وزارة التعليم العالي و البحث العلمي

BADJI MOKHTAR ANNABA UNIVERSITY
UNIVERSITÉ BADJI MOKHTAR ANNABA

جامعة باجي مختار – عنابة

Année 2012

FACULTE DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'INFORMATIQUE

MEMOIRE

Présenté en vue de l'obtention de diplôme de **MAGISTERE**

Indexation et recherche de documents par le contenu

Option :

Intelligence Artificielle

raP
BENCHALEL AMIR

DIRECTEUR DE MEMOIRE : Pr. M. Sellami Université Badji Mokhtar – Annaba

Président:

Dr. Nadir Farah	Prof	Université Badji Mokhtar – Annaba

Examinateurs:

Dr. Halima Bahi	MC-A	Université Badji Mokhtar – Annaba
Dr. Yamina Mohamed-Benali	MC-A	Université Badji Mokhtar – Annaba

Avant Propos

Ecrire un avant propos est toujours un moment de grande émotion. II s'agit, en effet, de remercier ceux qui nous ont aidés et d'exprimer nos sentiments à l'égard de ceux que nous aimons.

Je commencerai donc par remercier le Dr. Toufik Sari, le* Dr. Seridi hassina*, le Dr Salima Hacini et Mlle Meriem Abdeljoued qui m'ont fournit toute l'aide dont j'avais besoin pour la réalisation de ce travail, qui m'ont soutenue et qui m'ont été d'une réelle utilité, et sans qui ce modeste travail n'aurai jamais vu le jour.

Exprimer ses sentiments à sa mère, son père et sa sœur est déjà difficile bien que naturel et quant à exprimer son estime à ses Maitres, cela semble bien plus difficile encore.

Cependant, je n'hésite pas à le faire à l'endroit de tout le corps professoral du Département d'informatique qui a largement déjà, et continue toujours, de contribuer à ma formation.

Je n'hésite pas surtout, à l'occasion de la remise de ce modeste travail, d'exprimer à mon encadreur, le Pr. Mokhtar Sellami, toute ma gratitude pour n'avoir pas, un seul instant pendant toute une année de sérieux labeur, failli à sa mission en me faisant bénéficier, sans restriction aucune, de son savoir, ses compétences et son expérience.

Benchalel Amir

Résumé

Bien que les domaines de traitement d'images et de reconnaissance de formes aient fait des progrès considérables dans différents champs d'application, les étapes de localisation et de caractérisation adéquates d'objets restent des processus très délicats et non triviaux.

Caractériser des objets par leur forme est une tâche délicate dans plusieurs applications de vison par ordinateur. Différentes approches théoriques très intéressantes existent mais qui, néanmoins, restent très couteuses en terme de temps de calcul. Dans la recherche d'image par le contenu (RIC) « Content-Based Image Retrieval (CBIR)» la forme est une caractéristique de bas niveau. Elle doit satisfaire différentes propriétés : invariance, robustesse, compacité, complexité faible et mesurabilité perceptible.

Dans les domaines où l'image possède un contenu complexe, mélange entre objets textuels et objets graphiques par exemple, les images doivent être tout d'abord segmentées en texte/graphique. On parle alors de séparation texte/graphique.

L'objectif de notre travail est de faire une étude bibliographique et expérimentale des différentes manières de caractériser des objets qu'on rencontre dans les images. Nous nous intéressons néanmoins à deux classes uniquement qui sont les objets **textuels** et **non-textuels**.

Nous allons étudier et implémenter un très grand nombre de descripteurs de formes très connus dans la littérature qu'on va tester dans le domaine de la séparation texte/graphique.

Nous allons aussi présenter AdaBoost avec les arbres de décision, les réseaux de neurones artificiels, C-means et Fuzzy C-means et les implémenter dans un même système.

Beaucoup de test vont être effectués, les classifieurs vont être testés sur chacun des descripteurs indépendamment et en combinaison.

Nous allons également présenter et implémenter notre méthode de segmentation automatique d'image basée sur la transformée de distance. Le résultat sera un nouveau jeu de test assez conséquent composé d'objets textes et objets graphiques mais séparés automatiquement évidemment .

Nous montrerons par la suite que les résultats de validation prouvent encore une fois la robustesse de la méthode AdaBoost avec un temps de réponse très appréciable. De point de vue descripteurs, nous noterons que ce sont les moments de Hu qui ont donné les meilleurs résultats.

Abstract

Although the areas of image processing and pattern recognition have made considerable progress in various fields of application, the steps for locating and characterizing appropriate objects are very delicate process and not trivial.

Characterize objects by their shape is a delicate task in many computer vision applications. Different theoretical approaches are interesting but which nevertheless remain very expensive in terms of computation time. In the image search by content "Content-Based Image Retrieval (CBIR)" form is characteristic of low level. It must meet different properties: invariance, robustness, compactness, low complexity and measurability noticeable.

In areas where the image has complex content, such as mixtures of graphic objects and text objects images must first be segmented into text / graphics. This is called separation text / graphics.

The aim of our work is to do a literature survey and experimental ways to characterize objects encountered in the images. Our interest, however only two classes those are text objects and non-text objects.

We will study and implement a large number of descriptors of shapes known in the literature that will be tested in the field of separation text / graphics.

We will also introduce AdaBoost with decision trees, artificial neural networks, C-means and Fuzzy C-means and implement them in the same system.

Many tests will be performed, the classifiers will be tested on each descriptor independently and in combination.

We will also introduce and implement our method of automatic image segmentation based on the distance transform. The result will be a new set quite big of test objects consisting of text and graphic objects, but clearly separated automatically.

We will show later that the validation results show once again the robustness of the method AdaBoost with a response time significantly. Point of view descriptors, we note that these are the moments in general who gave the best results.

Keywords

Content-Based Image Retrieval, CBIR, descriptor, classifier, AdaBoost, C-Mean, fuzzy C-Mean, artificial neural networks, segmentation text/graphics, separation text/ graphics.

ملخص

على الرغم من أن مجالات معالجة الصور والتعرف على نمط قد حققت تقدما كبيرا في مختلف مجالات التطبيق ، والخطوات اللازمة لتحديد وتوصيف كائنات المناسبة عملية حساسة جدا وتافهة لا.

تميز الكائنات التي شكلها هو مهمة حساسة في كثير من التطبيقات الرؤية الكمبيوتر. المقاربات النظرية المختلفة مثيرة للاهتمام .ولكنها تبقى مع ذلك مكلفا جدا من حيث الوقت اللازم للحساب.

في البحث عن الصور من خلال المحتوى (RIC) "استرجاع الصور على المحتوى النموذج (CBIR)' هو سمة من مستوى منخفض. يجب أن تلبي خصائص مختلفة : اللاتبدل invariance ومتانة بنيتها والاكتناز، والتعقيد منخفض و ملحوظ للقياس.

في المناطق التي تكون فيها صورة ومحتوى معقد، لا بد من خليط الكائنات الرسومية وكائنات النص مثل يجب أن يكون أول صور مجزأة في النص / الرسومات. يسمى هذا الفصل النص / الرسومات.

والهدف من عملنا هو القيام بمسح الأدب وطرق تجريبية لتحديد خصائص الأجسام التي تصادف في الصور. مصلحتنا ، ولكن اثنين فقط من الفئات التي هي غير كائنات النص و لا النص.

وسوف نقوم بدراسة وتنفيذ عدد كبير من واصف من الأشكال المعروفة في الأدبيات التي سيتم اختبارها في مجال فصل النص / الرسومات.

وسوف نقدم أيضا AdaBoost بأشجار القرار ، والشبكات العصبية الاصطناعية ,"C-means " و "Fuzzy C-means " وتنفيذها في نفس النظام.

سيتم تنفيذ العديد من الاختبارات ، وسوف يتم اختبار كل مصنف على نحو مستقل واصف في الجمع.

وسوف نقدم أيضا وتنفيذ أسلوبنا في تجزئة صورة آلي يعتمد على المسافة التحويل. وسوف تكون النتيجة مجموعة جديدة من الكائنات الاختبار يتألف من نص كبيرة جدا ، والكائنات الرسومية ، ولكن يفصل بوضوح تلقائيا.

وسوف نعرض لاحقا ان التحقق من صحة النتائج تظهر مرة أخرى على متانة الأسلوب AdaBoost مع وقت الاستجابة بشكل ملحوظ. نقطة اصفات الرأي ، فإننا نلاحظ أن هذه هي لحظات في العام الذي أعطى أفضل النتائج.

الكلمات الرئيسية :
البحث عن الصور بواسطة اصفات المحتوى ، المصنف ، AdaBoost ، C-means ، Fuzzy C-means ، الشبكات

العصبية الاصطناعية ، تجزئة نص / رسومات ، الانفصال نص / رسومات .

Table des matières

Introduction générale

La représentation des formes est un problème fondamental des nouvelles applications multimédias. Caractériser des objets par leur forme est une tâche délicate dans plusieurs applications de vision par ordinateur. Différentes approches théoriques très intéressantes existent mais qui, néanmoins, restent très coûteuses en terme de temps de calcul. Dans la recherche d'image par le contenu (RIC) « Content-Based Image Retrieval (CBIR)» la forme est considérée comme une caractéristique de bas niveau. Elle doit satisfaire différentes propriétés : invariance, robustesse, compacité, complexité faible et mesurabilité perceptible. Une panoplie de représentations des formes a été proposée pour une variété d'objectifs. Ces méthodes peuvent toutefois être regroupées en deux grandes classes « basée contour » et « basée région ». Les méthodes basées sur le contour, telles que la chaine de codes, la signature des formes, l'approximation polygonale, etc., exploitent les informations de bordure des objets qui sont prépondérantes dans la perception humaine afin de juger les similarités entre formes. Les méthodes basées région, telles que les moments géométriques, les moments de Zernike ou la représentation des surfaces, exploitent au contraire les informations internes à la forme, et par conséquent peuvent être appliquées à plus d'un type de formes.

Dans les domaines où l'image possède un contenu complexe, tel que le mélange entre objets textuels et objets graphiques, les images doivent être tout d'abord segmentées en texte/graphique. On parle alors de séparation texte/graphique. Cette rude besogne est apparue avec les premiers systèmes de tri automatique du courrier où les colis postaux contiennent, en plus des adresses, les cachets et les timbres. Si ces derniers types de graphiques sont assez simples à localiser et à extraire, vu qu'ils possèdent des formes régulières, les graphiques qu'on rencontre dans les images sur internet, par exemple, peuvent avoir différentes formes, tailles, orientations, couleurs, etc.

Dans ce contexte, notre travail consiste à réaliser une étude qui vise à identifier la plupart des descripteurs de forme qui existent, à implémenter plusieurs méthodes de différentes classes et à les tester dans la séparation texte/graphique.

Le mémoire est structuré en quatre chapitres :

Chapitre I : illustre la variété de descripteurs de forme qui existent dans la littérature indépendamment du domaine d'application. En effet, ce chapitre a pour objectif de passer en revue les différents descripteurs de forme selon deux points de vues : mathématique et applicabilité.

Chapitre II : afin de tester les descripteurs, une classification doit être implémentée. A cet effet, le chapitre II expose les différentes approches de classification.

Chapitre III : ce chapitre présente la méthodologie employée ainsi que les détails des différents modules.

Chapitre IV : ce chapitre concerne le protocole expérimental, les descripteurs testés ainsi que les classifieurs implémentés.

Chapitre I

Les descripteurs

1.1 Introduction

En reconnaissance de formes plusieurs étapes sont nécessaires afin d'identifier les objets en entrées. L'extraction de caractéristiques, ou description, est l'étape la plus importante car elle permet de bien décrire les formes à reconnaître. Une mauvaise caractérisation engendre impérativement des erreurs de classification, par contre une bonne description faciliterait la tâche du classifieur. Cette phase intervient après la segmentation. Une fois l'image segmentée, on peut représenter les régions obtenues par des descripteurs de formes.

On peut représenter une forme comme une région, par ses contours, localement en utilisant des points d'intérêts ou par sa texture pour extraire les descripteurs.

Ce chapitre présente les différentes manières de représentation des formes quelque soit leur nature et leur utilisation.

1.2 Forme et objet

La reconnaissance de formes c'est détecter et localiser des formes dans les images.

On ne raisonne plus en terme de "pixel par pixel" mais en terme d'objet.

Mais la détection d'objets implique un nombre important de difficultés dont la détermination de critères pertinents permettant la reconnaissance d'un objet d'où l'utilité d'utiliser une mesure qui va représenter l'image par la suite.

