'une part, l'apprentissage et d'autre part la reconnaissance. [8]

III.7.5.1 Apprentissage

La faculté d'apprendre est essentielle à l'être humain pour reconnaître une voix, une personne, un objet, symbole, caractère... L'apprentissage par généralisation où l'on apprend à partir d'exemples un modèle qui nous permettra de reconnaître de nouveaux exemples. Dans les systèmes de reconnaissance d'écriture, il est difficile d'apprendre par généralisation de la même manière que l'homme. Par exemple, il leur est difficile de construire un bon modèle d'un mot ou d'un caractère et d'être ensuite capable de le reconnaître efficacement dans de nouvelles images.

Dans la littérature, plusieurs recherches sont penchées sur la question de l'apprentissage automatique ([Assr07], [Augu00], [Dupr03], [Beno07]). D'après ces travaux l'apprentissage peut se décliner en différents types principaux suivants :

Apprentissage supervisé : qui suppose que chaque donnée d'apprentissage soit étiquetée, par exemple lorsqu'on dispose d'un ensemble de données dont on connaît la classe d'appartenance a priori. Il s'agit alors de chercher les surfaces de décision séparant au mieux les classes. Ainsi c'est l'ensemble des techniques qui visent à deviner l'appartenance d'un individu à une classe.

Apprentissage non-supervisé : Contrairement à l'apprentissage supervisé, dans l'apprentissage non-supervisé il n'y a pas des données étiquetées. Autrement lorsqu'on dispose d'une base de données sans connaître leur classe d'appartenance. Il s'agit de découvrir la structure sous-jacente aux données sans imposer aucun modèle.

Apprentissage semi-supervisé : Cet Apprentissage est un bon compromis entre apprentissage supervisé et non-supervisé. Car, il suppose qu'on dispose de peu de données étiquètes est d'un grand nombre de données non étiquetées. L'apprentissage s'effectue alors à partir des deux sources de données. Apprentissage par renforcement : désigne toute méthode adaptative permettant de résoudre un problème de décision séquentielle. Ce type d'apprentissage est utile dans le cas d'apprentissage interactif. Pendant cet apprentissage, le système adaptatif agit en interaction avec son environnement, et en retour reçoit des signaux de renforcement.

Récemment, des méthodes d'apprentissage croisé à base des HMMs sont proposées pour résoudre les problèmes de ré-estimation des modèles de lettres dans les approches analytiques. Ces méthodes consistent à entraîner les modèles des lettres globalement, au travers des mots. Cela se traduira par la maximisation de la vraisemblance des mots au travers des modèles de lettres. [8]

III.7.5.2 Reconnaissance

La reconnaissance est effectuée par la recherche du modèle discriminant, elle peut se faire simplement par le calcul des probabilités d'émission de la forme par les modèles que l'on suppose a priori équiprobables. La forme à reconnaître est affectée à la classe dont le modèle fournit la probabilité la plus importante.

Les approches de reconnaissance peuvent être regroupées en trois groupes principaux : L'approche statistique, l'approche structurelle l'approche stochastique

- **Approche statistique**

La classification statistique est une manière de différencier divers objets en utilisant des lois de probabilité et de statistiques. Elle peut être utilisée pour déterminer à quelle classe appartient un caractère que l'on a isolé précédemment. [9]

Les approches statistiques bénéficient des méthodes d'apprentissage automatique qui s'appuient sur des bases théoriques fondées, telles que la théorie de la décision bayésienne, les méthodes de classification non supervisées, etc.

Nous pouvons citer trois méthodes statistiques parmi celles les plus couramment utilisées : L'approche bayésienne, les réseaux de neurones, Machines à Vecteurs de Support (SVM).

- **Approche structurelle**

Les méthodes structurelles reposent sur la structure physique des caractères. Elles cherchent à trouver des éléments simples ou primitifs, et à décrire leurs relations. Les primitives sont de type topologique telles que : une boucle, un arc... et une relation peut être la position relative d'une primitive par rapport à une autre. [3]

- **Approche stochastique**

Contrairement aux méthodes précédemment décrites, l'approche stochastique utilise un modèle pour la reconnaissance, prenant en compte la grande variabilité de la forme. La distance communément utilisée dans les techniques de « comparaison » est remplacée par des probabilités calculées de manière plus fine par apprentissage. La forme est considérée comme un signal continu observable dans le temps à différents endroits constituant des états « d'observations ». Le modèle décrit ces états à l'aide de probabilités de transitions d'états et de probabilités d'observation par état.

La comparaison consiste à chercher dans ce graphe d'états, le chemin de probabilité forte correspondant à une suite d'éléments observés dans la chaîne d'entrée [2]. Les méthodes les plus répondues dans cette approche sont les méthodes utilisant les modèles de Markov cachés (H.M.M).

III.7.6 Post-traitement

Le post-traitement comprend la vérification, l'exécution de l'action et l'adaptation. L'objectif de la vérification est d'accroître le niveau de confiance dans la classification effectuée ; une telle vérification peut être effectuée de diverses façons. L'une de ces façons consiste à utiliser une base de données comportant des combinaisons de 2 ou 3 lettres pour vérifier si la séquence des lettres reconnues ne comprend pas de combinaisons impossibles. Une autre possibilité est d'utiliser un dictionnaire pour vérifier si une certaine séquence de caractères constitue un mot valide. En règle générale, cette méthode est moins fiable puisque les mots exacts qui ne sont pas consignés au dictionnaire sont rejetés. En plus des dictionnaires contenant des lettres et/ou des mots, un modèle grammatical formel de niveau plus élevé peut être utilisé pour vérifier l'exactitude d'expressions ou de phrases entière. [10]

IV. Conclusion

Dans ce chapitre, nous avons présenté les généralités et la description de l'architecture générale des systèmes de reconnaissance de caractères.

Dans le chapitre qui suit, nous allons décrire la méthode de reconnaissance de caractères Markov caché et les principales notions de l'écriture arabe.

CHAPITRE II
L'ÉCRITURE ARABE ET LE MODÈLE DE MARKOV CACHÉ

CHAPITRE II : L'ECRITURE ARABE ET LE MODELE DE MARKOV CACHE

I. Introduction

L'écriture arabe est un phénomène qui peut être étudié, soit en tant q'un système graphique de l'arabe, soit au point de vue des modalités techniques de cette écriture.

Le système graphique présente un alphabet, des signes diacritiques, des chiffres. Ce système évolue au cours de son histoire, du quatrième siècle de l'ère courante à nos jours. L'écriture de la langue arabe dans d'autres systèmes d'écriture pose le problème de la translittération et celui de la transcription. L'utilisation du système d'écriture arabe pour écrire d'autres langues pose des problèmes d'adaptation.

L'Arabe a un plus grand vocabulaire au niveau du montant littéral que l'anglais et le français. Le grand nombre des composantes secondaires (les points diacritiques, le hamza) cause plus de difficultés au niveau de la segmentation. En outre, les règles grammaticales de l'Arabe permettent de grandes variations dans l'écriture des montants littéraux. Par conséquent, il est nécessaire de construire une base de données arabe afin d'apprendre, de tester et de comparer les systèmes de reconnaissance.

Dans ce chapitre, nous allons décrire les principales notions de l'écriture arabe et la méthode de reconnaissance de caractères Markov Caché.

II. L'écriture arabe

II.1 Types principaux de la langue arabe

Il existe deux types principaux de la langue arabe :

1. L'arabe classique - la langue du Coran et de la littérature classique. Elle est la langue pure et diffère de l'arabe standard moderne dans le vocabulaire, dont une partie d'elle reste non définie, inconnue, et implicite.

2. L'arabe standard moderne - la langue universelle du monde Arabe compréhensible par tous les parlant. En plus de l'arabe pur, elle inclut nouveaux mots : étranger arabisés, scientifiques et technologiques. C'est la langue scolaire et académique de la grande majorité, écrite et regardée via les médias et les conférences, etc. Son vocabulaire est bien déterminé aussi bien qu'explicite.

II.2. Alphabet arabe

L'alphabet arabe est l'alphabet utilisé pour écrire, entre autres, la langue arabe. Bien que très souvent désigné comme un alphabet, à la manière de l'écriture d'autres langues sémitiques, c'est généralement un abjad, terme décrivant un système d'écriture ne notant que les consonnes de la langue (ou peu s'en faut). Cet alphabet comporte 28 lettres.(figure II.1) En tant qu'alphabet de la langue du Coran, sacrée pour les musulmans, son influence s'est étendue avec celle de l'islam et il a été aussi utilisé (ou l'est encore) pour écrire d'autres langues qui n'ont aucune parenté avec l'arabe, comme le persan, le turc (avant 1928, date à partir de laquelle Mustafa Kemal Atatürk a imposé la transcription latine), la kashmiri, le sindhi, ou encore l'ourdou et le kurde (toutes ces langues, d'ailleurs, sauf le turc qui est une langue altaïque, étant indo-européennes).

On a souvent dû ajouter ou modifier certaines lettres pour adapter cet alphabet au système phonologique des langues en question. Certaines langues d'Afrique, comme lhaoussa et le somali, s'écrivirent par des adaptations de l'alphabet arabe avant d'être écrites avec l'alphabet latin.

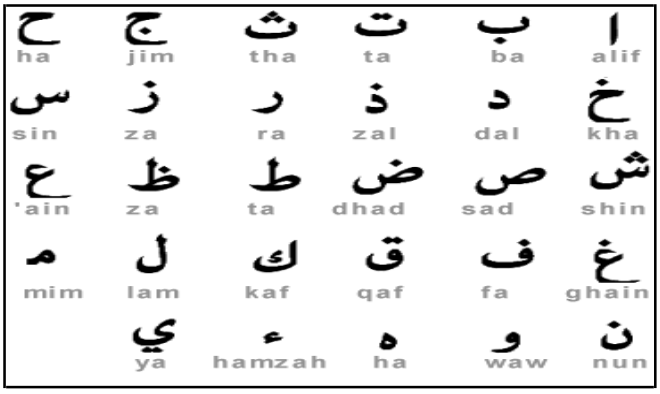


Fig II.1 : Les lettres isolées de l'alphabet arabes.

L'arabe s'écrit de droite à gauche avec un sens du tracé respecté (voir figure II.2) et la construction d'un mot se fait en collant les lettres une à une, de droite à gauche. Mais chaque lettre de l'alphabet s'écrit différemment selon sa position dans le mot. Les notions de lettre majuscule et lettre minuscule n'existent pas (l'écriture est donc monocamérale).

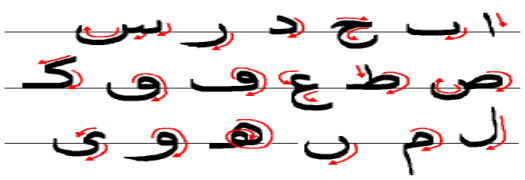


Fig. II.2 : Sens d'écriture des lettres arabes.

II.3. Caractéristiques de l'écriture Arabe

L'alphabet Arabe est composé de 28 lettres dont la forme change selon la position dans le mot. Les lettres s'écrivent différemment selon qu'elles sont isolées, au début, au milieu ou à la fin du mot (voir tableau II.1). [11]

Pour la plupart des lettres, les formes début/milieu et fin/isolé sont identiques à la ligature près. La présence d'une ligature avec la lettre précédente ou avec la lettre suivante ne modifie pas la forme de la lettre de manière significative (pas plus que dans l'écriture manuscrite cursive latine). En arabe, les ligatures se situent toujours au niveau de la ligne d'écriture, c'est-à-dire qu'il n'existe pas de lettre à liaison haute comme le "o" ou le "v" en alphabet latin. Le plus grand alphabet aujourd'hui contient maintenant environ 100 formes possibles. Donc, il existe une influence du scripteur sur les modèles d'écriture arabes et de divers modèles calligraphiques, certains sont montrés sur la fig.II.3.

lettre	nom	fin	Milieu	début	lettre	nom	fin	milieu	début
ا	alif	ـا	ـا	ا	ط	Ṭa	ـط	ـطـ	طـ
ب	ba	ـب	ـبـ	بـ	ظ	Ẓa	ـظ	ـظـ	ظـ
ت	ta	ـت	ـتـ	تـ	ع	ɛayn	ـع	ـعـ	عـ
ث	tha	ـث	ـثـ	ثـ	غ	ghayn	ـغ	ـغـ	غـ
ج	jim	ـج	ـجـ	جـ	ف	fa	ـف	ـفـ	فـ
ح	Ḥa	ـح	ـحـ	حـ	ق	qaf	ـق	ـقـ	قـ
خ	kha	ـخ	ـخـ	خـ	ك	kaf	ـك	ـكـ	كـ
د	dal	ـد	ـدـ	دـ	ل	lam	ـل	ـلـ	لـ
ذ	dhal	ـذ	ـذـ	ذـ	م	mim	ـم	ـمـ	مـ
ر	ra	ـر	ـرـ	رـ	ن	nun	ـن	ـنـ	نـ
ز	zay	ـز	ـزـ	زـ	ه	ha	ـه	ـهـ	هـ
س	sin	ـس	ـسـ	سـ	و	waw	ـو	ـوـ	وـ
ش	shin	ـش	ـشـ	شـ	ي	ya	ـي	ـيـ	يـ
ص	Ṣad	ـص	ـصـ	صـ	ء	hamza	ـئ ـإ ـؤ ـأ		
ض	Ḍad	ـض	ـضـ	ضـ					

Tableau II.1: Les lettres arabes avec leurs positions dans le mot.

Fig. II.3 : Quelques styles de Calligraphie arabe.

II.4. Signes diacritiques

L'écriture Arabe est riche en diacritique d'une façon générale, et plus particulièrement en points. Le terme "signe diacritique" peut porter à confusion : dans certains travaux, seules les voyelles arabes sont appelées diacritiques. Dans d'autres travaux, en revanche, tous les signes secondaires sont appelés diacritiques, qu'il s'agisse des voyelles, des points ou des autres signes (chadda, madda, hamza, ...). C'est cette deuxième terminologie que nous employons ici : un signe diacritique est une composante secondaire d'une lettre, qui vient la compléter ou en modifier le sens. Dans cette thèse, les "signes diacritiques" désigneront à la fois points, voyelles et autres signes secondaires.

II.4.1 Les points

Nous comptons 15 lettres, parmi les 28 de l'alphabet qui comportent des points. Ces points apparaissent au-dessus ou en dessous du caractère uniquement. Le nombre maximal de points que peut avoir une lettre est de trois points au-dessus du caractère, ou deux points en-dessous. Ces points permettent de différencier la prononciation des lettres Arabes comme montrée dans la figure II.4.

Fig. II.4: Changement de prononciation de la même forme selon la position des points.

Le tableau II.2 illustre la variabilité des styles d'écriture des points ou groupes de points en écriture manuscrite arabe. Un groupe de deux points peut ainsi s'écrire sous forme d'une seule, ou de deux composantes connexes. On remarque la très forte similarité entre deux points relies par un trait, et une voyelle de type "A" ou "I" dont les exemples sont donnés figure II.4. Un groupe de trois points peut donner lieu à une, deux ou trois composantes connexes, en fonction du style d'écriture. [13]

un point				
2 points				
3 points				

Tableau II.2 : Variabilité des styles d'écriture des points.

II.4.2 Les voyelles

En arabe, les voyelles ne sont pas seulement des lettres, mais aussi des signes diacritiques associés aux lettres sur lesquelles ils s'appliquent (figure II.5).

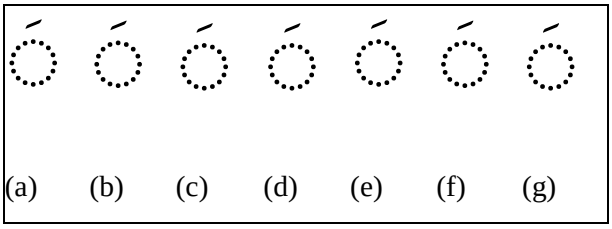


Fig. II.5 : Voyelles en rabe : (a) A, (b) OU, (c) I, (d) ~, (e) AN, (f) OUN, (g) IN

En général on ne représente pas les voyelles signes, sauf dans les manuels scolaires. L'absence de voyelles peut toutefois être source de confusions. Un mot peut avoir plusieurs voyellations possibles et par conséquent plusieurs catégories grammaticales. Par exemple comme mentionnés dans la table, le mot « البر » peut prendre trois sens en jouant sur trois voyellations possibles.

Le mot	Le sens
البر	La lithosphere
البر	Le blé
البر	La bienfaisance

Les voyelles peuvent parfois être mentionnées sur certaines lettres pour lever l'ambigüité et faciliter la lecture. Mais en général, les scripteurs les omettent purement et simplement, et c'est au lecteur qu'est réservé le soin d'interpréter correctement le sens de la phrase en fonction du contexte.

II.4.3 Les autres signes diacritiques

Les autres signes diacritiques sont la hamza, la chaddet la madda , (voir figure II.6). La chadda est une accentuation de la lettre (c'est l'équivalent d'une consonne doublée). Hamza et madda suivent des contraintes morphosyntaxiques plus complexes. [13]

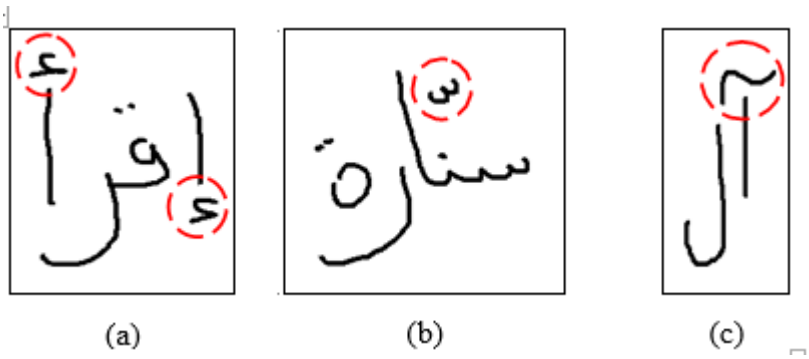


Fig. II.6: Autres signes diacritiques : (a)hamza, (b) chadda, (c) madda.

II.4.4 Ascendants et descendants

Comme dans l'écriture latine, l'écriture arabe contient des ascendants et des descendants (voir figure II.7). En arabe, les descendants peuvent se prolonger horizontalement sous la bande de base, ce qui introduit une superposition verticale entre la lettre qui comprend le descendant et la lettre suivante. [13]

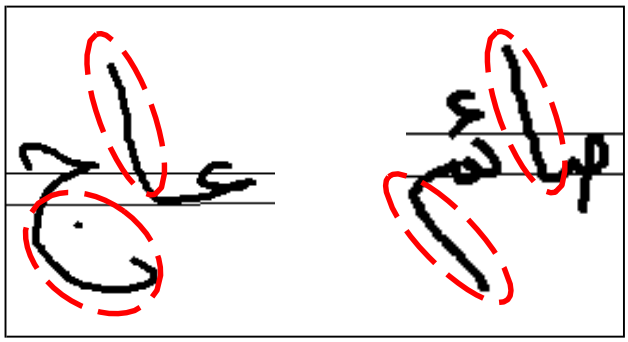


Fig. II.7 : Les ascendants et descendants sont entourés.

II.4.5 Une ou plusieurs composantes connexes par mot

L'écriture arabe est semi-cursive dans ses deux formes, imprimée et manuscrite. En effet, un mot arabe est une séquence d'entités connexes entièrement séparées appelées pseudo- mot. Un mot peut être composé d'un ou de plusieurs pseudo-mots ; ceci est dû à la présence de caractères qui ne peuvent pas être attachés à leur successeur. Chaque pseudo-mot est une séquence de lettres liées, ce qui donne l'aspect de cursivité à cette écriture. Notons qu'un caractère isolé peut constituer un pseudo-mot à lui seul, comme le montre la figure II.8. Six lettres ne sont pas liées à leur successeur : د, ذ, ر, ز, و et. ا Ces lettres introduisent donc une coupure dans le mot.

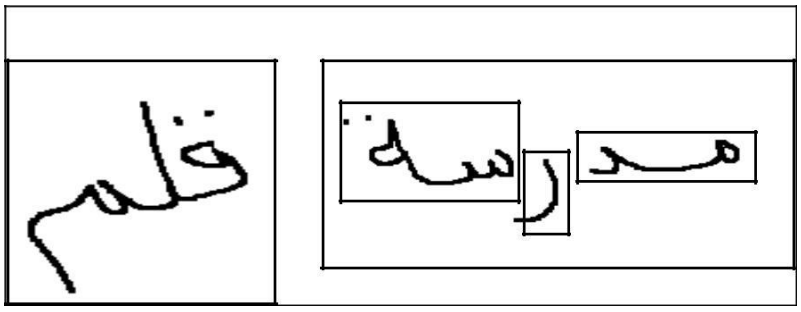


Fig. II.8 : (a) Le mot « crayon » est composé d'un pseudo-mot, (b) Le mot « école » est composé de 3 pseudo-mots.

En manuscrit, l'espacement entre les différents pseudo-mots d'un même mot n'est pas forcément systématiquement supérieur à l'espacement entre deux mots différents, ce qui pose parfois des problèmes de segmentation. Lorsqu'une des 22 lettres (28 moins les 6 qui ne se lient pas avec la suivante) apparait dans sa forme "fin de mot" ou "isolée", cela signifie obligatoirement que l'on arrive à la fin d'un mot.

Certaine lettre des 22, présente un phénomène appelé : jointure conditionnelle où la lettre changera sa forme à la fin du mot. (Voir figure II.9)

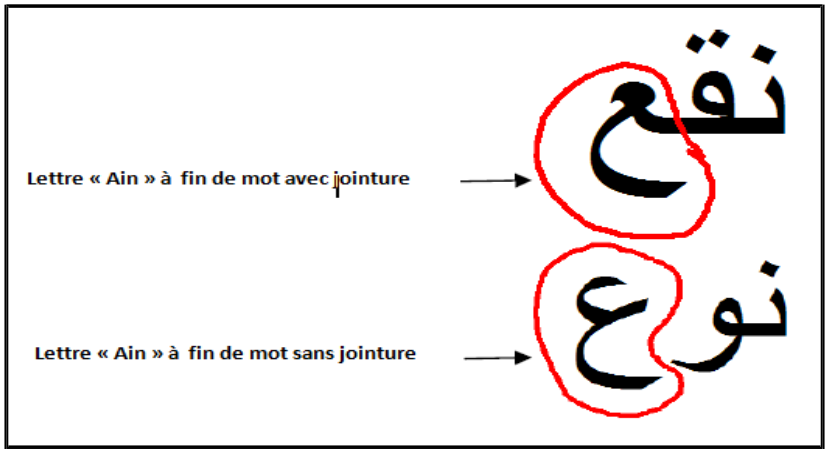


Fig. II.9 : Jointure Conditionnelle

II.4.6 Ligatures verticales

En écriture arabe, il n'y a pas de liaisons hautes comme le "v" ou le "o" en latin : les ligatures se situent au niveau de la ligne support de l'écriture (ligne de base). En revanche, les scripteurs sont libres de constituer certains groupes de deux ou trois lettres liées verticalement en début de pseudo-mot. Ce sont les ligatures verticales (figures II.10). En général très complexes à segmenter, nous choisissons de les reconnaître telles quelles. [12]

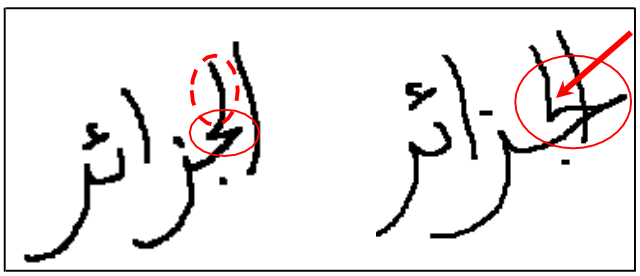


Fig. II.10 : Ligatures verticales et inversion de l'ordre du tracé.

