République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université SAAD DAHLAB Blida
Faculté des Sciences
Département d'Informatique

Mémoire de Fin d'Etudes

Présenté en vue de l'obtention du

Diplôme de Master en Informatique

Spécialité : Traitement Automatique de la Langue (TAL)

Thème

Elaboration d'un OCR basé sur les modèle de Markov cachés : application au texte Arabe imprimé non voyellé

Réalisé par

- **Benkessirat Walid**
- **Khenniche Oussama**

Dirigé par

- **Mme Droua Ghania**
- **Mr Cherif Zahar Amine**

Présidente du jury

- **Mme Boutoumi Bachira**

Examinatrice

- **Mme Lahiani Nesrine**

2019/2020

Remerciements

Au terme de ce travail, nous voudrions exprimer nos profondes gratitudes envers dieu, le tout puissant qui, grâce à son aide, nous avons pu finir ce travail.

Nous voudrions présenter nos remerciements à madame Droua de nous avoir proposé un sujet qui a enrichi nos connaissances et de l'avoir bien dirigé. Nous la remercions de nous avoir encadré et orienté.

Nous somme très reconnaissants envers monsieur Cherif Zahar de nous avoir encadré, aidé et conseillé.

Merci à madame Boutoumi d'avoir accepté de présider le jury et à madame Lahiani d'avoir accepté d'examiné notre travail.

Nous adressons nos remerciements à toute la famille universitaire, en particulier, une profonde gratitude aux membres du département d'Informatique.

Merci à tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

Tables des matières

Introduction générale…….. 1

Chapitre 1 : Reconnaissance Optique de Caractères.

1.1 Introduction……..... 3

1.2 Définition……..…… 3

1.3 Différents aspects de L'OCR…….....…….……. 4

1.3.1 Reconnaissance en-ligne et hors-ligne…… 4

a La reconnaissance en-ligne…….. 4

b La reconnaissance hors-ligne……... 4

1.3.2 Les approches globales ou analytiques …….. 4

a L'approche globale (holistic approach) ……... 4

b L'approche analytique…….. 5

1.4 Procédure Générale de l'OCR……... 5

1.4.1 Acquisition De L'image……... 6

1.4.2 Prétraitement…… 6

a La binarisation…….. 6

b Le lissage…….. 7

c La squelettisation…… 7

d La suppression du bruit……. 7

e Redressement…… 8

f La normalisation……... 8

1.4.3 La Segmentation……..... 8

a La Segmentation en lignes…… 8

b La Segmentation en pseudo-mot…… 9

c La Segmentation en caractères…… 9

- La technique de segmentation basée sur la projection…….. 10
- La technique de segmentation basée sur la squelettisation... 11
- La technique de segmentation basée sur le contour…….. 13
- La technique de segmentation basée sur Template Matching……... 17

1.4.4 L'extraction de caractéristiques……... 18
1.4.5 La Classification……..... 19
a L'étape de l'apprentissage…… 19
b L'étape de la classification……... 20
- k plus proche voisins knn…… 20
- Machine à vecteur de supports SVM…… 21
- Arbre de décision DT…… 21
- Modèle de Markov caché (Hidenn Markovien Model HMM)..... 22

1.5 Conclusion…… 22

Chapitre 2 : OCR et l'Arabe

2.1 Introduction……... 23
2.2 Caractéristiques de la langue Arabe…….. 23
2.3 Spécifiés de la lange Arabe……... 25
2.4 Obstacles liées à l'OCR Arabe…….. 30
2.4.1 Prétraitement……. 31
2.4.2 Segmentation…… 31
2.4.3 Extraction de caractéristiques……... 32
2.5 Conclusion……..... 33

Chapitre 3 : Implémentation du modèle AOCR proposé

3.1 Introduction……..... 34
3.2 Environnement de développement…… 34
3.2.1 Machine…… 34
3.2.2 Langage et bibliothèque…… 34
3.3 Mise en œuvre de l'OCR……..…….. 36
3.3.1 Acquisition des données……..……..... 36
3.3.2 Prétraitement……. 36
a La binarisation…….. 37
b Le lissage…….. 38
c La suppression du bruit……. 38
d La normalisation……... 39
3.3.3 La segmentation……..... 39
a La segmentation en ligne……. 39

b La segmentation en pseudo mot…… 41
c La segmentation en caractères……. 42
3.3.4 L'extraction de caractèristiques…… 44
3.3.5 La classification……..…….. 47
a La classification des caractères……. 47
b La classification des mots……. 50
3.4 Coclusion…….. 52

Chapitre 4 : Résultats d'implémentation.

4.1 Introduction……... 53
4.2 Corpus de données…… 53
4.2.1 Introduction……... 53
4.2.2 Problèmes liés à la segmentation…….. 53
4.2.3 Construction de notre corpus de données……. 54
4.2.4 Conclusion…… 54
4.3 Résultats de la segmentation……. 55
4.3.1 Segmentation en ligne……... 55
4.3.2 Segmentation en pseudo-mots…….. 56
4.3.3 Segmentation en caractères……... 57
4.4 Résultats de la reconnaissance…….. 57
4.4.1 Résultat de $Caractères\_Classification$…….……... 57
4.4.2 Résultat de $mots\_classification$…… 59
4.5 Conclusion…… 60
Conclusion Générale…… 61

Liste des figures

Figure 1.1	Etapes générale de l'OCR …….……	5
Figure 1.2	Segmentation en ligne……	9
Figure 1.3	Segmentation en pseudo mots.....	9
Figure 1.4	Classification des techniques utilisées dans le domaine de la segmentation du texte Arabe …….	10
Figure 1.5	La squelettisation……	11
Figure 1.6	Flow-chart de détermination du point de départ……	12
Figure 1.7	Construction du caractère……...	12
Figure 1.8	Technique de segmentation basée sur le traçage des contours…...	13
Figure 1.9	Exemple de localisation des points de segmentation (flèche)……	14
Figure 1.10	Caractères touchants sous la ligne de base…….	15
Figure 1.11	Segmentation du caractère ى……..	15
Figure 1.12	Détection du contour supérieur en utilisant les codes de Freeman	16
Figure 1.13	Les angles de jonction entre les caractères…….	18
Figure 1.14	Système de segmentation du texte à partir d'une capture vidéo [33]…….	18
Figure 2.1	Exemple d'écriture Arabe montrant la ligne de base…….	25
Figure 2.2	Exemple de formes de boucles dans des styles différents……..	26
Figure 2.3	Variation du mot «**الشعب** » écrit avec un nombre différent de de traits d'allongement dans différentes positions……...……...	28
Figure 2.4	Exemples de ligatures horizontales et verticales……	29
Figure 2.5	Exemple de formes de PAWs sans et avec caractères ligaturés verticalement (respectivement à droite et à gauche de « ≡ »)……	29
Figure 2.6	Le nom de la ville «البليدة», en forme différentes formes……..	30
Figure 3.1	Architecture globale du système……	35
Figure 3.2	Passage d'une image en couleur à une image noir et blanc……..	38
Figure 3.3	Passage d'une image bruitée à une image sans bruit……...	38
Figure 3.4	Application de l'ouverture morphologique.……..….	39
Figure 3.5	Sur-segmentation des lignes……..	40
Figure 3.6	Texte en image avant et après l'application de la dilatation…….	41
Figure 3.7	Résultat d'une segmentation en pseudo mots……	42

Figure 3.8	Template proposée dans [4]……..	42
Figure 3.9	Template proposée…….	43
Figure 3.10	Exemple de cas particuliers……	43
Figure 3.11	Exemple de cas particuliers……	44
Figure 3.12	Caractère en image avant et après squelettisation……	45
Figure 3.13	Exemple d'extraction de caractéristiques pour le caractère «ڎ »…….…….	46
Figure 3.14	Topologie d'un HMM propre à un caractère…….	48
Figure 4.1	Texte extrait de notre corpus.…….....	55
Figure 4.2	Résultat de segmentation en ligne.…….....	56
Figure 4.3	Résultats de la segmentation en pseudo-mots.……..	56
Figure 4.4	Résultats de la segmentation en caractères……	57
Figure 4.5	Précision de la classification en fonction du nombre de segment et facteur de classification.…….	58
Figure 4.6	Précision de la classification en fonction du nombre d'instance d'apprentissage.……..	59

Liste des tableaux

Tableau 2.1	L'alphabet arabe dans ses différentes formes.……	23
Tableau 2.2	Hamza et Madda et les positions qu'elles occupent en association avec Alif, Waw et Ya.……..……	24
Tableau 2.3	Hamza et Madda et les positions qu'elles occupent en association avec le caractère « لا»……..	25
Tableau 2.4	Les caractères additionnels ……	25
Tableau 2.5	Les quatre formes des caractères « ain » et « he » en fonction de leur position dans la chaîne de caractères...	26
Tableau 2.6	Exemple de mots composés de la droite vers la gauche de 1,2,3,4 et 5 PAWs respectivement…….……	27
Tableau 2.7	Le PAW « قر » dans différents mots et différentes positions……..……	27
Tableau .28	: Exemples de caractères avec et sans matta……..	28
Tableau 2.9	Caractères susceptibles d'être ligaturés verticalement selon [6]. …….……	29

Résumé :

Des efforts considérables ont été déployé pour le développement des systèmes optique de reconnaissance de caractères, par la communauté des chercheurs. Le but de ce projet est l'implémentation d'un AOCR (Arabic Optical Character Recognition). La segmentation et la classification sont les opérations cœur des OCR en général. La nature cursive des caractères Arabe biaise les résultats finaux de la reconnaissance. Les caractères non segmentés ou sur-segmentés conduisent à de mauvais résultats. C'est pour cela que la segmentation et la classification dans les AOCR sont un sérieux problème de recherche. La segmentation d'un texte Arabe comprends 3 niveaux, à savoir la segmentation en ligne, en pseudo-mots et en caractères. Au cours de notre projet, nous avons choisi les techniques du contour, la projection verticale et *template matching* pour faire la segmentation des 3 niveaux respectivement. D'autre part, la classification comprend deux niveaux, à savoir la classification des caractères et la classification des mots. Au cours de notre projets nous avons choisi des modèles de classification basé sur les Modèles de Markov Cachés (HMM Hidden Markovian Model). Au cours de notre projet, nous avons aussi discuter quelques problèmes liés à la langue Arabe et étudier d'autre module concernant les OCR, à savoir l'acquisition des données, le prétraitement et l'extraction des caractéristiques. Les résultats d'implémentation sont prometteurs.

Mots clés : reconnaissance, segmentation, classification, caractère, Arabe…

Abstract:

Tremendous efforts have been put into the development of OCR systems by the community researchers. The aim of this project is the implementation of an Arabic Optical Character Recognition (AOCR). Segmentation and classification are the main operations of OCRs in general. The cursive nature of Arabic characters biases the final results of recognition. Unsegmented or over-segmented characters lead to wrong results. This is why segmentation and classification in AOCRs is a serious research problem. The segmentation of an Arabic text includes 3 levels, namely line segmentation, pseudo-words and characters segmentation. In our project, we chose the techniques of the contour, the vertical projection and template matching to make the segmentation of the 3 levels respectively. On the other hand, the classification has two levels, namely the characters classification and the words classification. In our projects we have chosen classification models based on the Hidden Markovian Model (HMM). We also discussed some problems related to the Arabic language and study another module concerning OCR, namely data acquisition, preprocessing and features extraction. The implementation results are promising

Keywords: recognition, segmentation, classification, character, Arabic ...

ملخص:

بذلت جهود كبيرة لتنمية أنظمة التعرف على الحروف البصرية من قبل مجتمع البحث. الهدف من هذا المشروع هو تنفيذ التعرف االبصري على الحروف العربية AOCR.التجزئة والتصنيف هي العمليات الأساسية لل OCRs بشكل عام. الطبيعة الربطية للأحرف العربية توارب النتائج النهائية للتعرف. تؤدي الأحرف غير المجزئة أو المجزئة بإفراط إلى نتائج سيئة. تتكون تجزئة النص العربي من ثلاثة مستويات ، وهي التجزئة الخطية، التجزئة لشبه كلمات و التجزئة الحرفية. خلال مشروعنا ، اخترنا تقنيات المحيط ، والإسقاط العمودي والمطابقة لتنفيذ تجزئة المستويات 3 على التوالي. من ناحية أخرى ، يتكون التصنيف من مستويين ، هما تصنيف الحروف وتصنيف الكلمات. خلال مشروعنا ، اخترنا نماذج التصنيف بناءً على Hidden Markovian Model HMM. خلال مشروعنا ، ناقشنا أيضًا بعض المشكلات المتعلقة باللغة العربية ودراسة وحداة أخرى تتعلق بالتعرّف البصري على الحروف OCR ، وهي الحصول على البيانات والمعالجة المسبقة واستخلاص الخصائص. تحصلنا على نتائج تنفيذ واعدة.

الكلمات المفتاحية: التعرف ، التجزئة ، التصنيف ، حروف ، اللغة العربية ...

Introduction générale

Introduction générale

Les techniques liées aux traitements de l'information connaissent actuellement un développement très actif en liaison avec l'information et présentent un potentiel de plus en plus important dans le domaine de l'interaction *Homme-Machine.* Ecrire pour communiquer a été de tous les temps une préoccupation première de l'homme. L'écrit a été, et restera, l'un des grands fondements des civilisations et le mode par excellence de conservation et du transmission du savoir. De ce fait, l'homme a toujours développé des techniques visant sa pérennité à travers les générations. En effet, avec l'apparition des nouvelles technologies d'information : l'électronique et l'informatique, et l'augmentation de la puissance des machines, l'automatisation des traitements (la lecture, la recherche et l'archivage. . .) apparaît incontournable. Ce besoin d'automatisation a donné naissance au domaine de la reconnaissance optique de caractère ou OCR (en anglais : Optical Recognition Character).

L'OCR est le transfert de l'image du texte dans un texte éditable pour éviter de le retaper. Cette reconnaissance pourrait être utilisée dans diverses applications, par exemple: l'automatisation de la saisie des formulaires administratifs, le tri automatique du courrier postal et de l'échange, le traitement des chèques ou la modification de documents anciens. Sur le plan méthodologique, l'OCR propose des approches différentes suivant le mode d'écriture : manuscrit ou imprimé. Deux domaines distincts sont considérés : il s'agit de la reconnaissance statique dite encore «hors-ligne » et la reconnaissance dynamique « en-ligne » [1]. Lors de la reconnaissance en ligne, les caractères sont reconnus pendant le processus d'écriture à l'aide de la trace numérisée du stylo. Cependant, la reconnaissance hors ligne concerne les images numérisées de documents déjà écrits.

De nos jours, les problèmes de l'écriture latine manuscrite sont partiellement résolus, et la lecture automatique de l'imprimé a fait une grande avancée dans beaucoup de domaines. La recherche dans ce domaine s'oriente vers l'analyse de documents beaucoup moins contraints que ceux traités jusqu'à présent. Contrairement au latin, la reconnaissance de l'écriture Arabe manuscrite ou imprimée reste encore aujourd'hui au niveau de la recherche et de l'expérimentation, le problème n'est pas encore résolu même si dans certaines applications à vocabulaires limité et en mono-fonte, des résultats appréciables sont communiqués. Le retard

Chapitre 1

Reconnaissance optique de caractères

de l'écriture Arabe par rapport à l'écriture latine peut être attribué à la complexité morphologique de l'alphabet Arabe. Les recherches sur la reconnaissance de l'écriture Arabe datent des années 80. Depuis, les recherches se sont multipliées dans ce domaine. Durant ces dernières décennies, plusieurs approches et méthodes ont été proposées par les chercheurs, dans le but, d'améliorer les taux de reconnaissance. C'est dans ce cadre que s'inscrit notre travail.

Pour réaliser ce projet de fin d'étude, dont le thème est : *« Elaboration d'un OCR basé sur les modèle de Markov cachés : application au texte Arabe imprimé non voyellé »*, proposé par l'équipe de « Dialogue Homme-Machine » de la division «Communication parlée et pathologie du langage » au Centre de Recherche Scientifique et Technique pour le Développement de la Langue Arabe(CRSTDLA), notre mémoire est organisé comme suit :

- Chapitre 1 est consacré aux généralités sur la reconnaissance optique de caractères.
- Chapitre 2 est dédié à l'étude des caractéristiques de la langue Arabe.
- Chapitre 3 présente la méthodologie développée pour la réalisation de notre système.
- Chapitre 4 est consacré à la présentation des résultats d'implémentation et à l'étude des performances de notre système.

Nous terminerons par une conclusion générale qui inclut nos perspectives.

1.1 Introduction

La reconnaissance optique de caractères ou OCR (en anglais : Optical Character Recognition) est une technologie qui permet de convertir différents types de documents tels que les documents papiers scannés, les fichiers PDF ou les photos numériques vers des formats modifiables et exploitables. Sur le plan méthodologique, l'OCR propose des approches différentes suivant le mode d'écriture : manuscrit ou imprimé. Deux domaines distincts sont considérés, il s'agit de la reconnaissance statique, dite encore «hors-ligne », qui travaille sur un instantané d'encre numérique (sur une image) et la reconnaissance dynamique « en-ligne » où les symboles sont reconnus au fur et à mesure qu'ils sont écrits à la main. La technologie d'OCR a été appliquée ces dernières années à travers tout le spectre d'industries en train de révolutionner le processus de gestion des documents. Les systèmes d'OCR ont permis à des documents numérisés de se transformer en documents entièrement consultables avec le contenu du texte qui est reconnu par les ordinateurs. Cependant, après plus de deux décennies de recherche sur la numérisation des documents, ces systèmes peuvent encore laisser quelques imperfections pour parvenir à une réédition du document ce qu'il peut être dû aux différents problèmes dont la qualité du document et de l'impression, la discrimination de la forme, le type d'acquisition, les variations des dimensions, le nombre de scripteurs, la taille du vocabulaire, etc. Dans le reste du chapitre, nous allons donner un aperçu sur les différentes étapes d'un système OCR.

1.2 Définition

La Reconnaissance Optique de Caractères (OCR) est un procédé qui permet de traduire une image de texte numérisée en un document texte modifiable [2]. Les images sont représentées sous forme de matrices de pixels. Le but d'un OCR est la segmentation des images pour effectuer la reconnaissance de caractère. La segmentation de caractères est une opération permettant de décomposer une image en une sous-image de symboles individuels [3]. Un système de reconnaissance de caractères comprend principalement trois phases : le prétraitement, la segmentation et la reconnaissance.

1.3 Différents aspects de l'OCR

Actuellement, aucun système générique d'OCR n'est mis en œuvre. Les OCRs dépendent fortement du type de données traitées et de l'application visée. Plusieurs types des systèmes OCR existent, citons entre autre :

- Les systèmes « en-ligne » ou « hors-ligne » suivant le mode d'acquisition.
- Les approches globales ou analytiques.

1.3.1 Reconnaissance en-ligne et hors-ligne

Ce sont deux modes différents d'OCR. Chacun a ses outils propres d'acquisition et ses algorithmes de reconnaissance.

a La reconnaissance en-ligne (on-line)

Ce mode est généralement consacré à l'écriture manuscrite. Il s'opère en temps réel. Les symboles sont reconnus au fur et à mesure qu'ils sont écrits à la main. La possibilité de correction et de modification de l'écriture de manière interactive est un avantage majeur du mode vu la réponse en continu du système [4]. L'acquisition de l'écrit nécessite l'utilisation d'un équipement tel qu'une tablette graphique ou un stylo électronique.

b La reconnaissance hors-ligne (off-line)

La reconnaissance hors-ligne démarre après l'acquisition de l'écrit. Ce mode est adapté avec les documents imprimés et les manuscrits déjà rédigés. Il se rapproche du mode de la reconnaissance visuelle. Il est considéré comme le cas le plus général de la reconnaissance de l'écriture. L'interprétation de l'information est indépendante de la source de génération [5]. La reconnaissance en mode hors ligne dépend fortement de la qualité des images numérisées [6].