C'est cette mesure qu'on appelle descripteur de formes ou d'objets.

1.3 Descripteurs de formes

Il existe plusieurs classes de descripteurs de formes :

1.3.1 Les descripteurs contours

Ce sont des descripteurs obtenus à partir des contours externes de l'objet.

1.3.1.1 Le codage de chaîne

A partir d'un point de contour initial, on traverse le contour en codant les points consécutifs par la direction du déplacement. Les chiffres de 0 à 7 sont utilisés pour coder les 8 directions principales. Le codage est plus économique que représenter le contour par les coordonnées des points de contours, et il est invariant à la translation. Pour assurer l'invariance à la rotation on utilise le codage de chaîne différentiel en calculant les différences des directions des déplacements consécutifs en mode 8 connexité. La sélection du point de départ qui donne le code avec la plus petite valeur numérique rend le code invariant à la sélection du point de départ.

1.3.1.2 Application d'une transformation globale

On peut également caractériser une fonction par une description fréquentielle. Un exemple de description fréquentielle globale est la transformée de Fourier.

L'intérêt de la transformée de Fourier, ainsi que des autres méthodes de description fréquentielle, réside dans le fait que la phase est normalisée, c'est à dire indépendante de la luminosité des images ainsi que de leur contraste.

Les transformées de Fourier et de Mellin

La caractérisation d'un signal f la plus connue dans le domaine des fréquences est sans aucun doute la transformée de Fourier [1-2-01], dont la formulation continue dans le cas bidimensionnel est :

$$F(u,v) = \iint f(x,y)e^{-i(ux+vy)}dxdy$$

La transformée de Fourier d'une paire de fréquences $(u,v) \in R^2$ est caractérisée par une amplitude et une phase. L'intérêt de son utilisation (dans sa version continue comme discrète) réside dans le fait qu'au moins l'un de ces deux attributs reste constant à travers certaines transformations de l'image.

Des variantes existent sous certaines contraintes, comme notamment la transformée de Fourier circulaire [1-2-02], pour laquelle l'amplitude est rendue invariante à la rotation. Dans [1-2-03], une méthode est également développée pour rendre cette caractérisation invariante à la translation, à la rotation et au changement d'échelle.

Il existe également une autre transformée, la transformée de Mellin, qui s'applique sur des nombres complexes dans sa forme la plus générale et sa formulation bidimensionnelle pour une paire de fréquences$(u,v) \in R^2$ est :

$$z(u,v) = \int_0^\infty \int_0^\infty f(x,y)x^{iu-1}y^{iv-1}dxdy$$

La transformée de Mellin peut être vue comme étant la Transformée de Fourier appliquée après un changement de variable exponentiel. Si elle est utilisée avec des valeurs imaginaires pures, alors son amplitude est invariante au changement d'échelle. Comme pour la transformée de Fourier, il en existe une variante, appelée la Transformée de Fourier-Mellin, qui permet d'obtenir en plus l'invariance à la rotation [1-2-04].

Des travaux ont été réalisés pour combiner les Transformées de Fourier et de Mellin et ainsi obtenir l'invariance à la fois à la translation, à la rotation et au changement d'échelle [1-2-05].

Un état de l'art complet sur les Transformées de Fourier et de Mellin peut etre trouvé dans [1-2-06].

La transformée de Gabor :

La transformation de Fourier est globale : elle permet une localisation en fréquence et non pas en espace. C'est à dire elle ne permet pas de dire quelles fréquences appartiennent à quel point. Ce problème est connu sous le nom du principe d'incertitude. Ce principe montre donc que l'on ne peut pas être à la fois précis en espace et en fréquence. En fait, si un filtre est très précis en espace il l'est très peu en fréquence, et réciproquement. Pour remédier à ce problème et minimiser à la fois l'incertitude en espace et en fréquence, il est préférable d'utiliser un fenêtrage. Gabor [1-2-18] a proposé d'utiliser un fenêtrage gaussien et a démontré qu'un tel fenêtrage est optimal pour obtenir une bonne précision à la fois en fréquence et en espace. La transformation de Gabor est donc la convolution du signal par un filtre dont l'expression est la suivante :

$$G_{\sigma w_x w_y}(x,y) = e^{i(w_x x + w_y y)} \frac{1}{\sigma 2\pi} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Dans le cadre du calcul d'appariements, une égalité de phase entre deux points signifie une grande probabilité qu'il s'agisse de points à mettre en correspondance, en sachant néanmoins qu'une même valeur de phase peut apparaitre plusieurs fois pour des points différents. Pour être certain qu'une égalité de phase correspond à un appariement exact, une approche multi-échelle s'impose. Toutefois cette mesure est locale en fréquence mais directionnelle en espace. Elle résiste donc malheureusement peu à des rotations et des changements d'échelles faibles. Dans [1-2-19], Wu a alors proposé une implémentation des filtres de Gabor dans plusieurs directions et à plusieurs échelles permettant de résoudre ce problème. Etant donnée que ce filtre donne une information locale de la phase du signal, une utilisation classique est donc par exemple l'estimation de la disparité entre deux images [1-2-20 ; 1-2-21].

Il faut tout de même noter que les résultats obtenus à partir de cette approche dépendent fortement de la taille de la fenêtre employée, c'est-à-dire du support de la Gaussienne. En effet, le choix de ce paramètre est subordonné à une connaissance a priori du signal que l'on désire analyser information que l'on possède rarement lorsque le signal est une image.

Les ondelettes :

La première définition de la Transformée en Ondelettes a été établi par Grossmann et Morlet [1-2-22] en temps et échelle continus. La théorie des Ondelettes repose sur l'idée que le signal peut être caractérisé par différentes échelles et différentes résolutions. En résumé, elle consiste à remplacer le paramètre de fréquence de la Transformée de Gabor par un paramètre d'échelle, ce qui conduit à une analyse temps-échelle et non plus temps-fréquence. L'analyse par ondelettes de la fonction f est définie par :

$$W_\psi(a,b) = \frac{1}{\sqrt{a}} \int f(x)\bar{\psi}(\frac{x-b}{a})dx$$

Ou ψ est une fonction continue quelconque appelée fonction génératrice de l'ondelette, centré en b et d'échelle a.

L'inconvénient majeur de la Transformée en Ondelettes telle qu'elle vient d'être présentée est sa résolution fréquentielle est d'autant plus mauvaise que l'échelle est petite. Par exemple, un signal ayant un spectre de fréquence étroit et essentiellement localisé aux hautes fréquences ne sera pas bien représenté. Une solution à ce problème a émergé au début des années 90 avec l'apparition de la théorie des décompositions atomiques (ou par paquets) d'ondelettes. Ces méthodes ont pour caractéristique commune l'analyse des signaux suivant trois paramètres physiques : le temps, la fréquence et l'échelle. Citons notamment la méthode de Mallat [1-2-23; 1-2-24] qui a également étendu ses travaux dans le domaine de la vision par ordinateur au cas des signaux discrets.

Comme pour la Transformée de Gabor, il reste néanmoins comme inconvénient le fait que les résultats obtenus dépendent fortement de la taille du voisinage sur lequel sont effectués les calculs.

Il existe une autre transformation, la Transformée de Wigner qui permet un calcul de la fréquence en tout point :

$$W_I(x,y,\omega_x,\omega_y) = \iint R_I(x,y,\alpha,\beta)e^{-i(\alpha\omega_x+\beta\omega_y)}d\alpha d\beta$$

avec

$$R_I(x,y,\alpha,\beta) = I(x+\frac{\alpha}{2},y+\frac{\beta}{2})I^*(x-\frac{\alpha}{2},y-\frac{\beta}{2})$$

Cette transformation permet donc un calcul de la fréquence en tout point. Malheureusement, cette représentation est difficile et lourde à calculer.

Descripteur basé sur la transformation de Radon

La transformée de Radon [1-2-25] T_{Rf} d'une fonction f est définie par :

$$T_{Rf}(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos(\theta) + y\sin(\theta) - \rho)dxdy$$

avec :

- $-\infty < \rho < \infty$
- $0 \leq \theta < \pi$
- δ, la fonction de Dirac.

Intuitivement, ceci revient à intégrer la fonction f le long d'une droite pour n'importe quel(ρ,θ).

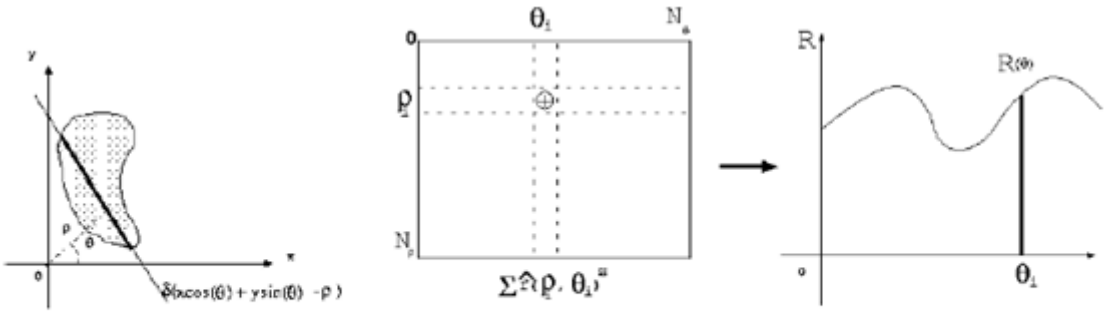


Figure 1.1Définition de la transformée de Radon et calcul de la $\mathcal{R}$-signature à partir de la matrice de Radon

Signature issue de la transformée de Radon

On peut alors introduire une signature appelée R-signature [1-2-26; 1-2-27], qui est définie à partir de la transformée de Radon:

$$\mathcal{R}_f(\theta) = \int_{-\infty}^{\infty} T_{Rf}^2(\rho,\theta)d\rho$$

Cette figure montre la transformation de Radon d'un carré, ainsi que sa R-signature.

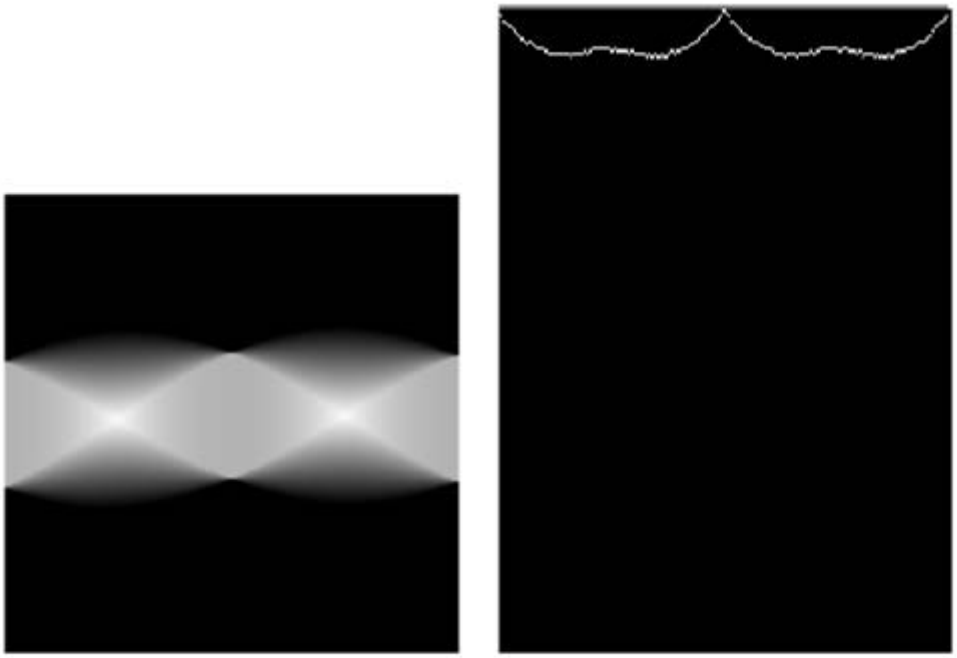


Figure 1.2 – Transformée de Radon et signature d'un carré.

A partir des propriétés de la transformée de Radon, les propriétés de Périodicité, de Rotation, de Translation et d'échelles sont vérifiées pour la R-signature [1-2-27].

Méthodes fractales

Le modèle fractal est basé essentiellement sur l'estimation par des méthodes spatiales de la dimension fractale de la surface représentant les niveaux de gris de l'image.

Le calcul de cette dimension suivant plusieurs orientations, procure une structure géométrique anisotrope. La dimension fractale et la lacunarité d'une image sont, entre autres, utilisées pour en caractériser ses textures.

