République Algérienne Démocratique et Populaire
Université Amar Télidji - Laghouat
Faculté de Technologie
Département d'Informatique

# Mémoire

Pour obtenir le diplôme de

## Magister en informatique

Option : Informatique Répartie et Mobile

## Thème

---

# Techniques d'apprentissage automatique pour la reconnaissance des formes : application à la reconnaissance de l'écriture arabe manuscrite

---

Présenté par : Mr. **Nadir HADJ SAD**
Dirigé par : Prof. **Abdelouahab MOUSSAOUI**

Devant le jury composé de :

| | | | |
|---|---|---|---|
| YAGOUBI Mohamed Bachir | Professeur | Président | Université de Laghouat |
| MOUSSAOUI Abdelouahab | Professeur | Rapporteur | Université de Sétif 1 |
| CHERROUN Hadda | MCA | Examinatrice | Université de Laghouat |
| OUINTEN Youcef | Professeur | Examinateur | Université de Laghouat |

Année universitaire 2014/2015

# *DÉDICACES*

*À ma Mère et à mon Père,*

*J'exprime mes sincères remerciements et toute ma reconnaissance pour leurs efforts, sans lesquels je n'aurai jamais pu achever mes études.*

*À ma femme,*

*Je te remercie pour ton aide, ta patience, ta compréhension et ton soutien permanent.*

*À mes frères, à mes sœurs.*
*À tous mes amis.*
*Je dédie ce travail.*

# *Remerciements*

*C'est avec un grand plaisir que je réserve ces lignes en signe de gratitude et de reconnaissance à tous ceux qui ont contribué à l'aboutissement de ce travail.*

*Tout d'abord, je tiens à exprimer ma très grande gratitude à mon promoteur, M. A. MOUSSAOUI, professeur à l'université de Ferhat Abbas Sétif1, pour la confiance et l'intérêt qu'il m'a porté pour cette étude, m'a consacré, ses conseils et ses encouragements tout au long de la réalisation du présent travail.*

*Mes remerciements sont également adressés à l'ensemble des membres du jury, pour leur disponibilité et l'intérêt qu'ils ont accordé au présent travail.*

# ملخص

الهدف من هذه المذكرة هو تطوير نظام تصنيف آلي للوثائق العربية المكتوبة بخط اليد، الإشكالية التي لم تعالج من قبل، أو ربما لقيت اهتماما قليلا جدا من طرف الباحثين. التصنيف هو عملية اكتشاف الموضوع الذي تتناوله وثيقة ما من خلال فحص الكلمات الواردة فيها. من أجل استخراج كلمات الوثيقة، قمنا بإنجاز نظام للتعرف الآلي على الكلمات العربية المكتوبة بخط اليد. المنهجية المتبعة للتعرف هي منهجية تحليلية تعتمد على نماذج ماركوف المخفية (HMM) والتقسيم الضمني : يتم تقسيم صور الكلمات باستخدام النوافذ المنزلقة والتي تسمح بتحويل الصور إلى سلاسل من أشعة الخصائص. يتم نمذجة الحروف بواسطة نماذج HMM، ثم بناء نماذج الكلمات بربط نماذج الحروف المكونة لها.

تجرى عملية تصنيف الوثائق على النصوص الناتجة عن عملية التعرف. في البداية يتم تحديد مجموعة جزئية من الكلمات لتمثيل الوثائق بواسطة أشعة الخصائص، هذه الأشعة تقدم بعد ذلك لمصنف من نوع *الجار الأقرب* لإجراء التصنيف. نتائج التجارب التي أجريناها بينت أن نظامنا لتصنيف الوثائق حصل على أداء مرضٍ على قاعدة البيانات التي أنشأت خصيصا لهذه الدراسة.

# Résumé

L'objectif de ce mémoire est d'élaborer un système pour la catégorisation de documents manuscrits arabes, une problématique très peu abordée, voir pas du tout, dans la littérature. La catégorisation consiste à détecter le thème abordé dans un document à travers l'examen des mots contenus dans celui-ci. Afin d'extraire les mots des documents, nous avons mis en place un système de reconnaissance de mots manuscrits arabes. L'approche utilisée pour la reconnaissance est une approche analytique à base des modèles de Markov cachés (HMM) avec segmentation implicite : les images de mots sont découpées de manière implicite par l'utilisation de fenêtres glissantes qui permettent de transformer les images en séquences de vecteurs de caractéristiques. Les caractères des mots sont modélisés par des HMMs gaussiens, et les mots sont reconstruits ensuite par concaténation des modèles de caractères qui les composent.

La catégorisation des documents est effectuée sur les transcriptions issues de la reconnaissance; un sous-ensemble de mots est sélectionné d'abord pour représenter les documents par des vecteurs de caractéristiques, ces vecteurs sont soumis par la suite à un classifieur de type *k*-ppv qui fait la catégorisation. Les résultats obtenus montrent que notre système de catégorisation obtient des performances satisfaisantes sur la base de documents construite spécialement pour cette étude.

# Table des matières

# Table des figures

# Liste des tableaux

# Introduction

Ces dernières années ont connu l'apparition de quantités de plus en plus importantes de documents arabes manuscrits sur papier. Ils peuvent consister par exemple en de courtes phrases, de paragraphes, des équations, des dessins. Ces documents sont générés par les différentes activités humaines : administratives, économiques, scientifiques, etc. Une grande quantité des documents sont numérisés pour faciliter leur stockage et leur transfert électroniques. Cela pose la question de l'accès à l'information textuelle contenue dans ces documents.

La reconnaissance automatique de documents manuscrits, qui vise à transformer un texte manuscrit en un texte électronique, a connu un essor rapide durant la dernière décennie. Le spectre de ses applications est très large, telle que la reconnaissance de montants sur les chèques, la reconnaissance d'adresse postale, le traitement automatique de courrier entrant et l'analyse de documents historiques. Ce processus est très complexe due à l'énorme variabilité d'écriture entre scripteurs, chaque scripteur ayant son propre style d'écriture. La variabilité d'écriture peut exister aussi, même au sein d'un ensemble de mots écrits par un même scripteur. De plus, en considérant les particularités morphologiques de l'écriture arabe, le challenge s'accentue.

D'autre part, la catégorisation automatique de documents textuels devenue un domaine de recherche très actif. La tâche de catégorisation vise à la détection du thème abordé dans un texte à travers l'examen des mots contenus dans celui-ci. Le filtrage de courriers électroniques indésirables, l'organisation de documents et la fouille d'opinion dans le web social sont par exemple des applications de ce domaine. Dans ce contexte, il est possible d'envisager une application de la reconnaissance de l'écriture manuscrite sur le problème de catégorisation en apportant diverses fonctionnalités aux documents manuscrits, telles que l'organisation automatique, le filtrage de courriers indésirables ou l'indexation thématique de documents manuscrits.

L'objectif de ce mémoire est d'élaborer un système complet pour la catégorisation automatique de documents manuscrits arabes. Une façon d'appréhender le problème de la

catégorisation de ce type de documents est de se ramener à des données textuelles grâce à un système de reconnaissance de l'écriture. Bien que la catégorisation de documents électroniques arabes fait l'objet de nombreux travaux, cette problématique est encore très peu abordée, voir pas du tout, pour le manuscrit. La contribution apportée dans ce mémoire est que ceci à notre connaissance la première fois que la catégorisation est appliquée aux documents manuscrits arabes. D'autre part, vu l'indisponibilité d'un corpus standardisé nous avons crée une base de documents manuscrits arabes multi-scripteurs spécifiquement pour cette étude.

La présentation de ce mémoire se déroule en quatre chapitres.

Le premier chapitre introduit la reconnaissance de formes en décrivant les étapes composant ce processus. Il fait également une revue des principales approches d'apprentissage et de classification automatiques avec une présentation de l'état de l'art de ces approches. Diverses domaines d'applications de la reconnaissance de formes sont explorés dans ces travaux.

Dans le deuxième chapitre la problématique de reconnaissance de l'écriture manuscrite arabe est abordée. Un état de l'art se fait selon les différentes phases de traitement. Ceci nous permet d'introduire notre système de reconnaissance de l'écriture manuscrite arabe, présenté dans le Chapitres 4.

Le troisième chapitre est dédié à l'extraction de connaissances à partir de données textuelles. Nous nous intéressons aux techniques et algorithmes qui permettent de réaliser des tâches d'extraction d'information, de catégorisation et de segmentation de documents électroniques. Une définition de chacune de ces tâches est présentée ainsi qu'un ensemble de méthodes d'apprentissage automatique qui peuvent être appliquées sur des données textuelles.

Le quatrième chapitre présente notre système complet de catégorisation automatique de documents manuscrits arabes que nous avons développé. Une description des deux principales tâches du système est présentée avant d'aborder aux travaux expérimentaux. Nous décrivons les bases de données que nous avons construit pour la mise en œuvre du système. Les résultats montrent que notre approche de catégorisation de documents manuscrits arabes permet d'obtenir des performances acceptables sur la base de documents manuscrits construite spécialement pour cette étude.

# Chapitre 1

# Reconnaissance des formes : état de l'art

## Introduction

Il est généralement facile pour l'Homme de distinguer sans réfléchir une multitude de formes, issues de différentes sources. Par exemple, nous pouvons facilement et naturellement différencier le son d'une voix humaine, de celle d'un violon, un chiffre manuscrit "3", d'un "8", et l'arôme de rose, de celle d'un oignon. Cette diversité de tâches que l'homme peut réaliser est sans doute sa plus grande force ;

La reconnaissance des formes peut être vue comme la transposition à l'informatique de la faculté humaine d'analyser les signaux qui l'entourent, visuels ou sonores, afin de les comparer, de les classifier ou de les identifier. Cette discipline intervient dans de nombreux domaines tels que la reconnaissance vocale, la reconnaissance d'écriture, l'automatisation industrielle, le diagnostic médical, la classification de documents, etc. La complexité de la tâche, la diversité des applications et la montée en puissance des ordinateurs font que la reconnaissance des formes est devenue un domaine où les travaux de recherche évoluent depuis des décennies et tentent de se rapprocher le plus possible des capacités humaines.

Le présent chapitre introduit la reconnaissance des formes et propose un état de l'art des principales méthodes d'apprentissage et de classification. Nous présenterons en Section 1.1 la structure générale d'un tel système en décrivant ses différentes étapes. Par la suite, en Section 1.2 nous proposerons un bref panorama des méthodes et techniques utilisées pour la classification, où nous nous intéresserons aux méthodes statistiques, largement utilisées en reconnaissance des formes. Enfin, nous terminerons ce chapitre avec une conclusion.

# 1.1 Architecture d'un système de reconnaissance des formes

L'architecture d'un système de reconnaissance des formes est composée de trois grandes étapes, comme le montre la Figure 1.1 : le prétraitement, l'extraction des caractéristiques, l'apprentissage et la classification. Une fois l'ensemble de données est acquis, il est prétraité, de sorte qu'il devient approprié à l'étape suivante. L'extraction des caractéristiques consiste à transformer l'ensemble de données en un ensemble réduit de caractéristiques qui sont censées être représentatives des données d'origine. Ces caractéristiques sont utilisées pour apprendre un classifieur, qui va être utiliser ensuite pour séparer les données d'entrée en différentes classes en fonction du problème à résoudre.

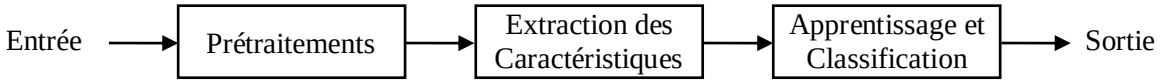



FIG. 1.1– Système de reconnaissance des formes.

## 1.1.1 Prétraitement

On appelle une observation, les données brutes issues des capteurs du système. Ces données ne se présentent pas toujours comme le souhaite l'utilisateur. Le problème est surtout lié à [Tsopzé, 2010]:

- la présence de bruits dans les données qui peuvent être dus au défauts des capteurs ou à des interférences avec d'autres sources de signaux (la parole en milieu sonore, l'encre du verso qui traverse le papier et dont la trace est visible sur la feuille du manuscrit, les fonds imagés des chèques, etc.) ;
- la présence de valeur inconnue pour certains attributs (données manquantes) ;
- l'inconsistance et l'incohérence des données.

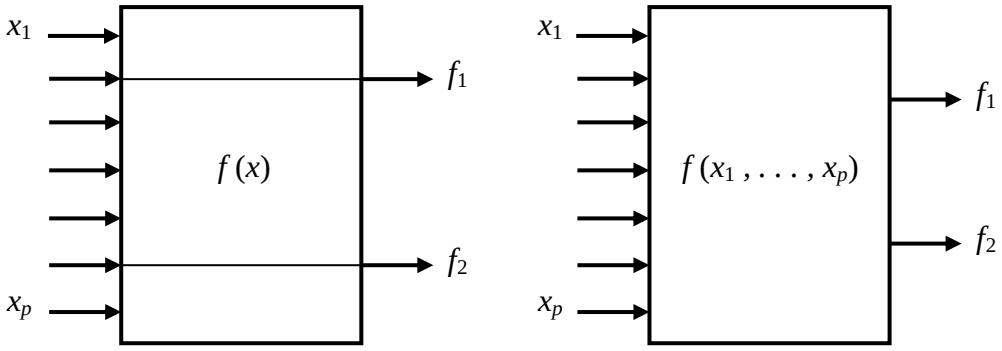
L'objectif du prétraitement est d'améliorer la qualité des données brutes afin d'améliorer l'efficacité des algorithmes qui utilisent ces données, et par conséquent les résultats obtenues. L'étape du prétraitement regroupe [Cornuejols et Miclet, 2010] :

- L'intégration des données : ce processus permet de regrouper les données provenant de plusieurs sources et d'uniformiser le format.

- Le nettoyage : il s'agit de supprimer les bruits et de corriger les inconsistances.
- La transformation des données : elle consiste à mettre les données sous un type plus approprié aux algorithmes de traitement.
- Le traitement des valeurs manquantes : elle consiste à remplacer les valeurs non renseignées dans les observations.

## 1.1.2 Extraction des caractéristiques

L'étape d'extraction de caractéristiques (aussi appelées descripteurs), est une étape très importante pour un système de reconnaissance des formes. En effet, même si on utilise un classifieur très performant celui-ci ne peut compenser un mauvais choix des caractéristiques. On peut définir l'extraction des caractéristiques comme l'extraction à partir des formes (données d'entrée), de l'information la plus pertinente pour un problème de classification donnée, et qui permet de décrire de façon non équivoque les formes appartenant à une même classe tout en les différenciant des autres classes. A l'issus de cette étape, les formes sont représentées par un ensemble de valeurs numériques ou symboliques.




(a) sélection de caractéristiques (b) transformation de caractéristiques

FIG. 1.2 – Réduction de données par (a) sélection de caractéristiques et (b) transformation de caractéristiques (extraite de [Andrew et Keith, 2011]).

Généralement un grand nombre de caractéristiques sont calculées à partir des formes d'entrée. Pour cela, le processus d'extraction de caractéristiques est normalement suivi par une étape de réduction de dimension des données conduisant à un ensemble de caractéristiques aussi petit que possible. Cette étape permet d'améliorer les performances de classification. Étant donné un ensemble de *p* caractéristiques, la réduction peut être obtenue en suivant deux

manières différentes : sélection et transformation. Ces deux approches sont illustrées sur la Figure 1.2.

**La sélection de caractéristiques,** consiste à ne conserver parmi l'ensemble des caractéristiques, que les plus pertinentes pour la discrimination entre les formes de différentes classes. Dans cette approche les caractéristiques ne contribuant pas à la classification sont identifiées et négligées. Diverses méthodes peuvent être utilisées afin de sélectionner les caractéristiques les plus pertinentes : l'information mutuelle, le chi-carré ($\chi^2$) et le gain d'information (IG). Certaines de ces méthodes serons présentées dans le Chapitre 3, Section 3.3.3.

**La transformation de caractéristiques,** consiste à créer un nouvel (et plus petit) ensemble de caractéristiques par une transformation linéaire ou non linéaire de *p* caractéristiques vers un espace de caractéristiques de dimension inférieure. La méthode la plus largement utilisée pour la transformation de caractéristiques est l'analyse en composantes principales (ACP; en anglais *PCA, Pincipal Component Analysis*).

### 1.1.3 Apprentissage et classification

Une fois l'ensemble des caractéristiques choisi, il faut donc choisir une méthode de reconnaissance des formes. La classification consiste alors à identifier les classes auxquelles appartiennent les formes à partir des caractéristiques préalablement choisies et calculées. L'algorithme qui réalise cette application est appelé classifieur (*classifier* en anglais). Pour ce faire, on va induire (apprendre) un classifieur à partir d'exemples (données ou échantillon d'apprentissage).

En fonction des informations *a priori* disponibles sur les classes, on distingue trois types de méthodes d'apprentissage ou de classification : les méthodes supervisées, les méthodes non supervisées et les méthodes semi-supervisées.

Lorsque les classes sont connues initialement et que l'échantillon d'apprentissage est constitué de données étiquetées (exemples de formes), on parle d'apprentissage ou de classification supervisé. Ces données sont utilisées à apprendre des caractéristiques de classes pour construire un classifieur qui affecte les nouvelles formes à une classe. Le but dans la méthode supervisée est de minimiser l'erreur de classification. Le modèle de chaque classe est alors représenté par une fonction d'appartenance qui détermine la valeur d'appartenance d'une forme à une classe.

Au contraire, quand aucune information n'est disponible sur les classes des formes, l'apprentissage est non supervisé. Les méthodes non supervisées, aussi appelées méthodes de segmentation ou partitionnement (*clustering* en anglais), sont basées sur des fonctions de similarité. Quand des formes aux caractéristiques similaires apparaissent, elles sont classifiées dans la même classe et à l'inverse quand leurs caractéristiques sont différentes une nouvelle classe est crée par le classifieur. Une fois que le classifieur a estimé les fonctions d'appartenance des classes, les nouvelles formes peuvent être assignées à la classe pour laquelle elles ont la valeur d'appartenance maximale [Hartert, 2010].

Le troisième type d'apprentissage, concerne l'apprentissage semi-supervisés qui combine les deux modes d'apprentissage : supervisé et non supervisé. Il utilise l'ensemble d'apprentissage étiqueté pour estimer les fonctions d'appartenance de chaque classe connue et l'apprentissage non supervisé pour améliorer la précision de cette estimation, détecter les nouvelles classes et apprendre leurs fonctions d'appartenance.

## 1.2 Approches de reconnaissance des formes : état de l'art

Dans cette section, nous présentons un état de l'art des méthodes et techniques d'apprentissage et de classification, où nous nous intéressons aux méthodes statistiques, largement utilisées en reconnaissance des formes.

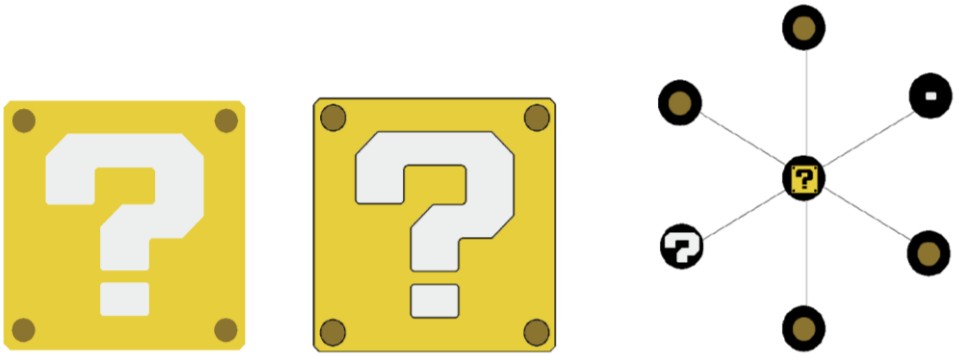
### 1.2.1 Approche syntactique et structurelle

Dans l'approche structurelle, une forme est représentée par une décomposition en sous-parties et par la description de la relation entre ces sous-parties. Pour ce faire, plusieurs structures de données peuvent être utilisées pour représenter une forme.

L'exemple montré dans la Figure 1.2, extrait de [Sidère, 2012], illustre une représentation structurelle de l'image (1.3(a)), sous forme de graphe. L'image est segmentée en régions (1.3(b)). Chaque sommet du graphe représente alors une région de l'image et les arêtes représentent les relations d'adjacence entre les régions (1.3(c)).

Notons que des étiquettes peuvent être ajoutées sur les sommets et/ou sur les arêtes. L'étiquetage des sommets permet une description des sous-parties et celui des arêtes de préciser la nature des relations. Il est possible de combiner l'usage des graphes avec une

approche statistique ; les outils statistiques étant individuellement utilisés sur les étiquettes de nœuds ou des arcs.



(a) (b) (c)

FIG. 1.3 – Exemple de la représentation d'une image par un graphe d'adjacence (extraite de [Sidère, 2012]).

Avec les méthodes à base de grammaires, la reconnaissance d'une image proposée consiste à analyser sa représentation pour tester si elle peut être générée par la grammaire. Cette méthode est utile pour les applications où les formes des images peuvent être définies précisément par un ensemble de règles [Barrat, 2009].

Les approches syntactique et structurelle nécessitent une mesure de la similarité entre deux représentations structurelles en utilisant des opérations de comparaison complexes. Cette complexité représente l'inconvénient majeur de ces approches car elle augmente les coûts de traitements et rend la reconnaissance plus difficile.

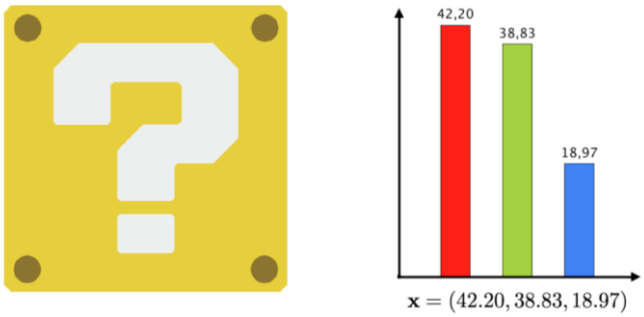
## 1.2.2 Approche statistique

Dans l'approche statistique, la forme est représentée par un vecteur de caractéristiques de $d$ dimensions $X = [x_1, x_2, ..., x_d]^T$ , appelé aussi observation ou mesure, ce vecteur est calculé grâce à des fonctions mathématiques, ou des algorithmes, appelés descripteurs.

Une représentation statistique possible de l'image (a) de la Figure 1.3 est présentée dans la Figure 1.4. La couleur de l'image peut être caractérisée par un vecteur numérique. Le vecteur est composé des intensités moyennes des 3 couleurs primaires (rouge, vert et bleu) calculées sur l'ensemble de l'image.

Pour un problème donné, tous les vecteurs qui représentent l'ensemble de formes peuvent être positionnés dans un espace vectoriel (Euclidien) $R^d$, appelé espace d'observations ou

espace de caractéristiques, où chaque vecteur correspond à un point. Ceux-ci peuvent alors être regroupés en amas ou nuage, chacun de ces amas étant associé à une classe particulière.




(a) (b)

FIG. 1.4 – Exemple de la représentation statistique d'une image par l'histogramme des 3 couleurs primaires (extrait de [Sidère, 2012]).

Un exemple pour un problème à deux classes est illustré à la Figure 1.5. Un bon choix de caractéristiques pertinentes est indispensable pour atteindre un grand pouvoir de discrimination. Si les caractéristiques sont bien déterminées, les classes sont bien discriminées et elles sont situées dans différentes régions de l'espace de représentation. En raison du bruit, des imperfections de mesure, des distorsions de forme, etc., les observations possibles de chaque classe se traduisent par des points un peu loin les uns des autres.

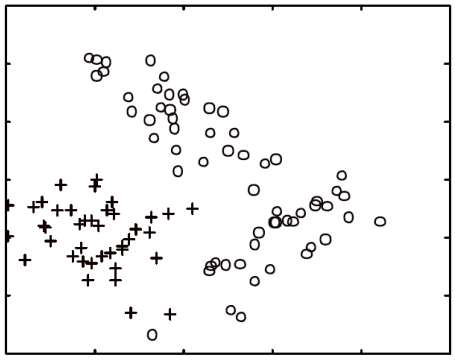


FIG. 1.5 – Représentation des formes appartenant à deux classes distinctes, dans un espace à deux dimensions (extrait de [Gosselin, 2000]).

### 1.2.3 Notion de classifieur statistique

Le rôle d'un classifieur est de déterminer, parmi un ensemble fini de classes, à laquelle appartient une forme donnée, à partir des vecteurs de caractéristiques extraits de cette forme.

Un classifieur doit être capable de modéliser au mieux les frontières qui séparent les classes les unes des autres. Cette modélisation fait appel à la notion de fonction discriminante, qui permet d'exprimer le critère de classification de la manière suivante [Gosselin, 2000]:

Assigner la classe $w_i$ à la forme représentée par le vecteur $X$ si, et seulement si, la valeur de la fonction discriminante de la classe $w_i$ est supérieure à celle de la fonction discriminante de n'importe quelle autre classe $w_j$ , ou encore, sous forme mathématique:

$$X \in w_i \Leftrightarrow \Phi_i(X) \geq \Phi_j(X) \quad \forall\, j = 1,2,...,C;\ i \neq j. \tag{1.1}$$

où $\Phi_i(X)$ est appelé fonction discriminante de la classe $w_i$ , et $C$ est le nombre total de classes.

## 1.2.4 Méthodes statistiques bayesiennes

Soient :

$P(w_i)$ est la probabilité *a priori* d'avoir la classe $w_i$ .

$P(X|w_i)$ est la fonction de densité de probabilité d'observer $X$, étant donné la classe $w_i$ .

$P(w_i|X)$ est la probabilité *a posteriori* que la classe correcte soit $w_i$ , lorsque l'on observe $X$.

La méthode de classification de bayes transforme la probabilité a *priori* d'une classe $w_i$ en probabilité a *posteriori* à l'aide des informations contenues dans l'observation $X$, en utilisant la règle de Bayes qui s'exprime :

$$P(w_i \mid X) = \frac{P(w_i)P(X \mid w_i)}{P(X)} \tag{1.2}$$

où $$P(X) = \sum_{i=1}^{C} P(X \mid w_i)P(w_i) \tag{1.3}$$

Soit une fonction $\lambda(i \mid j)$, qui désigne le coût encouru lorsque la classe $w_i$ est assignée à une forme appartenant à la classe $w_j$. Le classifieur optimal est celui qui minimise le coût total obtenu, étant donné une fonction de coût particulière. Une telle fonction peut être définie de manière élémentaire sur base de l'opérateur delta de *Krönecker*:

$$\lambda(i \mid j) = 1 - \delta_{ij} = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases} \tag{1.4}$$

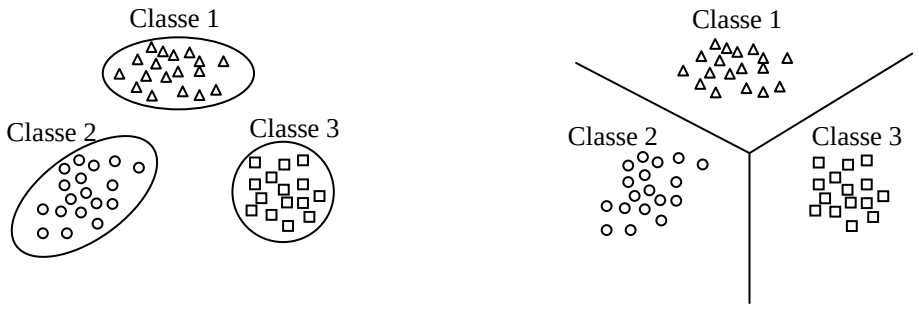
L'objectif du classifieur optimal, dit *Bayesien*, est de minimiser la probabilité d'erreur, c'est-à-dire la probabilité d'affecter une forme à une classe incorrecte. Le critère de classification devient ainsi [Fukunaga, 1990] :

$$X \in w_i \Leftrightarrow P(w_i \mid X) > P(w_j \mid X) \quad \forall\, j = 1,2,...,C;\ i \neq j. \qquad (1.5)$$

où $P(w_i \mid X)$ est la probabilité *a posteriori* de la classe $w_i$ . La classe attribuée à la forme représentée par le vecteur $X$ est alors celle dont la probabilité étant donné $X$ est supérieure à la probabilité de n'importe quelle autre classe, étant donné $X$.

Le calcul exact des probabilités *a posteriori* est cependant un problème très complexe, et s'avère rarement possible en pratique, ainsi, le classifieur de Bayes est non réalisable en général, mais représente une référence théorique. Des méthodes de classification ont été développées pour estimer la densité de probabilité (les noyaux de Parzen, les *k*-plus proches voisins, ...), ou estimer directement la probabilité *à posteriori* (les méthodes neuronales, les modèles de Markov cachés, ...).

Parmi l'ensemble des méthodes statistiques de classification, il est possible de distinguer deux catégories d'approches [Milgram et al., 2005] : celles agissant par modélisation et celles agissant par séparation. Le premier type d'approche cherche à déterminer un modèle le plus fidèle possible de chacune des classes, alors que l'objectif du second type est d'optimiser des frontières de décision de manière à séparer au mieux les classes. Un exemple de chaque approche est illustré à la Figure 1.6 :




(a) (b)

FIG. 1.6 – Deux types d'approche de classification statistiques: (a) par modélisation, (b) par séparation.

## 1.2.5 Approches par modélisation

Ce type d'approche est basé sur le développement d'un modèle pour chacune des classes et l'utilisation d'une mesure de similarité pour comparer la donnée à classifier à chacun des modèles [Milgram et al., 2005]. Ces classifieurs sont entièrement définis par un ensemble fini de paramètres qu'il suffit de calculer (d'où ils sont aussi appelés des classifieurs

paramétriques). Les approches par séparation, de part leur nature discriminante, sont plus performantes pour traiter les données ambiguës, mais peu aptes à gérer les données ou points abérants (*outliers*) [Liu et al., 2002].

L'un des plus simples classifieurs qui puissent être conçus est le **classifieur Euclidien**. La classe dont le vecteur de caractéristiques moyen est le plus proche (au sens de la distance Euclidienne) du vecteur de caractéristiques de la forme à classifier est assignée à cette dernière. En pratique, les vecteurs de caractéristiques moyens ne sont pas disponibles, ils doivent être estimés à partir d'un ensemble fini de prototype de chaque classe.

Pour le **classifieur Quadratique**, comme le nom l'indique, les frontières de décision fournies sont décrites par des fonctions discriminantes quadratiques, dont les vecteurs de caractéristiques moyens et les matrices de covariance ne peuvent qu'être estimées à partir des formes disponibles.

Les fonctions discriminantes utilisées dans le **Classifieur Gaussien** sont basées sur une estimation paramétrique des fonctions de répartition des vecteurs de caractéristiques $P(X|w_i)$. Ce classifieur suppose que les éléments de chaque classe possèdent une distribution Gaussienne multivariable. Dans la mesure où cette hypothèse s'avère exacte, le classifieur Gaussien permet d'obtenir les frontières optimales de décision de Bayes. Pour calculer les fonctions de répartition des vecteurs $P(X|w_i)$, on doit estimer les paramètres suivants : les probabilités *a priori*, les vecteurs de caractéristiques moyens et les matrices de covariance.

## 1.2.6 Approche par séparation

Le moyen le plus simple de faire de la classification est de définir une fonction de distance entre les vecteurs caractéristiques, et d'affecter chaque forme d'entrée inconnue à la classe dont le barycentre est le plus proche de la forme requête, selon la fonction de distance définie. La motivation première d'une telle approche est qu'il est naturel de considérer qu'un élément appartient à une classe s'il est plus proche de cette classe que de toutes les autres.

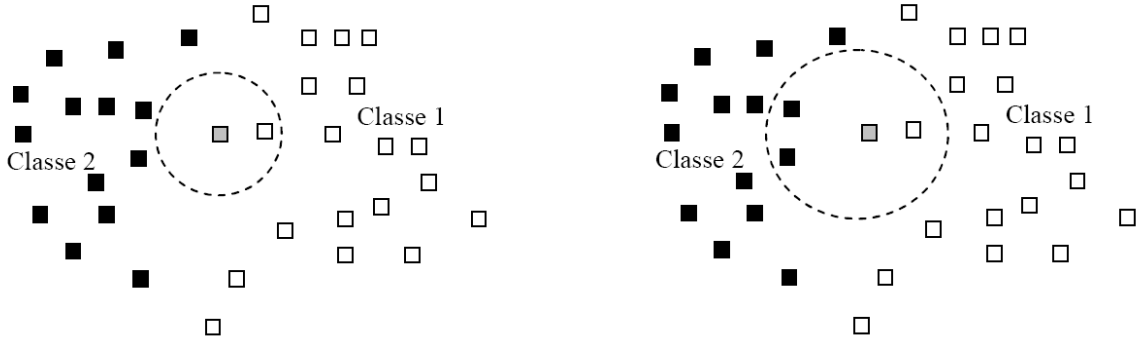
### 1.2.6.1 La Méthode des *k*-plus Proches Voisins

Un classifieur par les *k*-plus proches voisins ou *k*-ppv (*k*-Nearest-Neighbour en anglais) est l'un des classifieurs supervisés. Il représente une extrapolation du classifieur Euclidien décrit précédemment. Au lieu d'utiliser le vecteur de caractéristiques moyen $M_i$ comme

unique prototype d'une classe, plusieurs représentants sont pris pour chaque classe (l'échantillon d'apprentissage). On calcule la distance entre chacun de ceux-ci et celui de la forme à classifier *x*, puis on construit l'ensemble des *k*-plus proches voisins de *x*. La règle de décision consiste à attribuer à la forme *x* la classe qui a le plus de représentants dans cet ensemble.

Cette règle est appelée règle des *k*-plus proches voisins où *k* est le nombre de voisins considérés. Dans le cas où *k* = 1, la règle s'appelle la règle de classification du plus proche voisin. Elle assigne à *x* simplement la même classe que le point d'apprentissage le plus proche [Cornuejols et Miclet, 2010]. La fonction discriminante d'une classe est simplement le nombre de prototypes de cette classe qui se situent parmi les *k*-ppv de la forme à classifier.

L'exemple suivant, extrait de [Boukharouba, 2011], est utilisé pour illustrer cette méthode. La Figure 1.7(a) représente un problème de classification à 2 classes avec un algorithme *k*-ppv où *k* = 1. Les données étiquetées appartenant à la classe 1 et la classe 2 sont respectivement représentées par des carrés blancs et noirs. Une observation non étiquetée *x* (donnée à classer), est représentée par un carré gris. Afin de classer la donnée *x* dans un voisinage de *k*= 1, on cherche, au sens de la métrique choisie pour le problème (sur ce dessin, euclidienne), le plus proche voisin de *x*. Le cercle entoure le point à classer et son plus proche voisin. Le plus proche voisin de *x* est un point de la classe 1, d'où *x* sera donc affecté à la classe 1.




(a) (b)

Fig. 1.7 – Exemple de classification bi-classe avec un *k*-ppv : (a) *k*=1, (b) *k*=3.

La Figure 1.7(b) représente le même problème, mais avec un voisinage de *k*= 3. Afin de classer le point *x*, on recherche les 3 points d'apprentissage les plus proches de *x*. Le cercle entoure le point à classer et ses trois plus proches voisins. Parmi les 3 points les plus proches

de *x*, il y en a 2 de la classe 2. Un majoritaire est effectué et le point *x* sera donc affecté à la classe 2.

L'approche par les *k*-ppv possède au moins deux avantages : elle est très simple et elle ne nécessite pas d'apprentissage mais simplement le stockage des données d'apprentissage. Les seuls aspects nécessitant une pré-spécification sont le nombre de voisins, *k*, la métrique de distance, et l'échantillon d'apprentissage [Andrew et Keith, 2011]. Cependant que le volume de calcul ainsi que la quantité de mémoire exigées par les classifieurs de cette méthode, sont souvent prohibitifs, au vu du grand nombre de prototypes à prendre en considération et de distance à calculer. La règle des *k*-ppv fait implicitement une estimation comparative de toutes les densités de probabilités des classes apparaissant dans le voisinage de *x* (données à classifier) et choisit simplement la plus probable : elle approxime donc la décision bayesienne [Cornuejols et Miclet, 2010].

Caglayan et al. [Caglayan et al. 2013] ont développé une approche pour classifier les plantes en fonction des caractéristiques de la feuille. Des caractéristiques de forme et de couleur sont extraites à partir de l'images de la feuille. Ces caractéristiques sont utilisées avec les algorithmes de classification suivants : *k*-ppv, SVM, Naive Bayes, et forêts d'arbres décisionnels (forêts aléatoires) pour reconnaître 32 espèces de plantes.

L'expérimentation sur 1897 images de feuilles montre que l'utilisation des caractéristiques de forme et de couleur ensemble peut améliorer le taux de précision de reconnaissance. Les auteurs ont étudié les paramètres des algorithmes obtenant les meilleurs scores, *k*-ppv et forêts aléatoires: *k* et le nombre d'arbres, respectivement, et montrent que le meilleur taux de classification 96% est obtenu par l'algorithme des forêts aléatoires.

Ali et al. [Ali et al., 2013] ont proposé un modèle géométrique de visage dont le but est de manipuler le défi de variations d'âge pour le visage qui affectent le processus de reconnaissance de visage. Les auteurs localisent 12 points caractéristiques faciales pour extraire 6 zones triangulaires différents. Les triangles caractéristiques du même visage sont comparées deux à deux en appliquant des équations mathématiques, les 50 valeurs de similarité calculées sont enregistrées dans un vecteur de caractéristiques. Le modèle géométrique développé a pour objectif de maintenir le degré de similarité entre les six triangles caractéristiques du même visage.

Les auteurs ont effectué leur expérimentation sur la base FG-NET qui contient 1002 images de visage de 82 personnes divisées en 3 catégories d'âge pour tester des différentes

méthodes de sélection de caractéristiques. Ils utilisent pour la reconnaissance 5 classifieurs à savoir : *k*-moyenne, *k*-ppv, forêts aléatoires, naïf Bayésien et réseau Bayésien. Le meilleur taux de précision de reconnaissance dépasse 99 %, il est obtenu en utilisant le classifieur *k*-ppv.

### 1.2.6.2 La classification floue non supervisée : C-moyennes floues

La classification floue permet de partitionner l'espace de caractéristiques en groupes (ou *clusters*) de points similaires de telle sorte que les points d'un même groupe soient plus similaires entre eux qu'avec les points des autres groupes. Chaque point de l'espace de données peut appartenir à toutes les groupes avec différents degrés d'appartenance entre 0 et 1. Chacun de ces groupes ou régions flous est caractérisé par un vecteur appelé centre de groupe. L'appartenance des données à un groupe est basée sur la vérification d'un degré de similarité. Habituellement, il est calculé en utilisant une mesure appropriée de similarité qui quantifie la distance entre les données représentées comme des points dans l'espace de caractéristique, et les centres des groupes. La classification floue ne demande aucune connaissance *a priori* sur la structure des données. Cependant elle nécessite l'initialisation des algorithmes par un nombre de classes *C*, paramètre d'entrée.

Un exemple typique des algorithmes de classification floue non supervisée est l'algorithme des C-moyennes floues (*Fuzzy C-Means : FCM*) [Ergen et al., 2012] [Shi et al., 2013] et l'algorithme des *C*-moyennes floues possibilistiques (*Possibilistic Fuzzy C-Means : PFCM*). Dans cette section nous présentons le principe de l'algorithme des C-moyennes floues.

l'algorithme des C-moyennes floues est développé par Bezdek. C'est une extension directe de l'algorithme classique des k-moyennes (*k-means*), en introduisant la notion d'ensemble flou. Il est basé sur la minimisation de la fonction objective suivante :

$$E = \sum_{j=1}^{N} \sum_{i=1}^{C} \mu_{ij}^{m} \left\| x_j - v_i \right\|^2 \qquad (1.6)$$

où :

*C* est le nombre de classes, et *N* est le nombre d'échantillons de données,

$x_j$ l'échantillon numéro *j*, et $v_i$ désigne le centroïde de la *i*-ième classe, $\mu_{ij}$ est le degré d'appartenance de l'échantillon $x_j$ à la *i*-ième classe.

Les $\mu_{ij}$ forment une matrice $C \times N$ , dite matrice d'appartenance $M$, qui vérifie $\sum_{i=1}^{c} \mu_{ij} = 1$ et $\mu_{ij} \in [0,1]$,

||*|| est toute norme exprimant la distance entre les échantillons mesurées et le centre de classe,

$m$>1 est une indice pondérée, qui détermine le degré de flou de résultats de classification. Si $m$ tend vers 1, on tend vers des clusters "nets", c'est-à-dire que la matrice $M$, comportera uniquement des 0 et des 1. Par contre, plus $m$ est grand, plus les clusters sont flous, c'est-à-dire que leur fonction d'appartenance est très étendue.

Cette classification floue est réalisée grâce à une optimisation itérative de la fonction objective indiquée ci-dessus, avec la mise à jour de l'appartenance $\mu_{ij}$ et les centres des clusters $v_i$. Les étapes de l'algorithme des C-moyennes floues sont les suivantes:

1. on fixe arbitrairement un nombre de clusters $C$ , et une matrice d'appartenance $M$= [$\mu_{ij}$]
2. on calcule les centroïdes des classes :

$$v_i = \frac{\sum_{j=1}^{N} (\mu_{ij})^m x_j}{\sum_{j=1}^{N} (\mu_{ij})^m} \qquad 1.7$$

3. on réajuste la matrice d'appartenance, suivant la position des centroïdes :

$$\mu_{ij} = \left( \sum_{k=1}^{C} \left( \frac{\|x_j - v_i\|}{\|x_j - v_k\|} \right)^{2/(m-1)} \right)^{-1} \qquad 1.8$$

4. on calcule la fonction (1.8), si $\left| E^{(l+1)} - E^{l} \right| \leq \varepsilon$ , aller à l'étape 5, sinon, on retourne à l'étape 2.
5. on classe l'échantillon selon l'appartenance maximale : si $\mu_{ij} > \mu_{ik}$ , $x_j$ est affecté à la $i$-ème cluster, $k$=1, 2,..., $C$; $i \neq k$.

B. Ergen et al. [Ergen et al., 2012] ont proposé une méthode pour la détection de contours dans des images médicales en utilisant la transformée en ondelettes de Gabor et l'algorithme des C-moyennes floues. La transformée en ondelettes de Gabor est utilisée pour l'amélioration des contours de l'image d'entrée. Les contours sont révélés par l'algorithme des C-moyennes et la squelettisation morphologique. Pour ce faire, l'algorithme des C-moyennes effectue le partitionnement de l'ensemble de pixels de l'image suivant leurs dégrées d'appartenance

floues. Les auteurs ont testé leur méthode sur des images de scanner cérébral et abdominal et ont obtenu un taux de détection de 86.61%.

Z. Shi et al. ont proposé dans [Shi et al., 2013] une méthode de segmentation d'images médicales qui utilise un algorithme des C-moyennes amélioré et la transformation en ondelettes. Les auteurs ont effectué un segmentation multi-résolution sur l'image. En premiers temps, ils ont appliqué la méthode des C-moyennes pour la segmentation de l'image filtrée en basses fréquence de l'échelle grossière, et obtenir les centres de clusters et les étiquettes initialisées. Après la reconstruction de l'images originale à partir de l'image filtrée, ils obtient l'étiquette de chaque pixel et les centroïdes des clusters. Ils ont considéré les informations des pixels voisins. Ils définissent les caractéristiques de chaque pixel en fonction de sa propre niveau de gris et la moyenne des niveaux de gris des pixels voisins. Par la suite, un algorithme des C-moyennes améliorés est appliqué sur l'image reconstruite, de tel sorte que si deux pixels ont la même étiquette (initialisée précédemment), leur similarité augmente, et la distance de deux pixels est réduit.

### 1.2.7 Réseaux de neurones

L'idée de base derrière les réseaux de neurones est de s'inspirer des propriétés du cerveau pour construire des systèmes de calcul capable d'effectuer des opérations de classification par apprentissage. Leur principal avantage par rapport aux autres outils est leur capacité d'apprentissage et de généralisation de leurs connaissances à des entrées inconnues.

#### 1.2.7.1 Neurone formel

L'unité de traitement élémentaire dans un réseau de neurones est capable de faire seulement certaines opérations simples. Cette unité est souvent appelée *neurone formel* (voir Figure 1.8) pour leur similitude grossière avec les neurones du cerveau. Sa modélisation est inspirée du neurone biologique. Ces unités sont classées en 3 types de neurones [Cornuejols et Miclet, 2010]:

- Un neurone d'entrée ou, simplement, une entrée, est une unité chargée de transmettre une composante du vecteur *x* des données (en particulier, les données d'apprentissage pendant la phase d'apprentissage).

- Un neurone de sortie est une unité qui fournit une hypothèse d'apprentissage, par exemple dans un problème de classification, une décision sur la classe à laquelle est attribué *x*.
- Enfin, un neurone caché est un neurone qui n'est ni un neurone d'entrée, ni un neurone de sortie.

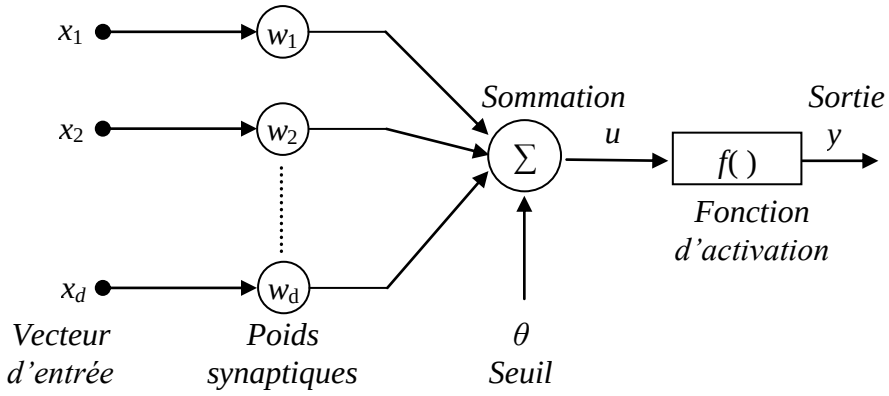



FIG. 1.8 – Modèle d'un neurone formel.