II.5. Quelques systèmes existants de reconnaissance d'écriture arabe

Plusieurs chercheurs ont mené plusieurs travaux afin de proposer des systèmes de reconnaissance de caractères arabes. Dans ce qui suit nous présentons par ordre chronologique un récapitulatif précisant les caractéristiques et les performances de certains systèmes existants.

ICDAR 1997:(Adnan Amin): Il analyse différentes approches pour la reconnaissance automatique de l''écriture arabe. Il présente les approches On-Line/Offline, ainsi que les approches Globale/Analytique. Il pose quelques unes des problématiques fondamentales de la reconnaissance de l''écriture arabe (alphabet, signes diacritiques, styles d'écriture, codage de l''écriture, pseudo-mots, ...). [14]

CIFED 2000: Ils s'intéressent aux approches markoviennes qui ont été mises en œuvre pour résoudre le problème de la reconnaissance de l'écriture arabe. Concernant l''écriture arabe manuscrite, ils passent notamment en revue les HMM à une dimension, les HMM-planaires (PHMM). [15]

Selon Ben Amara et al, les HMM-1D donnent des résultats encourageants, tout comme dans le cas de l'écriture latine. Les HMMs sont indépendants de alphabet considéré. Mais Ben Amara et al. soulignent également les limitations classiques d'une modélisation linéaire, qui là encore, ne sont pas propres à l''écriture arabe. Pour pallier ce problème, ils introduisent les PHMM, qui prennent en compte des variations bidirectionnelles de l''écriture. Les auteurs justifient en particulier ce choix sur l'arabe en raison du degré de difficulté supplémentaire que représente la présence de ligatures verticales. Ce dernier argument concernant la présence de ligatures verticales, n'est pas le plus pertinent. Les symboles qui correspondent aux ligatures verticales peuvent en effet être reconnus comme tels dans leur globalité, sans qu'il soit nécessaire de les segmenter.

PAMI 2006: Le système de Liana M. Lorigo et Venu Govindaraju est le plus complet sur la reconnaissance de l'écriture arabe à l'heure actuelle. Ils passent en revue la plupart des problèmes spécifiques à la reconnaissance de l'écriture arabe et ils présentent quelques bases de données. [16]

Ils découpent la chaîne de traitement d'un système de reconnaissance en fonctionnalités: representation, segmentation, extraction de primitives, moteur de reconnaissance, où ils distinguent: réseau de Neurones, modèles de Markov Cachés et systèmes hybrides.

En conclusion, les auteurs expliquent que des systèmes de reconnaissance de l''écriture arabe donnent des performances satisfaisantes sur des applications constraints (petite taille de lexique, forme des mots relativement contrainte...). L'avenir de la discipline se situe dans la capacité des systèmes à traiter de l'écriture libre, comme des courriers manuscrits. De telles applications nécessitent des modèles de langage développés, qui sont pour l'instant des voies Largement inexplorées en langue arabe. L'intégration de contraintes morphologiques (analyse de la formation des mots à partir de suffixes et racines) permettrait également de reconnaître des mots hors-vocabulaire.

ISSPA 2007: Mohamed Cheriet fait une synthèse des différentes approches, avec la Volonté d'ouvrir la réflexion sur de futures applications industrielles. Il part du constat suivant: les systèmes de reconnaissance de l'écriture manuscrite arabe, même s'ils obtiennent Des performances encourageantes, n'ont pour l'instant fait leur preuves que sur des données Académiques, dans le cadre d'applications à l'environnement contraint. Il n'y a pas de système Commercial de reconnaissance de l'écriture arabe manuscrite. [17]

Il pose un certain nombre de questions ouvertes: Nécessité de procéder à une segmentation? Quel paradigme de reconnaissance utiliser? Quelle technique de reconnaissance utiliser? Faire un post-traitement, ou essayer d'intégrer les contraintes au sein de la chaîne de Reconnaissance, Redéfinir de nouvelles architectures, ou s'appuyer sur des techniques Existantes?

Il insiste également sur l'importance de l'analyse morphologiques. Il s'interroge sur le meilleur endroit pour intégrer l'analyseur morphologique: au sein de la chaîne, ou à la fin en tant que post-traitement?

Il s'interroge, également, sur les moyens de constituter un lexique pour guider un système de reconnaissance à vocabulaire ouvert, et conclut par l'intérêt d'utiliser le Coran en tant que corpus, pour en dériver un lexique de manière automatique ou semi-automatique

III. Modèle de Markov caché

La modélisation stochastique permet l'utilisation des modèles probabilistes pour traiter les problèmes à information incertaine ou incomplète. Ainsi, les modèles de Markov connaissent un regain d'intérêt tant dans leurs aspects théoriques qu'appliqués.

Les modèles de Markov cachés MMCs (HMM, pour Hidden Markov Model) sont des méthodes de modélisation stochastiques introduites fin des années 60, début des années 70 pour de nombreuses applications, principalement dans le domaine de la parole, pour l'analyse de l'ADN, la prédiction de l'évolution des épidémies,

Les modèles de Markov connaissent actuellement un essor important en reconnaissance des formes grâce à leur capacité d'intégration du contexte et d'absorption du bruit.

L'utilisation des MMCs en reconnaissance automatique de l'écriture a permis d'obtenir des résultats intéressants pour certaines applications. Les différents travaux réalisés reposent pour une grande part sur l'expérience accumulée dans le domaine de la reconnaissance de la parole où les MMCs sont fréquemment utilisés. Comparés à d'autres approches de reconnaissance (structurelle, géométrique, etc.), les MMCs se distinguent par leur capacité de modéliser efficacement différentes sources de connaissance. En effet, d'une part ils offrent une intégration cohérente de différents niveaux de modélisation (morphologique, lexical et syntaxique) et d'autre part, il existe des algorithmes puissants permettant de déterminer la valeur optimale des paramètres fournissant la meilleure adéquation entre le modèle et la base de données (connue) qualifiée d'apprentissage. [18]

III.1 Définition des MMCs

Un modèle de Markov caché est un double processus stochastique (X t, Y t) 1<=t<=T. La chaîne interne X t non observable, et la chaîne externe Y t observable, s'allient pour générer le processus stochastique. La chaîne interne est supposée, pour chaque instant, être dans un état où la fonction correspondante génère une composante de l'observation. La chaîne interne change d'état en suivant une loi de transition. L'observateur ne peut voir que les sorties des fonctions aléatoires associées aux états et ne peut pas observer les états de la chaîne sous-jacente, d'où le terme de Modèles de Markov Cachés (ou Hidden Markov Model). [19]

Le processus (X t) 0<=t<=T est une chaîne de Markov d'ordre 1, il doit vérifier :

P (Xt+1 = q j / X t = q i, ..., X 0 = q 0) = P (X t+1 = q j / X t = q i) = a ij pour tout t >= 0.

Le processus (Y t) 0<=t<=T, processus observable, vérifie :

P (Y t = y t / X t = q i, X 1=q 1, Y t-1=y t-1..., Y 1=y 1) = P (Y t=y t / X t = q i) = b i (yt)

Les observations sont supposées indépendantes les unes des autres conditionnellement à la suite d'états. Chaque réalisation de Y t ne dépend que de l'état courant caché. Les observations y t peuvent être de nature :

- **discrète** => b i est une distribution de probabilité discrète : une loi discrète est généralement représentée par les fréquences d'apparitions des observations discrètes.

- **continue** => b i est une fonction de densité de probabilité définie sur R d : les densités traditionnelles utilisées sont des densités gaussiennes, entièrement définies par le vecteur moyen et la matrice de covariance, ou des densités de type multi-gaussiennes (sommes pondérées de densités gaussiennes).

Et pour mieux comprendre, voici un exemple théorique extrait de. Considérant deux personnes séparées par un mur. La première personne possède trois dés biaisés. Tour par tour, elle choisit un dé, le lance et énonce à voix haute le numéro de la face résultante à la deuxième personne. La deuxième personne de l'autre côté du mur, ne connait que la séquence de numéros de dés, pas la séquence des dés qui a été choisie par la première personne. La séquence des dés est appelée séquence des états cachés, et la séquence des numéros de face est appelée séquence d'observations.

III.2 Les éléments d'un MMC

Un MMC est caractérisé par :

- N, le nombre d'états dans le modèle. On note S = {s 1, s2. . ., s N}, l'ensemble des N états du modèle, qt l'état au temps t (qt appartient à S).

- M, le nombre de symboles d'observation par état, c-à-dire la taille de l'alphabet discret, les symboles d'observation correspondent aux sorties physiques du système à modéliser. On note V = {v l, v2 . . .. v M}, l'ensemble discret des M symboles, et ot un symbole au temps t (ot appartient à V). [20]

- La distribution de probabilité de transition d'un état A= {a ij}. Tel que :

aij = P [q t+1 = S j\qt =S i], ,. 1 jN i ≤≤

Pour le cas spécial où à partir de tout état, on peut atteindre directement n'importe quel autre état, on a : a ij >0, pour tout i, j. Pour autre types d'MMC, on a aij =0 pour un seul pair (i, j) ou plus.

- La distribution de probabilité de symboles d'observation dans un état j, B= {b j(k)}, où :

bj(k) = P [v k en t\ qt = S j], 1, jN ≤≤ 1. kM ≤≤

- La distribution d'état initial π = {πi} tel que : [ ], 1 ii qsp == π 1. i N ≤≤

Donc, la spécification complète d'un MMC nécessite la spécification des deux paramètres du modèle N et M, la spécification des symboles d'observations, et la spécification des trois mesures de probabilités A, B et π. L'ensemble complet des paramètres d'un MMC peut être noté : λ= (A, B, π).

Dans un MMC, les contraintes (markoviennes) suivantes doivent être respectées :

$$\sum_{i=1}^{N} \pi_i = 1,$$

$$\sum_{j=1}^{N} a_{ij} = 1, \qquad 1 \le i \le N,$$

$$\sum_{k=1}^{M} b_j(k) = 1, \qquad 1 \le j \le N$$

III.3. Extensions des MMCs

Selon le type de densité de probabilité d'observations, discrète ou continue, il est possible de construire deux types de modèles de MMC : soit un MMC discret soit un MMC continu.

- MMC discret « Discret Hidden Markov Models (DHMM) » : les observations en général sont continués puisqu'elles proviennent de phénomènes physiques continus. Dans le cas d'un MMC discret, les observations continues sont quantifiées à l'aide d'un dictionnaire (l'alphabet d'observations dénombrables).

- MMC continu « Continuous Hidden Markov Models (CHMM) » : bien qu'il soit possible de quantifier les observations continues, il peut y avoir une sérieuse dégradation d'information associée à cette quantification. Il sera, alors, avantageux de choisir une fonction de densité de probabilités d'observations continues, conditionnée par les états du processus.

Voici un tableau comparatif de ces deux modèles :

	A. MMC continu	B. MMC discret
Nombre de paramètres à estimer	Un nombre élevé	Moins que A
Précision de la classification	Précis	Moins précis que A
Hypothèses sur la nature des observations	Importantes	Moins importantes que A
Implémentation	Difficile et lente	Plus facile et plus rapide que A
Nombre de corpus d'apprentissage	Moyen	Plus élevé que A

Tableau II.3: comparaison entre DHMM et CHMM.

III.3.1 Durée de séjour dans un état

L'un des inconvénients des MMC de base est le manque d'informations concernant la variabilité de la durée de séjour dans un état en favorisant les courtes durées. Pourtant, c'est d'une importance majeure dans certains processus physiques, à titre d'exemple, la variabilité de la durée des sons dans la parole.

Principalement deux méthodes ont été développées :

- la méthode de Ferguson basée sur un MMC de Durée variable Discrète « Variable Duration Hidden Markov Model VDHMM ».

- et celle de Levinson basée sur un MMC de durée variable continue « Continuous Duration Hidden Model CDHMM ».

III.3.2 Ordre d'une chaine

Une limitation des Modèles de Markov Cachés de base est dans sa définition. Ils modélisent un processus markovien. C'est-à-dire que son évolution ne dépend pas de son passé, mais uniquement de son état présent. Le processus est d'ordre un, ce qui n'est pas le cas de nombreuses applications. Dans un processus, si l'état futur dépend des k états précédents, la chaîne est d'ordre k.

Une application à la reconnaissance de l'écriture a été faite par Kendu et Bahl ; ils ont montré l'avantage des MMC du second ordre sur ceux du premier ordre. Ils ont également signalé la difficulté d'implémentation des MMC du second ordre due à l'indisponibilité de leurs formules de réestimation dans la littérature.

Ceci a été corrigé par He et Krouille. Ils ont développé et traité une technique d'extension des algorithmes de Viterbi et de Baum-Welch pour un MMC du second ordre et d'ordre quelconque.

III.4. Les types des MMCs

III.4.1 MMC ergodique

Dans ce type, tout état est directement atteignable depuis tout autre état. Il est plus général et intéressant lorsque le modèle représente un processus dont on veut suivre les évolutions des états. [8]

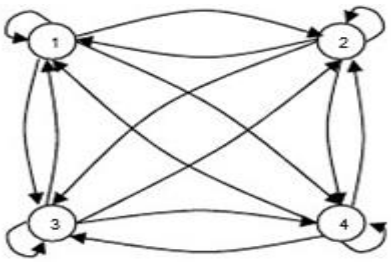


Fig. II.11 : un MMC ergodique

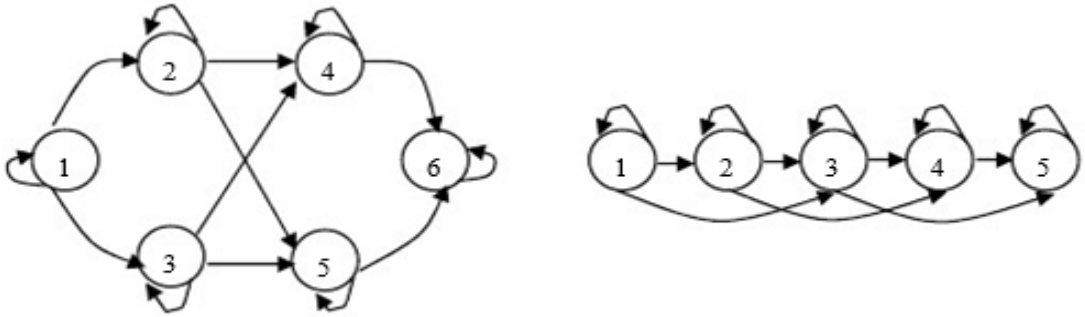
Formellement, on peut écrire :

$$\Pi i \neq 0,\ aij \neq 0,\ \forall\ i,\ j$$

III.4.2 MMC gauche-droite

Dans ce type, si le temps augmente, alors les indices des états augmentent également. Il est utilisé pour suivre des observations dont l'évolution se fait dans un ordre donné tel que la reconnaissance de la parole.

La figure II.12 suivante illustre deux exemples du MMC gauche-droite (parallèle et séquentielle



(a) Parallèle (b) Séquentiel

Fig. II.12 : un MMC gauche-droite (a) Parallèle (b) séquentiel.

Formellement, on peut écrire :

$$\text{Aij si } j<i \; ; \quad \pi i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases}$$

III.5 Les trois problèmes fondamentaux pour les MMCs

L'utilisation des MMCs nécessite la résolution des trois problèmes fondamentaux suivants :

- problème d'évaluation.

- problème de reconnaissance.

- problème d'apprentissage.

Dans les sections suivantes on va détailler ces problèmes et présenter les algorithmes les plus utilisés pour les résoudre.

III.5.1 Problème d'évaluation

Etant donnés une suite d'observations O= {o1, o 2…o m} et un modèle λ, comment peut-on calculer efficacement la probabilité (vraisemblance) que la suite d'observations O soit produite par λ, c'est-à-dire P (O\λ).

Il existe deux procédures récurrentes de calcul de la vraisemblance :

- l'algorithme Forward-Backward, qui fournit une solution exacte à ce problème faisant intervenir tous les chemins dans le modèle MMC.

- l'algorithme Viterbi, fournissant une solution approximative faisant intervenir uniquement le meilleur chemin dans le modèle MMC**.**

III.5.1.1 L'algorithme Forward Backward

Dans cette approche, on considère que l'observation peut se faire en deux étapes :

- L'émission de la suite d'observations {o1, o 2…o t} et la réalisation de l'état st au temps t : **Forward**

- L'émission de la suite d'observations {ot+1, o t+2 …, o T} en partant de l'état si au temps t : **Backward**

La 1 ière étape suit l'algorithme 1

- Initialisation :

$$\alpha_1(i) = \pi_i * b_i(o_1), \quad 1 \le i \le N,$$

- itération :

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) * a_{ij}\right] * b_j(o_{t+1}), \text{Pour } 1 \le t \le T-1, \quad 1 \le j \le N,$$

- terminaison :

$$P(O \mid \lambda) = \sum_{t=1}^{N} \alpha_T(i)$$

L'algorithme 1 : Indique comment l'état s j peut être atteint au temps t+

1 à partir de N états possibles s i, 1≤i≤N, au temps t.

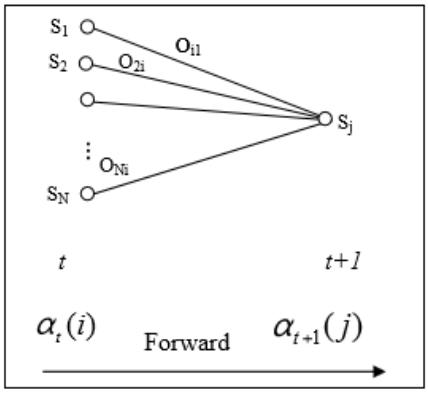


Illustration of the séquence of opérations required for the computation of

The Forward variable αt+1 (j) .

La 2ième étape (Backward) se déroule comme suit :

On considère une variable Backward définie comme :

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, \ldots\ldots O_T) | qt = s_i, \lambda),$$

C'est-à-dire la probabilité de la séquence d'observation partielle, à partir de t+1 jusqu'à la fin, sachant l'état s i au temps t et le modèle λ (voir figure II.13).

- initialisation :

$$\beta_T(i) = 1, \qquad 1 \le i \le N,$$

- itération :

$$\beta_t(i) = \left[\sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)\right], \quad \text{pour } 1 \le t \le T-1, 1 \le j \le N,$$

Algorithme 2 : l'algorithme Backward.

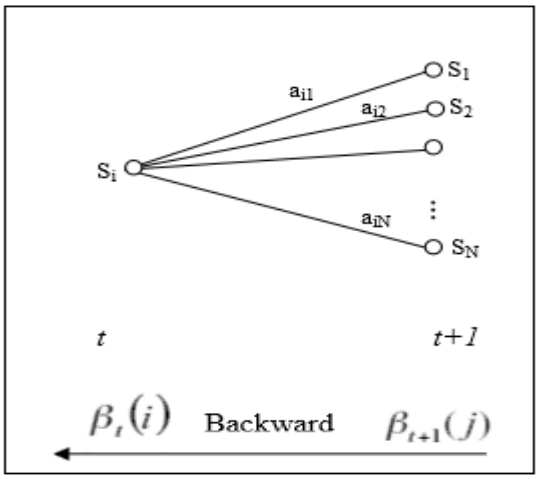


Fig.II.13 : Illustration of the séquence of opérations required for the Computation of the Backward variable βt(i)

III.5.1.2 Algorithme de Baum-Welch

L'algorithme Baum-Welch est basé sur le théorème de Baum qui garantit l'atteinte d'un maximum local de la fonction de vraisemblance par réestimation des paramètres A, B, π. Cette méthode de Maximum de Vraisemblance est la plus utilisée dans les applications.

Cet algorithme peut être représenté sous la forme itérative suivante :

1. Charger le modèle initial.

   λ0= {π, A, B}, N et M

   Avec N : le nombre d'états.

   M : le nombre de symboles distincts.

2. Charger les observations de toutes les occurrences d'un mot.

3. Calculer la probabilité totale à partir de la matrice alpha.

4. Retour à l'étape 3, si cette dernière probabilité est supérieure à un seuil fixé déjà et on n'a pas dépassé le nombre d'itération maximal, sinon retour à l'étape 1 pour traiter un autre mot.

Algorithme 3 : l'algorithme Baum-Welch.

Le test d'arrêt est généralement un nombre d'itérations qui est fixé empiriquement.

Le choix d'un modèle initial influe sur les résultats : toutes les valeurs nulles de A et de B au départ, restent à zéro à la fin de l'apprentissage. Il est à noter que l'algorithme converge vers des valeurs de paramètres qui forment un point critique de P(O\λ). Donc, nous obtenons un maximum local ou un point d'inflexion. D'où la nécessité de bien choisir le modèle initial.

Pour avoir une estimation convenable du modèle, les ré-estimations se font sur un ensemble de plusieurs suites d'observations appelées corpus d'apprentissage. Donc la taille du corpus d'apprentissage influe, elle aussi, sur les résultats. Il est souhaitable que celle-ci soit importante.

III.5.1.3 Algorithme de Viterbi

Au lieu de prendre en compte tous les chemins autorisés, seul le plus probable est gardé. Ainsi, il suffit de remplacer dans les équations précédentes l'opérateur ∑ par max. Ce critère est largement utilisé du fait du faible coût qui lui est associé (en effet, il est évident que l'opérateur max est moins coûteux en temps de calcul que l'opérateur ∑ sur tous les états). L'algorithme de Viterbi est donc une simplification de la récurrence avant.

III.5.2 Problème de reconnaissance

Etant donnés un MMC λ et une séquence observée O, comment connaitre la séquence des états cachés du λ qui a la plus probablement généré O ?

Pour résoudre ce problème la procédure de Viterbi est utilisée. Elle est basée sur les techniques de programmation dynamique. C'est un algorithme récursif qui permet de trouver à partir d'une suite d'observations, une solution optimale au problème d'estimation de la suite d'états.

La procédure complète de l'algorithme de Viterbi pour trouver la meilleure séquence d'états est comme suit :

- initialisation :

$\delta_1(i) = \pi_i * b_i(o_1)$ δ = 1<= i <= N

$\psi_1(i) = 0$

- calcul récursif :

[ ] () () *) max (1 1 itt ij iN t bO iaj − ≤≤ = δδ

2, tT ≤≤ 1, jN ≤≤

Nnnnnnnnnnnn [] t ij

N t ia j ( )* argmax() 1 1 − ≤≤ = δψ

2, tT ≤≤ 1, jN ≤≤

- terminaison :

[ ] ( )max 1

* pi T iN δ ≤≤ = [ ] () argmax 1

* qi T iN T δ ≤≤ =

- le backtracking :

(), * 11 * ++=Ψ ttt qq 1. ,...,21, −−= TTt

Algorithme 4 : l'algorithme de Viterbi

III.5.3 Problème d'apprentissage

Etant donnés un MMC λ= {Π, A, B} et une séquence observée O, Comment peut-on ajuster les paramètres du modèle λ pour maximiser la vraisemblance P(O\λ) ?

Les approches les plus utilisées sont basées sur des adaptations de l'algorithme EM (Expectation-Maximisation) appelées :

- algorithme de Baum-Welch : P(O\λ) est estimée en tenant compte de tous les chemins possibles (implémentation de l'algorithme Expectation-maximisation (EM)).

- algorithme de Viterbi : P(O\λ) est estimée en tenant compte du meilleur chemin uniquement (approximation de l'algorithme EM).

IV. Conclusion

Nous avons présenté dans ce chapitre les principales notions de l'écriture arabe et la méthode de reconnaissance de caractères.

Les modèles de Markov cachés qui sont des méthodes stochastiques largement utilisées dans le domaine de la reconnaissance de l'écriture. Comparés à d'autres approches de reconnaissance, les MMCs se distinguent par leur capacité de modéliser efficacement différentes sources de connaissance. En effet, d'une part ils offrent une intégration cohérente de différents niveaux de modélisation (morphologique, lexical et syntaxique) et d'autre part, il existe des algorithmes puissants pour la reconnaissance et l'apprentissage.