Plusieurs méthodes ont été développées pour calculer la dimension fractale comme la méthode de comptage par boîte (Box Counting) et la méthode de dilatation. La dimension de Box-Counting est considérée comme la plus simple méthode pour le calcul de la dimension fractale et elle est appliquée sur des images non vides [1-2-28].

1.3.1.3 Les profils

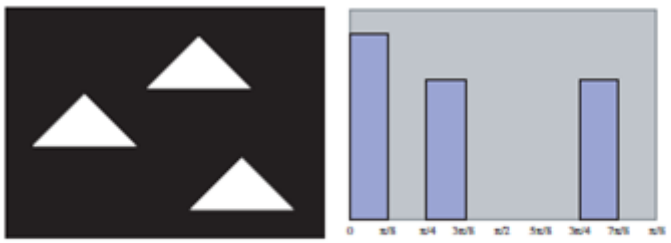
Un contour peut être assimilé à un couple de fonctions discrètes à une seule variable (x ou y) correspondant chacune à un demi-contour de la forme.

Chaque demi contour est délimité par les points extrêmes de la forme (haut / bas ou gauche / droite).

A partir de ces profils, plusieurs approches sont possibles : l'intégralité du profil, la dérivée des fonctions, la différence entre profil droit et profil gauche ou des valeurs discrètes caractéristiques comme les pics par exemple peuvent être utilisées en tant que modèle.

1.3.1.4 Approche Histogramme des Orientations des Contours (Edge Histogram Direction)

Le premier descripteur de contours exploitant l'orientation des contours fut introduit par Jain et al. [1-4-1]. Celui-ci fut développé pour la recherche d'images dans une base de donnée. Ce descripteur consiste en un simple histogramme des orientations des contours pour l'image entière. L'histogramme représente donc une discrétisation de l'espace angulaire (N secteurs angulaire) et chaque contour incrémente le secteur angulaire comportant son orientation.



(a) (b)

1.3 : image(a). (b) histogramme des orientations des contours de (a).

La position des contours n'entrant pas en compte dans l'histogramme, ce descripteur est invariant aux translations. La normalisation de l'histogramme rend la signature invariante aux changements d'échelle. Il souligne cependant que cette normalisation rend impossible la détection d'un objet occulté. Cet inconvénient vient avant tout de la nature globale (à opposer à la notion de descripteur local) de cette méthode.

Le point faible de ce descripteur concerne sa robustesse aux rotations. En effet, comme le souligne Jain et al. [1-4-1], une rotation de l'objet entraîne un décalage des angles dans l'histogramme, décalage d'autant plus complexe car lié à la discrétisation de l'espace angulaire de l'histogramme. Jain propose l'application d'un lissage sur l'histogramme afin d'autoriser de petites rotations.

Ce descripteur n'exploite pas la répartition spatiale des contours.

1.3.1.5 Pairwise Geometric Histogram

Le Pairwise Geometric Histogram (PGH) [1-5-01, 1-5-02, 1-5-03] est une approche combinant à la fois l'orientation et la position des contours. Le PGH s'appuie sur l'écart angulaire entre deux contours et la distance orthogonale entre les points de ces deux mêmes contours.

1.3.1.6 Approximation polygonale :

Les contours de la forme peuvent être représentes par des segments de droites obtenus par une approximation polygonale.

1.3.1.7 Transformée de Hough :

La transformée de Hough [1-7-01] est une méthode pour détecter les formes simples à partir d'un ensemble de points. L'application la plus classique est la détection des segments de droites. Le principe est de représenter les segments entre chaque paire de points dans un espace de coordonnées défini par l'angle et la distance du segment à l'origine et d'accumuler les nouvelles coordonnées dans une matrice. Les maxima locaux de cette matrice indiquent les segments. On peut appliquer la transformée de Hough également pour détecter les cercles. Cette fois-ci on fait la transformation vers un espace à trois coordonnées, position x, position y et le rayon.

Une version généralisée de la Transformée de Hough (GHT) fut introduite dans [1-7-02, 1-7-03], permettant la détection de formes quelconques. Cette méthode crée, lors d'une phase d'apprentissage, une description du contour sous la forme d'une *R-Table*.

La GHT présente néanmoins de nombreuses faiblesses. La représentation du contour sous la forme d'une R-Table n'est ni robuste aux rotations et encore moins aux changements d'échelles.

1.3.1.8 OrderType

L'approche de Thureson et Carlsson [1-8-01] consiste à construire un histogramme d'indices à partir des triplets de points de fort gradient. La procédure se compose de 3 étapes. D'abord, l'image est lissée par un filtre Gaussien, puis son Gradient est seuillé pour obtenir des contours. Ensuite, pour toutes les combinaisons de 3 points de contour, un indice représentant la relation d'angles est calculé. Finalement, un histogramme d'occurrence de différents indices est construit.

1.3.1.9 Les moments de contours

Les moments des contours peuvent être utilisés pour réduire les dimensions de la représentation des contours. En supposant que les contours ont été représentés en signature de forme z(i), le moment d'ordre r, m_r , et le moment centralisé μ_r peuvent être estimé par

$$m_r = \frac{1}{N}\sum_{i=1}^{N}[z(i)]^r \quad \text{and} \quad \mu_r = \frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^r,$$

Ou N est le nombre de point de contour. Les moments normalisés $\bar{m}_r = m_r/(\mu_2)^{r/2}$ et $\bar{\mu}_r = \mu_r/(\mu_2)^{r/2}$ sont invariants à la translation, rotation et changement d'échelles.

Des descripteurs moins sensibles au bruit peuvent être obtenus à partir
$F_1 = (\mu_2)^{1/2}/m_1$, $F_2 = \mu_3/(\mu_2)^{3/2}$, et $F_3 = \mu_4/(\mu_2)^2$.

La méthode dans [1-9-01] traite l'amplitude de la signature d'une fonction z(i) comme une variable aléatoire *v* et crée un histogramme $p(v_i)$ à partir de z(i). Puis, le moment d'ordre r est obtenu par :

$$\mu_r = \sum_{i=1}^{K} (v_i - m)^r \, p(v_i) \quad \text{and} \quad m = \sum_{i=1}^{K} v_1 \, p(v_i).$$

L'avantage de ces descripteurs est qu'ils sont faciles à implémenter.

1.3.1.10 Les signatures basées sur le contour

Une signature représente une forme par une fonction monodimensionnelle dérivée du contour. Il existe de nombreuses signatures [1-10-01] comme :

- le profil centroidal.
- les coordonnées complexes.
- la distance au centroıd.
- l'angle tangent.
- l'angle cumulé.
- la courbure.
- l'aire.
- la longueur de la corde.

Ces signatures sont habituellement normalisées pour être invariantes à la translation et à l'échelle. L'invariance à la rotation peut être obtenue en faisant une permutation circulaire de la signature, ce qui vient alourdir le temps de calcul pour un appariement.

L'avantage de ces signatures est globalement leur coût peu élevé en temps de calcul, mais en contrepartie, elles sont souvent assez sensibles au bruit, et le coût de l'appariement entre deux signatures peut être élevé.

1.3.1.11 Histogrammes de contraintes entre pixels

Une autre approche récente et intéressante se basant aussi sur la géométrie de la forme est celle proposée par Yang [1-9-01]. Pour chaque point P_0 de la forme, Yang considère deux autres points P_i et P_j appartenant aussi à la forme. Pour ce triplet de point il est alors possible de calculer l'angle α, et le rapport minimum des distance L_{ij}qui sont définis par :

$$a = (P_0P_i, P_0P_j)$$

$$L_{ij} = min(\frac{P_0Pi}{P_0Pj}, \frac{P_0Pj}{P_0Pi})$$

A partir de ce point de référence, on peut examiner tous les voisins possibles, et l'on pourra alors construire deux histogrammes qui rendent compte de la distribution des angles et du rapport minimum des distances que l'on normalise par le nombre de couples de points.

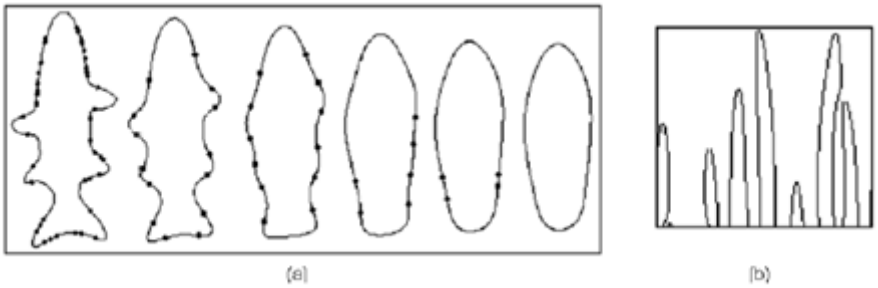
On utilise ensuite chaque point de la forme comme point de référence, ce qui nous donne un ensemble d'histogrammes. On peut alors calculer un histogramme moyen pour les angles et pour le rapport minimum des distances, que l'on utilise comme descripteur.

Cependant la complexité rend ce descripteur peu adéquat dans le cadre de grandes bases de données. Pour réduire le coût en terme de calcul, il est plus judicieux d'appliquer ce descripteur sur le squelette ou sur le contour des formes recherchées. Les deux seuls paramètres à régler sont l'échantillonnage pour l'histogramme des angles et l'histogramme du rapport minimum des distances.

1.3.1.12 Curvature Scale Space : CSS

CSS [1-12-01] décrit un contour fermé, de manière invariante à la taille, la rotation et la translation. Plus précisément, il considère les positions des points d'inflexion du contour lors d'une série de filtrages gaussiens. À mesure que la largeur du filtre augmente, les inflexions non significatives sont éliminées et le contour devient plus lisse. Il considère que les points d'inflexion restants sont représentatifs de la forme.

Les points d'inflexion du contour sont ensuite déduits des zéros de la courbure. Le résultat de ce traitement par filtrages successifs est la création d'une image d'empreinte, qui présente la liste des points d'inflexion au cours des filtrages, en fonction de leur abscisse curviligne le long du contour.



1.4 : évaluation d'un contour au gré des filtres gaussiens (a) et représentation CSS associée (b).

La mesure de similarité entre un contour issu d'une image R et un contour issu d'une image tierce I consiste en une mesure L2 entre les pics appariés, modifiée par une pénalité pour chaque pic non associé.

1.3.1.13 Les modélisations autorégressives

De la même façon que les signaux monodimensionnels sont analysés par des modélisations autorégressives unidimensionnelles (AR ou ARMA) afin de mettre en évidence les périodicités qu'ils contiennent, les signaux bidimensionnels peuvent être analysés par des modèles autorégressifs bidimensionnels. Une étude très complète de ces méthodes de description est présentée dans [1-13-01].

Dans une approche autorégressive à moyenne ajustée (ARMA), l'image est décrite par la formule :

$$f(i,j) = \sum_{(k,l)\in\mathcal{D}_a} a_{k,l} f(i-k, j-l) + b_{0,0} B(i,j) + \sum_{(k,l)\in\mathcal{D}_c} b_{k,l} B(i-k, j-l)$$

Les modèles AR permettent de représenter très bien n'importe quelle densité spectrale de puissance. Ceci se fait en utilisant les équations normales associées de Yule Walker [1-13-02]. Les représentations par modélisation autorégressive des textures est particulièrement efficace pour des textures très périodiques et de large extension.

1.3.1.14 Shape Matrix adapté au contour :

Pour ajouter la robustesse aux transformations affines, Zuliani et al. [1-14-01] exploite la matrice de covariance associée à la région. De plus, sa méthode ne caractérise plus une région mais le contour externe de celle-ci, ce qui fait des Shape Matrix (dans la partie concernant les regions) un descripteur de contour.

Chalechale propose également une extension du Shape Matrix en appliquant celui-ci aux contours [1-14-02]. Il n'obtient pas la robustesse affine mais améliore la robustesse aux rotations en utilisant une transformée de Fourrier. De plus, la Shape matrix n'est plus binaire mais pondérée par le nombre de pixel de contour par secteur, rendant ainsi la méthode moins sensible aux bruits et contours parasites.

Carmichael et al. proposent une méthode appelée Edge Probe Cascade [1-14-03] s'inspirant de la technique des Shape matrix. Ils présentent une méthode d'échantillonnage mieux adaptée à la description de contours en introduisant la densité de contours dans le voisinage du point échantillonné.