Le neurone formel recalcule son état *y* à chaque instant en fonction de l'influence globale du réseau. Ce calcule est donné par les équations suivantes :

$$u = \sum_{j=1}^{d} w_j x_j - \theta \qquad (1.6)$$

$$y = f(u) \qquad (1.7)$$

où :

*u* : est appelée la valeur d'activation du neurone.

$x_j$ : valeur de la *j-ème* entrée.

$w_j$ : intensité (poids ou paramètre) de la *j-ème* entrée.

$\theta$ : le seuil.

*y* : la sortie du neurone.

*f* ( ) : fonction d'activation ou de transfert.

Un Réseau de Neurones est un ensemble de neurones formels associés en couches et fonctionnant en parallèle. Tous les neurones d'une couche donnée ont la même fonction d'activation (des exemples de fonctions de transfert sont donnés dans le tableau 1.1). Chaque neurone peut transférer le résultat de son calcule aux neurones connectés à sa sortie. Ce processus est répété jusqu'aux neurones de sortie.

Les réseaux de neurones ressemblent au cerveau en deux points :

- la connaissance est acquise au travers d'un processus d'apprentissage.
- les poids des connections entre les neurones sont utilisés pour mémoriser la connaissance.

| Exemple de fonction d'activation | Expression : $f(x)=$ |
|---|---|
| Fonction sigmoïde entre [0,1] | $\frac{1}{1+e^{-x}}$ |
| Exponentielle | $e^{x}$ |
| Fonction tanh | $\frac{e^{2x}-1}{e^{2x}+1}$ |
| Fonction softmax | $\frac{e^{x}}{\sum_{i} e^{x_i}}$ |

TAB. 1.1 – Quelques fonctions de transfert usuelles. *x* est le vecteur d'entrée.

### 1.2.7.2 Apprentissage

La caractéristique la plus intéressante d'un réseau de neurones artificiels est sa capacité d'apprendre, c'est-à-dire de modifier les poids de ses connexions (les paramètres du réseau de neurones) en fonction des données d'apprentissage, de telle sorte qu'après un certain temps d'entraînement il ait acquis une faculté de généralisation [Cornuejols et Miclet, 2010]. La capacité de généralisation est la capacité du réseau de neurones à élargir les connaissances acquises après l'étape d'apprentissage à des données nouvelles et d'associer donc un vecteur d'entrée *x*, qui n'a pas fait l'objet d'un apprentissage, à une classe donnée [Barrat, 2009].

Dans l'apprentissage supervisé, on connaît la classe que le réseau de neurones doit associer au vecteur d'entrée *x*. L'apprentissage consiste à ajuster les poids des connexions du réseau de neurones afin de minimiser l'erreur entre la sortie désirée et la sortie réelle. Par contre, dans l'apprentissage non supervisé, aucune connaissance *a priori* n'est fournie sur la sortie désirée, il s'agit dans ce cas de déterminer les poids des connexions suivant le problème à résoudre.

Dans [Srivastava et al., 2013], V. Srivastava et al. présentent une méthode de reconnaissance d'images humaines en utilisant la classification non supervisée floue évolutive

et le réseau modulaire de neurones. Les auteurs utilisent l'algorithme des C-moyennes évolutif pour distribuer les images en un nombre de clusters, et l'Analyse en Composante Principale (ACP) pour l'extraction des caractéristiques. Chaque réseau modulaire correspond à un cluster d'images donné dont le nombre d'images égale au nombre de neurones de sortie du réseau correspondant. Le vecteur de caractéristiques d'image à reconnue est calculé et introduit dans les réseaux modulaires. Un intégrateur collecte les sorties maximales de chaque module. Ensuite, les valeurs supérieures à un seuil sont choisies et les valeurs de similarité entre les images correspondantes et l'image à reconnaître sont calculées. La reconnaissance est faite en fonction de similarité maximale.

L'expérimentation est faite sur deux ensembles de données standards : la base de données d'image de visage AT & T et la version 1 de la base de données d'images d'iris CASIA. Pour la première base de données, le teste est effectué sur un ensemble de 200 images qui contient 5 images par personne. Le taux de reconnaissance correcte est 99.0%. Pour la deuxième base de données, un taux de précision de 98.0% est obtenu sur 160 images d'iris.

W. Xiaobin et al. [Xiaobin et al., 2013] ont proposé un système de reconnaissance de plaques d'immatriculation de véhicules en utilisant le réseau de neurones à base d'algorithme génétique. 4 réseaux de neurones à 3 couches sont conçus en fonction de types de données de la plaque (caractère chinois, lettre latin, numéro et lettre-nombre). Chaque réseau de neurones a 512 entrées correspondant au nombre de pixels de l'image de caractère. Le nombre des nœuds de sortie des réseaux sont respectivement 51, 10, 25 et 34. L'entraînement est effectué en utilisant un algorithme génétique qui permet de trouver une combinaison optimale de poids de connexions et de seuils. Le système a obtenu les taux de précision suivants 94%, 97%, 91% et 89% pour les données de plaques d'immatriculation suivantes : caractère chinois, lettre latin, numéro et lettre-nombre.

### 1.2.8 Les séparateurs à vastes marges

Introduits dans [Vapnik, 1995], les séparateurs à vastes marges (ou SVM pour Support Vector Machine) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de classification et de régression (prédiction). Les SVMs ont rapidement été adoptés pour leur capacité à travailler avec des données de grandes dimensions, le faible nombre d'hyper-paramètres, le fait qu'ils soient bien fondés théoriquement, et leurs bons résultats en pratique.

Ces techniques présentent deux avantages :

- de très bonnes performances en généralisation (i.e. quand les exemples de test sont légèrement éloignés en terme de caractéristiques des individus de l'ensemble d'apprentissage).
- une capacité à résoudre des problèmes où l'ensemble d'apprentissage ne possède pas nécessairement de séparation linéaire dans l'espace des caractéristiques.

### 1.2.8.1 Classification binaire

Un SVM est un classifieur binaire. Un classifieur est dit binaire lorsque les données qu'il traite appartiennent à deux classes seulement. Dans ce cas, le problème de classification revient à trouver une surface de séparation qui sépare l'espace d'entrées en deux demi-espaces, chacun affecté à une classe.

Le principe théorique des SVMs comporte deux points fondamentaux :

1. la transformation non linéaire ( $\Phi$ ) des exemples de l'espace d'entrée $R^d$, dans lequel la séparation est non linéaire, vers un espace $E$ dit de redescription (espace de Hilbert) de plus grande dimension, dans lequel les données transformées deviennent linéairement séparables.
2. la détermination dans le nouvel espace un hyperplan qui permet de séparer les données d'apprentissage de manière optimale, c'est la notion de *marge maximale*.

La Figure 1.9 illustre un exemple de transformation. La séparation de la classe $w_1$ de la classe $w_2$ dans l'espace d'entrée $R^2$ nécessite une séparation non linéaire, comme le montre la frontière de décision de la Figure 1.9 à gauche. Si on prend la fonction polynomiale suivante $(x, y) \mapsto (z_1, z_2, z_3)$ tel que $z_1 = x_1^2, z_2 = x_2^2, z_3 = x_1 \cdot x_2$ qui fait passer d'un espace de dimension 2 à un espace de dimension 3, on obtient un problème en trois dimensions linéairement séparable, comme le montre la Figure 1.9 à droite. Nous constatons, à travers cette figure, que la séparation entre les deux classes est linéaire par rapport à chaque plan de l'espace de représentation.

La transformation non linéaire ( $\Phi$ ) est réalisée via une fonction noyau $\kappa$ facile à calculer, appelée *kernel trick* permettant d'effectuer les calculs nécessaires à l'algorithme dans l'espace de départ $R^d$ sans passer explicitement dans $E$. Ainsi l'espace de re-description reste virtuel.

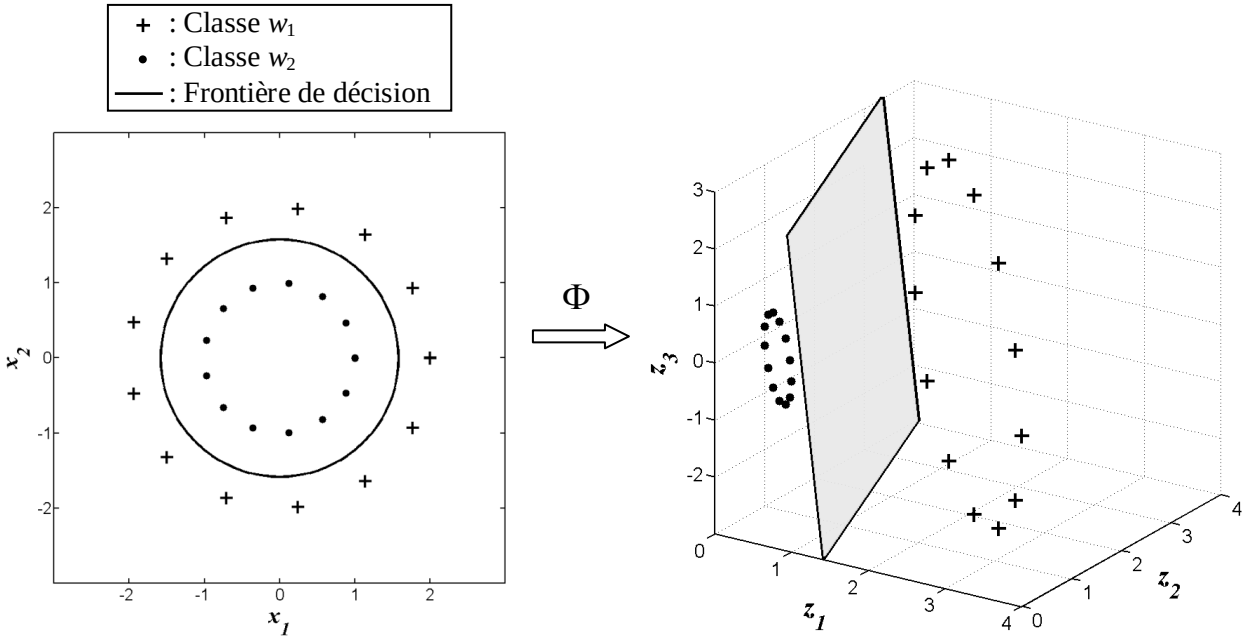



FIG. 1.9 – Séparation non linéaire, à gauche. Séparation linéaire grâce à la transformation de l'espace de représentation par une fonction polynomiale, à droite.

Le principe de l'optimisation des SVMs est de maximiser la marge entre les classes, par le choix d'un hyperplan qui va fournir la plus grande distance possible entre la frontière de décision et les plus proches exemples. Pour cela, l'algorithme d'apprentissage sélectionne judicieusement parmi les exemples d'apprentissage un certain nombre de points les plus proches de l'hyperplan. Ces points sont appelés *vecteurs de support*. Ils définissent la frontière de décision optimale (Figure 1.10).

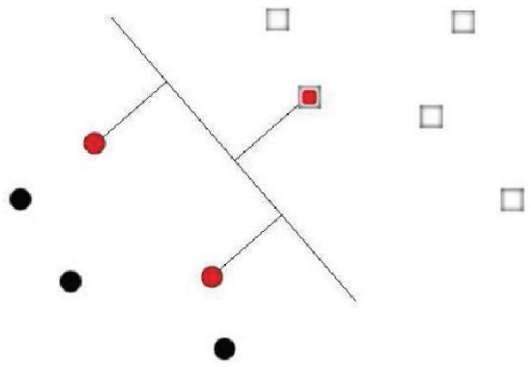
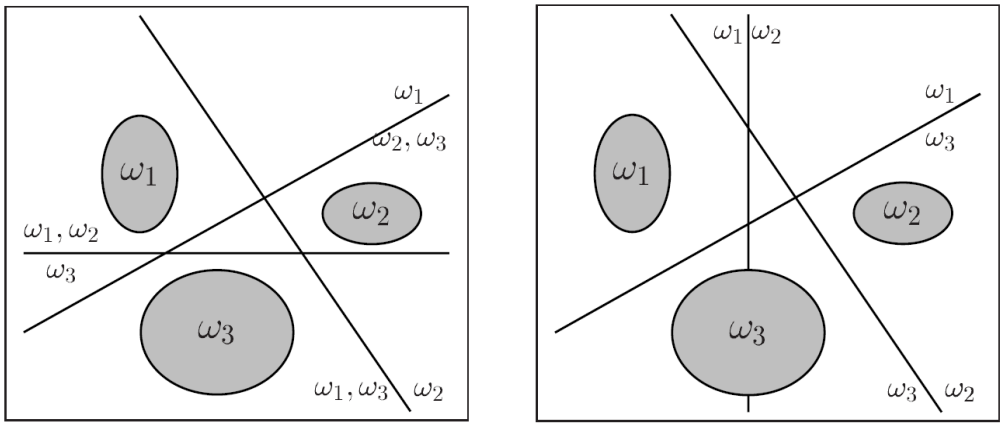


FIG. 1.10 – Hyperplan avec 3 vecteurs de support (en rouge). La position de la frontière maximise la distance entre ces points et leur projeté sur l'hyperplan.

## 1.2.8.2 Classification multi-classe

Dans le cas où l'on a un ensemble d'apprentissage représentatif de plus de deux classes ($C$>2), un ensemble de classifieurs binaires doivent être construits. Il existe deux façons de généraliser la classification binaire afin d'avoir une classification multi-classe [Nasreddine, 2010]:

1. **Un contre tous** : chaque classifieur binaire est appris pour distinguer une classe de toutes les autres (voir Figure 1.11(a)). Cette méthode nécessite de former $C$ classifieurs.




(a) (b)

FIG. 1.11 – Séparation linéaire à plus de deux classes, (a) séparation de chaque classe de toutes les autres : il y a $C$ hyperplans, (b) séparation de chaque couple de classes : il y a *C(C-1)/2* hyperplans.

2. **Un contre un** : chaque classifieur binaire est appris pour distinguer entre deux classes (Figure 1.11(b)). Cette méthode nécessite l'apprentissage de $C(C-1)/2$ classifieurs binaires mais avec un ensemble d'apprentissage plus petit comparativement à la méthode un contre tous.

Valuvanathorn et al. [Valuvanathorn et al., 2013] ont présenté un système de reconnaissance des visages qui utilise l'Analyse en Composantes Principales bidimensionnelle (ACP 2D) et le SVM. Le système utilise deux types de caractéristiques : globales (visage entier) et locales (œil gauche, œil droit, nez et bouche). Un modèle géométrique du visage est construit à l'aide d'un ensemble de visages annotés manuellement (données d'apprentissage), ce modèle est appliqué pour calculer les positions des composants

du visage. Les auteurs utilisent l'ACP 2D pour l'extraction des caractéristiques à partir de chaque composant. Ces caractéristiques vont êtres utilisées, par la suite, pour entraîner et tester un classifieur SVM. Pour déterminer le meilleur résultat de reconnaissance, les auteurs ont intégré toutes les composantes dont la décision suit la règle du vote majoritaire. Le système obtient un taux de reconnaissance correct de 97,83 % pour 115 images et 23 classes de visage.

Dans [Wen-ge, 2012], Wen-ge utilise des images de surveillance des rayonnement infrarouges dans les mines de charbon pour tester la capacité de reconnaissance du SVM, où deux catégories des images infrarouges doivent être reconnues : image de charbon brisé et image de bloc de charbon, et 4 types de caractéristiques sont utilisées pour entraîner un classifieur SVM.

## 1.2.9 Modèles de Markov cachés

Depuis quelques années, les modèles de Markov cachés (MMC ou HMM pour *Hidden Markov Model*) connaissent un essor important en reconnaissance des formes. Ce sont des méthodes de modélisation stochastiques parfaitement adaptées à la modélisation de séquences temporelles. Ces modèles peuvent servir également à modéliser des successions de mesures obtenues en progressant le long d'un axe, et sont donc aussi utilisés pour la reconnaissance de la parole et de l'écriture, manuscrite comme imprimée.

Les modèles de Markov cachés permettent de calculer la probabilité d'appartenance d'une forme à une classe. La forme observée et la classe sont supposées être des réalisations de variables aléatoires. Nous présentons dans ce qui suit les HMMs et leurs fonctionnement.

### 1.2.9.1 Modèles de Markov cachés unidimensionnels : HMM 1D

#### 1.2.9.1.1 Définitions

Une chaîne de Markov discrète d'ordre *n* est un processus stochastique discret $X = \{X_t / t = 1,...,T\}$ avec des variables aléatoires discrètes (les réalisations de ces variables sont appelées états $E = \{e_1, ..., e_N\}$), vérifiant la propriété de Markov : la probabilité de se trouver dans un état donné à un instant donné ne dépend que des *n* états visités avant. Si on considère $n = 1$, on a :

$$P(X_{t+1} = s_{t+1} \mid X_t = s_t, X_{t-1} = s_{t-1}, ..., X_1 = s_1) = P(X_{t+1} = s_{t+1} \mid X_t = s_t) \quad (1.8)$$

$\forall t \in [1, T]$ où $s_t$ représente l'état du processus $X$ dans l'instant $t$.

La probabilité $P(X_{t+1} = s_{t+1} \mid X_t = s_t)$ correspond à la probabilité de transition de l'état $s_t$ à l'instant $t$ vers l'état $s_{t+1}$ à l'instant $t+1$.

Elle est définie par :

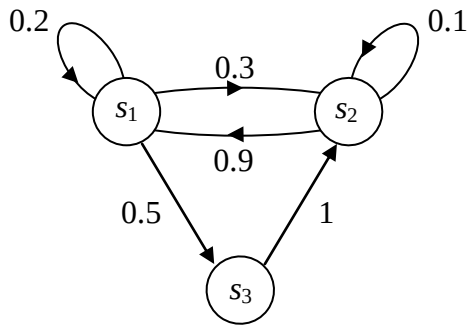
- la matrice des probabilités de transition $A = \{a_{ij}\}$, où $a_{ij}$ est la probabilité de passer de l'état $s_i$ à l'état $s_j$ ;
- Le vecteur des probabilités initiales des états $\Pi = \{\pi_i\}$ où $\pi_i$ est la probabilité de commencer dans l'état $i$ .

Une chaîne de Markov est homogène (dans le temps) si et seulement si les probabilités de transition ne dépendent pas du temps $t$ (les probabilités de transition sont stationnaires), c'est-à-dire que pour tout $t$ et $t'$ on a :

$$P(X_{t+1} = s_j \mid X_t = s_i) = P(X_{t'+1} = s_j \mid X_{t'} = s_i) \quad (1.9)$$

On note $a_{i,j}$ cette probabilité.

On peut représenter une chaîne de Markov graphiquement par un graphe, où les états sont représentés par des sommets et les transitions entre états sont représentées par des arêtes, elles sont pondérées par leurs probabilités. La Figure 1.12 présente la représentation graphique de la chaîne de Markov ($A$, $\Pi$).




$$\Pi = \begin{pmatrix} 0.4 \\ 0.2 \\ 0.4 \end{pmatrix} \qquad A = \begin{pmatrix} 0.2 & 0.3 & 0.5 \\ 0.9 & 0.1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

FIG. 1.12 – Représentation graphique de la chaîne de Markov ($\Pi$, $A$). (exemple extrait de [Aupetit, 2005]).

Un HMM est une chaîne de Markov stationnaire, dont les états n'est pas directement observable (d'où le nom de caché). Chaque état caché $s_t$ émet une observation ou symbole $v_t$ , c'est-à-dire qu'à chaque instant donné on observe une réalisation d'une variable aléatoire $Y_t$ suivant la densité de probabilité associée à l'état visité à cet instant.

Par conséquent, un HMM est un processus doublement stochastique, dans lequel les observations résultent d'une fonction aléatoire de l'état, et dont l'état change à chaque instant en fonction des probabilités de transition issues de l'état antérieur.

Un HMM peut être discret ou contenu. Si les observations prennent des valeurs discrètes, on dit que les observations sont des symboles d'un alphabet fini, et dans ce cas le HMM est discret. Par contre, si les observations prennent des valeurs continues, elles sont dites vecteurs de caractéristiques et le HMM est dit continu.

**Les HMMs *discrets*** du premier ordre est noté $\lambda = \{N, M, A, B, \Pi\}$ et se définit par :

- $N$ états du modèle, qui compose l'ensemble $S=\{s_1, s_2, ..., s_N\}$. L'état où se trouve le modèle à l'instant $t$ est noté $X_t \in S$ ;
- $M$ observations (symboles) possibles dans chaque état, qui compose l'ensemble $V=\{v_1,v_2,...,v_M\}$. $Y_t \in V$ est le symbole observé à l'instant $t$ ;
- Le vecteur $\Pi$ de probabilités initiales des états : $\Pi= \{\pi_i\}$ où $\pi_i$ est la probabilité que l'état du départ du modèle soit l'état $s_i$ : $P(X_1 = s_i)$ et $1 \le i \le N$ ;
- la matrice $A$ de probabilités de transition d'état à état : $A = \{a_{ij}\}$, où $a_{ij}$ est la probabilité que le modèle évolue de l'état $s_i$ vers l'état $s_j$ :
  $P(X_{t+1} = s_j \mid X_t = s_i)$ *et* $1 \le i, j \le N$ ;
- la matrice $B$ de distribution de probabilités de symboles observables dans chacun des états du modèle : $B = \{b_j(k)\}$, où $b_j(k)$ est la probabilité que l'on observe le symbole $v_k$ alors que le modèle se trouve dans l'état $s_j$ :
  $P(Y_t = v_k \mid X_t = s_j)$ *et* $1 \le j \le N$, $1 \le K \le M$ ;
- Un ou plusieurs états finaux.

**Les HMMs *continus***, très utilisés dans les systèmes de reconnaissance de l'écriture ou de la parole, diffèrent des HMMs discrets dans le sens où il est imposé aux $b_j$ une forme continue. Ainsi on ne parle plus de la probabilité d'émission d'un vecteur d'observations mais de la vraisemblance qu'un tel vecteur soit émis alors que le système est dans un état donné. Dans le cas classique, une loi normale multi-gaussienne est utilisée et les $b_j$ sont de la forme :

$$b_j(o_t) = \sum_{m=1}^{M} c_{jm} N(o_t, \mu_{jm}, U_{jm})$$

où $M$ est le nombre de gaussiennes, $c_{jm}$ est le coefficient de la gaussienne numéro $m$ associée à l'état $s_j$ et $\mu_{jm}$,$U_{jm}$ sont respectivement sa moyenne et sa matrice de covariance. D'autres systèmes cependant utilisent des lois différentes, comme par exemple la loi de Bernouilli.

## 1.2.9.1.2 Les fonctionnalités d'un HMM

Dans ce que suit nous présentons les fonctionnalités des HMMs en prenant comme exemple la reconnaissance d'écriture, sur lequel nous allons travailler dans le chapitre 4. Pour un système de reconnaissance d'écriture un HMM peut modéliser un mot pour lequel chacune de ses lettres est un état caché ou une chaîne de Markov à $N$ états cachés. Un mot est représenté sous forme d'une séquence de vecteurs de caractéristiques (en anglais, *frames*) qu'on appelle également observations. La séquence d'observations $O$ de longueur $T$ est définie comme : $O = o_1, o_2, ..., o_T$ où $o_t$ représente la $t$-ième observation. Le problème de la reconnaissance de mots peut être vu comme le calcul de :

$$\arg\max_i \{P(\lambda_i \mid O)\} \qquad (1.10)$$

où $\lambda_i$ est le HMM correspondant au *i-ème* mot du vocabulaire. Cette probabilité n'est pas calculable directement, mais la règle de Bayes nous donne :

$$P(\lambda_i \mid O) = \frac{P(O \mid \lambda_i) P(\lambda_i)}{P(O)} \qquad (1.11)$$

*P(O)* est constant et donc n'intervient pas dans le calcul du maximum. Les probabilités *a priori* $P(\lambda_i)$ pour tous les mots sont supposées égales. Par conséquent, le mot le plus probable dépend seulement de la vraisemblance $P(O \mid \lambda_i)$.

La figure 1.13 est un exemple extrait de [Menasri, 2008], qui montre comment un modèle à six états génère la séquence d'observation $O = o_1,..., o_6$ en passant par la séquence d'états $E$ = 1, 1, 2, 3, 3, 4. Dans cet exemple, l'état initial et l'état final sont non-émetteurs. La probabilité que le modèle $\lambda$ génère la séquence $O$ à travers la séquence d'états $E$ est donnée par : $P(O,E \mid \lambda) = \pi_1 b_1(o_1) a_{11} b_1(o_2) a_{12} b_2(o_3)...$ En pratique, la séquence d'observation $O$ est connue, mais pas la séquence d'états $E$. On dit que la séquence d'états est cachée. C'est la raison pour laquelle on parle de Modèles de Markov Cachés. Comme $E$ est inconnu, il est possible de calculer la séquence d'états la plus vraisemblable selon les données observées $O$

en évaluant la vraisemblance $P(O|\lambda)$ de la séquence d'observation $O$ par rapport au HMM $\lambda$. La vraisemblance est calculée comme la somme des vraisemblances sur toutes les séquences d'états cachés possibles.

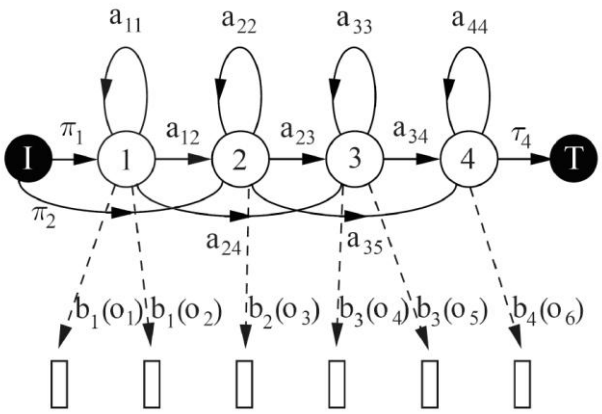



FIG. 1.13 – HMM : modèles génératifs. Dans cet exemple, l'état initial (*I*) et l'état final (*T*) sont non-émetteurs. (extraite de [Menasri, 2008])

L'utilisation des HMMs en reconnaissance des formes et particulièrement en reconnaissance de l'écriture, se décompose en trois problèmes: l'évaluation de la probabilité d'observation d'une séquence *O*, l'apprentissage à partir d'un ensemble d'échantillons (séquences d'observations) et la reconnaissance d'une séquence.

***Evaluation***

Etant donnés la séquence d'observations *O* et le HMM $\lambda$, on cherche à calculer $P(O|\lambda)$, c'est à dire la probabilité que la séquence *O* ait été engendrée par le HMM $\lambda$. Ce calcule est très coûteux en nombre d'opérations car il énumérerait tous les chemins possibles [Kriouile et al., 1990]. Une évaluation optimale de cette probabilité est obtenue par les fonctions *forward-backward* [Baum et Egon, 1967].

***Reconnaissance (décodage)***

La reconnaissance se ramène à l'idée de dévoiler les états cachés *E*, sans y avoir accès directement. Cette tâche peut être effectuée de deux façons différentes, soit dans le cas d'un modèle par classe, par recherche du modèle discriminant, soit dans le cas d'un seul modèle pour toutes les classes, par recherche du chemin optimal qui fournira la classe [Chen et al., 1995].

Dans le premier cas, un HMM $\lambda_i$ est construit pour chaque mot $w_i$ du vocabulaire. La tâche de reconnaissance consiste à résoudre l'équation (1.10) en utilisant l'équation (1.11) et en considérant que : $P(O \mid w_i) = P(O \mid \lambda_i)$. C'est-à-dire, on calcule la vraisemblance d'émission de la séquence d'observation par chaque modèle. La séquence *O* est affectée au mot du vocabulaire dont le modèle fournit la vraisemblance la plus importante.

Dans le deuxième cas, la reconnaissance consiste à déterminer dans le modèle $\lambda$ le chemin optimal correspondant à la séquence d'observation *O*, c'est-à-dire à trouver dans le modèle, la meilleure suite d'états qui maximise la probabilité $P(O \mid \lambda)$: elle est souvent obtenue grâce à l'algorithme de *Viterbi*.

***Apprentissage***

Cette étape est très importante car la qualité du décodage est étroitement liée à la qualité des modèles en sortie de l'apprentissage. Etant donné un ensemble de séquences d'observations qui représente l'échantillon d'apprentissage. Apprendre un HMM par des séquences d'observations c'est ajuster les paramètres du modèle de manière à maximiser un certain critère. Différents critères sont disponibles dans la littérature, on peut citer deux critères:

Dans l'apprentissage étiqueté de *Viterbi*, on dispose de deux informations : la séquence d'observations *O* et la séquence d'états cachés *E* qui a engendré la séquence précédente. Le critère que l'on cherche à maximiser est : $P(V = O, X = E \mid \lambda)$.

Le critère de maximum de vraisemblance consiste à trouver le modèle $\lambda^*$ maximisant la probabilité $P(V = O \mid \lambda)$ [Rabiner, 1989]. Les paramètres de ce modèle peuvent être déterminés automatiquement à travers l'algorithme de *Baum-Welch*. Une étude des algorithmes *forward-backward*, *Viterbi* et *Baum-Welch* a été réalisée et détaillée dans [Aupetit, 2005] et [Menasri, 2008].

## 1.2.9.1.3 Topologies des HMMs

Selon la topologie du réseau des états, il y a généralement deux types des HMMs : le *modèle ergodique* et le modèle de *Bakis*. Le Modèle ergodique est un modèle sans contraintes où toutes les transitions d'un état vers l'autres sont possibles (Figure 1.14(a)).

Le modèle *gauche-droite séquentielle* (Figure 1.14(b)), encore connu sous le nom de *Bakis*. Ce modèle suppose les états totalement ordonnés. Il impose des contraintes sur la

matrice des probabilités de transitions *A* de manière qu'il n'autorise de transitions d'un état que vers lui-même ou vers chacun des deux états suivants. Sachant cela, on impose aussi une contrainte sur la matrice Π de probabilités initiales des états : la séquence d'états doit commencer par l'état 1 et terminer par l'état *N*. Ce modèle est dit séquentiel car il représente bien l'évolution temporelle, et dit *gauche-droite* à cause de l'ordonnancement total des états.

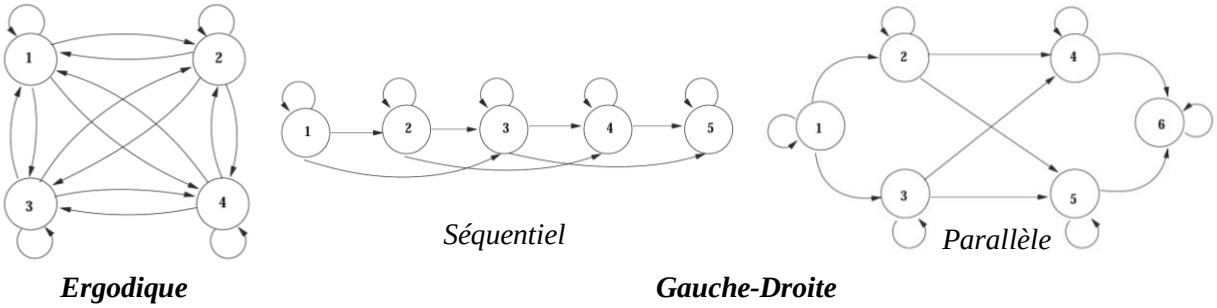



FIG. 1.14 – Quelques exemples de topologie de HMM (figure extraite de [Gosselin, 2000]).

Le modèle *gauche-droite séquentiel* est souvent utilisé pour reconnaître l'écriture car il est convenable à modéliser l'aspect séquentiel de l'écriture, ou les observations correspondent aux vecteurs de caractéristiques parcourus dans l'ordre de lecture. En plus il y a moins de paramètres à estimer dans la phase d'apprentissage (la distribution de probabilité initiale Π n'est pas nécessaire).

Une variante du modèle gauche-droite est le modèle gauche-droite *parallèle* (Figure 1.14(c)), utile, par exemple en reconnaissance de la parole, où il permet de prendre en compte la possibilité de multiples prononciations d'un même mot.

Le modèle *gauche-droite séquentiel* est souvent utilisé pour reconnaître l'écriture car il est convenable à modéliser l'aspect séquentiel de l'écriture, ou les observations correspondent aux symboles (vecteurs de caractéristiques) parcourus dans l'ordre de lecture. En plus il y a moins de paramètres à estimer dans la phase de l'apprentissage. Dans ce modèle, la distribution de probabilité initiale *Π* n'est pas nécessaire.

Les Modèles de Markov Cachés ont été utilisés dans différentes activités de reconnaissance. Par exemple, Mariusz Kubanek et al. [Kubanek et al., 2013] ont proposé un système biométrique pour la vérification des utilisateurs en fonction de la géométrie de la main et de l'empreinte palmaire. Pour analyser la forme de la main, les auteurs utilisent une plaque avec des clous placés sur la surface de celui-ci. Les clous doivent fournir les mêmes dispositions sur la plaque pour chacune des mains analysées, ce qui facilite le calcule des

caractéristiques géométriques de la main. Un ensemble de 19 caractéristiques géométriques sont utilisées, telles que : la largeur et la longueur de la main et de chaque doigt, la largeur du poignet. En plus, l'empreinte palmaire de la main est analysé pour détecter les lignes. L'image de la main est divisée en 144 sous-images carrées, et dans chaque sous-images qui contient des palm-prints, les auteurs calcule le moyen des angles formés entre l'axe horizontal et 3 points de chaque palm-print. Le système a donnée un Taux de Faut Rejet TFR= 2.60% et un Taux de Faux Acceptation TFA=1.40%, avec 500 utilisateurs.

Dans le deuxième Chapitre, Section 2.5, nous présenterons quelques travaux qui utilisent les modèles de Markov cachés pour la reconnaissance de l'écriture manuscrite arabe.

### 1.2.9.2 Modèles de Markov cachés Planaires : PHMM

Un modèle couramment utilisé pour la reconnaissance des formes est celle de PHMM, appelé aussi HMM pseudo 2D ou réseau de Markov. Les PHMMs sont des HMMs où la probabilité d'observation dans chaque état est donnée par un HMM secondaire (voir Figure 1.15) .

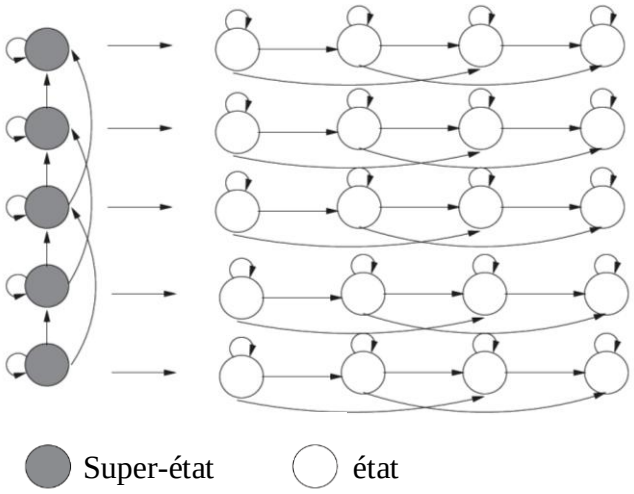



FIG. 1.15 – Architecture d'un PHMM.

En général, le HMM principal décrit l'image verticalement ligne par ligne, tandis que chaque HMM secondaire décrive une ligne site par site. En reconnaissance de l'écriture (Figure 1.16), une site horizontale peut correspondre à un ensemble de sous lignes pour lesquelles il y a les mêmes transitions noir blanc.

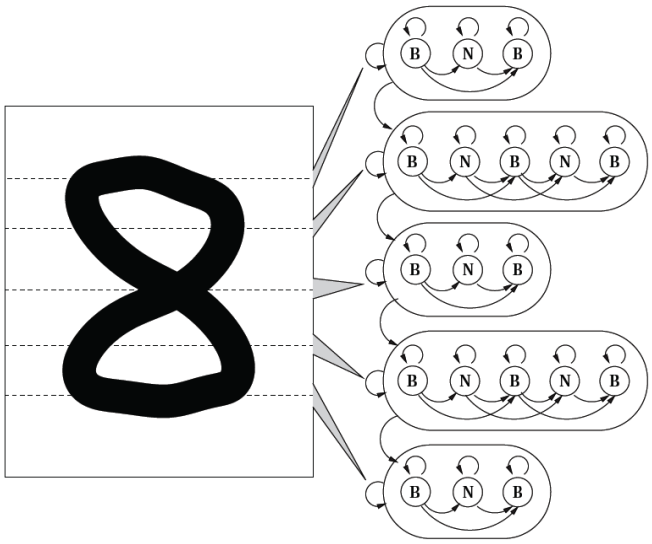



FIG. 1.16 – Exemple d'application des PHMM pour la reconnaissance de chiffres. (Exemple extrait de [Lemaitre, 2008]).

# Conclusion

Ce premier chapitre porte sur la reconnaissance des formes. Nous avons présenté dans la première partie les différentes fonctionnalités d'un système de reconnaissance des formes en exposant l'architecture générale d'un tel système, qui se compose de trois étapes principales : le prétraitement, l'extraction des caractéristiques, l'apprentissage et la classification. L'idée de base est d'apprendre des formes (des mesures sur ces formes) pour pouvoir les reconnaître par la suite. La forme reconnue est celle, parmi toutes les formes apprises, qui ressemble le plus à la forme inconnue.

Dans la deuxième partie, nous avons proposé un état de l'art des méthodes de classification automatiques. Nous nous sommes intéressés à l'approche statistique la plus couramment utilisées par les chercheurs en reconnaissance des formes. D'abord une définition du classifieur statistique est présentée. Par la suite, nous avons expliqué le principe du classifieur bayesien naïf qui fourni la limite théorique de la performance d'un classifieur. En pratique, diverses méthodes de classification ont été développées pour approcher cette limite théorique. Nous avons enfin présenté certaines de ces méthodes de classification et quelques travaux réalisés pour chaque méthode.

Ce chapitre nous permet de se familiariser avec le domaine de la reconnaissance des formes, quelle que soit le type de ces formes. Dans le chapitre suivant nous aborderons une

application typique de la reconnaissance des formes. Il s'agit de la reconnaissance d'écriture manuscrite arabe.

# Chapitre 2

# Reconnaissance de l'écriture manuscrite arabe : état de l'art

## Introduction

Ce chapitre porte sur la reconnaissance de l'écriture manuscrite. Cette tâche s'inscrit dans le domaine plus général de la reconnaissance des formes étudiée dans le Chapitre 1. La reconnaissance de l'écriture manuscrite est le processus de transformation d'un texte manuscrit en un texte électronique, c'est-à-dire la retranscription des mots contenus dans ce texte.

Selon les modes d'acquisition des textes manuscrits, une distinction est souvent effectuée entre deux approches de reconnaissance : hors-ligne et en-ligne. Dans le cas de la reconnaissance hors-ligne, l'acquisition du texte est réalisée après l'opération d'écriture, et se fait généralement à l'aide d'un scanner. Les données se présentent sous forme d'images numériques à deux dimensions. Dans le cas de l'écriture en-ligne, l'acquisition est, cette fois-ci, réalisée durant l'opération d'écriture. Le texte est saisi avec de stylets ou des doigts sur une surface sensible, fournissant ainsi un signal d'entrée temporel. Dans ce chapitre nous nous intéressons à la reconnaissance d'écriture hors-ligne, et plus particulièrement à l'écriture manuscrite arabe.

L'objectif de ce chapitre est de présenter un état de l'art de la reconnaissance de l'écriture manuscrite arabe. Nous verrons dans un premier temps les caractéristiques générale de l'écriture arabe. Nous aborderons par la suite les grandes étapes qui composent le processus de reconnaissance de l'écriture manuscrite, à savoir : les prétraitements, la segmentation, l'extraction des caractéristiques, la reconnaissance et le post-traitement. Nous présenterons pour chaque étape de traitement quelques travaux réalisés sur l'écriture manuscrite arabe. Enfin, ce chapitre se termine par une conclusion.

## 2.1 Caractéristiques générales de l'écriture arabe

Comme illustré sur la Figure 2.1, l'arabe s'écrit de la droite vers la gauche. L'écriture arabe est semi-cursive aussi bien dans sa forme imprimée que manuscrite. Les lettres dans un mot sont liées sur la ligne d'écriture dite ligne de base, à l'exception de six lettres. Si l'une de ces six lettres apparaît dans un mot, le mot est divisé en deux parties, chacune est appelée pseudo-mot. S'il y a plus d'une de ces lettres dans un mot, ce dernier se divise en plus de deux parties.

| Etiquette de lettre | isolé | fin | milieu | début | Etiquette de lettre | isolé | fin | milieu | début |
|---|---|---|---|---|---|---|---|---|---|
| Alif | ا | ـا | | | Daad | ض | ـض | ـضـ | ضـ |
| Baa | ب | ـب | ـبـ | بـ | Taad | ط | ـط | ـطـ | طـ |
| Taa | ت | ـت | ـتـ | تـ | Daa | ظ | ـظ | ـظـ | ظـ |
| Thaa | ث | ـث | ـثـ | ثـ | Ayn | ع | ـع | ـعـ | عـ |
| Jiim | ج | ـج | ـجـ | جـ | Ghayn | غ | ـغ | ـغـ | غـ |
| Haa | ح | ـح | ـحـ | حـ | Faa | ف | ـف | ـفـ | فـ |
| khaa | خ | ـخ | ـخـ | خـ | Qaaf | ق | ـق | ـقـ | قـ |
| Daal | د | ـد | | | Kaaf | ك | ـك | ـكـ | كـ |
| Thaal | ذ | ـذ | | | Laam | ل | ـل | ـلـ | لـ |
| Raa | ر | ـر | | | Miim | م | ـم | ـمـ | مـ |
| Zaay | ز | ـز | | | Nuun | ن | ـن | ـنـ | نـ |
| Siin | س | ـس | ـسـ | سـ | Haa | ه | ـه | ـهـ | هـ |
| Shiin | ش | ـش | ـشـ | شـ | Waaw | و | ـو | | |
| Saad | ص | ـص | ـصـ | صـ | Yaa | ي | ـي | ـيـ | يـ |

TAB. 2.1 – L'alphabet arabe : 22 lettres ont quatre formes différentes, et 6 lettres ont seulement deux formes : isolée et fin.

L'alphabet arabe contient 28 lettres insensibles à la casse ; le concept de majuscule et minuscule n'existe pas en écriture arabe. La forme d'une lettre diffère selon sa position : au début, au milieu ou à la fin du pseudo-mot. Par exemple : la lettre « ج (Jiim) » a quatre formes d'apparitions : isolée « ج » comme dans « خرج (sortir) », au début « جـ » comme dans « جبل (montagne) », au milieu « ـجـ » comme dans « مجموعة (ensemble) », à la fin « ـج » comme dans « برنامج (programme) ». Le tableau 2.1 présente les 28 lettres arabes avec leurs

différentes formes d'apparitions dans un mot. Les lettres qui ont juste deux formes d'apparitions ne peuvent pas être liées à la lettre suivante.

De plus, 15 lettres arabes incluent dans leur forme de base d'un à trois points diacritiques. Ces points peuvent se situer dans une des trois positions suivantes : au-dessus, au-dessous ou au milieu de la lettre. Plusieurs lettres peuvent avoir le même corps mais un nombre et /ou une position de points diacritiques différents. D'autre part, des signes diacritiques comme Hamza ou Madda sont également utilisés comme montré dans le tableau 2.2.

En outre, les lettres d'un pseudo-mot, sont ligaturées horizontalement sur la ligne de base. Des ligatures verticales de deux, trois ou même quatre lettres sont possibles. Ceci rend la segmentation à priori en lettres quasi-impossible. Finalement, des chevauchements verticaux peuvent se produire par l'intersection des pseudo-mots ou des mots pour quelques combinaisons de lettres. La Figure 2.1 montre quelques exemples de mots présentant des ligatures verticales et des chevauchements.

| Etiquette de lettre | isolé | fin | milieu | Début | Etiquette de lettre | isolé | fin |
|---|---|---|---|---|---|---|---|
| Hamza | ء | | | | Alifmaqsura + ٔ | ئ | ـئ |
| Alifmaqsura | | ى | | | LamAlif | لا | ـلا |
| Alif + ~ | آ | ـآ | | | LamAlif + ~ | لآ | ـلآ |
| Alif + ٕ | إ | ـإ | | | LamAlif + ٔ | لأ | ـلأ |
| Alif + ٔ | أ | ـأ | | | LamAlif + ٕ | لإ | ـلإ |
| Waaw + ٔ | ؤ | ـؤ | | | Tamabutra | ة | ـة |

TAB. 2.2 – Quelques lettres arabes spéciaux avec les suppléments ء Hamza et ~ Madda. La ligature commune لا LamAlif et le ة Tamarbuta sont présentés.

(a) (b) (c)

FIG. 2.1 – Exemple de mots arabes présentant quelques caractéristiques : (a) pseudo-mot لج présentant une ligature des deux lettres : ل Laam et ج Jiim, (b) chevauchements des trois lettres : ر Raa, و Waaw et ع Ayn, (c) ligature verticale des deux lettres : ل Laam et م Miim et chevauchement avec la lettre ا Alif.

Toutes ces caractéristiques typiques de l'écriture arabe constituent les problèmes majeurs liés au traitement de cette écriture. En effet, ces problèmes et bien d'autres (variabilité intra et inter-scripteurs, conditions de l'écriture, fusion de points diacritiques...) influencent le traitement et la reconnaissance de l'écriture manuscrite arabe.

## 2.2 Prétraitements des images

Une fois l'image numérisée, une séquence de prétraitements est appliquée. L'un des objectifs des prétraitements est de réduire la variabilité qui peut exister entre les écritures de différents scripteurs mais également au sein de l'écriture d'un même scripteur. Cette étape est la plus importante dans un système de reconnaissance de texte, car elle influe directement sur la fiabilité et l'efficacité de la segmentation, l'extraction des caractéristiques, et le processus de reconnaissance. Une grande variété de prétraitements existent dans la littérature, on peut citer les prétraitements suivants :

- binarisation ;
- élimination du bruit ;
- squelettisation et extraction de contour ;
- détection et correction de la ligne de base ;
- correction de l'inclinaison des lignes d'écriture ;
- correction de l'inclinaison des lettres.