Les étapes et détails de la conception et l'analyse de notre application sera l'objet du prochain chapitre.

CHAPITRE III
ANALYSE
ET
CONCEPTION

I. Introduction :

Cette partie est consacrée aux étapes fondamentales pour le développement de notre système de reconnaissance de mots manuscrit arabe. Pour la conception et la réalisation de notre application, nous avons choisis de modéliser avec le formalisme UML (Unified Modeling Language) qui offre une flexibilité marquante qui s'exprime par l'utilisation des diagrammes.et on termine ce chapitre par la description de notre système**.**

II. Présentation de langage UML :

II.1 Historique :

- **En 1995 :** Méthode unifiée 0.8 (intégrant les méthodes Booch'93 et OMT)
- **En 1995 :** UML 0.9 (intégrant la méthode OOSE)
- **En 1996 :** UML 1.0 (proposée à l'OMG)
- **En 1997 :** UML 1.1 UML Tarak Chaari (ISECS) 20 (standardisée par l'OMG)
- **En 1998 :** UML 1.2
- **En 1999 :** UML 1.3
- **En 2000 :** UML 1.4
- **En 2003 :** UML 1.5
- **En 2004 :** UML 2.0
- **En 2010 :** UML 2.3beta. [25]

II.2 Définition :

UML (Unified Modeling Language), se définit comme un langage de modélisation graphique et textuel destiné à comprendre et à définir des besoins, spécifier et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue. Il véhicule en particulier :

- Les concepts des approches par objets : classe, instance, classification, etc.
- Intégrant d'autres aspects : associations, fonctionnalités, événements, états, séquences, etc.

II.3 Les diagrammes d'UML :

- **Diagramme de cas d'utilisation :** représente les relations entre les acteurs et les fonctionnalités du système. Les cas d'utilisation présentent une vue externe de la façon d'utiliser un système, que ce soit l'application, un sous-système, une fonction, un composant.
- **Diagramme de classes :** est un ensemble d'éléments statiques qui montre la structure d'un modèle (les classes, leur type, leur contenu et leurs relations).
- **Diagramme d'objets (objet : instance d'une classe) :** représente les objets et les liens entre eux. Il permet d'affiner un aspect particulier d'un diagramme de classes pour un contexte donné.
- **Diagramme d'états/transitions :** décrit le cycle de vie des objets formalisés dans une classe (une classe ne se voit donc associer qu'un cycle de vie).
- **Diagramme de composants :** montre les éléments logiciels (exécutables, librairies, fichiers qui constituent le système) et leurs dépendances.
- **Diagramme de déploiement :** indique la répartition physique des matériels du système (processeurs, périphériques) et leurs connexions.
- **Diagramme de séquence :** représente les messages échangés entre les objets. Il donne une notion temporelle aux messages.
- **Diagramme de collaboration :** représente les messages échangés entre les objets. Il insiste plus particulièrement sur la notion organisationnelle.
- **Diagramme d'activités :** décrit le déroulement d'un processus formalisé éventuellement dans un cas d'utilisation, il modélise les actions effectuées sur le système (peut permettre de présenter un processus métier).

II.4 Objectif de l'application :

Afin d'effectue le développement d'une application il faut répondre aux questions suivantes :

- A qui sera destinée l'application ?

- Cette application sera destinée à toute utilisateur désirant de faire la reconnaissance de mots manuscrits arabes.

- Quel est son objectif ?

- L'objectif de notre travail est la réalisation de système de reconnaissance de noms arabes manuscrit.

II.5 Modélisation avec L'UML :

UML permet de représenter des modèles, mais il ne définit pas les processus d'élaboration de ces modèles. Les auteurs d'UML conseillent tout de même une démarche pour favoriser la réussite d'un projet, cette démarche doit être :

- **Une démarche itérative et incrémentale** : Pour comprendre et représenter un système complexe, pour analyser par étapes, pour favoriser le prototypage et pour réduire et maitriser l'inconnu.
- **Une démarche guidée par les besoins des utilisateurs :** Tout est basé sur le besoin des utilisateurs du système, le but du développement lui-même est de répondre à leur besoin. Chaque étape sera affinée et validée en fonction des besoins des utilisateurs.
- **Une démarche centrée sur l'architecture logicielle :** c'est la clé de voute du succès d'un développement, les choix stratégiques définiront la qualité du logiciel.

II.6 La démarche de modélisation avec UML

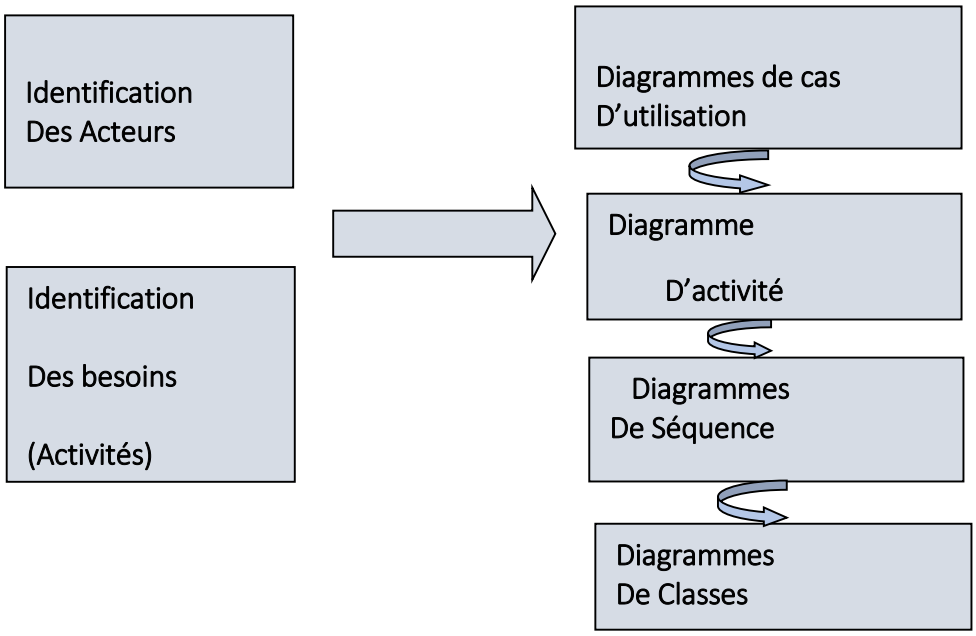


Fig. III.1 : La démarche de modélisation de l'application.

II.7 Extension d'UML pour le web :

L'extension d'UML pour le web définit un ensemble de stéréotypes, d'étiquettes et de contraintes, qui rend possible la modélisation d'application web. Ces stéréotypes et ces contraintes sont appliqués sur certains des composants propres aux applications web, permettant ainsi de les représenter au sein du même modèle et sur les mêmes diagrammes que ceux qui décrivent le reste du système. Ainsi nous avons opté pour cette méthode pour modéliser notre application. Donc nous commencerons par l'élaboration des diagrammes de cas d'utilisations pour la spécification des besoins de notre système, les diagrammes de séquence pour le modèle d'analyse ainsi que les diagrammes de classe pour la conception.

- **Un stéréotype :** C'est une extension du vocabulaire d'UML, il permet d'associer une nouvelle signification à un élément du modèle.
- **Une étiquette :** C'est une extension des propriétés d'un élément. Elle permet la description d'une nouvelle propriété d'un élément d'un modèle.
- **Une contrainte :** C'est une extension de la sémantique d'UML.

III. Analyse et conception de l`application :

La modélisation UML s'articule autour de deux étapes :

L'étape d'analyse.

L'étape de conception.

III.1 Analyse :

L'activité d'analyse débute par la spécification des besoins fonctionnels du système, en identifiant les acteurs et leurs différentes tâches ainsi que les scénarios. Le résultat est un diagramme de cas d'utilisation traduisant la dynamique du système et qui sera utilisé par la suite dans l'activité de conception.

III.1.1 Spécification des besoins :

III.1.1.1 Identification des acteurs :

a. Définition :

Un acteur représente un rôle que peut jouer l'utilisateur ou toute entité externe pouvant interagir avec le système. Autrement dit c'est un objet actif qui utilise les fonctions du système.

b. Extraction des acteurs :

Administrateur : la personne chargée du maintien et de la mise à jour du l'application.

III.1.1.2 Identification des cas d'utilisations :

a. Spécification des tâches :

Une tâche est l'ensemble des différentes fonctions aux quelles un acteur peut accéder. Les acteurs définis précédemment effectuent un certain nombre de tâches, ces taches sont résumées dans le tableau ci-dessous :

Acteurs	**Taches**
Utilisateur	**T1 :** accède à l'interface d'accueil. **T2** : la segmentation **T3 :** le prétraitement. **T4 :** l'apprentissage. **T5 :** la reconnaissance. **T6 :** Quitter

Fig. III.2 : Spécification des tâches.

b. Spécification des scénarios

C'est une succession particulière d'actions, s'exécutant du début à la fin du cas d'utilisation Chaque fois qu'une instance d'un acteur déclenche un cas d'utilisation, un scenario est créé. Ce scenario suivra un chemin particulier dans le cas d'utilisation.

Acteurs	Taches	Scénarios
Utilisateur	T1 : accède à l'interface d'accueil.	**S0 :** lancer l'application
	T2 : la segmentation	**S1 :** segmentation du texte en lignes **S2 :** segmentation de la ligne en mots **S3 :** segmentation du mot en caractères
	T3 : le prétraitement	**S4 :** cliquer sur le bouton prétraitement. **S5 :** sélectionner l'image. **S6 :** lancer le prétraitement
	T4 : l'apprentissage.	**S7 :** cliquer sur le bouton apprentissage. **S8 :** sélectionner l'image de la lettre. **S9 :** sélectionner le type de la lettre. **S10 :** lancer l'apprentissage
	T5 :la reconnaissance	**S11 :** reconnaissance du mot **S12 :** reconnaissance de la lettre. **S13 :** retour à la page principale

Fig.III.3 : Spécification des scénarios.

III.2 Conception:

L'analyse comprend les activités qui permettent d'aboutir au modèle d'analyse du système en partant des cas d'utilisation et du besoin fonctionnel. Ces activités ne font aucune supposition sur la manière d'implémentation la solution finale. Pour l'analyse d'un système, on fait appel aux cas d'utilisation, aux diagrammes d'activité et de séquences.

Le diagramme du modèle d'analyse sont formés avec classes traduisant la dynamique du système et qui seront utilisés par la suite dans l'activité de conception.

III.2.1 Diagrammes de classes :

Un diagramme de classe est une collection d'éléments de modèles (statiques), tels que des classes ou des interfaces et leurs relations, connectés entre eux comme un graphe. En effet un diagramme de classe montre uniquement des aspects statiques du modèle et fait abstraction des aspects dynamiques ou temporels, mêmes si les éléments du diagramme de classe peuvent avoir un comportement dynamique important.

Dans ce qui suit nous allons présenter le diagramme de classes globale.

III.2.1.1 Diagramme de classe globale :

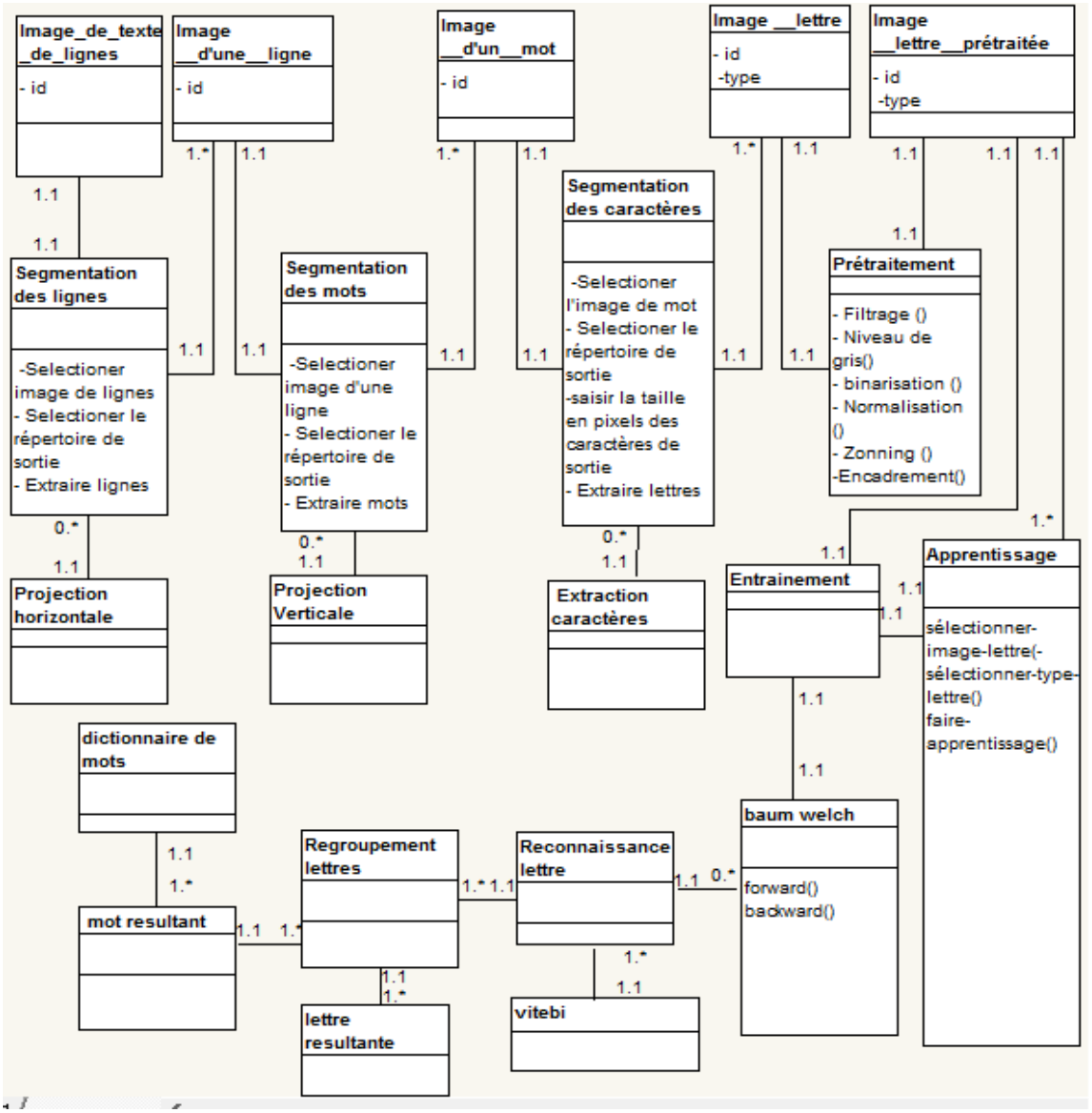


Fig.III.4: Diagramme de class global.

IV. Description de notre système

Notre système prend en entrée les images des lettres manuscrites numérisées pour la phase d'apprentissage et les images des noms manuscrits numérisées concernant la phase de reconnaissance.

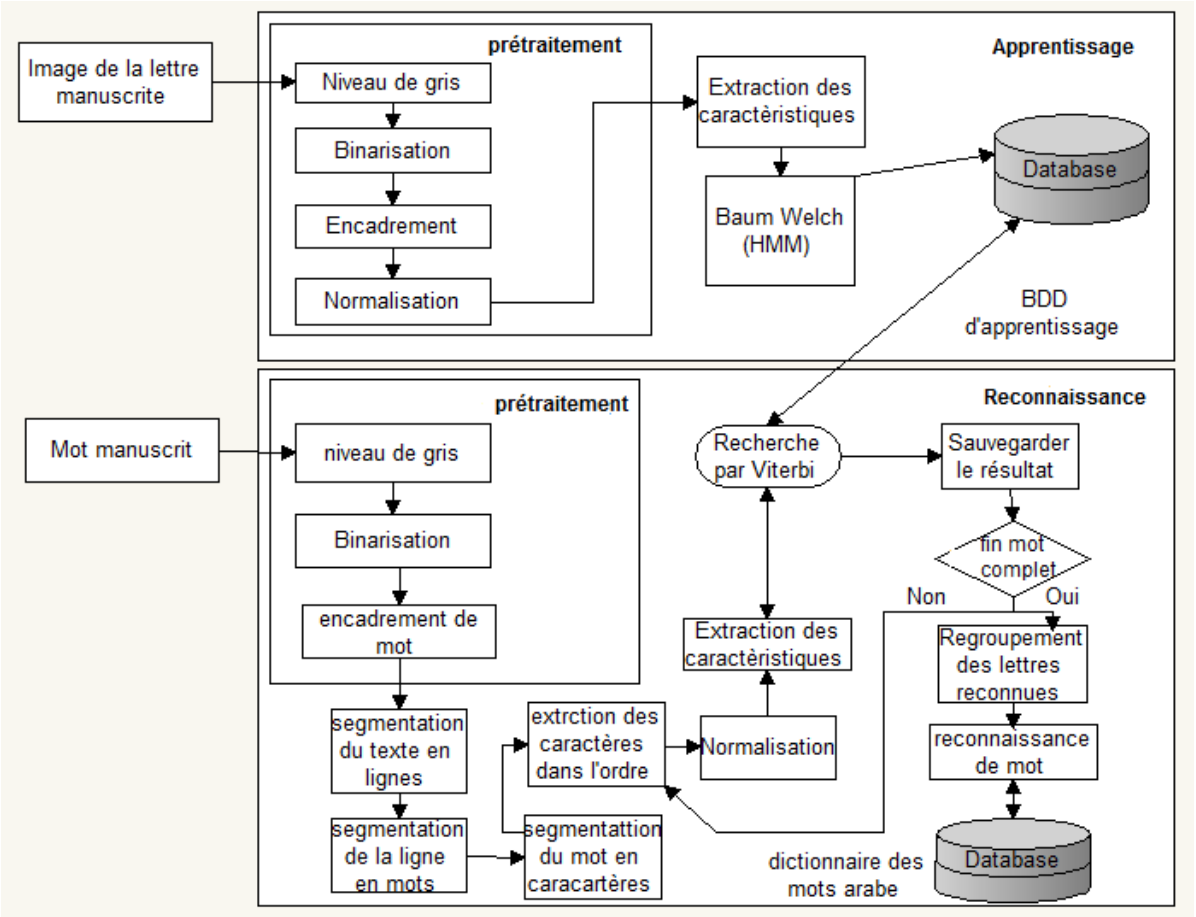


Fig.III.5 : Schéma de la reconnaissance des mots arabes

IV.1 Sous système d 'apprentissage :

Le sous-système d'apprentissage est la première phase dans notre système, ou on va créer la base d'apprentissage des différentes lettres manuscrites arabes pour 1 gramme (une lettre) et 2 gramme (2lettre) afin de reconnaître les mots.

IV.1.1 Prétraitement

Les prétraitements se donnent pour objectif la simplification de la procédure d'extraction de caractéristiques. Les prétraitements qui ont été appliqués sur les images numérisée sont : Niveau de gris, Binarisation, l'encadrement et la normalisation.

1.Niveau de gris

Le niveau de gris est la valeur de l'intensité lumineuse en un point. La couleur du pixel peut prendre des valeurs allant du noir au blanc en passant par un nombre fini de niveaux intermédiaires, en utilisant l'algorithme de niveau de gris [21].

Ouvrir l'image 1

Créer une image résultat de même taille que l'image1.

Pour chaque colonne faire

Pour chaque ligne faire

Extraire les canaux Rouge, Vert et Bleu du pixel de l'image 1.

Calculer la valeur de niveau de gris correspondante (1).

Ecrire la valeur de l'entier RGB dans le pixel de l'image résultat.

Fin pour

La représentation des images en niveaux de gris, fait en attribuant à chaque pixel de l'image une valeur correspondant à la quantité de lumière renvoyée. Cette valeur peut être comprise entre 0 et 255.

2. Binarisation

La Binarisation est effectuée dans le but de simuler le processus d'impression à l'encre noir sur du papier blanc en utilisant l'algorithme de Binarisation [21]. Elle est une opération qui produit deux classes de pixels, en général, ils sont représentés par des pixels noirs et des pixels blancs. Ainsi les pixels correspondant à des points élevés doivent être binarisés en noire (valeur=0) et ceux dans les creux doivent être binarisés en blanc (valeur=1).

Image1 : image d'entrée, Image2 : image de sortie

Pour i de 1 à largeur image1 Faire

Pour j de 1 à hauteur image1

Faire pixel = Image1.pixel (i, j)

SI pixel < seuil

ALORS val=0 Sinon

val=255

Fin si

Image2. Mettre Pixel (i, j, val)

Fin pour

Fin pour

Fin.

3.L'encadrement

L'encadrement (algorithme d'encadrement) [21] .c'est le processus de localisation de la lettre, c'est de définir les coordonnées de la lettre dans l'image.

Nous présentons ci-dessous les algorithmes utilisés :

Initialisation haute = -1, gauche = -1 ; droit = -1 ; bas = -1 ;

Entier indiced, indiceb ;

```
// A gauche

Pour x = 0 à x <img. Largeur faire
        Pour y = haut à y <img. Hauteur faire

         // récupérer la couleur de chaque pixel
          Si ((img. GetRGB (x, y) ! = blanc) et (img. GetRGB (x, y) ! = -1))
              Alors gauche=x ;

      Sortir de la boucle
    Si gauche ! =-1 Alors

      Sortir de la boucle
     FinSi

Fin pour
```

// A Droite

Pour x =img. Largeur-1 ; x >= gauche, x++ faire

Pour y = img. GetHeight () -1 ; y >= haut ; y—faire

// récupérer la couleur de chaque pixel

Si (img. getRGB (x, y) ! =blanc) et (img. getRGB (x, y) ! =-
Alors

Indiced=x ;

droit =img. Largeur - x ;

Sortir de la boucle

Fin Si

Fin Pour

Si (droit ! =-1) Alors

Sortir de la boucle

Fin Si

Fin Pour.

```
//En bas

Pour y = img.hauteur-1à y >haut Faire

    Pour x =0 à x <img. largeurFair
     // récupérer couleur de chaque pixel

Si (img. getRGB (x, y) !=blanc) et (img.getRGB(x, y) !=-1)

                Indiceb=y ;
                bas=img. Hauteur-y ;
                sortir de la boucle ;
Fin Si
Fin Pour
   Si bas! =-1 Alors
     Sortir de la boucle
   Fin Si
Fin Pour
```

```
//L'encadrement
Entiers w=0, h=0 ;

Pour col = gauche à col < indiced faire

   Pour row = haut à row < indiceb faire

        // Recopie de l'image 1 dans l'image résultant

         img2. setRGB (w, h , img.getRGB( col , row )) ;

                  Incrémentation de h ;
    Fin Pour
   Incrémentation de w;
   h=0;
Fin Pour
```

4.Normalisation

La normalisation consiste à transformer la taille de l'image et l'adapter à une dimension fixée a priori par le réalisateur de système. Pour cela nous avons proposés une procédure qui permet de normaliser l'image encadrée dans une dimension de 64x64 pixels. Cette procédure copie le contenu d'une première image pixel par pixel et la copie dans une seconde image rétrécie si l'image dépasse 64x64, sinon elle sera agrandie. Nous avons utilisé une échelle qui sera calculée automatiquement en fonction de la dimension de la lettre encadrée dans l'image par rapport à la dimension 64*64 pixels, dans l'algorithme de normalisation on verra les étapes à suivre [21].

```
Procédure Normalisation (Image1 : Bitmap, Image2 : Bitmap, Echelle : Réel) : Bitmap

Variable  X, Y: Réel ; c :
          entier ; i, j : Réel ;

          Hauteur, Largeur : Réel ;

Début

    Hauteur = bas - haut

   Largeur= droit- gauche

  EchelleH = Hauteur / Image2. Hauteur

  EchelleW = Largeur / Image2. Largeur j = 0 ;


 Pour y de haut à bas (pas de EchelleH) Faire i = 0

Pour x de droit à gauche (pas de EchelleW) Faire e c =
Image1. Point (x, y)

     Image2. PSet (i, j), c

        i = i + 1

  Fin Pour

  j = j + 1 ;

 Fin Pour

Fin
```

IV.1.2 L'extraction des primitives

Après les prétraitements sur l'image de la lettre manuscrite, nous nous intéressons à l'étape de sa caractérisation (extraction des primitives), qui a pour objectif de déterminer le vecteur représentatif de cette image. Ce vecteur servira à représenter de manière compacte la séquence des formes (la lettre) contenues dans l'image. Dans notre système, nous extrayons plutôt des primitives de type statistiques qui conviennent pour le modèle de Markov caché (MMC) grâce à l'algorithme ci-dessous [20]. A partir de l'image normalisée, nous tirons des primitives du type zoning. La méthode zoning consiste à partitionner une image en M sous-images de mêmes tailles appelées zones (8x8). Nous calculons pour chaque zone le pourcentage de pixels noir en obtenant une matrice de pourcentage qui représente le vecteur de primitives, celui-ci sera utilisées pour l'apprentissage et la reconnaissance.