1.3.2 Les approches globales ou analytiques

Deux approches principales sont adoptées pour la reconnaissance des caractères dans la littérature :

a L'approche globale (*holistic approach*)

L'approche globale**,** également connue sous le nom d'approche sans segmentation. Considère les éléments du texte en totalité, sans les subdiviser en unités plus petites. Les approches globales sont plus faciles à mettre en œuvre. Cependant, il existe plusieurs inconvénients tels que la sensibilité au bruit et aux petites variations du modèle [7]. L'utilisation de l'approche holistique est limitée à un lexique prédéfini [2].

b L'approche analytique

Dans cette approche, les mots ne sont pas considérés en entier, mais comme des séquences d'unités de petite taille [7]. En d'autres termes, le mot est segmenté en unités qui peuvent être des graphèmes, des segments, des pseudo-lettres, fragments, etc. La reconnaissance du mot débute par la reconnaissance des entités segmentées puis la reconnaissance du mot, ce qui constitue une tâche délicate pouvant générer différents types d'erreurs [8]. La reconnaissance analytique est basée sur deux phases : la phase de segmentation et la phase d'identification des segments. La phase de segmentation est classifiée en deux catégories : la segmentation explicite (externe) ou la segmentation implicite (interne) [9]. Cette approche présente quelques avantages par rapport à l'approche globale, car elle est moins sensible au bruit et aux faibles variations du modèle. Elle présente aussi l'avantage de pouvoir se généraliser à la reconnaissance d'un vocabulaire sans limite à priori, car le nombre de caractères est naturellement défini. De plus l'extraction des primitives est plus aisée sur un caractère que sur une chaîne de caractères [10].

1.4 Procédures générale de l'OCR

La procédure générale d'un système OCR passe par 5 étapes, comme le montre la figure suivante :

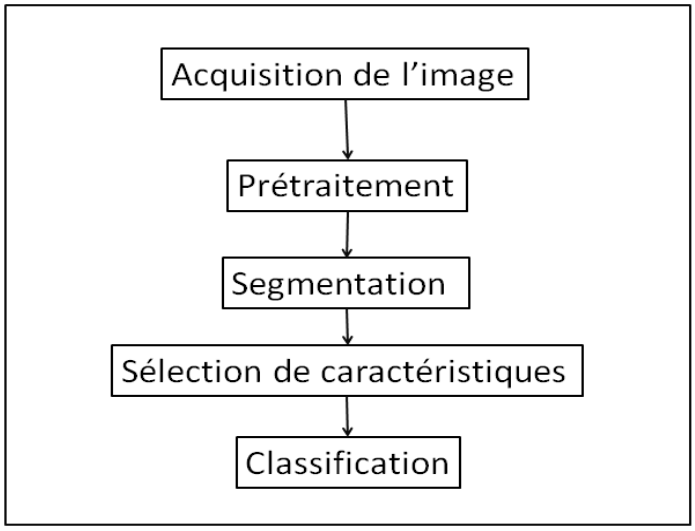


Figure 1.1 : Etapes générales de l'OCR.

1.4.1 Acquisition de L'image

Cette étape consiste à numériser (scan) le document et le stocker en tant qu'image, elle se fait par balayage optique. Le résultat du balayage optique est stocker dans un fichier de pixel, sa taille dépend de la résolution [11]. Les valeurs des pixels varient selon les types d'images [12]:

Les images binaires : comme leur nom indique, elles ne contiennent que 2 valeurs de pixels, 0 pour la couleur noir et 1 pour la couleur blanche. Les pixels dans ce type d'image est dit « monochrome ». L'image résultante est donc composée de noir et blanc uniquement.

Les images en niveau de gris : la valeur du pixel est un nombre qui représente la luminosité du pixel. Les valeurs possibles varient de 0 à 255. Généralement, zéro est considéré comme noir et 255 comme blanc. Les valeurs intermédiaires constituent les différentes nuances de gris.

Les images en couleurs : l'image est représentée en supposant un espace colorimétrique RVB (RGB en Anglais pour Red-Green-Blue). Par conséquent, la valeur du pixel est un vecteur de trois dimensions, chaque composante est un nombre qui varie entre 0 et 255.

A noter que jusque-là, l'image n'est qu'une matrice de pixels à exploiter pour extraire de l'information.

1.4.2 Prétraitement

Les manuscrits à numériser peuvent être écrits sur un arrière-plan bruyant ou coloré et la qualité des images peut être dégradée en raison du bruit introduit lors du processus de numérisation [13]. Cela empêche la distinction de la continuité des pixels qui constituent les mots. Il existe deux types de bruit [14]: le bruit indépendant/ dépendant du signal. Le bruit indépendant du signal ajoute un ensemble aléatoire de niveaux de gris aux pixels de l'image. Dans le bruit dépendant du signal, la valeur de chaque pixel est une fonction du niveau de gris. Le prétraitement vise à produire des données faciles à analyser par le système OCR, notamment pour le processus de la segmentation [13].

Ce processus inclut plusieurs étapes, à savoir [13][15] :

a La binarisation

La binarisation permet de passer à une image binaire composée de deux valeurs 0 et 1. Pour cerner et traduire les contrastes dans l'image, un seuil de binarisation est défini. Cependant, si dans l'image la distribution de niveaux de gris n'est pas clairement bimodale (une image peu

contrastées), il est difficile de fixer un seuil. La plus simple des façons pour le mesurer est de calculer l'histogramme des niveaux de gris de l'image [16]. Dans ce cas, la valeur du seuil est égale à la valeur du niveau de gris entre les deux pics de l'histogramme. Les pixels appartenant au fond sont ceux qui ont un niveau de gris supérieur à la valeur du seuil, les pixels ayant une valeur inferieur appartiennent donc à l'objet. La détermination du seuil peut être local ou global [17], les méthodes locales sont déterministes mais elles sont plus couteuse en termes de temps de calcul, contrairement aux méthodes globales qui ne donnent pas d'exactes résultats mais en temps plus réduit [18].

b ***Le lissage***

Durant le processus de l'acquisition, l'image est sujette à différents types de bruit. Elle peut avoir des cavités (absence de pixels) ou une surcharge. Pour corriger cela, nous appliquons le lissage sur les images bruitées. C'est l'analyse du voisinage des pixels pour éliminer les pixels isolés et pour boucher les cavités. Une façon classique pour appliquer le lissage consiste à parcourir l'image pixel par pixel, l'analyse de chaque pixel se fait en se basant sur ses 8 voisins. Les pixels de valeur 1 sont mis à 0 s'il n'y a pas assez de pixels noirs autour d'eux et les pixels de valeur 0 sont mis à 1 s'il n'y a pas assez de pixels blancs autour d'eux. Une autre approche consiste à inverser la valeur du pixel (entre 0 et 1) si la somme des pixels voisins est inférieure à un seuil déterminé.

c ***Squelettisation***

L'image binaire est une succession de traits d'épaisseurs variables, généralement avec bruit. Par conséquent, la squelettisation s'applique sur les images après la binarisation, son but est de réduire l'épaisseur du tracé des caractères à un pixel, en concevant sa continuité. La procédure consiste à faire des opérations d'érosion conditionnelle successive jusqu'à ce que le but soit atteint [19]. Le squelette obtenu doit préserver la forme du caractère, sa connexité, sa topologie et ses extrémités, et ne doit pas introduire du bruit [20].

d ***La suppression du bruit***

Lors de la phase d'acquisition, de nouveaux composants peuvent s'introduire à l'image ; c'est ce qu'on appelle « *brui t* ». Ce prétraitement consiste à détecter les pixels qui n'appartiennent pas aux composants significatifs et à les supprimer. Cela permet d'augmenter la discrimination [13].

e ***Le redressement***

Cela permet de corriger l'inclination de l'image due à un mauvais positionnement du document sur le scanner ou à une mise en page irrégulière. La première étape consiste à détecter l'angle de l'inclination puis à le corriger [15].

f ***La normalisation***

Ce prétraitement est appliqué sur les images pour réduire tous les types de variations, et pour obtenir des données normalisées. Cependant, il peut engendrer une distorsion et élimine quelques informations utiles. Usuellement, la normalisation porte sur la taille des caractères, l'inclination des lignes et l'inclination des caractères [13].

Après l'application des prétraitements nécessaires, un document propre et amélioré est obtenu pour la prochaine étape, en l'occurrence la segmentation. En principe, le document après le prétraitement ne comporte que les informations significatives des caractères.

1.4.3 La Segmentation

C'est le processus qui permet de découper l'image du texte en entité (phrase, graphème, mot, caractère, etc.), tout dépend de l'objectif du système. L'entité de sortie est toujours une matrice de pixels. Dans le cadre des OCRs systèmes, nous nous intéressons à la segmentation par caractères. C'est l'étape la plus décisive pour tout système OCR. Par conséquent le choix de l'algorithme est le facteur clé pour décider la précision du système [13]. Le processus de segmentation pour le problème de reconnaissance de caractères peut être divisé en trois niveaux : segmentation en lignes, segmentation en mots ou pseudo-mots et segmentation en caractères.

a **La segmentation en lignes**

Pour la segmentation du texte en lignes dans les AOCR (Arabic Optical Character Recognition), les lignes sont extraites en utilisant la méthode de la projection Horizontal [21]. La méthode calcule l'histogramme horizontal de l'image. Si le nombre de lignes blanches successives rencontrées (du haut vers le bas) est supérieur à un seuil fixé, le nombre de ligne est augmenté, puis le système extrait le début et la fin de chaque ligne. La Figure 1.2 montre un exemple de la segmentation en ligne.

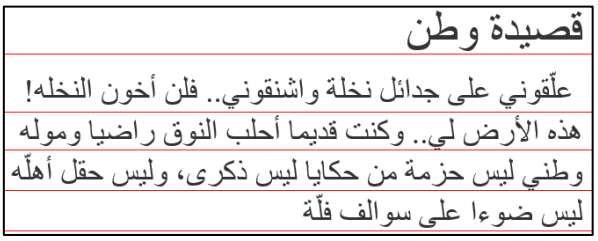

قصيدة وطن
علّقوني على جدائل نخلة واشنقوني.. فلن أخون النخله!
هذه الأرض لي.. وكنت قديما أحلب النوق راضيا ومواله
وطني ليس حزمة من حكايا ليس ذكرى، وليس حقل أهلّه
ليس ضوءا على سوالف فلّة

Figure 1.2 : Segmentation en ligne.

b Segmentation en pseudo-mots

Cette étape suit la segmentation en lignes de texte. Il s'agit d'analyser la ligne pour la découper en mots ou pseudo-mots. Cette étape est réalisée en déterminant l'histogramme des projections verticales des lignes pour détecter les espaces entre les mots et pouvoir les séparer. Le nombre de pixels noirs est compté sur chaque colonne de l'image texte. Si le nombre de pixels noirs n'est pas égal à zéro, il indique une partie connectée et par conséquent le texte n'est pas segmenté. D'autre part, si le nombre de colonnes blanches successives rencontré est supérieur à un seuil, la partie courante est segmentée. La figure 1.3 montre un exemple de la projection verticale pour une image donnée.


علّقوني على جدائل نخلة واشنقوني.. فلن أخون النخله! هذه الأرض

Figure 1.3 : Segmentation en pseudo mots.

c Segmentation en caractères

C'est l'étape la plus décisive pour tout système OCR. Par conséquent le choix de l'algorithme est le facteur clé pour décider la précision du système OCR [13]. Un point de segmentation se trouve dans la ligne de base qui ne contient aucune information. Il est situé entre deux caractères. La segmentation consiste à localiser deux points de segmentations, puis extraire le caractère entre eux.

Plusieurs méthodes ont été proposées pour la segmentation des mots en caractère dans les systèmes OCR analytiques. Dans ce qui suit nous allons présenter des techniques de segmentation des mots arabes imprimés en caractères. Les premiers systèmes utilisaient la projection verticale. Puis, la tendance était d'obtenir le squelette du mot et de l'analyser pour trouver les points de segmentation appropriés. Cette méthode a été suivie de tentatives de segmentation des mots en traçant le contour du mot. Plusieurs travaux ont été basés sur la

technique de Template-Matching seule ou combinée avec l'une des techniques précédemment citées.

Pour les systèmes OCR globaux, ils utilisent l'approche holistique dans laquelle les mots sont reconnus sans segmentation. La figure 1.4 illustre les techniques utilisées dans le domaine de la segmentation du texte arabe.

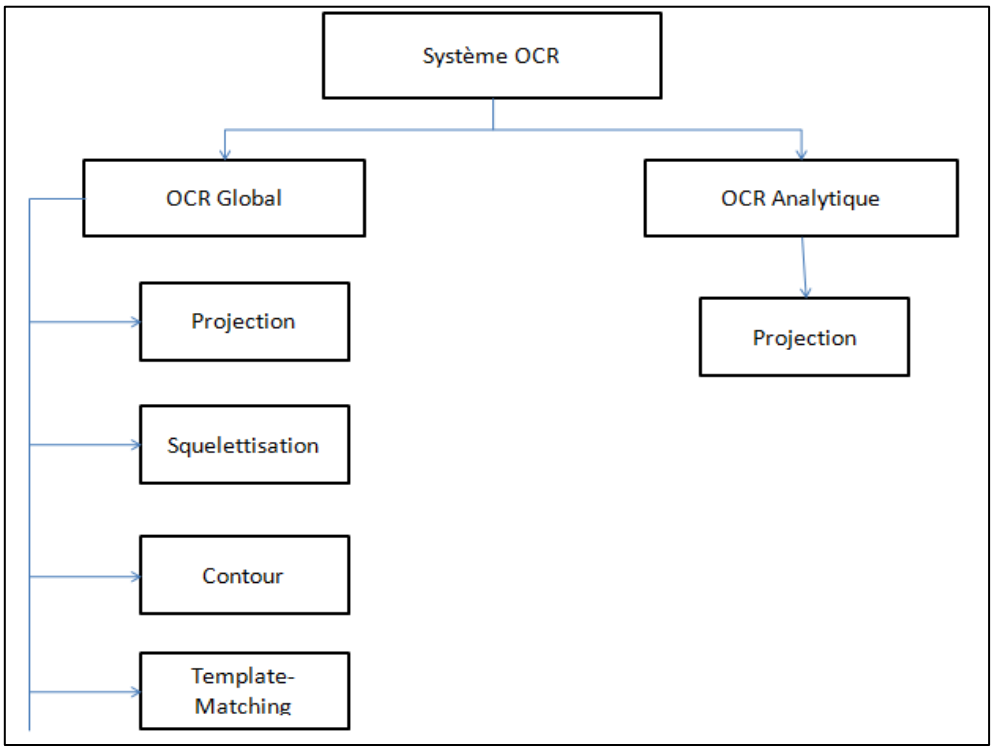


Figure 1.4 : Classification des techniques utilisées dans le domaine de la segmentation du texte Arabe.

- **La technique de segmentation basée sur la projection**

La projection horizontale est habituellement utilisée pour la segmentation de ligne et l'extraction de ligne de base, tandis que la projection verticale est utile pour segmenter les mots, les sous-mots et les caractères. Le but de cette méthode est de simplifier un système OCR en réduisant les informations 2D en 1D. Ces méthodes sont basées sur le fait que les traits de connexion sont toujours de moins d'épaisseur que d'autres parties des mots. Dans ces procédés, on calcule la projection verticale de l'image :

$$V_i = \sum P(i,j)$$

Où $P(i,j)$ est la valeur de pixel à la position (i,j) qui est soit 0 (blanc) soit 1 (noir). L'ensemble V des V_i forme l'histogramme de l'image, qui peut être manipulé de plusieurs manières pour déterminer les points de segmentation.

- **La technique de segmentation basée sur la squelettisation**

Le squelette d'une forme contient ses informations essentielles. C'est pourquoi la squelettisation est une technique qui peut aider à résoudre le problème de segmentation de caractères [22]. Diverses méthodes ont été proposées pour extraire le squelette. Dans cette méthode, le système crée une image contenant le squelette à traiter. Ensuite, à partir du squelette généré, la technique cherche à identifier les points de liaison entre les caractères en introduisant des calculs de courbures et d'angles basés sur des seuils prédéfinis. *La figure 1.5* montre un exemple d'un mot arabe et son squelette.

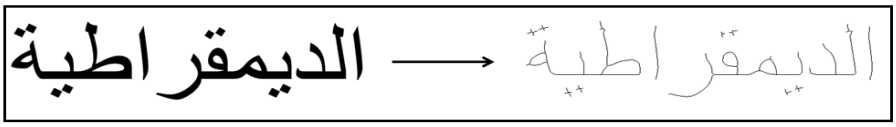


Figure 1.5 : La squelettisation.

Dans le travail proposé par El-Khaly et Sid-Ahmed [23] La ligne de base du mot aminci (squelette) est trouvée en première étape, c'est la ligne qui contient le nombre maximal des points noirs. Puis seulement les colonnes qui n'ont pas de pixels au-dessus ou en dessous de la ligne de base sont considérés pour trouver les points de segmentation. Le point de segmentation sera au milieu du segment de connexion.

La méthode de segmentation développée par H.Goraine et M. Usher [24] est constituée de trois principales étapes. Tout d'abord, le squelette du mot est obtenu en utilisant l'algorithme de Hilditch [25]. La deuxième étape consiste à trouver le point de départ en utilisant l'organigramme présenté dans la figure 1.6 tout en se basant sur une classification des points rencontrés : point de connexion, point caractéristique ou jonction, trait. La troisième étape a pour but de trouver le point final et ensuite de tracer un trait depuis le début jusqu'à la fin comme montré dans la Figure 1.7. Ensuite, le tracé est isolé et éliminé de l'image. Le premier point (à droite) détecté est pris comme point final du tracé suivant, et la procédure de segmentation est répétée jusqu'à ce qu'il n'y ait plus de traits.

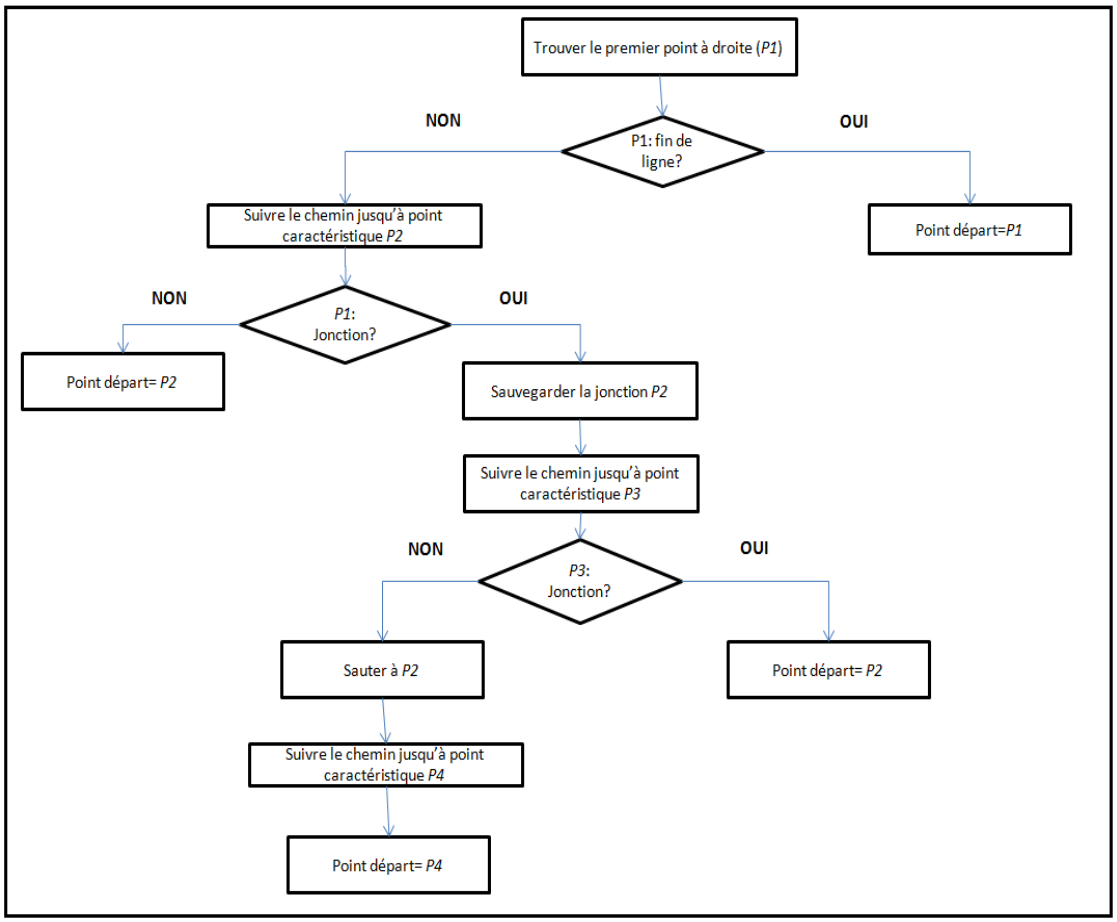


Figure 1.6 : Flow-chart de détermination du point de départ.

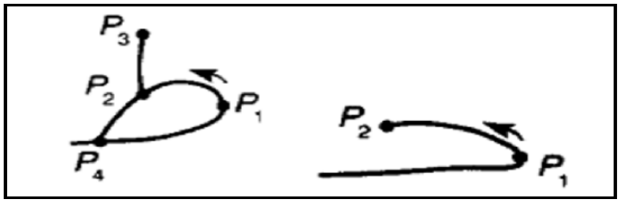


Figure 1.7 : Construction du caractère.