### 2.2.1 Binarisation

Les images binaires sont composées de pixels qui ne peuvent prendre que la valeur 0 ou 1 (noir ou blanc). On peut donc leur appliquer des opérations logiques booléennes (vrai ou faux). Pour stocker chaque valeur on a besoin d'un seul bit, ce qui réduit fortement la taille des images lors de la constitution de grandes bases de données, et rend les traitements sur les images binaires particulièrement rapides.

Les systèmes de reconnaissance prennent en entrée des images numérisées, qui peuvent être binarisées, en niveaux de gris ou en couleurs. Lorsque les images sont en niveaux de gris ou en couleurs, les systèmes de reconnaissance incluent généralement une étape de binarisation de l'image, qui consiste a la transformer en une image en noir et blanc à fin de séparer l'information utile (l'écriture) du fond.

On peut distinguer deux types de binarisation d'images, globale et locale (adaptative). La binarisation globale utilise un seuil identique pour toute l'image, tel que les pixels dont la valeur est au-dessus du seuil sont considérés comme l'arrière-plan (blanc) et les autres comme l'information utile (appartenant aux mots écrits) (noir). Cette méthode convient pour les images qui sont très propres, c'est-à-dire lorsqu'il y a une bonne séparation entre les mots écrits et l'arrière-plan de l'image.

Il existe cependant des types d'images pour lesquels un seuillage global est inadapté, comme par exemple les images de documents historiques dont le fond est taché et considéré comme de l'information utile. Dans ce cas, un seuil de binarisation locale est calculé en fonction du voisinage du pixel traité.

Dans [El-etriby et Amin, 2010], El-etriby et al. ont traité des images de documents historiques manuscrits arabes, dont certains mots écrits en rouge. Dans ce cas, un seuillage globale sur l'image transformée en niveau de gris donne une image dégradée, en particulier pour les caractères écrits en rouge. Afin de remédier à ce problème, les auteurs ont proposé une méthode de seuillage hybride, qui repose sur les deux techniques connues : globales et adaptative. L'image de gris est divisée en grands blocs rectangulaires avec dimensions. Un seuil de binarisation globale est calculé pour chaque sous-image. Par la suite, chaque bloc est binarisé en fonction de son seuil pré-calculée. L'image binaire est construite en assemblant les blocs binarisés.

Toutefois, il est à remarquer que certains auteurs [Pechwitz et al., 2012] [Benouareth et al., 2008] ne passent pas par cette étape intermédiaire et extraient directement les primitives utiles à la reconnaissance à partir de l'image en niveau de gris.

### 2.2.2 Elimination du bruit

Du bruit peut apparaître sur les images du texte. On appelle bruit l'ensemble des éléments qui ne sont pas désirés dans une image, il peut se traduire par une distorsion sur l'image du caractère : irrégularité le long de contour, discontinuité ou trous (absence de pixels) dans le corps principal du caractère. Ce bruit peut être déjà présent sur le support avant la numérisation ou être introduit par la phase d'acquisition ou les premiers prétraitements. Il peut dégrader les performances de reconnaissance. D'autre part, les pages numérisées sont souvent des images dont le fond n'est pas uniforme (chèque, page de cahier avec lignes ou

quadrillage) ou des images de documents historiques dont le fond est dégradé (taches, pages rongées ou vieillies) par exemple.

Avant l'étape de reconnaissance, il est donc nécessaire de nettoyer l'image pour éliminer ces imperfections. En général, le nettoyage est effectué par un filtrage de l'image. L'idée de base est de convoluer un masque prédéfini avec l'image pour attribuer une valeur à un pixel en fonction des valeurs des pixels avoisinants.

Dans [Saleem et al., 2009], Saleem et al. ont présenté un algorithme pour détecter et éliminer les traits de soulignement dans l'image du texte manuscrit arabe. Leurs méthode détermine le profil de projection horizontale normalisée de l'intensité, ensuite, applique des masques de lissage sur chaque position du profil. Les pixels formant le trait sont trouvés dans le profil lissé en utilisant une recherche heuristique. Cet algorithme suppose que les traits de soulignement correspondent aux niveaux de gris minimaux, et que l'élimination de ces traits peut conduire à une découpage dans les corps principaux des caractères.

### 2.2.3 Représentations

L'un des objectifs des prétraitements est de réduire la quantité d'informations à traiter; l'image de texte acquise est parfois transformée en une représentation plus concise avant l'extraction de caractéristiques et la reconnaissance. Dans certains cas, les caractéristiques sont extraites directement à partir de l'image de texte. Dans d'autres cas, le squelette et / ou le contour de l'image du texte sont extraits avant l'extraction des caractéristiques.

#### 2.2.3.1 Squelettisation

La squelettisation est une opération qui permet de passer d'une image à sa représentation en "fil de fer". Un squelette a un pixel d'épaisseur. Il représente l'information indépendamment de l'épaisseur initiale de l'écriture. L'algorithme de squelettisation doit préserver la connexité des caractères et garder inchangés les courbes, les arcs et les points diacritiques qui sont des primitifs pertinents pour la discrimination des mots. La Figure 2.2(b) illustre un exemple d'extraction de squelette du mot de la Figure 2.2(a) par la méthode de Huang et al. [Huang et al., 2003]. Cette traitement a réduit le nombre de pixels dans l'image du mot de 4174 à 578 pixels. Dans la littérature, il existe seulement quelques algorithmes spécialement conçus pour l'extraction de squelette d'écriture manuscrite arabe. D'autres

travaux ont utilisé des algorithmes conçus initialement pour le Latin, afin d'extraire des squelettes de mots arabes.


(a) (b)

FIG. 2.2 – Exemple d'extraction de squelette : (a) mot d'origine ; (b) squelette extrait du (a).

Benouareth et al. utilisent dans [Benouareth et al., 2008] l'algorithme crée par Pavlidis [Pavlidis, 1982] pour la reconnaissance de mots arabes en utilisant des HMMs de durée d'état explicite. L'algorithme de Pavlidis détermine le squelette par des opérations locales. En même temps, les pixels sont étiquetés pour que l'image originale peut être reconstruise à partir de son squelette. L'extraction de squelette peut introduire quelques difficultés tels que des mauvaises localisations des caractéristiques et des ambiguïtés propres à chaque algorithme de squelettisation. La Figure 2.3 illustre quelques problèmes pour les algorithmes d'extraction de squelette.


(a) (b)

FIG. 2.3 – Illustration de quelques problèmes des algorithmes de squelettisation : (a) traits superflus dans le squelette ; (b) difficulté à représenter les boucles.

Un autre prétraitement possible est de normaliser l'épaisseur d'écriture à un nombre prédéterminé de pixels. Ce prétraitement est nécessaire lorsqu'il existe des différences dans l'épaisseur des mots du texte. Il permet de réduire la variabilité d'écriture des scripteurs et d'améliorer considérablement la qualité de la reconnaissance.

Dans [Azeem et Azeem, 2013], Abdel Azeem et Ahmed traitent la différence dans l'épaisseur de mots de la base IFN/ENIT. Ils ont normalisé l'épaisseur d'écriture en passant par deux étapes :

- squelettiser le mot, comme le montre la Figure 2.4(b), le mot original est plus mince qu'un autre pour le même mot illustré dans la Figure 2.4(a) Après la squelettisation,

l'épaisseur de deux mots devient le même (1 pixel) comme indiqué dans la Figure 2.4(c).

- dilater l'image binaire avec 3 pixels entourant le pixel amincie, comme indiqué dans la Figure 2.4(d). Ainsi, l'épaisseur des deux mots devient le même.

Une expérimentation effectuée dans ce travail indique que la normalisation de l'épaisseur de caractères rend le système insensible aux différents épaisseurs, et améliore sa performance globale.

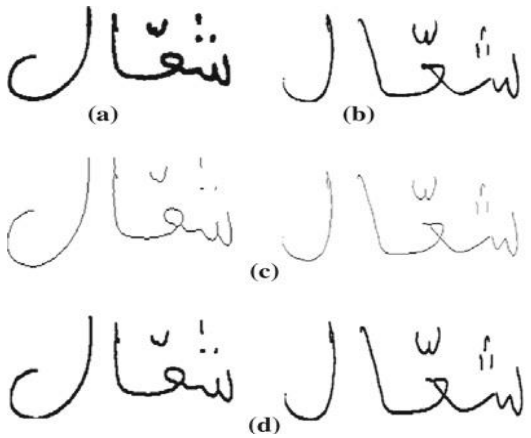
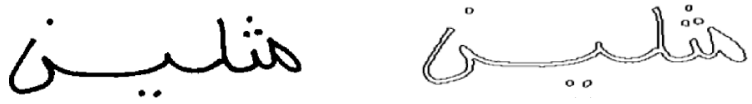



FIG. 2.4 – Les étapes utilisées dans le changement de l'épaisseur de mot, (a) et (b) : les mêmes mots originaux dont l'épaisseur est différent. (c): après la squelettisation, (d) : après la fixation de l'épaisseur.

## 2.2.3.2 Extraction de contour

Une alternative à la squelettisation est l'extraction du contour de l'image du texte. Cette approche évite les difficultés rencontrées dans le squelette, car dans cette représentation aucune information de forme n'est perdue. L'extraction du contour d'une image de mot/caractère est plus facile et plus rapide par rapport à l'extraction du squelette. En outre, nous pouvons reconstruire l'image originale à partir de son contour. Un contour est une représentation compressée d'une image. La Figure 2.5(b), extraite de [Parvez et Mahmoud, 2013a]), représente un contour extrait du mot de 2.5(a).



(a) (b)

FIG. 2.5 – Exemple d'extraction de contour : (a) mot d'origine ; (b) contour extrait de (a).

La méthode la plus populaire pour représenter le contour est le code de Freeman [Freeman, 1961]. Cette méthode a pour but de coder le contour d'un objet par :

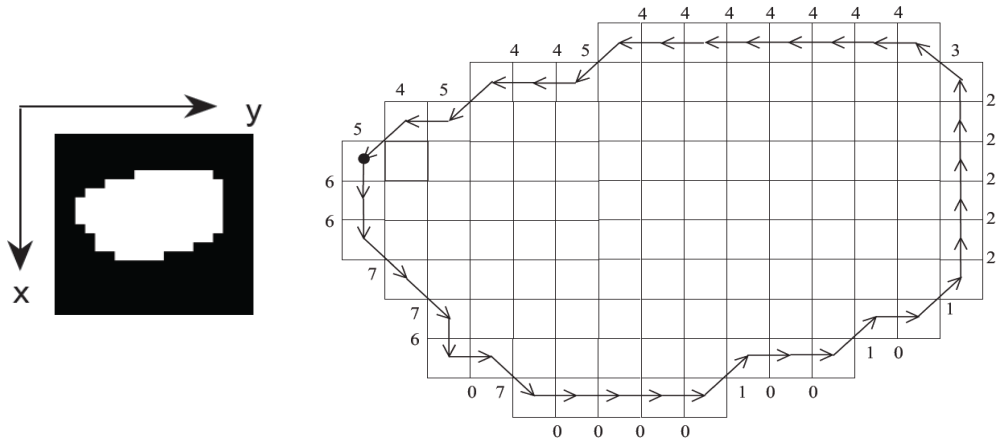
- la position absolue d'un point de départ ;
- une chaîne de codants donnant la position relative du point suivant du contour de l'objet selon une des représentations présentées dans la Figure 2.6 (codage utilisant 4 ou 8 directions).

| | | |
|---|---|---|
| 3 | 2 | 1 |
| 4 | *X* | 0 |
| 5 | 6 | 7 |

| | | |
|---|---|---|
| | 1 | |
| 2 | *X* | 0 |
| | 3 | |

FIG. 2.6 – Code de Freeman à 4 ou 8 directions. *X* désigne le pixel courant et les chiffres correspondent aux 4 ou 8 directions possibles pour lesquelles *X* a un voisin appartenant au contour de l'objet.

La Figure 2.7(a) présente un objet *A* blanc sur fond noir. La méthode d'extraction du code de Freeman de l'objet *A* est illustrée dans la Figure 2.7(b). On sélectionne un point appartenant à la frontière interne de *A* (par exemple le premier pixel rencontré par balayage électronique), on mémorise la position de ce premier point puis on cherche son plus proche voisin appartenant à *A* selon un sens de rotation donné, on mémorise sa position. On réitère ensuite cette dernière opération jusqu'à revenir au point de départ. Ainsi, de proche en proche, on reconstitue la forme de l'objet *A* en donnant le codage de Freeman de la frontière de *A* par la chaine 667760700001001012222234444444544545.




(a) (b)

FIG. 2.7 – Exemple du codage de contour par le code de Freeman (a) objet digital ; (b) code de Freeman à huit directions de contour de (a).

Le contour a été largement utilisé dans les systèmes de reconnaissance de mots arabe. Il a été utilisé pour l'estimation de la ligne de base [Boukerma et Farah, 2010], la détection de l'inclinaison de l'écriture, la segmentation en caractères [Parvez et Mahmoud, 2013b], la segmentation en lignes [Bukhari et al., 2009] et l'extraction des caractéristiques [Kessentini et al., 2010].

## 2.2.4 Détection et correction de la ligne de base

La ligne de base est une ligne virtuelle sur laquelle les caractères de texte sont alignés et / ou sont joints. En effet, elle représente pour l'écriture ainsi que pour la lecture, une référence de positionnement verticale de chaque caractère et ligatures ainsi que pour la distinction des graphèmes qu'ils composent. La Figure 2.8 présente la ligne de base d'une ligne de texte manuscrit arabe.

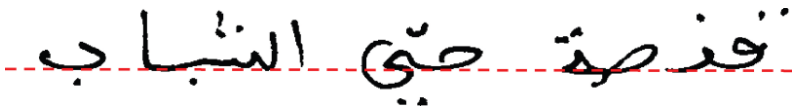


FIG. 2.8 – Illustration de la ligne de base (la ligne en pointillé) pour une ligne de texte arabe manuscrit (figure extraite de [Parvez et Mahmoud, 2013b]).

L'écriture manuscrite arabe est naturellement cursive. Une des principales difficultés dans la reconnaissance de cette écriture est la variabilité de la ligne de base considérée non seulement à travers les différents scripteurs, mais aussi entre les mots et les caractères écrits par un même scripteur dans la même ligne. Ainsi, l'estimation précise de la ligne de base est cruciale pour la performance de l'extraction de caractéristiques et la reconnaissance de l'écriture manuscrite arabe [Natarajan et al., 2011].

La ligne de base contient des informations précieuses sur l'orientation du texte et de l'emplacement des points de connexion entre les caractères. En outre, la position des ascendants / descendants et l'emplacement des signes diacritiques est déterminée par la ligne de base [Boukerma et Farah, 2010]. Cette ligne peut être utilisée pour la correction de l'inclinaison [Chergui et al., 2012], la segmentation du texte en mots ou caractères [Jamal et Jamal, 2013], ou biens le calcule des caractéristiques [Likforman-Sulem et al., 2012].

Nous pouvons citer quatre problèmes relatives à l'estimation de la ligne de base [Pechwitz et al., 2012] (voir Figure 2.9):

- des mots très courts, qui se composent de nombreux pseudo-mots (Parts of Arabic Words ou PAWs) (Figure 2.9(a)). Ces deux caractéristiques sont très défavorables à une estimation réussite d'une ligne de base.
- des mots longues (ou groupes de mots) avec un saut en ligne de base, qui résultent de la non-conformité avec les règles de l'écriture (Figure 2.9(b)). Les mots sont constitués de plusieurs longues PAWs (ou mots) qui sont composés de nombreux caractères et ainsi deviennent longues.
- des mots qui ont des descendants parmi les caractères qui le composent (Figure 2.9(c)). Certains scripteurs ont l'habitude d'écrire le dernier caractère d'un mot ou d'un PAW avec un grand essor. Ainsi, s'il y a beaucoup de descendants dans un mot avec des portions horizontales suffisamment longues, ils deviennent très importants et peuvent affecter négativement le processus d'estimation de la ligne de base.
- d'autres constellations défavorables montrées dans la Figure 2.9(d). Le premier exemple est l'insouciance du scripteur, deux PAWs sont reliés entre eux de manière très défavorable. Le deuxième est écrire même un mot très court avec l'utilisation de ligatures.

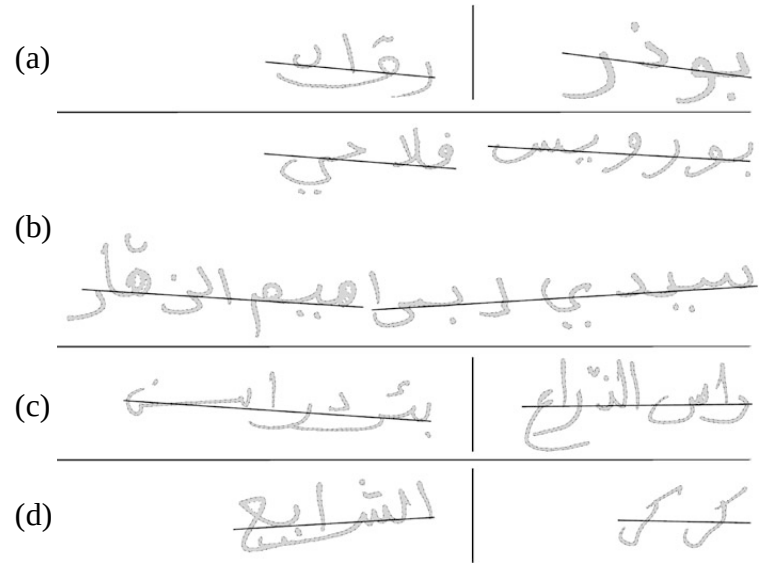



FIG. 2.9 – Des exemples (extraits de la base IFN/ENIT) de mots manuscrits où l'estimation de la ligne de base semble être un véritable défi. La position optimale de la ligne de base est donné pour chaque mot par une ligne continue.

Plusieurs méthodes sont proposées dans la littérature pour la détection de la ligne de base dans un texte arabe. L'approche standard est la projection horizontale [Kessentini et al., 2010]

[Lawgali et al.,2011] [Likforman-Sulem et al., 2012]. Cette approche est basée sur l'analyse de l'histogramme de projection horizontale des pixels noires de l'image binaire sur un axe vertical. Le pic maximal de l'histogramme indique la position de la ligne de base (voir Figure 2.10).

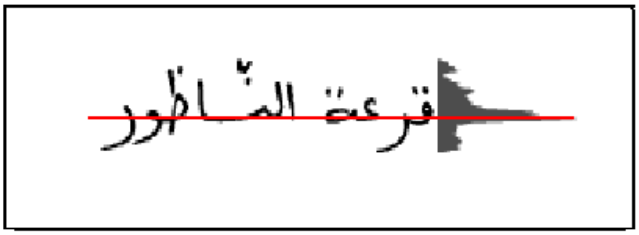


FIG. 2.10 – Estimation de la ligne de base par la méthode de projection horizontale (figure extraite de [Al-Shatnawi et Omar, 2009a]).

C'est une technique facile à implémenter et est la plus utilisée pour l'écriture arabe. Elle fonctionne bien avec le texte imprimé (où les lignes sont droites), mais elle présente des difficultés pour le manuscrit, où la variabilité de l'inclinaison d'écriture au sein d'un même mots est très courante. En outre, l'existence d'un grand nombre de signes diacritiques et des descendants avec des portions horizontales suffisamment longues peuvent affecter considérablement la performance de cette technique en créant des pics parasites dans l'histogramme.

Par conséquent, des techniques plus sophistiquées sont développées pour localiser les lignes de base dans les textes manuscrits arabes. Dans ce qui suit, nous présentons quelques-unes de ces méthodes.

Dans [Akhateeb et al., 2008], Alkhateeb et al. ont proposé une modification de l'approche basée sur la projection horizontale. Ils ont divisé l'image de mot par une ligne médiane, et ils ont supposé que la ligne de base arabe se trouve toujours au-dessous de la ligne médiane de l'image, ce qui implique qu'une projection horizontale uniquement sur la moitié inférieure de l'image donne de meilleurs résultats qu'une projection de l'ensemble de l'image. Toutefois, cette modification peut fonctionner correctement seulement si la ligne de base est relativement droite.

Boubaker et al. ont proposé dans [Boubaker et al., 2009] un algorithme pour la détection de la ligne de base de courtes séquences de mots arabes en ligne et hors ligne. Le processus est composé de deux étapes: en premier temps, il détecte sur l'image du squelette des regroupements de points des voisins alignés. La détection de la ligne de base approximée

consiste à rechercher le regroupement le plus nombreux parmi ceux qui sont constitués. Ensuite, certaines conditions topologiques spécifique à la langue arabe sont utilisées pour évaluer la pertinence de la ligne de base estimée, puis à traiter l'erreur de la détection.

Les auteurs ont signalé un taux de détection correcte de la ligne de base de 97.9% pour un ensemble de 1000 échantillons de la base IFN/ENIT. Dans la reconnaissance de texte hors-ligne, généralement nous traitons des longues séquences de mots. Ainsi, l'algorithme proposé peut être utile dans les applications en ligne, où les phrases courtes sont communes.

Un mot arabe peut se composer de deux ou plusieurs pseudo-mots (PAWs), la distribution des pseudo-mots peut créer de différentes angles d'inclinaison au sein du même mot. D'où la nécessité de concevoir un algorithme dédié à l'écriture manuscrite arabe, et capable de détecter la ligne de base la plus adéquate à la variabilité de l'inclinaison d'écriture des pseudo-mots. Il existe dans la littérature quelques travaux qui visent à remédier à ce problème.

Dans [Boukerma et Farah, 2010], Boukerma et al. ont présenté un algorithme de détection de la ligne de base basé sur l'extraction des pseudo-mots. Cet algorithme présente l'avantage de la prise en compte des pseudo-mots plutôt que les mots complets, comme étant l'entité élémentaire du traitement lors de la détection de la ligne de base.

Ils ont divisé l'image du squelette de mots en trois bandes horizontales d'hauteur identique, puis, ils ont éliminé les points diacritiques et localisé les pseudo-mots. La deuxième bande est utilisée comme une première approximation de la bande de base de chaque pseudo-mot, où ils ont détecté des points primitifs (les points d'embranchement, de croisement et les points les plus bas des boucles). Ensuite, la bande de base finale est estimée pour chaque pseudo-mot en se basant sur ces points primitifs.

Pour estimer la ligne de base, deux types de points supports les plus pertinents sur le contour sont sélectionnés, ces point sont situés dans la bande de base du pseudo-mots : les minimums locaux du contour inférieur et les points des boucles les plus bas localisés. La ligne de base du mot est construite par une interpolation linéaire à partir de points supports sélectionnés de tous les pseudo-mots.

L'algorithme proposé est testé sur les premières 2240 images de l'ensemble-*a* de la base IFN/ENIT. Les auteurs ont évalué la précision de l'estimation de leur algorithme en lui comparant à l'annotation de la ligne de base des mêmes images de la base IFN/ENIT, et ils ont obtenu un taux de bonne détection de 87.19% (avec 15 pixels de décalage).

Natarajan et al. ont introduit dans [Natarajan et al., 2011] une approche à deux étapes : en premier temps, la ligne de texte est segmentée en ensemble de composantes connexes. Les point et les signes diacritiques sont éliminés, et les composantes qui se sont superposées horizontalement ou sont approximées sont fusionnées. Ensuite, une ligne de base est calculée pour chaque composante en utilisant deux méthodes différentes (en considérant chaque image d'une composante connexe comme une matrice de pixels):

- Maximum de projection : consiste à traverser la matrice verticalement, la ligne de la matrice ayant le maximum nombre de pixels de texte représente la ligne de base de la composante.
- Centre de gravité : cette méthode consiste à balayer la matrice de pixels horizontalement et calculer pour chaque ligne de la matrice le centre de gravité verticale de pixels de texte, qui va être utilisé ensuite pour estimer la ligne de base de la composante.

L'algorithme est testé sur de 200 images de lignes de texte manuscrit arabe, les estimations obtenues ont été comparées avec une annotation sur les mêmes images. Les taux d'erreur de détection sont 20±24 et 12±18 pour les méthodes "Maximum de projection" et "Centre de gravité", respectivement.

Slimane et al. ont proposé dans [Slimane et al., 2012] une approche basée sur un modèle stochastique capable de proposer des positions probables pour la ligne de base. Cet approche a utilisé trois modèles de mélanges Gaussiens GMMs entraînés sur des caractéristiques locales des caractères imprimés. Chacun de ces GMMs est utilisé pour estimer une distribution de probabilité de chacune des positions de la ligne de base (bas, milieu et haut). A partir de ces distributions, une distribution globale a été calculée. Par la suite, la ligne de base correspondante est affinée en utilisant la projection horizontale classique.

Dans [Parvez et Mahmoud, 2013b], Parvez et Mahmoud proposent d'utiliser uniquement les composantes de base pour l'estimation de la ligne de base. Ils définissent une composante de base comme un pseudo-mot occupant une zone supérieure à la zone moyenne de tous les pseudo-mots du texte. Ainsi, la ligne de base estimée pour une séquence de mots ne coïncide pas nécessairement avec tous les lignes de base de pseudo-mots qui constituent cette séquence.

Dans un premier temps, une estimation initiale de la ligne de base est obtenue par une régression linéaire en utilisant les centres de gravité de toutes les composantes de base. Ensuite, pour raffiner la ligne estimée, les auteurs appliquent une autre régression linéaire en

utilisant seulement les centres de gravité des composantes de base où les limites de leurs boites sont croisés avec la ligne de base initiale. Cette régression donne la ligne de base du texte.

Ils existe des travaux qui extraient deux lignes de base sur l'image de mot: ligne haute et ligne basse (voir Figure 2.11). Ces lignes permettent de séparer l'image du mot en trois zones: une zone médiane ou bande de base qui correspond à la zone centrale de l'écriture, une zone supérieure où se trouvent les hampes des caractères (ascendantes) et les points diacritiques hauts, et une zone inférieure où se trouvent les jambages des caractères (descendants) et les points diacritiques bas. Certains travaux utilisent ces lignes de base pour l'extraction des caractéristiques [Al-Hajj et al., 2005] [Likforman-Sulem et al., 2012]. D'autres les utilisent pour détecter des succession des caractères descendants qui se touchent [Boukerma et Farah, 2012].

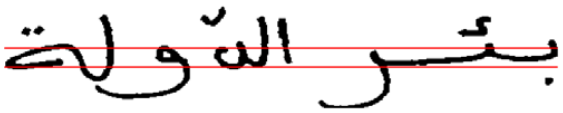


FIG. 2.11 – Les lignes de base haute et basse.

## 2.2.5 Normalisations

L'objectif des normalisations d'images est de réduire la variabilité de l'écriture lors de l'apprentissage et de la reconnaissance. Nous présentons deux techniques de normalisation:

- Correction de l'inclinaison des lignes.
- Correction de l'inclinaison des lettres.

## 2.2.5.1 Correction de l'inclinaison des lignes

La correction de l'inclinaison (en anglais *skew*) est une étape importante dans la reconnaissance du texte arabe, car elle a un effet direct sur la fiabilité et l'efficacité de la détection de ligne de base, la segmentation et l'extraction des caractéristiques [Al-Shatnawi et Omar, 2009b]. L'inclinaison est généralement introduite dans l'image lors d'un défaut d'orientation du document pendant l'acquisition, ou d'une écriture imprécise. Le processus de la correction de l'inclinaison consiste à détecter la déviation horizontale ou verticale de

l’angle d’orientation du document [Sarfraz et al., 2007], puis à redresser horizontalement les lignes d’écriture inclinées.

Dans une image de document, il est possible d’avoir trois catégories d’inclinaison différentes : (1) inclinaison au niveau de paragraphe ou de document, qui est généralement introduite dans l’image lors d’un défaut d’orientation du document pendant l’acquisition, (2) inclinaison au niveau de ligne, (3) inclinaison au niveau de mot. Les deux dernières se produisent souvent en raison de la nature de l'écriture arabe ou d’une écriture imprécise. La Figure 2.12 présente un exemples de ligne de mots nécessitant la détection et la correction de l’inclinaison au niveau de linge.

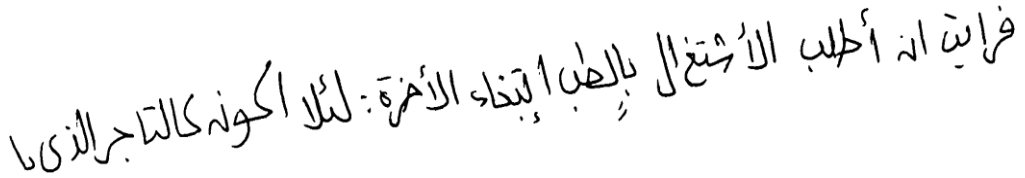


FIG. 2.12 – Exemples de ligne de mots manuscrits nécessitant la correction de l‘inclinaison au niveau de linge (extrait de la base de textes manuscrits arabes KHATT).

Plusieurs méthodes de correction de l’inclinaison existent à ce jour. Certaines méthodes utilisent un calcule de ligne de base non horizontale puis redressent l’image (par une rotation sur le centre de gravité) jusqu’à ce que les lignes soient horizontales.

Dans [Al-Shatnawi et Omar, 2009b], Al-Shatnawi et Omar ont traité l'inclinaison dans les images due à un défaut d’orientation du document pendant l’acquisition. L'algorithme proposé inscrit l'image de texte dans un rectangle (en général, un polygone avec au moins deux dimensions), en tenant en compte les pixels les plus éloignés dans les quatre directions. En supposant que le rectangle est construit d'un matériau de densité uniforme, les quatre points sont utilisés pour calculer le centre de gravité du texte. La ligne joignant le centre de gravité et l'origine représente la ligne de base qui donne l'angle et l'orientation de l'inclinaison. Pour corriger cette inclinaison, il suffit d’appliquer sur l’image une rotation d’angle calculé. L'algorithme proposé a été testé sur 150 documents arabes numérisés. Les auteurs ont rapporté une précision de 87% sur 150 images de documents arabes différents.

Dans [El-etriby et Amin, 2010], El-etriby et Amin ont traité une image de document historique manuscrit arabe constituée de quatre blocs de texte, chaque bloc a un angle d'inclinaison différent. En calculant le centre de gravité, l'algorithme présenté détecte pour

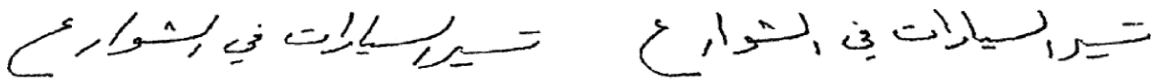
chaque composante connexe une ligne de base, l'angle d'inclinaison obtenue est utilisée pour effectuer une rotation sur l'image de la composante connexe.

D'autres méthodes sont basées sur le profil de la projection horizontale [Sarfraz et al., 2007]. Généralement, le profil de la projection horizontale est un histogramme de nombre de pixels noirs comptés en parcourant l'image selon des directions proches de l'horizontale (petites variations d'angle). Les creux et les pics sont calculées, alors que pour un texte ayant des lignes horizontales la projection a des pics aux positions de lignes du texte et des creux à des endroits entre lignes. La différence entre le creux et le pic est calculée à chaque angle et la différence maximale correspond à l'angle de l'inclinaison [Al-Shatnawi et Omar, 2009b]. La méthode de projection horizontale est robuste et facile à implémenter, mais elle est performante uniquement avec des lignes droites et des mots longues.

## 2.2.5.2 Correction de l'inclinaison des lettres

Certains scripteurs écrivent les caractères de façon inclinée vers la droite ou vers la gauche par rapport à l'axe vertical. Cette inclinaison est également appelée *slant*. Il convient de corriger cette inclinaison pour diminuer la variabilité de l'écriture de scripteurs et améliorer la qualité de la segmentation des mots en caractères.

En général, les algorithmes de correction de l'inclinaison des caractères sont fondés sur la détection des traits quasi-verticaux (NVSs). On estime leur inclinaison moyenne par rapport à la verticale et avec ce même angle on corrige l'inclinaison en appliquant une transformation de cisaillement sur l'image du mot [Parvez et Mahmoud, 2013b]. La Figure 2.13 illustre le concept de la correction de l'inclinaison.



(a) (b)

FIG. 2.13 – Illustration de la correction de l'inclinaison : (a) l'image originale du texte ; (b) l'image du texte après la correction de l'inclinaison.

Dans [Ziaratban et Faez, 2009], Ziaratban et Faez proposent un algorithme pour l'estimation et la correction de l'inclinaison non-uniforme de mots manuscrit arabe/farsi. L'algorithme supprime les points et les signes diacritiques et détecte l'inclinaison globale de l'image du squelette. Les auteurs ont utilisé des filtres Prewitt dans sept directions pour

obtenir dans le mot des traits quasi-verticaux (NVSs). Ensuite, l'inclinaison globale est estimée et corrigée à partir de tous les traits obtenus. Dans la deuxième étape, l'algorithme détecte les inclinaisons restantes des traits initialement verticaux, qui ont été écrits comme des traits quasiverticaux. Cet algorithme a été testé par les auteurs sur l'ensemble-a de la base IFN/ENIT.

Dans [Parvez et Mahmoud, 2013b], Parvez et Mahmoud ont extrait d'abord le contour du corps principal du mot à base d'approximation polygonale. Pour estimer l'inclinaison d'un mot, les auteurs ont identifié les traits quasi-verticaux (NVSs): l'angle d'inclinaison de chaque arête du polygone est mappé à la direction la plus proche parmi les directions standards (0, 30, 60, 90, 120, 150, 180, 210, 240, 270, 330). Chaque arête mappée à 90° ou à 270° est identifiée comme un NVS. Par la suite, l'inclinaison associée avec un NVS est estimée en fonction de la direction mappée, l'angle et la longueur d'arête. Tous les inclinaisons des NVSs servent pour estimer l'angle d'inclinaison globale du mot. Une fois cet angle est calculé, l'inclinaison de pseudo-mot est corrigée en appliquant une transformation de cisaillement sur chaque pixels de l'image (voir Figure 2.14).

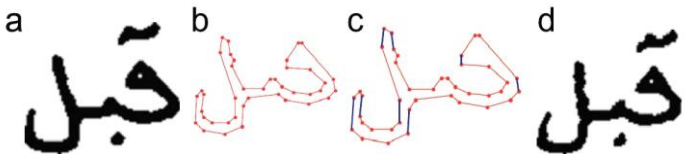



FIG. 2.14 – Illustration de la méthode de la correction de l'inclinaison, (a) l'image originale du mot; (b) l'approximation polygonale du corps principal du mot ; (c) les traits quasi-verticaux marqués en gras et (d) l'image du mot sans inclinaison.

## 2.3 Segmentation

De manière générale, le but d'un système de reconnaissance d'écriture est, pour une image scannée de texte en entrée de retranscrire intégralement le texte écrit. Afin d'être traitées, les images du texte sont segmentées, en le divisant en unités simples : images du texte en lignes, les lignes en mots, et les mots en caractères ou graphèmes (morceaux de caractères). C'est l'une des étapes les plus critiques et difficiles. Elle présente un défi dans les systèmes de reconnaissance d'écriture plus que le processus de reconnaissance lui-même.

L'emploi ou non de l'étape de segmentation en caractères ou en graphèmes permet de distinguer deux approches principales de reconnaissance de mots cursives : l'approche globale ou holistique et l'approche analytique ou locale.

**L'approches holistique** (ou globale) considère le mot dans son ensemble sans chercher à identifier chacune des lettres qui le compose. Des caractéristiques sont extraites sur le mot entier: boucles, ascendants, descendants, profils haut/bas, vallées, longueur, points terminaux, points de croisements, ainsi que d'autres primitives locales et globales. L'approche globale est plus simple que l'approche analytique, car elle ne nécessite pas de segmentation. Cependant, cet approche présente l'inconvénient de subir la variabilité des mots, plus importante encore que celle observée sur les lettres. Ainsi, elle requière des bases de mots conséquentes. Elle est, de plus, peu discriminante pour les mots différents dont la forme est proche, ce qui rend cet approche n'est applicable que pour des tâches de reconnaissance à vocabulaire distinct et réduit.

Contrairement à l'approche globale, **l'approche analytique** (ou locale) propose de découper le mot en ses éléments constitutifs (caractères ou graphèmes). La reconnaissance du mot complet sera obtenue par la combinaison de ces symboles. Une étape de segmentation est donc nécessaire. Cette tâche est particulièrement délicate du fait de l'absence de segmentation idéale. La difficulté de cette approche a été explicitée par Sayre [Sayre, 1973] : « pour reconnaître les lettres, il faut segmenter le tracé et pour segmenter le tracé, il faut reconnaître les lettres ». Deux problèmes sont envisagés : la sur-segmentation lorsque l'élément constitutif est lui-même fragmenté, et la sous-segmentation lorsque plusieurs éléments constitutifs n'ont pu être isolés. L'avantage de. cette approche est qu'elle permet de travailler sur des vocabulaires ouverts ou de grande taille. Cependant, la difficulté de cette approche est directement liée à la complexité de la segmentation..

Une autre approche spécifique à l'écriture arabe est **l'approche pseudo analytique**. La notion de pseudo-mots (ou PAW : Piece of Arabic Word) introduit une segmentation naturelle de l'écriture arabe et fait apparaître l'approche de reconnaissance pseudo analytique. Cette approche offre une solution intermédiaire aux limites de deux approches analytique et globale: d'une part, elle évite le problème délicat de la segmentation en lettres lié à l'approche analytique. D'autre part, l'interprétation des PAWs plutôt que les mots conduit à une réduction de la taille et de la complexité du vocabulaire et ouvre la voie à l'exploration des

vocabulaires plus étendus que ceux abordés par l'approche globale [Boukerma et Farah, 2012].

Il existe deux types d'approches analytiques suivant que l'on effectue une segmentation explicite ou implicite.

- ***La segmentation explicite***

L'approche à segmentation explicite (appelée aussi dissection) utilise des algorithmes de segmentation (basés sur le contour, le squelette, le profil de projection, etc.) pour proposer des hypothèses de points de segmentation. Une fois les points de segmentation potentiels sont identifiés, il existe deux méthodes permettant de choisir la segmentation finale du tracé : les méthodes dites de segmentation puis reconnaissance (segmentation-based) et les méthodes de segmentation-reconnaissance (segmentation-free ou recognition-based).

Dans la méthode de segmentation puis reconnaissance, les meilleurs points de segmentation sont choisis indépendamment de la reconnaissance. La segmentation est donc antérieure à la reconnaissance et n'est pas remise en cause par son résultat. Tandis que dans la méthode de segmentation-reconnaissance, les phases de segmentation et de reconnaissance sont réalisées en même temps de manière à évaluer et corriger les hypothèses de segmentation par la reconnaissance. Dans ce cas, la segmentation est le résultat de la reconnaissance.

- ***La segmentation implicite***

Dans l'approche à segmentation implicite les caractères sont segmentés tout en étant reconnus. En réalité, il n'y a pas de pré-segmentation du mot, tous les points du tracé sont considérés comme des points de segmentation potentiels en utilisant la technique du fenêtre glissante. Plusieurs travaux de segmentation sont réalisés dans l'état de l'art de la reconnaissance d'écriture manuscrite arabe. Dans cette partie, nous présentons quelques-unes de ces travaux en les divisant en cinq niveaux de segmentation ; en lignes, en mots, en pseudo-mots, en caractères et graphèmes, et en bandes verticales.

## 2.3.1 Segmentation en lignes

De nombreuses méthodes ont été proposées pour extraire des lignes de texte à partir de documents manuscrits arabes. Certaines de ces méthodes sont spécifiques au script arabe [Kumar et al., 2010] [Khayyat et al., 2012]. L'approche commune d'identification des lignes de texte dans une image est d'utiliser le profil de projection horizontale et de chercher la

position des valeurs ayant de densité nulle. Une ligne de texte est alors située entre les positions de deux valeurs de profil de densité nulle.

L'utilisation de la projection horizontale pour la segmentation en ligne est en général suffisante pour le texte imprimé, où les lignes de texte sont relativement droites. Cependant, cette technique impose des difficultés dans le cas de l'écriture manuscrite cursive sans contrainte. Des erreurs de segmentation en ligne peuvent se produire pour plusieurs raisons, ils comprennent : des pages ou lignes inclinées, des lignes courtes qui entraînent une faible longueur de projection et donc ne seront pas considérées comme lignes, des taches non désirées entre les composantes de différentes lignes qui en résultent la combinaison des lignes en une seule ligne. Quelques notes / commentaires sur la marge de la page qui entraînent la combinaison des lignes de texte [Parvez et Mahmoud, 2013a].

La projection partielle [Belaïd et Ouwayed, 2012] est une alternative intéressante afin de faire face à l'inclinaison de texte ; l'inclinaison dans une petite zone de l'image du texte est moins importante que l'inclinaison globale. Cette approche consiste à subdiviser l'image en blocs et utiliser la projection horizontale localement sur chaque blocs pour localiser les lignes de texte.

Kumar et al. proposent dans [Kumar et al., 2010] un algorithme de segmentation en lignes à base de graphe. Dans la première étape, l'algorithme supprime tous les points et les signes diacritiques et estime l'orientation locale de la première composante de chaque mot pour créer un graphe de similarité clairsemé.

Un système de coordonnées local est centré au centre de gravité de chaque composante, et est divisé en cinq régions pour détecter l'orientation locale. Ensuite, l'algorithme retrouve les sous-ensembles disjoints de sommets de graphe de similarité pour lesquels il existe un chemin à partir de chaque élément à chacun des autres éléments de l'ensemble. Ces sous-ensembles représentent les composantes connexes du graphe de similarité qui sont obtenues en utilisant le parcours en largeur (Breadth-First Search). Chaque composante connexe représente (idéalement) une ligne de texte. Pour affiner les lignes de texte, une méthode de regroupement appelée *propagation d'affinité* est utilisée pour affecter les composantes aux lignes du texte. Dans la deuxième étape, les composantes diacritiques qui ont été enlevées sont réaffectées aux lignes de texte correspondantes.

L’algorithme a obtenu un taux de précision de 96% sur un ensemble de données de 125 images de documents arabes. L’approche utilisée est rapide. Toutefois, il ne fonctionne pas bien en présence de composantes touchés.

Dans [Belaïd et Ouwayed, 2012], Belaïd et Ouwayed proposent une méthode d’extraction de lignes de texte dans les documents multi orientées. La méthode est basée sur le maillage d’image qui permet de détecter les orientations (les inclinaisons) progressivement et localement. Ces orientations locales sont ensuite élargies afin d'extraire les orientations des zones plus grandes. Pour extraire les lignes de texte, la méthode exploite les pics de la projection horizontale pour suivre les composantes connexes formant des lignes de texte. L'approche se termine par une séparation définitive des composantes connexes qui se touchent en lignes adjacentes, basée sur l'exploitation de la morphologie des lettres terminales. Une précision de 97% a été rapportée pour 100 document arabes anciennes qui contiennent 2500 lignes. Ces documents sont obtenues à partir de sites web.

L’algorithme présenté par Khayyat et al. [Khayyat et al., 2012] utilise un masque dynamique adaptatif pour séparer les lignes de texte. En utilisant la moyenne de dimensions des composantes connexes et la projection horizontale, l’algorithme détermine pour chaque zone du document la meilleure masque. Ce masque divise le document en grands blobs qui donnent la disposition potentielle des lignes. En fonction des caractéristiques de l’écriture arabe, les composantes connexes au sein de différents blobs se repoussent ou s’attirent, ce qui produit une séparation de lignes de texte. Ensuite, pour résoudre le problème de chevauchement entre les lignes, l’algorithme considère toute composante ayant un hauteur qui dépasse une certaine valeur en tant que deux pseudo-mots, qui doivent être séparés. Enfin, pour affiner les signes diacritiques aux leurs lignes originales, l’algorithme combine ces signes avec les plus proches grands blobs en utilisant la distance euclidienne.

La méthode est testé sur la base CENPARMI qui contient de documents manuscrits arabes ayant des lignes avec angles d’inclinaison différentes et des lignes chevauchées. Les auteurs ont rapporté un taux de précision de 96.3%.

### 2.3.2 Segmentation en mots

Un mot peut être formé de plusieurs composantes connexes car même si l'écriture arabe est cursive, il existe des lettres qui ne s'attachent jamais à la lettre suivante. Ces lettres sont appelées des caractères de la fin du composante connexe.

La segmentation du texte en mots, en particulier en arabe, fait face à quatre défis principaux (voir la Figure 2.15 et la Figure 2.16) :

- l'absence de limites bien définies entre les mots ;
- les composantes qui se touchent ;
- les composantes déconnectées ;
- la fin du mot.

De nombreuses méthodes ont été proposées pour segmenter un texte en mots. Ces méthodes peuvent être classées en deux approches: le seuillage et la classification [Jamal et Jamal, 2013]. Dans [AlKhateeb et al., 2009], Alkhateeb et al. proposent une méthode de segmentation de texte en mots en utilisant la projection verticale. En s'appuyant sur le fait que l'espace séparant les mots est plus important que celui qui séparant les pseudo-mots qui appartiennent au même mot, la distance entre chaque paire de pseudo-mots consécutifs est mesurée et comparée à un seuil pour déterminer si la distance correspond à une séparation de deux mots ou non. La méthode proposée est testée sur 200 images et le taux de segmentation correcte est de 85%.

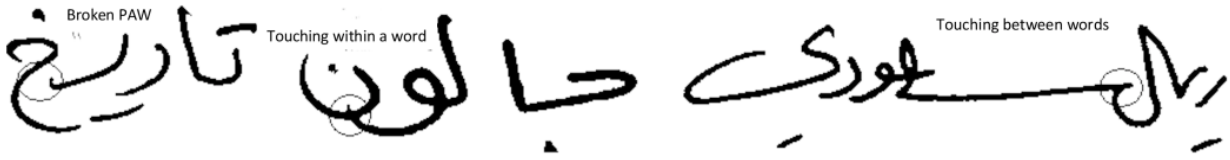



FIG. 2.15 – Composantes touchantes et composantes déconnectées (figure extraite de [Jamal et Jamal, 2013]).