1. *Initialisation à λ_0 des paramètres du MMC.*
2. *Calculer un nouveau jeu de paramètres λ à partir de λ_0.*
3. *Si log ($p(Y_{1:T} \mid \lambda)$) – log ($p(Y_{1:T} \mid \lambda_0)$) <, arrêt des itérations.*
4. *Sinon, le nouveau jeu de paramètres λ prend la place de λ_0 et on*

IV.1.3 Apprentissage

Il s'agit lors de cette étape, d'apprendre au système les propriétés pertinentes du vocabulaire utilisé et de l'organiser en modèles de références. L'idéal serait d'apprendre au système autant d'échantillons que de formes d'écritures différentes, mais cela est impossible à cause de la grande variabilité de l'écriture qui conduirait à une explosion combinatoire de modèles de représentation. La tendance consiste alors à remplacer le nombre par une meilleure qualité des traits caractéristiques. L'apprentissage consiste en deux concepts différents : l'entraînement et l'adaptation. [21]

L'entraînement consiste à enseigner au système la description des caractères tandis que l'adaptation sert à améliorer les performances du système en profitant des expériences précédentes. Certains systèmes permettent à l'utilisateur d'identifier un caractère lorsqu'ils échouent à le reconnaître et ils utilisent l'entrée de l'utilisateur à chaque fois que le caractère est rencontrée. Les procédés d'apprentissage sont différents selon qu'il s'agisse de reconnaissance de caractères imprimés ou manuscrits.

IV.1.3.1 Modèle de Markov Caché

Les MMC permettent de calculer la probabilité d'appartenance d'une forme à une classe.

Il est caractérisé par les matrices de probabilités de transition d'états **A**, de probabilités des symboles d'observation **B** et le vecteur de probabilités d'état initial **Π** ;

Les notations suivantes sont formellement définies pour un **HMM** :[22]

N : nombre d'états ;

M : nombre de symboles d'observation ;

T : la longueur de la séquence d'observation ;

S={s_t} l'ensemble d'états, s_t € {1, 2, …N}, 1<=t<=T,

O={o_k} : l'ensemble discret des symboles d'observations possibles,1<=k<=M

A={a_{ij}} : la distribution de probabilité de transition d'état, où a_{ij}=P (s_{t+1} =j| s_t =i), 1<=i, j<=N,

B={$b_j(o_k)$} : la distribution de probabilité de symbole d'observation où $b_j(v_k)$ = p (o_k à t|s_t=j), 1<=j<=N, 1<=k<=M,

Π= {Π $_i$} : la distribution de probabilité d'état initial où Π $_i$ = p(s_1=i), 1<=i, <=N,

Nous pouvons, d'une façon plus compacte désigner l'ensemble des paramètres **d'un HMM par λ = (A, B, Π),** Par conséquent, un HMM est complètement spécifié par λ.

IV.1.3.2 Le vecteur d'observation

Le vecteur d'observation est le vecteur représentatif de l'image (ou la lettre manuscrite). Ce vecteur servira à représenter de manière compacte la séquence des formes (la lettre) contenues dans l'image. Dans notre système, nous extrayons plutôt des primitives de type statistiques qui conviennent pour le modèle de Markov caché (MMC). A partir de l'image normalisée, nous tirons des primitives du type **zoning.** La méthode zoning consiste à partitionner une image en M sous-images de mêmes tailles appelées zones (8x8). Nous calculons pour chaque zone le pourcentage de pixels noir en obtenant une matrice de pourcentage qui représente le vecteur de primitives, celui-ci sera utilisées pour l'apprentissage et la reconnaissance.

Nous avons élaboré un MMC (Π, A, B) pour chaque lettre de l'alphabet arabe à l'aide de la matrice résultante de la phase d'extraction de primitive.

Nous avons fixé le nombre d'états à N=4, selon les travaux réalisés.

Formellement, la base d'apprentissage (BDD) est un doublet (L, λ) où :

- ✓ L présente la lettre arabe à modéliser par un MMC,
- ✓ λ presente l'ensemble des paramètres d'un MMC (A, B, Π).

  Où :

  A, présente la matrice de destribution de probabilité de transition d'état .

  B, présente la matrice de destribution de probabilité de symbole d'oservation .

  Π, présente la destribution de probabilité d'état initiale .

En effet, (A, B, Π) sont des matrices aux quelles nous avons accordé les valeurs initiales. Ces valeurs nous les avons fixées arbitrairement. vous pouvez referer au :

nText="4" represente nombre d'etat,

pText="0.25,0.25,0.25,0.25"; ou Π,

Tel que :Π est un vecteur à quatres colonnes : Π [i] represente la probabilité d'etre à l'etat i / i=1..4 (i represente nombre d'etat).

mText : represente l'alphabet(symbole d'oservation) qui est dans l'interval [0, 100]. C'est par rapport à la matrice resultante de la methode zoning dont chaque zone represente le pourcentage de pixels noir.

aText ou A. A, représente la matrice de destribution de probabilité de transition d'état . c.a.d la probabilité de transiter de l'etat i à l'etat j.

bText, B, représente la matrice de destribution de probabilité de symboles d'oservations. B[i][j], represente la probabilité d'étre à l'etat i en lisant l'alphabet (symbole d'oservation) j telque j=1..100

L'algorithme de Baum Welch (6-5) qui évalue les probabilités des états en fonction des paramètres courants (Π, A, B) et la séquence de vecteurs d'observation. Il permet la maximisation qui consiste à réestimer les paramètres (Π, A, B) et produit en sortie un HMM (Π, A, B) optimisé.

- **la matrice A**

aText ou A. A, représente la matrice de destribution de probabilité de transition d'état . c.a.d la probabilité de transiter de l'etat i à l'etat j.

Pour les valeurs initiales de cette matrice, nous les avons fixés arbitrairement. L'essentiel de respecter la contrainte que la somme des valeurs de chaque ligne ne dvraient pas depasser 1. Par exemple pour la ligne 1 de **aText** nous avons les valeurs suivantes : 0-0=0.180,0-1=0.274,0-2=0.426,0-3=0.120 dont la somme est 1.

La matrice finale **aText** est donnée par l'algorithme de Baum Welch.

- **La matrice B**

Représente la matrice de distribution de probabilité de symboles d'observations. B[i][j], représente la probabilité d'être à l'état i en lisant l'alphabet (symbole d'observation) j.

C'est pareil que la matrice aText, les valeurs initiales de cette matrice, nous les avons fixés arbitrairement. Et que La matrice finale **bText** est donnée (calculée) par l'algorithme de Baum Welch.
Concernant sa taille, elle est de 4x100. Tel que 100 représente le nombre de symboles qui dans notre cas le pourcentage (regarde la méthode zoning).

IV.1.4 Base de données du notre système

La base de données utilisée est construite à partir des lettres arabes imprimées ainsi que ses paramètres de HMM (Π, A, B). Chaque lettre doit être représentée dans la base sous ces différentes formes (début, milieu, fin, isolée) ainsi pour les lettres de 2 grammes ʎ :

Lettres	**MMC**	Lettres	**MMC**	**Lettres**	**MMC**
Alif D	ʎ1	ra D	ʎ37	fa D	ʎ73
Alif F	ʎ2	ra M	ʎ38	fa M	ʎ74
ba D	ʎ3	za D	ʎ39	fa F	ʎ75
ba M	ʎ4	za M	ʎ40	fa E	ʎ76
ba F	ʎ5	sa D	ʎ41	qa D	ʎ77
ba E	ʎ6	sa M	ʎ42	qa M	ʎ78
ta D	ʎ7	sa F	ʎ43	qa F	ʎ79
ta M	ʎ8	sa E	ʎ44	qa E	ʎ80
ta E	ʎ9	cha D	ʎ45	la D	ʎ81
ta F	ʎ10	cha M	ʎ46	la M	ʎ82
tha D	ʎ11	cha F	ʎ47	la F	ʎ83
tha M	ʎ12	cha E	ʎ48	la E	ʎ84
tha E	ʎ13	ssa D	ʎ49	ma D	ʎ85
tha F	ʎ14	ssa M	ʎ50	ma M	ʎ86
dja D	ʎ15	ssa F	ʎ51	ma F	ʎ87
dja M	ʎ16	ssa E	ʎ52	ma E	ʎ88
dja F	ʎ17	dhha D	ʎ53	na D	ʎ89
dja E	ʎ18	dhha M	ʎ54	na M	ʎ90
da D	ʎ19	dhha E	ʎ55	na F	ʎ91
da M	ʎ20	dhhaF	ʎ56	na E	ʎ92
da F	ʎ21	tta D	ʎ57	hha D	ʎ93
da E	ʎ22	tta M	ʎ58	hha E	ʎ94

tha D	ʎ23	tta F	ʎ59	hha M	ʎ95
tha M	ʎ24	tta E	ʎ60	hha F	ʎ100
tha F	ʎ25	3a D	ʎ61	wa D	ʎ101
tha E	ʎ26	3a M	ʎ62	wa M	ʎ102
dha D	ʎ27	3a F	ʎ63	wa E	ʎ103
dha M	ʎ28	3a E	ʎ64	wa F	ʎ104
dha E	ʎ29	gha D	ʎ65	ya D	ʎ105
dha F	ʎ30	gha M	ʎ66	ya M	ʎ106
3a d la F	ʎ31	La D hha F	ʎ67	dja D hha F	ʎ107
Da D sa F	ʎ32	Hha E da F	ʎ68	3a D la M	ʎ108
Ka M maF	ʎ33	na D maF	ʎ69		
tha D maF	ʎ34	ma D alifF	ʎ70		
Hha D dha F	ʎ35	hha d alifF	ʎ71		
Sa M da F	ʎ36	Sa d ma M	ʎ72		

Fig.III.6 : la base de données de notre système.

IV.2 Le sous-système de reconnaissance

Le sous-système de reconnaissance du nom manuscrit consiste à reconnaitre chaque lettre de nom qui est représentée dans une matrice d'observation. Cette reconnaissance se fait par référence aux modèles de la base de données d'apprentissage par le biais de Viterbi. La reconnaissance de toutes les lettres abouties à la reconnaissance du nom lui-même. Toutes les différentes parties de la reconnaissance seront détaillées dans ce qui suit.

IV.2.1 Prétraitement

Le module de prétraitement dans la phase de reconnaissance des noms contient en plus de celui de la phase d'apprentissage, la correction de l'inclinaison des lignes et des caractères. Ces opérations sont nécessaires puisque l'inclinaison des lignes et l'inclinaison des caractères sont des sources d'erreur classique, relativement gênantes pour la phase de segmentation donc pour la reconnaissance.

IV.2.2 La segmentation

La segmentation permet à partir d'une image acquise (l'image du mot) l'extraction des lignes (segmentation en lignes), ensuite à partir de ces lignes sont extraits les mots ensuite et à la fin les caractères. Cette étape est indispensable dans notre système, puisqu'elle produit en sortie des caractères qui sont la base pour la reconnaissance des mots.

1.La segmentation du texte en lignes

La segmentation du texte en lignes consiste à extraire les lignes à partir d'une image du mot acquise. Cette phase est nécessaire dans le cas où le nom est sur plusieurs lignes. Le principe de notre algorithme est d'utiliser les lignes blanches (les espaces interlignes) pour distinguer et extraire les régions (les lignes) contenant des pixels noir (les parties de nom).

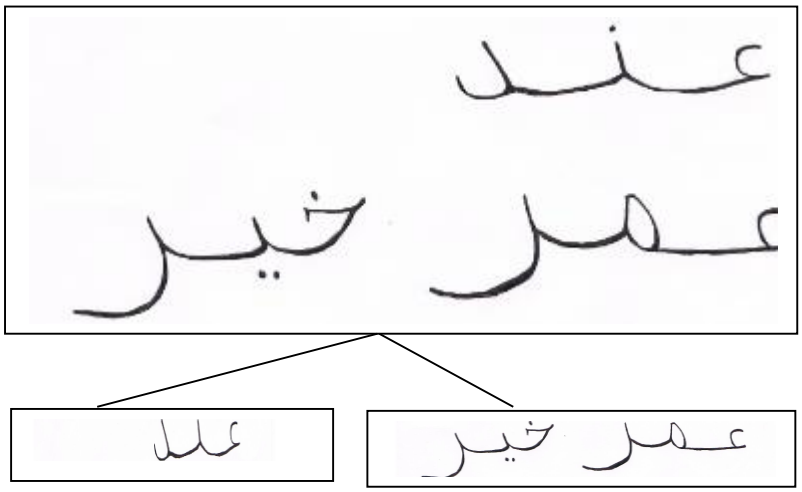


Fig.III.7 : Exemple d'une segmentation du texte en lignes.

2.La segmentation de la ligne en mots

Dans cette phase chaque ligne de texte est segmentée en pseudo-mots (PAWs). Pour cela l'histogramme des projections vertical de la ligne est déterminé afin de trouver les différents blocs qui sont séparés par au moins une colonne vide. Les parties contiguës sont séparées. Ces parties peuvent être des mots entiers, dans le cas où les mots s'écrivent à partir de lettres attachées, ou des parties de mots, dans le cas où les mots contiennent des lettres qui de nature s'écrivent isolés. Les pseudo-mots subissent ensuite un redressement et lissage.

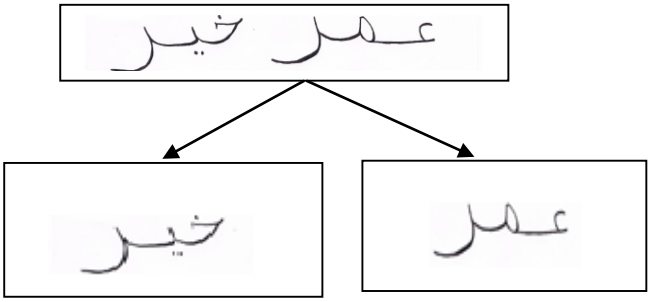


Fig.III.8 : Exemple d'une segmentation de la ligne en mots.

3.La segmentation du mot en caractères

Le but de la segmentation du mot en caractères est de déterminer (approximativement) où débute et où prend fin chaque caractère dans un mot.

Manuscrites par exemples : les problèmes de discontinuité de l'écriture, les ligatures, de chevauchement et d'accolement de pseudo mots, de grandes variabilité inter et intra scripteur, de variation de dimension des pseudos mots ainsi que les signes diacritiques... « Il n'est pas évident de juger de l'efficacité d'un algorithme de segmentation en lettres (4), le résultat peut être décevant ».

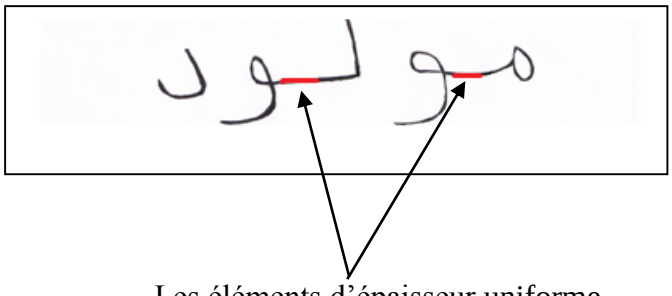


Fig.III.9 : Représentation des éléments d'épaisseur uniforme.

IV.2.3 Normalisation et extraction des caractéristiques

Après la segmentation, nous faisons une opération de normalisation de chaque caractère pour l'adapter à une dimension de 64*64 pixels, puis nous allons construire son vecteur de primitives qui va être traité par Viterbi pour trouver la lettre correspondante dans la base de données d'apprentissage.

VI.2.4 La reconnaissance par Viterbi

Après les phases de normalisation et l'extraction de vecteur d'observation de la lettre, ce dernier sera donné en entré de l'algorithme de Viterbi pour être comparé aux modèles λk de la base de données réalisée dans la phase d'apprentissage, k=1, n. Le modèle retenu sera celui qui fournira la meilleure probabilité correspondante à l'évolution de sa suite de primitives, c'est-à-dire : (5)

V = arg max [P (O/ λk)], 1 <= V <= n Où

O : présente l'observation qui correspond à la lettre.

λk : présente les paramètres de MMC (Πk, A k, B k).

L'algorithme de Viterbi qui fait la comparaison entre la base de données de l'apprentissage et le dictionnaire de mot déjà créer dans la base de données que nous avons utilisé pour la reconnaissance des noms arabes se trouve allégé, en effet il n'y a pas une suite d'états emprunté pour arriver à la probabilité maximale. La découverte du modèle donnant la plus forte probabilité P (O/ λ) est suffisante pour identifier la lettre. De plus, sachant que les matrices de MMC A et B sont connues à l'avance, on peut optimiser l'algorithme de reconnaissance en les transformant (A, B) en valeur logarithmique.

Remarque : on suit les mêmes étapes déjà cité pour (1 grammes) pour une image de 2 grammes (2 lettres).

V. Conclusion

Dans ce chapitre, nous avons présenté la conception de l'application en nous appuyant sur une démarche de modélisation basée sur la méthode UML et on a donné la description de notre système.

Nous avons entamé le présent chapitre par la spécification des besoins et la présentation de divers cas d'utilisations, ceci pour la phase analyse, et nous avons conclu par un diagramme de classe globale.et par la fin on a décrit le déroulement de notre système et les algorithmes utilisés.

Dans le dernier chapitre, qui va suivre, seront définie les outils et les langages de programmation qui ont servies à mettre en œuvre l'application ainsi que la présentation de certaines interfaces ainsi les résultats expérimentaux de la segmentation et la reconnaissance.

CHAPITRE IV
RÉALISATION
ET
IMPLÉMENTATION

CHAPITRE IV : IMPLEMENTATION ET REALISATION

I. Introduction

Cette partie du rapport consiste en l'implémentation et la réalisation des différentes fonctionnalités de l'application. C'est la traduction de la conception en code source exécutable. Dans ce chapitre nous allons présenter l'environnement de développement et les différents outils utilisés pour réaliser notre application, puis expliquer son fonctionnement en présentant quelques interfaces illustratives.

II. L'environnement de travail :

II.1 Système d'exploitation utilisé :

Pour la réalisation de notre projet, nous avons utilisé le système d'exploitation **Windows 7(Seven).** Ce choix se justifie essentiellement par :

- Une interface très facile à utiliser.
- Une rapidité de développement des applications.
- Est un système avec lequel on travail usuellement.

Windows 7

Est un système d'exploitation d'ordinateur personnel développé par Microsoft, une version de Windows NT. Développement de 7 s'est produit dès 2006 sous le nom de code « Blackcomb » Windows 7 a été libéré à la fabrication le 22 Juillet 2009 et est devenu généralement disponible le 22 Octobre 2009, de moins de trois ans après la sortie de son prédécesseur, Windows Vista. L'homologue de serveur de Windows 7, Windows Server 2008 R2, a été libéré en même temps.

II.2 Présentation du langage de programmation (java) :

Java est le langage phare de la société Sun Microsystems. Il a été créé en 1991, à l'époque dans le but d'intégrer des appareils domestiques pour un projet de domotique, dont le nom de code était « Green ». Il fallait donc que le langage soit léger et portable sur toute configuration. Baptisé dans un premier temps « Oak », il changea rapidement de nom, Oak étant déjà utilisé, pour devenir « Java », terme de l'argot en anglais qui signifie café. Les versions suivantes, avec l'ajout notamment de Swing permettant de créer des interfaces graphiques stables, rendront le langage de plus en plus populaire pour la création de nombreux types d'application, avec comme atout principal : sa portabilité.

Java se décline en de nombreuses versions, pour ordinateurs (avec une version de la machine virtuelle par système d'exploitation), pour téléphones mobiles, pour la programmation d'applications commerciales… Chacune des versions possède certaines bibliothèques en commun et certaines autres spécifiques, écrites différemment ou absentes. Certains systèmes d'exploitation possèdent la machine virtuelle dans leur configuration de base (Mac OS X…), tandis qu'elle doit être installée séparément sur d'autres (Windows, Mac OS 9…)

Java ressemble en plusieurs points au C++, Java est donc un langage objet ; toutefois, étant prévu pour tourner sur des petites configurations (souvenons-nous qu'il soit prévu à la base

pour les appareils ménagers), il est aussi utilisé à la manière d'un langage procédural. Il possède plusieurs autres caractéristiques, dont le fait d'être fortement typé (le type des variables en détermine strictement le type de contenu).

Il est aussi, et c'est là une de ses grandes forces, portable (quasiment) sans modifier une seule ligne du code.

II.3. Les avantages du JAVA

Les avantages de Java sont nombreux. Le Byte-code, tout d'abord, qui assure à Java une Portabilité complète vers de très nombreux systèmes**.**

L'importance de l'api de base qui offre tous les services de base, notamment pour la construction des interfaces graphiques.

La 3ème force de Java, c'est son adaptabilité dans de nombreux domaines, autant pour le web que pour les systèmes embarqués.

II.4. Présentation de l'environnement de développement (NetBeans) :

II.4.1 Qu'est-ce que NetBeans ?

NetBeans est un projet open source ayant un succès et une base d'utilisateur très large, une communauté en croissance constante, et près 100 partenaires mondiaux et des centaines de milliers d'utilisateur à travers le monde. Sun Microsystems a fondé le projet open source NetBeans en Juin 2000 et continue d'être le sponsor principal du projet.

Aujourd'hui, deux projets existent : L'EDI NetBeans et la Plateforme NetBeans. L'EDI est un environnement de développement - un outil pour les programmeurs pour écrire, compiler, déboguer et déployer des programmes. Il est écrit en Java - mais peut supporter n'importe quel langage de programmation. Il y a également un grand nombre de modules pour étendre l'EDI NetBeans. L'EDI NetBeans est un produit gratuit, sans aucune restriction quant à son usage.

Également disponible, La Plateforme NetBeans ; une fondation modulable et extensible utilisée comme brique logicielle pour la création d'applications bureautiques. Les partenaires privilégiés fournissent des modules à valeurs rajoutées qui s'intègrent facilement à la Plateforme et peuvent être utilisés pour développer ses propres outils et solutions. Les deux produits sont open source et gratuits pour un usage commercial et non-commercial.