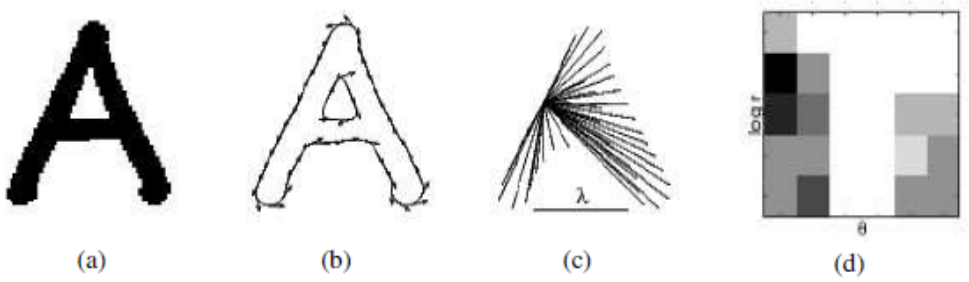
D'autre part, pour augmenter la robustesse de sa méthode, Carmichael remplace la matrice binaire ainsi que la méthode de comparaison de matrice par un arbre de probabilité [1-14-04].

1.3.1.15 Shape contexts :

La notion de *shape context* a été introduite par Belongie *et al.* [1-15-01].

Le Shape Contexte a pour principe d'échantillonner un contour en N points, et pour chacun de ces points, appliquer un shape matrix centré sur celui-ci. Belongie et al. ne caractérisent alors plus un contour par une signature mais une collection de signatures. Supposons que la forme d'un objet soit représentée par un ensemble de n points, par exemple des points échantillonnés sur le contour P = {p1; p2…….pn}, pi ∈ R2. Si l'on prend un point pi comme point de référence et l'on construit des vecteurs reliant le point pi aux autres points, ces vecteurs expriment l'apparence relative de la forme par rapport au point de référence. Lorsque n est suffisamment grand, de tels ensembles de vecteurs représentent exactement la forme entière.

Partant de cette idée, Belongie a proposé de décrire chaque point par un histogramme dans un système de coordonnées *log-polaire*. Pour être invariantes à la position et à l'échelle, toutes les distances sont normalisées par la distance moyenne de toutes les paires de points. La comparaison de deux shape contexts est une comparaison de deux histogrammes.



(a) (b) (c) (d)

1.5 : Calcule du shape context, (a) image binaire d'un objet. (b) Vecteurs de gradient sur les points de contours. (c) les vecteurs originaux d'un point aux autres. (d) le shape context du point consideré.

1.3.2 Les descripteurs régions

Pour calculer ces descripteurs, la forme est considérée comme l'ensemble des pixels qui constituent sa région interne.

1.3.2.1 Les descripteurs géométriques et topologiques simples :

– **Occupation spatiale** (smallest x (et y) coordinate, largest x (et y) coordinate): Coordonnées des coins du plus petit rectangle contenant l'objet et de côtés parallèles aux bords de l'image

– **Le taux de remplissage/d'occupation**.
– **Centre de gravité**

$$G:(x_G,y_G) \qquad x_G=\sum_{y=0}^{M-1}\sum_{x=0}^{N-1}x.f(x,y) \qquad y_G=\sum_{y=0}^{M-1}\sum_{x=0}^{N-1}y.f(x,y)$$

– **Centre géodésique**

Centre géodésique Cg : lieu du minimum de la fonction de propagation sur X

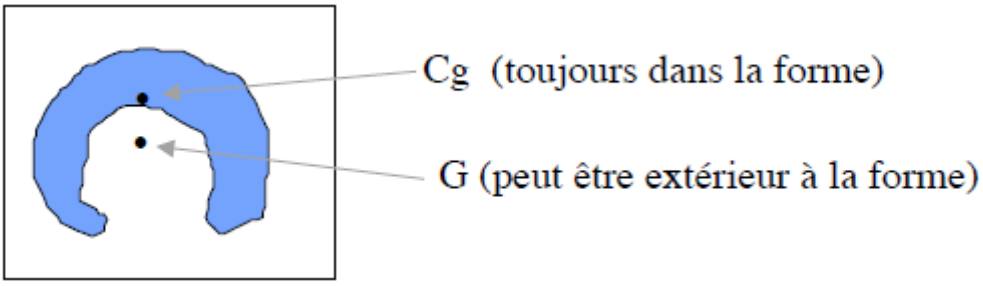


Figure 1.6 : Centre géodésique

- **Dispersion (irrégularité):**
  Est mesurée par le ratio de longueur de corde majeure à la zone.

Aire de X (nombre de pixels):

$$A(X) = \sum_{y=0}^{M-1}\sum_{x=0}^{N-1} f(x,y)\ ,\ f(x,y) = 1 \text{ si } (x,y) \in X$$

Aire des trous:

$$A_0(X) = \sum_{y=0}^{M-1}\sum_{x=0}^{N-1} f_0(x,y) \quad \text{avec} \quad f_0(x,y) = 1 \text{ si } (x,y) \text{ intérieur à } X \text{ et } f(x,y) = 0$$

- **Aire totale de X: (area size)**
  aire de X+ aire des trous
- **Périmètre P : (border size)**
- **Périmètre des trous :**
  P0
- **Périmètre extérieur :**
  périmètre – périmètre des trous
- **Largeur.**
- **Hauteur.**
- **Diagonale.**
- **Diamètre :**
  longueur du plus grand segment inclus dans la forme
- **Diamètre (longueur) géodésique :**

plus grande distance géodésique entre 2 points de X

- **Rayon géodésique :**

$$R_g(X) = \min_{x \in X} \{T_X(x)\}$$

Ou Tx(x) est la fonction de propagation

- **La compacité ou la circularité** :

$(\frac{Périmètre^2}{4*\pi*Aire})$. La forme la plus compacte est un cercle avec une compacité égale à 1. Paramètre invariant en rotation, réflexion et changement d'échelle.

- **Indice de déficit isopérimétrique** :

$$IP(X) = 1 - \frac{4\pi \cdot A(X)}{P(X)^2} = 1 - \frac{1}{F_c}$$

Il varie entre 0 (disque) et 1 (objet de surface nulle).

- **Indice d'écart au cercle inscrit :**

$$IR(X) = 1 - \frac{\pi \cdot R(X)^2}{A(X)}$$

R(X) = rayon du cercle maximal inscrit dans X.

L'indice d'écart est moins sensible au bruit que l'indice de déficit isopérimétrique (à cause du calcul du périmètre).

Indice de concavité en surface :

$$: IC_a(X) = \frac{A(X)}{A(co(X))}$$ co(X): enveloppe convexe

IC=1 si l'objet est convexe (pas d'anfractuosité).

- **Indice d'allongement :**

$$IA(X) = \frac{\pi \cdot L_g(X)^2}{4 \cdot A(X)}$$

IA = 1 pour un disque et ∞ pour un objet de surface nulle.

- **la convexité :**

$$C = \frac{\text{Périmètre enveloppe convexe}}{\text{Périmètre contour}}$$

- **La rectangularité :**

C'est le rapport de l'aire de la forme à l'aire du plus petit rectangle englobant ;

- **La complexité**

CX=Périmètre=A. il informe si la composante connexe possède une frontière "torturée" :

- **La courbure :**

Le contour est représenté par l'angle de la tangente à chaque point de contour.

On peut ensuite calculer la dérivée des courbures sur chaque point de contour ou on peut détecter les courbures importantes du contour comme descripteur.

- **l'énergie de courbure:**

$$BE = \frac{1}{P}\int_0^P |K(p)|^2 dp$$

avec K(p), la fonction de courbure, p la paramétrisation, et P la longueur totale de la courbe.

- **l'excentricité :**

$$E = \frac{\text{longueur de l'axe principal}}{\text{longueur de l'axe mineur}}$$

- **Les axes principaux** :

l'axe majeur et l'axe mineur de l'ellipse qui a les mêmes moments d'ordre deux que la forme.

- **l'orientation de l'axe principal.**
- **la variance circulaire et elliptique.**
- **le ratio (rapport) de l'axe principal.**

- **L'élongation :**

le rapport de la longueur à la largeur du plus petit rectangle englobant.

- **La caractéristique d'Euler :**

calculée pour chaque composante connexe, elle renseignera sur le nombre de trous de celle-ci.

- **Genre d'une forme g(X) :**

nombre de coupures de la forme que l'on peut faire telle que la surface obtenue soit encore connexe.

- **Nombre de branches du squelette.**
- **Nombre de points triples.**
- **Diamètres de Feret :**

mesure une taille dans une direction α donnée.

- **l'inertie parallèle à l'axe principal**
- **l'inertie orthogonale à l'axe principal**

Ces descripteurs sont en général très rapides à calculer mais sont aussi peu discriminants.

1.3.2.2 Les moments :

Hu [2-2-01], a fait le premier pas dans l'utilisation des moments invariants pour la reconnaissance des formes vers 1960 en proposant les six moments de Hu qui sont basés sur les moments géométriques. Par la suite beaucoup d'autres moments ont été proposés : les Moments de Legender[2-2-02], les Moments Zernike[2-2-03] [2-2-04] [2-2-05] [2-2-16], les Pseudo- Moments de Zernike[2-2-03], les Moments Rotationnel, les moments de Chebyshev, les moments de Bamieh[2-2-08], les moments de Schimd, les moments de figueiredo et les **Moments Complexes**. Ces moments sont devenus de plus en plus populaires grâce à leur description compacte, leurs performances et la possibilité de choisir le niveau de détail à atteindre, ce qui explique leur présence dans diverses applications [2-2-19, 2-2-17].

Un moment est une somme sur tous les pixels du modèle d'image pondéré par des polynômes liés aux positions des pixels.

Les moments décrivent l'agencement des formes (l'arrangement de ces pixels), un peu comme la combinaison de la surface, la compacité, l'irrégularité et les descripteurs d'ordre supérieur ensemble.

Toutefois, les moments fournissent une description globale avec des propriétés d'invariance et avec les avantages d'une description concise visant à éviter les effets du bruit, Comme tels, ils se sont avérés populaires et couronnés de succès dans de nombreuses applications.

Les moments cartésiens de deux dimensions sont en fait associés à un ordre qui part de zéro vers des ordres supérieurs. Le moment d'ordre p et q, m_{pq}d'une fonction I(x,y) est défini comme:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q I(x,y)\,dx\,dy$$

Pour les images discrètes, l'équation est habituellement approximée par :

$$m_{pq} = \sum_x \sum_y x^p y^q I(x,y)\Delta A$$

Le moment d'ordre zéro,m_{00}, représente la masse totale d'une fonction. Dans la définition de moments, ces valeurs sont généralement liées à la densité. Les deux moments du premier ordre, m_{01} et m_{10}, sont proportionnelles à la forme du centre de coordonné (les valeurs exigent une division par la surface de la forme) pour les images binaires. En général, le centre de la masse $(\bar{x}, \bar{y})$ peut être calculé par le rapport entre le 1ere ordre et l'ordre zéro

$$\bar{x} = \frac{m_{10}}{m_{00}} \qquad \bar{y} = \frac{m_{01}}{m_{00}}$$

Étant donné que nous avons maintenant une évaluation du centre d'une forme (en fait, un point de référence pour cette forme), les moments centralisés, μ_{pq}, qui sont invariant à la translation, peuvent être définis comme :

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I(x, y) \Delta A$$

Le moment d'ordre zéro centralisé est à nouveau la surface de la forme. Toutefois, les moments d'ordre 1 centralisé μ_{01} et μ_{10} n'ont aucune possibilité de description puisqu'ils sont les deux égal à zéro. Un des moments de second ordre, μ_{20}, a lui des possibilités descriptives.

$$\begin{aligned}
\mu_{20} &= \sum_x \sum_y (x - \bar{x})^2 I(x, y) \Delta A \\
&= \sum_x \sum_y (x^2 - 2x\bar{x} + \bar{x}^2) I(x, y) \Delta A \\
&= m_{20} - 2 m_{10} \frac{m_{10}}{m_{00}} + \left(\frac{m_{10}}{m_{00}} \right)^2 m_{00} \\
&= m_{20} - \left(\frac{m_{10}}{m_{00}} \right)^2
\end{aligned}$$

Les moments d'ordres 2 sont appelés les moments d'inertie : m_{02} , m_{11} , et m_{20} et μ_{02} , μ_{11} et μ_{20}. Ils donnent une représentation de la distribution des pixels d'un objet autour de son centre de gravité et permettent de calculer différents paramètres :

- les axes principaux de l'objet, son orientation
- l'ellipse image
- le rayon de giration

Les ensembles m_{p0} et m_{0q} peuvent être considérés comme les moments de la projection de l'image suivant l'axe des x et l'axe des y (resp.).

On peut alors considérer la projection comme une distribution de probabilités.