Récemment, Jamal et Suen ont présenté dans [Jamal et Jamal, 2013] une approche de segmentation en mots et pseudo-mots en se basant sur la connaissance sur l'écriture arabe et l'analyse des formes de caractères. L'approche commence par la détection des composantes connexes et l'extraction de leurs squelettes. Ensuite, en utilisant une reconstruction morphologique, les composantes primaires du texte sont extraites et les coupures sont corrigées. Après l'extraction des points de segmentation primaire à l'aide d'une métrique d'espace, l'approche extrait le caractère de la fin de la composante primaire, qui va être reconnu par un classifieur dédié. Ensuite, en se basant sur les formes des lettres arabes lorsqu'elles sont en position finale du mot, le caractère extrait peut être classé à l'un des deux classes : Fin du Mot ou pas Fin du Mot. Enfin, l'approche proposée estime la ligne de base de mot pour faciliter la segmentation des composantes qui se touchent.

## 2.3.3 Segmentation en pseudo-mots

Dans l'écriture manuscrite arabe, il n'y a pas de différence entre les espaces intra-mots (les espaces entre les pseudo-mots d'un même mot) et les espaces inter-mots, comme indiqué dans la Figure 2.16. A cause du manque des limites claires entre les mots, et le fait que l'écriture arabe est naturellement cursive et non contrainte, plusieurs approches ont tendance à segmenter le texte arabe en pseudo-mots plutôt que de mots. Certaines de ces approches reconnaissent les pseudo-mots qu'il extraient [Boukerma et Farah, 2010] [Parvez et Mahmoud, 2013b]. D'autres reconstruisent les mots à partir de leurs pseudo-mots [Moghaddam et Cheriet, 2009] [Khayyat et al., 2012].

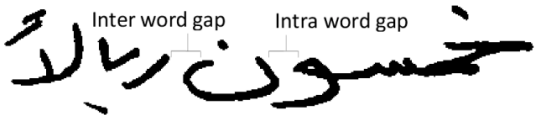



FIG. 2.16 – Espaces intra-mot et inter-mot dans la langue arabe (figure extraite de [Jamal et Jamal, 2013]).

Dans [Parvez et Mahmoud, 2013b], Parvez et Mahmoud introduisent un algorithme de segmentation en PAWs en se basant sur les composantes connexes. Les auteurs commence par l'extraction des composantes connexes. La ligne de base est estimée par l'application d'une régression linéaire des centre de gravité de composantes connexes de base (les plus grandes composantes de la ligne). Ensuite, pour localiser la composante primaire (le corps principal) d'un PAW, la méthode utilise la distance entre la ligne de base et le centre de gravité de la composante, la surface occupée par la composante et l'épaisseur moyen de l'écriture. Toutes les autres composantes sont appelées des composantes secondaires (points ou signes diacritiques). Le chevauchement de chaque composante secondaire avec toutes les composantes primaires est calculé, et la composante secondaire est assignée à la composante primaire où le chevauchement est le maximum. Enfin, la composante primaire et les composantes secondaires associées forment un PAW.

Boukerma et Farah ont présenté dans [Boukerma et Farah, 2012] un algorithme de segmentation en pseudo-mots dont l'objectif est la construction d'une base annotée de pseudo-mots à partir de la base de mots arabes IFN/ENIT. En premier temps, les signes diacritiques du mot sont éliminées, ces signes vont être ensuite réaffectées à leurs composantes primaires correspondantes selon des critères de recouvrement vertical et de

proximité. Ensuite, l'algorithme améliore la segmentation en traitant le problème de sous-segmentation due à la présence de succession des caractères avec jambes (des descendants) qui se touchent. La solution proposée se base sur la détection de la ligne de base et l'extraction du squelette du mot, elle comprend trois étapes :

- Chercher sur le squelette les points d'embranchement qui se situent en dessous de la bande de base et qui n'est pas à un point d'embranchement d'une boucle ;
- Partant du point d'embranchement détecté, un parcours du squelette est fait selon les cinq directions de Freeman, afin de détecter le point de coupure pour dissocier les descendants connectés ;
- Enfin, un lissage du contour de l'image obtenue est appliqué.

L'algorithme proposé ne permet pas la résolution des problèmes de la sur-segmentation et de la sous-segmentation.

D'autres méthodes peuvent encore être citées pour la segmentation en pseudo-mots, comme la projection verticale ([Jiang et Al-Muhtaseb, 2011]).

### 2.3.4 Segmentation en caractères et en graphèmes

La segmentation idéale en lettre d'un mot cursif latin est un problème déclaré insoluble depuis longtemps dans la communauté de la reconnaissance automatique de l'écriture cursive latine. Pour l'écriture manuscrite arabe le problème est plus complexe à cause de la présence des ligatures verticales, de la diversité des formes de caractères, et de la variabilité des liaisons entre caractères.

Dans [Lawgali et al.,2011], Lawgali et al. proposent un algorithme de segmentation en caractères en se basant sur l'extraction de la ligne de base de chaque pseudo-mots. Ils commencent par la segmentation de mots en pseudo-mots puis l'extraction de ses squelettes. Les auteurs supposent que les points de connexion de caractères se trouvent dans la région entourant la ligne de base, pour cela, ils calculent la ligne de base de chaque pseudo-mot en utilisant la projection horizontale et les points de branchement sur le squelette du pseudo-mot. Ensuite, ils suppriment chaque descendant de pseudo-mot qui a un point de départ en dessous de la ligne de base et il n'y a pas de pixels noirs sur la partie gauche de ce point.

La projection verticale est utilisée pour trouver les traits horizontaux ayant un pixel d'épaisseur, où les points candidats pour la segmentation sont déterminés. Un point candidat pour la segmentation est celui qui appartient au trait d'au moins trois pixels de longueur et qui

se situe dans une zone proche de la ligne de base. Si le côté gauche d'un point candidat n'a pas de branchements et n'est pas le caractère Alif (ا), alors ce point est éliminé, sinon il est accepté.

L'algorithme proposé ne parvient pas à segmenter les caractères (س) et (ش). Il est testé sur 800 mots arabes manuscrits pris de la base IFN/ENIT. Les résultats obtenues montrent que 87.9% des points de segmentation ont été extraites correctement.

Dans [Boukharouba et Bennia, 2011], Boukharouba et Bennia présentent un algorithme de segmentation de pseudo-mots en graphèmes. Aprés éliminer les signes diacritiques et mémoriser leurs coordonnées, deux étapes sont réalisées. En premier temps, l'algorithme utilise deux critères pour déterminer les points de segmentation primaires (PSPs). Pour chaque image binarisée d'un pseudo-mot, l'algorithme effectue un balayage vertical colonne par colonne, et localise chaque transition horizontale blanc/noir au pixel noir en retenant son niveau de gris 0 et en remplaçant les autres pixels noirs par des 1s. Le second critère de segmentation est la détection des variations brusques du contour supérieur de la composante qui possède une seule transition. Les points de segmentation primaires obtenus sont représentés par des coupures verticales dans l'image du pseudo-mot (voir Figure 2.17). Ensuite, les points de segmentation primaires sont filtrés en utilisant quelques règles afin de valider les points de segmentations réelles.

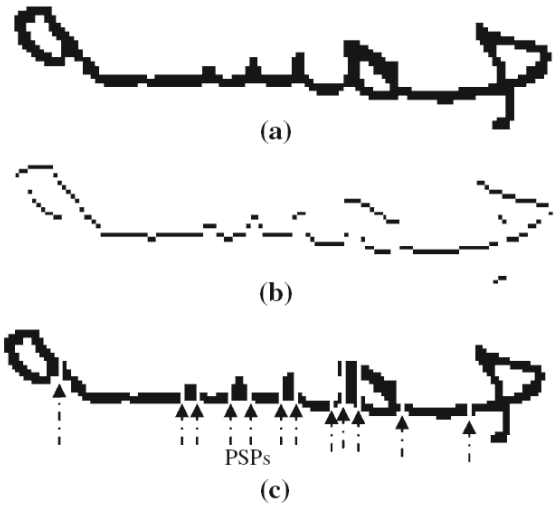



FIG. 2.17 – Les étapes de base d'extraction des points de segmentation primaires (PSPs), (a) le tracé principal du mot cinq " خمسة ", (b) résultat de la localisation verticale de transitions blanc/noir, (c) les points de segmentation primaires.

Récemment, dans [Parvez et Mahmoud, 2013b] Parvez et Mahmoud ont proposé un algorithme de segmentation en caractères et graphèmes qui est intégré dans la phase de reconnaissance, où les deux phases sont réalisées simultanément. Par conséquent, la segmentation obtenue par l'algorithme est affinée et corrigée dans la phase de reconnaissance. Pour trouver l'ensemble des points dominants du contour supérieur du mot, une approximation polygonale de contour est utilisée. Après, les points dominants marqués comme des points concaves et qui vérifient certaines règles sont marqués comme des points de segmentation candidats. Enfin, en utilisant des heuristiques sur ces points candidats, l'algorithme détermine l'ensemble final des points de segmentation.

### 2.3.5 Segmentation en bandes verticales

Plusieurs systèmes de l'état de l'art utilisent une segmentation implicite par fenêtre glissante, ces systèmes sont généralement basés sur des HMMs. La fenêtre glissante parcourt l'image de mot de droite à gauche et la découpe en bandes verticales qui peuvent être régulières ou non, éventuellement avec recouvrement partiel des bandes successives (voir Figure 2.18). La fenêtre glissante est divisée verticalement en cellules permettant d'extraire des vecteurs de caractéristiques de bas niveau qui seront soumis en entrée du reconnaisseur.

FIG. 2.18 – Illustration de la segmentation en bandes verticales en utilisant la fenêtre glissante.

Cette technique présente l'avantage d'être simple, robuste au bruit, et est indépendante de la connexité. Le défaut de cette méthode est que la séquence générée contient beaucoup de bruit (recouvrement de deux lettres successives). C'est également vrai dans le cas des lettres superposées verticalement, mais qui ne se touchent pas nécessairement : un descendant comme 'ر' ou 'و' avec la lettre suivante [Menasri, 2008]. L'utilisation de la fenêtre glissante sera détaillée dans la Section 2.4.

## 2.4 Extraction de caractéristiques

Avant de pouvoir être interprétées par un système de reconnaissance d'écriture

manuscrite, les images sont transformées. La retranscription correspond à un ensemble de caractéristiques extraites des images ou de parties des images. Il existe plusieurs façons d'extraire des caractéristiques d'une image de mot, soit par fenêtre glissante, soit par segmentation explicite de l'image (en graphèmes), soit directement sur l'image complète. Quelle que soit la manière de segmenter les images, des propriétés des fragments d'image sont ensuite évaluées et représentées numériquement dans un vecteur de taille *n*. On dit alors que le nombre de caractéristiques extraites est *n*. L'ensemble des vecteurs extraits d'une image est appelé la séquence de vecteurs de caractéristiques de l'image [Bianne-Bernard, 2011].

Lorsque le nombre de caractéristiques devient trop élevé, des techniques de sélection de caractéristiques comme l'analyse en composantes principales (ACP; en anglais *PCA, Pincipal Component Analysis*), appelée également transformation de Karhunen-Loève, peuvent être utilisées. Ces techniques permettent de réduire le nombre de dimensions de l'espace de caractéristiques, en projetant les données sur un sous-ensemble d'axes qui maximisent la dispersion des échantillons disponibles.

Dans cette section, nous présentons les caractéristiques les plus utilisées dans l'état de l'art de la reconnaissance d'écriture manuscrite arabe. Ces caractéristiques peuvent être classées en deux types [Parvez et Mahmoud, 2013a] : statistiques et structurelles.

### 2.4.1 Caractéristiques statistiques

Plusieurs systèmes utilisent des caractéristiques que nous qualifions de statistiques. Ces caractéristiques sont basées sur une distribution statistique des pixels sur toute l'image. Elles ont l'avantage d'être robustes face au bruit ou à la variabilité de l'écriture car elles utilisent des valeurs réelles au lieu de décrire des formes. Parmi ces caractéristiques nous pouvons citer les suivantes :

- Les caractéristiques les plus simples sont les valeurs mêmes des pixels, utilisées par exemple dans [Pechwitz et al., 2012]. L'avantage d'utiliser les valeurs des pixels comme caractéristiques est de ne nécessiter aucun traitement, mis à part une étape de normalisation des caractères.
- Les densités des pixels des images sont également utilisées. Cette caractéristique est la plus extraite par la fenêtre glissante dans les systèmes à base de HMMs. Certains systèmes calculent le nombre de pixels d'écriture dans la fenêtre [Azeem et Azeem,

2013] ou dans chaque cellule de la fenêtre [Kessentini et al., 2010], d'autres calculent la moyenne des densités des pixels d'écriture [Jiang et Al-Muhtaseb, 2011], ou utilisent un '1' pour chaque pixel noir dans la cellule et un '0' sinon [Mohamad et al., 2009]. Chaque cellule compte une seule caractéristique.

- Le nombre de transitions des pixels écriture/fond est une caractéristique très utilisée. Certains systèmes utilisent le nombre de transition dans la fenêtre glissante [Mozaffari et al., 2008], d'autres utilisent le nombre de transitions entre les cellules de la fenêtre [Likforman-Sulem et al., 2012] [Bianne-Bernard et al., 2011]. Le nombre de transitions horizontales et verticales dans l'image du graphème est aussi utilisé [Boukharouba et Bennia, 2011]. Dans [Likforman-Sulem et al., 2012], les auteurs calculent la différence des positions verticales (ordonnée y) du centre de gravité des pixels d'écriture dans deux fenêtres consécutives.
- Les moments invariants de *Hu* sont des caractéristiques intéressantes car elles sont invariantes en translation, taille et rotation. Ce sont des mesures statistiques de la distribution des pixels autour du centre de gravité du caractère. Ces caractéristiques peuvent être facilement et rapidement extraites d'une image de texte, elles peuvent tolérer modérément les bruits et les variations. Ces caractéristiques sont utilisés, par exemple, dans [Akhateeb et al., 2009] [Abandah et Anssari, 2009]. Abdullah et al. [Abdullah et al., 2011] utilisent des moment invariants améliorés. Ils effectuent le calcule en remplaçant le centre de gravité par le centre de l'image comme point de référence.
- Les quatre profils (haut, bas, droite, gauche) sont utilisés dans certains systèmes, par exemple dans [Boukharouba et Bennia, 2011]. La zone du profil est calculée comme étant le nombre de pixels entre les bords de l'image (la boite englobant) et le contour du graphème. La caractéristique de profil est calculée comme étant le rapport entre la zone de chaque profil et la zone du graphème.
- L'histogramme des directions de Freeman est également très intéressante, il est basé sur les points du contour. Kessentini et al. [Kessentini et al., 2010] travaillent sur des fenêtres glissantes avec recouvrement partiel, positionnées sur les contours de l'image du mot. A chaque position de la fenêtre, les points du contour supérieur et inférieur sont extraits, et le code de Freeman de chaque points est déterminé. Enfin un

histogramme de 8 directions est calculé sur tous les points de la fenêtre, constituant ainsi un ensemble de 8 caractéristiques.

- D'autres caractéristiques statistiques peuvent encore être citées, comme l'histogramme de gradient [Azeem et Azeem, 2013] ou encore les descripteurs de Fourier.

### 2.4.2 Caractéristiques structurelles

Les caractéristiques structurelles sont des aspects intuitifs de l'écriture. Elles nécessitent une étape de squelettisation ou d'extraction de contour. Les boucles, les points de branchement, les points d'intersections, les points terminaux, la concavité, les ascendants et les descendants, le nombre de points diacritiques et leur position par rapport à la ligne de base, les traits et leurs tailles, la catégorie de la forme (partie primaire ou point diacritique) constituent autant de caractéristiques pertinentes pour la discrimination des caractères manuscrits.

Dans [Kessentini et al., 2010], Kessentini et al. utilisent une caractéristique structurelle basée sur les points du contour. Ils déterminent pour chaque point ($p$) du contour supérieur (resp. inférieur) la nature du point ($p'$) qui lui correspond verticalement sur le contour en vis à vis. Chaque points ($p'$) du contour inférieur (resp. supérieur) peut appartenir au contour inférieur (resp. supérieur), à une occlusion, au contour supérieur (resp. inférieur) ou pas de point, ce qui veut dire que le point du contour supérieur (resp. inférieur) est un point extrême dans la direction horizontale. Ces caractéristiques fournissent des informations concernant des traits simples, des boucles, des rebroussements horizontaux et les points extrêmes sur l'image du mot.

### 2.4.3 Approche mixte statistique et structurelle

Une combinaison de différents types de caractéristiques est souvent mise en œuvre afin d'obtenir plusieurs représentations d'un même forme et d'améliorer la discrimination. Dans ce qui suit nous présentons quelques travaux qui proposent des caractéristiques alliant données statistiques et données structurelles :

Dans [Likforman-Sulem et al., 2012], les auteurs passent en revue les caractéristiques utilisées dans les systèmes de reconnaissance à base des modèles de Markov cachés. Ils distinguent deux classes de caractéristiques : celles extraites par fenêtre glissante, et celles

extraites par segmentation en graphèmes. Les auteurs utilisent une fenêtre glissante de hauteur égale à la hauteur de l'image et de largeur 8 pixels divisées verticalement en 20 cellules homogènes. La fenêtre parcoure l'image de droite à gauche avec recouvrement. Dans chaque position de la fenêtre est extrait 3 ensembles de caractéristiques : caractéristiques de distribution, caractéristiques de concavité et caractéristiques dynamiques, dont certaines dépendent de la position estimée des lignes de base. Les deux premiers ensembles de caractéristiques permettent de détecter la présence des ascendants et des descendants, des points diacritiques et des concavités locales. Le premier ensemble comporte 16 caractéristiques de distribution. Elles caractérisent la densité des pixels noirs dans la fenêtre:

- 2 comptent le nombre de transitions d'écriture (noir)/fond (blanc) entre deux cellules consécutives : l'une dans la fenêtre, l'autre au-dessus de la ligne de base basse.
- 11 sont reliées directement aux densités des pixels d'écriture : l'une est la densité globale de pixels dans la fenêtre. 2 sont les densités de pixels au-dessus et en dessous de la ligne de base basse. Les 8 dernières sont les densités de pixels pour chaque colonne de pixels de la fenêtre (ici une fenêtre est de largeur 8).
- Les 3 dernières caractéristiques sont liées à la position du centre de gravité des pixels d'écriture : l'une est dérivative (différence des positions verticales du centre de gravité de deux fenêtres consécutives). La deuxième donne sa position par rapport à la ligne de base basse (en terme de distance de pixels). La dernière enfin donne la zone à laquelle appartient le centre de gravité (au-dessus de la ligne de base haute, en dessous de la ligne de base basse ou entre les deux).

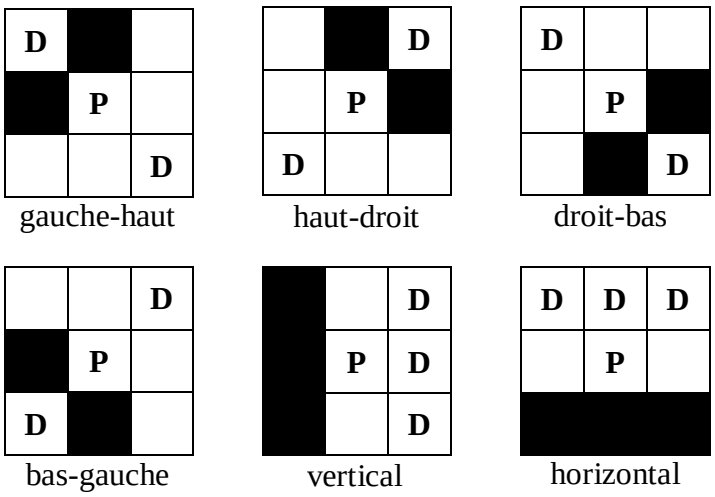



FIG. 2.19 – Masques pour le calcul de caractéristiques de concavité.

Le deuxième ensemble regroupe 12 caractéristiques de concavités. Les auteurs utilisent 6 configurations de pixels qui sont illustrées sur la Figure 2.19. Pour chacune des configurations, ils comptent le nombre de pixels lui correspondant dans l'ensemble de la fenêtre glissante, ainsi que le nombre d'occurrences entre les lignes de base haute et basse. Un vecteur de 28 caractéristiques sont obtenues donc par chaque fenêtre avec 23 d'entre eux étant indépendants de la position estimée des lignes de base.

Likforman-Sulem et al. introduisent le contexte au niveau de l'extraction de caractéristiques en utilisant des caractéristiques dérivées. Elles représentent la dynamique de caractéristiques des fenêtres glissantes entourant la fenêtre courante. La dérivation est calculée par une régression du premier et du deuxième ordre du vecteur de caractéristiques en utilisant les vecteurs des fenêtres entourant la fenêtre courante. Le vecteur de caractéristiques final en entrée du système, est alors la concaténation du vecteur initial et de ses régressions.

Outre les caractéristiques extraites par les fenêtres glissantes, les auteurs utilisent 74 caractéristiques extraites par segmentation en graphèmes :

- 3 caractéristiques sont liées à la hauteur et à la largeur et le rapport hauteur-largeur de la boîte englobant du graphème.
- 2 sont liées à la position de la boîte englobant du graphème par rapport à la ligne de base.
- 2 sont reliées à la position du centre de gravité du graphème dans la boîte.
- 4 sont liées aux densités des pixels noirs : l'une est la densité des pixels noirs dans la boîte. 3 sont les densités des pixels noirs dans les trois zones : au-dessus de la ligne de base haute, en dessous de la ligne de base basse, ou entre les deux.
- 3 sont les surfaces des boucles dans les trois zones précédentes.
- 20 sont les valeurs de quatre profils : haut, bas, gauche et droite, chacun est pris en cinq points.
- 20 sont reliées aux épaisseurs cumulées du graphème horizontalement, verticalement, et le long des deux diagonales, chacun est pris dans les 5 régions parallèles.
- 20 sont les nombres d'intersections le long des mêmes quatre directions, dans les mêmes 5 régions parallèles.

Pechwitz et al. [Pechwitz et al., 2012] proposent deux ensembles de caractéristiques différentes pour entraîner un système de reconnaissance à base de HMM. Ils font glisser une fenêtre de trois colonnes sur une image normalisée en niveaux de gris. Chacune de ces trois

colonnes extrait les valeurs des pixels en tant que caractéristiques (voir Figure 2.20). Cet ensemble de valeurs forment un vecteur de caractéristique.

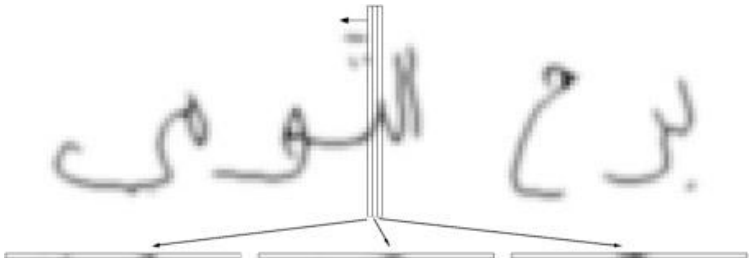


FIG. 2.20 – L'extraction des valeurs des pixels en utilisant une fenêtre glissante des trois colonnes.

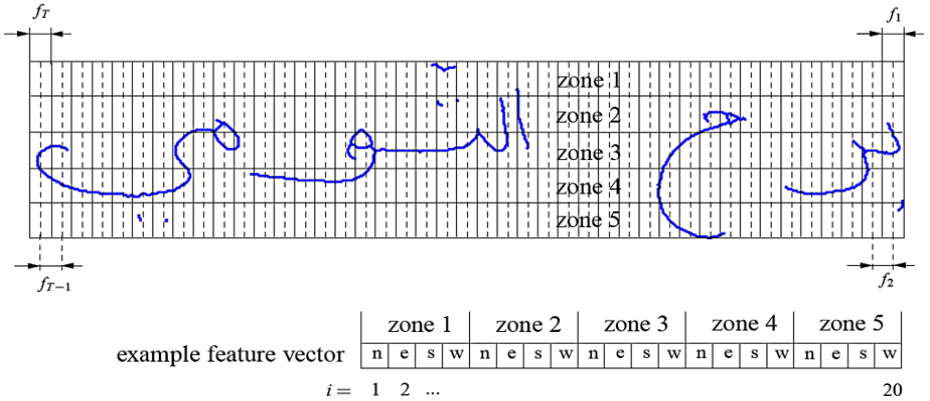



FIG. 2.21 – L'extraction des caractéristiques directionnelles de squelette dans cinq zones en utilisant des bandes verticales avec recouvrement.

Le deuxième ensemble de caractéristiques utilisées sont des caractéristiques directionnelles. L'image du mot est divisée en bandes verticales avec recouvrement, chaque bande est divisée horizontalement en 5 zones de même hauteur (voir Figure 2.21). La longueur de chaque trait contenu dans une zone est calculée dans les quatre directions nord, est, sud, ouest et considérée comme étant une caractéristique. Chaque bande verticale est représentée par un vecteur de 20 caractéristiques. Pour assurer l'invariance de la hauteur du mot (problème de très différentes plages de valeurs de chaque caractéristique), les auteurs ont effectué une normalisation des valeurs de caractéristiques avec la hauteur de la zone.

Abdel Azeem et Ahmed [Azeem et Azeem, 2013] utilisent trois boîtes englobant pour chaque mot :

- une boite horizontale ;
- une boite inclinée vers la droite ;

- une boite inclinée vers la gauche.

L'introduction des boites inclinées dans l'extraction des caractéristiques est pour faire face à l'inclinaison du mot sans appliquer aucune correction. Les trois boites englobant sont divisées en *n* bandes horizontales non uniformes avec approximativement le même nombre de pixels noirs dans chaque bande. Une fenêtre (6 pixels de largeur et 3 pixels de superposition) divisée en *n* cellules de hauteur différentes se déplace de droite à gauche sur chaque boite englobant, et extrait des caractéristiques de concavité et de gradient (voir Figure 2.22).

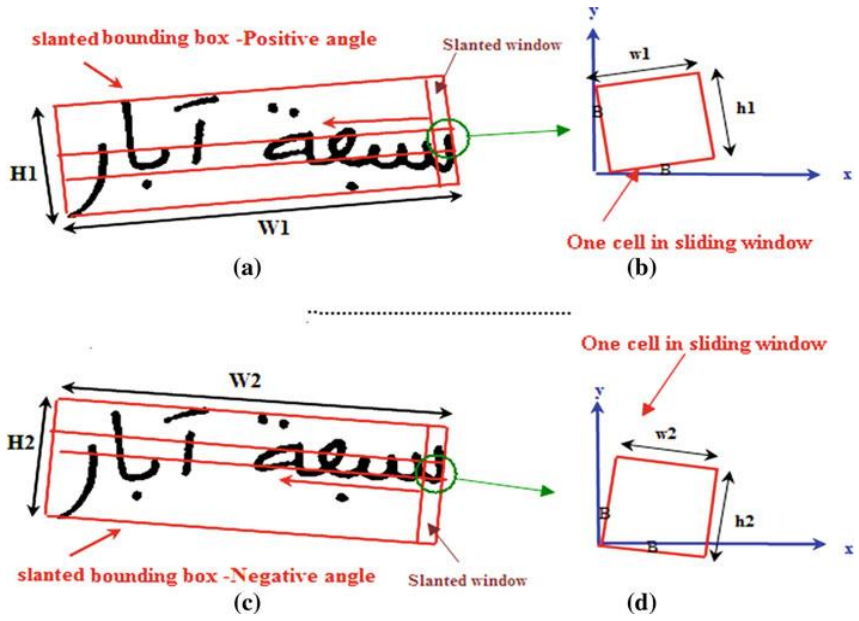



FIG. 2.22 – Boîtes englobant inclinées et fenêtres glissantes inclinées, (a) inclinaison avec angle positif (b) inclinaison de cellule avec angle positif (c) inclinaison avec angle négatif (d) inclinaison de cellule avec angle négatif.

Les auteurs utilisent huit configurations de pixels pour décomposer l'image du mot en huit images représentant les traits verticaux, horizontaux, diagonale gauche, diagonale droit (voir Figure 2.23). Sur chaque image, le nombre des pixels noirs et le centre de gravité normalisé dans la fenêtre sont calculés formant 16 caractéristiques de concavité.

Pour extraire les caractéristiques de gradient, une fenêtre des trois cellules non uniformes est utilisée, les huit histogrammes des directions de gradient est calculé dans chaque cellule formant 24 caractéristiques par fenêtre. Ainsi, un vecteur de taille 40 est généré par fenêtre, il comprend les caractéristiques de concavité et les caractéristiques de gradient de l'image du mot. D'autres caractéristiques seront présentées dans la Section 2.5.

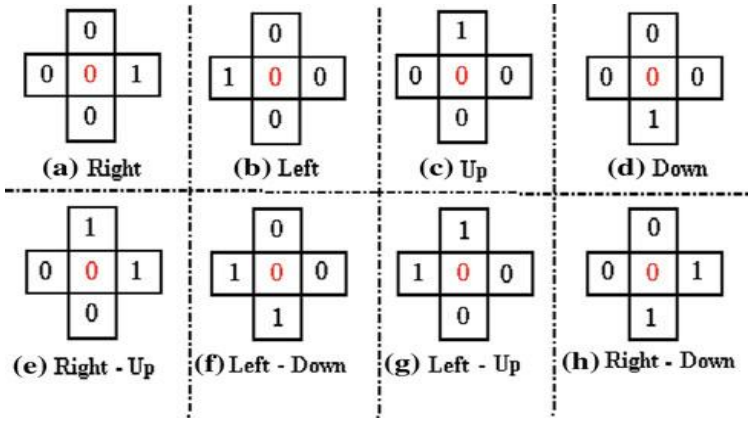



FIG. 2.23 – Huit configurations utilisées pour calculer les caractéristiques de concavité.

## 2.5 Reconnaissance

Le rôle d'un reconnaisseur est de se prononcer sur l'appartenance d'une forme à chacune des classes de caractère à partir du vecteur de caractéristiques. Il existe de nombreux classifieurs qui peuvent être utilisés dans la reconnaissance d'écriture manuscrite. Une description des classifieurs les plus connus est présentée dans la Section 1.3. L'approche la plus utilisée dans la reconnaissance d'écriture manuscrite est l'approche séquentielle où l'image de mot en entrée du système est converti en séquence de vecteurs de caractéristiques. L'avantage de cette approche est qu'elle respecte la nature des données. La plupart des méthodes de reconnaissance d'écriture manuscrite arabe à ce jour utilisent les HMMs qui sont l'un des outils les plus puissants pour la modélisation de séquences. L'utilisation des HMMs offrent plusieurs avantages pour la reconnaissance d'écriture cursive [Parvez et Mahmoud, 2013a] :

- ils ne nécessitent pas de segmentation explicite de l'écriture ;
- ils sont robustes face au bruit et à la variabilité de l'écriture ;
- leur apprentissage ainsi que le décodage sont basés sur des algorithmes performants et éprouvés ;
- les outils des HMMs sont disponibles gratuitement.

Les HMMs sont très convenables pour la mise en œuvre de l'approche de reconnaissance analytique. Un système analytique peut ou ne peut pas pré-segmenter les mot en unités plus petites (caractères ou graphèmes). Dans les deux cas, le système HMM extrait des vecteurs de caractéristiques à partir de chaque unité pré-segmentée ou, en faisant glisser une fenêtre sur l'image de mot. La fenêtre glissante parcourt l'image de mot de droite à gauche et la découpe

en bandes verticales telle que deux bandes consécutives se chevauchent. Des vecteurs de caractéristiques de bas niveau sont extraits en utilisant la fenêtre glissante dans son ensemble et les cellules le composent, ces vecteurs seront soumis en entrée du reconnaisseur.

Dans ce qui suit, nous présentons quelques systèmes de l'état de l'art de reconnaissance de l'écriture manuscrite arabe basés sur les HMMs.

Dans [Mohamad et al., 2009], Mohamad et al. proposent un système de reconnaissance à base de HMM à fenêtre glissante. Trois types de fenêtres sont utilisées (voir Figure 2.24) :

- une fenêtre verticale ;
- une fenêtre inclinée d'angle $+\alpha$ ;
- une fenêtre inclinée d'angle $-\alpha$.

Les auteurs ont proposé ces fenêtres glissantes pour faire face aux problèmes de l'inclinaison de l'écriture, le chevauchement des ascendants et descendants et le décalage dans les positions des signes diacritiques. Ces 3 fenêtres permettent d'entraîner 3 classifieurs HMM homogènes, dont le classifieur principal utilise la fenêtre verticale. Les réponses obtenues par ces classifieurs sont combinées pour calculer la réponse du système global.

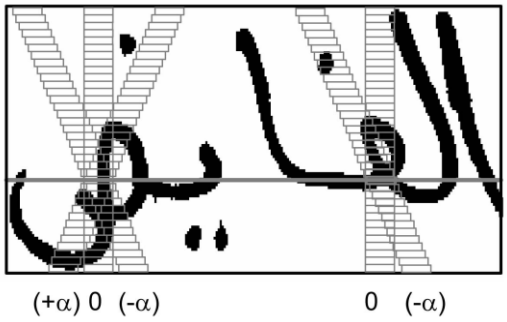



FIG. 2.24 – Image de mot découpée en bandes verticales et inclinées. La ligne grise est la ligne de base inférieure.

Des séquences de caractéristiques différentes sont extraites pour chaque classifieur, résultant en classifieurs avec des paramètres différents. Les trois classifieurs ont la même topologie et utilisent le même algorithme d'apprentissage et de reconnaissance. Les modèles de mots sont construits par la concaténation des modèles de caractères. Pour fusionner les résultats des classifieurs, les auteurs ont proposé trois schémas de combinaison : une somme, un vote, ou l'utilisation d'un réseau de neurones de type perceptron multicouche (MLP) qui renvoie le classifieur à sélectionner à partir des scores des trois classifieurs.

Mohamad et al. réalisent leurs expériences sur l'ensemble-*d* de la base IFN/ENIT. Un taux de reconnaissance de 90.26% a été rapporté en utilisant le schéma de combinaison basé sur le MLP. Cependant, avec chaque classifieur séparément, la meilleur résultat est obtenue par le classifieur principal. Les auteurs ont démontré que la combinaison MLP a surperformé la somme et le vote.

Dans [Kessentini et al., 2010], Kessentini et al. utilisent une approche de reconnaissance basée sur une modélisation par des modèles de Markov cachés multi-flux. L'approche proposée consiste à modéliser chaque flux de données (caractéristiques de bas niveau) par un HMM. Cette approche permet la fusion des caractéristiques indépendantes de différents types d'une manière asynchrone à travers des modèles de Markov coopératifs (voir Figure 2.25).

Un apprentissage indépendant des modèles est réalisé sur les différents flux de données, par l'algorithme de *Baum-Welch*. Lors de la recombinaison des flux, les auteurs utilisent deux stratégies : combinaison par poids égaux, et combinaison par fréquence relative en utilisant un paramètre de pondération. Une fois appris, le décodage simultané des différents modèles est réalisé, ce processus consiste à calculer la vraisemblance du meilleure chemin de la séquence de vecteurs d'observations des différentes sources d'informations. Pour ce faire, les auteurs ont étudié deux solutions : la recombinaison au niveau état et la combinaison au niveau sous-unité lexicale. Enfin, une généralisation de l'approche a été faite de façon à combiner *N*-flux.

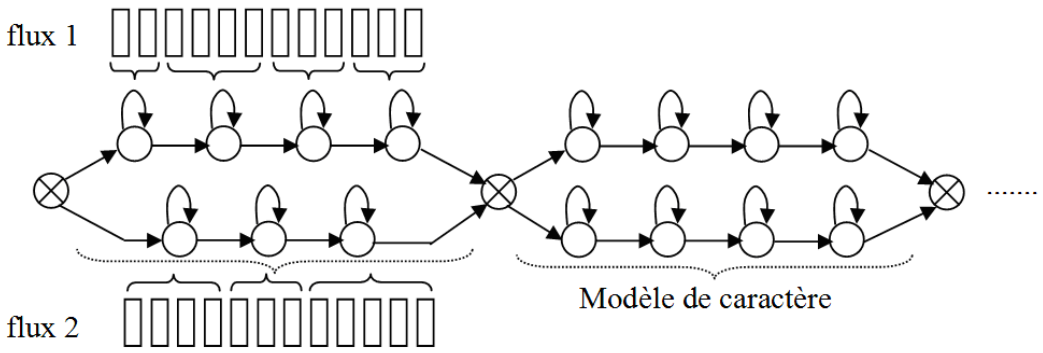



FIG. 2.25 – Structure générale d'un modèle multi-flux.

Deux types de caractéristique sont proposés dans ce travail. Les caractéristiques de premier type sont basées sur le contour. Elles sont extraites des points de contour inférieure et supérieur en utilisant une fenêtre glissante avec un recouvrement partiel entre deux positions successives. Les caractéristiques extraites sont :

- l'histogramme des directions de Freeman des pixels.

- l'histogramme des boucles, les points tournants, les traits simples et des points extrêmes sur l'image du mot.
- la position de chaque points du contour : appartient à la zone médiane, la zone supérieure ou la zone inférieure (en utilisant les lignes de base supérieure et inférieure).

Le deuxième type des caractéristique utilisées comprend les caractéristiques basées sur les densités des pixels noirs et d'autres caractéristiques de concavité comme décrites dans la Section 2.4.3. Pour calculer ces caractéristiques, les auteurs utilisent deux fenêtres glissantes de largeurs différentes, chacune est divisée verticalement en 4 cellules. Les deux ensembles de caractéristiques composent 4 flux de données.

Le système de reconnaissance est réalisé sur deux bases publiques différentes. La base de mots arabes IFN/ENIT et la base de mots Latins IRONOFF. Les auteurs obtiennent un taux de reconnaissance correct de 79.6% pour l'ensemble-*e* de la base IFN/ENIT.

Pechwitz et al. [Pechwitz et al., 2012] proposent un système de reconnaissance de mots à base de Modèles de Markov Cachés. Ils utilisent deux fenêtres glissantes différentes pour extraire deux ensembles de caractéristiques, les valeurs de pixels en nivaux de gris et les caractéristiques directionnelles des traits. Le processus d'extraction des caractéristiques est décrit dans la Section 2.4.3.

Les auteurs modélisent les lettres par des modèles de Markov cachés semi-continus, dont la topologie suit le modèle de *Bakis* avec 7 états émetteurs (transitions gauche-droite et saut d'état autorisé). Ces modèles de lettres sont entraînés pour décoder les séquences de vecteurs de caractéristiques extraites. L'apprentissage est effectué à l'aide de la méthode de *Baum-Welch*. Les modèles de mots sont obtenus par concaténation de modèles de lettres (sous-modèle), et l'initialisation de la segmentation du mot en caractères est effectuée par un algorithme de programmation dynamique. Ce système est testé sur la base IFN/ENIT et donne de bonnes performances : 92.1% de reconnaissance.

Dans [Azeem et Azeem, 2013], Abdel Azeem et Ahmed présentent un système de reconnaissance de mot par combinaison de classifieurs à base de HMM continu. Une certaine prétraitement est effectué sur l'image binarisée du mot. Le prétraitement consiste à normaliser l'épaisseur du mot à trois pixels et à fixer l'espacement entre les différentes parties du mot en le réduisant si elle est supérieure à un seuil prédéterminé. Les auteurs utilisent une fenêtre glissante verticale et deux fenêtres inclinées pour extraire deux types de caractéristiques;

celles de concavité et celles de gradient. La méthode utilisée dans l'extraction des caractéristiques est décrite dans la Section 2.4.

Le système proposé est composé des trois classifieurs à base de HMM continu, chaque classifieur observe l'image du mot à partir d'une orientation donnée, chacun utilise des séquences de vecteurs obtenues par l'une des trois fenêtres glissantes. Pour chaque caractère est construit un modèle spécifique, et le modèle du mot est construit par la concaténation des modèles de caractères. 166 modèles de caractères sont générés pour représenter les différentes formes des caractères. Un même mot est reconnu par chaque classifieur, et la décision finale est obtenue en fusionnant les résultats des trois classifieurs. La fusion est appliquée au niveau de classifieur en utilisant le score de confiance normalisé produit par chaque classifieur, la normalisation consiste à diviser le score de confiance d'un mot par le nombre de ses fenêtres. Les auteurs ont proposé la combinaison entre trois méthodes de fusion : une somme, un vote, et l'utilisation des règles de maximum. Ce système donne de bonnes performances de reconnaissance : 93.1% et 84.8% respectivement sur la base IFN/ENIT.

## 2.6 Post-Traitements

Une étape de post-traitement peut être rajoutée à un système de reconnaissance d'écriture. Elle peut améliorer significativement le taux de la reconnaissance en affinant la décision prise par l'étape précédente et en reconnaissant les mots par l'utilisation de dictionnaire ou des informations contextuelles. Cette étape peut nécessiter des techniques de traitement du langage naturel (TLN). Généralement, l'étape de post-traitement utilise un lexique de mots arabes pour la vérification et l'amélioration du taux de reconnaissance [Parvez et Mahmoud, 2013a].

Une fois le texte reconnu est disponible, des techniques automatique pour la vérification et la correction orthographique peuvent être appliquées pour améliorer les résultats.

Deux types d'erreurs peuvent être distingués [Kukich, 1992] :

- Nous trouvons tout d'abord les erreurs conduisant à des « non-mots » c'est-à-dire que le mot écrit n'est pas valide dans le langage considéré. Un dictionnaire peut être utilisé pour détecter ces erreurs. L'utilisation de ces informations linguistiques se situe donc au niveau lexical.

- Le second type d'erreur donnent lieu à des mots qui sont valides dans le langage. Pour détecter (et par conséquent corriger) de façon efficace ce type d'erreurs, il est intéressant de tirer partie des connaissances linguistiques au niveau syntaxique. Dans ce cas, les informations concernent le contexte du mot écrit et notamment les mots qui l'entourent. Ces connaissances linguistiques sont généralement représentées grâce à des modèles de langage : ils permettent de définir les suites de mots possibles d'une langue donnée.

Ces deux niveaux d'informations peuvent ensuite être utilisés conjointement, de façon complémentaire.

## 2.7 Mesures de performances

La performance de la reconnaissance est mesurée par un ensemble de métriques permettant de calculer l'écart entre un document de référence et le résultat de la reconnaissance. Ce document de référence est couramment appelé vérité terrain ou *ground-truth* en anglais. Nous nous intéressons sur les mesures de performance au niveaux mots : taux de reconnaissance de mots (*Word Recognition Rate*) et taux d'erreur de mots (*Word Error Rate*).

Le taux de reconnaissance au niveau mot représente le pourcentage de mots correctement reconnus dans le document. Il est définit par la formule suivante :

$$T_{reco} = \frac{N_{corr}}{N_{tot}}$$

$T_{reco}$ : taux de reconnaissance

$N_{corr}$ : nombre de documents correctement reconnus

$N_{tot}$ : nombre total de mots du document de référence.

De manière analogue, le taux d'erreur au niveau mot représente le pourcentage de mots mal reconnus dans le document. Il est calculé comme suit :

$$T_{err} = \frac{N_{err}}{N_{tot}}$$

$T_{err}$ : taux d'erreur

$N_{err}$ : nombre de mots mal reconnus

# Conclusion

Dans ce chapitre, nous avons introduit la reconnaissance de l'écriture manuscrite comme étant la tâche de transcription de séquences de mots. Nous avons vu que le traitement d'un document manuscrit se fait en plusieurs étapes. Pour chaque étape nous avons présenté les traitements pouvant être appliqués avec une présentation de l'état de l'art sur l'écriture manuscrite arabe.

En premier temps, nous avons vu qu'il est nécessaire de prétraiter les images. Certains documents manuscrits sont en effet bruités et les systèmes de reconnaissance peuvent en être affectés. Ensuite, nous avons présenté les différents type de segmentation peuvent être appliquées sur une image de texte en le divisant en unités simples : lignes, mots, caractères ou graphèmes.

Dans la deuxième section nous avons décrit comment transformer une image afin qu'elle soit interprétée par un système de reconnaissance : c'est l'extraction de caractéristiques. Généralement, deux types de caractéristiques peuvent être extraites à partir d'une image : statistiques et structurelles. Ces caractéristiques sont extraites suivant l'une des trois approches : l'approche holistique, l'approche à segmentation explicite et l'approche à segmentation implicite par fenêtres glissantes. Cette dernière approche que nous avons choisis comme nous le verrons dans le Chapitre 4. Enfin, nous avons abordé l'étape de reconnaissance en se focalisant sur l'utilisation des modèles de Markov cachés HMMs, que nous avons choisis pour élaborer notre système de reconnaissance de l'écriture manuscrite arabe.

De manière générale, la reconnaissance d'un document a pour objectif la retranscription des données contenues dans ce document. La transcription qui résulte de ce processus permet ensuite d'effectuer des tâches de fouille sur des textes en vue d'en extraire de connaissances. C'est le sujet du chapitre suivant.

# Chapitre 3

# Extraction de connaissances à partir de données textuelles (fouille de textes) : techniques et algorithmes

## Introduction

L'ère de l'information a rendu facile de stocker de grandes quantités de données. La prolifération des documents textuels disponibles sur le Web, sur des intranets d'entreprise, sur les fils de presse est écrasante. Cependant, bien que la quantité de données dont nous disposons ne cesse de croître, notre capacité à absorber et traiter ces informations reste constante. Les moteurs de recherche d'informations ne font qu'exacerber le problème en faisant de plus en plus de documents disponibles en quelques appuis de touches, d'où la nécessité de progrès dans la conception des algorithmes qui peuvent efficacement traiter de gros volumes de données textuelles.