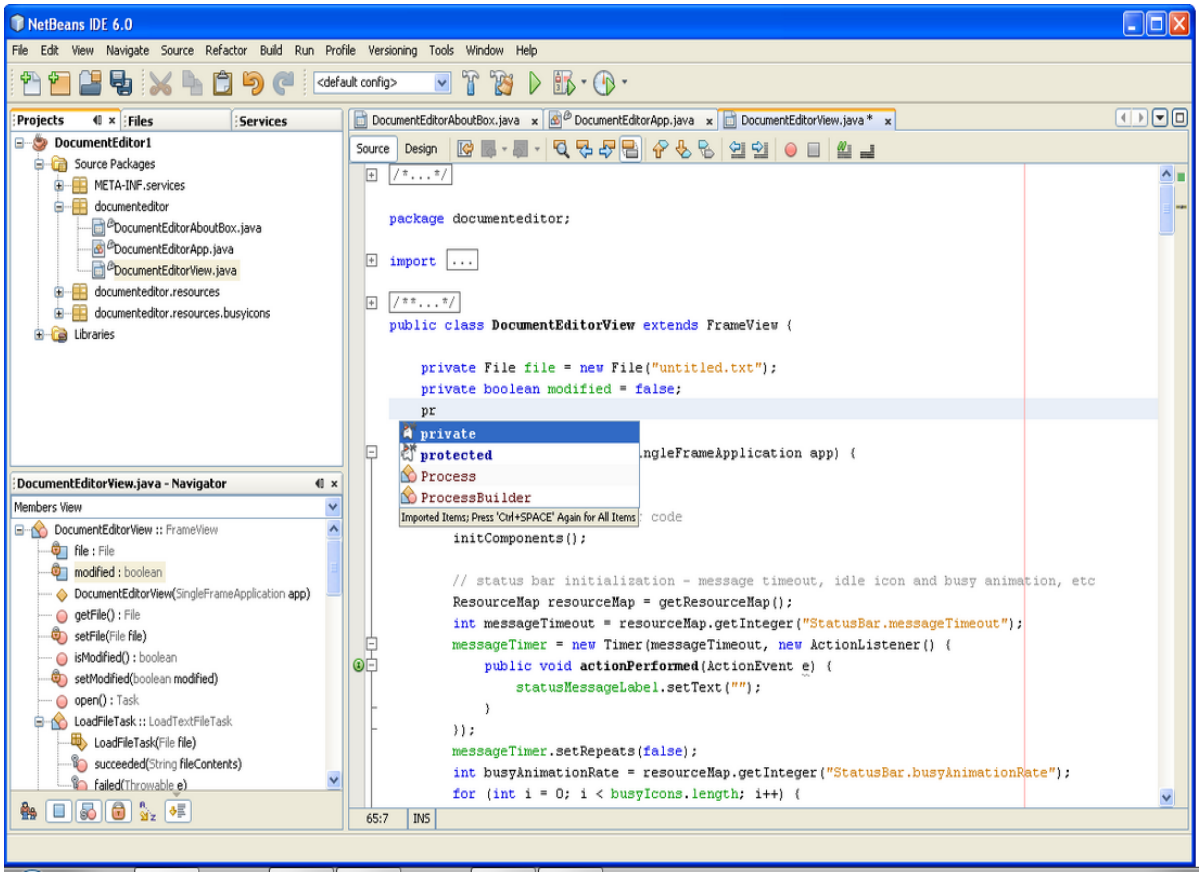


Fig.IV.1 : Environnement de développement NetBeans ID8.0

II.5. Environnement de base :

L'environnement de base comprend les fonctions générales suivantes :

- Configuration et gestion de l'interface graphique des utilisateurs.
- Support de différents langages de programmation.
- Traitement du code source (édition, navigation, formatage, inspection...).
- Fonctions d'import/export depuis et vers d'autres IDE, tels qu'Eclipse ou JBuilder.
- Accès et gestion de bases de données, serveurs Web, ressources partagées.
- Gestion de tâches (à faire, suivi …).

II.6. Présentation du JDK utilisé (JDK6 Windows) :

Le Java Développement Kit (JDK**)** est l'environnement dans lequel le code Java est compilé pour être transformé en byte code (code intermédiaire plus concret que le code machine non directement exécutable) afin que la machine virtuelle Java (JVM) puisse l'interpréter.

Les composants primaires du JDK sont une sélection d'outils incluant :

- Javac – le compilateur, qui convertit le code source en fichier .class (contenant le bytecodeJava)
- Jar – l'archiveur, qui met sous forme d'un paquetage unique l'ensemble des fichiers class en un fichier JAR,
- Java doc– le générateur de documentation, qui génère automatiquement de la documentation à partir des commentaires du code source,
- Jdb – le débogueur.

L'environnement d'exécution Java fait également partie du JDK.

II.7 Le SGBD MYSQL

MYSQL est un véritable serveur de base de données SQL (Structured Query Language) qui est un langage de requêtes vers les bases de données exploitant le modèle relationnel. Il en reprend la syntaxe mais n'en conserve pas toute la puissance puisque de nombreuses fonctionnalités de SQL n'apparaissant pas dans MYSQL (sélection imbriquées, clés étrangères…).

III. Présentation de notre système

Le système de reconnaissance des noms arabes manuscrits que nous avons développé est composé de deux sous-systèmes, un sous-système d'apprentissage et un sous-système de reconnaissance.

III.1. Les interfaces de notre système

III.1.1. Interface d'accueil

C'est la première page visualisée par l'utilisateur de notre système. Cette page contient quatre boutons essentiels :

- **Bouton segmentation** : permettant d'afficher l'interface de segmentation
- **Bouton prétraitement**. : permettant d'afficher l'interface de prétraitement
- **Bouton apprentissage** : permettant d'afficher l'interface d'apprentissage.
- **Bouton reconnaissance** : permettant d'afficher l'interface la reconnaissance.
- **Bouton Quitter** : permettant de sortir de l'interface d'accueil.

La figure suivante représente l'interface d'accueil de notre système :

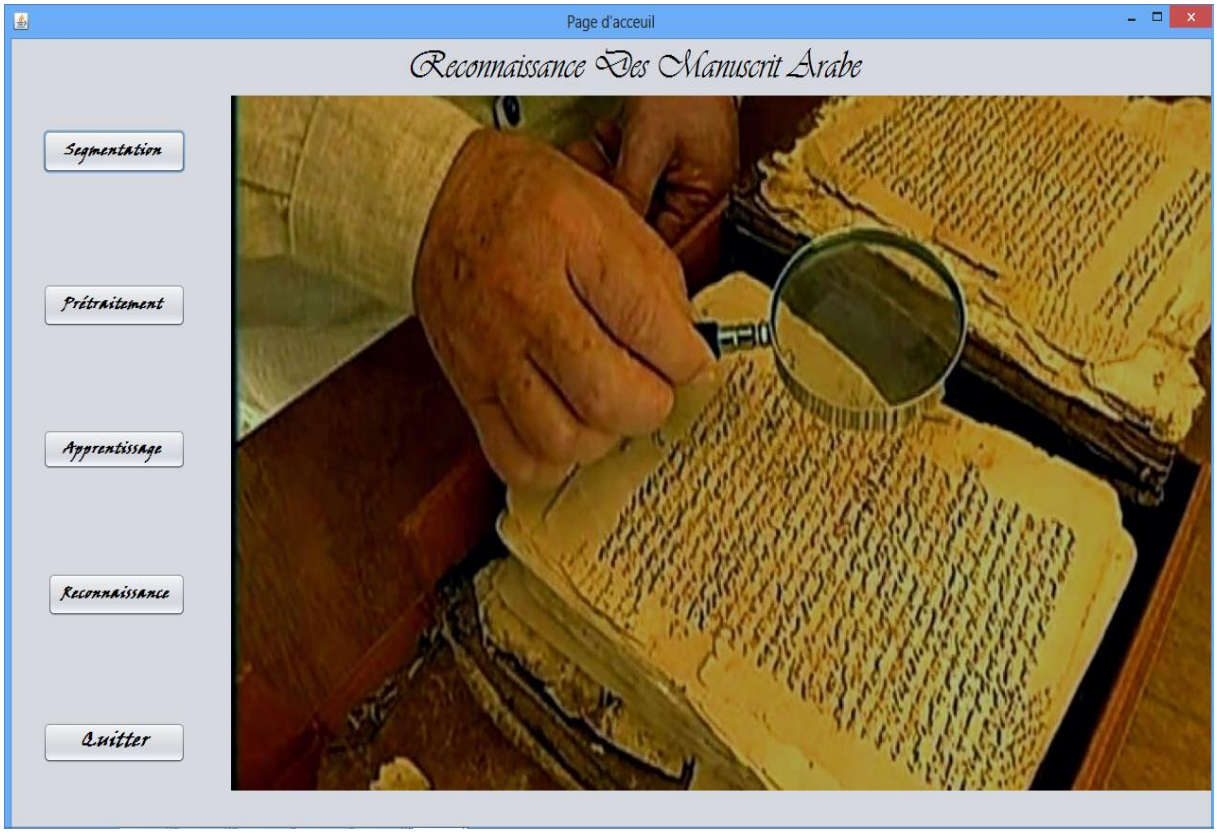


Fig.IV.2 : interface d'accueil.

III.1.2. Interface de la segmentation du texte en lignes

Cette étape permet la segmentation du texte en lignes d'une image. Elle contient :

- **Bouton sélection de l'image d'entrée** : il permet le chargement de l'image du texte à segmenter.
- **Bouton extraire les lignes** : il permet de déclencher l'opération de segmentation en ligne
- **Bouton sélection de répertoire de sortie** : il permet de sélectionner le répertoire où les résultats de segmentation seront enregistrés.

La figure qui suit présente l'interface de segmentation du texte en lignes :

Fig.IV.3 : interface segmentation du texte en lignes.

III.1.3 Interface de prétraitement

Elle permet d'effectuer les prétraitements sur l'image de la lettre et l'extraction de primitives (matrice d'observations). Elle contient deux boutons :

- **Bouton sélection** : il permet le chargement de l'image.
- **Bouton lancement de prétraitement** : il permet de déclencher les opérations de prétraitement et l'extraction de primitives.

Le résultat de l'extraction de primitives sera affiché dans la zone de texte, et l'image prétraitée sera affichée sur écran.

La figure suivante présente l'interface de prétraitement

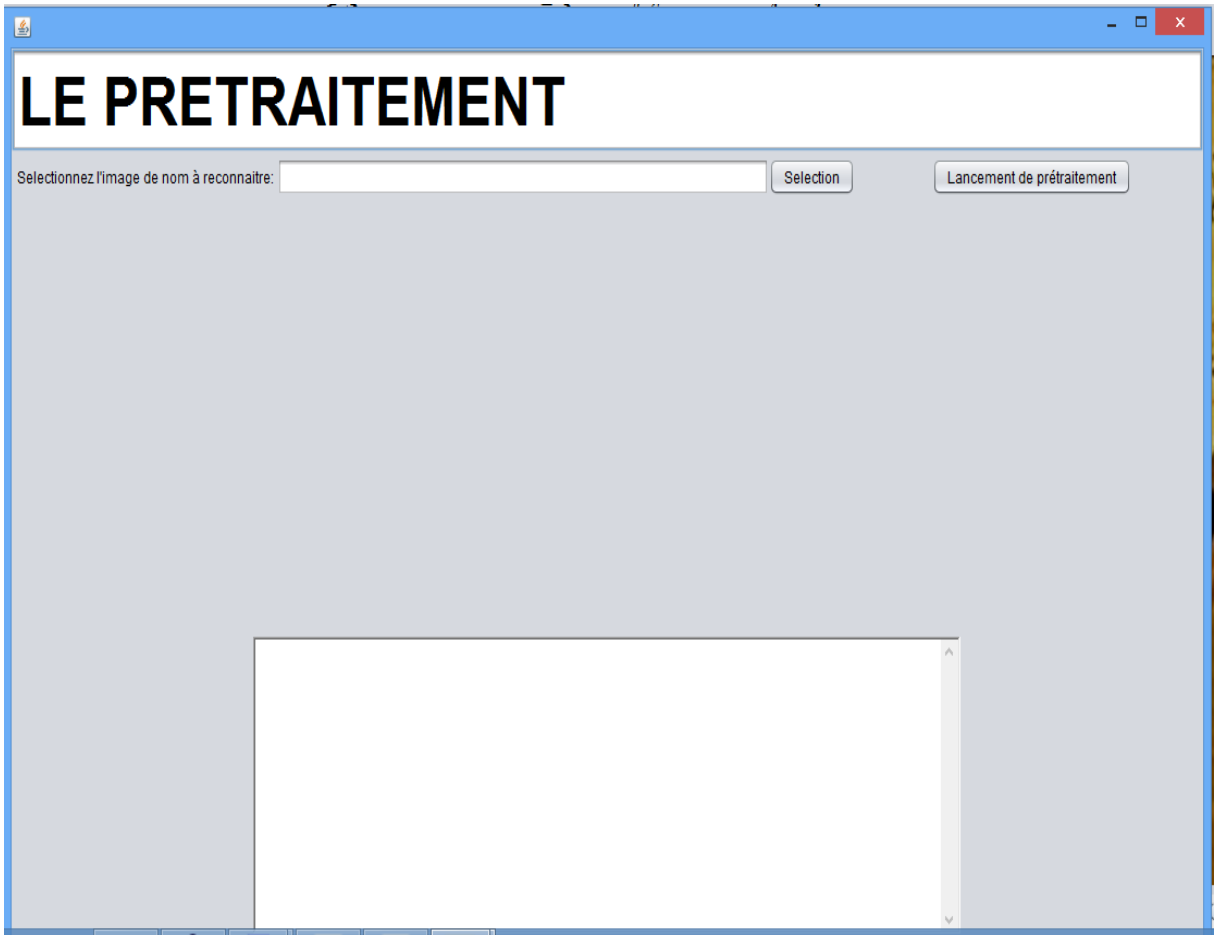


Fig.IV.4 : interface prétraitement**.**

III.1.4. Interface d'apprentissage

Cette interface permet l'apprentissage de l'image de la lettre prétraitée en appliquant l'algorithme de Baum Weltch. Elle contient :

- **Bouton sélection** : il permet le chargement de l'image.
- **Bouton d'apprentissage** : il permet de déclencher l'opération d'apprentissage.
- **Listes de choix** : qui permet le choix de type de la lettre.

Les résultats de l'apprentissage seront affichés dans la zone de texte.

La figure suivante montre l'interface d'apprentissage :

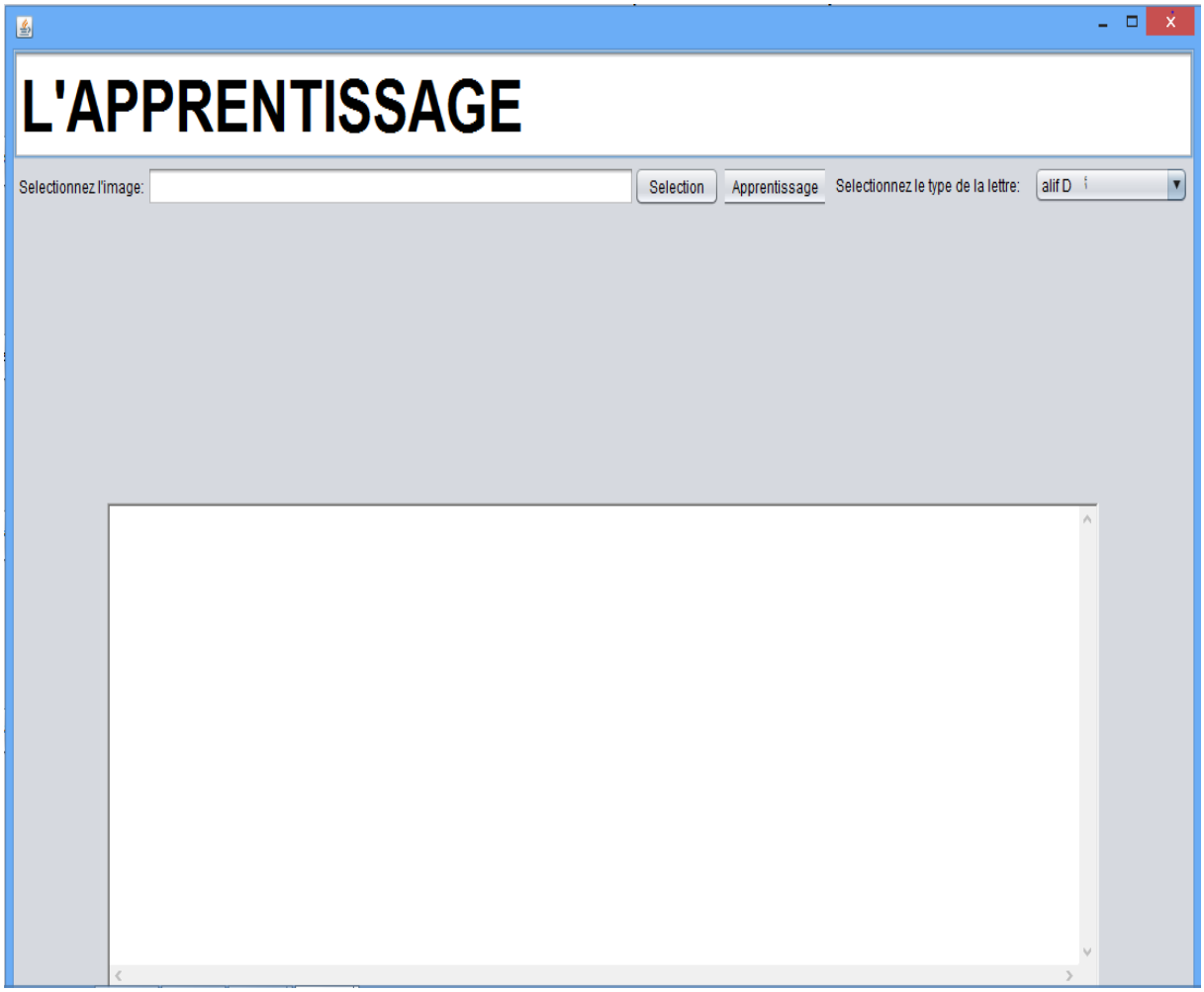


Fig.IV.5 : interface Apprentissage.

III.1.5. Interface de reconnaissance

Cette interface permet de passer à la phase reconnaissance ou on passe par plusieurs étapes (bouton) :

- **Bouton reconnaissance de la lettre :** Permettant de faire reconnaissance de la lettre.
- **Bouton reconnaissance du mot.** Permettant de faire reconnaissance du mot
- **Bouton retour :** permettant de revenir à l'interface précédentes.

La figure suivante montre l'interface de reconnaissance :

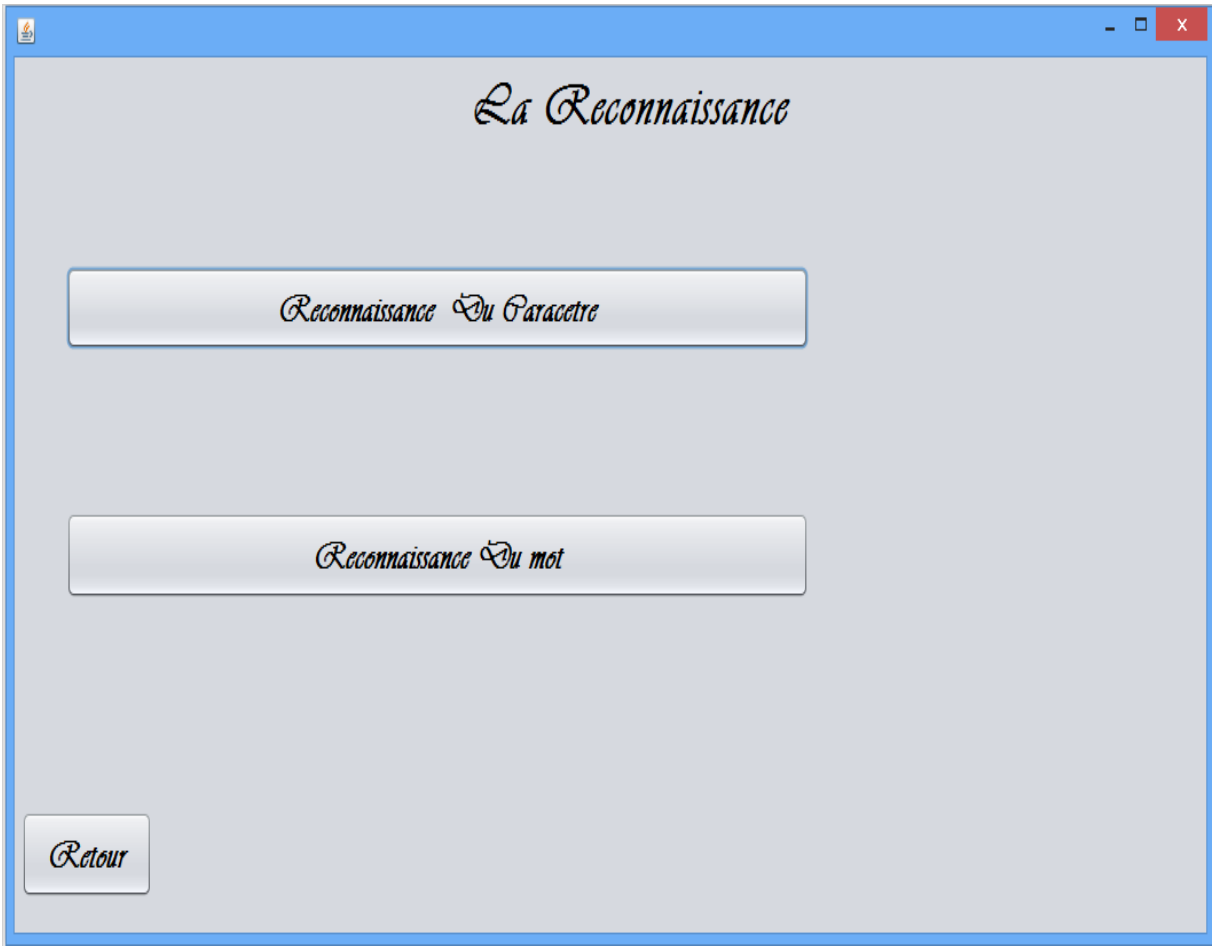


Fig.IV.6 : interface de Reconnaissance.

III.1.6. Interface reconnaissance de la lettre

Cette interface permet d'effectuer la reconnaissance de la lettre en appliquant l'algorithme de Viterbi. Elle contient :

- **Bouton sélection** : il permet le chargement de l'image de la lettre.
- **Bouton reconnaissance** : il permet de déclencher l'opération de reconnaissance de la lettre.

Les résultats de la reconnaissance de la lettre seront affichés dans la zone de texte.