Parmi les inconvénients des méthodes basées sur la squelettisation, les différents algorithmes d'amincissement peuvent produire différents caractères amincis, de plus, le processus d'amincissement peut modifier la forme du caractère, en particulier dans le cas des images de mauvaise qualité qui, à leur tour, rendent le caractère difficile à reconnaître. Certains problèmes courants rencontrés pendant le processus de squelettisation comprennent l'élimination ou l'érosion des caractères secondaires. Ces modifications rendent la

reconnaissance de l'image du squelette une tâche difficile même pour le traitement visuel humain de la nature.

- **La technique de segmentation basée sur le contour**

Cette technique est basée sur le traçage des contours d'un mot pour la segmentation. A partir du corps principal du mot, le système génère le traçage du contour extérieur. Ensuite, le système fait un balayage et une analyse sur le traçage du contour et les règles topologiques est réalisé afin de détecter les points de liaison des caractères ou graphèmes en se basant sur les locaux du contour inférieur de chaque pseudo-mot. Ces derniers sont faciles à ajuster lorsque la qualité de l'écriture est bonne, comme ils peuvent y avoir des comportements erratiques lorsque l'écriture est de mauvaise qualité [22][26]. Les résultats de cette technique sont très remarquables par rapport aux autres techniques par ce que le contour reste le meilleur choix à utiliser pour isoler les caractères connectés, et ceux en chevauchement [22]. Parmi les problèmes rencontrer lors de l'analyse du contour pour la segmentation est la discontinuité de celui-ci, et cela peut être du au bruit, c'est pour cela que la phase de prétraitement doit éliminer au maximum le bruit présent dans l'image. Un autre problème dans cette technique est la détection de la ligne de base devient très difficile [22]. La figure **1.8** illustre la technique.

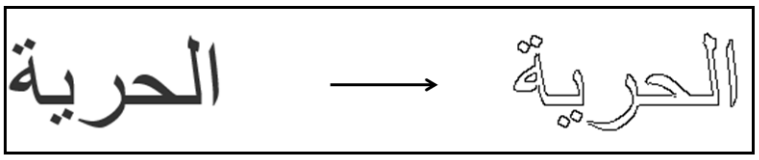


Figure 1.8 : Technique de segmentation basée sur le traçage des contours.

La segmentation des pseudo-mots proposée par Bushofa.BMF et Spann.M [27] examine le contour supérieur pour les points candidats. Cela se fait en traçant cette partie de gauche à droite en commençant par le premier point au-dessus de la ligne de base. Lorsqu'un point maximum dans la direction verticale est atteint, il est considéré comme un pic si sa valeur est supérieure à une valeur du seuil ($t1$ = ligne de base + $t/6$) où t représente la distance entre le top du mot et la ligne de base. Après cette étape, la valeur des coordonnées de contour commence à descendre jusqu'à ce qu'elle arrive à un point minimum : Figure 1.9.

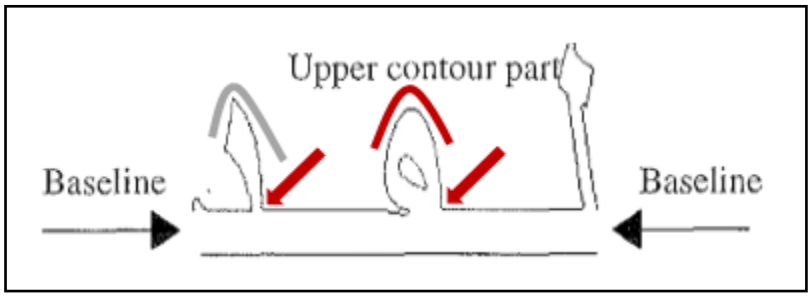


Figure 1.9 : Exemple de localisation des points de segmentation (flèche).

Si cette valeur est inférieure à la valeur de seuil $t1$, elle est considérée comme un point de segmentation à condition qu'elle soit suivie d'un pic. Cette procédure se poursuit jusqu'à ce que toutes les coordonnées de la partie supérieure du contour soient examinées. Le point de segmentation est considéré entre deux pics. Si aucun pic n'a été trouvé après avoir rencontré un point minimum, ce point minimum est négligé. De plus, si deux points minimum ou plus sont trouvés entre deux pics et les deux satisfont la condition de seuil, le point le plus proche du premier point de crête et le plus proche de la ligne de base est pris comme point de segmentation. Cette procédure se poursuit jusqu'à ce que tous les points de segmentation soient trouvés. La partie inférieure du contour est d'abord examinée pour voir s'il y a des caractères touchants ou s'il existe un (ى). Les caractères arabes peuvent se toucher sous la ligne de base, comme le montre la figure 1.10.

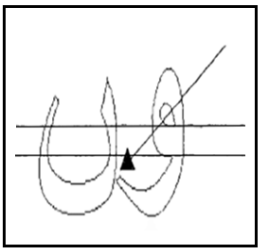


Figure 1.10 : Caractères touchants sous la ligne de base.

La segmentation des caractères touchants est obtenue en traçant la partie inférieure du contour de droite à gauche. Les valeurs les plus basses dans la direction verticale sont enregistrées. Le point de contact se trouve entre deux de ces points est la valeur la plus élevée dans la direction verticale qui satisfait la condition de seuil mentionnée avant. Une fois qu'un point touchant est trouvé, les caractères sont séparés. Cela conduit à diviser le contour en deux parties ou plus en

fonction du nombre de caractères touchants. Par conséquent, le contour de la première partie est extrait à nouveau. La partie inférieure de ce contour est examinée pour l'apparition du caractère (ى). Dans le cas idéal, le contour du caractère (ى) rencontre à la ligne de base quatre fois comme le montre la Figure 1.11. Cependant, cela peut varier en fonction de l'effet du bruit dans l'image. Lorsque cette condition n'est pas satisfaite, les changements de signe dans la direction horizontale sont considérés. Cinq changements de signe ou plus indiquent la présence de ce caractère. Lorsque ce caractère est identifié, les deux caractères sont séparés en un point entre eux comme le montre la Figure 1.11.

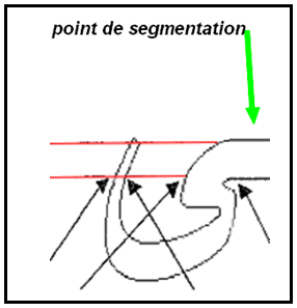


Figure 1.11 : Segmentation du caractère ى.

La segmentation dans [28] est basée sur le contour du corps principal des mots. Tout d'abord, le point de début et de fin du contour supérieur est déterminé. Il est important de trouver le point inférieur droit et inférieur gauche du contour en raison des longues lignes verticales au début d'un mot. Ensuite, une segmentation du contour supérieur en parties se fait par une courbure du même signe. En commençant par une courbure positive par exemple, le changement à une courbure négative va terminer ce segment et commencer par un nouveau. Un filtre passe-bas sur les points de contour sert à réduire la sensibilité au bruit de cette procédure. Dans certains cas, la ligne horizontale entre les caractères diffère beaucoup, mais cette longueur ne contient aucune information. Ensuite, un détecteur de ligne horizontale est utilisé pour marquer ces lignes comme sans importance pour le processus de reconnaissance. La classification dans ce système requiert des caractères segmentés et cette segmentation se réalise à l'étape de la reconnaissance. Un caractère ne peut être segmenté que si la classification des segments correspondants est réussie. Le taux de segmentation n'a pas été mentionné, mais le taux de la reconnaissance était de 96.9%

D'un autre coté, C. Olivier, H. Miled [29] ont proposé une méthode qui utilise le contour supérieur pour segmenter le texte arabe manuscrit. Tout d'abord, le contour de chaque pseudo-mot est detecté en utilisant le code de Freeman [30]. Le contour est étudié dans le sens inverse des aiguilles de la montre, ensuite le contour supérieur est déterminé par les directions F3, F1 et F5 comme montré dans la Figure 1.12. Un filtre est appliqué pour ne garder que les points du contour –supérieur (situés dans la partie supérieure du pseudo-mot). Puis un automate est utilisé pour détecter les minimums locaux du contour supérieur, ces minimums sont pris pour la segmentation du pseudo-mot. Le système est évalué sur une base de données contenant 6000 mots arabe, qui sont les noms du peuple tunisien écris par 20 fontes différentes. Ils ont pu avoir un taux d'erreur de 2.59%.

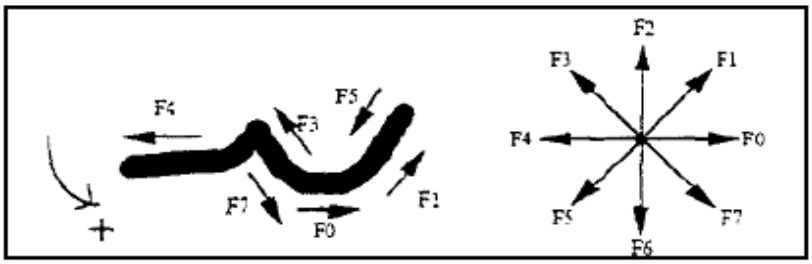


Figure 1.12 : Détection du contour supérieur en utilisant les codes de Freeman.

La méthode présentée par Sari et al. [31] utilise le contour pour détecter les points de segmentation en appliquant des règles aux minimums locaux du contour inférieur de chaque pseudo-mot. Les caractères, verticalement chevauchés en raison de la nature de la fonte ou du style d'écriture (italique) ont été traités dans un stade avancé. Le taux de réussite était de 86% sur un ensemble de données limité de 100 mots seulement. Par conséquent, la méthode présentée ci-dessus montre la nécessité de combiner les fonctionnalités de contour avec d'autres fonctionnalités afin d'obtenir une meilleure segmentation de caractères, ce qui donnerait finalement un taux de reconnaissance plus élevé pour les systèmes OCR.

Le contour reste le meilleur choix à utiliser pour isoler les caractères connectés, et ceux en chevauchement. Les résultats de cette technique (contour) sont très remarquables par rapport aux autres techniques vus précédemment. Cependant, la segmentation basée sur la détection du contour a ses points a ses points faibles. Parmi les problèmes qu'on peut rencontrer lors de l'analyse du contour pour la segmentation est la discontinuité de celui-ci, et cela peut être dû au bruit. C'est pour cela que la phase de prétraitement doit éliminer au maximum le bruit

présent dans l'image. Un autre problème pour les systèmes qui utilisent en plus du contour, la ligne de base pour la segmentation, la détection de la ligne de base devient très difficile et ses problèmes influent sur tout le système. Un autre problème de l'utilisation du contour pour la segmentation c'est qu'il y a plusieurs méthode de détection de contour tels que le filtre de Sobel, filtre de Prewitt, filtre de Canny [32]**,** ou l'annulation du Laplacien, et chaque méthode peut donner un contour diffèrent. Et il faut également noter que les systèmes de la segmentation basés sur le contour montrent une très faible performance face aux ligatures.

- **La technique de segmentation basée sur Template Matching**

C'est une technique de traitement d'image numérique pour la recherche de petites parties d'une image qui correspondent à une image de modèle. Il s'agit d'une technique de visibilité automatique de haut niveau qui destine les parties d'une image qui correspondent à un modèle prédéfini [33]. Cette méthode est flexible et relativement simple à utiliser, ce qui en fait l'une des méthodes les plus connues. Template Matching a été utilisé dans les applications de reconnaissance de caractères. Elle est utilisée comme une technique pour reconnaître des caractères ou des mots avec une référence à une base de données stockée contenant un ensemble d'images pour les caractères ou les mots [33]. Pour cette technique, la partie du caractère recherchée dans l'image doit être choisie manuellement pour être utilisée à la comparaison.

B.Bushofa et M.Spann [34] ont proposé une méthode pour chercher l'apparition d'un angle formé par la jonction de deux caractères à la ligne de base (Figure 1.13) en utilisant une fenêtre 7 × 7, pour examiner le voisinage des caractères. Cet angle est pris comme un point de segmentation potentiel. Pour valider les points précédemment calculés, le caractère extrait est comparé avec un model déjà stocké manuellement.

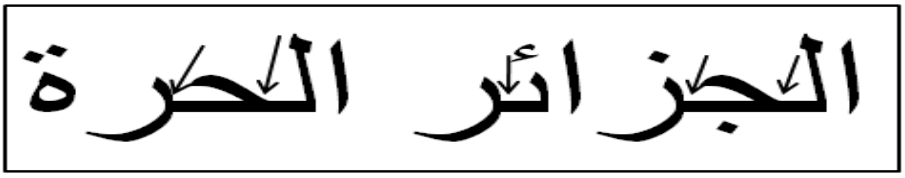


Figure 1.13 : Les angles de jonction entre les caractères.

D'un autre coté, P.Bharatratna , R.Ramesh et M.Ganesh [35] ont utilisé la technique du Template Matching dans un système de segmentation automatique d'une scène vidéo pour séparer le script du reste de la capture. La Figure 1.14 montre le système développé. Le

système a été testé sur 35 vidéos avec 700 images pour chaque vidéo. Le résultat de l'expérimentation en termes de précision du système est de 91,52%.

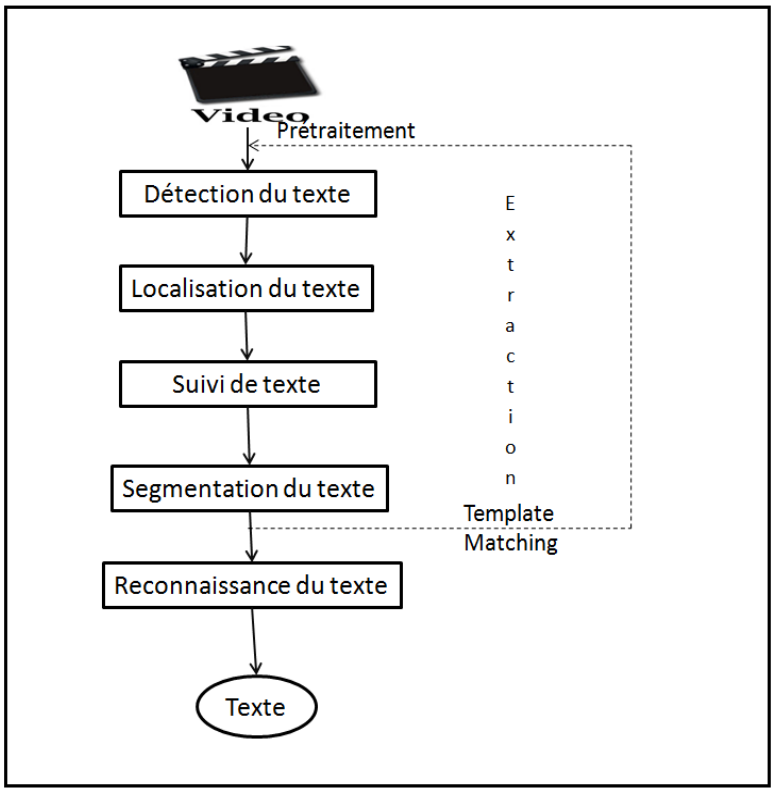


Figure 1.14 : Système de segmentation du texte à partir d'une capture vidéo [35].

Bien que la Template-Matching ait pu obtenir de bons résultats, sa réussite à trouver l'angle approprié dépend fortement du bruit dans l'image.

1.4.4 L'extraction de caractéristiques :

L'extraction de caractéristiques est une étape cruciale dans les systèmes de reconnaissance. Cette étape est réalisée pour obtenir l'information la plus discriminante et de dimension limitée pour l'étape de la classification, toute en évitant le risque de perte des informations importantes et signifiantes [36]. Elle consiste à représenter un caractère par un vecteur de caractéristiques (primitives) de dimension fixe. Le codage le plus élémentaire consiste à construire un vecteur constitué d'autant de composantes qu'il y a de pixels dans l'image. En effet, un mauvais choix des primitives influence négativement et nettement les résultats même si on utilise un classifieur performant. Les différentes techniques d'extraction sont classées en fonction des types de primitives : caractéristiques globales, caractéristiques structurelles, caractéristiques

morphologiques, caractéristiques de contour, caractéristiques statiques, caractéristiques métriques et caractéristiques adaptatives [37] . Un choix judicieux de caractéristique permet d'améliorer le calassifieur en le simplifiant et le rendant plus robuste. Le nombre de caractéristiques du vecteur est fonction de la complexité des formes à identifier. Un grand nombre de composantes permet de distinguer mieux les différentes classes mais complique le classifieur. A l'inverse un faible nombre permet d'obtenir une reconnaissance très rapide mais moins fiable dans la séparation des classes. Un très grand nombre de techniques d'extraction de caractéristiques a été mis en œuvre afin de tenter d'optimiser ce compromis.

Exemples de caractéristiques liées aux carac1ères :

Les techniques d'extraction de caractéristiques pour la reconnaissance de caractères hors-ligne sont basées sur l'analyse d'image. Les différentes techniques sont classées en fonction des types de primitives. Selon Amin [7] les caractéristiques peuvent être classées en deux catégories :

• ***Les caractéristiques locales*** : qui sont souvent géométriques (par exemple : concavité/convexité, les fins de traits, les jonctions (en *T* ou *Y*), ainsi que les intersections en (*X*) ...etc.

• ***Les caractéristiques globales*** : qui sont souvent topologiques (connectivité, nombre de composantes connexes, …etc.) ou statistiques (transformé de Fourier, moments invariants…etc.).

1.4.5 La Classification

La classification est une forme d'analyse de données qui extrait des modèles décrivant des classes de données. Ces modèles, appelés classifieurs, prédisent des étiquettes de classe. Cette analyse peut nous aider à mieux comprendre l'ensemble de données. De nombreuses méthodes de classification ont été proposées par les chercheurs en apprentissage automatique. Plus concrètement, la classification de données passe par deux grandes étapes :

a L'étape de l'apprentissage

Dans cette étape le modèle de classification est construit, décrivant un ensemble prédéterminé de classes de données ou de concepts. Dans cette étape se fait l'apprentissage des caractères à classifier plus tard. Cette phase est nécessaire pour stocker les caractéristiques de chaque caractère dans la mémoire, ceci avant d'effectuer la reconnaissance. Certains

auteurs ont élaboré des systèmes de reconnaissance sans avoir recours à l'apprentissage [38], mais aucun de ces systèmes n'est fiable [38].

b L'étape de la classification

Dans cette étape le modèle construit est utilisé pour prédire les étiquettes de classe des données en entré, c'est-à-dire l'identification des caractères appris au préalable. Cette étape permet de déterminer, à partir des caractéristiques apprises, la classe d'appartenance du caractère en entré.

Soit $L = \{(x_i, y_i) | i = 1, \dots, m\}$ l'ensemble de données, tel que x_i est le vecteur caractéristique de la $i^{ème}$ instance et $y_i \in \{1, .., C\}$ l'étiquette de la classe correspondante. Puisque l'étiquette de classe de chaque instance est fournie, l'ensemble de données est sujet à un apprentissage *supervisé*. C'est-à-dire, il est indiqué à quelle classe appartient chaque instance d'apprentissage. Contrairement à l'apprentissage *non supervisé*, ou l'étiquette de classe n'est pas connu à priori, et le nombre de de classe à apprendre n'est pas connu. Ce type d'apprentissage s'appelle le *clustering*(regroupement),quand l'étiquette de classe n'est pas fournie, nous essayons de déterminer les groupes d'instances similaires.

Comme mentionné ci-dessus, le model est ensuite utilisé pour la classification. Pour tester les performances du model établi, la précision prédictive est calculée. Ceci en utilisant un autre ensemble de données, ensemble de test, pour la simple raison : si on utilise le même ensemble que celui de l'apprentissage, l'estimation de la précision de la classification sera probablement optimale. La précision d'un classifieur sur un ensemble de test est le pourcentage d'instances correctement classés par le classifieur. Dans la littérature, il existe plusieurs mesures d'évaluation de la précision. Après le calcul, si la précision est acceptable, le classifieur peut être utilisé pour classifier les futures instances, pour lesquelles l'étiquette de classe n'est pas connue.

- ***k* plus proches voisin *knn***

k plus proche voisin est un classifieur assez trivial et l'un des plus simples. Il stocke tous les ensembles de données d'apprentissage. ***Knn*** ne classe les objets que si leurs attributs correspondent à l'un des exemples d'apprentissage déjà enregistrés. ***knn*** [39] forme un voisinage des k objets les plus similaires de l'ensemble de données d'apprentissage qui sont proches de l'objet à tester et attribue une étiquette à la classe prédominante dans ce voisinage.