La fouille de textes, traduit du terme anglais *Text Mining*, est un nouveau et passionnant domaine de recherche qui tente de résoudre le problème de la surcharge d'information par l'analyse automatique de grande collection de documents afin de découvrir des informations inattendues ou nouvelles. Pour ce faire, la fouille de textes combine des techniques issues de disciplines scientifiques diverses telles que la fouille de données, l'apprentissage automatique, la recherche d'information et le traitement automatique des langues naturelles. La fouille de textes est apparue dans la deuxième moitié des années 90, en écho aux travaux réalisés depuis les années 80 sur des bases de données. Cependant, selon la culture scientifique dont sont issus les chercheurs, le terme de fouille de textes désigne différents types d'activités et techniques.

Ce chapitre a pour objectif de présenter les principales techniques et algorithmes de la fouille de textes. Nous nous limiterons surtout à la classification de documents et l'extraction d'information comme des problèmes d'apprentissage automatique. Après présenter des concepts de base, nous allons aborder le problème de représentation de documents textuels. Nous nous intéressons ensuite à la classification supervisée (catégorisation) et non supervisée (segmentation) de textes. Enfin, nous verrons l'extraction d'information dans les documents textuels.

# 3.1 Concepts de base

Avant d'aborder les techniques de la fouille de textes, nous avons vu qu'il est nécessaire de présenter des concepts de base à propos ce domaine de recherche. Dans cette section nous allons commencer par définir la fouille de textes. Ensuite, faire une confrontation entre cette discipline et la recherche d'information, et enfin, présenter les caractéristiques des données textuelles.

## 3.1.1 Définition de la fouille de textes

De manière analogue à la fouille de données, la fouille de textes, aussi connue comme l'Extraction de Connaissances à partir de Texte (ECT), se réfère au processus d'extraction de motifs intéressants à partir de sources de données pour en extraire de connaissances (un motif est un ensemble de caractéristiques de textes). Dans le cas de la fouille de textes, les sources de données sont des collections de documents, et les motifs intéressants ne se trouve pas dans des enregistrements de base de données mais dans les données textuelles non structurées dans ces collections. La fouille de textes applique les mêmes fonctions analytiques de la fouille de données, mais applique également des fonctions analytiques du Traitement Automatique de la Langue (TAL) et de la Recherche d'Information (RI).

Le processus de la fouille de textes commence par la modélisation des textes afin de les préparer pour la fouille de données, et se termine par l'interprétation des résultats de la fouille et l'enrichissement des connaissances. La fouille de données n'est donc qu'une étape du processus de fouille de texte.

Les outils de la fouille de textes sont utilisés pour [Ben-Dov et Feldman, 2010]:

- Extraire des informations pertinentes des documents.

- Trouver des tendances ou des relations entre les informations extraites (personnes / lieux / organisations, etc.).
- Catégoriser et organiser les documents en fonction de leur contenu.
- Recherche d'information.
- Segmenter les documents en fonction de leur contenu.

## 3.1.2 La fouille de textes versus la recherche d'information

Il est important de différencier entre la fouille de textes et la Recherche d'Information (RI). La Recherche d'Information se définit par un ensemble de méthodes et d'outils qui permettent à un utilisateur de formuler une requête (i.e. un ensemble de critères) et qui sélectionnent dans un fond documentaire les documents répondant à ces critères [Toussaint, 2011].

L'amélioration de l'efficacité de la recherche est un sujet central de recherche dans le domaine de la recherche d'information, où de nombreux sujets de recherche telles que la catégorisation (classification supervisée), la segmentation (classification non supervisée), et le résumé de textes sont également étudiés. Cependant, les travaux dans le domaine de la recherche d'information ont traditionnellement concentré davantage sur la facilitation de l'accès à l'information plutôt que d'analyser l'information pour découvrir des motifs, ce qui est l'objectif principal de la fouille de textes.

La fouille de textes peut être considérée comme allant au-delà de l'accès à l'information pour aider les utilisateurs à analyser et digérer l'information et de faciliter la prise de décision. Il y a de nombreuses applications de la fouille de textes où l'objectif principal est d'analyser et de découvrir des motifs intéressants y compris les tendances et les valeurs aberrantes dans les données textuelles, et la notion d'une requête n'est pas indispensable, ni même pertinente [Aggarwal et Zhai, 2012a].

## 3.1.3 Caractéristiques des données textuelles

Il existe un nombre de caractéristiques qui distinguent les données textuelles des autres formes de données telles que les données relationnelles ou quantitatives. La caractéristique la plus importante de données textuelles est qu'elle sont clairsemée ‘‘*sparse*’’ et de grande dimension. Par exemple, un corpus de documents textuels donné peut être établi d'un lexique

d'environ 100000 mots, mais un document donné peut contenir seulement quelques centaines de mots différents. Ainsi, un corpus peut être représenté comme une *matrice creuse* de mot-document (*sparse matrix* en anglais) de taille $n \times d$, où $n$ est la taille du lexique et $d$ est le nombre de documents. Le $(i, j)$-ième entrée de cette matrice est la fréquence du $i$-ième mot du lexique dans le document $j$. C'est une matrice creuse, car pour la plupart des lignes (mots), la plupart des colonnes (documents) contiennent des valeurs nulles pour ces lignes et seulement quelques colonnes en avoir une valeur positive.

En outre, les données textuelles peuvent être analysées à différents niveaux de représentation. Un texte peut facilement être traité comme un *sac-de-mots*, ou comme une séquences de $n$ mots ou $n$ caractères consécutifs (*n-grammes*), comme nous verrons dans la Section 3.3. Cependant, dans un certain nombre d'applications il est souhaitable de représenter le texte *sémantiquement* pour permettre une analyse et une fouille plus significatives. Par exemple, la représentation des données textuelles au niveau des entités nommées (personnes, organisations, lieux, …) et leurs relations peuvent permettre la découverte de motifs plus intéressants que de représenter le texte comme un *sac-de-mots*. Cette représentation sémantique du texte sera détaillée dans la Section 3.5.

De plus, contrairement aux autres formes de données, les données textuelles regorgent d'imprécisions, d'ambiguïté et l'information utile est parfois dissimulée dans des tournures complexes. Par exemple, la phrase suivante :

"الإفراط في تناول السكريات الضرورية لنشاط الجسم يؤدي إلى زيادة الوزن"

exprime une relation de dépendance entre deux phénomènes, "الإفراط في تناول السكريات" et "زيادة الوزن". Cependant, il existe d'autre relation "cachée". Le terme :

"تناول السكريات الضرورية لنشاط الجسم"

peut exprimer ainsi une relation de dépendance entre "تناول السكريات" et "نشاط الجسم".

Les caractéristiques des données textuelles indiquent que ces données sont très différentes des données issues des bases de données ou de données, comme des mesures. Ces caractéristiques ont des implications immédiates sur certaines techniques d'analyse de données telles que la représentation de données et la réduction de la dimensionnalité.

## 3.2 Représentation des documents textuels

Les algorithmes d'apprentissage courants ne sont pas capables de traiter directement les données textuelles dans leur forme originale (non structurée). Par conséquent, il est nécessaire

de transformer chaque document en une représentation plus gérable qui lui remplace par un certain nombre de caractéristiques de leur contenu.

Généralement, un document textuel peut être représenté de deux manières différentes. La première méthode consiste à représenter un texte sous forme de séquences de *n* mots ou *n* caractères consécutifs (*n-grammes* de mots ou de caractères). Une telle représentation a l'avantage de conserver l'ordre des mots du texte. La deuxième méthode est d'utiliser une représentation simple appelée sac de mots (ou *bag-of-words* en anglais), dans laquelle chaque document est considéré comme un ensemble de mots (ou ses lemmes, racines) où l'ordre des mots, les combinaisons dans lesquelles ils apparaissent, les structure des paragraphes, les ponctuations et bien sûr les significations des mots sont tous ignorés.

La plupart des méthodes de classification de texte utilisent la représentation sac-de-mots en raison de sa simplicité à des fins de classification [Aggarwal et Zhai, 2012b]. Les processus de classification de documents étudiés dans ce chapitre (Sections 3.3 et 3.4), se basent sur ce type de représentation que nous adoptons aussi dans notre système de catégorisation (Chapitre 4). Etant donné un ensemble de documents textuels, la première tâche à effectuer est d'identifier l'ensemble des mots qui apparaissent au moins une fois dans tous les documents. Ces mots vont être filtrés pour construire un lexique (cette tâche sera expliquée dans la Section 3.2.3). Nous présentons dans cette section un ensemble de techniques nécessaires à la classification des documents textuels. Ces techniques sont issues en grande partie de la Recherche d'Information. Nous commençons par le modèle vectoriel pour la représentation de documents.

### 3.2.1 Modèle vectoriel

Après avoir choisir un type de représentation (dans notre cas, sac-de-mots) qui remplace chaque document par un certain nombre de caractéristiques, nous considérons dans le cadre du modèle vectoriel que chaque caractéristique est un *terme* d'un certain type (souvent un mot, un nombre ou un symbole, mais peut également être une paire de mots ou une phrases). Le modèle vectoriel consiste à représenter un document par un vecteur dans un espace vectoriel dont chaque dimension correspond à un terme du lexique. Ainsi, chaque composante du vecteur aura une valeur pour chaque terme appelée poids du terme et est en général une fonction de fréquence d'occurrence du terme dans le document. Si un terme n'est pas présent dans le document, la composante correspondante dans le vecteur sera zéro. Sinon, il y aura

une certaine valeur positive. Par conséquent, en utilisant l'occurrence de chaque terme comme poids, les termes qui apparaissent le plus fréquemment sont plus importants et donc descriptifs du document.

Ainsi, un corpus de *m* documents peut être représenté par une matrice de *n*×*m* éléments, où *n* est la taille de l'espace de représentation, c'est-à-dire du lexique, telle que représentée dans la Figure 3.1. Cette représentation est fréquemment nommée le modèle vectoriel de *Salton* [Salton et al., 1975], elle peut être utilisée pour effectuer diverses tâches automatiques de fouille de textes, en particulier pour les tâches de classification.

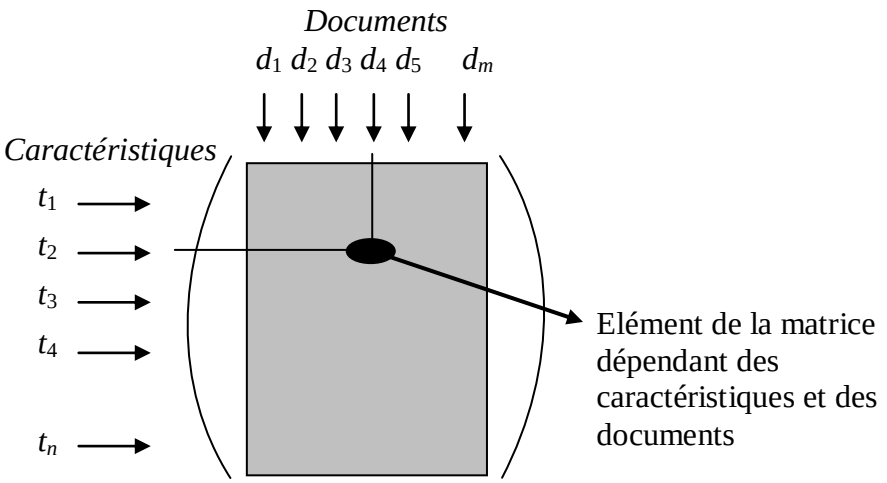



FIG. 3.1 – Représentation vectorielle d'un corpus de documents (figure extraite de [Béchet, 2009]).

Les textes brutes en lange naturelle subissent un certain nombre de transformations afin de se conformer au modèle vectoriel ; un ensemble de prétraitements linguistiques, suivi d'une étape de sélection des termes pertinents, permettent de construire un lexique de termes. Ensuite, il est nécessaire d'effectuer une pondération des termes sélectionnés afin de décrire le contenu du document de manière quantitative par l'intermédiaire d'un vecteur de caractéristiques. On commence par les prétraitements.

## 3.2.2 Prétraitements de textes

Les prétraitements ont pour but de transformer un texte brut en une liste de termes, ils sont décrits ci-dessous :

- segmentation en mots : qui consiste à découper la suite de caractères qui constituent le texte brut en une séquence de *tokens* (mots, mais aussi signes de ponctuation, symboles…).
- élimination de tout caractère qui ne correspond pas à une lettre de l'alphabet (points, virgules, traits d'union, chiffres etc.). Cette opération est motivée par le fait que ces caractères ne sont pas liés au contenu des documents et ne change rien au sens s'ils sont omis et par conséquent ils peuvent être négligés.
- normalisation, qui consiste pour un mot arabe à remplacer certaines lettres avec d'autres lettres selon un ensemble de règles prédéfinies, ou de supprimer les signes diacritiques.
- élimination des mots vides de sens (ou *stop-words* en englais), qui correspondent à tous les mots qui sont trop fréquents dans la base de documents (ils n'aident donc pas à discriminer entre les documents) ou jouent un rôle purement fonctionnel dans la construction des phrases (appelé les mots outils : articles, prépositions, conjonctions, déterminants, auxiliaires, etc.). Le but est alors de seulement conserver les mots ayant une signification représentative du texte.
- lemmatisation, qui consiste à remplacer chaque mot fléchi par sa forme canonique (lemme). Ainsi, les verbes par leur forme infinitive, les noms par leur forme au singulier etc.
- racinisation, qui consiste à extraire d'un mot son racine (ou pseudo-racine).

Les mots, les lemmes ou les racines seront appelés les caractéristiques textuelles des documents. La substitution des mots par leur racine ou leur lemme sont souvent combinés dans un seul processus nommé stemming. Le stemming peut améliorer considérablement les performances des systèmes de classification de textes en réduisant la dimension des vecteurs de caractéristiques représentant les documents.

La langue arabe est morphologiquement complexe. Cette complexité combine la dérivation et la flexion de façon étroite. La dérivation génère de nouveaux mots à partir d'une racine tout en restant dans le même champ sémantique. Par exemple, on peut dériver de la racine "قصد" les mots "يقصد", "مقاصد", "اقتصادية" et "الاقتصادي". La flexion en arabe peut être obtenue par différentes forme d'affixation (préfixe, suffixe, infixe). Par exemple, de la racine "علم" le pluriel de "عِلم" est "عُلوم". De plus, certains termes arabes peuvent être agglutinées (mots ou termes sont combinés), compliquant ainsi les méthodes d'analyse automatique.

Dans le contexte de classification de texte, les techniques du stemming des mots arabes suivent, généralement, deux approches : le stemming brut (communément appelé stemming) et le stemming léger. Le stemming brut transforme chaque mot dans le document à sa noyau (racine ou lemme). L'algorithme de Khoja [Khoja et Garside, 1999] figure parmi les méthodes les plus connues de cette approche. Cet algorithme est basé sur une analyse morphologique et l'utilisation des listes prédéfinies de racines. Il supprime les plus longs suffixes et préfixes. Il compare ensuite le mot restant avec des motifs verbaux et nominaux d'extraction de racine à l'aide d'un dictionnaire. L'algorithme de Khoja fait appel à plusieurs fichiers de données linguistiques comme une liste de tous les caractères diacritiques, caractères de ponctuation, articles définis et stops-words. L'inconvénient majeure de cette approche est que deux mots arabes sémantiquement différentes peuvent avoir le même stem. Par exemple, le stemming des deux mots donne le mot "رائع" et " روع " (horreur). Nous pouvons voir que la polarité de « رائع » (merveilleux) est inversée par le stemming [Mountassir et al., 2012]. Ainsi, le stemming brut de mots arabes peut affecter le sens de ces mots et par conséquent diminuer les performances en classification de texte.

L'approche du stemming léger [Larkey et al. 2007], en revanche, vise à améliorer la performance de classification du texte tout en conservant le sens des mots. Elle consiste simplement à supprimer un petit ensemble de préfixes et / ou suffixes du mot au lieu d'extraire la racine originale. Par exemple, le mot " المسافرون " (voyageurs) se transforme par le stemming léger en "مسافر " (voyageur) non pas en la racine "سفر " (voyage). Cependant, l'approche du stemming léger conduit à un grand nombre de caractéristiques.

## 3.2.4 Sélection de caractéristiques

Le nombre de mots différents est grand, même dans les documents relativement petits tels que des articles de presse ou des résumés d'articles scientifiques. La dimension de l'espace de caractéristiques sac-de-mots pour une grande collection peut atteindre des centaines de milliers, où la plupart de ces mots ne sont pas pertinents pour la discrimination entre les documents. Il est alors nécessaire d'effectuer une sélection de caractéristiques pertinentes en réduisant la dimension de l'espace de représentation avant d'entraîner un classifieur de documents.

Les méthodes de sélection de caractéristiques ont généralement deux objectifs. D'une part, elles réduisent le nombre de caractéristiques à modéliser de manière que le contenu du texte

est encore préservé, ce qui aide généralement à accélérer le processus d'apprentissage. D'autre part, elles filtrent les caractéristiques non pertinentes, il en résulte la création d'un modèle précis et efficace pour la classification de documents [Jiang, 2010].

Les prétraitements mentionnés précédemment (élimination des mots vides, stemming) sont considérés comme méthodes de sélection de caractéristiques, où les mots avec le même *stem* ou racine sont considérés comme une seule caractéristique, et celles disposant d'une fréquence plus élevée sont utilisées. Ces méthodes sont très utilisées à la fois dans les applications supervisées et non supervisées. Cependant, de nombreux systèmes effectuent un filtrage beaucoup plus agressif, en utilisant des mesures statistiques afin d'attribuer un score de pertinence à chaque caractéristique en fonction de son pouvoir discriminant. Il suffit alors de ne conserver que les caractéristiques dont le score est le plus élevé pour réduire considérablement la dimension de l'espace. Nous présentons ci-dessous quelques mesures statistiques fréquemment employées dans la littérature pour la sélection de caractéristiques pertinentes des données textuelles

### 3.2.4.1 Sélection à base de fréquence

Une première méthode statistique s'appuie simplement sur la fréquence d'occurrence d'un terme. La façon la plus simple consiste à définir un seuil sur la fréquence d'occurrence d'un terme donnée dans le corpus ou sur le nombre de documents indexés par ce terme. Cette méthode peuvent éliminer les termes peu ou trop fréquents. Les termes peu fréquents peuvent ne pas aider beaucoup aux calculs de similarité, et peuvent aussi ajouter certain bruit obscurcissant dans la tâche de classification. Quant aux termes trop fréquents, ils sont généralement des termes communs qui ne sont pas discriminants du point de vue de la classification.

Noteons que la méthode de pondération TF.IDF présentée en Section 3.2.3, peut éliminer de façon souple les termes non pertinents (ceux ayant une valeur TF.IDF faible), plutôt que d'utiliser des seuils de nombre d'occurrences. Un tel traitement peut donc se révéler plus pertinent particulièrement pour les processus fondés sur les N-grammes de caractères comme types de caractéristiques de texte, et qui n'utilisent pas de prétraitements (comme l'élimination des "stop words").

La sélection de caractéristiques est plus fréquente et facile à appliquer dans la tâche de classification automatique de texte avec apprentissage supervisé (catégorisation), dans

laquelle une supervision est disponible pour le processus de sélection de caractéristiques par l'utilisation des étiquettes de classes. Ce processus de sélection supervisée assure le choix des caractéristiques qui sont beaucoup plus susceptibles d'être en corrélation à la distribution d'une classe particulière que d'autres. Il existe plusieurs méthodes de sélection supervisée des caractéristiques pertinentes qui ont été largement utilisées dans la classification de texte [Sebastiani, 2002]. Elles utilisent des mesures statistiques plus complexes que le nombre d'occurrence d'une caractéristique (terme) et prennent en compte les relations entre les caractéristiques et les classes des documents. Nous présentons ci-dessous deux méthodes de sélection : le chi-carré ($\chi^2$) et le gain d'information (IG).

### 3.2.4.2 Le gain d'information (IG)

Cette méthode mesure la quantité d'information apportée pour une classe par la connaissance de l'apparition ou non d'une caractéristique (terme) dans un document. Plus précisément, l'IG d'une caractéristique *t* sur une classe *c* peut être exprimé comme suit :

$$IG(t,c) = \sum_{c' \in (c,\bar{c})} \sum_{t' \in (t,\bar{t})} P(t',c') \log \frac{P(t',c')}{P(t')P(c')}, \qquad (3.8)$$

où $P(c')$ et $P(t')$ dénotent la probabilité qu'un document appartient à la classe *c'* et la probabilité qu'une caractéristique *t'* apparaît dans un document, respectivement, et $P(t', c')$ est la probabilité jointe de *t'* et *c'*. Toutes les probabilités peuvent être estimées par le calcul de la fréquence à partir des données d'apprentissage [Jiang, 2010]. Il suffit alors de ne conserver que les *N* caractéristiques ayant le meilleur score.

### 3.2.4.3 Le chi-carré ($\chi^2$)

La méthode du chi-carré ($\chi^2$), appelée également chi-deux, mesure l'indépendance entre une caractéristique *t* et une classe *c*. En d'autres termes, les caractéristiques sont classées par rapport à la quantité $\chi^2$ calculée comme suit :

$$\chi^2(t,c) = \frac{n\left(P(t,c)P(\bar{t},\bar{c}) - P(t,\bar{c})P(\bar{t},c)\right)^2}{P(t)P(\bar{t})P(c)P(\bar{c})}, \qquad (3.9)$$

où *n* est le nombre total de documents. Les notations de probabilité ont les mêmes interprétations comme dans l'équation (3.8). Par exemple, $P(\bar{c})$ représente la probabilité qu'un document n'appartient pas à la classe *c*. Plus la valeur du $\chi^2(t, c)$ est grande,

plus *t* et *c* semblent dépendants l'un de l'autre [Jiang, 2010]. Classiquement, deux scores sont utilisés en sélection de caractéristiques en se basant sur la méthode du $\chi^2$. Soit $c_1$, $c_2$, … , $c_k$ l'ensemble des classes, les deux scores sont définis comme suit:

$$\chi^2_{moy}(t) = \sum_{i=1}^{k} P(c_i) X^2(t, c_i)$$

$$\chi^2_{\max}(t) = \max_{i=1}^{k} \left\{ X^2(t, c_i) \right\}$$

De même que pour le gain d'information, il suffit là encore de ne conserver que les *N* caractéristiques ayant le meilleur score.

## 3.2.3 Pondération des termes

Une représentation d'un texte en sac de mots s'appuie sur une pondération choisie de chacun des termes du texte. Si l'on considère que les mots d'un texte n'ont pas la même représentativité de ce texte, il est nécessaire d'établir une stratégie de pondération adéquate pour établir une bonne mesure de similarité entre les textes. Le poids d'un terme dans un document peut être obtenu de différentes manières : booléenne, fréquence d'apparition et *TF.IDF*.

**Méthode booléenne**

Si un terme est présent dans un document alors la valeur qui lui correspond vaut 1, sinon 0. L'approche booléenne est utilisée lorsque chaque terme est d'égale importance et les documents sont de petites tailles.

**Fréquence des termes** (*TF*)

Le *TF* prend en compte le nombre d'occurrences d'un terme dans un document. Cette mesure repose sur l'idée que plus un terme apparaît dans un document, plus il est important. Soit $n_{i,j}$ le nombre d'occurrences du terme $t_i$ dans le document $d_j$ , nous avons alors :

$$TF(i, j) = n_{i,j} \qquad (3.10)$$

Pour limiter l'impact de la taille des documents, on normalise la valeur précédente en comptant la proportion de chaque terme :

$$TF(i, j) = \frac{n_{i,j}}{\sum_k n_{k,j}} \qquad (3.11)$$

Le dénominateur est la somme du nombre d'occurrences pour chaque terme dans le document $d_j$.

***TF.IDF***

Le *TF.IDF* (pour "*Term Frequency*" et "*Inverse Document Frequency*") [Salton et Buckley, 1988] est l'approche de pondération la plus populaire en fouille de textes. Cette mesure consiste à calculer l'importance d'un terme dans un texte relativement à une collection. Or, l'importance d'un terme dans un texte sera d'autant plus grande que celui-ci apparaît beaucoup dans ce texte mais peu dans les autres. C'est cette combinaison particulière que cherche à capturer la pondération *TF.IDF*. Ainsi, un terme présent dans tous les documents d'une collection aura un poids moindre.

Dans ce cas, le poids d'un terme *i* dans un document *j* est calculé de la manière suivante : d'abord, la fréquence du terme *TF* est calculée (formule (3.10) ou bien (3.11)). Cette fréquence peut déterminer l'importance et/ou la représentativité d'un terme dans un texte. Ensuite, on calcule l'*IDF*. Comme son nom l'indique l'*IDF* mesure l'inverse de la fréquence d'un terme dans l'ensemble des documents (formule (3.6)). Il permet de donner un poids plus important aux termes les plus discriminants, c'est-à-dire les moins fréquents dans l'ensemble des documents du corpus.

$$IDF(i) = \log(\frac{m}{|\{d_j \mid t_i \in d_j\}|}) \qquad (3.6)$$

– $m$ : nombre de documents dans le corpus.

– $|\{d_j \mid t_i \in d_j\}|$ : nombre de documents dans lesquels le terme $t_i$ apparaît.

Ce poids représente en quelque sorte la rareté du terme $t_i$ dans la collection. En effet, si l'attribut apparaît dans tous les documents, alors il ne permet pas de distinguer un texte d'un autre, il est donc neutralisé (sa rareté est nulle) : $IDF(i) = \log(m/m) = \log(1) = 0$. Si au contraire il est très rare en n'étant présent que dans un seul texte (c'est le minimum), alors il vaut $IDF(j) = \log(m/1) = \log(m)$, sa valeur maximale.

Enfin, le produit (formule (3.7)) de ces deux critères est calculé afin d'obtenir une valeur globale:

$$TF.IDF = TF(i, j) \times IDF(j) \qquad (3.7)$$

## 3.2.5 La mesure de similarité

Pour faire face aux tâches de la fouille de textes: la catégorisation, la segmentation et la recherche d'information, il est nécessaire de définir une mesure de similarité entre les

documents, en comparant la proximité des différents vecteurs (issus des matrices de Salton, LSA), afin par exemple de regrouper des documents. Ainsi, une telle proximité correspondra à des vecteurs proches. En d'autres termes, ces vecteurs auront des directions semblables ou bien encore leurs extrémités proches. La mesure de similarité la plus couramment utilisée dans la fouille de textes et la recherche d'information est le calcul du *cosinus* de l'angle formé par deux vecteurs de documents. Soit les documents *d* et *q* représentés respectivement par les vecteurs *u* et *v*. Alors, la similarité entre les deux documents *d* et *q* est calculée par :

$$sim(d,q) = \frac{u.v}{|u| \times |v|} \tag{3.10}$$

où *u.v* représente le produit scalaire des vecteurs *u* et *v*.
||*u*|| et ||*v*|| représentent respectivement les normes de *u* et *v*.

Les résultats obtenus avec cette mesure varient entre 0 et 1. Un score de 1 signifie que l'angle formé entre les deux vecteurs est très faible, indiquant une forte similarité des documents représentés par ces vecteurs.

## 3.2.6 La réduction / projection

Tandis que la sélection de caractéristiques tente à réduire la dimension des données en attribuant un score à chaque terme d'une matrice de *Salton* puis la sélection des termes les plus pertinents, les méthodes de transformation de caractéristiques créent un nouvel (et plus petit) ensemble de caractéristiques en projetant les composantes de la matrice originale dans un espace vectoriel réduit.

**Latent Semantic Analysis (LSA)**

La méthode LSA se fonde sur le fait que des mots qui apparaissent dans un même contexte sont sémantiquement proches. Le corpus est représenté sous forme matricielle. Les lignes sont relatives aux termes et les colonnes représentent les différents contextes choisis (un document, un paragraphe, une phrase, etc.). Chaque cellule de la matrice représente le nombre d'occurrences des termes dans chacun des contextes du corpus. Deux termes proches au niveau sémantique sont représentés par des vecteurs proches. La mesure de proximité est généralement définie par le cosinus de l'angle entre les deux vecteurs [Béchet, 2009].

La théorie sur laquelle s'appuie LSA est la décomposition en valeurs singulières (SVD). Une matrice $A = [a_{ij}]$ où $a_{ij}$ est la fréquence d'apparition du terme *i* dans le contexte *j*, se décompose en un produit de trois matrices *T, S et* $D^t$. *T* et *D* sont des matrices orthogonales et

$S$ une matrice diagonale contenant les valeurs singulières de $A$. La Figure 3.2 représente le schéma d'une telle décomposition où $r$ représente le rang de la matrice $A$.

Soit $S_k$ où $k < r$ la matrice produite en enlevant de $S$ les $r - k$ colonnes qui ont les plus petites valeurs singulières. Soit $T_k$ et $D_k$ les matrices obtenues en enlevant les colonnes correspondantes des matrices $T$ et $D$. La matrice $T_k S_k D^t_k$ peut alors être considérée comme une version compressée de la matrice originale $A$.

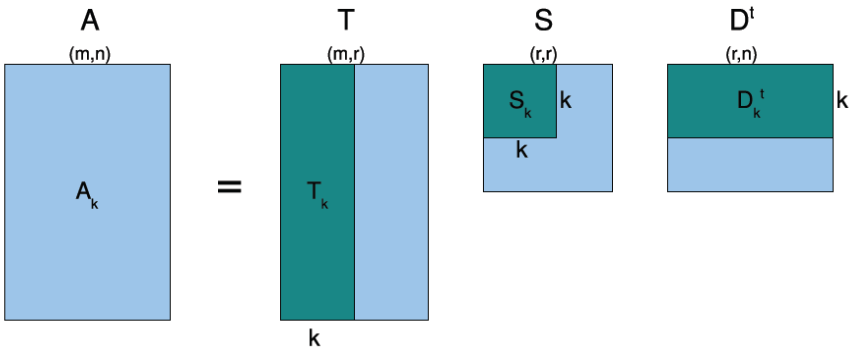



FIG. 3.2 – Décomposition en valeurs singulières.

# 3.3 Catégorisation automatique de documents textuels

Le principe de la classification de documents (comme un paragraphe ou un texte) est d'utiliser un modèle afin d'attribuer à un document une ou plusieurs classes. Nous pouvons distinguer deux types de modèles de classification : ceux nécessitant une phase d'apprentissage et ceux sans phase d'apprentissage. Parmi les modèles utilisant un apprentissage, nous distinguons l'apprentissage supervisé et non supervisé. Nous présentons dans cette section la classification supervisée de documents textuels (catégorisation). La classification non supervisée (segmentation) sera présentée dans la section 3.5.

## 3.3.1 Définition

La catégorisation automatique de documents textuels consiste à attribuer à un document donné un ou plusieurs étiquettes à partir d'un ensemble prédéterminé d'étiquettes. Par exemple, la classification de documents manuscrits aux différentes catégories ou classes telles que : l'arts, l'éducation, la science, etc. ou la classification d'articles de presse en fonction de leurs sujets. En général, on peut considérer les différentes propriétés d'un document et les combiner pour définir les classes, comme le type de document, les auteurs, le thème, etc.

Cette catégorisation est réalisée en grande partie grâce à des outils issus de la recherche d'information, permettant de représenter les documents sous forme vectorielle. Grâce à cette représentation, des techniques classiques d'apprentissage automatique peuvent être employées pour construire un classifieur de documents. Ainsi, comme tous les systèmes de reconnaissance de formes, un système de catégorisation est constitué de trois composantes principales :

- les prétraitements de documents ont pour but de convertir le flux de caractères entrants en un flux de termes,
- l'extraction des caractéristiques (pondération des termes) pour représenter les documents sous forme de vecteurs de termes,
- un classifieur de documents.

Les deux premières composantes qui conduisent à la représentation des données sont détaillées dans la Section 3.2. La troisième composante sera étudiée dans la Section 3.3.3.

Le processus de construction d'un système de catégorisation de documents se divise en trois phases : apprentissage, validation et teste. L'apprentissage supervisé suppose l'existence d'un ensemble de documents d'apprentissage dont chaque document est étiqueté avec une catégorie prédéfinie. L'objectif de cette phase est de créer des modèles de classification à partir d'ensemble d'apprentissage: l'algorithme d'apprentissage extrait les éléments caractérisant les différentes catégories afin de pouvoir affecter tout nouveau document à une catégorie en fonction de ses caractéristiques. La qualité des modèles de classification créés lors de la phase d'apprentissage dépend d'un ensemble de paramètres qu'il convient d'ajuster. Pendant la phase de validation, les paramètres du modèle de classification sont ajustés grâce à un ensemble de validation qui est souvent un sous-ensemble de l'ensemble d'apprentissage.

La phase de test a pour objectif l'évaluation du modèle de classification. L'ensemble de teste utilisé dans cette étape contient des documents pré-étiquetés dont l'information associée à la catégorie est utilisée pour estimer la qualité des réponses données par le classifieur.

### 3.3.2 Applications de la catégorisation de textes

La catégorisation de texte possède de nombreuses applications. Nous pouvons citer quelques domaines de fouille de textes dans lesquels la catégorisation est couramment utilisées [Aggarwal et Zhai, 2012b] :

***Filtrage et organisation d'articles de presse***

La plupart des services de presse d'aujourd'hui sont de nature électronique dans lequel un grand volume d'articles de presse sont crées chaque jour par les organisations. Dans un tel cas, il est difficile d'organiser les articles de presse manuellement. Par conséquent, des méthodes automatisées peuvent être très utiles pour catégoriser ces articles dans une variété de portails web. Cette application est également appelée *filtrage de texte*.

***Organisation et recherche de documents***

Une variété de méthodes de catégorisation peuvent être utilisées pour l'organisation des documents dans de nombreux domaines. Ils comprennent des grandes bibliothèques de documents numériques, des collections web, la littérature scientifique ou même des flux de réseaux sociaux. Les collections de documents organisés hiérarchiquement peuvent être particulièrement utiles pour la navigation et la recherche.

Un exemple d'application de ce type de problème de catégorisation est le processus de déclassification (lever le secret-défense) de grandes quantités de documents produits à l'origine par le gouvernement fédéral américain. Pour des raisons de sécurité nationale, il existe aujourd'hui un très grand nombre de documents qui restent classifiés comme secrets. Ces documents sont gardés dans des endroits sécurisés, car ils ont été considérés comme importants pour la sécurité nationale. Cependant, les coûts de maintenance élevés et de nouvelles lois exigent que ces documents devraient être réévalués, et ceux qui ne sont pas plus critiques devraient devenir publiques. Ainsi, ces documents doivent être (grosso modo) classés dans les deux catégories disjointes "secret" et "non secret" [Triantaphyllou, 2010].

***Classification d'emails et filtrage de Spams***

Il est souvent souhaitable de classifier les courriers électroniques, afin de déterminer leurs sujets ou pour déterminer ceux qui sont indésirables de façon automatisée. Ceci est également appelée *filtrage de spams* ou *filtrage de courriers électroniques.*

***fouille d'opinion***

L'accès à Internet permet à un grand nombre de personnes d'exprimer leur avis sur le web social (sites, forums de discussion, blogs, réseaux sociaux, etc.). La fouille d'opinion permet d'extraire rapidement de manière automatisée, les opinions des gens dans des données textuelles (messages, courtes textes, documents...). Cette tâche peut être considérée comme un problème de catégorisation automatique de textes où les catégories correspondent à des opinions (par exemple : positive, négative ou neutre) exprimés sur de produits ou services

d'entreprise, des hommes politiques, des lois, des questions de société, etc. Les avis ou les opinions des clients sont souvent des courts textes qui peuvent être minés pour déterminer les informations utiles à partir de l'avis.

### 3.3.3 Approches de catégorisation de textes

Dans les sections suivantes, nous nous intéressons aux algorithmes d'apprentissage supervisé employés dans le cadre de catégorisation de données textuelles. Ces algorithmes existent aussi pour d'autres types de données telles que les données quantitatives. Puisque le texte peut être représenté par des données quantitatives sous forme des fréquences de mots, il est possible d'utiliser la plupart des algorithmes de classification des données quantitatives directement sur des données textuelles.

Le domaine de la catégorisation de textes est si vaste qu'il est impossible de couvrir tous les algorithmes en détail dans un chapitre unique. Par conséquent, nous avons choisi de présenter ci-dessous un nombre non exhaustif des algorithmes utilisées actuellement. Le lecteur peut se référer à [Sebastiani, 2002] qui présente un large survey de différentes méthodes de catégorisation de textes.

Nous distinguons ici quelques types de classifieurs :

- les *k*-plus proches voisins.
- la méthode de Bayes naïf (Naive Bayes).
- les séparateurs à vastes marges (SVM).
- les classifieurs fondés sur des réseaux de neurones artificiels.

#### 3.3.3.1 Les *k*-plus proches voisins

Le principe de l'algorithme des *k*-ppv est de mesurer la similarité (cosinus, distance euclidienne, etc.) entre la représentation vectorielle d'un nouveau document et celles des documents ayant été préalablement classés. Ces documents peuvent être considérés comme un modèle d'apprentissage, bien qu'il n'y ait pas de réelle phase d'apprentissage avec les *k*-ppv. Les documents classés sont ordonnés de manière décroissante afin que le premier document soit celui ayant obtenu le meilleur score de similarité avec le document devant être classé. Les *k* premiers documents sont gardé (les *k*-plus proches voisins).

Il faut définir une méthodologie afin d'attribuer une classe au nouveau document. Il existe dans la littérature deux approches classiques décrites spécifiquement dans [Bergo, 2001] afin de répondre à cette problématique :

- soit proposer de classer le document dans la même catégorie que celui ayant obtenu le meilleur score de similarité parmi le jeu d'apprentissage,
- soit, si $k > 1$ de considérer les $k$ documents les mieux classés. Alors nous pouvons attribuer la classe suivant plusieurs options. Une première méthode peut être de calculer parmi les $k$ documents les plus proches, pour chaque catégorie, le nombre de documents appartenant à cette catégorie (1). Une seconde propose de prendre en compte le rang des $k$ documents (2). Il s'agit pour toutes les catégories, d'effectuer la somme des occurrences d'une catégorie multipliée par l'inverse de son rang.

Prenons par exemple un document $d_{new}$ à classer parmi quatre classes, C1, C2, C3 et C4. Définissons $k = 6$. Considérons le classement suivant de $d_{new}$ avec le jeu d'apprentissage $D$ contenant les documents $d_i$ :

| documents | Classe de documents | rang |
|---|---|---|
| d1 | C2 | 1 |
| d2 | C2 | 2 |
| d3 | C4 | 3 |
| d4 | C4 | 4 |
| d5 | C1 | 5 |
| d6 | C4 | 6 |

En utilisant la première méthode (1), nous aurions attribué la classe C4 à $d_{new}$. En effet la classe C4 est celle qui possède le plus de documents parmi les $k$ plus proches voisins (trois documents). La seconde méthode (2) aurait quant à elle classé $d_{new}$ dans C2. Nous obtenons en effet avec cette mesure par exemple pour la classe C1 : un seul document dans la classe au cinquième rang soit C1 = 1/5 = 0,2. Nous obtenons pour les autres classes C2 = 1,5, C3 = 0 et C4 = 0,75.

## 3.3.3.2 La méthode de Bayes naïf (Naive Bayes)

Un classifieur Naïve Bayes est un classifieur de type probabiliste fondé sur le théorème de Bayes (1763). Considérons $v_j = (v_{j1}, ..., v_{jk}, ..., v_{jd})$ un vecteur de variables aléatoires

représentant un document $d_j$ et $C$ un ensemble de classes. En s'appuyant sur le théorème de Bayes, la probabilité que ce dernier appartienne à la classes $c_i \in C$ est définie par:

$$P(c_i \mid v_j) = \frac{P(c_i)P(v_j \mid c_i)}{P(v_j)} \tag{3.11}$$

La variable aléatoire $v_{jk}$ du vecteur $v_j$ représente l'occurrence de l'unité linguistique $k$ retenue pour la classification dans le document $d_j$. La classe $c_r$ d'appartenance de la représentation vectorielle $v_j$ d'un document $d_j$ est définie comme suit :

$$c_r = \arg\max P(c_i \in C)\prod_k P(v_{jk} \mid c_j) \tag{3.12}$$

En d'autres termes, le classifieur Naïve Bayes affecte au document $d_j$ la classe ayant obtenue la probabilité d'appartenance la plus élevée.
Alors, $p(c_i)$ est définie de la façon suivante :

$$P(c_i) = \frac{nombre\ de\ documents \in c_i}{nombre\ total\ de\ documents} \tag{3.13}$$

En faisant l'hypothèse que les $v_j$ sont indépendantes, la probabilité conditionnelle $P(v_j|c_i)$ est définie ainsi:

$$P(v_j \mid c_i) = P(v_{jk} \mid c_i) \tag{3.14}$$

Une telle hypothèse d'indépendance des $v_j$ peut néanmoins dégrader qualitativement les résultats obtenus avec une telle approche [Lewis, 1998].

### 3.3.3.3 Les séparateurs à vastes marges (SVM)

L'algorithme des SVM est originalement un algorithme mono-classe permettant de déterminer si un élément appartient (qualifié de positif) ou non (qualifié de négatif) à une classe. L'idée principale de cet algorithme est de trouver un hyperplan qui sépare au mieux les exemples positifs des exemples négatifs en garantissant que la marge entre le plus proche des positifs et des négatifs soit maximale. Intuitivement, cela garantit un bon niveau de généralisation car de nouveaux exemples pourront ne pas être trop similaires à ceux utilisés pour trouver l'hyperplan, mais être situés franchement d'un côté ou de l'autre de la frontière. SVM est considéré comme un des algorithmes les plus performants en classification textuelle [Laroum et al., 2010].

### 3.3.3.4 Les classifieurs fondés sur des réseaux de neurones

Les approches neuronales furent dans les premières à être utilisées afin de réaliser un apprentissage. Ces approches s'inspirent du fonctionnement du système nerveux humain. Ainsi, elles se fondent sur l'utilisation de "neurones" qui vont effectuer la tâche d'apprentissage. Chaque unité reçoit un ensemble d'entrées, qui sont désignées par le vecteur $\overline{X}_i$, qui correspond dans ce cas, aux fréquences des termes dans le *i*-ème document. Chaque neurone est également associé à un ensemble de poids synaptiques *A*, qui sont utilisés afin de calculer une fonction *f*( ) de ses entrées. Une fonction typique qui est souvent utilisée dans le réseau de neurones est une fonction linéaire de la manière suivante:

$$p_i = A \cdot \overline{X}_i \qquad (3.15)$$

Ainsi, pour un vecteur $\overline{X}_i$ tiré d'un lexique de *d* mots, le vecteur de poids *A* devrait également contenir *d* éléments. Considérons maintenant un problème de classification binaire, dans lequel les étiquettes des classes appartiennent à l'ensemble {1, -1}. On suppose que l'étiquette de classe de $\overline{X}_i$ est notée $y_i$. Dans ce cas, le signe de la fonction prédite $p_i$ donne l'étiquette de classe.

L'objectif de cette approche est d'apprendre l'ensemble de poids *A* par l'utilisation des données d'apprentissage. L'idée est de commencer avec des poids aléatoires et les mettre à jour progressivement quand une erreur est faite en appliquant la fonction courante sur l'exemple d'apprentissage. Le grandeur de la mise à jour est réglée par un taux d'apprentissage $\mu$. Ceci constitue l'idée de base de l'algorithme de perceptron, qui est comme suit:

**Algorithme de Perceptron**

**Entrées :** un taux d'apprentissage $\mu$.

Données d'apprentissage ( $\overline{X}_i$ , $y_i$) $\forall i \in \{1...n\}$

Initialiser les poids dans *A* à 0 ou petits nombres aléatoires

**répéter**

Appliquer chaque donnée d'apprentissage au réseau de neurone pour vérifier si le signe de $A \cdot \overline{X}_i$ correspond à $y_i$;

Si le signe de $A \cdot \overline{X}_i$ ne correspond pas à $y_i$, alors

mettre à jour les poids *A* en fonction du taux d'apprentissage $\mu$

**jusqu'à** ce que les poids dans *A* convergent.

Les poids dans *A* sont généralement mis à jour (augmentation ou diminution) proportionnellement à $\mu \cdot \overline{X}_i$, de manière à réduire le sens de l'erreur du neurone. Nous notons en outre que de nombreuses règles de mise à jour ont été proposées dans la littérature. Par exemple, on peut simplement mettre à jour chaque poids par $\mu$, plutôt que par $\mu \cdot \overline{X}_i$. Ceci est particulièrement possible dans des domaines tels que le texte, dans lequel toutes les caractéristiques prennent des petites valeurs non négatives de grandeurs relativement similaires [Aggarwal et Zhai, 2012b].

## 3.3.4 Les mesures d'évaluation

L'évaluation d'un algorithme de classification se fait toujours sur un certain nombre de données pour lesquelles la classe "correcte" est supposée connue. Pour une catégorie *c* donnée, *S* est l'ensemble des documents identifiés par un algorithme comme faisant partie de *c* et *R* est l'ensemble des documents faisant réellement partie de *c*. La Figure 3.3 schématise ces deux ensembles.

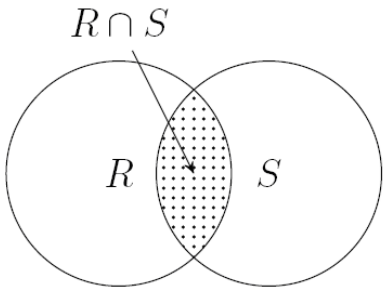



FIG. 3.3 - Ensemble de documents supposés pertinents (*S*) et ensemble de documents réellement pertinents (*R*) pour une catégorie donnée.

Les mesures classiques d'évaluation de la qualité d'un algorithme de classification sont la précision ($\pi$) et le rappel ($\rho$). Ces mesures peuvent être dérivées à partir des ensembles illustrés sur la Figure 3.3. La précision mesure la proportion de documents correctement classés dans *c* parmi les documents de *S*. La précision évalue la capacité du classifieur à ne pas introduire de documents d'une autre catégorie dans *c*. Elle est définie par la formule :

$$\pi(c) = \frac{|R \cap S|}{S},$$

Le rappel mesure la proportion de documents bien classés dans *c* parmi les documents de *R*. Le rappel évalue la capacité du classifieur à trouver tous les documents de *c*. Elle est définie par :

$$\rho(c) = \frac{|R \cap S|}{R},$$

Ces deux mesures permettent de tracer une courbe de précision vs rappel interpolée, ce qui permet une représentation visuelle des performances. Ces courbes sont généralement tracées en fonction de 11 points standards. Ces points correspondent à la précision lorsque le rappel vaut [0, 0.1, 0.2, . . . , 1.0] (un exemple est donné sur la Figure 3.4).