La figure suivante présente l'interface de la reconnaissance de la lettre :

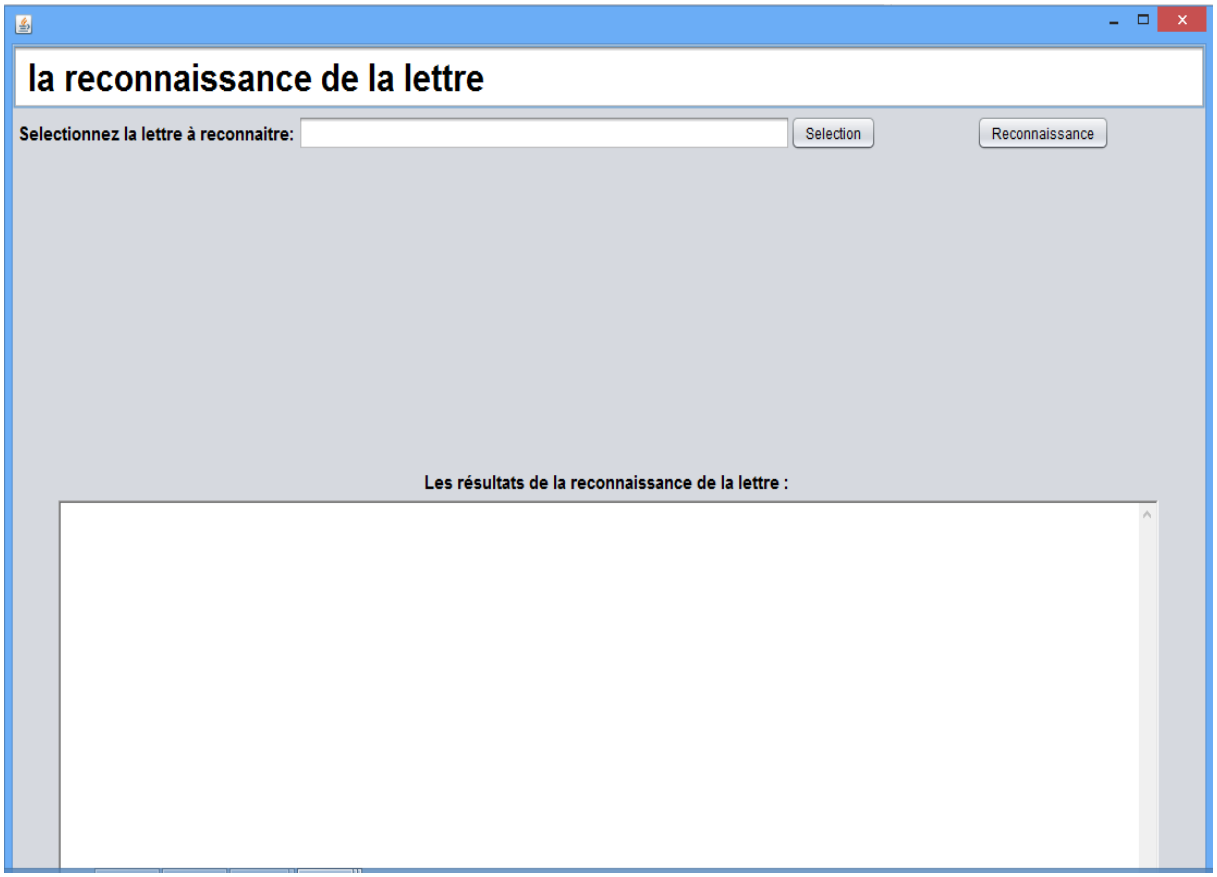


Fig.IV.7 : interface reconnaissance de la lettre.

III.1.7. Interface de reconnaissance du mot

Cette interface permet d'effectuer la reconnaissance du mot. Elle contient :

- **Bouton sélection** : il permet le chargement de l'image du mot à reconnaitre.
- **Bouton reconnaissance du mot** : il permet de déclencher l'opération de reconnaissance du mot.

Le résultat sera affiché dans la zone de texte.

La figure suivante présente l'interface de la reconnaissance du mot :

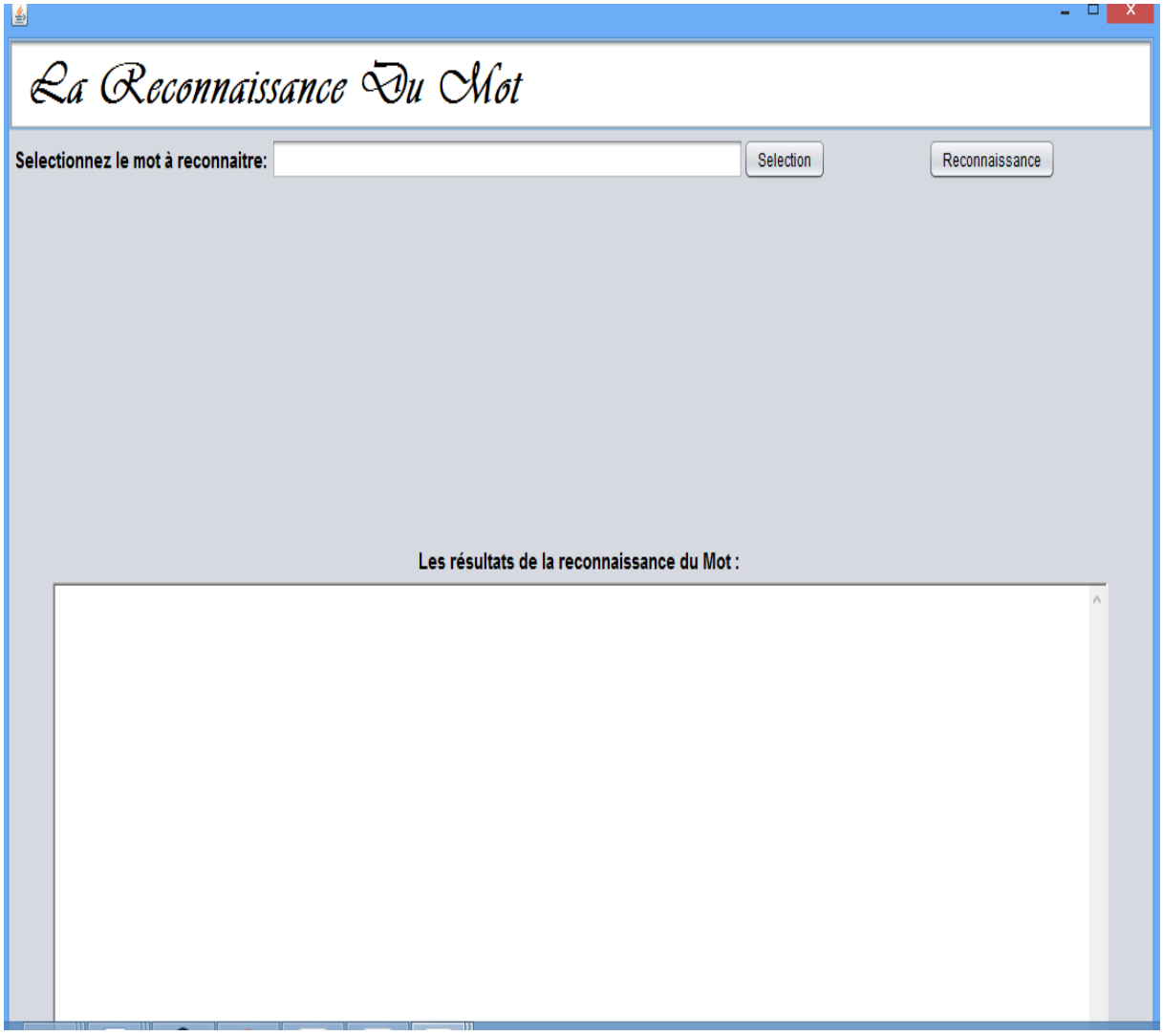


Fig.IV.8 : interface reconnaissance du mot.

IV. Le fonctionnement de système

Dans ce qui suit, nous présentons le déroulement d'un exemple sur notre système afin de

Comprendre son fonctionnement.

IV.1 Sous système d'apprentissage

Il s'agit lors de cette étape d'apprendre au système les propriétés pertinentes du vocabulaire

(Les différentes lettres arabes de 1 gramme à 2 gramme) et de l'organiser en modèle de référence (base de données d'apprentissage). Chaque lettre est représentée dans la base de données selon sa position dans Le mot : début, milieu, fin ou isolée.

IV.1.1 apprentissage Pour 1 gramme :

Avant l'apprentissage d'une lettre, elle doit d'abord passer par l'étape de prétraitement et L'extraction de primitives. Dans ce qui suit nous présentons un teste sur la lettre « hha D ».

- **Chargement de l'image** : Il consiste à charger l'image de la lettre à prétraiter.

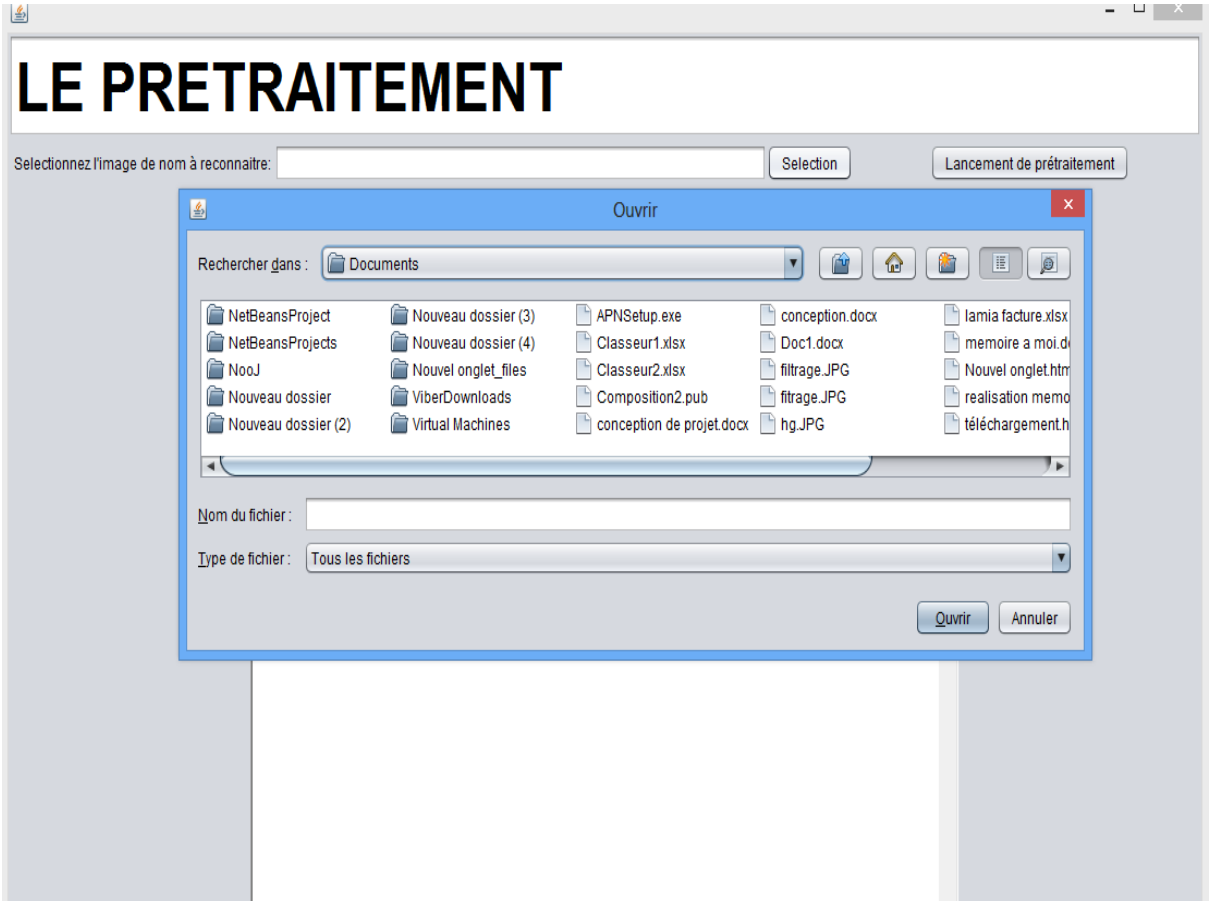


Fig.IV.9 : chargement de l'image de la lettre « hha D ».

- **Prétraitement et l'extraction de primitives** :

Ces opérations sont déclenchées en cliquant sur le bouton de « lancement de prétraitement », les résultats de cette étape sont l'image prétraitée affichée à l'écran et la matrice d'observations de la lettre affichée dans la zone de texte. Ceci est monté dans la figure suivante :

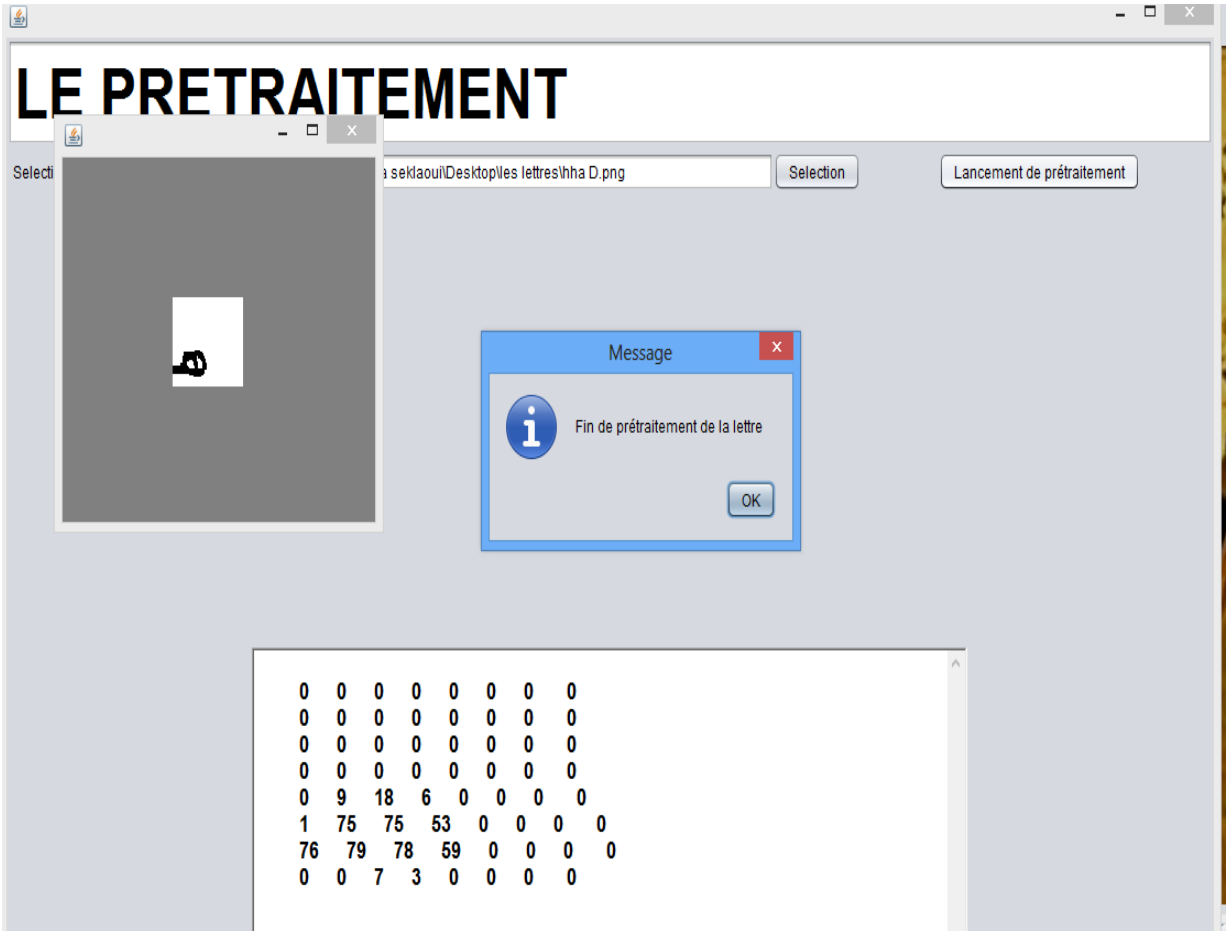


Fig.IV.10 : Les résultats de prétraitement et l'extraction de primitives 1 gramme**.**

- **L'apprentissage** :

Cette opération est effectuée après le chargement de l'image de la lettre et la sélection de son type dans la liste de choix.

Les résultats de l'apprentissage consistent en trois matrices A, B, et P qui correspondent aux paramètres de HMM. Ces matrices seront enregistrées dans la Base de données pour représenter la lettre « hha D »

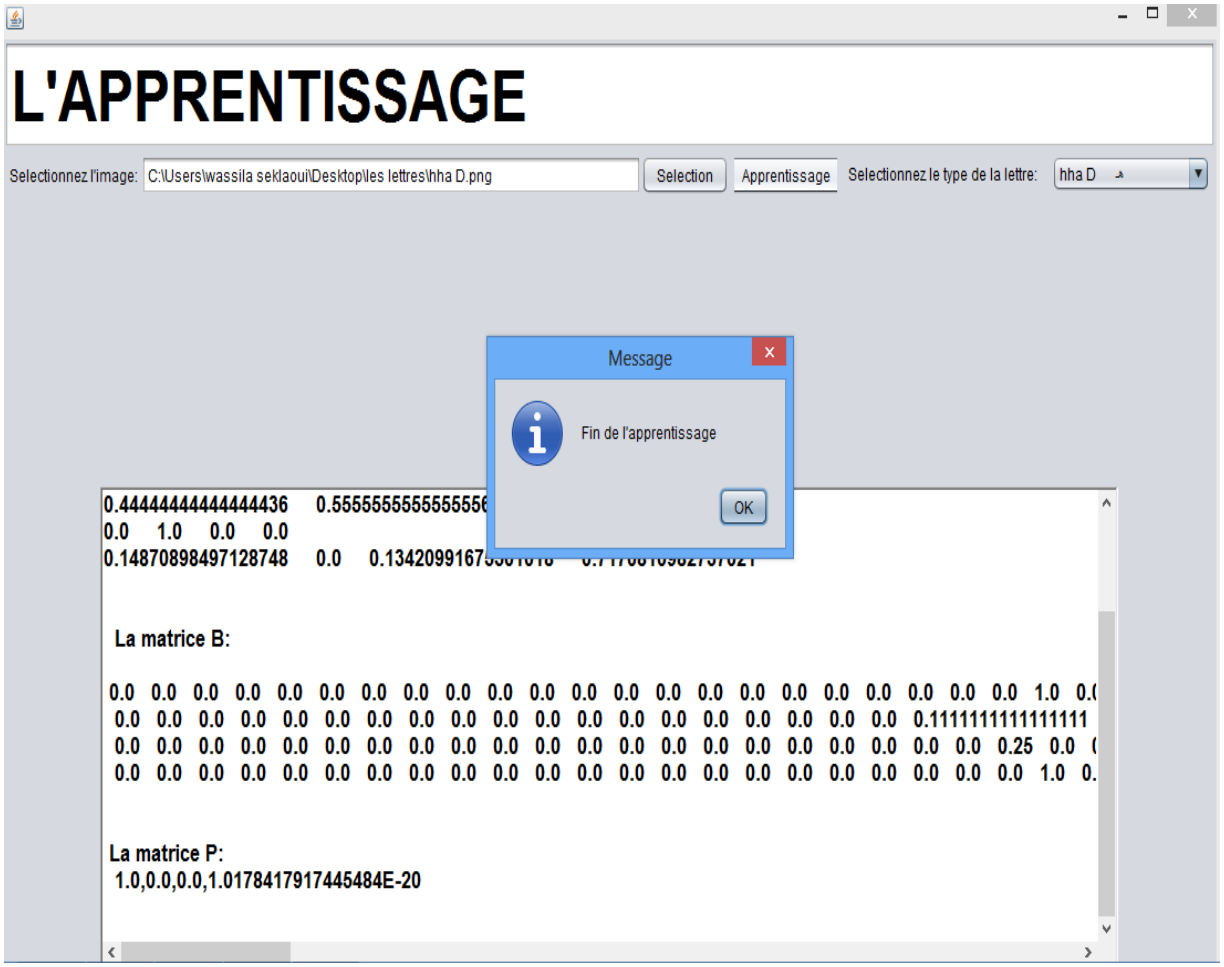


Fig.IV.11 : les résultats de l'apprentissage.

IV.1.2 apprentissage Pour n-gramme

Dans cette étape on va prendre 2 lettres arabe et on va faire l'apprentissage, on doit d'abord passer par l'étape de prétraitement et l'extraction de primitives. Dans ce qui suit nous présentons un teste sur la lettre « hha D alif F ».

- **Chargement de l'image** : même chose avec 1 gramme.
- **Prétraitement et l'extraction de primitives**

Ces opérations sont déclenchées en cliquant sur le bouton de « lancement de prétraitement », les résultats de cette étape sont l'image prétraitée affichée à l'écran et la matrice d'observations de la lettre affichée dans la zone de texte. Ceci est monté dans la figure suivante :

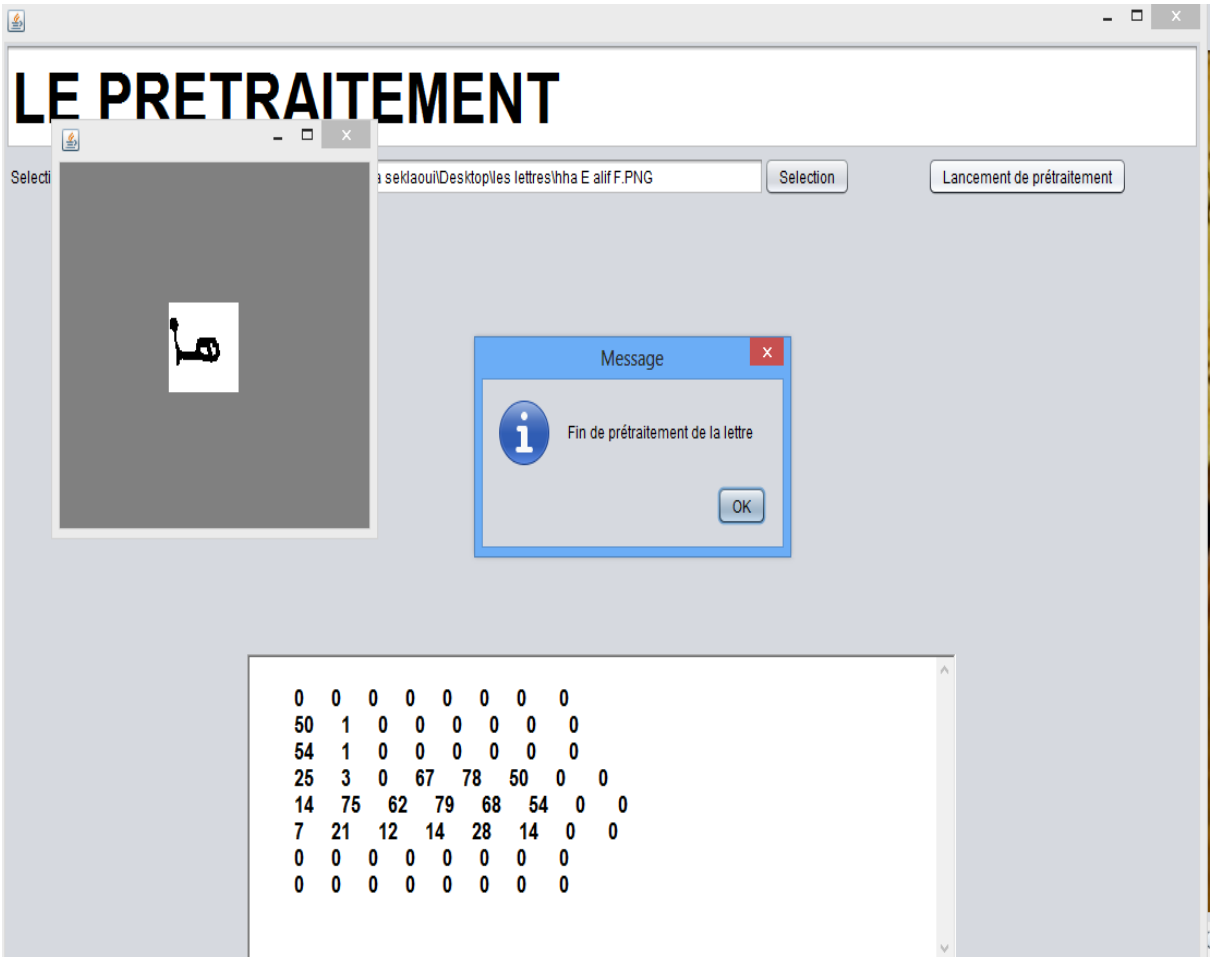


Fig.IV.12 : Les résultats de prétraitement et l'extraction de primitives pour 2grammes.

- **L'apprentissage**

Cette opération est effectuée après le chargement de l'image des deux lettres et la sélection de son type dans la liste de choix.

Les résultats de l'apprentissage consistent en trois matrices A, B, et P qui correspondent aux paramètres de HMM. Ces matrices seront enregistrées dans la Base de données pour représenter la lettre « hha D alif F ».