Pour ce faire, l'algorithme est basé sur deux paramètres clés: une métrique de distance ou de similarité pour calculer la distance entre les objets et la valeur de k, le nombre de voisins les plus proches. Pour classer un objet non étiqueté, ses k plus proches voisins sont découverts, en calculant la distance de cet objet aux objets étiquetés. Ensuite, les étiquettes de classe de ces voisins les plus proches sont utilisées pour déterminer l'étiquette de classe de l'objet. Un des problèmes majeurs qui affecte les performances de ***knn*** est le choix de k [40]. Un petit k peut déduire un résultat sensible au point de bruit. Un grand k peut déduire trop de points dans le voisinage à partir d'autres classes.

- **Machine à vecteur de support $\boldsymbol{SVM}$**

Les machines à vecteurs de support (***SVM***) [41] sont considérées comme les méthodes les plus robustes et les plus précises [40]. Dans une tâche d'apprentissage à deux classes, ***SVM*** vise à trouver la meilleure fonction de classification permettant de distinguer les membres des classes dans les données d'apprentissage. La métrique du concept de «meilleure» fonction de classification est linéaire. C'est un hyperplan séparateur $f\ (x)$ qui passe au milieu des deux classes. Il utilise des noyaux pour construire des limites de classification linéaires dans des espaces à plus grande dimension. Pour la classification multiclass, certaines extensions ont été proposées. Les approches pour les ***SVM*** multiclasses sont divisées en deux catégories, la première consiste à construire et à combiner plusieurs ***SVM*** binaires, la seconde consiste à prendre en compte toutes les données dans une formule d'optimisation [42].

- **Arbre de décision $\boldsymbol{DT}$**

L'apprentissage par arbre de décision [43] est l'une des méthodes les plus utilisées et les plus pratiques pour l'inférence inductive. L'arbre de décision trie les instances de la racine à un nœud feuille qui effectue la classification de l'instance. Les nœuds représentent un test de certains attributs de l'instance. Les branches descendant de ces nœuds représentent l'une des valeurs possibles pour ces attributs. Le test commence par le nœud racine de l'arbre et descend en suivant la branche correspondante à la valeur de l'attribut dans le nœud donné. Ce processus est ensuite répété pour le sous-arbre enraciné sur le nouveau nœud.

- **Modèle de Markov caché ((HiddenMarkovian Model $\boldsymbol{HMM}$))**

(Hidden Markovian Model HMM) sont des approches stochastiques qui résolvent parfaitement l'incertitude des données dans le problème de reconnaissance de formes (reconnaissance des caractères manuscrit par exemple), en s'appuyant sur des modèles probabilistes [44]. Les $HMMs$ traitent les données comme une séquence d'observations, en utilisant des états cachés qui sont connectés les uns aux autres par des probabilités de transition.

Un HMM est caractérisé par [45] :

- N, le nombre d'états dans le modèle.
- M, le nombre de symboles d'observation distincts.
- A, la distribution de probabilité de transition.
- B, la distribution de probabilité de symbole d'observation pour chaque état.
- π, la distribution initiale de l'État.

Contrairement à une approche fondée sur les connaissances, les HMM utilisent des algorithmes statistiques qui peuvent automatiquement extraire les connaissances des échantillons [44].

1.5 Conclusion

Dans ce chapitre, nous avons présenté les systèmes OCRs d'une manière général. Nous avons présenté les différents types de reconnaissance ainsi que les étapes d'un système OCR. L'étape de segmentation a été bien étudiée vu son rôle crucial dans les systèmes OCRs. Dans ce qui suit, nous allons donner un aperçu sur la langue Arabe puis les problèmes posés avec les OCR spécifique à la langue Arabe.

Chapitre 2

L’OCR et l’Arabe

2.1. Introduction

La reconnaissance de l'écriture Arabe manuscrite ou imprimé, bien qu'elle remonte à des années reste encore aujourd'hui au niveau de la recherche et de l'expérimentation. La nature de l'écriture Arabe très cursive fait un terrain privilégié mais difficile pour le développement des systèmes de reconnaissance d'écriture. Malgré les difficultés rencontrées avec la reconnaissance du texte Arabe, ces dernières années, ce domaine a pris un nouvel essor et fait l'objet d'applications de plus en plus nombreuses dans des domaines différents. Citons entre autres : le traitement automatique des dossiers et des formulaires administratifs, le tri postal, la lecture de chèques bancaires, la numérisation et sauvegarde du patrimoine culturel manuscrit, etc. Dans ce chapitre, nous présentons les caractéristiques morphologiques de l'écriture Arabe ainsi que les difficultés liées à l'AOCR.

2.2. Caractéristiques de l'écriture Arabe

L'Arabe est écrite par plus de 250 millions de gens [46], dans plus de vingt pays différents. L'écriture Arabe a été développée à partir d'un type d'Araméen [47]. La langue araméenne comporte moins de consonants que l'Arabe, alors de nouvelles lettres ont été créées en ajoutant des points aux lettres déjà existantes. L'Arabe est une écriture consonantique qui utilise un alphabet de 28 lettres (Tableau 2.1) auquel il faut ajouter la Hamza « ء », qui est le plus souvent considérée comme signe complémentaire [48].

Tableau 2.1 : L'alphabet Arabe dans ses différentes formes.

Caractère	Initiale	Médiane	Finale	Isolé
Alif			ـا	ا
Ba	بـ	ـبـ	ـب	ب
Ta	تـ	ـتـ	ـت	ت
Tha	ثـ	ـثـ	ـث	ث
Jim	جـ	ـجـ	ـج	ج
Ha	حـ	ـحـ	ـح	ح
Kha	خـ	ـخـ	ـخ	خ
Dal			ـد	د
Thal			ـذ	ذ
Ra			ـر	ر

Zay			ـز	ز
Sin	ـسـ	ـسـ	ـس	س
Chin	ـشـ	ـشـ	ـش	ش
Sad	ـصـ	ـصـ	ـص	ص
Dhad	ـضـ	ـضـ	ـض	ض
Tad	ـطـ	ـطـ	ـط	ط
Dha	ـظـ	ـظـ	ـظ	ظ
Ayn	ـعـ	ـعـ	ـع	ع
Ghayn	ـغـ	ـغـ	ـغ	غ
Fa	ـفـ	ـفـ	ـف	ف
Qaf	ـقـ	ـقـ	ـق	ق
Kaf	ـكـ	ـكـ	ـك	ك
Lam	ـلـ	ـلـ	ـل	ل
Mim	ـمـ	ـمـ	ـم	م
Noun	ـنـ	ـنـ	ـن	ن
He	ـهـ	ـهـ	ـه	ه
Waw			ـو	و
Ya	ـيـ	ـيـ	ـي	ي

La hamza « ء » a une orthographe spéciale qui dépend de règles grammaticales, ce qui multiplie les formes nécessaires à sa représentation, puisqu'elle peut s'écrire seule ou sur le support de trois voyelles (alif, waw et ya) dont elle suit le code (Tableau 2.2).

Tableau 2.2 : Hamza et Madda et les positions qu'elles occupent en association avec Alif, Waw et Ya.

Caractère	Initiale	Médiane	Finale	Isolé
Alif + ~			ـآ	آ
Alif + ء			ـأ	أ
			ـإ	إ
Waw+ ء			ـؤ	ؤ
Ya+ ء	ـئـ	ـئـ	ـئ	ئ

De plus l'alphabet Arabe comprend d'autres caractères additionnels tels que « ﺀ» et « ﻷ», de ce fait, nous pouvons considérer que l'alphabet Arabe comprend plutôt 31 lettres que 29. La considération du symbole « ~» qui s'écrit uniquement sur le support du caractère « ﺍ», fait apparaître d'autres graphismes (Tableaux 2.2 et 2.3).

Tableau 2.3 : Hamza et Madda et les positions qu'elles occupent en association avec le caractère « ﻷ».

Caractère	Initiale	Médiane	Finale	Isolé
Lamalif +~			ﻟﺂ	ﻵ
Lamalif ﺀ+			ﻟﺄ	ﻷ
			ﻟﺈ	ﻹ

Tableau 2.4 : Les caractères additionnels.

Caractère	Initiale	Médiane	Finale	Isolé
Ta marbouta			ﺔ	ﺓ
Lamalif			ﻼ	ﻻ

2.3. Spécificités de la langue Arabe :

La langue Arabe a plusieurs spécificités, citons :

- Les caractères Arabes s'écrivent de façon cursive, de droite vers la gauche, aussi bien dans le cas de l'imprimé que du manuscrit.
- Un trait caractéristique de l'écriture Arabe est la présence d'une *ligne de base* horizontale dite encore lige de référence ou d'écriture. C'est le lieu des caractères d'une même chaîne (figure 2.1).

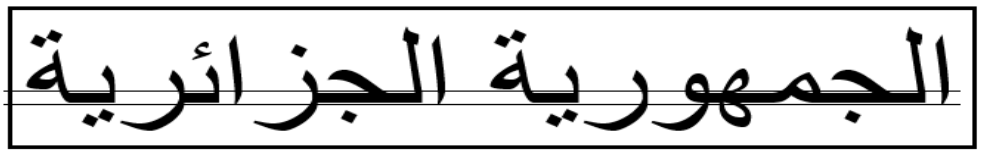


Figure 2.1 : Exemple d'écriture Arabe montrant la ligne de base.

- Les dimensions des caractères (chasse et hauteur) sont variables, même s'il s'agit des différentes formes d'un caractère (Tableau 2.1-2.4).
- La forme d'une lettre écrite dépend de son contexte et le dessin du glyphe associé diffère selon que le caractère apparaît en position initiale, médiane ou isolée dans une chaîne de

caractères (Tableau 2.5). A chaque caractère peut correspondre jusqu'à quatre glyphes différents ce qui lève à environ 100 le nombre de formes à reconnaître. Les formes correspondantes à un même caractère, souvent appelées « formes internes », présentent parfois de sensibles différences ; dans certains cas, il est même difficile d'en déduire s'il s'agit d'une même lettre. L'alphabet Arabe est représenté numériquement par un code d'échange de communication standard approuvé par l'Organisation Arabe de normalisation et de métrologie (ASMO) [49]. Semblable au code américain standard d'échange d'informations (ASCII), chaque caractère du code ASMO est représenté par un octet. Une lettre en latin a deux formes possibles, majuscules et minuscules. Le code ASCII fournit des représentations séparées pour ces deux formes, alors qu'une lettre Arabe n'a qu'une seule représentation en ASMO.

Tableau 2.5 : Les quatre formes des caractères « ain » et « he » en fonction de leur position dans la chaîne de caractères.

Position	Initiale	Médiane	Finale	Isoléé
Mots	عالم	معلم	سمع	ورع
	همس	مهد	لعبه	منتزه

- Plus de la moitié des caractères Arabes (16) incluent dans leur forme des points qui peuvent être au nombre de 1, 2 ou 3. Ces points peuvent se situer au dessus ou en dessous du corps du caractère, mais jamais en haut et en bas simultanément [48][50].
- Certains caractères Arabes incluent une boucle qui peut avoir différentes formes (Figure 2.2).

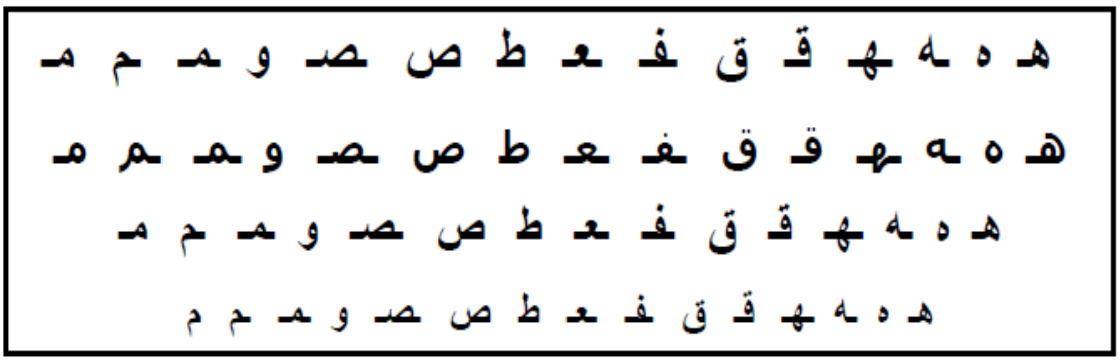


Figure 2.2 : Exemple de formes de boucles dans des styles différents.

- Certains caractères ne peuvent être rattachés à leur gauche et de ce fait ne peuvent se trouver qu'en position isolée ou finale, ce qui donnent, quand ils existent, des mots composés d'une ou de plusieurs parties qu'il est convenu d'appeler généralement PAW (*Peace of Arabic Word*) ou encore pseudo-mot [48]. Un PAW correspond donc à une chaîne d'un ou de plusieurs caractères (Tableau 2.6). L'écriture Arabe est ainsi semi-cursive plutôt que totalement cursive.

Tableau 2.6 : Exemple de mots par nombre de PAW 1,2,3,4 et 5 respectivement.

Nombre de PAW	Mot
1	مكثر ,مطر ,بلد
2	تونس, برد, عابد
3	العالية ,التاجر, البقاع
4	الجزائر ,الزهور, البذور
5	الرمادية, الوردية, الأزهار

Comme le caractère, le PAW peut se trouver dans des mots différents à des positions différentes, mais contrairement au caractère, il présente une structure morphologique stable, il garde la même calligraphie dans les différentes positions qu'il occupe (Tableau 2.7).

Tableau 2.7 : Le PAW « قر » dans différents mots et différentes positions.

Position	Initiale	Médiane	Finale	Isolé
Mot	**قر**ار	ر**قر**اق	أ**قر**	**قر**

- Pour des raisons de justification de texte et/ou d'esthétique, les ligatures horizontales peuvent être allongées en insérant entre les caractères d'une même chaîne une ou plusieurs élongations « matta » (ou tatwil), correspondant au symbole «-». L'élongation se situe toujours à gauche du caractère courant. Si le trait d'allongement est associé à un caractère en position de début ou finale, le caractère prend sa forme de milieu et voit sa chasse augmenter du nombre de « matta » insérées (Tableau .28) [51].Au niveau du PAW, l'insertion de traits d'allongement affecte uniquement sa largeur, la morphologie reste la même comme indiqué dans la figure 2-3 [52]. Les éditeurs de texte tels que Word

de Microsoft, insèrent dans les lignes de texte, le nombre approprié de « Matta », pour la justification gauche-droite d'un texte Arabe.

Tableau .28 : Exemples de caractères avec et sans matta.

Avec 6 mattas	Avec 3 mattas	Avec 1 mattas	Sans mattas
بـــــ	بـــ	بـ	ب
حـــــ	حـــ	حـ	ح
ســـــ	ســـ	ســ	س
فـــــ	فـــ	فـ	ف
مـــــ	مـــ	مـ	م
يـــــ	يـــ	يـ	ي

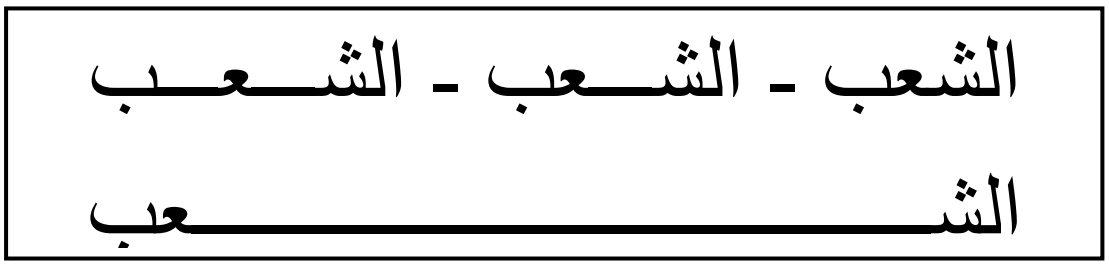


Figure 2.3 : Variation du mot «**الشعب** **»** écrit avec un nombre différent de traits d'allongement dans différentes positions.

- Le mot Arabe n'a pas de longueur fixe, il peut comprendre un ou plusieurs PAWs incluant chacun un nombre différent de caractères. De plus, différentes chasses possibles peuvent être associées à un même mot, en insérant un nombre variable de traits d'allongement.
- Dans certaines fontes plusieurs caractères peuvent être écrits de façon combinée. Ces combinaisons ou ligatures, dont le nombre dépasse 1500, sont optionnelles contrairement aux ligatures horizontales qui sont obligatoires [53]. Les ligatures verticales sont utilisées pour des raisons d'esthétique. Elles dépendent du dessin de la police et du degré de qualité artistique du document. Elles peuvent être formées de deux, trois ou quatre caractères et peuvent prendre plusieurs significations selon l'emplacement des points. On parle souvent de ligature de niveau « n » où n désigne le nombre de caractères ligaturés. Les ligatures verticales, souvent composées de façon particulière, peuvent avoir lieu soit au début ou à

la fin du PAW. La ligature classique de niveau 2 peut avoir lieu avec les couples de caractères donnés dans le tableau 2.9.

Tableau 2.9 : Caractères susceptibles d'être ligaturés verticalement selon [53].

قـ , جـ	فـ , جـ	مـ ,	لـ , مـ
قـ , حـ	فـ , حـ	جـ	لـ , جـ
قـ , خـ	فـ , خـ	مـ ,	لـ , حـ
		حـ	لـ , خـ
		مـ ,	
		خـ	

ل م ج ة : lettres disjointes .

لمجة : ligatures obligatoires .

لمـــجة : ligature esthétique entre les 2 premières lettres.

لمجة : ligature esthétique entre les 3 premières lettres.

Figure 2.4 : Exemples de ligatures horizontales et verticales.

Ce chevauchement modifie les dimensions des PAW et souvent la morphologie de certains caractères (figure 2-4 et 2-5). De plus, la ligne de base dans ce cas, n'est plus horizontale [54]. Ce processus dépend fortement des fontes : il existe des fontes qui ne présentent aucune ligature telles que les fontes « mudir » et simplified arabic », d'autres qui possèdes un ensemble de ligatures complètement différentes les unes des autres [53].

حما ≡ حمـا , لجم ≡ لـجـم .

Figure 2.5: Exemple de formes de PAWs sans et avec caractères ligaturés verticalement (respectivement à droite et à gauche de « ≡ »).

- Contrairement au latin, la notion de majuscule et de minuscule n'existe pas en Arabe. Cependant, tous les attributs de mise en forme tels que gras, italique, souligné sont valables dans les lettres Arabes (figure 2.6).

Figure 2.6 : Le nom de la ville «البليدة», en forme différentes formes.

- Les caractères Arabes peuvent être voyellés. Les voyelles appelées aussi diacritiques dans certains documents tels que [48] et [55] et courtes voyelles dans d'autres tels que [56], peuvent se placer au-dessus ou en dessous du caractère. Les voyelles sont d'une invention postérieure aux consonnes. Dans l'Arabe contemporain ordinaire, on écrit seulement les consonnes et les voyelles longues. Un même mot avec différentes voyelles courtes peut être compris comme verbe, nom ou adjectif …

  A titre d'exemple « علم » peut signifier « drapeau : عَلَمٌ » ou « savoir : عِلْمٌ » ou encore « enseigner : عَلَّمَ», selon sa voyellation.

  Il existe 8 signes de voyellation qui peuvent se placer au-dessus de la ligne d'écriture, tels que fathah (ـَـ) dhammah (ـُـ), soukoun (ـْـ) et chaddah (ـّـ) qui doit être accompagnée de l'une des vayellations fatha, Dammah ou kasrah, en dessous tels que Kasrah (ـِـ). De plus trois « tanwin » peuvent être formés à partir d'un double fatha (ـًـ), d'un double dhammah (ـٌـ) ou d'un double kasrah (ـٍـ).

  Si en français 5 signes orthographiques (les accents grave, aigu et circonflexe, le tréma et la cédille) modifient certaines lettres, en Arabe toutes les formes de consonnes sont susceptible de porter chacune des huit signes de voyellation et souvent deux d'entre eux superposés (par exemple chaddah+voyelle et chaddah+tanwin). Outre cela et comme le montre ce paragraphe les caractères Arabes voyellés nécessitent des matrices de dimensions importantes notamment en hauteur [53].

2.4. Obstacles liées à l'OCR Arabe

La reconnaissance de l'écriture Arabe (Arabic OCR : AOCR) remonte aux années quatre-vingt [57]. Depuis leurs apparitions, plusieurs solutions ont été proposées. Elles sont aussi variées que celles utilisées dans le latin. Dès les premiers travaux de reconnaissance de

l'écriture Arabe (imprimé ou manuscrit), les deux modes de reconnaissance, statistique et dynamique, ont été considérés [57]. Cependant, les travaux en ligne restent relativement peu nombreux [53]. Les caractéristiques morphologiques de l'écriture Arabe, compliquent la tâche de l'OCR à différents niveaux du traitement. Le but de cette section est de cerner les différents problèmes liés à la reconnaissance de cette écriture et de les localiser dans leur phase de traitement. Pour ce, dans ce qui suit, nous présentons une synthèse des principales particularités de l'AOCR suivant les étapes chronologiques d'un système OCR général.