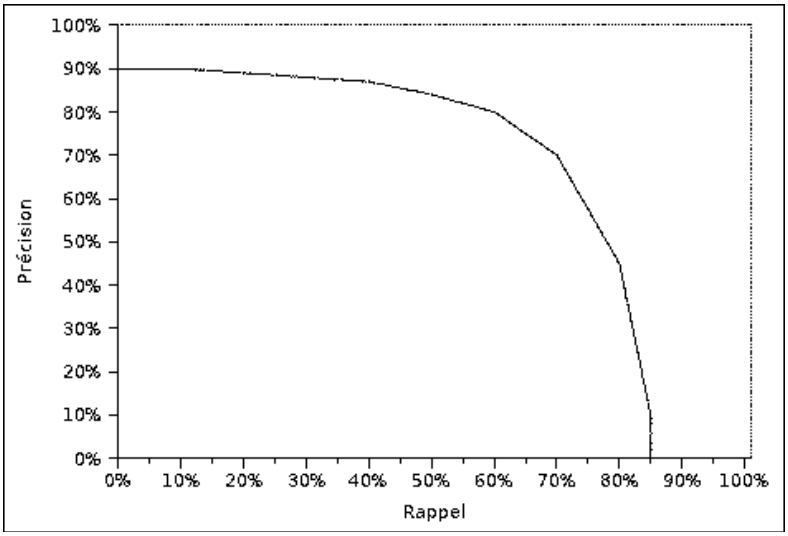



FIG. 3.4 - Exemple de courbe rappel/précision.

Il peut être utile de pouvoir résumer les performances des systèmes de classification par une seule mesure qui combine les deux mesure précédentes (précision et rappel). Pour cela, la F-mesure est souvent utilisée. C'est un indicateur prenant en compte la valeur relative de la précision et du rappel, défini par :

$$F_\beta(c) = \frac{(1+\beta^2) \times \pi(c) \times \rho(c)}{\beta^2 \times \pi(c) + \rho(c)},$$

Une valeur de $\beta > 1$ donne plus de valeur à $\rho$(c), une valeur comprise entre $0 < \beta < 1$ donne plus de valeur à $\pi$(c). Dans la littérature, les résultats sont souvent exposés avec $\beta = 1$. On appelle alors cette mesure la $F_1$-mesure.

Une autre façon de résumer les performances des systèmes est le «Break-Even Point» (BEP). Il s'agit, sur la courbe «Rappel/Précision», du point où le rappel est égal à la précision. Ceci permet une comparaison rapide, mais sommaire, des performances des différents systèmes.

L'évaluation des algorithmes se faisant sur plusieurs catégories, il est nécessaire de résumer les mesures faites sur chaque catégorie en une seule valeur moyenne. Il y a deux façons de moyenner les valeurs de la précision et du rappel (donc de la F-mesure et du BEP) : la macro et la micro-moyenne. La macro-moyenne est donnée par la moyenne simple des mesures par classe. Chaque catégorie a la même influence sur la moyenne. La macro-moyenne de la précision et du rappel est définie par :

$$\pi^M = \frac{\sum_{c=1}^{|C|} \pi(c)}{|C|}$$

$$\rho^M = \frac{\sum_{c=1}^{|C|} \rho(c)}{|C|}$$

D'autre part, la micro-moyenne est calculée à partir de la somme des effectifs des différents classes. Dans ce cas chaque document a la même influence sur la moyenne. La micro-moyenne de la précision et du rappel est donnée par :

$$\pi^\mu = \frac{\sum_{c=1}^{|C|} |R_c \cap S_c|}{\sum_{c=1}^{|C|} |S_c|}$$

$$\rho^\mu = \frac{\sum_{c=1}^{|C|} |R_c \cap S_c|}{\sum_{c=1}^{|C|} |R_c|}$$

La micro-moyenne tient compte de la répartition des données, alors que la macro-moyenne donne autant d'importance à chaque catégorie, indépendamment de ses effectifs. Lorsque le système de catégorisation est évalué dans une perspective centrée catégorie, des courbes de précision vs rappel données en macro-moyenne serons présentées. Dans une perspective centrée-document, l'évaluation doit être faite en fonction des documents, c'est-à-dire en utilisant la micro-moyenne. Dans cette perspective lorsque la catégorisation est mono-étiquette, les micro-moyennes de la précision et du rappel sont égales [Beney, 2008]. Dans ce cas, une seule mesure appelée *taux de classification*, sera alors utilisée pour l'évaluation du système.

## 3.4 Segmentation automatique de textes

La classification avec apprentissage non supervisé (ou segmentation) se différencie de l'approche supervisée avec la connaissance ou non des classes à prédire. De plus, aucune donnée étiquetée n'est disponible. Il n'est alors pas possible de générer un modèle d'apprentissage. Ainsi, la tâche ne consiste plus seulement à attribuer une classe à un nouvel élément (approche supervisée) mais également à définir les classes et leurs nombres. Notons que ces classes sont souvent appelées : groupes, partitions ou clusters.

### 3.4.1 Définition

Étant donné un ensemble *S* de *n* documents. La segmentation de documents est définie comme étant la division de l'ensemble des documents du corpus en *k* classes (ou groupes) de telle sorte que les documents d'un même groupe soient plus similaires entre eux qu'avec les documents des autres groupes. La similarité entre les documents est mesurée avec l'utilisation d'une fonction de similarité.

Le problème de segmentation peut être très utile dans le domaine des données textuelles, où les objets à classifier peuvent être de différentes granularités tels que des documents, des paragraphes, des phrases ou des termes. La segmentation est particulièrement utile pour l'organisation des documents afin d'améliorer la recherche et supporter la navigation. La segmentation de documents fait face à deux grands défis: la dimension de l'espace de caractéristiques tend à être élevée (c'est à dire un ensemble de documents est souvent constitué de milliers ou des dizaines de milliers mots uniques) et la taille d'une collection de documents tend à être grande.

### 3.4.2 Approches de segmentation

La plupart des algorithmes de segmentation de documents se fondent sur le calcul de distance pour mesurer la similarité entre les documents. Dans cette section, nous nous focalisons sur trois approches de segmentation à base de distance qui ont été largement utilisées pour la segmentation de documents : les algorithmes ascendants hiérarchiques (ou agglomératifs), les algorithmes des *k*-moyennes et l'approche hybride de *Scatter-Gather*.

### 3.4.2.1 Algorithmes de segmentation Ascendante Hiérarchique

La Classification Ascendante Hiérarchique (CAH), appelée également agglomératif, est particulièrement utilisée pour supporter une variété de méthodes de recherche d'information, car elle crée naturellement une hiérarchie arborescente qui peut être mise à profit pour le processus de recherche [Aggarwal et Zhai, 2012b].

L'algorithme de (CAH) regroupe les documents de façon hiérarchique. Il commence avec chaque document dans un cluster (groupe) distinct. A chaque itération, il procède à fusionner successivement les paires de clusters qui sont les plus similaires entre eux selon un critère choisi et recalcule une similarité entre ce nouveau cluster et les autres. L'algorithme se termine lorsque on obtient un seul cluster contenant tous les documents.

L'historique du processus de fusion se traduit par un arbre binaire de la hiérarchie des clusters (ou *dendrogramme*) dont les feuilles correspondent aux documents et dont les nœuds correspondent aux clusters fusionnés. La Figure 3.5 montre un exemple d'un *dendrogramme*. Lorsque deux clusters sont fusionnés, un nouveau nœud est crée dans cet arbre correspondant à ce cluster fusionné. Les deux fils de ce nœud correspondent aux deux clusters de documents qui ont été y fusionnés pour lui. Un ensemble de clusters s'obtient en coupant le *dendrogramme* à un certain niveau d'agglomération. C'est parce que cette technique part du particulier pour remonter au général qu'elle est dite « *ascendante* » ou *agglomérative*.

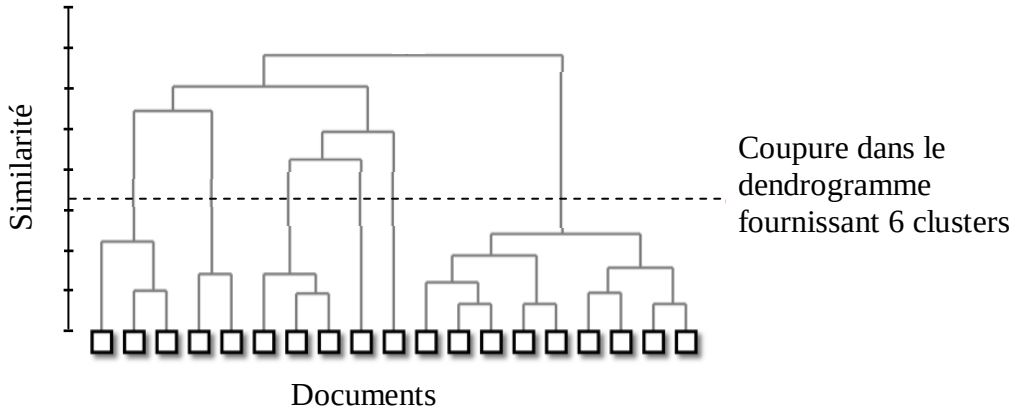



Fig. 3.5– Dendrogramme.

La principale source de variation entre les différentes méthodes agglomératives proposées dans la littérature est le choix de la mesure de similarité entre deux clusters permettant de sélectionner quelle paire de clusters à fusionner à chaque étape du processus. De telle mesure

de proximité est appelée *critère du fusion* ou *critère du lien (linkage metrics).* Les plus importantes métriques de liens inter-cluster sont: le critère du lien simple *(single link),* critère du lien complet *(complete link)* et critère du lien moyen *(average link)*.

Le critère du *lien simple* mesure la similarité entre deux clusters de documents par la similarité maximale entre un document appartenant au premier cluster et un document appartenant au deuxième cluster. On ordonne les liens par ordre décroissant. Le premier lien qui lie deux documents appartenant à deux clusters différents fusionne ces clusters. Comme il suffit d'un lien, le critère est appelé lien simple. Le critère du lien complet utilise la similarité minimale. Les liens sont ordonnés par ordre décroissant. Pour deux clusters disjoints, ce n'est que lorsque tous les documents de l'un sont liés à tous les documents de l'autre que les clusters sont fusionnés. Comme tous les liens sont nécessaires, ce critère est nommé lien complet.

En général, les deux critères précédents ne sont pas très efficaces, car leurs décisions sont basées soit sur une quantité limitée d'informations (*lien simple*), ou sur la supposition que tous les documents de cluster sont très similaires les uns aux autres (*lien complet*). D'autre part, le critère du *lien moyen* mesure la similarité entre deux clusters par la moyenne de toutes les similarités possibles entre un document appartenant au premier cluster et un document appartenant au deuxième cluster. Ce critère ne souffre pas des problèmes qui se posent avec les critères du *lien simple* et *lien complet* [Zhao et Karypis, 2011].

### 3.4.2.2 Les *k*-moyennes

La méthode des *k*-moyennes appartient à la famille des algorithmes de partitionnement. Dans cette méthode, chaque cluster est représenté par une moyenne ou une moyenne pondérée appelée le "centroïde" qui est le plus proche de tous les autres éléments du cluster. Ce centroïde est calculé en utilisant l'équation 1.

$$C_j = \frac{1}{n_j} \sum_{X_i \in J} X_i \qquad (3.15)$$

où $C_j$ est le centroïde du cluster $j$, $X_i$ est un élément de ce cluster, et $n_j$ est le nombre de ces éléments.

La forme la plus simple de la méthode des *k*-moyennes est de sélectionner arbitrairement à partir du corpus original *k* documents représentatifs (centroïde de cluster) autour desquels les

clusters sont construits. Chaque document du corpus est attribués au centre représentatif le plus proche (au sens de la mesure de similarité utilisée). Dans l'itération suivante, les documents de chaque cluster vont définir le nouveau centroïde de ce cluster. En d'autres termes, le nouveau centroïde est défini, de sorte qu' il est un meilleur point central de ce cluster. Cette opération est répétée jusqu'à ce que la dispersion des membres de chaque cluster soit minimale. Les clusters deviennent en effet à chaque itération plus compactes, permettant une convergence de l'algorithme.

L'algorithme des *k*-moyennes est probablement l'un des algorithmes de segmentation les plus connus dans la communauté de classification des données. Il est relativement simple et permet d'obtenir de bonnes performances. L'inconvénient majeur de l'algorithme est sa sensibilité au choix des centres initiaux de clusters. Ce choix affecte la qualité de segmentation, en particulier dans le cas du segmentation de documents. Plusieurs méthodes sont utilisées pour améliorer la qualité des centres initiaux utilisés pour le processus de segmentation. Par exemple, l'utilisation d'une autre méthode de segmentation telles que la Classification Ascendante Hiérarchique (CAH) afin de déterminer l'ensemble initial des centres de clusters, c'est à la base de la méthode *Scatter-Gather* utilisée dans [Cutting et al., 1992]. Cette méthode est décrite ci-dessous. D'autre inconvénient, le centroïde d'un cluster donné de documents peut contenir un grand nombre de mots, qui va ralentir les calculs de similarité dans la prochaine itération.

### 3.4.2.3 La méthode Scatter-Gather

Bien que les méthodes de classification hiérarchiques ont tendance à être plus robuste en raison de la comparaison de tous les paires de documents, ils ne sont généralement pas très efficaces car elles sont de complexité $O(n^2)$. En revanche, les algorithmes de type *k*-moyennes sont plus efficaces que les algorithmes hiérarchiques, mais peuvent parfois ne pas être très efficaces en raison de leur tendance à s'appuyer sur un petit nombre de points centraux.

Pour surmonter ces limitations, la méthode décrite dans [Cutting et al., 1992] utilise à la fois un algorithme de segmentation hiérarchique et un algorithme partitif. Plus précisément, les auteurs utilisent l'algorithme de segmentation hiérarchique sur un échantillon du corpus, afin de trouver un ensemble initiale robuste de centres. Cet ensemble est utilisée en conjonction avec l'algorithme des *k*-moyennes afin de déterminer les bons clusters.

Il existe deux méthodes possibles pour la création de l'ensemble initial de centres, appelées respectivement *buckshot* et *fractionation*. Ce sont deux méthodes alternatives, et sont décrites comme suit:

***Buckshot***

Soit *k* le nombre de centres à trouver et *n* le nombre de documents dans le corpus. Au lieu de choisir les *k* centres aléatoirement de la collection, la méthode *Buckshot* prend une surestimation $\sqrt{k \times n}$ des centres, puis l'algorithme standard de classification ascendante hiérarchique est appliqué à cet échantillon pour obtenir un ensemble initiale de *k* centres. Nous notons que cet ensemble est robuste, en raison de la récapitulation d'un grand échantillon de documents dans un ensemble robuste de *k* documents centraux.

***Fractionation***

L'algorithme *Fractionation* commence par scinder le corpus en *n*/*m* groupes de même taille *m*>*k*. Sur chacun de ces groupes, un algorithme d'agglomération est appliquée pour les réduire par un facteur de *v*. Ainsi, à la fin de la phase, on a un total de $v \times m$ points agglomérés. Le processus est répété en traitant chacun de ces points agglomérés comme un enregistrement individuel. Ceci est obtenu par la fusion des différents documents d'un cluster aggloméré dans un seul document. L'approche se termine lorsque on obtient *k* documents centraux.

Une fois que les centres de clusters initiaux ont été déterminés en utilisant l'algorithmes *Buckshot* ou l'algorithmes *Fractionation*, nous pouvons appliquer l'algorithme partitif *k*-moyennes. Plus précisément, chaque document est affecté à la plus proche des *k* centres de clusters. Le centroïde de chaque cluster est déterminé comme étant la concaténation des différents documents dans ce cluster. Ces centroïdes remplacent les centres de la dernière itération. Ce processus est répété itérativement afin d'affiner successivement les centres de clusters.

Il est également possible d'utiliser un certain nombre de procédures pour améliorer encore la qualité des clusters. Ces procédures sont les suivantes:

***Opération de scindage***

Le processus de scindage peut être utilisé pour affiner les clusters en groupes de meilleure granularité. Ceci peut être obtenu en appliquant la méthode *buckshot* sur les documents individuels dans un cluster en utilisant $k = 2$, puis en regroupant ces documents autour de ces centres. Cependant, il n'est pas nécessaire de scinder tous les groupes. Seulement les groupes

qui ne sont pas très cohérents et contiennent des documents de nature disparate qui peuvent être scindés. Pour mesurer la cohérence d'un groupe, on calcule l'autosimilarité d'un cluster, qui est calculée en terme de la similarité des documents dans un cluster à son centroïde ou en terme de la similarité des documents du cluster les uns aux autres.

Le critère de scindage peut ensuite être appliqué de manière sélective uniquement aux groupes qui ont une faible autosimilarité. Cela aide à la création de clusters plus cohérentes

***Opération de jointure***

L'opération de jointure tente de fusionner les clusters similaires en un seul cluster. Pour effectuer la fusion, nous calculons les mots topiques de chaque cluster en examinant les mots les plus fréquents du centroïde. Deux clusters sont considérés comme similaires, si il y a un chevauchement important entre les mots topiques des deux clusters.

La méthode Scatter/Gather [Cutting et al., 1992] ne peut pas être définie uniquement comme une méthode de segmentation de documents. En effet, l'idée prédominante des auteurs de cette méthode est d'utiliser une méthode de segmentation en tant qu'outil de navigation dans le processus de recherche d'information.

# 3.5 Extraction d'information

## 3.5.1 Définition

L'extraction d'information à partir du texte est une tâche importante dans la fouille de textes. L'objectif général est d'identifier un ensemble de concepts prédéfinis dans un domaine spécifique, en ignorant toute autre information non pertinente, où le domaine est constitué d'un corpus de textes avec un besoin d'information clairement spécifié [Piskorski et Yangarber, 2013]. Les informations à extraire sont pré-spécifiées par l'utilisateur dans des structures définies appelées formulaire, chacun consistant en un nombre d'attributs. L'extraction d'information alors, consiste à remplir automatiquement un formulaire à partir d'un énoncé en langue naturelle.

En d'autres termes, l'extraction d'information est la tâche de découvrir des informations structurées à partir de texte non structuré ou semi-structuré, afin de construire une représentation plus significative de leur contenu sémantique. Cette représentation peut être directement présentée à un utilisateur final, et plus couramment, elle peut être utilisée par des

moteurs de recherche ou pour peupler des bases de données qui fournissent des entrées structurées pour l'extraction de motifs plus complexes (des tendances, des résumés, …) dans des collections de textes. On peut citer l'exemple suivant : compte tenu de la phrase arabe suivante :

في عام 1832، أسس الأمير عبد القادر الدولة الجزائرية الحديثة.

on peut extraire les informations suivantes :

مؤسس (الأمير عبد القادر , الدولة الجزائرية الحديثة ).

تأسست في (الدولة الجزائرية الحديثة , 1832 ).

Les grandes tâches classiquement identifiées en extraction d'information sont :

- reconnaissances d'entités nommées: les gens, les entreprises, les lieux, les gènes, les médicaments, etc.
- résolution des coréférences: nécessite l'identification des multiples mentions (coréférences) de la même entité dans le texte. Une mention d'entité peut être: pronominale, nominale (exemple : Premier Ministre), abréviations et variantes orthographiques (béta-lactamase, $\beta$-lactamase), etc.
- extraction de propriétés: des caractéristiques des entités extraites comme le titre d'une personne, l'âge d'une personne, le type d'une organisation, etc.
- identification de relations: des relations qui existent entre les entités comme : interaction entre des protéines, une relation d'emploi entre une personne et une société, etc.
- identification des événements: les événements sont des relations complexes telle que la description d'une attaque terroriste dans laquelle doivent être identifiés les terroristes, les victimes, les lieux, la date. . .

Les informations extraites fournissent des données plus concises et précises pour le processus de la fouille de textes que les plus naïves approches à base de mot, telles que celles utilisées pour la catégorisation de textes, et tendent à représenter des concepts et des relations qui sont plus significatifs et sont directement liés au domaine examiné du document [Ben-Dov et Feldman, 2010].

## 3.5.2 Stratégies d'extraction d'information

La conception des systèmes d'extraction d'information est basée sur deux approches fondamentales : l'ingénierie de connaissances et l'apprentissage automatique [Aggarwal et

Zhai, 2012c]. Les premiers systèmes d'extraction d'informations tels que ceux qui ont participé dans les MUCs (*Messages Understanding* ⟶ *Conferences*) ont été développés en utilisant l'approche d'ingénierie de connaissances, où la création de connaissances linguistiques sous forme de règles grammaticales est effectuée par des experts humains. Ces règles utilisent des patrons linguistiques et les mettent en correspondance avec le texte pour localiser des unités d'information. Une règle se compose d'un patron et d'une action. Le patron est une description d'enchaînement possible de syntagmes nominaux ou verbaux, attendu qu'ils expriment l'information à repérer. Lorsque ce patron correspond à une séquence de mots, l'action spécifiée est déclenchée. Par exemple, pour étiqueter n'importe quelle syntagme de la forme "Mr. *X*" où *X* est un mot capitalisé comme une entité personne, la règle suivante peut être définie :

(token = ''Mr.'' Orthography type = *FirstCap*) → person name.

Le côté gauche est un patron qui correspond à un syntagme de deux mots lorsque le premier mot est "Mr." et le deuxième mot a le type d'orthographe *FirstCap* (la première lettre du mot est en majuscule). Le côté droit indique que le syntagme correspondant doit être étiqueté comme un nom de personne.

Ces systèmes peuvent obtenir de bonnes performances sur le domaine spécifique cible, mais la conception de bonnes règles d'extraction s'agit d'une tâche longue et très difficile, ainsi que les règles développées sont très dépendantes du domaine [Jung, 2012]. Cela a motivé la recherche et le développement de systèmes d'extraction d'information entraînables en utilisant les méthodes d'apprentissage automatique statistique. Bien que le développement de l'approche d'apprentissage automatique reste plus rapide que celui de l'ingénierie de connaissances, l'apprentissage automatique exige néanmoins un volume de données assez conséquent (quantitativement mais aussi qualitativement) [Eikvil, 1999].

L'architecture générale d'un système d'extraction d'information à base d'apprentissage automatique est donnée dans la Figure 3.6 [Weiss et al., 2010]. Généralement, il y a deux modules principaux impliqués dans un tel système, à savoir la reconnaissance d'entités nommées et l'extraction de relations. Dans ce qui suit, nous nous intéressons plus particulièrement à ces deux tâches et les méthodes d'apprentissage automatique utilisées.

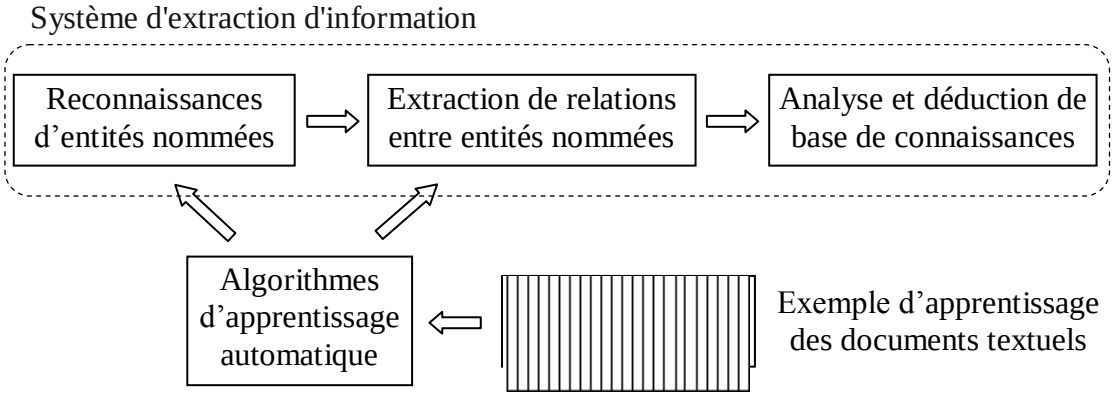



FIG. 3.6 – Système entraînable pour l'extraction d'information.

## 3.5.3 Reconnaissance d'entités nommées

Dans [Ehrmann, 2008], la définition proposée pour une entité nommée est toute expression linguistique qui réfère à une entité unique du modèle de manière autonome dans le corpus. Dans la définition, le *«modèle»* fait référence à une description préalable des informations pertinentes pour un contexte applicatif donné. Cela revient le plus souvent, en pratique, à déterminer la liste de types d'informations, en rapport avec un domaine particulier, que l'on souhaite extraire. Selon la compagne d'évaluation ACE (*Automatic Content Extraction*), une entité nommée dans le texte est une représentation pour nommer un objet réel. En 2007, l'ACE a étendue la définition traditionnelle des entités nommées (ACE 2007) en ajoutant aux types habituels (personne, organisme, lieu, expressions numériques) les types véhicule et arme.

La tâche de reconnaissance d'entités nommées dans un texte a pour objectif de déterminer les frontières d'une entité nommée, et de lui attribuer un type. Cette tâche est probablement la plus fondamentale dans l'extraction d'information. L'extraction des structures plus complexes telles que des relations et des événements dépend d'une reconnaissance précise d'entités nommées. Dans une étape de prétraitement, des taches connues dans le domaine du Traitement Automatique du Langage Naturel (TALN) sont appliquées d'abord sur le texte, telles que :

- Segmentation de texte en unités de base appelées *tokens* (mots, chiffres ou ponctuations). Cette tâche est appelée *Tokenization* en anglais.
- Etiquetage de séquence de *tokens* (*Tagging* en anglais). Les étiquettes les plus traditionnelles sont appelées "parties du discours" (*Part-of-speech*): elles caractérisent

la nature morpho-syntaxique de chaque *token* (exemple: DET désigne les déterminants, ADJ les adjectifs, NC les noms communs, V les verbes et PONCT les ponctuations).

- Analyse syntaxique superficielle (*Shallow Parsing* ou *Chunking*) : regrouper les mots du texte en consituants simples disjoints (*chunks*), de telle sorte que les mots de chaque groupe sont liés syntaxiquement (groupe nominal, verbal, etc.).
- Analyse syntaxique totale (*Full Parsing*) : déterminer les constituants syntaxiques et leurs fonctions syntaxiques dans la phrase. Un arbre syntaxique peut être construit à partir de cet analyse représentant la structure de la phrase complète. Des dépendances syntaxiques entre mots ou phrases (par exemple la dépendance sujet-verbe) peuvent également être calculées.

Une façon de voir l'extraction d'entités nommées dans un texte est d'annoter des *chunks* avec certains types prédéfinis, par exemple, nous voulons trouver les mentions de personnes, d'organisations et de lieux dans la phrase suivante :

عبد القادر نور أول من شغل منصب مدير عام المؤسسة الوطنية للإذاعة في الجزائر

Cette phrase peut être annotée comme suit:

[عبد القادر نور PER] أول من شغل منصب مدير عام [المؤسسة الوطنية للإذاعة ORG] في [الجزائر LOC]

Dans l'exemple ci-dessus, le chunk qui commence par PER indique personne, le chunk qui commence par POS désigne position, et le chunk qui commence par ORG dénote organisation.

Les travaux les plus récents sur la reconnaissance d'entités nommées est généralement basés sur l'apprentissage automatique statistique. De nombreux algorithmes de reconnaissance d'entités nommées à base d'apprentissage statistique traitent la tâche comme un problème d'étiquetage (classification) de séquences afin de prendre en compte l'aspect séquentiel des phrases et les dépendances éventuelles entre les types d'entités (par exemple il y a une probabilité non nulle que les chiffres suivant une entité de type organisation soient une date).

Pour faire correspondre la reconnaissance d'entités nommées au problème d'étiquetage de séquence, chaque mot dans une phrase est traité comme observation. Il s'agit alors de trouver, pour une phrase $x$ contenant $n$ mots notés $x = \{x_1, \ldots, x_n\}$, les étiquettes des types d'entités nommées $y = \{y_1, \ldots, y_n\}$ qui lui correspondent. Les étiquettes de classes doivent indiquer clairement à la fois les limites et les types d'entités nommées dans la séquence.

L'annotation en *Begin Inside Outside* (*BIO*) est habituellement utilisée pour ce problème. Avec cette notation, pour chaque type d'entité *T*, deux étiquettes sont créées, à savoir *B-T* et *I-T*.

Un mot marqué par *B-T* est le début d'une entité nommée de type *T* tandis qu'un mot marqué par *I-T* est à l'intérieur (mais pas au début) d'une entité nommée de type *T*. En outre, il y a une étiquette *O* pour les mots qui sont en dehors de toute entité nommée. La Figure 3.7 montre la phrase de l'exemple précédant où les mots sont étiquetés en utilisant l'annotation *BIO*.

| عام | مدير | منصب | شغل | من | أول | نور | القادر | عبد |
|---|---|---|---|---|---|---|---|---|
| *O* | *O* | *O* | *O* | *O* | *O* | *I-PER* | *I-PER* | *B-PER* |

| الجزائر | في | للإذاعة | الوطنية | المؤسسة |
|---|---|---|---|---|
| *B-LOC* | *O* | *I-ORG* | *I-ORG* | *B-ORG* |

FIG. 3.7 – Reconnaissance d'entités nommées par étiquetage de séquence.

L'un des premiers systèmes de reconnaissance d'entités nommées à base d'apprentissage statistique est Nymble [Bikel et al., 1997], qui utilise un Modèle de Markov Caché (HMM) composé de huit états internes (les noms de classes, y compris la classe not-a-NAME), avec deux états spéciaux, START-OF-SENTENCE et END-OF-SENTENCE. Le HMM est entraîné et appliqué pour reconnaitre les EN en deux langues, l'Anglais et l'Espagnol, en utilisant des algorithmes standards bien connus dans la littérature. Le système a obtenu des taux de reconnaissance entre 90% et 93%.

D'autres modèles d'apprentissage statistique ont également été appliqué à la reconnaissance d'entités nommées tels que les MEMM (Maximum Entropy Markov Models) et les CRF (Conditional Random Fields).

## 3.5.4 Extraction de relations entre entités nommées

L'extraction de relations entre entités nommées est la tâche de détecter et de caractériser les relations sémantiques entre les entités nommées de texte. L'exemple suivant est extrait de [Jean-Louis, 2011] après traduire le texte en arabe. Le tableau 3.1 présente quelques relations sémantiques et leurs types pour l'extrait ci-dessous, dont le but est d'extraction d'information dans le domaine sismique :

[زلزال] شدته [5] على مقياس ريختر حدث يوم [الأحد] في [شمال غرب إيران]، ذكره التلفزيون الحكومي دون إعطاء حصيلة. وحسب نفس المصدر فإن [الزلزال] ضرب مدينة [خلخال] في محافظة أردبيل، على الساعة [16:41] بالتوقيت المحلي (12:41 بتوقيت غرينتش).

Une grande partie des travaux sur l'extraction de relations sont basés sur la définition de la tâche par le programme ACE (Automatic Content Extraction). L'ACE se concentre sur les relations binaires, c'est à dire les relations entre deux entités (qu'on peut appeler aussi des

arguments de la relation), comme les exemples de relations du tableau 3.1. Un ensemble de types principaux de relation et leurs sous-types est défini par l'ACE. Les types principaux de relations ACE comprennent : physique (par exemple une entité est physiquement proche d'une autre entité), personnelle/sociale (par exemple, une personne est un membre de la famille d'une autre personne), et emploi/affiliation (par exemple, une personne est employée par une organisation). Néanmoins, les relations entre les entités ne sont pas toujours binaires et peuvent impliquer plus de deux entités.

| Relation (R) | Entité (e1) | Entité (e2) |
|---|---|---|
| lieu d'un événement | زلزال (Evénement) | شمال غرب إيران (Lieu) |
| date d'un événement | زلزال (Evénement) | الأحد (Date) |
| lieu d'un événement | زلزال (Evénement) | خلخال (Lieu) |
| date d'un événement | زلزال (Evénement) | 16:41 (Heure) |

TAB. 3.1. Exemple de relations entre entités nommées.

L'approche la plus répandue et la plus simple pour l'extraction de relations est basée sur l'apprentissage automatique, où la tâche est traitée comme un problème de classification supervisée. Plus précisément, toute paire d'entités co-occurrentes dans la même phrase est considérée comme instance de relation candidate. L'objectif est d'attribuer une étiquette de classe à cette instance où l'étiquette est l'un des types de relations prédéfinis, ou *nil* pour les paires d'entités indépendantes [Jung, 2012]. L'approche de classification suppose l'existence d'un corpus d'apprentissage, dans lequel les mentions des types de relation prédéfinis ont été annotées manuellement. Ces mentions sont utilisées comme exemples positifs de l'ensemble d'apprentissage. Les paires d'entité co-occurrentes dans la même phrase, mais qui ne sont pas étiquetées sont utilisées comme des exemples négatifs de l'ensemble d'apprentissage [Jung, 2012].

Deux types de méthodes de classification ont été proposées pour l'extraction de relations entre les entités: les méthodes à base de caractéristiques et les méthodes à noyau.

## 3.5.4.1 Méthodes à base de caractéristiques

En général, les méthodes d'extraction de relations à base de caractéristiques suivent le processus indiqués dans la Figure 3.8, où chaque exemple d'apprentissage (instance de

relation candidate) est représenté par un vecteur de caractéristiques. Ces vecteurs *peuvent mieux exprimer les exemples d'apprentissage*. Des algorithmes standards d'apprentissage tels que les SVM peuvent ensuite être utilisés pour entraîner des classifieurs de relation. L'extraction de caractéristiques est une étape critique pour cette approche de classification de relations. Ces informations représentent principalement de l'information lexicale, sémantique ou syntaxique. Elles ne sont pas toujours extraites dans le même contexte. Il est en effet possible de prendre en compte tous les mots de la phrase, ou uniquement les mots entre les deux entités, ou les trois mots avant ou après chaque entité, etc. Les caractéristiques les plus couramment utilisées pour la classification de relations sont :

***Caractéristiques d'entité* :** souvent les deux entités d'arguments, y compris les mots d'entités eux-mêmes et les types d'entités, sont corrélées avec certain type de relation. Par exemple, les mots d'entités telles que أب , أم , أخ et أخت et le type d'entité de شخص sont tous des indicateurs forts de sous-type de relation familiale.

***Caractéristiques contextuelles lexicales*:** intuitivement les contextes entourant les deux entités d'arguments sont importants. La façon la plus simple d'intégrer l'évidence de contextes est d'utiliser les caractéristiques lexicales. Ces caractéristiques peuvent être les mots de la phrase. Par exemple, si le mot أسس survient entre les deux arguments, ils sont plus susceptibles d'avoir la relation مؤسس. Elles peuvent être aussi la distance (en nombre de mots) entre les deux entités, les mots qui forment les deux entités, le fait qu'il n'y a pas de mots entre les deux entités, etc.

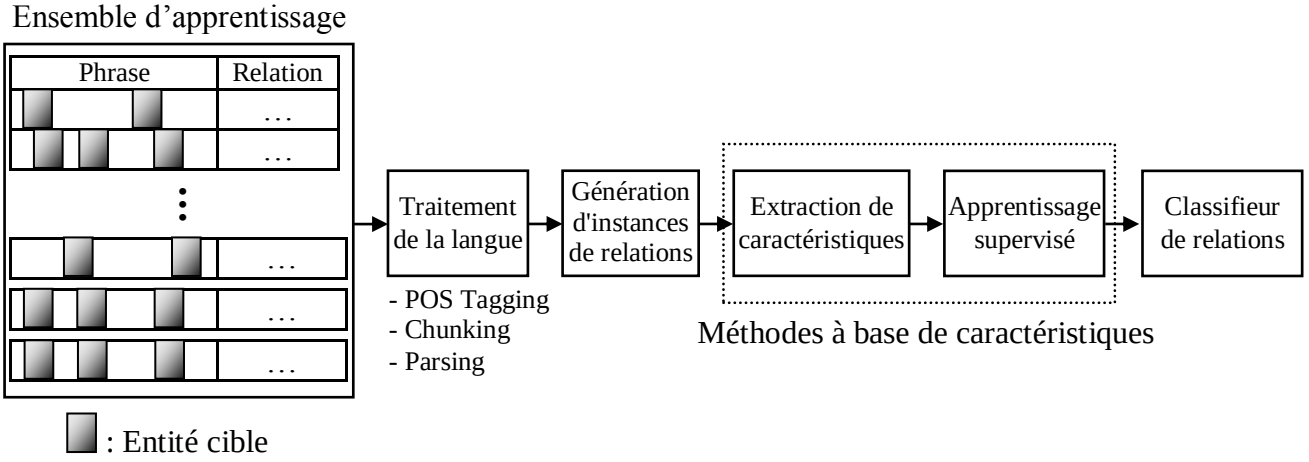



FIG. 3.8 - Processus d'apprentissage pour les méthodes d'extraction de relations à base de caractéristiques.

***Caractéristiques contextuelles syntaxiques* :** l'information syntaxique semble indispensable pour décrire et identifier une relation. Elle permet par exemple d'identifier le ou les verbe(s) de la phrase (souvent déclencheur de la relation), les groupes prépositionnels, etc. Les caractéristiques syntaxiques les plus couramment utilisées pour l'extraction de relations, sont les catégories morphosyntaxiques des mots du contexte des entités, elles peuvent être obtenues à partir de l'arbre syntaxique de la phrase contenant l'instance de relation. Par exemple, si le premier argument est le sujet du verbe أسس et le deuxième argument est l'objet de ce verbe, alors on peut presque dire immédiatement que la relation مؤسس existe entre les deux arguments.

Il est également possible d'utiliser des informations provenant d'arbres de dépendances syntaxiques reliant les entités candidates. Van Landeghem et al. [Landeghem et al., 2008] définissent six classes de caractéristiques provenant du chemin le plus court reliant deux protéines dans l'arbre de dépendances pour extraire des interactions entre ces protéines.

***Caractéristiques sémantiques (connaissances extérieures)* :** les informations sémantiques transposées sous forme de caractéristiques peuvent provenir de ressources pour la langue générale ou de ressources d'un domaine de spécialité. Chan et Roth [Chan et Roth, 2010] ont étudié l'utilisation des connaissances extérieures venant d'une source encyclopédique (Wikipedia) pour l'extraction de la relation. Si deux arguments surviennent dans le même article de Wikipedia, le contenu de l'article peut être utilisé pour vérifier si les deux entités sont liées.

## 3.5.4.2 Méthodes à noyau

En générale, dans la classification de relations à base de caractéristiques, les exemples d'apprentissage sont représentés par des vecteurs de caractéristiques, et l'apprentissage est effectuée en utilisant des algorithmes d'apprentissage automatique. Cependant, dans de nombreux cas, il n'est pas possible d'exprimer le vecteur de caractéristiques raisonnablement. Par exemple, un espace de représentation est requis pour exprimer des informations syntaxiques (graphe de dépendances, arbre de constituants, etc.) d'une phrase spécifique en tant que vecteur de caractéristique, et dans certains cas, il est presque impossible d'exprimer ces informations sous forme d'un vecteur de caractéristiques dans un espace limité [Cristianini et Shawe-Taylor, 2000].

Comme alternative aux méthodes à base de caractéristiques, les méthodes à noyau peuvent explorer implicitement (sans la nécessité de représenter les caractéristiques explicitement) un espace de caractéristiques beaucoup plus grand ; ces méthodes ont été proposées pour l'apprentissage en calculant des fonctions noyau entre deux exemples tout en gardant l'exemple original d'apprentissage sans expression de caractéristiques supplémentaires [Cristianini et Shawe-Taylor, 2000]. La fonction noyau est définie comme le *mapping* $K : X \times X \to [0,\infty)$ de l'espace d'entrée *X* au score de similarité $\phi(x) \cdot \phi(y) = \sum_i \phi_i(x) \cdot \phi_i(y)$.

Ici, $\phi(x)$ est la fonction du *mapping* à partir d'exemples d'apprentissage dans un espace d'entrée *υ* à un espace de caractéristiques multidimensionnel. Avec la fonction noyau, il n'est pas nécessaire de calculer toutes les caractéristiques un par un, et l'apprentissage automatique peut donc être effectué en se basant uniquement sur la similarité entre les deux exemples d'apprentissage [Jung et al., 2012].

Donc, on peut dire que la fonction noyau remplace le processus d'extraction de caractéristiques illustré dans la Figure 3.8 (méthodes à base de caractéristiques) et que cette fonction *peut calculer la plus efficacement (directement) la similarité de deux exemples d'apprentissage* représentés dans un espace vectoriel sous-jacent. Le calcule de la similarité est défini comme le produit scalaire des deux exemples d'apprentissage. Nous allons illustrer cette idée dans le paragraphe 3.5.4.2.2 en utilisant le noyau de convolution d'arbre.

Ici, la mesure de similarité entre les exemples d'apprentissage n'est pas dans un sens général. Autrement dit, la fonction de similarité entre les deux phrases ou les deux instances qui expriment la même relation est la fonction noyau la plus efficace du point de vue de l'extraction de relation.

Par exemple, les deux phrases "القدس عاصمة فلسطين" et " تقع الأغواط في الجزائر " utilisent des entités nommées différentes, mais disposent de la même relation ("تقع"). Par conséquent, une fonction noyau efficace permettrait de détecter une grande similarité entre ces deux phrases. D'autre part, puisque les phrases "القدس عاصمة فلسطين" et "القدس تقع في فلسطين" expriment les mêmes entités nommées mais disposent de relations différentes, la similarité entre ces deux phrases doit être déterminée comme très faible. A ce titre, dans les méthodes d'extraction de relations à noyau, la sélection et la création de fonctions noyau représentent la partie la plus fondamentale qui affecte la performance globale.

Il y a généralement trois types de noyaux pour l'extraction de relations entre les entités nommées: noyaux de séquences, noyaux d'arbres et noyaux composés. Dans ce qui suit nous allons présenter brièvement ces trois types de noyaux.

## 3.5.4.2.1 Noyaux de séquence

Les noyaux de séquences ont tous pour but de compter le nombre de sous-séquences communes à deux objets textuels. Dans [Bunescu et Mooney, 2005], Bunescu et Mooney ont travaillé sur le corpus ACE et ont observé que le plus court chemin entre deux entités dans l'arbre d'analyse de dépendance syntaxique contribue le plus à établir la relation entre ces entités. Sur cette base, ils ont défini un noyau simple qui représente chaque nœud dans une séquence de mots par un vecteur de caractéristiques qui peuvent être le mot lui-même, sa catégorie morphosyntaxique, ou son type d'entité. Ce noyau considère que deux chemins de dépendance sont similaires s'ils ont la même longueur et ils partagent de nombreux nœuds communs (il compare chaque paire de nœud dans la même position dans les deux chemins pour calculer les valeurs communes de caractéristiques). L'exemple dans la Figure 3.9 montre comment calculer le noyau de chemin de dépendance entre les deux entités الجيش et العاصمة où trois types de caractéristiques sont utilisées : le mot lui-même, sa catégorie morphosyntaxique détaillée (NNS, VBD, IN, NNP) et générale (Noun, Verb), son type d'entité (Person, Location, ..).

**بغداد** ⟵ في ⟵ ينتشر ⟶ **الجيش** ⟶ '**الجيش** ينتشر في **بغداد**'

**تكريت** ⟵ نحو ⟵ تتقدم ⟶ **القوات** ⟶ '**القوات** تتقدم نحو **تكريت**'

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| الجيش<br>NNS<br>Person | ⟶ | ينتشر<br>VBD<br>Verb | ⟵ | في<br>In | ⟵ | بغداد<br>NNP<br>Noun<br>Location | |
| القوات<br>NNS<br>Person | ⟶ | تتقدم<br>VBD<br>Noun | ⟵ | نحو<br>In | ⟵ | تكريت<br>NNP<br>Noun<br>Location | |
| 2 | 1 | 2 | 1 | 1 | 1 | 3 | **12** |

FIG. 3.9 - Calcul du noyau de chemin de dépendance.

Bunescu & Mooney [Bunescu et Mooney, 2006] ont introduit un noyau de sous-séquences où la similarité entre deux séquences de mots est définie sur leurs sous-séquences similaires (considérant que les sous-séquences où les entités candidates existent, et les mots appartiennent à trois patrons prédéfinis : les mots avant et entre les deux entités, seuls les mots entre les deux entités, entre et après les deux entités). Chaque nœud dans une séquence est représenté par un vecteur de caractéristique et la similarité entre deux nœuds est le produit scalaire de leurs vecteurs de caractéristiques. La similarité entre deux sous-séquences de la même longueur est calculée à partir de la valeur de similarité de chaque paire de leurs nœuds de la même position. Bunescu & Mooney ont testé leur noyau de sous-séquences pour la détection des interactions protéine-protéine sur le corpus ACE.

## 3.5.4.2.2 Noyaux d'arbres

Les noyaux d'arbres utilisent la même idée d'utiliser des sous-structures communes pour mesurer la similarité. Zelenko et al. [Zelenko et al., 2003] définissent un noyau d'arbres qui utilise les arbres d'analyse dérivés de l'analyse syntaxique superficielle (*chunking*) pour mesurer la similarité entre des phrases contenant des entités nommées. Cette étude est connue comme la première application d'une méthode à noyau pour l'extraction de relation. La motivation principale est que si deux arbres partagent de nombreux sous-arbres, alors les deux instances de relation sont similaires entre eux.

Culotta et Sorensen [Culotta et Sorensen, 2004] étendent l'idée aux arbres d'analyse de dépendance. Zhang et al. [Zhang et al., 2006] en outre appliquent le noyau de convolution d'arbre initialement proposé par Collins et Duffy [Collins et Duffy, 2001] à l'extraction de relation.