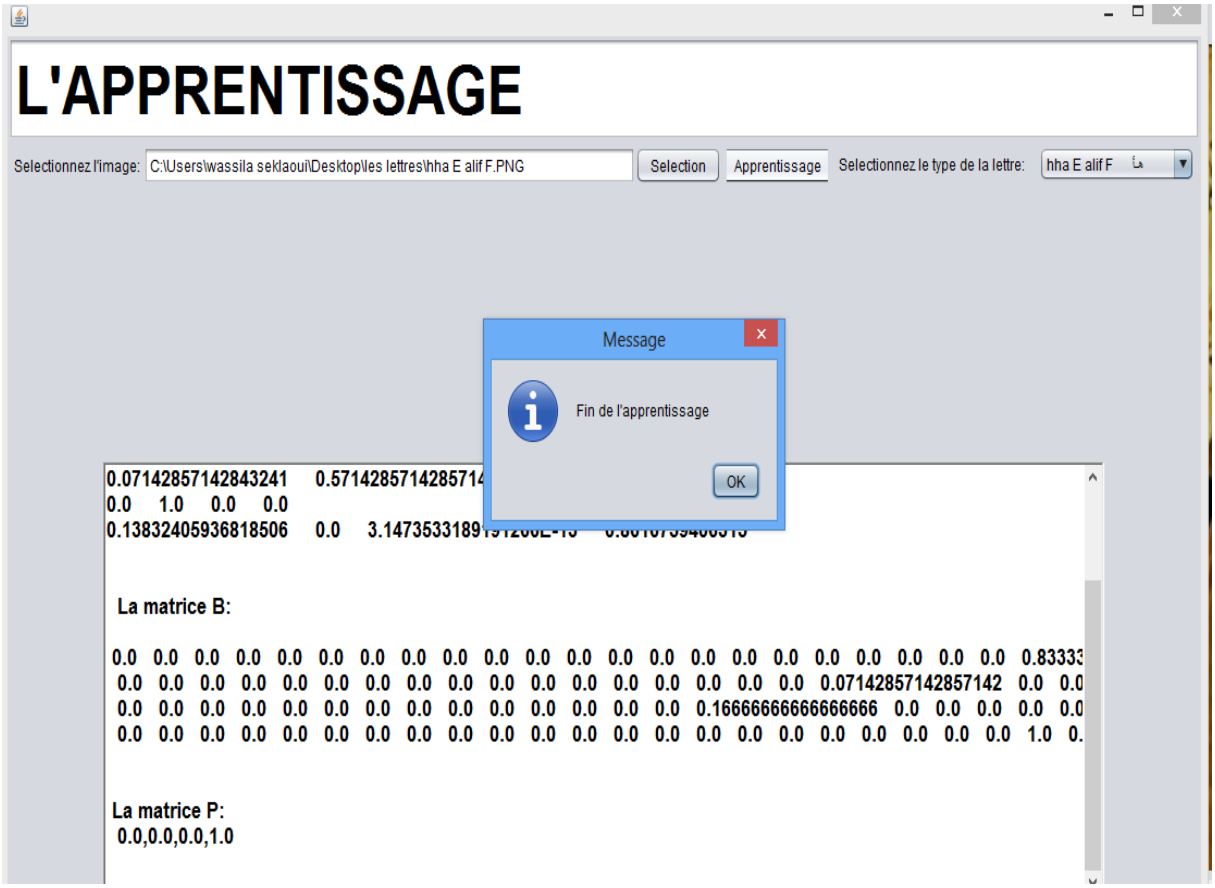


Fig.IV.13 : les résultats de l'apprentissage(2grammes).

IV.2. Sous système de reconnaissance

Le but de cette étape est la reconnaissance du mot. Afin de reconnaitre un mot, on doit D'abord passer par la segmentation en lignes de ce mot, s'il est sur plusieurs lignes, sinon on passe directement à la segmentation en caractères. Après, on passe à la reconnaissance de ces caractères pour reconnaitre le mot globalement.

La figure suivante montre comment faire une segmentation en caractère du mot « harot » :

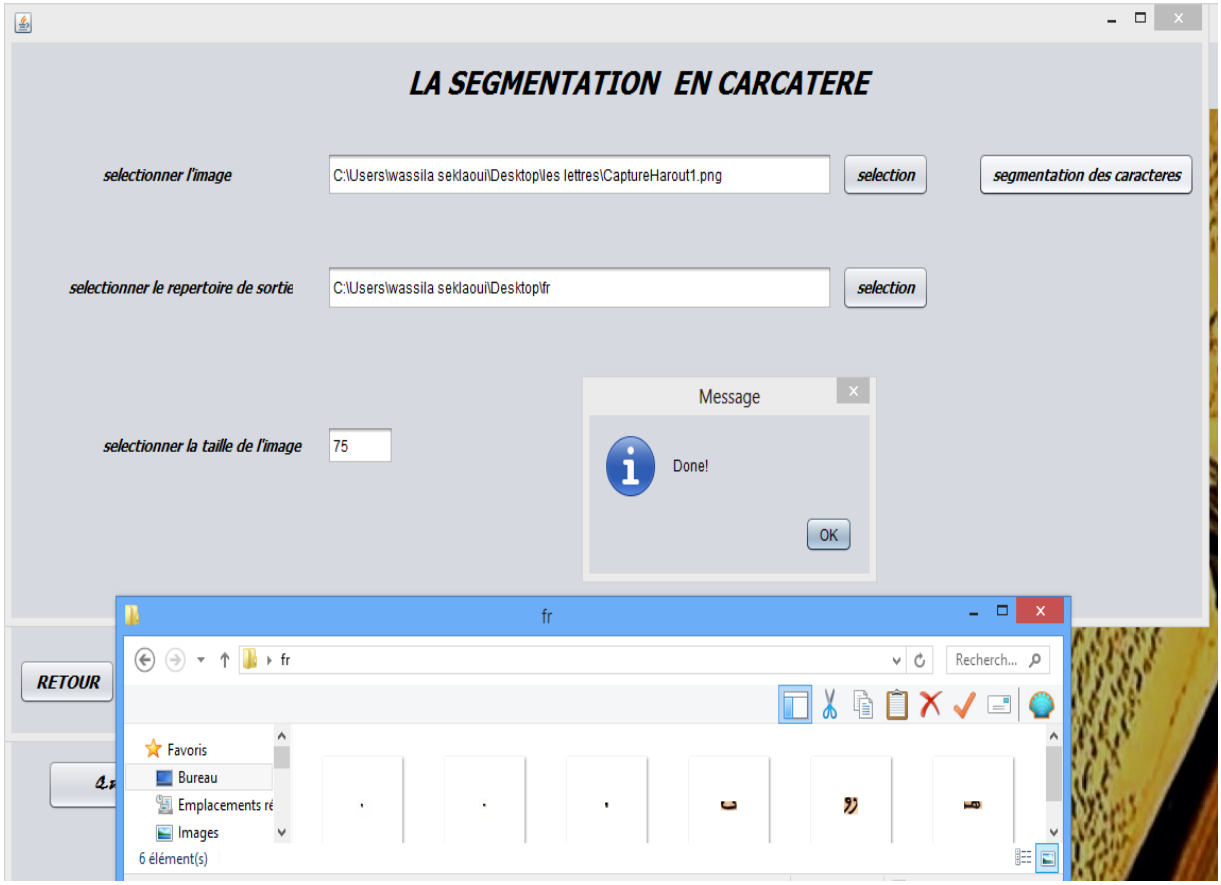


Fig.IV.14 : les résultats de la segmentation du mot en caractère.

IV.2.1 La reconnaissance de la lettre :

IV.2.1.1 reconnaissance pour 1-gramme

Cette étape permet la reconnaissance de la lettre donnée en entré. Après le chargement de la lettre à reconnaitre, en cliquant sur le bouton « reconnaissance », l'opération de reconnaissance sera effectuée et le résultat sera affiché. La lettre correspond à l'image : il représente le mot de la lettre reconnue.

La lettre : elle représente la lettre reconnue en arabe.

La figure suivante représente le résultat de la reconnaissance de la lettre « cha D » extraite de nom donné dans la figure 13 :

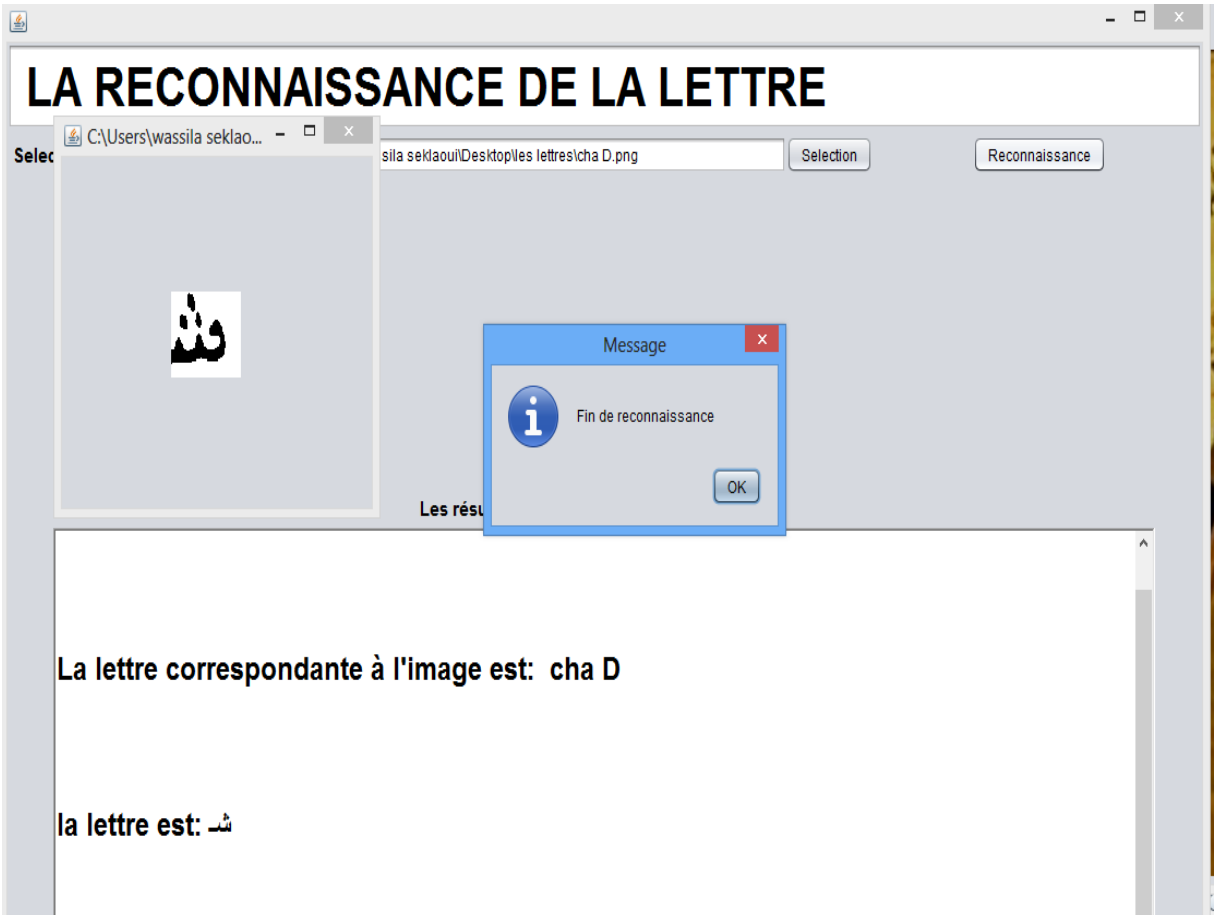


Fig.IV.15 : les résultats de la reconnaissance de caractère.

IV.2.1.2 reconnaissance pour n-gramme

Cette étape permet la reconnaissance de deux caractères donnés en entré. Après le chargement de l'image à reconnaitre, en cliquant sur le bouton « reconnaissance », l'opération de reconnaissance sera effectuée et le résultat sera affiché les deux lettres correspond à l'image : il représente le mot des deux lettres reconnues.

La lettre : elle représente les lettres reconnues en arabe.

Fig.IV.16 : les résultats de la reconnaissance de caractère(2grammes).

V. Résultats Expérimentaux

Afin d'évaluer notre système de reconnaissance du mots arabes, nous avons effectué des expérimentations sur la reconnaissance des lettres arabes manuscrites et sur la segmentation en caractères. Car la reconnaissance du mot est fortement dépendante de la segmentation de mot en caractères et de la reconnaissance de ces derniers. Alors on a pris deux images une écrite manuellement et l'autre représente un manuscrit

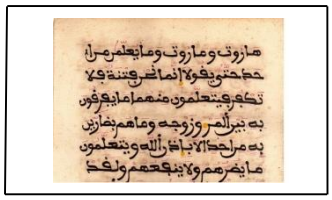


Image d'un manuscrit

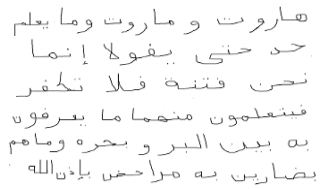


Image écrite manuellement

Les résultats obtenus sont décrits ci-dessous :

V.1. Tache de la segmentation

Pour calculer les taches de la segmentation (ligne, mot, caractère) on va prendre une image écrite manuellement et on va comparer le résultat de la segmentation de l'image avec le résultat donné par la segmentation d'un manuscrit arabe.

V.1.1 La segmentation du texte en lignes :

- **Pour une image écrite manuellement :**

La performance de la segmentation est calculée comme suit :

Taux de segmentation = nombre de ligne bien segmentés / sur le nombre total de ligne

Nous avons testé la segmentation sur une image qui contint 6 lignes et le résultat est de 6 lignes donc le taux de segmentation est de100%.

- **Pour un manuscrit arabe :**

La performance se calcule comme précédemment :

Le résultat donner après la segmentation du texte en lignes du manuscrit qui contient six lignes est : deux lignes segmentées ce qui donne un taux de segmentation de 33%.

V.1.2 La segmentation de la ligne en mots

La performance de la segmentation de la ligne en mots est calculée comme suit :

Taux de segmentation = nombre de caractère bien segmentés / sur le nombre total de caractère.

Nous avons testé la segmentation de la ligne sen mots sur une image qui contint 30 mots et le résultat donne un taux de 100%.

V.1.3 La segmentation du mot en caractères :

- **Pour une image écrite manuellement :**

La performance de la segmentation du mot en caractères est calculée comme suit :

Taux de segmentation = nombre de caractère bien segmentés / sur le nombre total de caractère.

Nous avons testé la segmentation du mot en caractères sur une image qui contint 62caractères et le résultat donne un taux de 90%.

- **Pour un manuscrit arabe :**

La performance se calcule comme précédemment :

Le résultat donné après la segmentation en caractères du manuscrit est : le résultat donner après la segmentation du mot en caractère du manuscrit est 50 %

D'après nos tests, nous avons remarqués les mots contenant des lettres chevauchées et ligaturées sont mal segmentés. La performance de notre segmentation est dépendante aussi de la hauteur des mots, tel que, dans certains cas où la hauteur de nom est grande le mot sera mal segmenté.

V.2. Tache de la reconnaissance des lettres arabes manuscrites :

La performance de la reconnaissance de lettres peut être mesurée en calculant le taux de reconnaissance et le taux de substitution :

- **Pour une image manuscrite :**

➢ **Pour 1 gramme (une lettre) :**

Taux de reconnaissance = Nombre de lettres reconnues / Nombre total de lettres.

Taux de substitution = Nombre de lettres males reconnues / Nombre total de lettres.

Après avoir fait la segmentation en caractère de l'image, on teste la reconnaissance de lettre sur notre base de données composée de 62 lettres arabes, nous avons obtenu respectivement les taux de reconnaissance et de substitution suivants : **77 %**pour la reconnaissance et **23%** pour la substitution.

D'après nos tests, nous avons remarqué que notre système fait des confusions entre quelques lettres par exemple :

- 'Ta'milieu et 'qa'milieu,car la lettre 'ta milieu 'est reconnu comme qa milieu ;
- Ensuite on a la lettre tha début qui est reconnu comme ta début ;
- On outre la lettre 'za' entré reconnu par ' ra' entré (le système a ignoré le point de za entré)
- Même chose pour la lettre' ddha début' qui est reconnu comme ' tta début ' ;
- La lettre 'la milieu' est reconnu comme 'alif F' ; et la lettre la fin est reconnu comme da fin
- La lettre 'ma milieu' est reconnu comme 'hha milieu' ;
- La lettre' na' entrée est reconnu comme 'dha fin '.

Ces confusions sont dues à l'étape de l'extraction de primitives qui est basé sur le calcule de pourcentage de pixels dans les zones de l'image. En effet, cette méthode ne fait pas la différenciation entre certaines lettres qui se ressemblent.

- **Pour 2 gramme (2 lettres)**

Taux de reconnaissance = Nombre de lettres reconnues / Nombre total de lettres.

Taux de substitution = Nombre de lettres males reconnues / Nombre total de lettres.

Voilà les résultats après avoir testé la reconnaissance de lettre(2grammes) sur notre base de données composée de 20mots de 2 lettres, nous avons obtenu respectivement le taux de reconnaissance qui est le **:73%** et le taux substitution qui est : **26%.**

D'après nos tests, nous avons remarqué que notre système fait des confusions entre quelques lettres par exemple : lorsque on a pris le mot « 3a D la F » ce dernier est reconnu comme « 3a D da F » parce que la lettre la fin a été reconnu comme da fin précédemment.

- Le mot « na d ma M » est reconnu comme « dha D ma M » vu que la lettre na D étais reconnu comme dha fin.
- Le mot « hha D dha M » est reconnu comme « hha D kha M ».

- **Pour une image écrite manuellement :**

- **Pour 1 gramme (une lettre) :**

Taux de reconnaissance = Nombre de lettres reconnues / Nombre total de lettres.

Taux de substitution = Nombre de lettres males reconnues / Nombre total de lettres.

Après avoir fait la segmentation en caractère de l'image, on teste la reconnaissance de lettre sur notre base de données composée de 62 lettres arabes, nous avons obtenu respectivement les taux de reconnaissance et de substitution suivants :80 **%**pour la reconnaissance et**20%** pour la substitution.

D'après nos tests, nous avons remarqué que notre système fait des confusions entre quelques lettres par exemple :

- On a la lettre tha début qui est reconnu comme ta début ;
- On outre la lettre 'za' entré reconnu par ' ra' entré (le système a ignoré le point de za entré)
- 'Kha fin 'et 'dja fin ' reconnu comme ha fin

On remarque que le processus ignore les points des lettres alors il fait des confusions avec d'autre lettre.

- **Pour 2 gramme**

Taux de reconnaissance = Nombre de lettres reconnues / Nombre total de lettres.

Taux de substitution = Nombre de lettres males reconnues / Nombre total de lettres.

Voilà les résultats après avoir testé la reconnaissance de lettre(2grammes) sur notre base de données composée de 20 mots de 2 lettres, nous avons obtenu respectivement le taux de reconnaissance qui est le **:80%** et le taux substitution qui est : **20%.**

V.3. Tache de reconnaissance des mots

La performance de reconnaissance des mots est dépendante de la segmentation de ce dernier en caractères et reconnaissance des lettres.

Nous avons effectué des tests de reconnaissance des mots sur la base des mots utilisée dans la segmentation en caractères. Nous avons obtenu une reconnaissance nulle dans les deux images soit le manuscrit soit l'image écrite.

VI. Conclusion

Nous avons présenté dans ce chapitre les différents aspects du développement de notre système. Nous avons décrit précisément notre système et nous avons montré le mécanisme de reconnaissance des mots arabes manuscrits**.**

Pour valider notre système nous avons effectué des tests sur la segmentation du mot en caractères, la reconnaissance des caractères ainsi des tests sur la reconnaissance des mots. Les résultats obtenus sont très encourageants, mais des détails demandent à être améliorés, comme par exemple l'extraction de primitives qui est l'une des raisons qui fait que le taux de reconnaissance des lettres est de (77%). Ainsi, certaines erreurs de segmentation

Lorsque on a comparé les résultats des tests effectuer par le système de reconnaissance du mot en utilisant les réseaux de neurone on a conclu que la méthode de Markov caché est plus efficace car le taux de la reconnaissance de la lettre avec la méthode de Markov caché était de 77% alors que pour les réseaux de neurone est de 70%. Alors que la reconnaissance du mot ne donne aucun résultat pour les deux modèles et c'est dû au problème de l'extraction de primitives et les erreurs de segmentation

CONCLUSION GÉNÉRALE

CONCLUSION GENERALE

La reconnaissance de l'écriture arabe manuscrite hors-ligne est un sujet qui a beaucoup de difficulté ce qui a mené plusieurs chercheurs à conduire plusieurs travaux pour remédier au problème de la reconnaissance.

Le problème posé dans la reconnaissance pour les approches existantes est l'opération de la segmentation. Pour remédier à ce dernier nous avons proposé un algorithme qui permet la segmentation du mot en caractères en éliminant les éléments d'épaisseur uniforme qui relient ces caractères.

Le système que nous avons proposé est composé de deux sous-systèmes : un sous-système d'apprentissage et un sous-système de reconnaissance. Le système d'apprentissage a la capacité de traitement des images, et se charge aussi d'extraction des caractéristiques sous forme d'un vecteur de description, qui sera destiné à être traité par le modèle de Markov caché. Trois matrices (A, B, P) représentant chaque lettre sont générées (cf. algorithme de Baum Welch). Les résultats obtenus seront sauvegardés dans une base de données d'apprentissage. Chaque lettre est représentée sur la base de données d'apprentissage sous quatre formes différentes (début, milieu, fin, isolée).

Le sous-système de reconnaissance a pour objectif la reconnaissance du mot. Il se charge d'abord du prétraitement de l'image du mot d'entrée. L'image sera segmentée en lignes puis en mot et à la fin en caractères. Chaque caractère passe par le module d'extraction des caractéristiques pour trouver son vecteur de description. Ce dernier sera utilisé par l'algorithme de Viterbi dont le but de trouver la lettre correspondante dans la base de données d'apprentissage. En fonction des lettres reconnues, le système sélectionne le mot le plus probable dans une base de mots arabes.

Nous avons implémenté notre système en langage Java. Le prototype réalisé respecte l'architecture que nous avons proposée pour la reconnaissance. Pour évaluer la performance de notre système en termes de précision de reconnaissance, nous avons mené une étude expérimentale qui porte sur la segmentation, la reconnaissance de caractères et la reconnaissance de mots. Les résultats obtenus sont prometteurs. En effet, nous avons reconnus un nombre important des échantillons des mots testés.

Le travail réalisé nous ouvre plusieurs perspectives. Nous essayerons d'entamer :

- Le développement de nouvelles approches dans l'étape de prétraitement (Squelettisation, seuillage...),
- Développer de nouvelles méthodes de segmentation Avec l'aide de modèles linguistiques
- Faire une hybridation entre les méthodes de segmentation
- Le développement de nouvelles approches pour L'extraction des primitives
- Proposer un nouveau processus d'extraction des primitives.

ANNEXE

(1) _Histogramme d'une image

L'histogramme d'une image est représenté par un graphique, l'axe horizontal (abscisse) représente l'ensemble des valeurs de luminosité potentielles de l'image : c'est la quantité de lumière (ou l'intensité lumineuse) sur 256 niveaux, allant de 0 (le noir) à 255 (le blanc). L'axe vertical (ordonnée) représente la proportion de pixels dans l'image pour chacun des niveaux de luminosité

Les tons foncés sont représentés à gauche de l'histogramme. Plus on va vers la droite, plus on augmente la luminosité. De même, lorsque l'histogramme monte, le nombre de pixels augmente. Lorsqu'il descend, le nombre de pixels diminue. Si un grand nombre de pixels se situent sur la partie gauche de l'histogramme, cela signifie qu'une majorité de pixels se trouvent dans les basses lumières. Il peut être utilisé pour améliorer la qualité d'une image (Rehaussement d'image) en introduisant quelques modifications, pour pouvoir extraire les informations utiles de celle-ci.