2.4.1 Prétraitement

Le problème conventionnel de cette phase est lié aux boucles qui risquent d'être bouchées ou ouvertes et aux points diacritiques qui peuvent être éliminés à la suite de certaines opérations de prétraitement ou encore confondus avec le bruit. En effet, le prétraitement peut altérer la forme des points diacritiques de manière à les confondre avec du bruit s'ils sont trop amincis. Une autre difficulté peut aussi survenir, c'est le fait que les points risquent d'être accolés au corps du caractère associé à cause d'une dégradation ou d'une normalisation de taille. Un autre problème typique rencontré à la suite d'une mauvaise squelettisation, particulièrement dans le cas du manuscrit, provient de la confusion de deux points diacritiques avec un seul point, très souvent, dans les deux cas, nous obtenons un segment de droite [58]. Pour ces différentes raisons, dans la plupart des travaux, les points sont éliminés au début du traitement et les étapes de ce dernier sont alors effectuées sur le corps du caractère. Par la suite, des traitements ultérieurs sont effectués sur les points diacritiques de manière individuelle [53].

2.4.2 Segmentation

Le problème majeur se ramène à la détection de la ligne base à la segmentation des chaînes de caractères. Les méthodes de segmentation en ligne de texte se basent souvent sur la projection horizontale pour extraire les lignes. Cependant la présence des points diacritiques complique cette extraction et conduit parfois à la fusion des paragraphes. Dans certaines fontes, deux ou trois caractères peuvent se chevaucher verticalement. Très peu de travaux ont tenté à résoudre le problème de la ligature. El Badr et Al [48] considèrent les lieux des ligatures verticales parmi l'ensemble des formes préalablement apprises au système. A l'issue de l'analyse de ces algorithmes de segmentation, nous retenons les points suivants :

• Les liaisons entre les caractères sont variables, elles sont soit trop courtes soit, au contraire, relativement longues, ce qui occulte les marques de séparations ;

• L'existence de caractères ligaturés verticalement complique la tâche de segmentation, souvent, des caractères sont segmentés comme une seule entité (problème de sous segmentation) ;

• Des points de segmentation indésirables peuvent apparaître dans le tracé de l'écriture. En effet, une dégradation du document peut introduire des irrégularités sur le tracé.

2.4.3 Extraction de caractéristiques

La synthèse des travaux considérés montre que les différents types de primitives (structurelles, géométriques, statistiques, transformations globales, corrélations…) et les différentes méthodes de classification (statistiques, structurelles, syntaxiques,…) qui existent dans la littérature, ont été pratiquement toutes utilisées dans la description de l'écriture Arabe. Toutefois, les caractéristiques les plus utilisées dans les systèmes de reconnaissance des mots Arabes sont les caractéristiques perceptuelles [59] [60] [61]. Plus récemment, en reconnaissance de l'écriture cursive, il a été montré que la plupart de l'information discriminante est contenue dans la partie primaire du mot cursif (ascendant, descendant, les boucles). Par ailleurs, les boucles constituent les primitives les plus informatives dans la zone centrale du mot [61]. Pour la reconnaissance des caractères, le calcul des moments, les projections, les concavités et les convexités sont appliqués dans un nombre relativement important de travaux [62], [63]. En effet, la sélection des primitives pour l'Arabe reste une étape complexe à cause volume des corpus des formes à considérer. Le nombre de fontes et styles est important pour l'imprimé. Dans le cas du manuscrit, notamment omni-scripteurs sans contraintes, les formes sont innombrables et difficiles à modéliser. De plus, certaines caractéristiques des lettres, particulièrement les points diacritiques et les boucles, sont sensibles au bruit et à la dégradation. Donc, on peut dire qu'il existe un manque d'étude préliminaire permettant de sélectionner les primitives les plus discriminantes, pour un style et une qualité donnée d'écriture, et les mieux appropriées relativement au type de classifieur retenu.

2.5. Conclusion

Au cours de ce chapitre, nous avons présenté les principales propriétés morphologiques et typographiques de l'écriture Arabe. La reconnaissance optique de la langue Arabe reste une tâche encore non résolue. Les problèmes majeurs dans ce domaine se ramènent à la cursivité de l'écriture et à la sensibilité de certaines caractéristiques topologiques de la langue à la dégradation, en l'occurrence les points diacritiques et les boucles.

Chapitre 3

Implémentation du modèle AOCR proposé

3.1 Introduction

Dans ce chapitre, nous allons présenter le système AOCR que nous avons élaboré.
Notre système englobe les quatre grandes phases suivantes : le prétraitement, la segmentation, l'extraction de caractéristiques et la reconnaissance qui est la phase de classification. Chaque phase comporte un ensemble d'opérations. Notre système est basé essentiellement sur la technique du Template Matching pour la segmentation et basé sur les Modèles de Markov caché pour la mise en œuvre du module de la reconnaissance. Dans ce qui suit, nous allons expliquer de manière détaillée les différentes techniques utilisées pour la réalisation de ce système. Nous allons aussi présenter la base de donnée utilisée pour effectuer le test. L'architecture globale de notre système est illustrée dans la figure3.1.
Le système a été implémenté en utilisant le langage python, dans ce qui suit, nous allons donner une vue générale sur l'environnement du développement.

3.2 Environnement de développement

3.2.1 Machine

Les expériences ont été réalisées à l'aide d'un ordinateur portable, sous Windows 10. L'ordinateur est doté d'un processeur i7 5500U à 2,8 GHz, une RAM de 8 GB. Les algorithmes ont été implémentés en langage Python en utilisant l'environnement de développement *Spyder*.

3.2.2 Langage et bibliothèques

Python est un langage de programmation interprété, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions ; il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et Tcl.Sa collection de librairies est impressionnante. On peut y trouver un module pour un grand nombre de tâches variées (statistiques, Machine Learning, jeux vidéo, mathématiques, etc.) [64]. Son majeur inconvénient c'est qu'il est lent en exécutant le code.
Les bibliothèques suivantes sont les bibliothèques qui nous ont servi pour implémenter nos modules.

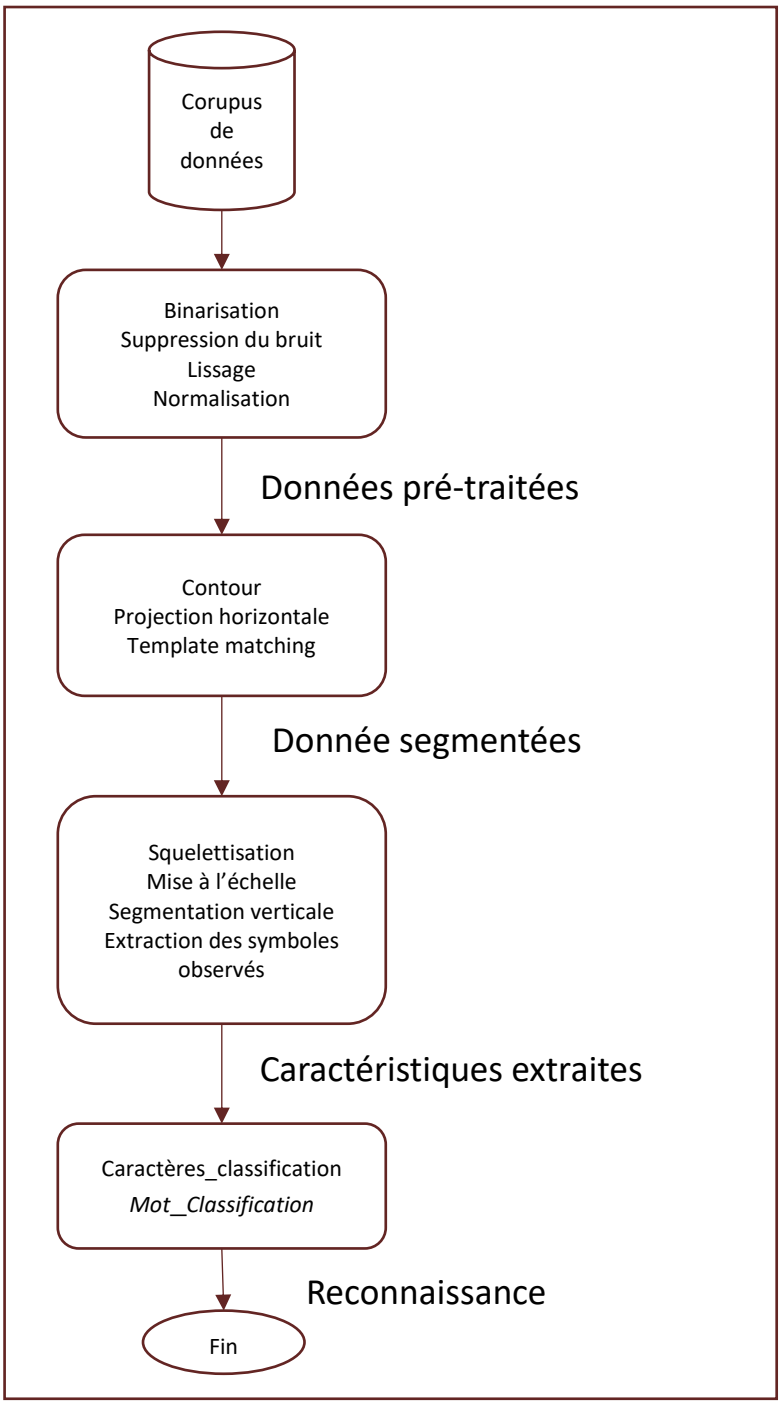
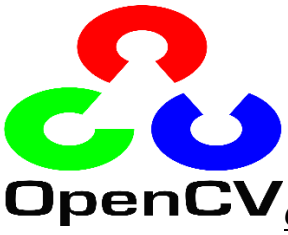


Figure 3.1 : Architecture globale du système.



OpenCv [65][66]: Open Source Computer Vision est une bibliothèque de fonctions de programmation principalement destinées à la vision par ordinateur en temps réel. Développé à l'origine par Intel, il a ensuite été pris en charge par Willow Garage, puis par Itseez (acquis plus tard par Intel). La bibliothèque est multi-plateforme et libre d'utilisation sous la licence open source BSD. OpenCV supporte les frameworks d'apprentissage en profondeur TensorFlow, Torch / PyTorch et Caffe.

NumPy [66]:C'est est une bibliothèque pour le langage de programmation Python, ajoutant la prise en charge de grands tableaux multidimensionnels et matrices, le long avec une grande collection de fonctions mathématiques de haut niveau pour opérer sur ces tableaux. L'ancêtre de NumPy, *Numeric*, a été créé par Jim Hugunin avec les contributions de plusieurs autres développeurs. En 2005, Travis Oliphant a créé NumPy en incorporant des fonctionnalités de Numarray en concurrence dans Numeric, avec d'importantes modifications. NumPy est un logiciel open-source et a de nombreux contributeurs.

Jython2.7 [67] : Jython, est un interprète Python écrit en Java, créé en 1997 par Jim Hugunin. Jython offre les fonctionnalités suivantes

- Compilation de code Python en bytecode Java
- Utilisation d'objets Java dans le code Python.

3.3 Mise en œuvre de l'OCR

3.3.1 Acquisition de données

Cette phase n'a pas été requise dans nos tests, car nos données de tests sont déjà numérisées.

3.3.2 Prétraitement

Comme montré dans la figure 3.1, le prétraitement est le premier module de notre système de reconnaissance. Dans la section 1.5.2**,** nous avons présenté quelques types de prétraitement requis dans les systèmes OCR. Dans notre travail nous n'avons considéré que :

- La binarisation,
- Le lissage,
- La suppression du bruit,
- La normalisation.

L'objectif de ce module est la suppression du bruit pour ne conserver que l'information significative des caractères, d'aiguiser les détails du tracé, de mettre les caractères en tailles standards et réduire tous les types de variations. Ce module a pour but de simplifier le module de l'extraction de caractéristiques.

a. La binarisation

C'est le processus de passer d'une image en couleur, représentée en RGB à une image binaire, c'est-à-dire chaque pixel peut avoir la valeur 0 pour le noir et la valeur 1 pour le blanc. Pour passer d'une image en RGB à une image binaire, il faut passer par une image auxiliaire représentée en niveau de gris.

La fonction qui nous a permis de passer d'une image RGB à une image en niveau de gris est la suivante :

```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

Ensuite, nous avons appliqué la binarisation par méthode de seuillage. Nous avons utilisé un seuillage simple qui consiste à comparer les niveaux de gris de chaque pixel de l'image avec un seuil fixé à 127. La fonction qui nous a permis de faire le seuillage pour passer d'une image en niveau de gris à une image noir et blanc est la suivante :

```
threshold th, threshed = cv2.threshold(gray, 127,
255, cv2.THRESH_BINARY_INV| cv2.THRESH_OTSU )
```

Les images (a), (b), (c) de la figure 3.2représente une image en couleur, une image en niveau de gris et une image noir et blanc respectivement, et ceci en appliquant les fonctions mentionnées ci-dessus.

Figure 3.2 : Passage d'une image en couleur à une image noir et blanc.

Dans notre système, nous avons choisis de faire une binarisation inverse ; c'est-à-dire le font en noir et les objets en blanc.

b. La suppression du bruit

Cette étape permet d'éliminer les pixels qui n'appartiennent à aucun composant significatif de l'image. Pour cela, nous avons appliqué le filtre *MedianBlur*. Le principe de ce filtre est de remplacer chaque entrée par la valeur médiane de son voisinage.

La fonction qui nous a aidé à éliminer les pixels non significatifs est la suivante :

```
gray = cv2.medianBlur(threshed,3)
```

Les images (a) et (b) de la figure 3.3 représente une image binaire bruitée, et une image après l'application du filtre « $MedianBlur$ » pour éliminer le bruit respectivement.

Figure 3.3 : Passage d'une image bruitée à une image sans bruit.

c. Le lissage

Il permet de réduire au maximum les discontinuités introduites dans l'image et ainsi de rétablir la régularité et la continuité du tracé des caractères. Nous avons utilisé une fonction d'*érosion* suivie d'une fonction de *dilatation* pour cela. Cette suite de deux fonctions s'appelle $Ouverture\ morphologique$.

L'*érosion* amincit les bordures de l'objet du premier plan. Le principe est de faire glisser un noyau sur l'image. Les pixels dans l'image originale (soit 1 ou 0) seront considérés « 1 », si tous les pixels sous le noyau sont à « 1 », sinon il est érodé ; c'est-à-dire mis à zéro.

La *dilatation* c'est l'inverse de l'érosion. Ici, un élément de pixel est à «1» si au moins un pixel sous le noyau est à «1». Par conséquent, la taille de l'objet du premier plan est augmentée. La *dilatation* s'applique après l'*érosion*, pour ne pas agrandir le bruit.

Les fonctions que nous avons appliquées pour faire l'*ouverture morphologique* sont les suivantes :

```
kernel = np.ones((2,2),np.uint8)

erosion = cv2.erode(gray,kernel,iterations =3)

dilate = cv2.dilate(gray,kernel,iterations = 2)
```

Les images (a), (b) et (c) de la figure 3.4 représente une image avant l'application de l'ouverture morphologique, après application d'*érosion* et après application de la *dilatation*, respectivement.

(a) Image originale (b) Image après *érosion* (c) Image après *dilatation*

Figure 3.4 : Application de l'ouverture morphologique.

d. Estimation de la ligne de base

Dans cette étape, nous n'avons considéré que l'estimation de la ligne de base. C'est la ligne sur laquelle reposent les lettres qui ne possèdent pas de dépassement bas. Dans un mot écrit en Arabe, les caractères sont connectés entre eux par une ligne de base. La détection de la ligne de base prépare le terrain pour la segmentation des pseudo-mots en caractères. La partie du code qui nous a permis d'estimer la ligne de base est la suivante :

```
#calcule de l'histograme de la projection horizontale
hist = cv2.reduce(threshed,1, cv2.REDUCE_AVG).reshape(-1)
plt.plot(hist)
ma=max(hist) #valeur maximal dans l'histogramme de projection
s=(ma*70)/100#la ligne de base est de 70% de la valeur max
tab=[]
he, we =img.shape[:2] #extraction des dimensions de notre image
ligne_base=[y for y in range(he-1) if hist[y]>=s] #estimation de la ligne de base
print(ligne_base)
#zone mediane
print(ligne_base)
ligne_sup=ligne_base[0]    #la ligne superieure de la ligne de base
print("lignesup",ligne_sup)
ligne_inf=ligne_base[-1]  #la ligne inferieure de la ligne de base
print("ligne_inf",ligne_inf)
cv2.line(img, (0,ligne_sup), (we,ligne_sup), (0,0,0), 1)
cv2.line(img, (0,ligne_inf), (we,ligne_inf), (0,0,0), 1)
*****fin ligne de base*****
```

3.3.3 La segmentation

a. La segmentation en-ligne

Pour faire la segmentation en lignes, c'est-à-dire découper le texte en lignes, nous avons opté pour une stratégie basée sur le contour. Comme cité dans la section 1.4.3.a, la projection horizontale est la plus couramment utilisée pour faire la segmentation en ligne. Cependant, le plus grand défi à relever est le seuil à choisir, tel qu'il existe des cas ou l'espace entre le mot et les points diacritiques peut dépasser le seuil fixé, par conséquent les points diacritiques sont considérés comme des lignes. Ce problème est dit : sur-segmentation. Par exemple si le seuil de segmentation est de 3, les points diacritiques du mot de la figure 3.5 seront considérés comme une ligne.

Figure 3.5 :Sur-segmentation des lignes.

Pour éviter le problème de sur-segmentation, nous avons proposé un noyau de 35 de largeur et 5 de longueur pour faire une dilatation. Le 5 a été fixé pour boucher les cavités entre mot et les points diacritiques. Ça nous permettra d'éviter la sur-segmentation. Le 35 a été fixé pour boucher les trous entre les pseudo-mots. Par conséquent, les pseudo mots de chaque ligne sont soudés et semble être un seul élément. Ensuite, nous appliquons une méthode de contour pour limiter l'élément avec un rectangle.

La portion du code qui nous a permis de faire la segmentation en ligne est la suivante :

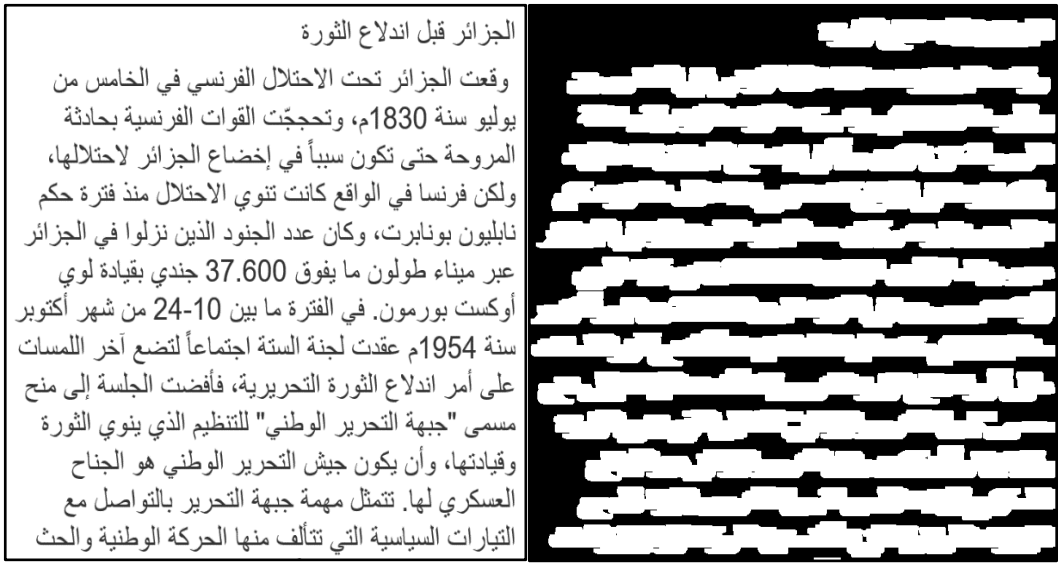
```
#dilation kernel = np.ones((5,35), np.uint8)

img_dilation = cv2.dilate(thresh, kernel, iterations=1)