Dans ce que suit nous allons expliquer brièvement les noyaux de convolution d'arbre. Comme expliquée précédemment, une fonction noyau correspond à un espace vectoriel sous-jacent dans lequel les instances observées peuvent être représentées. Pour les noyaux de convolution d'arbre, chaque dimension de l'espace vectoriel sous-jacent correspond à un sous-arbre. Un arbre d'analyse $T$ est exprimé en tant que vecteur de fréquence d'occurrence des sous-arbres comme suit :

$$\phi(T) = (\#\ subtree_1(T), ..., \#\ subtree_i(T), ..., \#\ subtree_n(T)) \qquad (3.1)$$

Dans l'équation 3.1, $\#\ subtree_i(T)$ représente la fréquence d'occurrence de la *i-ième* sous-arbre de *T*. Seulement les sous-arbres contenant des règles complètes de

production de grammaire sont pris en compte. La fonction noyau de deux arbres $T_1$ et $T_2$ est calculée comme le produit scalaire de ses vecteurs comme suit :

$$K(T_1, T_2) = <\phi(T_1), \phi(T_2)> \quad (3.2)$$

La Figure 3.10 montre un exemple d'arbre d'analyse syntaxique et tous les sous-arbres sous le *NP* "the company". Où l'abréviation *S* est mise pour *Sentence* (phrase), *NP* pour *Noun Phrase* (syntagme nominal), *VP* pour *Verbal Phrase* (syntagme verbal), *D* pour *Determiner* (déterminant), *N* pour *Noun* (nom) et *V* pour *Verb* (verbe).

$$\begin{aligned} K(T_1, T_2) &= \langle \phi(T_1), \phi(T_2) \rangle \\ &= \sum_i \#subtree_i(T_1) \cdot \#subtree(T_2) \\ &= \sum_i \Big( \sum_{n_1 \in N_1} I_{subtree_i}(n_1) \Big) \cdot \Big( \sum_{n_2 \in N_2} I_{subtree_i}(n_2) \Big) \\ &= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_{subtree_i}(n_1) \cdot I_{subtree_i}(n_2) \end{aligned} \quad (3.3)$$

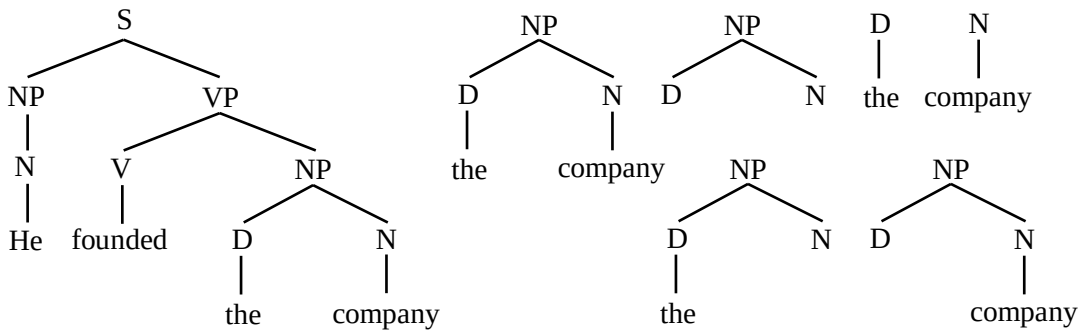



FIG. 3.10 - A gauche: l'arbre de constituants d'une phrase simple. A droite: tous les sous-arbres du NP "*the company*" considérés dans les noyaux de convolution d'arbres (figure extraite de [Jung, 2012]).

*N1* et *N2* sont les ensembles de tous les nœuds respectivement dans *T1* et *T2*. $subtree_i$ désigne un sous-arbre dans l'espace des caractéristiques. $I_{subtree_i}(n)$ vaut 1 si le nœud *n* est le nœud racine de l'arbre $subtree_i$ et 0 sinon. Le calcule directe de *K* tel que défini dans l'équation 3.3 n'est pas efficace, on peut définir $C(n_1, n_2) = \sum_i I_{subtree_i}(n_1) \cdot I_{subtree_i}(n_2)$.

$C(n_1,n_2)$ peut alors être calculée en temps polynomial basé sur la propriété récursive suivante:

- Si les productions de grammaire à *n1* et *n2* sont différentes, alors la valeur de *C*(*n1*, *n2*) est égal à 0.
- Si les productions de grammaire à *n1* et *n2* sont les mêmes, et *n1* et *n2* sont des préterminaux (des nœuds directement au-dessus des mots dans un arbre d'analyse), alors *C*(*n1*, *n2*) est égale à 1.
- Si les productions de grammaire à *n1* et *n2* sont les mêmes et *n1* et *n2* ne sont pas des préterminaux,

$$C(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + C(ch(n_1, j), ch(n_2, j))), \qquad (3.4)$$

où *nc*(*n*) est le nombre de nœuds enfants de *n*, et *ch*(*n*, *j*) est la *j*-ième nœud enfant de *n*. Notez qu'ici $nc(n_1) = nc(n_2)$.

### 3.5.4.2.3 Noyaux composites

Il est possible de combiner différents noyaux dans un noyau composite. C'est quand il nous est difficile d'inclure toutes les caractéristiques utiles dans un seul noyau. Zhang et al. [Zhang et al., 2006] ont combiné un noyau d'entité à base de caractéristiques (qui mesure le nombre de caractéristiques communes entre les deux entités), et un noyau de convolution d'arbre (qui compte le nombre de sous-arbres communs entre deux relations), de deux manières de combinaison différentes : linéaire et polynomiale qui vise à explorer les caractéristiques combinées à partir des deux entités de la relation.

## Conclusion

Dans ce chapitre, nous venons de passer en revue des techniques et algorithmes de la fouille de textes. Nous avons abordé en premier lieu une partie qui vise à se positionner au domaine de la fouille de texte, comprenant une définition de la fouille de texte, une confrontation de celui-ci avec la recherche d'information et les spécificités des données textuelles. Dans la deuxième partie, nous avons vu que la représentation d'une collection de documents textuels se fait en plusieurs étapes : d'abord un mode de représentation et choisi et ensuite tous les termes (mots) des documents sont extraits. Avant de coder ces termes, nous avons vu qu'il est nécessaire de les prétraiter afin de réduire le lexique à prendre en compte. Après l'extraction des vecteurs de caractéristiques en utilisant une méthode de pondération de

termes, nous avons présenté les méthodes de sélection de caractéristiques pertinentes pour la tâche de classification, la transformation des vecteurs extraites et la mesure de similarité entre deux documents.

Nous avons consacré la grande partie de ce chapitre pour présenter les tâches principales de la fouille de texte. Nous nous avons limité surtout à la classification de documents et l'extraction d'information comme des problèmes d'apprentissage automatique. Dans la troisième et quatrième partie, nous avons étudié un nombre non exhaustif d'algorithmes d'apprentissage automatique pour la classification supervisé (catégorisation) et non supervisé (segmentation) de documents textuels. Dans la dernière partie, nous avons défini l'extraction d'information puis présenté ses tâches principales (la reconnaissance d'entités nommées et l'extraction des relations) ainsi que les approches d'apprentissage utilisées.

Enfin, ce chapitre a pour but de nous nous familiariser avec les techniques et algorithmes de la fouille de texte, et surtout ceux qui nous permettent d'élaborer notre système de catégorisation dans le prochain chapitre.

# Chapitre 4

# Système de catégorisation automatique de documents manuscrits arabes

## Introduction

Nous présentons dans ce chapitre notre système complet pour la catégorisation automatique de documents manuscrits arabes. Ce système est composé de deux tâches principaux : la reconnaissance de mots manuscrits et la catégorisation de textes.

Le chapitre est organisé ainsi : la Section 4.1 est dédiée à la description générale de la structure et le fonctionnement du système. La Section 4.2 présente l'analyse et la segmentation des images de documents. Dans la Section 4.3, nous présentons le système de reconnaissance de mots manuscrits, en utilisant une approche analytique à base des modèles de Markov cachés HMMs avec une segmentation implicite. La tâche de catégorisation de documents est décrite en Section 4.4, où un classifieur *k*-ppv est utilisé. Enfin, la Section 4.5 est consacrée à la description des bases de données établies spécifiquement pour cette étude et les résultats des expériences réalisées.

## 4.1 Présentation générale du système

L'approche adoptée est composée naturellement de trois tâches principales : la segmentation de document, la reconnaissance de mots manuscrits et la catégorisation de textes (voir Figure 4.1). En premier temps, l'image de chaque document en entrée du système est analysée afin de la segmenter en lignes, puis les lignes sont segmentées en mots. Ceux-ci vont être ensuite reconnus par le système de reconnaissance.

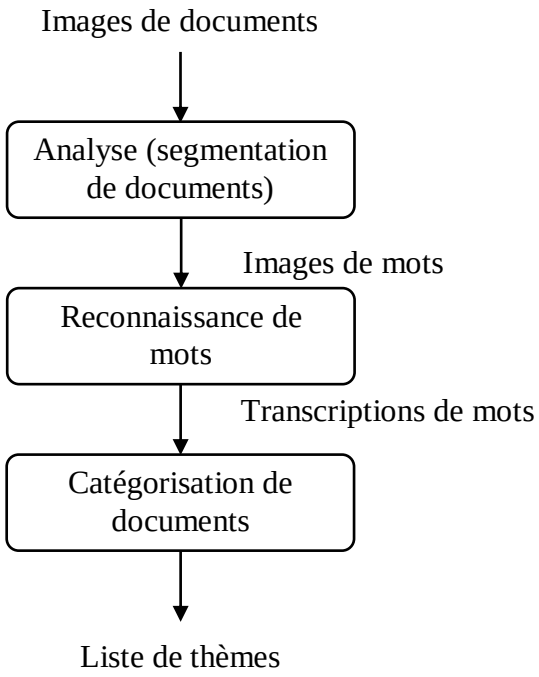



FIG. 4.1 – Schéma générale du système de catégorisation de documents manuscrits arabes.

Dans la deuxième étape le système de catégorisation utilise les transcriptions des mots issues de la reconnaissance pour transformer le texte de sa forme originale à une représentation plus gérable, afin de traiter celle-ci et l'utiliser ensuite pour la catégorisation en affectant chaque document à une catégorie spécifique prédéfinie.

## 4.2 Segmentation de documents

Le but de la segmentation est de diviser l'image de texte en unités simples : texte en lignes, les lignes en mots. L'approche commune pour la segmentation de texte est d'utiliser les profils de projection. Pour obtenir les lignes de texte nous avons utilisé le profil de projection horizontale. Cette technique est en général suffisante dans notre cas où les documents contiennent des lignes de texte relativement droites. Nous avons utilisé le profil de projection verticale pour la segmentation des lignes en mots. Cette méthode s'appuie sur le fait que l'espace séparant les mots est plus important que celui qui séparant les pseudo-mots qui appartiennent au même mot. Un seuil, calculé empiriquement à partir les données d'apprentissage, est utilisé pour détecter les espaces qui séparent les mots.

## 4.3 Système de reconnaissance de mots manuscrits arabes

L'objectif principal de la tâche de reconnaissance est de détecter et reconnaître les mots d'une image de document manuscrit afin de pouvoir les traiter par le système de catégorisation. D'abord, le prétraitement des images de mots suivi d'une détection des lignes de base sont effectués pour préparer les images à l'étape d'extraction des caractéristiques en utilisant les fenêtres glissantes.

L'approche utilisée est une approche analytique dirigée par le lexique. La plupart des méthodes de reconnaissance d'écriture manuscrite présentées dans le Chapitre 2 utilisent les modèles de Markov cachés HMMs avec une segmentation implicite. Dans notre système nous avons choisi les HMMs pour apprendre nos modèles de caractères. Leur utilisation se justifie par le fait qu'ils sont les méthodes les plus utilisées et qu'ils atteignent les meilleures performances. Dans la suite de cette section nous allons présenter les étapes de notre système de reconnaissance : les prétraitements, l'estimation des lignes de base, l'extraction des caractéristiques, l'apprentissage et le décodage de nos modèles HMMs.

### 4.3.1 Prétraitement

La première étape d'un système de reconnaissance d'écriture est le prétraitement. Une normalisation des images en entrée est effectuée afin de réduire la variabilité de l'écriture. Nous avons présenté dans le Chapitre 2 différents prétraitements qui peuvent être appliqués sur les images. Dans la base de données que nous avons utilisée les documents sont en général de bonne qualité, elles ne nécessitent pas de correction d'inclinaison de lignes et de mots.

Dans notre système, nous avons choisi de ne pas normaliser la taille des images de mots et donc d'extraire des caractéristiques indépendantes de la taille des images. Le premier prétraitement appliqué sur l'image est le lissage des contours du mot pour éliminer de petites taches. Par la suite, nous appliquons une normalisation d'épaisseur de traits de mots à un nombre prédéterminé de pixels afin de rendre notre système robuste au variation d'épaisseurs de traits d'écriture. Cette normalisation est effectuée en appliquant l'opération de dilatation sur le squelette du mot. L'algorithme de Zhang-Suen [Zhang et Suen, 1984] est utilisé pour l'extraction de squelette. Enfin, l'image résultat est transformée pour obtenir l'image miroir horizontale.

## 4.3.2 Estimation des lignes de base

L'écriture arabe utilise deux lignes de base: ligne haute et ligne basse, définissent trois zones dans un mot : la zone centrale qui contient le corps central d'un mot, la zone supérieure où les ascendants peuvent être trouvés et la zone inférieure pour les descendants. De nombreux systèmes commencent donc par calculer les lignes de base afin de les utiliser ensuite pour l'extraction des caractéristiques dépendantes de ces lignes. Ces caractéristiques indiquent la présence des ascendants et des descendants.

Il existe plusieurs méthodes pour extraire ces lignes souvent basées sur l'analyse de l'histogramme de projection horizontale des pixels de l'image sur un axe vertical. Nous avons adopté cette méthode dans notre travail. Nous avons utilisé l'algorithme décrit dans [Likforman-Sulem et al., 2012] qui est une version modifiée de l'algorithme proposé par Blumenstein et *al.* [Blumenstein et al., 2002]. La Figure 4.2 présente le résultat de l'estimation des lignes de base pour le mot "المعهد" (institut).

FIG. 4.2 – Estimation des lignes de base du mot "المعهد".

L'algorithme utilisé est basé sur le calcul du profil de la projection verticale des densités de pixels le long de l'axe horizontal dans l'image en entrée. En premier, la ligne ayant le profil maximal est détectée. La position de cette ligne définit la ligne de base basse. Ceci est justifié dans le cas de l'écriture arabe où la plupart des lettres ont beaucoup de pixels sur la ligne de base basse. Par la suite, l'algorithme parcourt l'image du mot de haut au bas pour trouver la position de la ligne de base haute correspondante à la position de la première ligne avec une valeur de projection supérieure ou égale à la densité moyenne des lignes.

## 4.3.3 Extraction de caractéristiques

L'objectif de cette étape est de transformer les images de mots en une séquence de vecteurs de caractéristiques. Nous avons présenté au Chapitre 2 les caractéristiques les plus

utilisées pour la reconnaissance d'écriture manuscrite. Nous présentons ici les caractéristiques que nous utilisons, présentées dans [Al-Hajj et al., 2005] et [Likforman-Sulem et al., 2012] (voir Chapitre 2, Section 2.4.3). L'avantage de ces caractéristiques est qu'elles dépendent des lignes de base, ainsi une normalisation de la taille des images n'est pas nécessaire. D'autre part, la qualité de l'étape d'extraction de caractéristiques dépend de celle de l'étape d'extraction des lignes de base. Nous adoptons, ici, cette approche car elle montre son efficacité pour la reconnaissance de l'écriture manuscrite arabe.

L'approche que nous utilisons, donc, est basée sur l'extraction d'une séquence de vecteurs de caractéristiques observées par le décalage (glissement) d'une fenêtre de droite à gauche, conformément à la progression de l'écriture arabe, sur une image binaire de mot (réellement de gauche à droite sur l'image miroir). La fenêtre glissante est de largeur *w* pixels, elle s'adapte en hauteur à celle du mot. Cette fenêtre commence à l'extrême gauche du mot (dans la position $t = 1$), et est successivement décalée jusqu'à $t = T$, telle que deux fenêtres consécutives se chevauchent, comme le montre la Figure 4.3. Dans chaque position de la fenêtre sont extraites un ensemble de caractéristiques, alliant données statistiques et données géométriques, dont certaines dépendent de la position des lignes de base pour prendre en compte la présence des ascendants (hampes) et des descendants (jambages). Les caractéristiques extraites sont de deux types : caractéristiques de distribution de densité et caractéristiques de concavités des pixels d'écritures. Elles sont présentées ci-dessous.

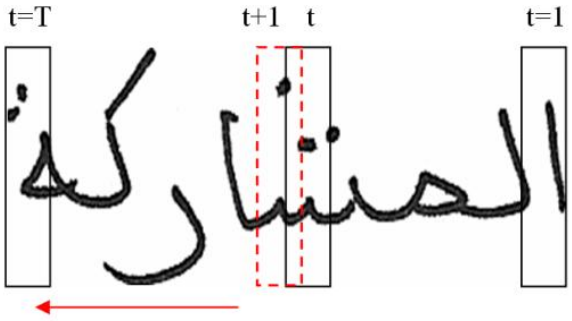



FIG. 4.3 – Fenêtres glissantes pour l'extraction des caractéristiques.

### 4.3.3.1 Caractéristiques de distribution

L'ensemble des caractéristiques de distribution se compose de 16 caractéristiques qui caractérisent la densité des pixels d'écriture dans la fenêtre. Ainsi, pour une fenêtre de largeur

$w$ pixels, de hauteur $H$ pixels (la hauteur de l'image), de chevauchement avec la fenêtre suivante $\delta$ pixels, les caractéristiques de distribution se calculent comme suit :

Soit $r(i)$ la somme des pixels d'écriture (pixels noirs) dans la ligne $i$ de la fenêtre courante:

$$r(i) = \sum_{j=1}^{w} I(i, j)$$

$f_1$ : densité des pixels d'écriture dans la fenêtre.

$$f_1 = \frac{1}{H \times w} \sum_{i=1}^{H} r(i)$$

$f_2$ : est une caractéristique dérivative définie comme la différence de position verticale entre les centres de gravité $g$ des pixels d'écriture dans deux fenêtres successives $t$ et $t$-1. Pour la première fenêtre la différence se calcule avec la position de la ligne de base basse.
Le centre de gravité, $g$, d'une fenêtre est donnée par :

$$g = \frac{\sum_{i=1}^{H} i \times r(i)}{\sum_{i=1}^{H} r(i)}$$

on a donc :

$$f_2 = g(t) - g(t-1)$$

$f_3$ : position verticale (ordonnée $y$) normalisée du centre de gravité des pixels d'écriture, par rapport à la ligne de base basse. Soit $i_{lb}$ la position verticale de la ligne de base basse, on a donc :

$$f_3 = \frac{g - i_{lb}}{H}$$

$f_4$ et $f_5$ : densités des pixels d'écriture au-dessus et au-dessous de la ligne de base basse.

$$f_4 = \frac{1}{H \times w} \sum_{i=1}^{i_{lb}} r(i)$$

$$f_5 = \frac{1}{H \times w} \sum_{i=i_{lb}+1}^{H} r(i)$$

## 4.3.3.2 Caractéristiques de concavité

Les caractéristiques de concavité $f_6$ à $f_{11}$ fournissent des informations de concavité et des directions de traits dans chaque fenêtre. Ces caractéristiques sont calculées en comptant, dans la zone concernée, le nombre de pixels de fond (pixels blancs) correspondant à l'une des six

configurations locales de la Figure 4.4. Le nombre de pixels dans chaque configuration est ensuite normalisé par la hauteur de la zone. Les caractéristiques $f_6$ à $f_{11}$ décrivent la concavité dans l'ensemble de la fenêtre, elles sont calculées comme suit :

On compte $C_{gh,fen}$ , $C_{hd,fen}$ , $C_{db,fen}$ , $C_{bg,fen}$ , $C_{v,fen}$ , et $C_{h,fen}$ les nombres de pixels de fond dans la fenêtre qui ont voisins pixels d'écriture, respectivement dans les directions suivantes : gauche-haut, haut-droit, droit-bas, bas-gauche, verticale et horizontale (les pixels frontières d'image sont exclus)

Puis :

$$f_6 = \frac{C_{gh,\,fen}}{H}, \qquad f_7 = \frac{C_{hd,\,fen}}{H}, \qquad f_8 = \frac{C_{db,\,fen}}{H}$$

$$f_9 = \frac{C_{bg,\,fen}}{H}, \qquad f_{10} = \frac{C_{v,\,fen}}{H}, \qquad f_{11} = \frac{C_{h,\,fen}}{H}$$

Les caractéristiques de concavité $f_{12}$ à $f_{17}$ sont relatives aux pixels dans la zone médiane d'une image de mot, entre les deux lignes de base. On compte $C_{gh,med}$ , $C_{hd,med}$ , $C_{db,med}$ , $C_{bg,med}$ , $C_{v,med}$ , et $C_{h,med}$ les nombres de pixels de fond dans la zone médiane qui ont voisins pixels d'écriture, respectivement dans les directions suivantes : gauche-haut, haut-droit, droit-bas, bas-gauche, verticale et horizontale (les pixels frontières d'image sont exclus).

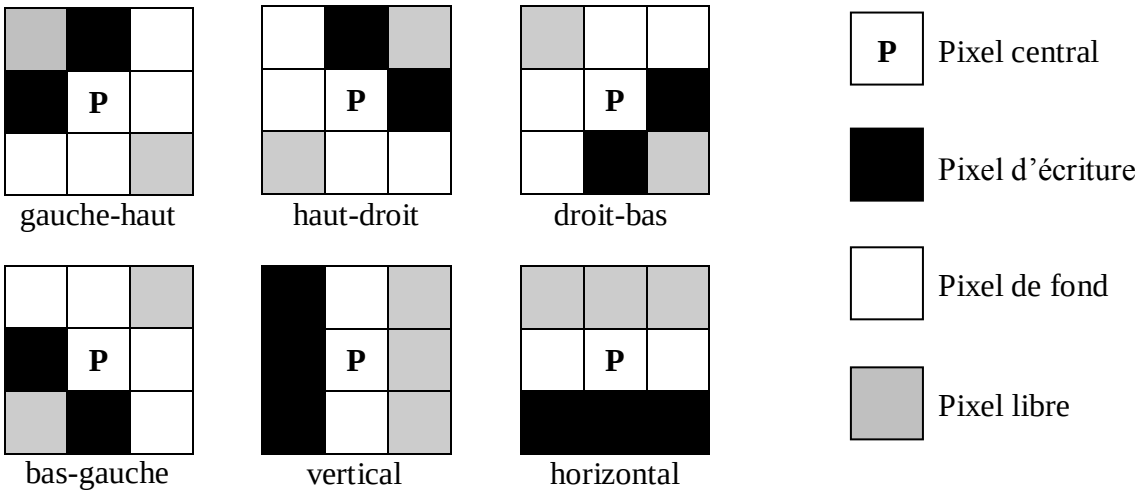



FIG. 4.4 – Les 6 types de configurations locales pour le calcul de caractéristiques de concavités.

Puis :

$$f_{12} = \frac{C_{gh,med}}{i_{ub} - i_{lb}}, \qquad f_{13} = \frac{C_{hd,med}}{i_{ub} - i_{lb}}, \qquad f_{14} = \frac{C_{db,med}}{i_{ub} - i_{lb}}$$

$$f_{15} = \frac{C_{bg,med}}{i_{ub} - i_{lb}}, \qquad f_{16} = \frac{C_{v,med}}{i_{ub} - i_{lb}}, \qquad f_{17} = \frac{C_{h,med}}{i_{ub} - i_{lb}}$$

où $i_{ub}$ est la position verticale de la ligne de base haute.

$f_{18}$ à $f_{18+w-1}$ : densité des pixels d'écriture dans chaque colonne $j$ de la fenêtre ($1 \leq j \leq w$).

$$f_{18+j-1} = \frac{1}{H} \sum_{i=1}^{H} I(i, j)$$

Ainsi, pour une fenêtre de largeur $w$ pixels, de hauteur égale à la hauteur de l'image et de décalage entre deux fenêtres consécutives $\delta$ pixels, $17+w$ caractéristiques sont extraites. Les paramètres d'extraction des caractéristiques $w$ et $\delta$ sont optimisés sur une base de validation. Le choix des paramètres optimaux sont présentés dans la section 4.5.2.

## 4.3.4 Apprentissage et décodage avec des HMMs gaussiens

Les modèles de Markov cachés sont une méthode depuis longtemps approuvée et connue pour la modélisation de séquence, comme présenté dans le Chapitre 1, Section 1.2.9. Dans cette section, nous présentons notre utilisation des HMMs pour la reconnaissance de mots manuscrits arabes.

Nous utilisons une approche analytique avec segmentation implicite par l'utilisation de fenêtres glissantes. Les HMMs modélisent les formes des lettres arabes (suivant la position de chaque lettre dans un mot) et les modèles de mots sont construits par la concaténation des modèles de lettres le composant, comme illustré sur la Figure 4.5. Ainsi, cette modélisation permet une fois l'alphabet arabe appris d'utiliser un lexique libre pour le décodage, contrairement à l'approche holistique pour laquelle le lexique est prédéfini et de taille réduit.

### 4.3.4.1 Apprentissage

Nous avons défini un HMM gaussien pour chaque forme de lettre arabe sans signes diacritiques. Tous ces modèles suivent la topologie de *Bakis* (transitions réflexives ou vers

chacun des deux états suivants) illustré dans la Figure 4.6. Cette topologie permet d'absorber les variations de longueur des séquences rencontrées au fil des données : chaque scripteur a sa style d'écriture, ainsi dans la segmentation implicite de mot, une même lettre peut prendre entre 5 et 15 fenêtres glissantes, selon la personne qui a écrit et les conditions d'écriture. Le nombre d'états émetteurs *S* est calculé sur une base de validation.

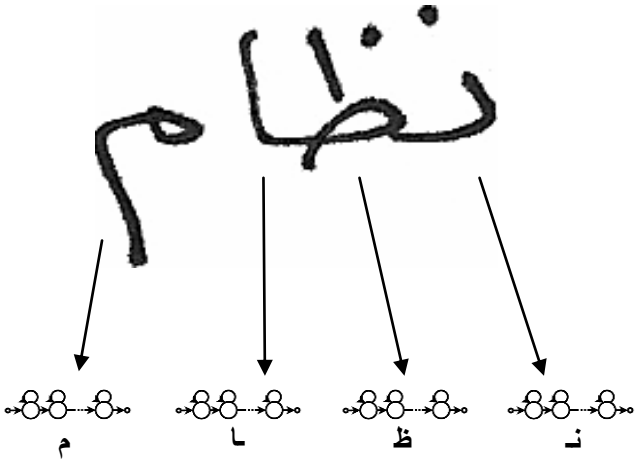



FIG. 4.5 – Le modèle HMM d'un mot est la concaténation des modèles de caractères qui le composent.

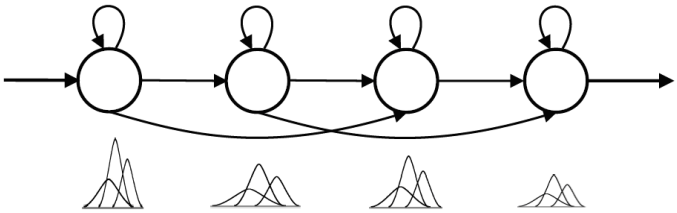


FIG. 4.6 – Illustration du modèle de *Bakis* utilisé pour nos modèles HMMs, où chaque état est représenté par un mélange de distributions gaussiennes.

La densité de probabilité des observations pour chaque état est un mélange de distributions gaussiennes. Le nombre optimal $N_G$ de gaussiennes dans les mélanges est calculé sur une base de validation. Le mélange final dans chaque état est obtenu par incrémentations successives du nombre de gaussiennes, depuis 1 gaussienne jusqu'à $N_G$. A chaque étape d'augmentation du nombre de gaussiennes, les paramèters des HMMs sont ré-estimés avec l'algorithme de *Baum-Welch* . La procédure d'apprentissage avec incrémentations successives du nombre de gaussiennes par mélange est décrite dans l'Algorithme 1.

**Algorithme 1** : Procédure d'apprentissage avec incrémentations successives du nombre de gaussiennes par mélange

$n \leftarrow 1$
**tant que** $n < N_G$ **faire**
  **si** $n < 2$ **alors**
    Ajouter 1 Gaussienne
    $n \leftarrow n + 1$
    Ré-estimation des paramètres avec l'algorithme *Baum-Welch*
  **sinon**
    Ajouter 2 Gaussiennes
    $n \leftarrow n + 2$
    Ré-estimation des paramètres avec l'algorithme *Baum-Welch*
  **fin**
**fin**

### 4.3.4.2 Décodage

En se basant sur les modèles HMM et leurs paramètres, qui ont été fixés lors du processus d'apprentissage, et un lexique de mots, le processus de reconnaissance est effectuée par la recherche de modèles qui correspondent le mieux à une séquence de vecteur de caractéristiques donné. La sortie de la reconnaissance est une ou plusieurs hypothèses de mots. Le décodage se fait avec l'algorithme de *Viterbi*.

## 4.4 Système de catégorisation

Nous avons présenté, dans la Section 4.3 un système de reconnaissance de mots manuscrits arabes. Nous disposons donc d'un outil d'extraction de contenu de documents. Dans cette section, nous présentons notre système de catégorisation de documents. Ce système vise à détecter le thème abordé dans un texte à travers l'examen des mots contenus dans celui-ci. En d'autre terme, affecter une catégorie à un texte donné en fonction de ses caractéristiques.

Pour réaliser notre système de catégorisation, il faut d'abord établir les bases de documents électroniques et manuscrits pour l'apprentissage et le test du système (voir Section

4.5). Ensuite, nous utilisons des techniques présentées dans le Chapitre 3, qui sont nécessaires pour la catégorisation ; pour la représentation des documents, nous nous sommes appuyés sur le modèle vectoriel appelé aussi *sac de mots*, l'un des approches les plus efficaces pour la représentation de textes. Comme nous l'avons vu en Chapitre 3, Section 3.3.1, un système de catégorisation est constitué de trois composantes principales : prétraitements, extraction de caractéristiques et classification, comme le montre la Figure 4.7.

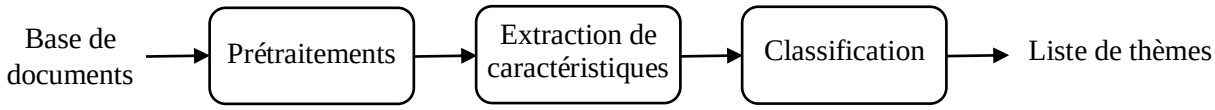



FIG. 4.7 – Processus pour la catégorisation de documents.

## 4.4.1 Prétraitements

Le premier prétraitement est l'élimination de nombres, de ponctuations et de signes diacritiques. La deuxième étape consiste à normaliser certains lettres arabes (tels que (أ) et (إ) à Alif (ا)). Vient ensuite le filtrage des mots vides. Ceci est réalisé grâce à une liste de mots arabes considérés comme les plus fréquents. Enfin, le stemming de mots en utilisant le stemming léger.

## 4.4.2 Extraction de caractéristiques

Dans cette étape, les documents sont convertis en vecteurs de poids dans l'espace des termes utilisés pour décrire les documents. Nous avons choisi d'évaluer les méthodes suivantes : booléenne, fréquence de termes et *TF.IDF*. Nous rappelons ci-dessous la définition de la mesure *TF.IDF* : le poids $w_{ij}$ assigné au terme $i$ dans le document $j$ est défini comme suit :

$$w_{i,j} = TF(i,j) \times IDF(i) = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log\left(\frac{m}{|\{d_j \mid t_i \in d_j\}|}\right)$$

avec :

- $n_{i,j}$ le nombre d'occurrences du terme $t_i$ dans le document $d_j$
- $k$ le nombre de termes du lexique.
- $m$ le nombre de documents dans la base.
- $|\{d_j \mid t_i \in d_j\}|$ le nombre de documents dans lesquels le terme $t_i$ apparaît.

Le lexique à considérer pour la construction des vecteurs de caractéristiques sera déterminé lors de la phase d'apprentissage en effectuant une sélection de caractéristiques.

### 4.4.3 Apprentissage et classification

Le processus d'apprentissage de notre système se déroule en quatre étapes, comme le montre la Figure 4.8. En premier temps, les prétraitements décrits plus haut sont appliqués sur chaque document de la base d'apprentissage. Ensuite, il faut sélectionner parmi les termes issus des prétraitements un sous ensemble de termes fortement discriminants qui ont servi à la représentation vectorielle des documents. Pour cela, nous avons utilisé et évalué les deux méthodes de sélection de caractéristiques suivantes : le Chi-carré (CHI), le gain d'information (GI). Une fois les termes choisis, chaque document de la base est décrit par un vecteur de caractéristiques en utilisant l'une des méthodes suivantes : booléenne, fréquence de termes TF et TF.IDF. La base de vecteurs ainsi obtenue va permettre l'apprentissage du classifieur. Pour réaliser la classification d'un document dans un thème, nous avons choisi d'évaluer 4 algorithmes de classification : le réseau de neurones de type Perceptron Multicouche (PMC), le Bayes naïf (BN), les séparateurs à vastes marges (SVM) et les *k*-plus proches voisins (*k*-ppv).

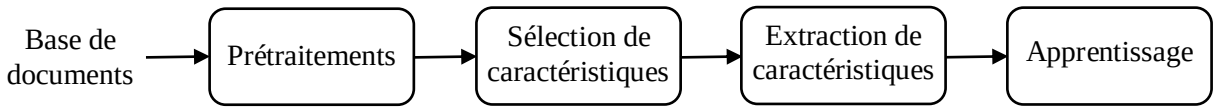



FIG. 4.8 – Processus d'apprentissage du système de catégorisation.

L'apprentissage de notre système de catégorisation nécessite l'apprentissage des paramètres suivants :

- la méthode de sélection de caractéristiques (CHI, GI),
- le nombre de termes à conserver,
- la méthode de pondération des termes,
- le modèle de classification (*k*-ppv, SVM, BN, PMC).

Les trois premiers paramètres influent dans la construction du vocabulaire sélectionné pour la représentation vectorielle. Une fois ces paramètres sont entraînés, nous choisissons ceux qui donnent la meilleure performance de classification.

## 4.5 Expérimentations

Cette section présente les expérimentations que nous avons réalisé pour la mise en œuvre de notre système. En premier temps, nous décrivons nos bases de données utilisées. Nous présentons ensuite les résultats obtenus par les deux systèmes : la reconnaissance et la catégorisation. Enfin, les résultats de test du système complet est présentés et discutés.

Toutes les expérimentations présentées dans cette section sont réalisées avec une machine dotée d'un processeur intel Core i3 et d'une mémoire de 4 GB, sur laquelle est installé le système Windows 7 Professionnel x64 et l'environnement Unix pour Windows, Cygwin 2.870 (64 bit). Les traitements que nous avons implémentés sont codés en langage Java. Pour cela, nous avons utilisé l'environnement de développement intégré (IDE) Eclipse version : Luna Service Release 2 (4.4.2) et la machine virtuelle JRE (Java Runtime Environment) 1.8.0_40 pour l'exécution de nos codes source. Nous avons utilisé aussi les codes source Java des logiciels libres suivants :

- Apache,
- ImageJ 1.48v, pour le traitement d'images,
- Weka 3.7.12, pour l'apprentissage de modèles et la catégorisation de textes.

L'apprentissage et le décodage de nos modèles HMMs sont réalisés par le logiciel HTK [Young et al., 2006].

### 4.5.1 Bases de données

Nous avons construit trois bases de données, décrites ci-dessous.

**Base de documents arabes électroniques (DAE) :** pour la mise en place de système de catégorisation nous avons rassemblé 238 textes arabes à partir de l'internet (archives de presse, sites Web telle que wikipedia, forums, cours universitaires), puis nous avons étiqueté chaque texte manuellement selon le thème (mono-étiquette) auquel il appartient, parmi les 4 thèmes suivants (économie, politique, société, sport). La base DAE est divisée en deux ensembles : l'ensemble d'apprentissage contient deux tiers des documents de chaque thème, alors que l'ensemble de test contient le tiers restant. Le tableau 4.1 récapitule les caractéristiques de la base de documents électroniques DAE.

**Base de documents manuscrits arabes (DMA) :** Pour le test de notre système complet, nous avons construit une base de documents manuscrits à partir de l'ensemble de test de la base

DAE. Ces textes sont écrits par 11 volontaires sur des pages blanches produisant 431 documents manuscrits arabes. Ensuite, ces documents ont été numérisés avec une résolution de 600 dpi. Quelques exemples de documents sont donnés sur la Figure 4.9. Le tableau 4.2 présente les caractéristiques de notre base de documents manuscrits DMA.

| Thème | Nombre de documents |
|---|---|
| Economie | 77 |
| Politique | 51 |
| Société | 51 |
| Sport | 59 |

TAB. 4.1- Effectifs des thèmes de documents sur la base DAE.

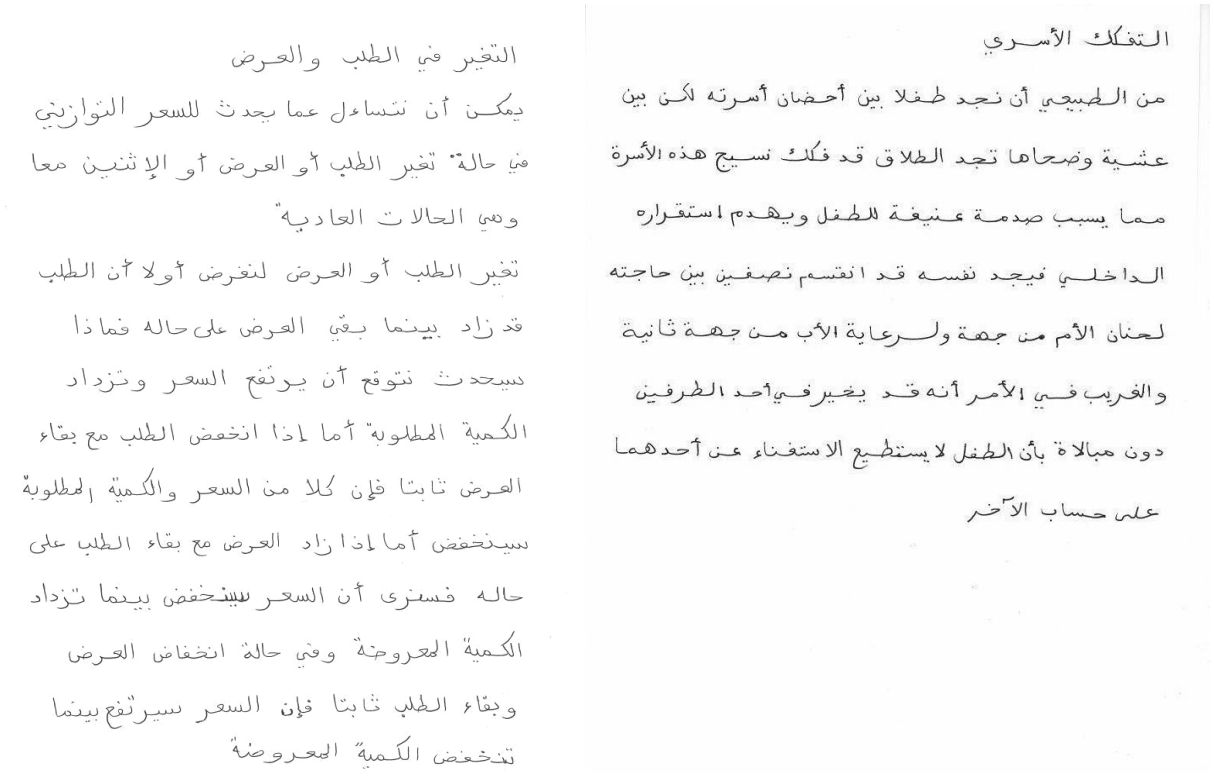


FIG. 4.9 – Exemples de documents manuscrits arabes de la base DMA.

| Thème | Nombre de documents |
|---|---|
| Economie | 29 |
| Politique | 18 |
| Société | 19 |
| Sport | 20 |

TAB. 4.2- Effectifs des thèmes de documents sur la base DMA.

**Base de mots manuscrits arabes (MMA) :** Cette base de données est associée à l'entraînement des HMMs et l'évaluation du système de reconnaissance. Elle est composée de 4438 images de mots manuscrits extraits des documents de la base DMA. Ces images correspondent à un lexique de 905 mots. La base de mots MMA est divisée en trois ensembles: l'ensemble d'apprentissage (70%), l'ensemble de validation (13%) et l'ensemble de test (17%). Chaque image est annotée par sa transcription arabe ainsi que par la séquence des formes de ses lettres, traduites en latin.

## 4.5.2 Système de reconnaissance de mots

### 4.5.2.1 Mise en place du système de reconnaissance

L'efficacité de notre système de reconnaissance dépend des paramètres suivants : les paramètres d'extraction de caractéristiques (la largeur de la fenêtre glissante $w$ et le décalage entre deux fenêtres consécutives $\delta$), la topologie des HMMs (nombre d'états $S$) et le nombre de gaussiennes $N_G$ dans les mélanges de chaque état. Il est nécessaire, donc, d'optimiser ces paramètres sur une base de validation. Une fois ces paramètres choisis, nous évaluons notre système sur la base de test. Dans les expériences présentées ci-après, nous avons crée des modèles HMMs pour les différentes formes de lettres arabes ayant au moins un exemple dans l'ensemble d'apprentissage, soit 125 HMMs. Le tableau 4.3 donne la liste des formes de lettres en arabe et en latin dans la base de mots MMA. Nous avons pris en compte les formes de ligatures verticales de lettres arabes en créant un HMM pour chaque forme présente dans l'ensemble d'apprentissage. Le tableau 4.4 donne les formes de ligatures verticales de lettres que nous avons modélisées avec les lettres arabes les composant.

| étiquette arabe | étiquette latine | étiquette arabe | étiquette latine | étiquette arabe | étiquette latine |
|---|---|---|---|---|---|
| ء | hhA | خ | khA | ف | faB |
| آ | amA | خـ | khB | فـ | faM |
| أ | aeA | ـخـ | khM | ـف | faE |
| إ | ahA | لخ | khMlaB | في | yaEfaB |
| ـإ | ahE | د | daA | ق | kaA |

| | | | | | |
|---|---|---|---|---|---|
| لآ | amElaB | ـد | daE | ق | kaB |
| لأ | aeElaB | ذ | dhA | ـقـ | kaM |
| ـلأ | aeElaM | ـذ | dhE | ـق | kaE |
| لإ | ahElaB | ر | raA | ك | keA |
| ؤ | awA | ـر | raE | ـكـ | keB |
| ـؤ | awE | ز | zaA | ـكـ | keM |
| ئ | alA | ـز | zaE | ـك | keE |
| ـئ | alB | س | seA | ﻛ | kkB |
| ـئـ | alM | ـسـ | seB | ﻜـ | kkM |
| ا | aaA | ـسـ | seM | ل | laA |
| ـا | aaE | ـس | seE | لـ | laB |
| لا | aaElaB | شـ | shB | ـلـ | laM |
| ـلا | aaElaM | ـشـ | shM | ـل | laE |
| ب | baA | ـش | shE | م | maA |
| بـ | baB | ص | saA | مـ | maB |
| بـ | baM | صـ | saB | ـمـ | maM |
| ـب | baE | ـصـ | saM | ـم | maE |
| ت | taA | ـص | saE | لمـ | maMlaB |
| تـ | taB | ض | deA | ن | naA |
| تـ | taM | ضـ | deB | نـ | naB |
| ـت | taE | ـضـ | deM | ـنـ | naM |
| ة | teA | ـض | deE | ـن | naE |
| ـة | teE | ط | toA | ه | heA |
| ث | thA | طـ | toB | هـ | heB |
| ثـ | thB | ـطـ | toM | ـهـ | heM |
| ثـ | thM | ـطـ | toE | ـه | heE |
| ـث | thE | ظ | zeA | و | waA |

| | | | | | |
|---|---|---|---|---|---|
| ج | jaA | ظ | zeB | ـو | waE |
| جـ | jaB | ظـ | zeM | ى | eeA |
| ـجـ | jaM | ـظ | zeE | ـى | eeE |
| ـج | jaE | ع | ayA | لى | eeElaB |
| لجـ | jaMlaB | عـ | ayB | لـى | eeElaM |
| ح | haA | ـعـ | ayM | ي | yaA |
| حـ | haB | ـع | ayE | يـ | yaB |
| ـحـ | haM | غـ | ghB | ـيـ | yaM |
| ـح | haE | ـغـ | ghM | ي | yaE |
| لحـ | haMlaB | ف | faA | | |

TAB. 4.3- Liste des formes de lettres en arabe et en latin dans la base MMA.

| Forme de ligature | Formes lettres le composant | |
|---|---|---|
| لآ | آ | لـ |
| لأ | أ | لـ |
| لإ | إ | لـ |
| لا | ا | لـ |
| لجـ | جـ | لـ |
| لحـ | حـ | لـ |
| لخـ | خـ | لـ |
| في | ي | فـ |
| لمـ | مـ | لـ |
| لى | ى | لـ |

TAB. 4.4- Formes de ligatures verticales de lettres modélisées, et les lettres arabe les composant.

### 4.5.2.2 Optimisation des paramètres

Dans cette section nous allons présenter la procédure d'évaluation des paramètres du système : la largeur de la fenêtre glissante *w*, le décalage entre deux fenêtres consécutives $\delta$, le nombre d'états *S* par HMM et le nombre de gaussiennes $N_G$ dans les mélanges de chaque état. Pour évaluer les meilleurs valeurs possibles de ces paramètres, nous avons fait des expériences sur 3129 images de l'ensemble d'apprentissage et 549 images de l'ensemble de validation de la base MMA. L'extraction de caractéristiques est réalisée en utilisant des fenêtres dont la largeur varie entre $w = 6$ pixels et $w = 12$ pixels avec $\delta \leq w/2$. Nous avons effectué des apprentissages des HMMs des lettres ayant le même nombre d'états émetteurs *S*. Ce nombre varie entre 5 et 16 états et chaque état est représenté par un mélange de 2 gaussiennes. Le taux de reconnaissance correspondant aux différentes valeurs des paramètres *w*, $\delta$ et *S* est donné sur le tableau 4.5. Le décodage est réalisé sur l'ensemble de validation avec un lexique de 322 mots.