(2) _Zoning : La méthode zoning consiste à partitionner une image en M sous-images de mêmes tailles appelées zones (8x8). Nous calculons pour chaque zone le pourcentage de pixels noir en obtenant une matrice de pourcentage qui représente le vecteur de primitives, celui-ci sera utilisées pour l'apprentissage et la reconnaissance.

(3) _Code Freeman [20]

Ce code sert à coder un contour dans une image, et toute structure connexe est bien codée par Freeman. Chaque pixel est représenté dans la chaîne par l'entier qui désigne sa position par rapport au pixel précédent.

Le code de Freeman se base seulement sur 3 éléments pour caractériser une forme :

• Les coordonnées (x, y) d'un point appartenant au contour de la forme que l'on veut caractériser

• Un sens de rotation

• Une chaine de caractères qui code le conteur

La chaîne commence conventionnellement par les coordonnées du point de départ. Dans l'exemple de la figure suivante, le code de la chaîne de Freeman serait :

(3,3)70701013345321013432170113344545665666676

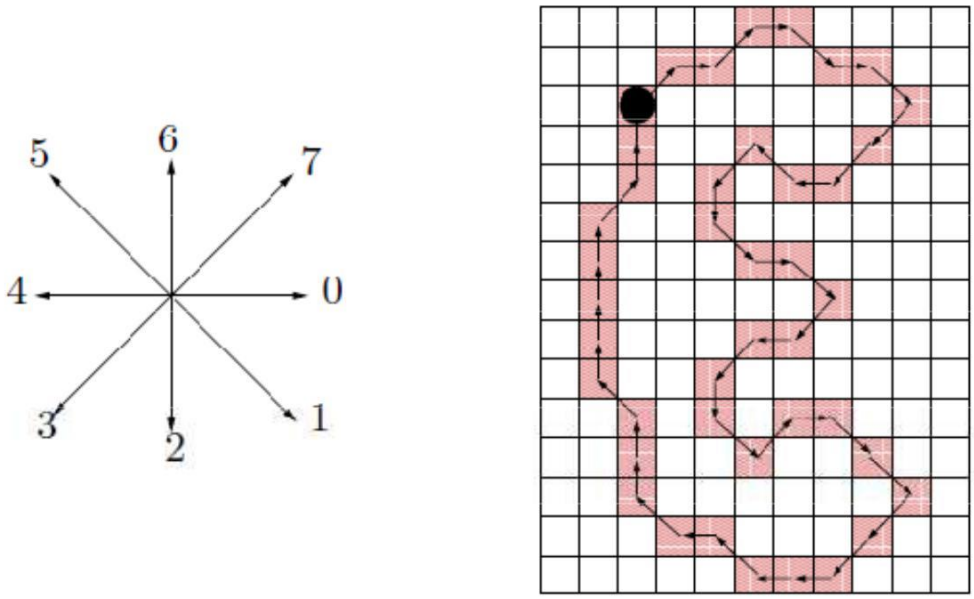


Figure2 : Codage de Freeman

(4) Principe de fonctionnement de l'algorithme de segmentation en caractères

Nous avons en entrée une image contenant le nom ou une partie de nom, cette image sera parcourue de droite à gauche en détectant les colonnes blanches en utilisant un seuil. Telle que si le nombre de pixel blanc dans cette colonne est supérieur ou égale à ce seuil, elle sera considérée une colonne blanche. Ce seuil est obtenu empiriquement comme suit : hauteur de l'image * 0.95f. Dans chaque itération, on calcul aussi le nombre de colonnes blanches successives. Si ce nombre est supérieur ou égale à un seuil, cette suite de colonne est considérée comme un bloque blanc, et ses indices de début et de fin serons sauvegarder dans une liste « al ». Ce seuil est obtenu empiriquement comme suit : maximum (1, hauteur * 0.05f). A l'aide de cette liste, nous déterminons les caractères à extraire de l'image d'entrée en parcourant la liste par pas de deux (2). Dans ce qui suit, nous présentons les détails de cet algorithme.

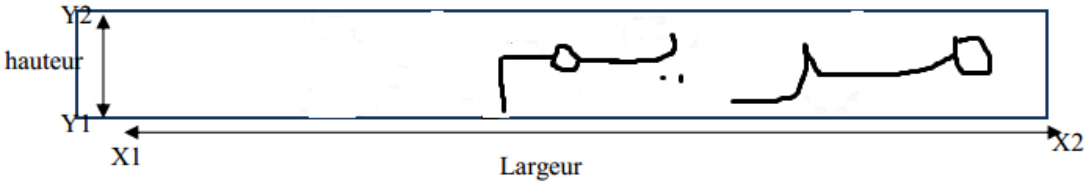


Image d'entrée résultante de la segmentation du texte en lignes

Entrée : l'image contenant le nom ou une partie de nom,
Sortie : images des trames résultantes de la segmentation de nom,

/ / hauteur de l'image d'entrée
Entier : hauteur = y2 - y1,

//seuil déterminant si un bloque peut-être négligé (blanc)
Entier : SeuilBlocBlanc = maximum (1, hauteur * 0.05f),

//seuil déterminant si une colonne d'un pixel d'épaisseur peut être négligée (blanche)
Entier : SeuilColonneBlanc = hauteur * 0.95f,

//Liste résultante contenant les indices où se débute et où se termine chaque trame à extraire dans l'image d'entrée
Liste d'entier : al,
Booleen : Separateur = **Vrais,**
Entier : CharX1 = 0, prevCharX1 = -1,
Booleen : EspaceBlancLibre = **Faux**,
Entier : NumBlancConsecutive = 0,
Etape 1 : Le but de cette étape est de trouver les indices des bloques considérés blancs, puis les sauvegarder dans la liste al.
Pour x = 1 à x < (largeur - 1), pas de 1 Faire

ANNEXE

// Teste effectuer dans le cas il a trouvé un bloque de largeur supérieure à la hauteur et il n'a pas marqué des colonnes blanches.
Si EspaceBlancLibre = Faux et NumBlancConsecutive == 0 et x - charX1>= hauteur alors x = charX1,
EspaceBlancLibre = **Vrais,**
FinSi

//Nombre de pixels blancs dans une colonne
Entier : numPixelsBlancColumn = 0,
Booléen : EspaceBlanc = **Vrais,**
Pour y = y1 à y < y2, pas de 1 Faire

//tester si le pixel de coordonnée x, y est blanc ou non
Si pixels (x, y) >= 128

//Incrémenter le nombre de pixel blanc d'une colonne x
Incrimenter numPixelsBlancColumn,
FinSi
Sinon

// le cas où le pixel de coordonnée x, y est noir (<128)
Si EspaceBlancLibre = Faux Alors
EspaceBlanc = **Faux**
Sortir de la boucle pour,
FinSi
FinSinon,
Fin pour,

//tester si la colonne x est négligeable (> SeuilColonneBlanc) ou non (<SeuilColonneBlanc)
Si EspaceBlancLibre = Vrais et numPixelsBlancColumn < SeuilColonneBlanc **Alors**
EspaceBlanc = **Faux**
FinSi
Si EspaceBlanc = Vrais Alors

//incrémenter le nombre de colonne blanche successive
Incrimenter NumBlancConsecutive,

//test si le bloque (succession de colonnes) est négligeable (>= SeuilBlocBlanc) ou non (<SeuilBlocBlanc)
Si NumBlancConsecutive >= SeuilBlocBlanc)
Si Separateur= Faux Alors
Separateur=Vrais,

//sauvegarder dans la liste al les indices des bloques blancs à supprimer
Ajouter dans al(CharX1), //début de bloque
Ajouter dans al (x- (NumBlancConsecutive - 1)), // fin de bloque
FinSi
FinSi

FinSi

// Le cas où la colonne x n'est pas blanche
Sinon NumBlancConsecutive = 0,
Si Separateur = Vrais Alors
Separateur = **Faux,**
prevCharX1 = charX1, charX1 = x, EspaceBlancLibre = **Faux,**
FinSi
Fin
Sinon
Fin Pour

//à la fin de nom, si le dernier bloque n'est pas un blanc alors il faut sauvegarder son indice pour le localiser Si NumBlancConsecutive = 0 Alors
Ajouter dans al(CharX1),
Ajouter dans al (largeur),
FinSi

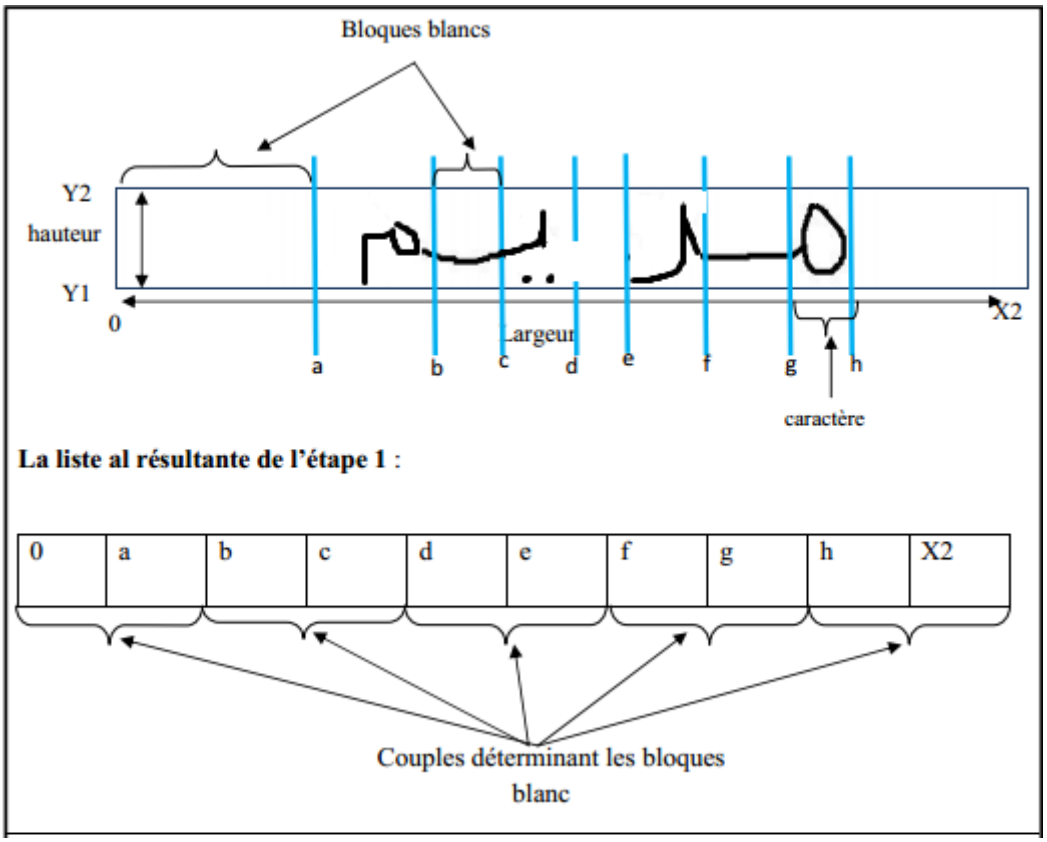


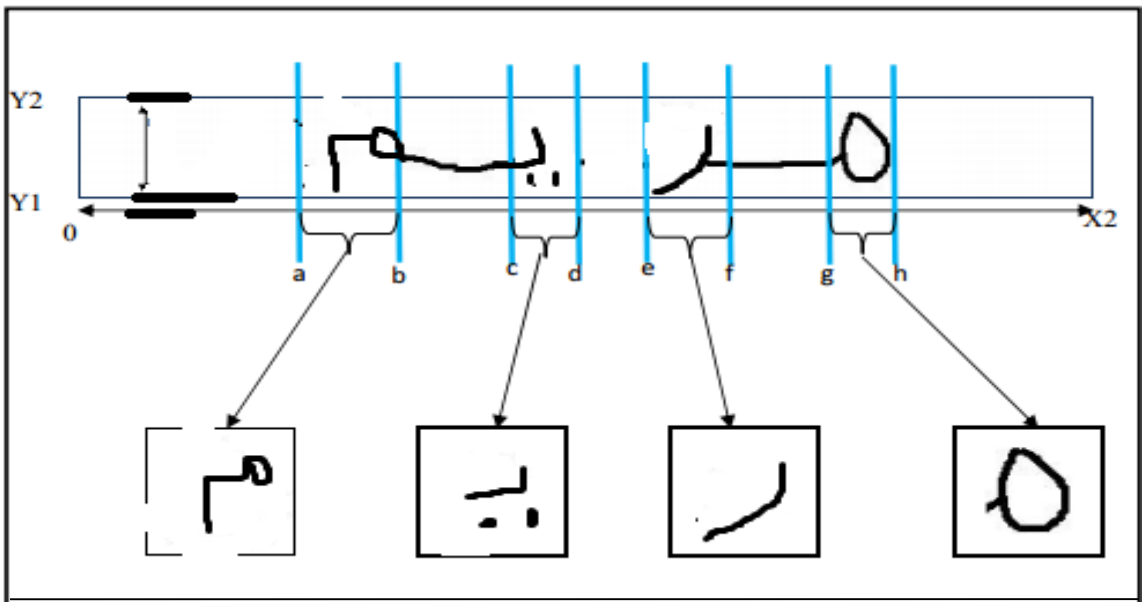
Illustration de l'étape 1. Couples déterminant les bloques blanc

Étape 2 : l'objectif est de fusionner les bloques blanc successifs les plus proche en utilisant le seuil Min largeur Char.
Entier : MinlargeurChar = hauteur * 0.15f,
Si MinlargeurChar < 1 Alors MinlargeurChar=1,
FinSi

//Test permettant de déterminer si deux bloques blanc successifs dans la liste peuvent être fusionnés
Pour i = 0 à (i + 4) < taille de la liste al, pas de 2 Faire
Entier LargeurChar = récupérer de al l'élément (i + 2) - récupérer de al l'élément (i),
Si largeurChar < MinlargeurChar ou largeurChar < 2))
Supprimer de la liste al l'élément (i + 2),
Supprimer de la liste al l'élément (i + 1) ; Décrémenter i de 2,
FinSi
FinPour

Etape 3 : Après l'obtention de la liste al optimisée qui contient les indices des blocs blancs, elle sera utilisée pour l'extraction des caractères de l'image d'entrée. Cette liste est utilisée comme suit : La liste seras parcourus de début jusqu'à la fin par pas de deux, dans chaque itération le contenu de la liste dont les indices sont (i+1) et (i+2) délimite où commence et où termine le caractère à extraire dans l'image d'entrée. Si par exemple i=2 alors le contenu des indices 3 et 4 délimite une trame à extraire.

Pour i = 0 a i < (taille de la liste al – 2), pas de 2 Faire
Si (al (0) =0) Alors
Entier cx1 = récupérer de la liste al l'élément (i +1), Entier cx2 = récupérer de la liste al l'élément (i+2), Extraire de l'image d'entrée trame de largeur (cx1, cx2),
FinSi
Sinon
Si al (0) <> 0 Alors Entier cx1 = récupérer de la liste al l'élément (i), Extraire de l'image d'entrée trame de largeur (0, cx1),
FinSi
Si i <> 0 Alors Entier cx1 = récupérer de la liste al l'élément (i +1), Entier cx2 = récupérer de la liste al l'élément (i+2), Extraire de l'image d'entrée trame de largeur (cx1, cx2),
FinSi
FinSinon
Fin Pour



Illustre le résultat de l'algorithme de la segmentation.

L'application de l'algorithme de segmentation en caractères sur l'écriture arabe manuscrite peut ne pas être fiable, vu les caractéristiques de l'écriture arabe manuscrites par exemples : les problèmes de discontinuité de l'écriture, les ligatures, de chevauchement et d'accolement de pseudo mots, de grandes variabilité inter et intra scripteur, de variation de dimension des pseudos mots ainsi que les signes diacritiques... « ***Il n'est pas évident de juger de l'efficacité d'un algorithme de segmentation en lettres, le résultat peut être décevant*** » [14].

(5) -Algorithme de Viterbi

Après les phases de normalisation et l'extraction de vecteur d'observation de la lettre, ce dernier sera donné en entré de l'algorithme de Viterbi pour être comparé aux modèles λ_k de la base de données réalisée dans la phase d'apprentissage, k=1, n. Le modèle retenu sera celui qui fournira la meilleure probabilité correspondante à l'évolution de sa suite de primitives, c'est-à-dire :

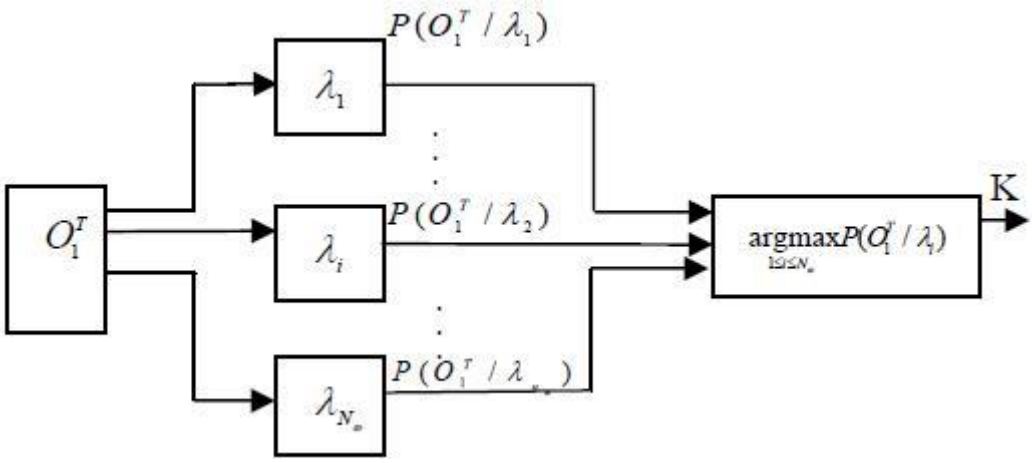
$V = \arg\max [P(O/\lambda_k)], 1 <= V <= n$ [29]

Où :

O : présente l'observation qui correspond à la lettre.

λ_k :présente les paramètres de MMC Π ($_k$, A_k, B_k).

Le principe de Viterbi est présenté dans la figure qui suit :



L'algorithme de Viterbi que nous avons utilisé pour la reconnaissance des noms arabe se trouve allégé, en effet il n'y a pas une suite d'état s emprunté pour arriver à la probabilité maximale. La découverte du modèle donnant la plus ortef probabilité P (O/ λ) est suffisante pour identifier la lettre.

De plus, sachant que les matrices de MMC A et B sont connues à l'avance, on peut optimiser l'algorithme de reconnaissance en les transformant (A, B) en valeur logarithmique.

ANNEXE

L'algorithme de reconnaissance est le suivant :

Initialisation :

$$\delta_1(i) = \log(\pi_i) + \log(b_i(O_1)) \quad 1 \leq i \leq N$$

Induction :

$$\delta_t(j) = \max_{1<i<N} [\delta_{t-1}(i) + \log(a_{ij})] + \log(b_j(O_t)) \quad 1 \leq j \leq N, \quad 2 \leq t \leq T$$

Terminaison :

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)]$$

L'algorithme de Viterbi peut retourner deux résultats : soit la lettre est reconnue dans ce cas elle sera sauvegardée, soit la lette est rejetée. Dans les deux cas, le système peut prendre l'une des décisions que nous définissons dans ce qui suit :

1. Si ce n'est pas la fin du nom : dans ce cas, le système recommence à partir de processus d'acquisition d'une nouvelle lettre tout en éliminant les lettres parcourues puisque nous avons abouti à une reconnaissance.
2. Si c'est la fin du nom : dans ce cas, le système regroupe tous les lettres reconnues pour aboutir à la reconnaissance du nom.

BIBLIOTHEQUE

[2] : ZERMI Narima. « Reconnaissance de mots manuscrits arabe par les modèles de Markov Cachés et les réseaux de neurones ». Thèse de doctorat. Université ANNABA.2007

[3] : Riadh Bouslimi, « système de reconnaissance hors-ligne des mots manuscrits arabe pour multi-scripteurs », mémoire de mastère, université de Jendouba, 2006

[4] : Farés Menasri, « contributions à la reconnaissance de l'´écriture arabe manuscrite », thèse de doctorat, université Paris Descartes, 2008.

[5] : Brahim Farou, Samir Hallaci et Hamid Seridi Système Neuro-Markovien pour la reconnaissance de l'écriture manuscrite arabe à vocabulaire limité. Département d'informatique, Université 08 mai 45 Guelma, Université de Reims, France

[6] : Mélanie Lemaitre, « approche markovienne bidimensionnelle d'analyse et de reconnaissance de documents manuscrits », thèse de doctorat, université paris 5 René Descartes département de mathématiques et informatique, 2007.

[7] : Al Falou Wassim, « Reconnaissance de caractères manuscrits par réseau de neurones », thèse de doctorat, Université de Rennes – IRISA, 1998.

[8] : Amrouch Mustapha, « Reconnaissance de caractères imprimés et manuscrits, textes et documents basée sur les modèles de Markov cachés ». thèse doctorat, Université Ibn Zohr.

[9] : Giraud Charles et Lacoste Bastien, « reconnaissance de caractères », Laboratoire de Recherche en Informatique Paris, 2006.

[10] : Nawwaf n. kharma et Rabab k. ward, « systèmes de reconnaissance de caractères pour les non-experts », université de Colombie-Britannique, 1999

[11] ; Redouane TLEMSANI. « La reconnaissance en ligne du manuscrit arabe », thèse de doctorat, université Mohamed Boudiaf, Oran,2012

[12] : N. Mezghani, A. Mitiche, et M. Cheriet, "A new représentation of Shape and its use for high performance in online Arabic character recognition by an associative memory," International Journal on Document Analysis and Recognition, vol. 7, n°. 4, p. 201–210, 2005.

[13] : F. Menasri, "Contributions à la reconnaissance de l'écriture arabe manuscrite," in thèse de l'Université Paris Descartes, 2008.

[14] : Adnan Amin, « Off line Arabic character recognition - a Survey », In ICDAR '97 : Proceedings of the 4th International Conference on Document Analysis and Recognition, pages 596–599, IEEE Computer Society, 1997.

[15] : Najoua Ben Amara, Abdel Belaïd et Noureddine Ellouze, « utilisation des modèles markoviens en reconnaissance de l'écriture arabe : Etat de l'art », 1'école nationale d'ingénieurs de Monastir - 5019 Monastir – Tunisie, 2003.

[16] : Liana M. Lorigo et Venu Govindaraju, « Offline arabic handwriting recognition –survey IEEE Transactions on Pattern Analysis and Machine Intelligence », 2006.

BIBLIOTHEQUE

[17] : M. Cheriet, « Strategies for Visual arabic handwriting recognition : issues and case study », International Symposium on Signal Processing and its Applications, Sharjah, United Arab Emirates, 2007.

[18]. Denis Arrivault, « Apport des Graphes dans la Reconnaissance Non-Contrainte de Caractères Manuscrits Anciens », thèse doctorat, Université de Poitiers, 2002.

[19] : A. Benouareth, A. Ennaji et M. Sellami**, «** Utilisation des HMMs de Durée d'Etat explicite pour la Reconnaissance des Mots Arabes Manuscrits », Laboratoire de Recherche en Informatique, Laboratoire PSI -FRE 2645 Université de Rouen, INSA de ROUEN, 2000.

[20] : Khalid Hallouli, « reconnaissance de caractère par méthodes Markovienne et réseau bayésiens », Telecom Paris Ecole nationale supérieur des télécommunications, 2004.

[21] : farida belkacem « reconnaissance automatique des noms arabe » diplôme master, université Mouloud Mammeri de TIZI OUOZ ,2006.

[22] : Xavier Dupré, « Contributions à la reconnaissance de l'écriture cursive à l'aide de modèles de Markov cachés », thèse de doctorat, Université René Descartes - Paris V U. F. R. d'Informatique et de Mathématiques Appliquées, 2004.