Les moments μ_{30} et μ_{03} mesurent le degré de déviation de la symétrie par rapport à la moyenne de la projection.

Coefficients de dissymétrie : $DS_x = {\mu_{30}}/{\mu_{20}{}^{3/2}}$, $DS_y = {\mu_{03}}/{\mu_{02}{}^{3/2}}$

Le signe indique de quel côté se fait la dissymétrie

Les moments centralisés d'ordre 4 décrivent l'aplatissement des projections.

μ_{40} , μ_{04}décrivent le caractère plus ou moins pointu de la distribution.

Coefficients d'aplatissement :

$Ap_x = {\mu_{40}}/{\mu_{20}{}^2} - 3$, $Ap_y = {\mu_{04}}/{\mu_{02}{}^2} - 3$. 0 →gaussienne, <0 →plus plat, sinon pointu

Toutefois, les moments centralisés sont jusqu'ici seulement invariants par translation. Afin d'accumuler l'invariance d'échelle, nous avons besoin des moments centraux normalisés, η_{pq}, définis dans [2-2-01].

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}$$

ou

$$\gamma = \frac{p+q}{2} + 1 \quad \forall p+q \geq 2$$

Les moments de Hu :

Hu [2-2-01] montre alors qu'on peut obtenir l'invariance à la rotation en utilisant les 7 moments suivants :

$$M1 = \eta_{20} + \eta_{02}$$
$$M2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$
$$M3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$
$$M4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
$$M5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) + ((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} - \eta_{03})^2) + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2)$$
$$M6 = (\eta_{20} - \eta_{02})((\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$
$$M7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2) + (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03})(3(\eta_{12} + \eta_{30})^2 - (\eta_{21} + \eta_{03})^2)$$

Les six premiers moments sont invariants à la translation, la rotation, l'échelle ainsi qu'aux réflexions. Le septième moment n'étant pas invariant aux réflexions, et permettant donc de différencier les images dites "miroirs".

Ces moments invariants ont les propriétés d'invariance les plus importantes. Cependant, ces moments ne sont pas orthogonaux en tant que tels il existe un potentiel de réduction de la taille de l'ensemble de moments nécessaires pour décrire une forme précise. Ceci peut être réalisé en utilisant les moments de Zernike [2-2-03] qui donnent un ensemble orthogonal de moments invariants à la rotation.

Les moments de zernike :

Les polynômes de Zernike ont été définis en 1934 dans le cadre de la théorie de la diffraction optique ([2-2-20]). Dérivés de ces polynômes, les moments ont été utilisés par de nombreux auteurs en reconnaissance de caractères ([2-2-03], [[2-2-05]], [2-2-04], [2-2-09]). Plusieurs études montrent également la supériorité de ces descriptions par rapport à d'autres approches ([2-2-18], [2-2-15]).

L'invariance par rotation est obtenue en utilisant une représentation polaire, par opposition à la paramétrisation cartésienne pour des moments centralisés. Le moment de Zernike complexe, Z_{pq}, est

$$Z_{pq} = \frac{p+1}{\pi} \int_0^{2\pi} \int_0^{\infty} V_{pq}(r, \theta)^* f(r, \theta) r dr d\theta$$

où p est maintenant l'ampleur radiale et q est la direction radiale et où * désigne le complexe conjugué d'un polynôme de Zernike, V_{pq} est donné par

$$V_{pq}(r, \theta) = R_{pq}(r) e^{jq\theta} \quad \text{where} \quad p - q \text{ is } even \text{ and } 0 \le q \le p$$

où R_{pq} est un polynôme à valeur réelle donnée par

$$R_{pq}(r) = \sum_{m=0}^{\frac{p-q}{2}} (-1)^m \frac{(p-m)!}{m! \left(\frac{p-2m+q}{2}\right)! \left(\frac{p-2m-q}{2}\right)!} r^{p-2m}$$

L'orthogonalité de ces polynômes assure la réduction de l'ensemble de nombres employés pour décrire une forme, ces moments sont peu corrélés et assurent une meilleure description de la forme.

Il est alors intéressant de noter que l'on peut exprimer ces moments en fonction des moments centrés normalisés vus plus haut. Il faut alors utiliser la forme radiale des polynômes.

Les moments de Zernike peuvent être invariants à l'échelle avec une normalisation. Un algorithme de normalisation comme "shape compacting" mentionné dans [2-2-11] peut être utilisé pour les rendre invariants en échelle et en translation.

Une fois ces moments calculés, on peut en extraire des invariants aux transformations affines. Pour ce faire, on peut citer les approches de Teague [2-2-03] et de Belkasim [2-2-09].

La description d'une forme selon les moments de Zernike revient à considérer un vecteur composé des moments de Zernike à différents ordres. Les moments d'ordre inférieur décrivent l'aspect global d'une forme, tandis que les moments d'ordre supérieur décrivent les détails de la forme.

Ces descripteurs ont montrés de bon effet dans l'application en reconstruisant une bonne approximation d'une forme avec seulement peu de descripteurs [2-2-10] et en l'identification [2-2-05]. Il y a de pseudo -moments de Zernike visant pour soulager la restriction à la normalisation au cercle d'unité, aussi bien que des moments complexes (2-2-06), visant encore une fois à donner une description plus simple des moments avec des propriétés d'invariance.

Les moments complexes et les invariants de Flusser et Suk.

Pour évaluer la qualité des moments invariants de HU, ABU-MOSTAFA ET PSALTIS ([2-2-07]) introduisent les moments complexes. L'expression discrète, non centrée, non normée des moments complexes de degré (p, q) s'exprime, pour une image de support I = (n × m), selon l'équation :

$$c_{pq} = \sum_{(x,y)\in I} (x+iy)^p (x-iy)^q f(x,y),\ \forall (p,q) \in \mathbb{N}$$

où i représente l'unité complexe. L'ordre de c_{pq} est donné par p + q, sa répétition par |p − q|.

FLUSSER ET SUK ([2-2-13]) donnent l'expression des moments complexes en fonction des moments statistiques

$$c_{pq} = \sum_{k=0}^{p} \sum_{j=0}^{q} C_k^p C_j^q (-1)^{(q-j)} . i^{(p+q-k-j)} . m_{(k+j,p+q-k-j)}$$

avec C_k^p le nombre de combinaisons de k éléments parmi p, et inversement :

$$m_{pq} = \frac{1}{2^{(p+q)} i^q} \sum_{k=0}^{p} \sum_{j=0}^{q} C_k^p C_j^q (-1)^{(q-j)} . i^{(p+q-k-j)} . c_{(k+j,p+q-k-j)}$$

Comme pour les moments statistiques, une version centrée invariante en translation des moments complexes s'obtient par :

$$\gamma_{pq} = \sum_{(x,y)\in I} (x+iy-\overline{(x+iy)})^p (x-iy-\overline{(x-iy)})^q f(x,y),\ \forall (p,q) \in \mathbb{N}$$

avec :

$$\begin{cases} \overline{(x+iy)} = \bar{x} + i\bar{y} = \frac{c_{10}}{c_{00}} = \frac{m_{10}+im_{01}}{m_{00}} \\ \overline{(x-iy)} = \bar{x} - i\bar{y} = \frac{c_{01}}{c_{00}} = \frac{m_{10}-im_{01}}{m_{00}} \end{cases}$$

De même que pour les moments statistiques, les $\left\{\frac{\gamma_{pq}}{(n+im)^p(n-im)^q}\right\}$ sont invariants par translation et changement d'échelle. En exprimant les moments complexes en coordonnées polaires, ABU-MOSTAFA ET PSALTIS ([2-2-07]) montrent facilement que les $|c_{pq}|$ sont invariants par rotation. Cependant , comme $c_{pq} = c^{\star}_{qp}$, **où** * représente le complexe conjugué, les $|c_{pq}|$ ne permettent d'obtenir qu'un ensemble de $\frac{N}{2}+1$ invariants à partir d'un ensemble original de N +1 invariants d'ordre N. Il y a donc perte d'information.

De plus, ils montrent que les attributs d'une image correspondant aux variations angulaires de N + 1 cycles/cycle et plus sont complètement perdus si les moments sélectionnés sont d'ordre au plus (N + 1) (N + 2)/2, ceci est valable également pour les moments de Zernike.

A partir des moments complexes, il est possible d'exprimer les moments de HU. Considérons la normalisation suivante :

$$\vartheta_{pq} = \frac{\gamma_{pq}}{\gamma_{00}^{(p+q+2)/2}}$$

Avec p+q >= 2 et (p,q) ∈ N. les 7 moments de Hu peuvent alors s'exprimer en fonction de ϑ_{pq} :

$$\begin{aligned} \phi_1 &= \vartheta_{11} \\ \phi_2 &= \vartheta_{20}\vartheta_{02} \\ \phi_3 &= \vartheta_{30}\vartheta_{03} \\ \phi_4 &= \vartheta_{21}\vartheta_{12} \\ \phi_5 &= Re(\vartheta_{30}\vartheta_{12}^3) \\ \phi_6 &= Re(\vartheta_{20}\vartheta_{12}^2) \\ \phi_7 &= Im(\vartheta_{30}\vartheta_{12}^3) \end{aligned}$$

Généralisant cette formulation, FLUSSER & SUK ([2-2-12], [2-2-13]) proposent une méthode pour obtenir un jeu complet de moments invariants par rotation à partir des c_{pq} .

Théorème 1 *Soit $n \geq 1$ et $(k_i, p_i, q_i) \in \mathbb{N}$ avec $i = (1, \ldots, n)$ et tels que :*

$$\sum_{i=1}^{n} k_i(p_i - q_i) = 0$$

alors tous les $\mathfrak{I}$ définis de la manière suivante :

$$\mathfrak{I} = \prod_{i=1}^{n} c_{p_i q_i}^{k_i}$$

sont invariants par rotation.

Le théorème 1 permet de construire une infinité d'invariants pour un ordre donné mais seulement quelques-uns d'entre eux sont indépendants mutuellement. Il convient d'en extraire une base, c'est-à-dire un ensemble complet de moments indépendants. Pour ce faire FLUSSER & SUK expriment et prouvent le théorème 2 :

Théorème 2 *Soit l'ensemble des moments complexes d'ordre inférieur ou égal à $r \geq 2$. Soit l'ensemble des invariants $\mathfrak{B} = \{\Phi_{pq} | \forall (p,q),\ p \geq q \wedge p + q \leq r\}$ définis de la manière suivante :*

$$\Phi_{pq} = c_{pq} c_{q_0 p_0}^{p-q}$$

avec q_0 et p_0 choisis arbitrairement tels que $q_0 + p_0 \leq r$, $p_0 - q_0 = 1$ et $c_{p_0 q_0} \neq 0$ pour les images traitées. Alors $\mathfrak{B}$ forme une base d'invariants par rotation.

En partant de la constatation que les moments de Hu ne sont clairement pas indépendants :

$$\phi_3 = \vartheta_{30}\vartheta_{03} = \frac{\vartheta_{03}\vartheta_{21}^3\vartheta_{30}\vartheta_{12}^3}{(\vartheta_{21}\vartheta_{12})^3} = \frac{\phi_5^2 + \phi_7^2}{\phi_4^3}$$

FLUSSER & SUK ([2-2-12]) explicitent, à partir du théorème 2, une base du deuxième, troisième et quatrième ordre de moments invariants par rotation. En utilisant la normalisation de HU et les moments complexes centrés, une version invariante par translation, changement d'échelle et rotation des moments de FLUSSER & SUK est définie par :

$$
\begin{aligned}
\psi_1 &= \vartheta_{11} = \phi_1, \\
\psi_2 &= \vartheta_{21}\vartheta_{12} = \phi_4, \\
\psi_3 &= Re(\vartheta_{20}\vartheta_{12}^2) = \phi_6, \\
\psi_4 &= Im(\vartheta_{20}\vartheta_{12}^2) \\
&= \eta_{11}((\eta_{30}+\eta_{12})^2 - (\eta_{03}+\eta_{21})^2) \\
&\quad -(\eta_{20}-\eta_{02})(\eta_{30}+\eta_{12})(\eta_{03}+\eta_{21}), \\
\psi_5 &= Re(\vartheta_{30}\vartheta_{12}^3) = \phi_5, \\
\psi_6 &= Im(\vartheta_{30}\vartheta_{12}^3) = \phi_7, \\
\psi_7 &= \vartheta_{22}, \\
\psi_8 &= Re(\vartheta_{31}\vartheta_{12}^2), \\
\psi_9 &= Im(\vartheta_{31}\vartheta_{12}^2), \\
\psi_{10} &= Re(\vartheta_{40}\vartheta_{12}^4), \\
\psi_{11} &= Im(\vartheta_{40}\vartheta_{12}^4),
\end{aligned}
\tag{2.42}
$$

Les moments de Bamieh et de Figueiredo

Au même titre que les moments de Hu, les moments de Bamieh font partie de la famille des moments algébriques invariants.