#find contours ctrs,

hier = cv2.findContours(img_dilation.copy(), cv2.RETR_EXTERNAL,
```

La figure 3.6 représente le résultat d'une dilatation avec un noyau de 5*35 et le texte original.



(a) Texte original (b) texte après dilatation 5*35

Figure 3.6 : Texte en image avant et après l'application de la dilatation.

b. La segmentation en pseudo mot

Pour chaque ligne précédemment extraite, les mots et pseudo-mots séparés par des espaces blancs sont extraits dans cette étape en utilisant la technique de projection verticale. Pour chaque colonne estimée par la projection verticale, le nombre de pixel noirs est stocké dans un tableau de la même largeur que l'image. Le début d'un pseudo-mot est indiqué par la fin d'une succession de zéros. La fin d'un pseudo-mot est indiquée par le début d'une succession de zéros. Après avoir localisé et sauvegardé les débuts et les fins des zones de séparation, nous extrayons l'image sous forme de partie de la matrice de l'image binaire d'origine.

Nous avons utilisé la partie du code suivante pour la segmentation en pseudo mots :

```
## (5) find and draw the upper and lower boundary of each lines
hist = cv2.reduce(threshed,0, cv2.REDUCE_AVG).reshape(-1)
plt.plot(hist)
print(hist)
th = 3
H,W = img.shape[:2]
uppers = [y for y in range(W-1) if hist[y]<=th and hist[y+1]>th]
lowers = [y for y in range(W-1) if hist[y]>th and hist[y+1]<=th]
threshed = cv2.cvtColor(threshed, cv2.COLOR_GRAY2BGR)
print(uppers,"\n",lowers)
for y in uppers:
    cv2.line(threshed, (0,y), (H, y), (255,0,0), 1)
for y in lowers:
    cv2.line(threshed, (0,y), (H, y), (0,255,0), 1)
for i in lowers:
    cv2.line(img,(i+1,0),(i+1,H),(0,0,255),1)
```

La figure3.7 montre un exemple d'une ligne de texte et ses pseudo-mots séparés

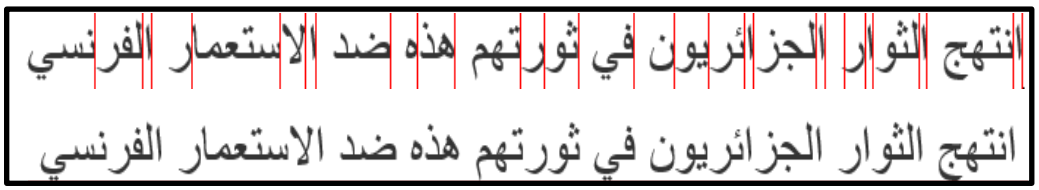


Figure 3.7 : Résultat d'une segmentation en pseudo mots.

c. La segmentation en caractères

Après la segmentation du texte en lignes par une méthode basée sur la combinaison dilatation/définition du contour ; et la segmentation des colonnes en pseudo-mots par la méthode de la projection verticale, l'étape la plus décisive pour reconnaissance peut être réalisé ; c'est la segmentation des pseudo mots en caractères. Cette étape consiste à localiser les points de segmentation possible. Pour cela, nous avons utilisé la technique de *Template Matching.*

Etant donné que la ligne de base est déjà estimée dans la phase de la normalisation, il faut tester le voisinage de chaque point de la rangée au-dessus de la ligne de base, en la balayant de gauche à droite, avec une template. La figure 3.8 représente la template proposée dans [68], c'est une fenêtre 7x7.

X	X	X	Y	0	0	0
1	X	X	Y	0	0	0
X	1	X	0	0	0	0
X	1	1	P	0	0	0
1	1	1	1	1	1	1
X	X	X	X	X	X	X
X	X	X	X	X	X	X

Figure 3.8 : Template proposée dans [68].

P représente le point courant, 0 un point noir et 1 un point blanc, *X* n'est pas important, si les deux Y sont à 1, cela indique la présence de (ـعـ). Si le voisinage de P satisfait cette condition, alors le point P est un point de segmentation. Et le caractère peut être segmenté dans ce point verticalement sur la hauteur de la ligne de base. Le problème avec cette template est qu'elle ne reconnait que le caractère (ـعـ).

Template proposée :

L'utilisation de la Template proposée dans [68] a montré des lacunes dans la segmentation. Nous proposons donc un changement au niveau de la template pour améliorer la segmentation. Après avoir essayé des différentes Template, on a constaté que certaines donnent de meilleurs résultats par rapport à d'autres. Nous avons opté pour la Template présentée dans la figure 3.9, elle donne de meilleurs résultats par rapport à la Template utilisée dans [68] et par rapport aux Template qu'on a testées.

X	X	X	X	X	X
1	X	X	0	0	0
1	1	X	0	0	0
X	1	1	P	0	0
1	1	1	1	1	1

Figure 3.9 : Template proposée.

Problèmes rencontrés :

Problème 1 :

Pour certains caractères Arabes, l'angle de jointure entre le caractère et la ligne de base se trouve au-dessous du caractère lui-même. De plus, il y a des caractères qui ont des boucles sur la ligne de base, comme : ((م) (ص) (ط) (ض) (ظ) (و) (ه)), ces caractères présentent aussi le même problème ; dans quelques cas, le système considère l'intérieur des boucles comme un point de segmentation. La figure 3.10 présente quelques cas du problème cité.

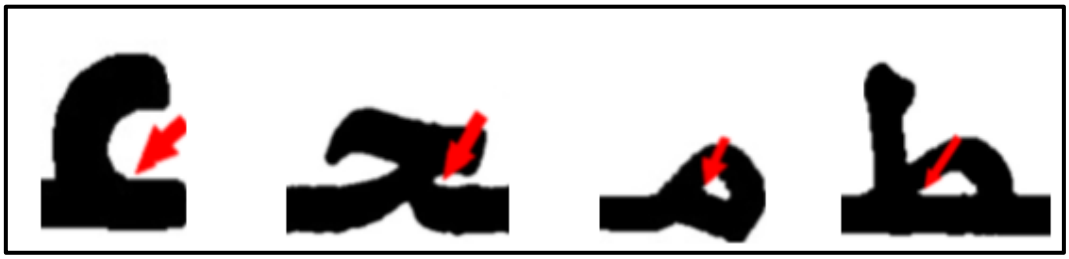


Figure 3.10 : Exemple de cas particuliers.

Solution 1 :

Pour éviter de considérer des points de segmentation qui découpent un caractère en plusieurs parties, nous avons proposé d'examiner les voisins qui se trouvent au-dessus du point trouvé. Si un point noir est trouvé ce point de segmentation sera ignoré.

Problème 2 :

Le système donne plus de 2 segments pour les caractères (ش) (س) (ص) (ض). La figure 3.11 illustre ce problème de sur-segmentation. En effet, ces points satisfassent la condition de la Template, mais ils ne doivent pas être pris en considération, pour éviter de découper un caractère en plusieurs parties.

Figure 3.11 : Exemple de cas particuliers.

Solution 2 :

Pour éviter qu'un caractère soit segmenté en plusieurs segment, nous avons enregistré tous les points de segmentations trouvé, ensuite nous avons calculé la distance entre le début d'un caractère et un point de segmentation trouvé ; si cette distance est inférieure à un seuil prédéfini le point de segmentation sera ignoré.

Problème 3 :

Parfois le tracé des mots n'est pas continu et/ou contient du bruit, cela influe négativement sur le rendu de la ligne de base. Par conséquent, certains caractères ne sont pas segmentés.

Solution 3 :

Pour éviter que les caractères ne soient pas segmentés, nous proposons de chercher les points de segmentation sur la ligne de base et les deux lignes en dessus et les deux lignes en dessous. Dans ce cas, les points de segmentation sont cherchés dans 5 différents niveaux.

3.3.4 L'extraction de caractéristiques

Notre système de classification prend en charge les deux types d'écriture : manuscrite et imprimée.

Pour cela, on a fait l'apprentissage sur des caractères squelettisés. Les caractères squelettisés sont des caractères imprimés sur lesquels on applique la méthode de squelettisation. Dans le cas de l'écriture manuscrite, les caractères ressemblent fortement aux caractères squelettisés.

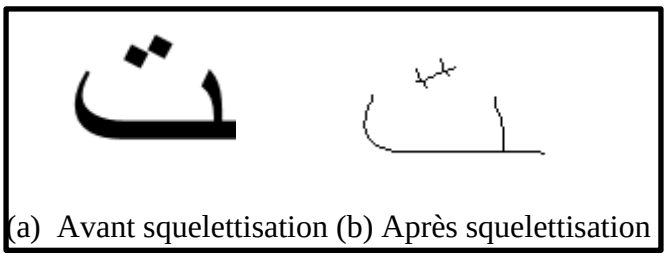
Comme nous avons cité auparavant, nous nous intéressant à l'écriture imprimée. L'entrée de notre système OCR prend des images de texte imprimé. Après la phase de segmentation,

nous obtenons des caractères imprimés. Ces caractères passent par une phase de squelettisation avant la phase de classification.

Nous avons fait la squelettisation à l'aide des lignes de code suivante :