D'après les résultats présentés dans le tableau 4.5, nous observons que des valeurs petites de décalage de fenêtres ($\delta \leq 3$) donnent de plus bons résultats que les valeurs plus grandes. Nous choisissons les valeurs de paramètres qui donnent les deux meilleurs taux de reconnaissance : $w$=11, $\delta$=3 et $S$=13 et $w$=11, $\delta$=3 et $S$=14 correspondant, respectivement, au taux de reconnaissance de 90.18% et 91.45%. Pour trouver les valeurs optimales de ces paramètres ainsi que le nombre optimal de gaussiennes dans les mélanges de chaque état, nous avons effectué une deuxième expérience en utilisant les paramètres choisis et en incrémentant le nombre de gaussiennes par mélange suivant la procédure d'apprentissage décrite en Section 4.3.4.1. Les HMMs sont appris toujours sur l'ensemble d'apprentissage et le décodage sur l'ensemble de validation. La Figure 4.10 illustre l'effet du nombre de gaussiennes $N_G$ par mélange sur le taux de reconnaissance pour les deux configurations de paramètres choisis précédemment nommées, respectivement, $w11\delta3S13$ et $w11\delta3S14$.

D'après cette figure, nous choisissons les paramètres optimaux suivants : $w$=11 pixels, $\delta$=3 pixels (ce qui donne 26 caractéristiques différentes par fenêtre), $S$=14 états et $N_G = 4$ gaussiennes dans chaque mélange, correspondant au taux de reconnaissance de 91.82%, c'est la meilleure performance obtenue sur l'ensemble de validation de la base MMA. Nous conservons ces paramètres pour être utilisés dans les expériences qui suivent.

| Paramètres d'extraction de caractéristique | | nombre d'états par HMMs | taux de reconnaissance |
|---|---|---|---|
| largeurs de fenêtres $w$ | décalages de fenêtres $\delta$ | | |
| $w = 6$ | $\delta = 2$ | $S = 14$ | 86.36% |
| $w = 6$ | $\delta = 3$ | $S = 12$ | 88.36% |
| $w = 7$ | $\delta = 2$ | $S = 14$ | 87.64% |
| $w = 7$ | $\delta = 3$ | $S = 11$ | 88.91% |
| $w = 8$ | $\delta = 2$ | $S = 15$ | 88.55% |
| $w = 8$ | $\delta = 3$ | $S = 10$ | 87.64% |
| $w = 8$ | $\delta = 4$ | $S = 8$ | 83.82% |
| $w = 9$ | $\delta = 3$ | $S = 12$ | 89.64% |
| $w = 9$ | $\delta = 4$ | $S = 11$ | 87.82% |
| $w = 10$ | $\delta = 3$ | $S = 12$ | 89.27% |
| $w = 10$ | $\delta = 4$ | $S = 12$ | 88.55% |
| $w = 10$ | $\delta = 5$ | $S = 5$ | 77.09% |
| $w = 11$ | $\delta = 3$ | $S = 13$ | **90.18%** |
| $w = 11$ | $\delta = 3$ | $S = 14$ | **91.45%** |
| $w = 11$ | $\delta = 4$ | $S = 13$ | 88.73% |
| $w = 11$ | $\delta = 5$ | $S = 6$ | 82.00% |
| $w = 12$ | $\delta = 3$ | $S = 16$ | 88.36% |
| $w = 12$ | $\delta = 4$ | $S = 13$ | 88.18% |
| $w = 12$ | $\delta = 5$ | $S = 6$ | 81.27% |
| $w = 12$ | $\delta = 6$ | $S = 6$ | 84.55% |

TAB. 4.5- Comparaison des taux de reconnaissance correspondant aux différentes valeurs de largeur $w$, de décalage $\delta$ de fenêtre et de nombre d'états $S$ par HMMs sur l'ensemble de validation de la base MMA. Pour chaque configuration des paramètres $w$, $\delta$ et $S$, le nombre d'états est fixe pour tous les HMMs de lettres et chaque état est représenté par un mélange de 2 gaussiennes. Les HMMs sont appris sur l'ensemble d'apprentissage de la base MMA.

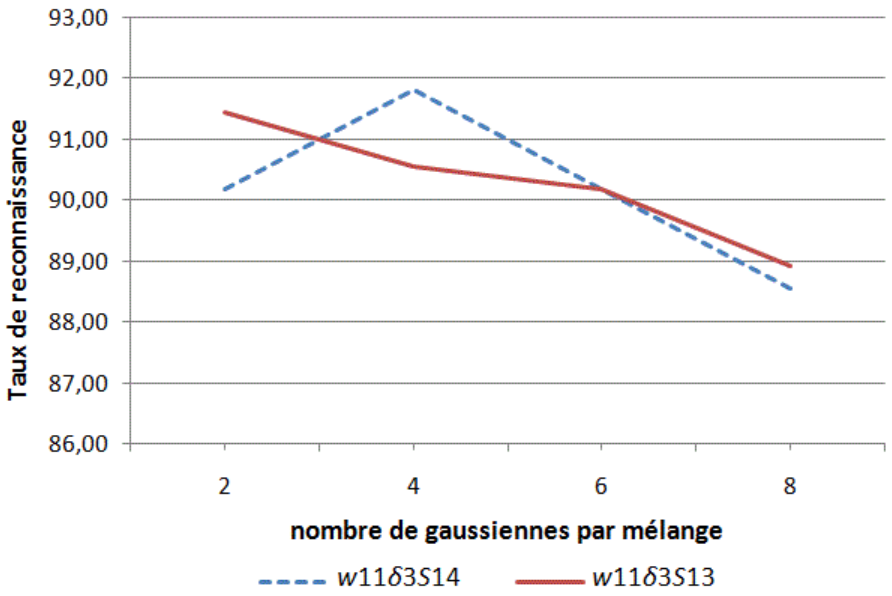



Fig. 4.10 – Influence du nombre de gaussiennes par mélange dans chaque état sur la reconnaissance pour les deux configurations de paramètres $w11\delta 3S13$ et $w11\delta 3S14$. Les HMMs sont appris sur l'ensemble d'apprentissage de la base MMA et testé sur l'ensemble de validation.

### 4.5.2.3 Evaluation sur la base de test

La Section précédente nous ont permis de calculer les valeurs des paramètres de notre système de reconnaissance à l'aide de l'ensemble de validation de la base MMA. Nous pouvons maintenant évaluer ce système sur la base de test. L'évaluation nous a permis de calculer que le taux de reconnaissance du système est de 84.74% sur les 760 images de l'ensemble de teste de la base MMA avec un lexique de taille 488. Nous observons que ce taux est moins que celle obtenu sur l'ensemble de validation. Cette différence peut être justifiée par le suivant :

- l'ensemble de test est constitué d'un ensemble d'images (760) et d'un lexique (488 mots) plus grand que celles de l'ensemble de validation qui est constitué de 549 images et d'un lexique de 322 mots.
- Tous les modèles HMMs des lettres sont appris sur les exemples qu'il possède dans l'ensemble d'apprentissage de la base MMA, dont certains modèles de lettres qui figurent dans l'ensemble de test mais pas dans l'ensemble de validation, ont trop peu

d’exemples pour être appris correctement comme le modèle khA (خ) qui a seulement 3 exemples dans toute la base étiquetés par le mot "المناخ".

## 4.5.3 Système de catégorisation de documents

### 4.5.3.1 Mise en place du système de catégorisation

Afin de pouvoir appliquer notre système de catégorisation sur les documents manuscrits, il est nécessaire de faire l’apprentissage et l’évaluation sur des documents électroniques ce qui permet de ne pas biaiser l’apprentissage par les erreurs de reconnaissance. Pour cela, nous avons utilisé la base des documents électroniques DAE. Une fois notre approche est évalué sur la base DAE, nous pouvons l’évaluer sur la base de documents manuscrits DMA en utilisant les textes issus du système de reconnaissance.

### 4.5.3.2 Evaluation sur la base DAE

Pour l’évaluation de notre méthodologie de catégorisation sur la base de documents DAE, nous avons utilisé des algorithmes de classification très fréquents : le réseau de neurones de type Perceptron Multicouche (PMC), le Bayes naïf (BN), le SVM et le k-ppv. Nous avons choisi deux méthodes de sélection de termes : le Chi-carré (CHI) et le gain d’information (GI). En utilisant ces mesures, les 10, 15, 20, 25 et 30 termes ayant les meilleurs scores dans l’ensemble d’apprentissage ont été choisis comme les termes représentatifs de textes. Enfin, trois méthodes de pondération des termes ont été utilisées : booléenne, fréquence de termes (FT) et TF.IDF. Le taux de réussite totale (micro-average accuracy) de chaque algorithme de classification pour les différents paramètres est présenté dans le tableau 4.6.

D’après les résultats obtenus, les algorithmes PMC et k-ppv montrent la plus grande précision parmi les quatre algorithmes. Ils obtiennent un taux de réussite de 97.47% en utilisant les 30 termes ayant le meilleur score en terme de la méthode de Chi-Carré quel que soit la méthode de pondération des termes.

Dans le tableau 4.7, nous indiquons le classement des quatre algorithmes de classification en fonction de leurs taux de réussite moyens. Les deux dernières colonnes du tableau montrent pour chaque algorithme de classification, le meilleur taux de réussite et les paramètres expérimentaux correspondants : méthode de pondération des termes, méthode de

sélection de termes et nombre de termes après la sélection. Les données montrent la supériorité de l'algorithme k-ppv suivie par le PMC avec de taux de réussite moyens de 93.21 et 93.19 respectivement. Nous observons que les meilleurs taux de réussite pour tous les algorithmes sont obtenues en utilisant le Chi-Carré, les 30 premiers termes et la pondération booléenne, et que pour les deux premiers algorithmes k-ppv et PMC, la meilleure performance est obtenue avec tous les méthodes de pondération.

| Classifieur | Nombre de termes | Booléenne | | FT | | TF.IDF | |
|---|---|---|---|---|---|---|---|
| | | CHI | GI | CHI | GI | CHI | GI |
| PMC | 10 | 88.65 | 85.29 | 88.65 | 85.29 | 88.65 | 85.29 |
| | 15 | 92.43 | 92.85 | 92.43 | 90.33 | 92.43 | 90.33 |
| | 20 | 94.11 | 94.53 | 94.11 | 94.53 | 94.11 | 94.53 |
| | 25 | 95.37 | 95.79 | 95.37 | 95.79 | 95.37 | 95.79 |
| | 30 | **97.47** | 97.05 | **97.47** | 97.05 | **97.47** | 97.05 |
| SVM | 10 | 88.65 | 85.29 | 85.29 | 84.03 | 85.29 | 84.03 |
| | 15 | 92.01 | 92.01 | 91.59 | 89.49 | 91.59 | 89.49 |
| | 20 | 94.11 | 94.53 | 92.85 | 93.27 | 92.85 | 93.27 |
| | 25 | 95.37 | 95.37 | 94.53 | 94.53 | 94.53 | 94.53 |
| | 30 | **97.05** | 96.63 | 96.63 | 94.95 | 96.63 | 94.95 |
| k-ppv | 10 | 88.65 | 85.29 | 88.65 | 85.29 | 88.65 | 85.29 |
| | 15 | 92.43 | 92.85 | 92.43 | 90.33 | 92.43 | 90.33 |
| | 20 | 94.11 | 94.53 | 94.11 | 94.53 | 94.11 | 94.53 |
| | 25 | 95.37 | 95.79 | 95.37 | 96.21 | 95.37 | 96.21 |
| | 30 | **97.47** | 97.05 | **97.47** | 97.05 | **97.47** | 97.05 |
| BN | 10 | 88.23 | 84.87 | 88.23 | 84.87 | 88.23 | 84.87 |
| | 15 | 89.49 | 89.91 | 89.49 | 89.91 | 89.49 | 89.91 |
| | 20 | 93.69 | 91.59 | 89.91 | 91.59 | 89.91 | 91.59 |
| | 25 | 91.59 | 94.11 | 91.59 | 91.59 | 91.59 | 91.59 |
| | 30 | **96.63** | 92.01 | 92.85 | 92.01 | 92.85 | 92.01 |

TAB. 4.6- Taux de réussite des algorithmes de classification PMC , SVM, k-ppv et BN correspondant aux différentes méthodes de sélection de termes, aux différents nombres de termes sélectionnés et aux différentes méthodes de pondération, obtenues sur l'ensemble d'apprentissage de la base de documents DAE.

| Classifieur | Taux de réussite moyen | Meilleure précision | |
|---|---|---|---|
| | | Taux de réussite | Paramètres |
| k-ppv | 93.21 | 97.47 | Booléenne, FT, TF.IDF, CHI, 30 |
| PMC | 93.19 | 97.47 | Booléenne, FT, TF.IDF, CHI, 30 |
| SVM | 92.18 | 97.05 | Booléenne, CHI, 30 |
| BN | 90.54 | 96.63 | Booléenne, CHI, 30 |

TAB. 4.7 – Taux de réussite moyen et le meilleur taux de réussite pour chaque algorithme de classification.

Nous avons choisi la configuration indiquée dans première ligne du tableau 4.7 pour l'évaluation sur la base de documents manuscrits DMA. La section suivante présente cette expérimentation.

### 4.5.4 Evaluation du système complet sur la base DMA

Nous avons choisi par hasard 50 documents de la base DMA, écrits par 8 scripteurs et couvrent les 4 catégories (économie, politique, société et sport). La phase de reconnaissance de documents est réalisée par le système mis en œuvre et validé dans la section 4.5.2. Le décodage est effectué avec un lexique de 850 mots arabes. La catégorisation est réalisée par le système validé en Section 4.5.3 (un classifieur de type k-ppv, la méthode de pondération TF.IDF, la méthode de sélection de termes CHI et les 30 premiers termes). Le tableau 4.8 présente le taux de réussite correspondant à chaque catégorie et la micro-moyenne pour la performance globale du système de catégorisation de documents arabes manuscrits.

| Thème | Taux de réussite | |
|---|---|---|
| | DAE | DMA |
| Economie | 100% | 100% |
| Politique | 100% | 94.10% |
| Sport | 94.10% | 80.00% |
| Société | 100% | 66.70% |
| Micro-moyenne | 97.47% | 90.00% |

TAB. 4.8- Taux de réussite et micro-moyenne obtenus sur les documents électroniques DAE et les documents manuscrits DMA.

Le taux de réussite (micro-moyenne) obtenu sur les documents électroniques (base DAE) est de 97.47%. Ce résultat reflète les performances du système de catégorisation en faisant l'hypothèse d'une reconnaissance parfaite. Avec les documents manuscrits (base DMA), notre système obtient un taux de réussite de 90.00%. La dégradation de performance est due aux erreurs de reconnaissance de mots. Ces erreurs affectent la catégorisation par la perte de mots et l'introduction d'autres mots. Ainsi, le lexique d'origine est peu préservé dans les documents transcrits issus de la reconnaissance. Cela peut influer sur la représentation des documents et par la suite sur la tâche de catégorisation.

Nous constatons que la dégradation des performances en catégorisation sont très inégale suivant les thèmes, le thème le plus stable (économie) a le plus grand nombre de documents (70) dans l'ensemble d'apprentissage. Cela semble logique : ce thème étant très représenté, l'apprentissage du classifieur est plus robuste.

# Conclusion

Ce chapitre présente un système complet de catégorisation des documents manuscrits arabes. Nous avons d'abord décrit d'une manière générale les deux tâches principaux qui composent notre système : la reconnaissance et la catégorisation. Ensuite, chaque tâche est abordée en présentant les différents traitements appliqués. La catégorisation des documents est effectuée sur les transcriptions issues de la tâche de reconnaissance. Enfin, les données expérimentales et les résultats obtenus sont présentés.

Dans la partie expérimentale, nous avons décrits les trois bases de données construites pour la mise en œuvre du système : DMA, MMA, DAE. Ensuite nous avons présenté en détail les expériences réalisées permettant de faire l'apprentissage, de choisir les meilleurs paramètres et d'évaluer les deux systèmes composant notre système complet : la reconnaissance et la catégorisation. Pour la reconnaissance, une optimisation de différents paramètres du système est effectuée par un test exhaustif. Sur la base de mots MMA, nos évaluations indiquent un bon taux de reconnaissance. Pour la catégorisation nous avons évalué et comparé différents algorithmes de classification, méthodes de sélection de termes, méthodes de pondération et nombre de termes utilisé pour décrire les documents. L'évaluation sur la base de documents électroniques DAE nous a permis de choisir les meilleurs paramètres du système.

Enfin, les résultats de l’évaluation ont montrent que notre approche de catégorisation de documents manuscrits permet d'obtenir un taux de réussite de 90.00% sur la base de documents manuscrits construite spécialement pour cette étude. Ces résultats sont satisfaisants et encourageants pour nous, vu l'indisponibilité d’un corpus standardisé de documents manuscrits arabes, et en plus la rareté des travaux de catégorisation de ce type de documents.

# Conclusion

La mise en œuvre d'un système de catégorisation automatique de documents manuscrits arabes implique l'exploitation des techniques issues de plusieurs disciplines scientifiques afin de résoudre un problème complexe. Le travail que nous avons réalisé est constitué des étapes suivantes : en premier temps, une étape nécessaire consiste à la construction des bases de données utiles à la mise en place de notre système. Ensuite, une analyse automatique doit être réalisée sur les documents en entrée du système afin d'en extraire les mots les composant. Ces mots vont être reconnus par un système de reconnaissance de l'écriture manuscrite. Enfin, un système de catégorisation permet de détecter le thème abordé dans le document reconnu à travers l'examen des mots contenus dans celui-ci.

Le premier chapitre a introduit la reconnaissance de formes. Une revue des principales approches de classification avec un état de l'art sont établis. Cette étude nous a permis d'acquérir un ensemble de techniques d'apprentissage automatique nécessaires pour aborder ce qui suivra dans ce document, à savoir la reconnaissance de l'écriture et la catégorisation de documents.

Le deuxième chapitre s'est intéressé à la reconnaissance de l'écriture manuscrite arabe. Nous avons présenté les différents traitements pouvant être appliqués sur les documents afin de les reconnaître. Une présentation de l'état de l'art selon les différentes étape du processus de la reconnaissance, nous a permis de faire le choix des méthodes nécessaire à la mise en œuvre du système de reconnaissance de mots décrit en Chapitre 4.

Dans le troisième chapitre nous avons abordé la fouille de textes. Nous nous intéressons aux techniques et méthodes qui permet de catégoriser, segmenter un texte ainsi que d'extraire d'informations à partir de données textuelles. Les techniques de représentation et de catégorisation de documents sont au cœur de notre travail.

Enfin, dans le quatrième et dernier chapitre nous avons présenté notre système qui a pour but d'apporter la fonctionnalité de catégorisation pour les documents manuscrits arabes. Une description détaillée est faite pour les deux tâches principales du système : la reconnaissance et la catégorisation et les travaux expérimentaux réalisés.

Les résultats des expérimentations montrent que notre système de catégorisation de documents manuscrits arabes a obtenu de bonne performance sur la base de documents manuscrits construite spécifiquement pour cette étude. Ces résultats sont satisfaisants et encourageants pour nous vu l'indisponibilité d'un corpus standardisé de documents manuscrits arabes, et en plus la rareté des travaux de catégorisation de ce type de documents.

# Bibliographie


[Abandah et Anssari, 2009] G. Abandah, N. Anssari. Novel moment features extraction for recognizing handwritten Arabic letters. *Journal of Computer Science*, 5(3), 226-232 (2009).

[Abdullah et al., 2011] T. N. Abdullah, K. Omar, M.F. Nasrudin. Enhancement of Moment Invariants Calculation for Arabic Handwriting Recognition. *International Conference on Pattern Analysis and Intelligent Robotics (ICPAIR)*, 28-29 (2011).

[Aggarwal et Zhai, 2012a] C.C. Aggarwal, C.X. Zhai. An introduction to text mining. Chapter 1 In *Mining Text Data, C.C. Aggarwal, C.X. Zhai (eds.). Springer Science+Business Media, New York*, 1-10 (2012).

[Aggarwal et Zhai, 2012b] C.C. Aggarwal, C.X. Zhai. A survey of text classification algorithms. Chapter 2 In *Mining Text Data, C.C. Aggarwal, C.X. Zhai (eds.). Springer Science+Business Media, New York*, 11-41 (2012).

[Aggarwal et Zhai, 2012c] C.C. Aggarwal, C.X. Zhai. A survey of text clastering algorithms. Chapter 2 In *Mining Text Data, C.C. Aggarwal, C.X. Zhai (eds.). Springer Science+Business Media, New York*, 77-128 (2012).

[Akhateeb et al., 2008] J. H. Alkhateeb, J. Ren, S.S. Ipson, J. Jiang. Knowledge-based baseline detection and optimal thresholding for words segmentation in efficient preprocessing of handwritten Arabic text. In *Proceedings of the 5th International Conference on Information Technology, New Generations (ITNG)*, 1158-1159 (2008).

[Akhateeb et al., 2009] J. H. Akhateeb, J. Jiang, J. Ren, F. Khelifi, S. S. Ipson. Multiclass classification of uncon-strained handwritten Arabic words using machine learning approaches. *The Open Signal Processing Journal*, 2, 21-28 (2009).

[Al-Hajj et al., 2005] R. Al-Hajj, L. Likforman-Sulem, C. Mokbel. Arabic handwriting recognition using baseline dependent features and hidden Markov modeling. In *Proceedings of the 8th International Conference on Document Analysis and Recognition (ICDAR)*, 2, 893-897 (2005).

[Ali et al., 2013] A. S. O. Ali, V. S. a/l Asirvadam, A. S. Malik, A. Aziz. A Geometrical Approach for Age-Invariant Face Recognition. In *Proceedings of the 3rd International Visual Informatics Conference - IVIC 2013, H. Badioze Zaman et al. (eds.). Springer International Publishing Switzerland.* Vol. 8237, 81–96 (2013).

[AlKhateeb et al., 2009] J. H. AlKhateeb, J. Jiang, J. Ren, S.S. Ipson. Component-based segmentation of words from handwritten Arabic text. *International Journal of Computer Systems Science and Engineering*, 5(1), 56-61 (2009).

[Al-Shatnawi et Omar, 2009a] A. Al-Shatnawi, K. Omar. A comparative study between methods of Arabic baseline detection. In *Proceedings of the International Conference on Electrical Engineering and Informatics*, 73-77 (2009).

[Al-Shatnawi et Omar, 2009b] A. AL-Shatnawi, K. Omar. Skew detection and correction technique for Arabic document images based on centre of gravity. *Journal of Computer Science*, 5(5), 363-368 (2009).

[Andrew et Keith, 2011] R. Andrew, D. Keith. Statistical pattern recognition. *WILEY 3rd edition* (2011).

[Aupetit, 2005] S. Aupetit. Contributions aux modèles de Markov caches métaheuristiques d'apprentissage nouveaux modèles et visualisation de dissimilarité. *Thèse de doctorat, Université François-Rabelais Tours* (2005).

[Azeem et Azeem, 2013] S. A. Azeem, H. Ahmed. Effective technique for the recognition of offline Arabic handwritten words using hidden Markov models. *International Journal on Document Analysis and Recognition (IJDAR)*, Springer-Verlag (2013).

[Barrat, 2009] S. Barrat. Modèles graphiques probabilistes pour la reconnaissance de formes. *Thèse de doctorat, Université Nancy 2* (2009).

[Baum et Egon, 1967] L. E. Baum, J. A. Egon. An Inequality with Applications to Statistical Estimation for Probabilistic Functions of a Markov Process and to a Model for Ecology. *Bull, Ants73* (1967).

[Béchet, 2009] N. Béchet. Extraction et regroupement de descripteurs morpho-syntaxiques pour des processus de Fouille de Textes. *Thèse de doctorat, Université Montpellier 2* (2009).

[Belaïd et Ouwayed, 2012] A. Belaïd, N. Ouwayed. Segmentation of ancient Arabic documents. *Guide to OCR for Arabic Scripts*, Springer-Verlag, 103-122 (2012).

[Ben-Dov et Feldman, 2010] M. Ben-Dov, R. Feldman. Text Mining and Information Extraction. Chapter 42 In *Data Mining and Knowledge Discovery Handbook, 2nd edition, O. Maimon, L. Rokach (eds.), Springer Science+Business, New York*, 809-835 (2010).

[Beney, 2008] J. Beney. Classification supervisée de documents, *Hermès Science Publications-Lavoisier, Paris* (2008).

[Benouareth et al., 2008] A. Benouareth, A. Ennaji, M. Sellami. Arabic handwritten word recognition using HMMs with explicit state duration. *Journal on Advances in Signal Processing*, 1 (2008).

[Bergo, 2001] A. Bergo. Text Categorization and Prototypes. *Technical report* (2001).

[Bianne-Bernard et al., 2011] A.-L. Bianne-Bernard, F. Menasri, R. Al-Hajj, C. Mokbel, C. Kermorvant, L. Likforman-Sulem. Dynamic and contextual information in HMM modeling for handwritten word recognition, *IEEE Transactions on Pattern Analysis and Machine*

*Intelligence*, 33(10), 2066-2080 (2011).

[Bianne-Bernard, 2011] A.-L. Bianne-Bernard. Reconnaissance de mots manuscrits cursifs par modèles de Markov cachés en contexte: application au français, a l'anglais et a l'arabe. *Thèse de doctorat, Telecom ParisTech, Institut des Sciences et Technologies* (2011).

[Bikel et al., 1997] D.M. Bikel, S. Miller, R. Schwartz, R. Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, 194-201 (1997).

[Blumenstein et al., 2002] M. Blumenstein, C. K. Cheng, X. Y. Liu. New preprocessing techniques for handwritten word recognition. In *Proceedings of the Second IASTED International Conference on Visualization, Imaging and Image Processing*, 480–484 (2002).

[Boubaker et al., 2009] H. Boubaker, M. Kherallah, A.M. Alimi. New algorithm of straight or curved baseline detec-tion for short Arabic handwritten writing. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR)*, 778-782 (2009).

[Boukerma et Farah, 2010] H. Boukerma, N. Farah. A novel Arabic baseline estimation algorithm based on sub-words treatment. In *Proceedings of the 12th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 335-338 (2010).

[Boukerma et Farah, 2012] H. Boukerma, N. Farah. PAW-IFN-ENIT- une nouvelle base de pseudomots Arabes pour une approche de reconnaissance pseudo analytique. *11th African Conference on Research in Computer Science and Applied Mathematics (CARI)*, 491-499 (2012).

[Boukharouba et Bennia, 2011] A. Boukharouba, A. Bennia. Recognition of handwritten arabic literal amounts using a hybrid approach. *Cognitive Computation*, Springer, 3(2), 382-393 (2011).

[Boukharouba, 2011] A. Boukharouba. Contribution à la segmentation et à la reconnaissance de l'écriture arabe manuscrite. *Thèses de doctorat, Université Mentouri Constantine* (2011).

[Bukhari et al., 2009] S. S. Bukhari, F. Shafait, T. M. Breuel. Script-independent handwritten textlines segmentation using active contours. In *Preceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR),* 2, 446-450 (2009).

[Bunescu et Mooney, 2005] R. Bunescu, R. Mooney. A shortest path dependency kernel for relation extraction. In *Procee-dings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing,* 724-731 (2005).

[Bunescu et Mooney, 2006] R. Bunescu, R. Mooney. Subsequence kernels for relation extraction. In *Advances in Neural Information Processing Systems, Proceedings of the 2005 Conference (NIPS), Y. Weiss, B. Schlkopf, J. Platt (eds.), MIT Press,* 18, 171-178 (2006).

[Caglayan et al. 2013] A. Caglayan, O. Guclu, A. B. Can. A Plant Recognition Approach Using Shape and Color Features in Leaf Images. In *Proceedings of the 17th International Conference of Image Analysis and Processing - ICIAP 2013, Part II, A. Petrosino (ed.). Springer-Verlag Berlin Heidelberg. Vol. 8157,* 161–170 (2013).

[Chan et Roth, 2010] Y.S. Chan, D. Roth. Exploiting Background Knowledge for Relation Extraction. *In Procee-dings of the 23rd International Conference on Computational Linguistics (COLING '10),* 152-160 (2010)

[Chen et al., 1995] M.Y. Chen, A. Kundu, S.N. Srihari. Variable duration hidden Markov and morphological segmentation for handwritten word recognition. *IEEE Trans. Image Process: 4(12)* (1995).

[Chergui et al., 2012] L. Chergui , M. Kef, S. Chikhi. New hybrid Arabic handwriting recognizer. In *Proceedings of the 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, 319-325 (2012).

[Collins et Duffy, 2001] M. Collins, N. Duffy. Convolution kernels for natural language. *In Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA,* 14, 625-632 (2001).

[Cornuejols et Miclet, 2010] A. Cornuejols et L. Miclet. Apprentissage Artificiel : Concepts et algorithmes. *Eyrolles, 2$^{ieme}$ édition* (2010).

[Cristianini et Shawe-Taylor, 2000] N. Cristianini, J. Shawe-Taylor. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, *Cambridge University Press.* (2000).

[Culotta et Sorensen, 2004] A. Culotta, J. Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, 423-429 (2004).

[Cutting et al., 1992] D.R. Cutting, D.R. Karger, J.O. Pedersen, J.W. Tukey. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '92), ACM Press, New York,* 318-329 (1992)

[Ehrmann, 2008] M. Ehrmann. Les Entités Nommées, de la linguistique au TAL : statut théorique et méthodes de désambiguïsation. *Thèse de doctorat, Université Paris 7 - Denis Diderot* (2008).

[Eikvil, 1999] L. Eikvil. Information extraction from World Wide Web - a survey. *Technical Report (Report N$^{o}$. 945), Norweigan Computing Center, Oslo, Norway* (1999).

[El-etriby et Amin, 2010] S.S. El-etriby, K.M. Amin. Detection and correction of deformed historical Arabic manus-cripts. *International Conference on Computer and Communication Engineering (ICCCE)*, 11-12 (2010).

[Ergen et al., 2012] B. Ergen, A. Çinar, G. Aydin. Gabor wavelet and unsupervised Fuzzy C-

means clustering for edge detection of medical images. In *Proceedings of the 6th Innovations in Intelligent Systems and Applications (INISTA), International Symposium on IEEE*, 1–4 (2012).

[Freeman, 1961] H. Freeman. On the encoding of arbitrary geometric configurations. *IRE Transactions on Electronic Computers*, EC-10, 260-268 (1961).

[Fukunaga, 1990] K. Fukunaga. Introduction to Statistical Pattern Recognition. *Academic Press, San Diego* (1990).

[Gosselin, 2000] B. Gosselin. Classification et reconnaissance statistique de formes. *Faculté Polytechnique de Mons* (2000).

[Hartert, 2010] L. Hartert. Reconnaissance des formes dans un environnement dynamique appliquée au diagnostic et au suivi des systèmes évolutifs. *Thèse de doctorat, Université de Reims Champagne-Ardenne* (2010).

[Huang et al., 2003] L. Huang, G. Wan, C. Liu. An improved parallel thinning algorithm. In *Proceedings of the 7th International Conference on Document Analysis and Recognition (ICDAR)*, 780-783 (2003).

[Jamal et Jamal, 2013] A.T. Jamal, C.Y. Suen. Shape-based Analysis for automatic segmentation of Arabic handw-ritten text. *Canadian Conference on Artificial Intelligence (AI)*, Springer-Verlag, 7884, 334-339 (2013).

[Jean-Louis, 2011] L. Jean-Louis. Approches supervisées et faiblement supervisées pour l'extraction d'événem-ents et le peuplement de bases de connaissances. *Thèse de doctorat, Université Paris 11* (2011).

[Jiang et Al-Muhtaseb, 2011] J. Jiang, H. Al-Muhtaseb. Offline handwritten Arabic cursive text recognition using hidden Markov models and re-ranking. *Pattern Recognition Letters*, 32(8), 1081-1088 (2011).

[Jiang, 2010] E.P. Jiang. Content-based spam email classification using machine-learning algorithms. Chapter 6 In *Text mining-Applications And Theory, M.W. Berry, J. Kogan (eds.), John Wiley & Sons, Ltd, Chichester, United Kingdom*, 37-56 (2010).

[Jung et al., 2012] H. Jung, S.-P. Choi, S. Lee, S.-K. Song. Survey on Kernel-Based Relation Extraction. Chapter 1 In *Theory and Applications for Advanced Text Mining, S. Sakurai (eds.), InTech*, 1-36 (2012).

[Jung, 2012] J. Jiang. Information Extraction From Text. Chapter 2 In *Mining Text Data, C.C. Aggarwal, C.X. Zhai (eds.). Springer Science+Business, New York*, 11-41 (2012).

[Kessentini et al., 2010] Y. Kessentini, T. Paquet, A. BenHamadou. Off-line handwritten word recognition using multi-stream hidden Markov models. *Pattern Recognition Letters*, 31(1), 60-70 (2010).

[Khayyat et al., 2012] M. Khayyat, L. Lam, C. Y. Suen, F. Yin, C.-L. Liu. Arabic handwritten

text line extraction by applying an adaptive mask to morphological dilation. In *10th IAPR International Workshop on Document Analysis Systems (DAS)*, 100-104 (2012).

[Khayyat et al., 2012] M. Khayyat, L. Lam, C. Y. Suen, F. Yin. Arabic handwritten word spotting using language models. In *Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 43-48 (2012).

[Khoja et Garside] S. Khoja and R. Garside. Stemming arabic text. Computing Department, Lancaster University, Lancaster, UK (1999).

[Kriouile et al., 1990] A. Kriouile, J.-F. Mari, J.-P. Haton. Some Improvements in Speech Recognition Algorithms Based on HMM. *In IEEE-ICASSP* (1990).

[Kubanek et al., 2013] M. Kubanek, D. Smorawa, L. Adrjanowicz. Users Verification Based on Palm-Prints and Hand Geometry with Hidden Markov Models. In *Proceedings of The 12th International* Conference *on* Artificial Intelligence *and* Soft Computing *ICAISC 2013, L. Rutkowski et al. (eds.). Part II, Lecture Notes in Artificial Intelligence (LNAI). Springer-Verlag Berlin Heidelberg,* 7895, 275–285 (2013).

[Kukich, 1992] K. Kukich. Techniques for Automatically Correcting Words in Text. *ACM Computing Surveys*, 24(4), 377-439 (1992).

[Kumar et al., 2010] J. Kumar, W. Abd-Almageed, L. Kang, D. Doermann. Handwritten Arabic text line segmen-tation using affinity propagation. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (DAS),* 135-142 (2010).

[Landeghem et al., 2008] S.V. Landeghem, Y. Saeys, B.D. Baets, Y.V.D. Peer. Extracting protein-protein interactions from text using rich feature vectors and feature selection. *In T. Salakoski, D. Rebholz-Schuhmann, S. Pyysalo (eds.).* In *Proceedings of the 3rd symposium on semantic mining in biomedicine (SMBM '08), Turku Centre for Computer Sciences (TUCS),* 77-84 (2008).

[Larkey et al., 2007] L. Larkey, L. Ballesteros and M. Connell. Light stemming for arabic information retrieval. In Arabic computational morphology, Springer, 221-243 (2007)

[Laroum et al., 2010] S. Laroum, N. Béchet, H. Hamza, M. Roche. Classification automatique de documents bruités à faible contenu textuel. *Revue des Nouvelles Technologies de l'Information (RNTI), Numéro spécial : Fouille de Données Complexes,* Vol. E-18 (2010).

[Lawgali et al.,2011] A. Lawgali, A. Bouridane, M. Angelova, Z. Ghassemlooy. Automatic segmentation for Arabic characters in handwriting documents, In *Proceedings of the 18th IEEE International Confer-ence on Image Processing*, 3529-3532 (2011).

[Lemaitre, 2008] M. Lemaitre. Approche markovienne bidimensionnelle d'analyse et de reconnaissance de documents manuscrits. *Thèse de doctorat, Université Paris 5 René Descartes* (2008).

[Lewis, 1998] D.D. Lewis. Naive (Bayes) at forty : The independence assumption in information retrieval. In *Proceedings of ECML-98, 10th European Conference on Machine*

*Learning, C. Nédellec, C. Rouveirol (eds.) . Springer Verlag, Heidelberg, DE*, 4-15 (1998).

[Likforman-Sulem et al., 2012] L. Likforman-Sulem, R. Al-Hajj, C. Mokbel, F. Menasri, A.-L. Bianne-Bernard, C. Kermorvant. Features for HMM-based Arabic handwritten word recognition systems. *Guide to OCR for Arabic Scripts*, Springer-Verlag, 123-143 (2012).

[Liu et al., 2002] C. L. Liu, H. Sako, H. Fujisawa. Performance evaluation of pattern classifiers for handwritten character recognition. *International Journal on Document Analysis and Recognition* (2002).

[Menasri, 2008] F. Menasri. Contributions à la reconnaissance de l'écriture arabe manuscrite. *Thèse de doctorat, Université Paris Descartes* (2008).

[Menasri, 2008] F. Menasri. Contributions à la reconnaissance de l'écriture Arabe manuscrite. *Thèse de doctorat, Uni-versité Paris Descartes* (2008).

[Milgram et al., 2005] J. Milgram, R. Sabourin, M. Cheriet. Système de classification à deux niveaux de décision combinant approche par modélisation et machines à vecteurs de support. *Traitement du Signal, 22(3) :293-304* (2005).

[Moghaddam et Cheriet, 2009] R. Moghaddam, M. Cheriet. Application of multilevel classifier and clustering for automatic word spotting in historical document images. In *Proceedings of the 10th Proceedings of the 11th International Conference on Document Analysis and Recognition (ICDAR)*, 511-515 (2009).

[Mohamad et al., 2009] R. A. Mohamad, L. Likforman-Sulem, C. Andmokbel. Combining slanted-frame classifiers for improved HMM-based arabic handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(7), 1165-1177 (2009).

[Mountassir et al., 2012] A. Mountassir, H. Benbrahim et I. Berrada. A cross-study of Sentiment Classification on Arabic corpora, Research and Development in Intelligent Systems XXIX, 259-272 (2012).

[Mozaffari et al., 2008] S. Mozaffari, K. Faez, V. Märgner, H. El-Abed. Lexicon reduction using dots for off-line farsi/arabic handwritten word recognition. *Pattern Recognition Letters*, 29(6), 724-734 (2008).

[Nasreddine, 2010] K. Nasreddine. Recalage de signaux et reconnaissance de formes Application à l'analyse des otolithes de poissons. *Thèse de doctorat, Université de Bretagne Occidentale* (2010).

[Natarajan et al., 2011] P. Natarajan, D. Belanger, R. Prasad, M. Kamali, K. Subramanian, P. Natarajan. Baseline dependent percentile features for offline Arabic handwriting recognition. In *Preceedings of the 11th International Conference on Document Analysis and Recognition (ICDAR)*, 329-333 (2011).

[Parvez et Mahmoud, 2013a] M. T. Parvez, S. A. Mahmoud. Offline Arabic handwritten text recognition- a survey. *ACM Computing Surveys*, 45(2), Article 23 (2013).

[Parvez et Mahmoud, 2013b] M. T. Parvez, S. A. Mahmoud. Arabic handwriting recognition using structural and syntactic pattern attributes. *Pattern Recognition*, 46(1), 141-154 (2013).

[Pavlidis, 1982] T. Pavlidis. Algorithms for graphics and image processing. *Rockville, Md, Computer Science Press* (1982).

[Pechwitz et al., 2012] M. Pechwitz, H. El Abed, V. Märgner. Handwritten Arabic word recognition using the IFN/ENIT-database. *Guide to OCR for Arabic Scripts*, Springer-Verlag, 169-213 (2012).

[Piskorski et Yangarber, 2013] J. Piskorski, R. Yangarber. Information Extraction : Past, Present and Future. Chapter 2 In *Multi-source, Multilingual Information Extraction and Summarization, T. Poibeau et al. (eds.), Springer-Verlag, Berlin, Heidelberg*, 23-49 (2013).

[Rabiner, 1989] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of IEEE,* vol. 77 :257-286 (1989).

[Saleem et al., 2009] S. Saleem, H. Cao, K. Subramanian, M. Kamali, R. Prasad, P. Natarajan. Improvements in BBN's HMM-based offline Arabic handwriting recognition system. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR)*, 773-777 (2009).

[Salton et al., 1975] G. Salton, A. Wong, C.S. Yang. A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 18(11), 613-620 (1975).

[Salton et Buckley, 1988] G. Salton, C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5), 513-523 (1988).

[Sarfraz et al., 2007] M. Sarfraz, S.A. Mahmoud, Z. Rasheed. On skew estimation and correction of text. In *Proc-eedings of the Conference on Computer Graphics Imaging and Visualization (CGIV)*, 308-313 (2007).

[Sayre, 1973] K. Sayre. Machine recognition of handwritten words: A project report. *Pattern Recognition*, 5(3), 213-228 (1973).

[Sebastiani, 2002] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1),1-47 (2002).

[Shi et al., 2013] Z. Shi, Y. Liu, Q. Li. Medical Image Segmentation Based on FCM and Wavelets. In *Proceedings of the 4th International Conference on Intelligence Science and big Data Engineering (IScIDE 2013), C. Sun et al. (eds.). Springer-Verlag Berlin Heidelberg.* Vol. 8261, 279–286 (2013).

[Sidère, 2012] N. Sidère. Contribution aux méthodes de reconnaissance structurelle de formes : approche à base de projection de graphes. *Thèse de doctorat, Université François Rabelais de Tours* (2012).

[Slimane et al., 2012] F. Slimane, S. Kanoun, J. Hennebert, R. Ingold, A. M. Alimi. A new baseline estimation method applied to arabic word recognition. In *Proceedings of the 10th*

*IAPR International Workshop on Document Analysis Systems (DAS)* (2012).

[Srivastava et al., 2013] V. Srivastava, B. K. Tripathi, V. K. Pathak. Evolutionary Fuzzy Clustering and Functional Modular Neural Network-Based Human Recognition. In*: Neural Computing and Applications*, 22 (Suppl. 11), 411–419 (2013).

[Toussaint, 2011] Y. Toussaint. Fouille de textes : des méthodes symboliques pour la construction d'ontologies et l'annotation sémantique guidée par les connaissances. *Projet de Recherche, Université Henry Poincaré, Nancy 1* (2011).

[Triantaphyllou, 2010] E. Triantaphyllou. Data Mining of Text Documents. Chapter 13 In *Data Mining and Knowle-dge Discovery via Logic-Based Methods, E. Triantaphyllou (eds.), Springer Science+Business, New York*, 257-276 (2010).

[Tsopzé, 2010] N. Tsopzé. Treillis de Galois et réseaux de neurones : une approche constructive d'architecture des réseaux de neurones. *Thèse de doctorat, Université d'Artois* (2010).

[Valuvanathorn et al., 2013] S. Valuvanathorn, S. Nitsuwat, M. L. Huang. Multi-Feature Face Recognition Based on 2D-PCA and SVM. *The Era of Interactive Media, J.S. Jin et al. (eds.). Springer Science+Business Media, LLC*, 65-75 (2013).

[Vapnik, 1995] V.N. Vapnik. The Nature of Statistical Learning Theory. *Springer, New York* (1995).

[Weiss et al., 2010] S.M. Weiss, N. Indurkhya, T. Zhang. Looking for Information in Documents. Chapter 6 In *Fundamentals of Predictive Text Mining, S.M. Weiss, N. Indurkhya, T. Zhang (eds.). Springer-Verlag, London Limited*, 113-139 (2010).

[Wen-ge, 2012] F. Wen-ge. Application of SVM Classifier in IR Target Recognition. In *Proceeding of the International Conference on Applied Physics and Industrial Engineering (ICAPIE-2012), Physics Procedia (Elsevier B.V)*, Vol. 24, Part C, 2138-2142 (2012).

[Xiaobin et al., 2013] W. Xiaobin, L. Hao, W. Lijuan, H. Qu. Genetic Algorithm Based Neural Network for License Plate Recognition. In *Proceedings of the 10th International Symposium on Neural Networks, ISNN 2013, C. Guo, Z.-G. Hou, Z. Zeng (eds.), Part I. Lecture Notes in Computer Science, Advances in Neural Networks. Springer-Verlag Berlin Heidelberg*. Vol. 7951, 391–400 (2013).

[Young et al., 2006] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev et P. Woodland : The HTK Book V3.4. *Cambridge University Press, Cambridge, UK* (2006).

[Zelenko et al., 2003] D. Zelenko, C. Aone, A. Richardella. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3,1083-1106 (2003).

[Zhang et al., 2006] M. Zhang, J. Zhang, J. Su, G. Zhou. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st*

*International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, 825-832 (2006).

[Zhang et al., 2006] M. Zhang, J. Zhang, J. Su. Exploring syntactic features for relation extraction using a convolution tree kernel. In *Proceedings of the Human Language Technology Conference of the North American. Chapter of the Association for Computational Linguistics*, 288-295 (2006).

[Zhang et Suen, 1984] T. Y. Zhang, C. Y. Suen. A fast parallel algorithm for thinning digital patterns. In: *Communications of the ACM*. Vol. 2, 236–239 (1984).

[Zhao et Karypis, 2011] Y. Zhao, G. Karypis. Document Clustering. In *Encyclopedia of Machine Learning, C. Sammut, G.I. Webb (eds.). Springer Science+Business Media LLC*, 293-298 (2011).

[Ziaratban et Faez, 2009] M. Ziaratban, K. Faez. Non-Uniform slant estimation and correction for farsi/arabic handw-ritten words. *International Journal on Document Analysis and Recognition (IJDAR)*, 12(4), 249-267 (2009).