La principale caractéristique de ce modèle numérique est sa taille, beaucoup plus réduit que celui utilisé de Hu puisqu'il consiste en l'utilisation d'uniquement 4 valeurs invariantes.

Ces valeurs sont issues de la théorie des tenseurs, dont la combinaison conduit à l'obtention des moments invariants de Bamieh (*BMI*).

Nous exprimons ici ces *BMI* fonction des μ définis de la façon suivante :

$$M^{ijk\cdots} = \int_{-}^{+\infty}\int_{-}^{+\infty} x^i x^j x^k \ldots f(x^1, x^2)\,dx^1\,dx^2, \quad i, j, k \in \{1, 2\}$$

$$(BMI)_1 = \mu_{02}\mu_{20} - \mu_{11}^2$$

$$(BMI)_2 = (\mu_{03}\mu_{30} - \mu_{21}\mu_{12})^2 - 4(\mu_{03}\mu_{12} - \mu_{21}^2)(\mu_{21}\mu_{30} - \mu_{12}^2)$$

$$(BMI)_3 = \mu_{40}\mu_{04} - 4\mu_{31}\mu_{13} + 3\mu_{22}^2$$

$$(BMI)_4 = \mu_{40}\mu_{22}\mu_{04} - 2\mu_{31}\mu_{22}\mu_{13} - \mu_{40}\mu_{13}^2 - \mu_{04}\mu_{31}^2 - \mu_{22}^3$$

Les Moments de Profile

Un certain nombre de moments peuvent être calculés à partir des profils des régions. Les profils sont par exemple:

Pv = vertical profile de $f(x, y)$

Ph = horizontal profile de $f(x, y)$

Pd = diagonal (45 dégrée) profile de $f(x, y)$

Pe = diagonal (−45 dégrée) profile de $f(x, y)$

$$\begin{aligned} \mu_{10} &= \mu_v \\ \mu_{01} &= \mu_h \end{aligned}$$

$$\begin{aligned} \mu_{20} &= \mu_{vv} \\ \mu_{02} &= \mu_{hh} \\ \mu_{11} &= \frac{1}{2}\left(\mu_{dd} - \mu_{hh} - \mu_{vv}\right) \end{aligned}$$

Ceux-ci peuvent alors être employés pour dériver l'orientation d'objet. Cette manière de calculer l'orientation peut être beaucoup plus efficace que le calcul des moments à deux dimensions directement.

1.3.2.3 L'enveloppe convexe :

Une région R est convexe si pour chaque paire de points p1, p2 de R, le segment [p1p2] est dans R. L'enveloppe convexe est la plus petite région convexe qui contient la forme.

1.3.2.4 Les profils :

On peut obtenir une signature de l'objet en le projetant sur l'axe horizontal ou vertical et en comptant le nombre de pixels sur chaque colonne ou ligne. On utilise ces descripteurs le plus souvent pour la reconnaissance des caractères [2-4-01].

1.3.2.5 Le squelette :

Le squelette de l'objet est un ensemble de segments connexes d'épaisseur un pixel obtenu par l'amincissement d'une forme. On l'appelle également l'axe médian, l'ensemble des points qui ont la même distance minimale aux frontières de la forme pour au moins deux points de frontière distincts. On peut obtenir le squelette d'une forme par les operateurs de la morphologie mathématique ou par une transformée de distance qui associe à chaque pixel la valeur de la distance minimale à la frontière.

1.3.2.6 Approche Shape Matrix

La notion de Shape Matrix a été introduite par Goshtasby [2-4-02] en 1985 puis repris par Taza et al. [2-4-03].

Le principe est simple : on place au centre d'une région une grille radiale de taille M*N (ie. composée de N secteurs et M anneaux) et pour chaque point d'intersection rayon/cercle de cette grille, on regarde l'appartenance de celui-ci à la région. On lui associe la valeur 1 s'il appartient à la région, 0 dans le cas contraire. Ces résultats sont alors regroupés au sein d'une matrice binaire.

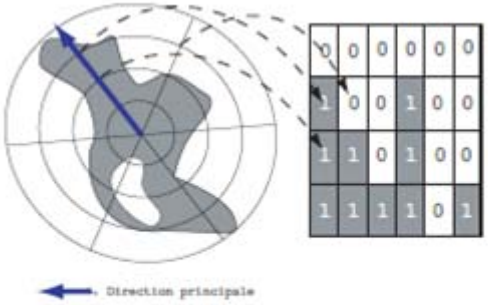


Figure 1.7 – Shape Matrix

Le fait de centrer la grille sur le centre de la région rend la méthode robuste aux translations. Pour obtenir l'invariance au changement d'échelle, le rayon du cercle le plus grand est défini comme le rayon du cercle circonscrit à la région, et le nombre de secteurs et cercles constituant la grille est fixe. Quant à l'invariance aux rotations, une direction principale doit être établie. Cette direction peut être définie par le point le plus éloigné du centre de gravité, bien que cette solution ne soit pas d'une grande robustesse. Pour ajouter la robustesse aux transformations affines, Zuliani et al. [2-4-04] exploite la matrice de covariance associée à la région.

1.3.2.7 Description de l'image par la technique ACP

La technique statistique de l'ACP (Analyse en Composantes Principales) est une technique permettant de déterminer un sous-espace optimal pour la représentation d'une quantité importante de données. Cette technique a été utilisée en vision par ordinateur pour reconnaître des visages [2-4-05], reconnaître des objets rigides [2-4-06], compresser des images vidéo [2-4-07, 2-4-08] ou estimer la position d'un robot mobile [2-4-09, 2-4-10].

Pour la reconnaissance d'objets, dans [2-4-11], Murase et Nayar ont proposé de construire une base de descripteurs en se basant sur les vecteurs propres correspondant aux plus grandes valeurs propres de la matrice de covariance d'une matrice définie par la concaténation des images d'apprentissage.

La dimension de ce sous-espace réduit significativement par rapport à celle de toutes les images. Après la construction de l'espace des images propres, les images d'apprentissage y sont projetées et les vecteurs de mesures seront stockés comme modèles.

La reconnaissance d'un nouvel objet consiste à segmenter l'objet de l'image, normaliser la taille et la luminosité, projeter dans l'espace des images propres apprises, puis comparer des vecteurs de mesures de faible dimensionnalité.

Dans un cas général, cette technique est sensible au bruit, au déplacement de l'objet, au changement d'échelle et de luminosité. Elle est plus appropriée à la reconnaissance de la pose et l'orientation de l'objet que la discriminance des classes d'objets.

1.3.2.8 Descripteur MPEG-7

Le standard multimédia MPEG7 [1] propose un ensemble de descripteurs et de composition de descripteurs permettant l'indexation normalisée des documents multimédia. Ces descripteurs contiennent non seulement des index textuels mais également des descripteurs de bas niveau calculés directement à partir des signaux audio et visuels tels que les descripteurs basés sur la texture, la couleur, les formes ou le mouvement de caméra, pour la vidéo et le rythme, la hauteur ou d'autres caractéristiques accoustiques pour le son.Ces descripteurs peuvent être utilisés dans de la cadre d'interfaces de navigation intuitives.

Une description MPEG-7 est un fichier de métadonnées détaillant différents aspects du contenu et de la gestion d'un document audiovisuel. Par exemple, en ce qui concerne le contenu, on pourra retrouver, dans le fichier de description, des informations sur les caractéristiques de bas niveau (couleurs, textures, mouvement, contenu fréquentiel, mélodie, etc.) ou des concepts sémantiques de plus haut niveau (objets, événements, interactions entre les objets, lieux de l'action, etc.).

Les outils MPEG-7 permettent de décomposer un document multimedia en différents segments spatiaux, temporels ou spatiotemporels, offrant ainsi plusieurs niveaux de description. On pourra, par exemple, diviser un extrait vidéo en séquences et associer une description à chacune d'elles, ou encore décrire différentes parties d'une seule image.

Les relations entre les divers segments peuvent également être exprimées.

1.3.2.9 Angular Radial Transform

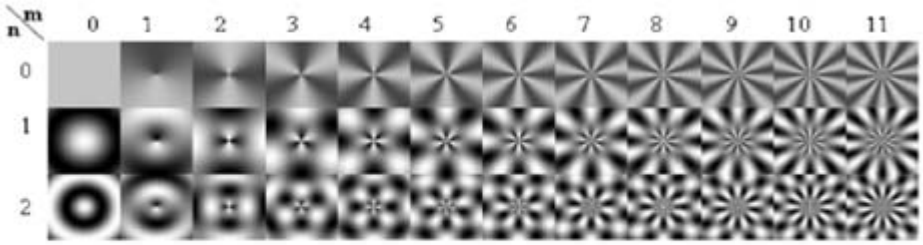


Figure 1.8

– Partie réelle des fonctions de bases de l'ART

L'ART [2-4-12] (Angular Radial Transform) est un descripteur particulièrement efficace.

En effet, il est robuste au bruit et aux changements d'échelle, invariant à la rotation, et possède une dimension raisonnable (36), ce qui lui vaut aussi d'avoir également été retenu dans le standard MPEG-7.

Ce descripteur se propose de projeter l'image sur une base orthogonale. Ainsi ses coefficients F_{nm} se calculent de la manière suivante :

$$F_{nm} = \int_0^1 \int_0^{2\pi} V_{nm}(\rho,\theta) f(\rho,\theta) d\rho d\theta$$

Ou $f(\rho,\theta)$ est l'image en coordonnée polaire, et V_{nm} est défini par :

$$V_{nm} = A_m(\theta) R_n(\theta)$$

$$A_m(\theta) = \frac{1}{2\pi} exp(jm\theta)$$

$$R_n(\theta) = \begin{cases} 1 & \text{si } n = 0 \\ 2cos(\pi n\rho) & \text{si } n \neq 0 \end{cases}$$

Le standard MPEG-7 préconise l'utilisation de n = 2 et m = 11, ce qui donne un vecteur de dimension 36. Le vecteur descripteur se déduit des coefficients F_{nm} avec de simples opérations. L'invariance à la rotation, s'obtient en prenant le module des coefficients. L'invariance à l'échelle s'obtient en divisant les coefficients par F_{00}.

1.3.2.10 Template Matching

Ces techniques font partie des méthodes d'extraction de caractéristiques relatives à un modèle. Soit f0 décrivant le modèle d'une forme et f l'application décrivant une forme inconnue dont les caractéristiques relatives sont recherchées. Si les images sont de même support, ces méthodes fournissent directement une mesure de distance ou de similarité qui peut être utilisée telle quelle dans une phase de reconnaissance.

1.3.2.11 La transformée de Karhunen-Loève

Les techniques de Template Matching décrivent une image binaire relativement à un modèle. La transformée de Karhunen-Loève ([2-4-13]), quant à elle, utilise un ensemble d'images binaires pour définir une base d'un espace vectoriel dans laquelle chaque caractère est décrit par un vecteur de coordonnées.

1.3.2.12 RSD:

Le descripteur RSD est recommandé par la norme MPEG-7 et repose sur le calcul des moments de Zernike.

Ce descripteur de forme possède de nombreuses propriétés : sa dimension est réduite ; il est robuste aux bruits et aux variations d'échelle ; il est invariant en rotation ; et permet de décrire des objets complexes. Pour ce faire, la description RSD exprime la distribution des pixels selon 36 coefficients ART (*Angular Radial Transformation*). La transformation ART est définie sur le disque unitaire par des fonctions de base sinusoïdales orthogonales.*

1.3.2.13 Radial Orthogonal Basis Functions

Similaires au descripteur MPEG-7, mais avec une composante orthogonale radiale :

$$R_n(\rho) = \begin{cases} 1 & n = 0 \\ 2\cos(2\pi n\rho) & n \neq 0 \end{cases}$$

1.3.3 Les Points d'intérêts :

Les points d'intérêt sont des régions de l'image riches en termes de contenu de l'information locale et stables sous des transformations affines et des variations d'illumination. Ils sont des indicateurs des régions susceptibles de contenir un objet, et en même temps des parties importantes de l'objet.

L'utilité des points d'intérêt a été constatée par Brady [3-1-01] qui a remarqué qu'ils imposent plus de contraintes sur les processus visuels que les contours. Selon lui, ces points fournissent des endroits de calcul fiable. De même, Dreschler et Nagel [3-1-02] ont constatés que le flot optique peut être calculé uniquement aux endroits des points d'intérêt.