```
# perform skeletonization
skeleton = skeletonize(image)
plt.imshow(skeleton)
plt.imsave('d.png',skeleton)
img = cv2.imread('d.png',0)
ret,thresh2 = cv2.threshold(img,127,255,cv2.THRESH_BINARY_INV)
cv2.imwrite("d.png",thresh2 )
```

La figure 3.12 montre le résultat d'une squelettisation.



(a) Avant squelettisation (b) Après squelettisation

Figure 3.12 : Caractère en image avant et après squelettisation.

Le système suppose qu'il y a une image par caractère, le tracé des caractères est d'une couleur unique (dû à la binarisation) et que le tracé est d'un pixel de largeur (dû à la squelettisation). L'étape de l'extraction de caractéristiques est composée de :

1- La mise à l'échelle : permet de s'assurer que le caractère rempli toute l'image. Nous avons utilisé la méthode standard de mise à l'échelle d'image, Java.

2- La nouvelle image mise à l'échelle est ensuite segmentée verticalement en N segments égaux.

3- Un symbole d'observation est extrait de chaque segment de la manière suivante :

a) Le nombre de pixels dans les trois plus grands composants du segment est calculé. Les tailles des composants sont ensuite mises dans un tuple $(s1, s2, s3)$ qui est trié par ordre décroissant. Un composant est défini comme un ensemble de pixels colorés qui sont connectés entre eux. Deux pixels colorés sont connectés s'ils sont voisins ou s'il est possible de créer un chemin de pixels colorés connectés entre eux. Tous les pixels sauf les pixels de bordure ont 8 voisins. Si le segment contient moins de trois composants, le tuple est rempli de zéros. Ainsi, par exemple, si un segment contient deux lignes. Une ligne contenant 10 pixels et une autre ligne contenant 5 pixels. Le triple résultant sera

(10, 5, 0).

b) Les éléments du tuple sont classés comme Grands, Petits ou Aucun. La fonction de classification c est définie comme dans l'équation 2. Le d constant dans l'équation est donné comme un paramètre à l'extraction des caractéristiques.

$c(s) = None$ if $s = 0$, Large if $s > d$ and Small otherwise. (2)

Le tuple $(s1, s2, s3)$ est traduit en un tuple de classes $(c1, c2, c3)$ en appliquant la fonction c à tous les éléments du tuple.

c) Le tuple des classes est cartographié selon un symbole d'observation. Au total, il y a 10 tuples différents et il y a un symbole d'observation par tuple. Donc, au total, il y a 10 symboles d'observation différents.

La classification constante d et le nombre de segments N sont des paramètres de l'extracteur de caractéristiques. Un exemple d'extraction des caractéristiques pour une image peut être vu dans la figure 3.13.

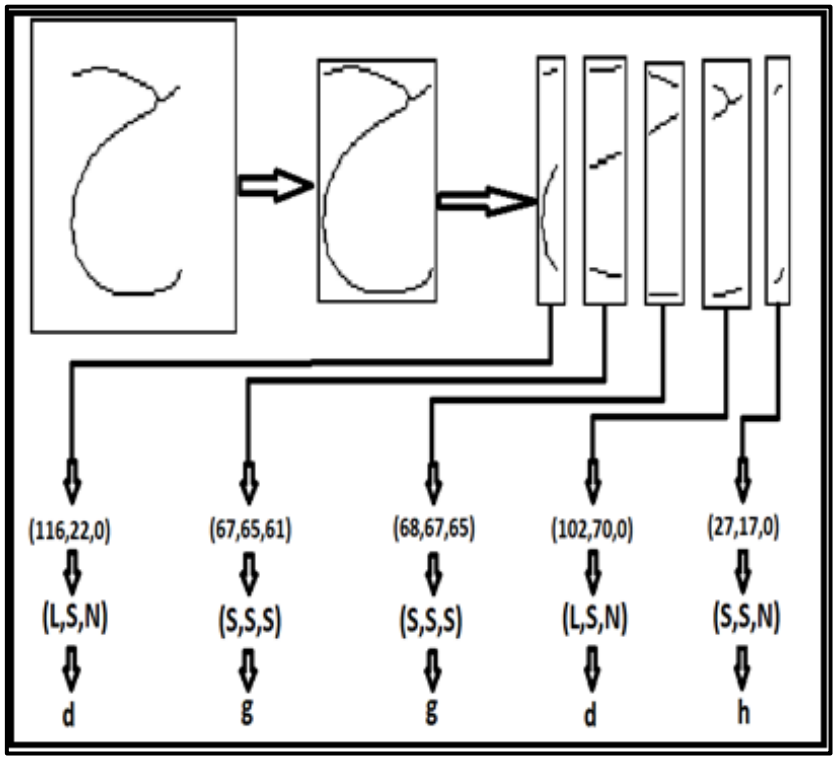


Figure 3.13 : Exemple d'extraction de caractéristiques pour le caractère « ڎ ».

3.3.5 La classification

a. La classification de caractères

Après l'extraction des caractéristiques vient l'étape de la reconnaissance des caractères, c'est l'étape de la classification. Pour la réalisation de ce module nous avons mis en œuvre un classifieur en utilisant des Modèles de Markov Cachés (Hidden Markovian Model HMM). Nous avons nommé ce classifieur $Caractère\_classification$.

Cette section décrit le module de reconnaissance développé. Nous commençons par donner une vue d'ensemble de haut niveau, puis nous décrivons les détails tels que la mise en œuvre HMM.

$Caractère\_classification$ contient plusieurs $HMMs$, tel que chaque HMM est propre à un élément de l'ensemble de sortie, par conséquent notre classifieur contient 126 $HMMs$. Au moment de l'apprentissage du classifieur, il reçoit des exemples d'entrée pour tous les éléments possibles de la sortie. Dans ce qui suit nous allons décrire la topologie des $HMMs$ mis en œuvre.

Topologie des $HMMs$ du $Caractère\_classification$:

Un modèle est mis en œuvre pour chaque élément de sortie. La topologie proposée est similaire à celle proposée par Laan et al [69].

Comme mentionné dans la section 1.4.5.b le HMM contient 5 caractéristiques, à savoir N le nombre d'états dans le modèle, M le nombre de symboles d'observation, A la distribution de probabilité de transition, B la distribution de probabilité de symbole d'observation pour chaque état et finalement π la distribution de d'état initiale.

Nous avons utilisé des états de début et de fin spéciaux. Comme mentionné dans la section 3.3.4 l'étape d'extraction des caractéristiques produit N segments qui sont considérés comme des états, ce qui donne en tout avec les états début et fin $N + 2$ segments. Donc les états début et fin sont concaténé avec les états propres à un caractère pour former une séquence d'observation. Nous avons ensuite utilisé l'algorithme $Baum - Welch$ [70] pour le calcul des probabilités de transition et d'émission. En effet, cet algorithme est utilisé pour faire l'apprentissage du classifieur ;étant donné un ensemble de séquence $O = \{o_1, o_2 \dots, o_m\}$, dont la séquence courante est o_k ; le but de l'apprentissage est de déterminer les paramètres du model de Markov caché qui maximisent la probabilité $P(O/\Lambda)$, à savoir A, B et π.

L'idée est d'utiliser une procédure de réestimation selon les étapes suivantes :

• *initialiser les paramètres de l'HMM de départ* Λ_0 *;*

• *calculer Λ_1 à partir de Λ_0, puis Λ_2 à partir de Λ_1, etc.*

• *répéter ce processus jusqu'à convergence.*

Pour chaque étape p d'apprentissage, on dispose de Λ_p et on cherche un Λ_{p+1} qui doit vérifier:

$$P(O/\Lambda_{p+1}) \geq P(O/\Lambda_p)$$

Autrement, Λ_{p+1} doit améliorer la probabilité de l'émission des observations de l'ensemble d'apprentissage.

L'initialisation des matrices de transition et d'émission est faite suivant les propriétés ci-dessous :

- L'état *début* émettra toujours le symbole spécial @ et l'état *fin* émettra toujours le symbole spécial $.
- *L'état début transis toujours vers le premier état normal (le 1er segment).*
- *L'état fin transis toujours vers l'état début.*
- *Tous les autres États passent toujours à un État qui n'a pas été visité depuis la dernière visite.*

La figure 3.14 présente la topologie du HMM mis en œuvre.

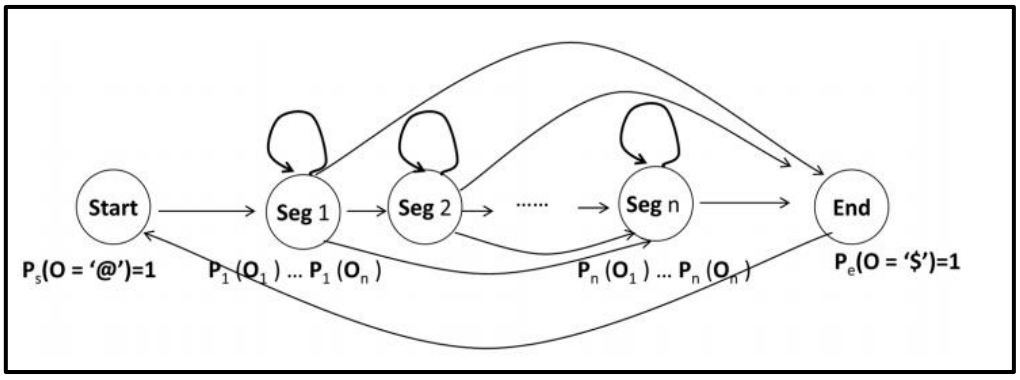


Figure 3.14 : Topologie d'un HMM propre à un caractère.

Les matrices de transition A et émission B sont calculées suivant l'algorithme $Baum - Welch$ comme déjà mentionné. La matrice de transition A est une matrice $N * N$, elle enregistre les probabilités de passer d'un état à l'autre. La matrice d'émission B est une matrice $N * M$, elle enregistre les probabilités pour chaque état d'émettre chacune des observations possibles.

Ces deux matrices sont des matrices stochastiques. C'est à dire chaque élément d'une ligne est une probabilité et que la somme des éléments de la ligne est égale à 1.

Formule de réestimation :

Comme proposé par $Baum-Welch$ [70], étant donnée une observation o_k, le model Λ_p, i et j, tel que il faudra calculer la probabilité $P_r$que l'état $s_i$ait émis la $t^{ème}$ lettre de l'observation o_k et que l'état s_j ait émis la $t+1^{ème}$ lettre de l'observation o_k.

$$P_{r_t}^{\ k}(i,j) = P(q_t = s_i, q_{t+1} = s_j | o_k, \Lambda_p)$$

Ce qui donne :

$$P_{r_t}^{\ k}(i,j) = P(q_t = s_i, q_{t+1} = s_j, o_k |\ \Lambda_p)/P(o_k|\Lambda_p)$$

Suivant l'algorithme $forward$ [45], nous en déduisons que :

$$P_{r_t}^{\ k}(i,j) = \alpha_t^k(i) a_{i,j} b_j(o_{t+1}^k)/P(o_k|\Lambda_p)$$

On définit$\gamma_t^k(i)$ comme la probabilité que la lettre de rang t de la phrase o^ksoit émise par l'état s_j :

$$\gamma_t^k(i) = \sum_{j=1}^{n} P_{r_t}(i,j) \frac{\alpha_t^k(i)}{P(o^k|\wedge)}$$

À noter que les formules de ré estimation sont tiré du tutoriel [45].
En résumé, comme l'a déclaré Rabiner dans [45] le nouveau modèle HMM se calcule à partir de l'ancien en faisant une nouvelle estimation deπ, A et Bpar comptage sur la base d'apprentissage :

$$a_{i,j} = \frac{nombre\ attendu\ de\ la\ transition\ de\ s_i \text{à}\ s_j}{nombre\ attendu\ des\ transitions \text{à}\ partir\ de\ s_i}$$
$$= \frac{\sum_{t=1}^{T-1} P_{r_t}(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$b_j(k) = \frac{nombre\ attendu\ de\ fois\ dans\ l'état\ s_j\ et\ en\ observant\ le\ symbole\ q_k}{nombre\ attendu\ de\ fois\ dans\ l'état\ s_j}$$
$$= \frac{\sum_{t=1}^{T} \gamma_t(j)\ s.t\ o_t = v_k}{\sum_{t=1}^{T} \gamma_t(i)}$$

$$\pi_i = nombre\ de\ fois\ dans\ l'état\ s_i\ en\ emettant\ le\ premier\ symbole$$

Comme déjà mentionné, l'algorithme $Baum-Welch$ se termine lorsque le HMM résultant d'une itération n'a pas changé par rapport au précédent.

Initialisation du model :

Peu importe l'algorithme d'apprentissage, un modèle initial doit être établie pour entamer l'apprentissage. Dans le cas des $HMMs$, il y a plusieurs type d'initialisation, à savoir : initialisation basée sur le comptage, initialisation aléatoire et l'initialisation uniforme. Dans notre travail, après avoir testés les 3 types d'initialisation, nous avons opté pour l'initialisation basée sur le comptage. Son principe est le suivant :

La matrice de transition est configurée de manière à ce que, pour chaque état normal, N_i, la probabilité de transition à l'état N_{i+1} soit fixée à 0,8. Les 0,2 restantes sont divisées de manière égale entre les transitions vers tous les autres états d'indice $\geq i$.

b. La classification des mots

Après la classification des caractères vient l'étape de la génération des mots. Nous avons mis pour ça en œuvre 2 classifieurs qui prennent une chaîne de caractères en entrée et génère un mot.

Le premier classifieur suit la même démarche que le classifieur de caractère et modélise un HMM pour chaque mot de l'ensemble d'entrée. Cela est décrit dans la section précédente. Si le mot à n lettres, il y aura $n + 2$ états dans le HMM correspondant. Il est raisonnable et plus pertinent d'implémenter un HMM propre à chaque mot de l'ensemble, mais ceci lorsque le vocabulaire est petit. Cette stratégie perd en performance lorsque la taille de l'ensemble grandit. Dans ce cas l'utilisation d'un seul HMM est une bonne solution.Pour cela, nous avons utilisé un deuxième classifieur $Viterbi$ [71] [72] et on a mis en œuvre un seul HMM pour tout l'ensemble du vocabulaire (l'ensemble des mots). Dans ce qui suit, nous allons décrire la topologie de l'HMM mis en œuvre pour le classifieur $Viterbi$.

Topologie de l'$HMMs$ du $Viterbi$:

Contrairement aux $HMMs$ utilisés dans le classifieur de caractères, l'HMM mis en œuvre pour le classifieur des mots a des états qui ont des transitions vers tous les autres états. Il a 128 états cachés (126 pour les 126 caractères de la langue Arabe et 2 avec des émissions spéciales @ et $).La matrice de transition est une matrice de 128 à 128, qui est estimée à partir de l'analyse du lexique, en utilisant la méthode expliquée ci-dessous.

Par exemple, s'il n'y a que trois mots dans notre vocabulaire :{تمر،توت،نمل}. Dans ce cas, la probabilité de passer de م à ل est de 0,5 ; la probabilité de passer de م à ر est de 0,5 aussi et la probabilité de passer de م aux autres caractères est de 0.

D’après la stratégie d’estimation des paramètres que nous avons déjà expliquée dans 3.3.5.a, la matrice de probabilité de transition a besoin que des informations de de lettres successives dans les mots.

Evidemment, les observations sont les lettres que nous avons observées à partir du classifieur de caractères. La matrice de probabilité d'observation est créée à partir des observations juste après le test du classifieur de caractères.

Par exemple, si nous avons 20 exemples d’images de test pour أ, et que 15 d’entre elles sont classées comme أ, 3 pour لet 2 pour ا, alors $P\ (observation = أ) = 0{,}75, P\ (observation = ل) = 0{,}15\ etP\ (observation = ا) = 0{,}1.$

Pour cet exemple, la ligne pour أ dans la matrice de probabilité sera définie sur [0.75, …,0,..0,0.15, 0, 0, 0.1, ..., 0], tel que : 0.75, 0.15, 0.1 sont écrit dans les emplacements de أ، ا، ل respectivement.

Donc, étant donné la topologie décrite ci-dessus, la classification d’une chaîne de caractères peut être effectuée de la manière suivante :

1. *Une chaîne de caractère est vu comme une séquence d'observation.*
2. *L'algorithme Viterbi est appliqué pour obtenir la séquence d'états la plus probable.*
3. *La similitude de la chaîne résultante à tous les mots de sortie possibles est calculée.*
4. *Le mot le plus similaire est retourné.*

Nous avons utilisé la distance de Hamming[73] pour calculer la similitude mentionnée dans 3.

Sur la base de notre topologie, les connaissances utiles extraites de l'étape de classification des caractères sont conservées. Nous avons nommé ce classifieur $Mot\_Classification$.

3.4 Conclusion

Dans ce chapitre nous avons présenté le modèle général de l’AOCR implémenté. Nous avons ensuite présenté les différents modules du modèle, à savoir : le prétraitement, la segmentation, l’extraction des caractéristiques et la classification.

Nous avons détaillé le module de la segmentation, en expliquant les différents types de segmentation requise, à savoir la segmentation en ligne, la segmentation en pseudo mots et la

segmentation en caractères. Nous avons aussi détaillé le module de la classification, en expliquant les deux niveaux de reconnaissance, c'est-à-dire la reconnaissance des caractères et la reconnaissance des mots ; chaque étape de reconnaissance a requis un certain model HMM que nous avons expliqué.

Pour chaque module développé, nous avons mis en avant les problèmes rencontrés –liés en général à la nature cursive de la langue Arabe- et nous avons proposé des solutions que nous avons expliquées.

Le chapitre suivant sera consacré à la présentation des résultats d'implémentation du modèle en générale et le résultat d'implémentation de quelques modules.

Chapitre 4

Résultats d'implémentation

4.1 Introduction

Dans notre travail, nous avons proposé et implémenté des méthodes pour différents modules d'un système AOCR. Le premier module visé est la segmentation, il comprend 3 niveaux, à savoir la segmentation en ligne, la segmentation en pseudo-mots et la segmentation en caractères. Le deuxième module visé est l'extraction des caractéristiques qui passe par plusieurs étapes, à savoir la squelettisation, la mise à l'échelle, la segmentation verticale et l'extraction des symboles observés. Le dernier module visé est la reconnaissance, il comprend deux niveaux de classification, à savoir la classification des caractères et la classification des mots. Le module de reconnaissance se base sur les modèles de Markov cachés. Ce chapitre est consacré à la présentation des résultats du module de segmentation et le module de la reconnaissance.

4.2 Corpus de données

4.2.1 Introduction

Après la lecture des travaux existant dans la littérature, nous avons constaté que les auteurs évaluent leurs méthodes en les testant sur leurs propres corpus de données générés de façon aléatoire (numérisation des textes choisis). Cette démarche ne permet pas de comparer leurs méthodes à d'autres méthodes, ce qui n'implique pas une bonne évaluation. Il est donc important de créer un corpus de donnée standardisé. Ce qui donnera plus de fiabilité aux résultats des tests.

4.2.2 Problèmes liés à la segmentation

Parmi les problèmes les plus courant de la segmentation :

- Le chevauchement des caractères, ce problème est lié directement à la fonte de l'écriture. Il y a des fontes qui font que deux caractères partagent les mêmes positions horizontales. Le bruit peut être aussi un facteur majeur dans ce genre de problème.
- Extraction des contours fermés cela est liée à un faux prétraitement (discontinuité des contours par exemple).
- Extraction du squelette liée à un faux prétraitement (discontinuité des contours par exemple).
- Détermination de la taille du stylo.
- Détermination de la ligne de base.

- Sur-segmentation de quelques caractères ayant plusieurs pics (par exemple : {ن،ث،ت،ب،ض،ص،ش،س}).
- Petit espacement entre les caractères du même mot, ce qui biaise la segmentation en caractères.

4.2.3 Construction de notre corpus de données

Pour que notre méthode puisse être comparée à d'autres méthodes (anciennes ou futures), nous allons citer les caractéristiques du corpus utilisé. Nous avons fait en sorte que notre corpus :

- Contiennent toutes les lettres de l'alphabet Arabe, dans toutes les positions possibles (début, milieu, fin et isolée).
- Différentes tailles de police.
- Une qualité d'image moyenne (images bruitées).
- Au moins 3000 caractères.

Pour évaluer notre système, nous avons construit une base de données à partir d'un document de texte arabe imprimé. La base de données se compose de **67 lignes**, **625 mots** arabes, contenant au total **3115 caractères** écrits avec les 5 **fontes** suivants :

- Time new Roman, Calibri, Andalus, AcsAlmas Bold, Arial …

La figure 4.1 montres des extraits du texte utilisé. Le texte complet peut être consulté dans l'annexe A.

4.2.4 Conclusion

La disponibilité d'une base de données référentielle est cruciale pour une évaluation objective des différents systèmes implémentés. Dans cette section, nous avons donné une vue sur le corpus de données qu'on a proposé et utilisé pour le test, afin que notre démarche puisse être comparée. Dans la section suivante, nous allons présenter nos résultats, en se basant sur le corpus mentionné dans l'annexe A.

4.3 Résultat de la segmentation

Dans cette section, nous allons présenter les résultats de la segmentation. Le texte a été numérisé, binarisé, traité du bruit et est ensuite alimenté au module de la segmentation. La figure 4.2 montre un extrait de notre texte, avec différentes fontes.


حملت الجمهورية الجزائرية لقب بلد المليون شهيد ونصف المليون
شهيد بعد أن روى أبناؤها ثرى الوطن بدمائهم الطاهرة، وبعد أن
ثار الجزائريون ضد الاستعمار الفرنسي في الأول من
وأن يكون جيش التحرير الوطني هو الجناح العسكري لها. تتمثل مهمة
جبهة التحرير بالتواصل مع التيارات السياسية التي تتألف منها الحركة
الوطنية والحث على الانضمام لصفوف الثُوّار، بالإضافة إلى تشجيع
الجزائرية فيما يتعلق بالاتصالات والأسلحة إلا أن الثورة قد تكللت
بالنجاح. نتائج الثورة الجزائرية قُرعت طبول وقف الحرب الجزائرية
الفرنسية، والتي تكللت بانتصار جبهة التحرير
هذا اليوم ذاته الذي وقعت فيه الجزائر قيد الاحتلال الفرنسي.
وجه الجنرال شارل ديغول عبر شاشة التلفزيون خطاباً للشعب

Figure 4.1 : Texte extrait de notre corpus.

4.3.1 Segmentation en ligne

Le texte de la figure 4.2 a subi une segmentation en ligne, par la méthode du contour. Les résultats sont des lignes stockées en images chacune. Les résultats sont montrés dans la figure 4.3.

Figure 4.2 : Résultat de segmentation en ligne.

Le taux de réussite de cette étape est de **100%**.

4.3.2 Segmentation en pseudo-mots

Dans cette étape, chaque ligne (image) résultante de la segmentation en ligne a été alimenté à la fonction de la segmentation en pseudo-mots qui est basé sur la projection horizontale. Les résultats (de la segmentation de quelques parties du texte) sont montrés dans la figure 4.3.

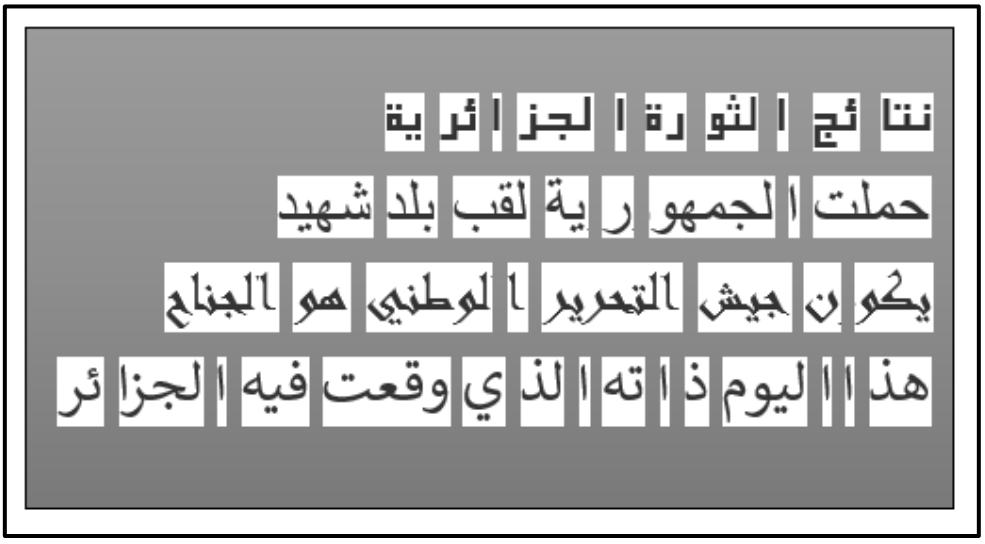


Figure 4.3 : Résultats de la segmentation en pseudo-mots.

Le taux de réussite de cette étape est de **100%**.

4.3.3 Segmentation en caractères

Dans cette étape, chaque pseudo-mot résultant de l'étape précédente est alimenté à la fonction chargée de la segmentation en caractères qui est basée sur la technique du template matching. Les résultats (de la segmentation de quelques parties du texte) sont montrés dans la figure 4.4.

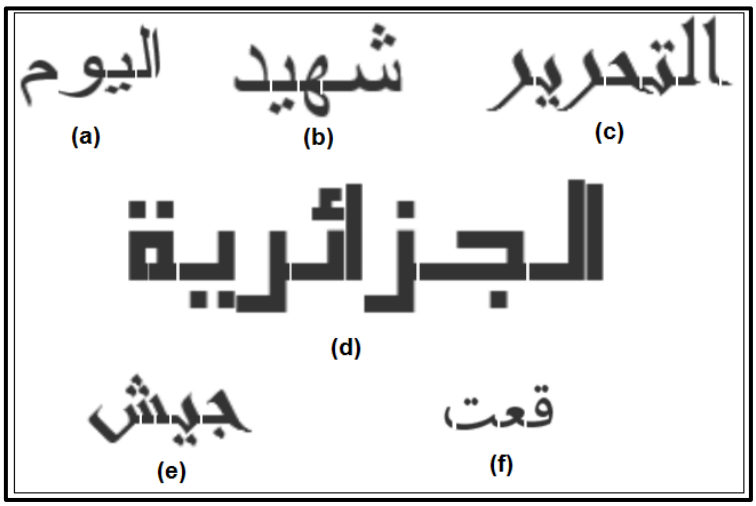


Figure 4.4 : Résultats de la segmentation en caractères.

La figure montre les résultats de la segmentation de 6 mots en caractères. Les mots (a), (b) et (c) sont segmenté en caractères correctement. Les 8 premiers caractères du mot (d) ont été segmenté correctement, par contre la partie inferieur du dernier caractère a été segmenté en deux, à cause d'une mauvaise estimation de la ligne de base. Cela n'influe pas négativement sur la reconnaissance du caractère. Les derniers caractères des mots (e) et (f) ont été sur-segmentés, parce qu'ils contiennent plusieurs pics.

Le taux de réussite de cette étape est de…

4.4 Résultat de la reconnaissance

Dans cette section nous présentons les résultats de la reconnaissance décrite dans la section

4.4.1 Résultats de $Caractère\_Classification$

Dans cette section, nous présentons les résultats de l'implémentation de la classification des caractères décrite dans la section 3.3.5.a. Dans la section 3.3.4, nous avons mentionné que l'extraction de caractéristiques produit N segments. Le nombre de segments à extraire N est introduit en paramètre par l'utilisateur, ainsi qu'un facteur de classification des segments. Pour l'expérimentation, nous avion 100 instance pour chacun des 126 caractères Arabe.

Comme mentionné dans la section 3.3.5.a, l'initialisation basée sur le comptage a montré de meilleur résultats par rapport à l'initialisation aléatoire et uniforme et ceci en ignorant la phase de l'apprentissage. Donc seule l'initialisation basée sur le comptage est prise en compte dans cette série de tests.

Nous avons effectué des tests avec une variété de valeurs pour les paramètres. Nous avons créé en tout (9×18) modèles, tel que le nombre de segments varie entre 4 et 12à raison de 1(ce qui donne 9) et le facteur de classification varie entre 0.7 et 5.8 à raison de 0.3 (ce qui donne 18).

Pour chaque caractère, nous avons utilisé 90 instances pour l'apprentissage et 10 instances pour le test. Nous avons considéré la précision de la classification pour tous les modèles. Les résultats sont montrés dans la figure 4.5.

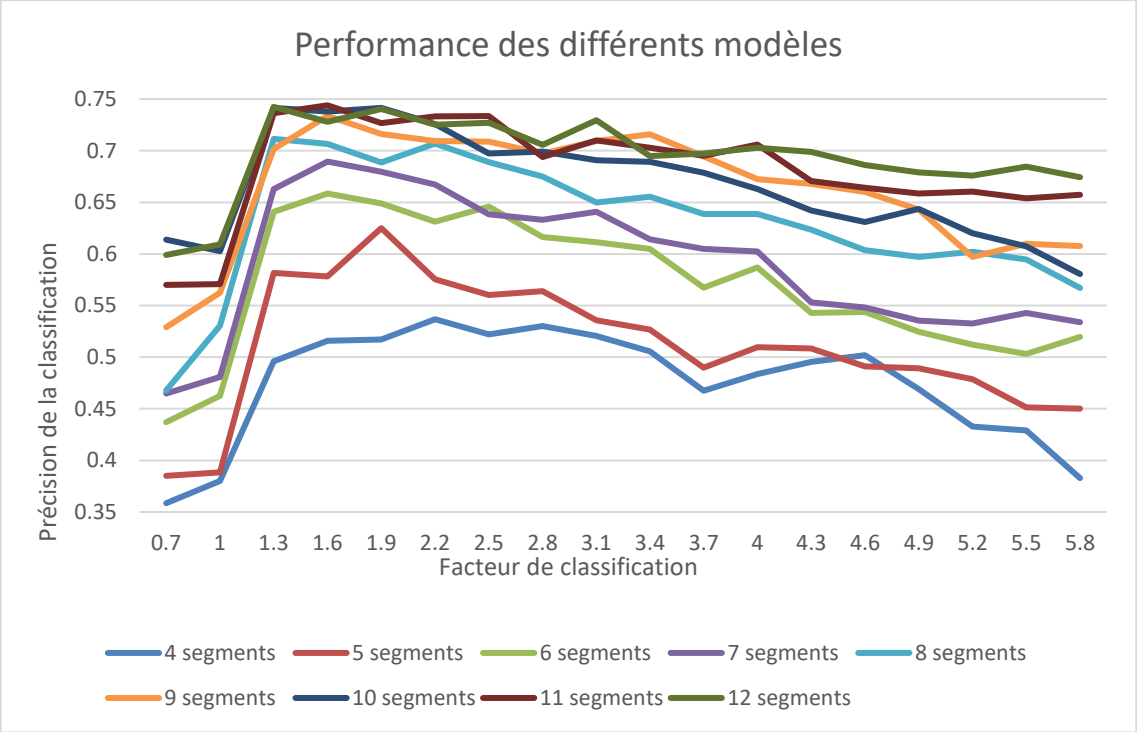


Figure 4.5 : Précision de la classification en fonction du nombre de segment et facteur de classification.

D'après l'analyse des résultats, nous constatons que l'augmentation du nombre des segments à produire lors de l'extraction des caractéristiques améliore les performances de la classification. Par contre ce n'est pas le cas pour le facteur de classification. Le meilleur score a été enregistré pour : (nombre de segment = 11 et facteur de classification=1.6)

4.4.2 Résultat de *Mots_Classification*

Dans cette section, nous présentons les résultats obtenus pour la phase *mots_classifieur*, en utilisant forward_classifieur expliqué dans 3.3.5.b. Les résultats considérés sont ceux de la classification des mots suivants :

["أسامة ","وليد ","حمامة ","بليد","قطة ", " بطة ","سعادة ","تعاسة ","كراهية ","رفاهية",
"حرب","حب","إمعة","قبعة","أسد" "إستقصاء", "إستدعاء","عهد","فهد","تخلف","تحضر""]

Les mots utilisés dans l'apprentissage et le test ont été généré aléatoirement, en s'appuyant sur [74]. Au total, nous avons utilisé 100 instance de test (5 pour chacun des 20 mots cités ci-dessus). Le modèle a été initialisé par la méthode basé sur le comptage. Le test a été effectué avec 100, 200, 400, 800,1600, 3200, 6400 et 12800 instances d'apprentissage. Nous avons considéré la précision de la classification pour tous les modèles avant et après apprentissage avec Baum-Welch. Les résultats sont montrés dans la figure 4.6.

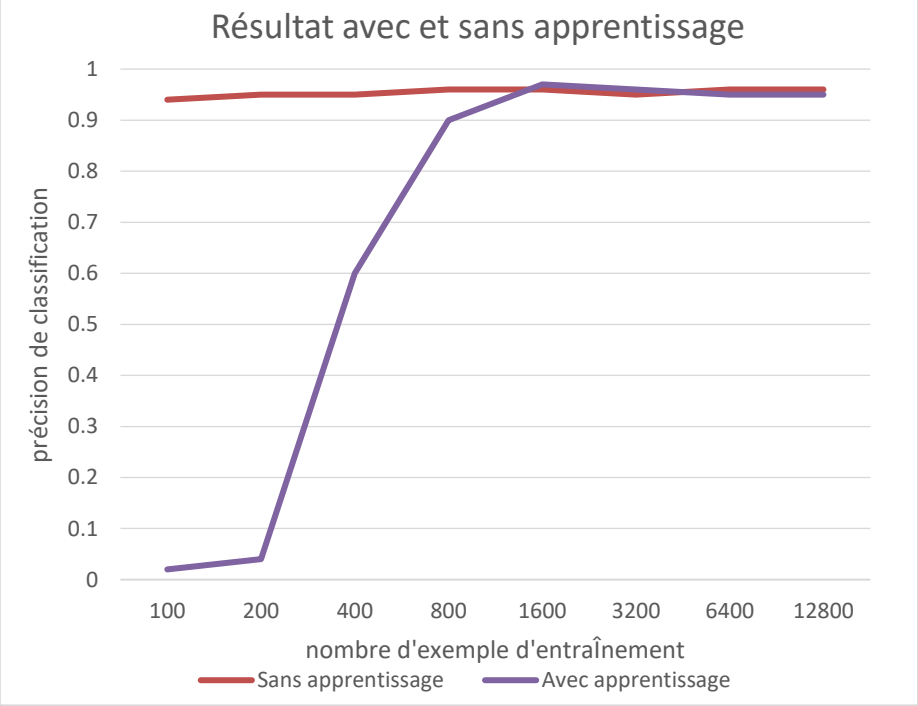


Figure 4.6 : Précision de la classification en fonction du nombre d'instance d'apprentissage.

Les résultats montrés dans la figure **4.6** montrent clairement l'importance de disposer de suffisamment de données d'apprentissage.

4.5 Conclusion

Ce chapitre a été consacré à la présentation des résultats d’implémentation, en termes de précision de classification.

Conclusion générale

Conclusion générale

Le domaine de la reconnaissance de la langue Arabe est très vaste et riche en connaissance, que ce soit sur la langue elle-même ou ses différentes tournures et ses ambiguïtés. La reconnaissance du texte Arabe est une tâche compliquée à cause de la cursivité des caractères. D'un point de vue industriel c'est une tâche très importante, elle peut être utilisée dans plusieurs applications telles que la lecture d'adresses postales et la reconnaissance de montants littéraux. Malgré les efforts et le progrès important fournis par la communauté des chercheurs, aucun système actuel n'est jugé fiable à 100%. Atteindre un taux de reconnaissance comparable aux performances humaine semble irréalisable. C'est pour cette raison, le terrain est encore fertile pour de futurs travaux de recherche.

Le présent travail s'inscrit dans le cadre général de la reconnaissance automatique de l'écriture Arabe. Plus précisément, il exploite une approche basée sur les modèles de Markov cachés appliqués à la reconnaissance de texte Arabe imprimé et non voyellé.

Un système de reconnaissance passe par plusieurs étapes, à savoir l'acquisition des données, le prétraitement, la segmentation, l'extraction des caractéristiques et la reconnaissance. Au cours de notre travail, nous nous sommes intéressés aussi à une étape très importante qui influe directement sur l'étape de la reconnaissance, c'est l'étape de la segmentation.

La segmentation est composée de trois phases, à savoir la segmentation e ligne, la segmentation en pseudo mots et la segmentation en caractères. Les problèmes rencontrés durant la segmentation sont liés à la qualité du document fournis et plus particulièrement à la nature cursive de la langue Arabe. Pour la segmentation en ligne, dans notre travail nous avons développé une méthode basée sur le contour, en utilisant la dilatation ; et nous avons obtenu un taux de segmentation de 100%. Pour la segmentation en pseudo mots, nous avons utilisé la technique de la projection verticale et nous avons obtenu un taux de segmentation de 100%. Au cours de la segmentation en caractères, nous avons utilisé la technique du Template Matchingen proposant notre propre Template, nous avons rencontré des problèmes de non-segmentation et de sur segmentation que nous avons réglé à notre façon, nous avons obtenu un taux de segmentation de 90%

En ce qui concerne la reconnaissance, elle passe par deux étapes, à savoir la classification des caractères et la classification des mots. A noter que l'étape de l'extraction des caractéristiques qui précède la reconnaissance est importante aussi, donc l'utilisation des méthodes puissante pour extraire les caractéristiques les plus significative est incontournable.

Dans l'étape de la classification des caractères, nous avons utilisé un classifieur basé sur les modèles de Markov Caché. Nous avons considéré un modèle propre à chaque caractère. Nous avons fait l'apprentissage du modèle par l'algorithme Baum-Welch. L'étape de l'apprentissage a diminué le taux de classification, en raison du nombre réduit des données d'apprentissage. Le taux de classification avant l'apprentissage est de 55% et après l'apprentissage est de 75%

Dans l'étape de la classification des mots, nous avons considéré deux différents classifieur. Le premier considère un modèle propre pour chaque mot et a la même topologie que le classifieur des caractères. Le deuxième classifieur considère un seul modèle pour tous les mots du corpus. Nous avons essayé d'augmenter le nombre de donnée d'apprentissage en utilisant un générateur de mots à partir de tous les mots existant ce qui aide à la correction des mots mal classifiés. Le taux de classification de mots est de: 90%

Malgré la réalisation d'un système de reconnaissance complet et efficace, plusieurs améliorations et perspectives sont envisageables. La première perspective envisagée est d'essayer d'autre méthode d'extraction de caractéristiques plus significatives. La prise en compte de toutes les fontes existantes serait intéressante pour généraliser notre système. Une autre piste importante est la prise en compte des textes voyellés. Finalement, l'augmentation des données d'apprentissage nous permettrait de tester les vraies performances du classifieur des caractères.

Bibliographie

[1] Ahmed Lawgali, "A survey on arabic character recognition," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 8, no. 2, pp. 401-426, 2015.

[2] KAUR AMANDEEP, BAGHLA SEEMA, and KUMAR SUNIL, "STUDY OF VARIOUS CHARACTER SEGMENTATION TECHNIQUES FOR HANDWRITTEN OFF-LINE CURSIVE WORDS: A REVIEW," *International Journal of Advances in Science Engineering and Technology, ISSN: 2321-9009*, vol. 3, no. 3, July 2015.

[3] Amjad Rehman, Dzulkifli Mohamad, and Ghazali Sulong, "Implicit vs explicit based script segmentation and recognition: A performance comparison on benchmark database," *International Journal Open Problems Compt. Math.,* vol. 2, no. 3, pp. 352--364, 2009.

[4] Pierre Michel Lallican, Christian Viard-Gaudin, and Stefan Knerr, *From off-line to on-line handwriting recognition*.: International Unipen Foundation, 2000.

[5] I.R. Tsang, *Pattern recognition and complex systems*, université d'Anterwerpen Thèse de doctorat, Ed., 2000.

[6] T. Kameswara Rao, K. Yashwanth Chowdary, I. Koushik Chowdary, K. Prasanna Kumar, and Ch. Ramesh, "Optical Character Recognition from Printed Text Images," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 5, no. 2, pp. 597-604, 2019.

[7] AMIN ADNAN, "OFF-LINE ARABIC CHARACTER RECOGNITION: THE STATE OF THE ART.," *Pattern Recognition*, vol. 31, no. 5, pp. 517-530, 1998.

[8] Adnan Amin, Humoud Al-Sadoun, and Stephen Fischer, "Handprinted A rabic character recognition system using an artifficial network," *Pattern recognition*, vol. 29, no. 4, pp. 663-675, 1997.

[9] Narasimha Reddy Soora, "A Simple Character Segmentation Technique for off-line Cursive Hand Written Documents," *International Journal of Control Theory and Applications*, vol. 10, pp. 71-77, April 2017.

[10] Al-Badr Badr and A. Mahmoud Sabri, "Survey and bibliography of Arabic optical text recognition," *Signal Processing*, vol. 41, pp. 49-77, 1995.

[11] Jonathan Sachs, "Digital Imaging Glossary," 2002.

[12] Andrew King and Paul Aljabar, *MATLAB Programming for Biomedical Engineers and Scientists*.: Academic Press, 2017.

[13] Amit and Rishi, Rahul Choudhary and Savita Ahlawat, "A new character segmentation approach for off-line cursive handwritten words," *Procedia Computer Science*, vol. 17, pp. 88--95, 2013.

[14] Mohammad S Khorsheed, "Off-Line Arabic Character Recognition – A Review," *Pattern analysis & applications*, vol. 5, no. 1, pp. 31-45, 2002.

[15] Leila CHERGUI, *Combinaison de classifieurs pour la reconnaissance de mots arabes manuscrits*. Constantine: Université Mentouri , 2013.

[16] Abderahmane Kefali, Toufik Sari, and Mokhtar Sellami, "Implémentation de plusieurs techniques de seuillage d'images de documents Arabes anciens," *inproceeding*, May 2009.

[17] ABA Safa and CHIKH Maroua, *Reconnaissance Des Mots Arabes Manuscrites*. Tébessa: Université de Larbi Tébessi, 2016.

[18] Oeivind Due Trier and Torfinn Taxt, "Evaluation of binarization methods for document images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 3, pp. 312-315, 1995.

[19] Sing T Bow, *Pattern recognition and image preprocessing*.: CRC press, 2002.

[20] Belaïd A, "Reconnaissance automatique de l'écriture et du document," *Campus scientifique, Vandoeuvre-Lès-nancy*, 2001.

[21] Supriana Iping and Nasution Albadr, "Arabic Character Recognition System Development," *Procedia Technology*, vol. 11, no. , pp. 334 – 341, 2013.

[22] M. Zaiz Faouzi, "Technique basée puzzle/SVM pour l'amélioration de la reconnaissance du texte arabe manuscrit," in *Thèse Doctorat en sciences Informatique*.: Université Mohamed Khider_ Biskra, 2017.

[23] Fatma El-Khaly and Maher A Sid-Ahmed, "Machine recognition of optically captured machine printed Arabic text," *Pattern recognition*, vol. 23, no. 11, pp. 1207-1214, 1990.

[24] Habib Goraine, Mike Usher, and Samir Al-Emami, "Off-line Arabic character recognition," *Computer*, no. 7, pp. 71-74, 1992.

[25] Cours en ligne, Hilditch's Algorithm for Skeletonization, Université de McGill, Montréal – Canada.. [Online]. http://cgm.cs.mcgill.ca/~godfried/teaching/projects97/azar/skeleton.html

[26] Maher Faik Esmaile, Estabraq Abdulredaa, and others, "Optical character recognition using active contour segmentation," *Journal of Engineering*, vol. 24, no. 1, pp. 146--158, 2018.

[27] B.M.F Bushofa and M Spann, "Segmentation of Arabic characters using their contour information.," in *Proceedings of 13th International Conference on Digital Signal Processing*.: IEEE, 1997, vol. 2, pp. 683–686.

[28] Volker Margner, "SARAT-A system for the recognition of Arabic printed text," in *Proceedings., 11th IAPR International Conference on Pattern Recognition. Vol. II. Conference B: Pattern Recognition Methodology and Systems*.: IEEE, 1992, pp. 561--564.

[29] G Olivier, Housem Miled, K Romeo, and Yves Lecourtier, "Segmentation and coding of Arabic handwritten words," in *Proceedings of 13th International Conference on Pattern Recognition*.: IEEE, 1996, vol. 3.

[30] Herbert Freeman, "On the encoding of arbitrary geometric configurations," *IRE Transactions on Electronic Computers*, no. 2, pp. 260--268, 1961.

[31] Toufik Sari, Labiba Souici, and Mokhtar Sellami, "Off-line handwritten Arabic character segmentation algorithm: ACSA," in *Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition*.: IEEE, 2002, pp. 452--457.

[32] Site official de OpenCV. [Online]. http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html

[33] Md Anwar Hossain and Sadia Afrin, "Optical Character Recognition based on Template Matching," *Global Journal of Computer Science and Technology*, 2019.

[34] BMF Bushofa and Michael Spann, "Segmentation and recognition of Arabic characters by structural classification," *Image and Vision Computing*, vol. 15, no. 3, pp. 167-179, 1997.

[35] Bharatratna P Gaikwad, Ramesh R Manza, and Ganesh Manza, "Automatic Video Scene Segmentation to Separate Script for OCR," *International Journal of Computer Applications*, vol. 975, p. 8887, 2014.

[36] Pritpal Singh and Sumit Budhiraja, "Feature extraction and classification techniques in OCR

systems for handwritten Gurmukhi Script--a survey," *International Journal of Engineering Research and Applications (IJERA)*, vol. 1, no. 4, pp. 1736--1739, 2011.

[37] Mark Nixon and Alberto S Aguado, *Feature extraction and image processing for computer vision*.: Academic Press, 2012.

[38] Stephane Lecoeuche, "Reconnaissance de caractères industriels par application d'un système de réseaux de neurones à boucle de rétroaction," in *Thès de Doctorat*.: UNIVERSITE DES SCIENCES ET TECHNOLOGIES DE LILLE , Nnovembre 1998.

[39] Steinbach M Tan P-N and Kumar V, *Introduction to data mining*.: Pearson Addison-Wesley., 2006.

[40] Xindong and Kumar, Vipin and Quinlan, J Ross and Ghosh, Joydeep and Yang, Qiang and Motoda, Hiroshi and McLachlan, Geoffrey J and Ng, Angus and Liu, Bing and Philip, S Yu Wu and others, "Top 10 algorithms in data mining," *Knowledge and information systems*, vol. 14, no. 1, pp. 1-37, 2008.

[41] Vladimir Vapnik, *The nature of statistical learning theory*.: Springer science & business media, 2013.

[42] Chih-Wei Hsu and Chih-Jen Lin, "A comparison of methods for multiclass support vector machines," *IEEE transactions on Neural Networks*, vol. 13, no. 2, pp. 415--425, 2002.

[43] Tom Mitchell, *Machine Learning*.: Mcgraw-Hill, 1997.

[44] Wongyu Cho, Seong-Whan Lee, and Jin H. Kim, "Modeling and recognition of cursive words with hidden Markov models.," *Pattern Recognition*, vol. 28, no. 121, pp. 1941-1953, 1995.

[45] L. R. Rabiner, "A tutorial on hidden markov models and selected applications," *speech recognition*, vol. 77, no. 2, pp. 257-286, 1989.

[46] Jawad H AlKhateeb, Jianmin Jiang, and Jinchang and Ipson, S Ren, "Component-based segmentation of words from handwritten Arabic text," *International Journal of Computer Systems Science and Engineering*, vol. 5, no. 1, 2009.

[47] Esraa Mohammad Ahmadoh and Adnan Abdul-Aziz Gutub, "Utilization of two diacritics for Arabic text steganography to enhance performance," *Lecture Notes on Information Theory Vol*, vol. 3, no. 1, 2015.

[48] Badr Al-Badr and Robert M Haralick, "Segmentation-free word recognition with application

to Arabic," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*.: IEEE, 1995, vol. 1, pp. 355--359.

[49] Karim Hadjar and Rolf Ingold, "Arabic Newspaper Page Segmentation," in *ICDAR*., 2003, vol. 3, pp. 895-899.

[50] Najoua Ben Amara, Abdel Bela, and Noureddine Ellouze, "Utilisation des modèles markoviens en reconnaissance de l'écriture arabe: état de l'art," , 2000.

[51] N Amara, Abdel Belaïd, Ben, and Noureddine Ellouze, "Modélisation pseudo bidimensionnelle pour la reconnaissance de chaînes de caractères arabes imprimés," in *Conférence Internationale Francophone sur l'Écrit et les Documents*., 1998, pp. 131-140.

[52] John and Gillies, Andrew Trenkle, Erik Erlandson, Steve Schlosser, and Stan Cavin, "Advances in Arabic text recognition," in *Proc. Symp. Document Image Understanding Technology*. Maryland, Columbia, April 23-25,2001.

[53] Ben Amara N.Essoukri, "Utilisation des Modèles de Markov Cachés Planaires en Reconnaissance de l'Ecriture Arabe Imprimé," in *Thèse de Doctorat, Ecole National d'Ingénieur de Tunisie*., 1999.

[54] N Ben Amara, Abdel Bela, and Noureddine Ellouze, "Modlisation pseudo bidimensionnelle pour la reconnaissance de cha

[55] Ahmed M Elgammal and Mohamed A Ismail, "A graph-based segmentation and feature extraction framework for Arabic text recognition," in *Proceedings of Sixth International Conference on Document Analysis and Recognition*.: IEEE, 2001, pp. 622-626.

[56] P. Burrow, "Arabic handwriting recognition," in *Master of science,thesis*.: School of Informatics, university of Edinburg, England, 2004.

[57] Adnan Amin, "Hand written Arabic Character Recognition by the IRAC system," in *5th international conference on pattern recognition, Miami, Florida*., 1980, pp. 729--731.

[58] A.Zahour, "Une Méthode de Reconnaissance de l'Ecriture Arabe Cursive," in *Thèse de Doctorat*., 1990.

[59] Nadir و Souici, Labiba و Sellami, Mokhtar Farah, "Arabic word recognition by classifiers and context," *Journal of Computer Science and Technology*, vol. 20, no. 3, pp. 402-410, 2005.

[60] S.Snoussi Madouri, "Modèle Perceptif Neuronal à Vision Globale-Locale pour la

Reconnaissance des Mots Arabes Omni-Scripteurs ," in *Thèse de Doctorat*.: Ecole National d'Ingénieurs de Tunis, 2002.

[61] Souci L, "Reconnaissance de Mots Arabes Manuscrits par Intégration Neuro-Symbolique," in *Thèse de Doctorat*, Université de Annaba, Ed., 2006.

[62] H Al-Yousefi and SS Upda, "Recognition of Arabic characters," *IEEE Transactions on Pattern Analysis \& Machine Intelligence*, no. 8, pp. 853-857, 1992.

[63] Sabri A Mahmoud, "Arabic character recognition using Fourier descriptors and character contour encoding," *Pattern recognition*, vol. 27, no. 6, pp. 815-824, 1994.

[64] Asif Mehedi. Quels-sont-les-qualités-et-défauts-du-langage-Python. [Online]. https://fr.quora.com/Quels-sont-les-qualités-et-défauts-du-langage-Python

[65] Kari Pulli, Anatoly Baksheev, Kirill Kornyakov, and Victor Eruhimov, "Real-time computer vision with OpenCV," *Communications of the ACM*, vol. 55, no. 6, pp. 61-69, 2012.

[66] Intel acquires Itseez. [Online]. https://opencv.org/intel-acquires-itseez/

[67] Jython. [Online]. https://www.jython.org/

[68] BMF Bushofa and Michael Spann, "Segmentation and Recognition of Printed Arabic Characters," in *BMVC*., 1995, pp. 543-552.

[69] Nicholas C Laan, Danielle F Pace, and Hagit Shatkay, "Initial model selection for the Baum-Welch algorithm as applied to HMMs of DNA sequences," *Queen's University, Kingston, ON, Canada*, 2006.

[70] Andrew McCallum, "Hidden Markov Models Baum Welch Algorithm," in *Introduction to Natural Language Processing CS 585*., 2004.

[71] Ethem Alpaydin, *Introduction to machine learning*.: MIT press, 2009.

[72] G David Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268-278, 1973.

[73] Mohammad Norouzi, David J Fleet, and Ruslan R Salakhutdinov, "Hamming distance metric learning," in *Advances in neural information processing systems*., 2012, pp. 1061-1069.

[74] Random-Word 1.0.4. [Online]. https://pypi.org/project/Random-Word/?fbclid=IwAR3mXDJUi5hqq4Zh1SzCS5RLpUnA5rxtuyoGR9rhQQVTauqJ43o2CFWJ2og