On peut également citer le travail de Zhang [3-1-03]. Il a montré que l'utilisation de points d'intérêt pour le calcul de la géométrie épipolaire donne de bons résultats.

D'autre part, les points d'intérêt sont locaux. Leur calcul est effectué sur une fenêtre locale, au moins en ce qui concerne les méthodes basées sur le signal. En présence d'occultation, de telles méthodes sont donc robustes.

L'extraction de signatures visuelles locales est alors réalisée en deux passes : la détection des points d'intérêt, puis la description des régions autour de ces points.

Certains descripteurs comme SIFT comprennent à la fois un algorithme de détection de points d'intérêt et des caractéristiques d'extraction.

Les caractéristiques utilisées sont évidemment extraites dans un voisinage immédiat du point afin de conserver les propriétés qui font leur attrait (leur reproductibilité, leur robustesse, notamment à l'occlusion, …), ces caractéristiques locales recherchent principalement à renforcer l'atout des points d'intérêt en visant l'invariance aux transformations affines et aux changements d'échelle.

Parmi les caractérisations possibles, il faut aussi citer une méthode simple, mais répandue : un point est décrit par les valeurs des pixels voisins. On stocke donc les niveaux de gris directement dans un vecteur.

Les moments statistiques comme les moments de Zernik et de Hu peuvent aussi très utilisés pour décrire ces points d'intérêts.

Nous présentons quelques une des méthodes les plus utilisées pour la description des points d'intérêts.

1.3.3.1 Les invariants différentiels :

Il est à noter que ce calcul des invariants différentiels s'opère pour des images en niveau de gris.

1.3. 3.1.1 Le jet local :

Une fonction peut être approximée localement par ses dérivées. Sachant calculer les dérivées d'une fonction en un point jusqu'à un ordre N, la série de Taylor décrit cette fonction localement jusqu'à cet ordre.

De ce fait il est possible de décrire une image en un point en stockant dans un vecteur l'ensemble des dérivées en ce point. Un tel vecteur a été utilisé par Koenderink [3-1-04] qui l'a nommé jet local. Koenderink calcule en outre le jet local de manière stable en utilisant un filtre passe-bas : la gaussienne et ses dérivées.

Un jet local est calculé au voisinage d'un point et décrit la géométrie locale de ce voisinage. Pour un point donné le jet local est fonction d'un paramètre : la taille σ de la gaussienne. Ce paramètre permet de caractériser une fonction à plusieurs niveaux d'échelle ou il peut être adapté à l'échelle de l'image considérée.

Le jet local est basé sur la dérivation du signal. Toutefois, comme on ne connait pas la fonction du signal de manière analytique, les dérivées doivent être estimées de façon

numérique : elles sont calculées par convolution avec une gaussienne et ses dérivées. On peut donc interpréter le jet local comme la projection - la décomposition - du signal sur une base.

Le principal inconvénient est que le vecteur n'est pas invariant aux diverses transformations de l'image. Plusieurs méthodes permettent de corriger ou au moins d'améliorer ce problème : les filtres directionnels et les invariants de Hilbert.

1.3.3.1.2 Les filtres directionnels :

Le jet local peut être interprété comme la projection du signal sur la base constituée de la gaussienne et de ses dérivées. [3-1-05] ont montré qu'à partir de cette base, il est possible de calculer ce qu'ils appellent des filtres directionnels, c'est-à-dire des filtres définis par dérivées calculées dans n'importe quelle direction. Il est donc possible de recalculer le jet local en considérant des directions différentes, afin de tenir compte d'une éventuelle rotation appliquée à l'image. La direction du gradient, quand elle n'est pas nulle, est une solution possible puisqu'elle traduit l'orientation de l'image.

Le principal inconvénient de cette méthode est que le résultat dépend fortement du calcul de l'angle (la direction), qui représente une importante source d'instabilité. Pour plus de précisions, consulter [3-1-06] qui utilisent cette méthode et calculent ainsi un jet local ajustable invariant en rotation.

1.3.3.1.3 Les invariants de Hilbert :

Hilbert -1893- a montré que n'importe quel invariant au groupe des déplacements SO(2) d'ordre fini pouvait s'exprimer comme un polynôme d'invariants irréductibles reposant sur la combinaison de dérivés du jet local. [3-1-04; 3-1-07; 3-1-08; 3-1-09] ont repris cette idée et ont proposé de caractériser un point par un vecteur contenant un ensemble complet de ces invariants. Si l'on considère une image scalaire, ces invariants représentent l'ensemble de base des primitives qui permettent de décrire toutes les propriétés locales intrinsèques de l'image. Cet ensemble est bien connu pour ses propriétés du premier et du second ordre et est rendu indépendant de la rotation image s'il est exprimé en coordonnées de Gauge.

1.3.3.2 Descriptions fréquentielles

On peut également caractériser une fonction par une description fréquentielle. Un exemple de description fréquentielle globale est la transformée de Fourier.

Cependant la transformation de Fourier est globale : elle permet une localisation en fréquence et non pas en espace. C'est-à-dire, elle ne permet pas de dire quelles fréquences appartiennent à quel point. Ce principe montre donc que l'on ne peut pas être à la fois précis en espace et en fréquence. En fait, si un filtre est très précis en espace il l'est très peu en fréquence, et réciproquement. Pour remédier à ce problème et minimiser à la fois l'incertitude en espace et en fréquence, il est préférable d'utiliser un fenêtrage. Gabor [3-1-10] a proposé d'utiliser un fenêtrage gaussien et a démontré qu'un tel fenêtrage est optimal pour obtenir une bonne précision à la fois en fréquence et en espace. La transformation de Gabor est donc la convolution du signal par un filtre.

La transformée de Gabor permet d'adapter l'analyse fréquentielle à la rapidité des changements de l'image, donc aux fréquences de l'image. Cependant, il faut pour ce faire déterminer la taille de la fenêtre à utiliser. Celle-ci correspond à la résolution que l'on veut obtenir. Ce choix fixe complètement la dispersion en fréquence de la description obtenue. Il est donc préférable d'effectuer une décomposition multi-échelle afin d'obtenir une description riche du signal.

Morlet , Grossmann et Meyer [3-1-11, 3-1-12] ont construit une théorie reposant sur l'idée de caractériser un signal par différentes échelles et différentes résolutions : cette théorie est à l'origine des ondelettes. L'idée principale des ondelettes est que l'apparition de hautes fréquences est de faible durée en espace. Cette idée se justifie par l'hypothèse que les hautes fréquences correspondent à des discontinuités du signal et doivent donc être considérées uniquement de manière locale. Mallat [3-1-13] a étendu ces travaux dans le domaine de la vision par ordinateur au cas des signaux discrets.

Les ondelettes nécessitent une répartition logarithmique en espace et en fréquence.

Les résultats obtenus par la transformation de Gabor ainsi que par les ondelettes dépendent de la taille du voisinage sur lequel sont effectués les calculs. Ceci n'est pas le cas pour la transformation de Wigner. Sa formulation est la suivante (cf. [3-1-14]) :

$$W_I(x, y, \omega_x, \omega_y) = \iint R_I(x, y, \alpha, \beta) e^{-i(\alpha\omega_x + \beta\omega_y)} d\alpha d\beta$$

avec

$$R_I(x, y, \alpha, \beta) = I(x + \frac{\alpha}{2}, y + \frac{\beta}{2}) I^*(x - \frac{\alpha}{2}, y - \frac{\beta}{2})$$

Cette transformation permet donc un calcul de la fréquence en tout point. Malheureusement, cette représentation est difficile et lourde à calculer.

1.3.3.3 Descripteur SIFT

Dans l'article [3-1-15], Lowe décrit la méthode SIFT (*Scale-Invariant Feature Transform*) pour la description des singularités locales. L'optique est de calculer une représentation locale stable invariante aux transformations affines, aux variations d'illumination et aux changements d'échelle. Ici, nous présentons la partie description du SIFT et non la partie détection de points d'intérêt, qui opère par analyse des extrema locaux des différences de gaussienne. Le descripteur SIFT représente une région par la distribution spatiale des magnitudes des gradients. Par ce procédé, la première étape consiste à calculer la magnitude et l'orientation des gradients autour des points saillants :

$$\begin{cases} m(x,y)^2 &= (I_j(x+1,y)-I_j(x-1,y))^2+(I_j(x,y+1)-I_j(x,y-1))^2 \\ \theta(x,y) &= \tan^{-1}\left(\frac{I_j(x,y+1)-I_j(x,y-1)}{I_j(x+1,y)-I_j(x-1,y)}\right) \end{cases}$$

Afin d'accentuer l'importance des gradients proches du point saillant, une pondération gaussienne est effectuée pour finalement obtenir un histogramme des gradients orientés. Les paramètres généralement utilisés permettent de décomposer chaque région d'intérêt en 16 sous-régions et de former 4×4 histogrammes de 8 orientations pour obtenir une signature finale de 128. La pertinence et l'efficacité de ce descripteur ont permis de nombreuses extensions, dont les descripteurs PCA-SIFT (*Principal Component Analysis SIFT*.

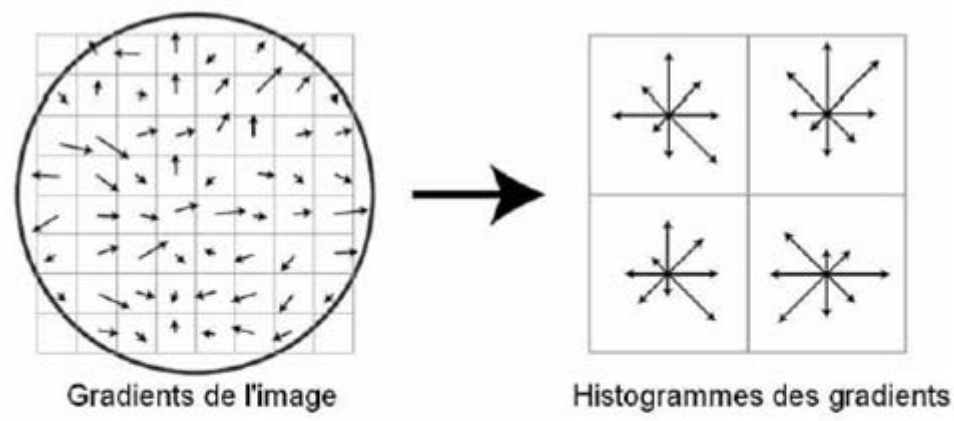


Figure 1.9 – Principe du descripteur SIFT

L'algorithme SIFT est très efficace cependant, le temps de calcul ainsi que l'espace mémoire nécessaire à son exécution sont assez importants.

1.3.3.4 Les blobs

Les blobs sont des régions de l'image qui sont plus claires ou plus sombres que leurs environs.

On peut les considérer comme des points d'intérêt qui signalent les objets ou les composants des objets. Le détecteur le plus fréquemment utilisé est le filtre (Laplacien de Gaussienne) LoG qui donne des valeurs positives élevées pour les blobs sombres et des valeurs négatives élevées pour les blobs claires.

Pourtant la réponse de l'operateur dépend de l'échelle du LoG. Lindeberg [3-1-16] a propose d'utiliser un operateur Laplacien normalisé sur un espace d'échelle et de trouver les maxima locaux dans l'espace et dans l'échelle simultanément. De cette façon on détecte les blobs à l'échelle correspondante.

1.3.3.5 Maximally Stable Extremum Regions (MSER)

La MSER proposée par [3-1-17] est une technique pour détecter les blobs dans une image. On obtient une série d'images noires et blanches en appliquant un seuillage pour chaque niveau de gris dans l'image. Les MSER sont les blobs qui restent stables sur plusieurs images consécutives.

1.3.3.6 Les descripteurs RIFT

Le descripteur RIFT [3-1-18] reprend les principes du descripteur SIFT avec quelques modifications afin de le rendre invariant en rotation. Le principe général reste le même : on décompose l'espace en régions dans lesquelles on calcule un histogramme d'orientations. La différence est qu'on utilise là des régions concentriques et que les orientations du gradient en un point sont calculées par rapport à la direction du centre en ce point. Les auteurs proposent 4 cercles et 8 bins d'orientation pour un vecteur de taille 32 (extraite de [3-1-18]).

1.3.3.7 Descripteur SURF

Le descripteur SURF (*SpeedUp Robust Features* [3-1-19]) présente des résultats convaincants en termes d'appariement de caractéristiques visuelles. Ce descripteur a la propriété d'être invariant en rotation et en changement d'échelle. La motivation principale est d'offrir une description locale peu coûteuse en temps de calcul. La détection des points d'intérêt est basée sur le calcul du déterminant de la matrice hessienne $H(x,\sigma)$. Les auteurs proposent une approximation de la convolution $L..(x,\sigma)$ de l'image par la dérivée gaussienne d'ordre deux, par des masques de convolution. Des filtres 9×9 sont appliqués pour l'estimation à l'échelle σ = 1,2 des dérivées gaussiennes Dxx ,$Dy\ y$,$Dx\ y$. La localisation des points se fait alors par la recherche de maxima, dans le déterminant approximé de la matrice hessienne, dans l'espace image et échelle avec la méthode proposée par Brown *et al.* [3-1-20].

Pour construire la signature locale, les auteurs s'intéressent à l'orientation principale contenue dans des fenêtres rectangulaires centrées sur les points précédemment déterminés. Ensuite les régions d'intérêts sous forme rectangulaire sont découpées en block 4×4, et sur ces sous régions sont calculées des caractéristiques simples sous la forme d'un vecteur v défini par :

$$v = (\Sigma d_x, \Sigma d_y, \Sigma |d_x|, \Sigma |d_y|)$$

dx et dy sont les réponses d'une analyse par ondelettes de Haar dans la direction horizontale et verticale. La signature locale est de dimension 4×4×4 = 64.

1.3.38 Descripteur RFD

La plupart des descripteurs caractérisent le voisinage local des points saillants par les contours présents dans cette région. Dans de récentes études [3-1-21], le contour est considéré comme une singularité décrite par les coefficients de Hölder.

Définition 1. *La fonction* $f:[a,b]\to\Re$ *est Hölder-*α $(\alpha\geq 0)$ *à* x_0 *si* $\exists K>0,\delta>0$ *et un polynôme* P *de degré* $m=\lfloor\alpha\rfloor:\forall x, x_0-\delta\leq x\leq x_0+\delta, \left|f(x)-P(x-x_0)\right|\leq K|x-x_0|^{\alpha}$.

Théorème 1. *L'exposant de Hölder* $h_f(x_0)$ *de* f *à* x_0 *est définie par :*
$h_f(x_0)=\sup\{\alpha, f$ *Hölder-*α *à* $x_0\}$.

La régularité locale d'une fonction au point *x*0 est ainsi la mesure de *hf* (*x*0). Plus cette valeur est faible, plus le signal est considéré comme singulier. Trois valeurs de l'exposant sont caractéristiques : *hf* (*x*0) = 0,9, *hf* (*x*0) = 0 et *hf* (*x*0) = −0,9. Ces trois valeurs décrivent respectivement une fonction triangle, une fonction échelon et une dirac. Pour une image, l'exposant de Hölder est calculé dans la direction de la régularité minimale de la singularité. Pour décrire une région d'intérêt, à la fois l'orientation et la régularité des singularités sont utilisées. L'orientation $\theta(x, y)$ et la magnitude$m(x, y)$ sont calculées pour chaque pixel comme pour le descripteur SIFT.

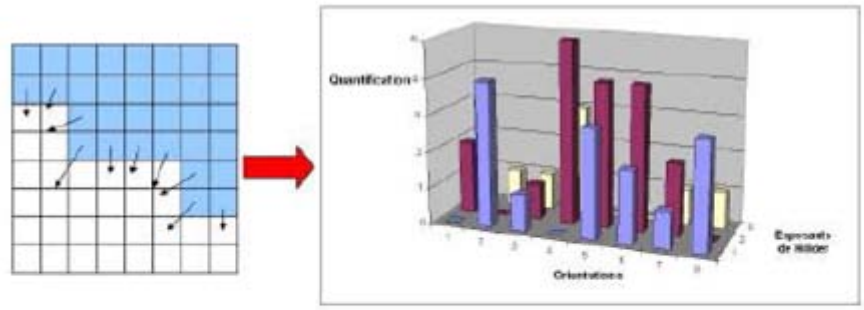


Figure 1.10

- Orientations des gradients et exposants de Hölder

L'exposant de Hölder est estimé selon des ondelettes fovéales comme dans l'étude [3-1-22].

Les orientations et les exposants de Hölder sont utilisés conjointement pour construire des histogrammes 3*D*. Chaque région d'intérêt est partitionnée en bloc 4×4. Pour chaque sous-bloc, sont calculés l'orientation du gradient et l'exposant de Hölder en chaque pixel. Ces valeurs incrémentent un histogramme par sous-bloc, comptabilisant le nombre de fois où chaque couple des valeurs est apparu. La signature locale est obtenue par la concaténation des histogrammes 3*D*. Sa taille est $n\times r\times o$ avec *n* le nombre de sous-régions, *r* le nombre d'exposants de Hölder compris entre [−1.5,1.5] et *o* le nombre d'orientation appartenant à l'intervalle $[-\pi/2,\pi/2]$. Classiquement, les auteurs utilisent 4×4 blocs et calculent 8 orientations et 3 quantifications des exposants de Hölder pour obtenir une signature locale de dimension 384.

1.3.4 La texture :

Certaines approches ne regardent plus une zone de texte comme un assemblage de lettres mais plutôt comme une zone dense de traits. Elles vont assimiler la zone de texte à une texture comportant des motifs plus ou moins réguliers et font donc une étude de la texture pour pouvoir séparer et détecter les parties textuelles.

Une texture se caractérise par la répétition d'un motif ou de quelques éléments.

La texture se manifeste donc par une information visuelle qui permet de la décrire qualitativement à l'aide des adjectifs suivants: grossière, fine, lisse, tachetée, granuleuse, marbrée, régulière ou irrégulière. Elle est d'ailleurs l'un des indicateurs courants utilisés pour l'indexation et la récupération d'images.

Il existe plusieurs méthodes et modèles proposés dans la littérature [4-1-01] pour la caractériser. Parmi ces caractéristiques, on peut noter l'orientation, la répétition ou périodicité (c.-à-d. la fréquence) et la granularité des formes qui, dans leur ensemble, définissent la texture. En fait, il s'agit des interactions entre chaque pixel et son voisinage car finalement la texture est caractérisée par des variations de luminosité dans un voisinage et par la symétrie. C'est ainsi qu'une texture est plutôt associée aux hautes fréquences du spectre d'une image. Afin de représenter numériquement la texture, il faut donc faire des mesures sur chacun des pixels en prenant en compte son voisinage et grouper ces mesures dans un vecteur. Ces mesures correspondent à l'extraction des caractéristiques de texture.

1.3.4.1 Descripteurs statistiques

1.3.4.1.1 Caractéristiques à partir de l'histogramme :

On peut calculer des mesures à partir des histogrammes en utilisant les statistiques du premier ordre.

Ces mesures ne prennent en compte qu'un pixel à la fois. Elles se déduisent de la probabilité empirique *p(n)* du niveau de gris *n* de l'histogramme et sont :

– les moments d'ordre *k* (non centrés) : $\mu_k = \Sigma_n n^k p(n),$

– les moments centrés d'ordre *k* : $\tilde{\mu_k} = \Sigma_n (n - \mu_1)^k p(n),$ et en particulier :

1. la moyenne : $\mu_1,$
2. la variance : $\sigma^2 = \tilde{\mu_2},$
3. le biais : $\gamma_1 = \frac{\tilde{\mu_3}}{\sigma^3},$
4. l'aplatissement (ou kurtosis) : $\gamma_2 = \frac{\tilde{\mu_4}}{\sigma^4} - 3.$

– l’´energie : $W = \Sigma_n |p(n)|^2,$

– l’entropie : $E = -\Sigma_n p(n) \log p(n),$

– le contraste : $C = \frac{\max(n) - \min(n)}{\max(n) + \min(n)},$

– la dynamique : $D = \max(n) - \min(n),$

– le coefficient de variation (surtout pour les images cohérentes dont le bruit est multiplicatif : radar, images ultra-sonores) : $v = \frac{\mu}{\sigma},$

– l’exposant de Holder, caractéristique de la dimension fractale : [4-1-05].

1.3.4.1.2 Les matrices de cooccurrences.

En 1973, Haralick [4-1-08] a proposé une méthode en se basant sur les matrices de cooccurrences de niveaux de gris. Elle est probablement une des méthodes la plus célèbre pour analyser la texture.

L’idée de cette méthode est d’identifier les répétitions de niveaux de gris selon une distance et une direction. Chaque résultat met en valeur les relations entre deux pixels. Les matrices de cooccurrences représentent donc les dépendances spatiales des niveaux de gris.

On considère généralement des matrices de cooccurrences pour un ensemble de voisinage.

L'interprétation des matrices de cooccurrences demande d'en extraire des paramètres. Ceux-ci sont nombreux et calculés directement sur les matrices.

Haralick a proposé 14 caractéristiques statistiques extraites à partir de cette matrice.

Actuellement, seulement les quatre caractéristiques les plus appropriées sont largement utilisées : l’énergie, l’entropie, le contraste et le moment inverse de différence.

notations

$m_x(i)$: somme de la ligne i $m_{x+y}(k)$: somme de la k ième diagonale secondaire

$m_y(i)$: somme de la colonne i $m_{x-y}(k)$: somme de la k ième diagonale principale

moment angulaire d'ordre 2 $f_1 = \sum_i \sum_j m(i,j)^2$

contraste $f_2 = \sum_{n=0}^{N} n^2 \cdot m_{x-y}(n)$

corrélation $f_3 = \frac{\sum_i \sum_i \left(i \cdot j \cdot m(i,j) - \mu_x \mu_y\right)}{\sigma_x \sigma_y}$

variance $f_4 = \sum_i \sum_i (i-j)^2 \cdot m(i,j)$

moment des différences inverses $f_5 = \sum_i \sum_i \frac{1}{1+(i-j)^2} \cdot m(i,j)$

moyenne des sommes $f_6 = \sum_{k=2}^{2N-1} k \cdot m_{x+y}(k)$

variance des sommes $f_7 = \sum_{k=2}^{2N-1} (k - f_6)^2 \cdot m_{x+y}(k)$

entropie de la somme $f_8 = -\sum_{k=2}^{2N-1} m_{x+y}(k) \cdot \log\left(m_{x+y}(k)\right)$

entropie $f_9 = -\sum_i \sum_j m(i,j) \cdot \log\left(m(i,j)\right)$

variance des différences $f_{10} = \sum_{k=0}^{N} \left(k - \mu_{x-y}\right)^2 \cdot m_{x-y}(k)$

avec $\mu_{x-y} = \frac{1}{N} \sum_{k=0}^{N} m_{x-y}(k)$

entropie des différences : $f_{11} = -\sum_{i=0}^{N} m_{x-y}(i)\log\left(m_{x-y}(i)\right)$

corrélation de l'information : $f_{12} = \dfrac{H_{XY} - H1_{XY}}{\max\left(H_X, H_Y\right)}$

$$f_{13} = \left[1 - \exp\left(-2.0\left(H2_{XY} - H_{XY}\right)\right)\right]^{\frac{1}{2}}$$

avec $H_{XY} = f_9$ $\quad H_X = -\sum_{k=0}^{N} m_x(k)\log\left(m_x(k)\right)$ $\quad H_Y = -\sum_{k=0}^{N} m_y(k)\log\left(m_y(k)\right)$

$$H1_{XY} = -\sum_i \sum_j m(i,j)\log\left(m_x(i)m_y(j)\right) \quad H2_{XY} = -\sum_i \sum_j m_x(i)m_y(j)\log\left(m_x(i)m_y(j)\right)$$

corrélation maximum : $f_{14} = \left(2ème \text{ des plus grandes valeurs propres de } Q\right)^{\frac{1}{2}}$

$$Q = \left\{Q(i,j)\right\} \begin{cases} i = 0 \cdots N \\ j = 0 \cdots N \end{cases} \qquad Q(i,j) = \sum_{k=0}^{N} \frac{m(i,k)m(k,j)}{m_x(i)m_y(j)}$$

La méthode des matrices de cooccurrences, bien que donnant de bons résultats d'analyse est peu adaptée à une étude détaillée de la texture car elle demande beaucoup de calculs.

Une alternative aux matrices de cooccurrences en plus rapide existe. Ce sont les matrices de caractéristiques statistiques [4-1-09]. La première étape de cette méthode est de calculer la différence entre l'image étudiée et sa translatée. On en ressort ensuite directement le contraste, la covariance et la dissimilarité de l'image ou d'une région de l'image.

1.3.4.1.3 Les caractéristiques d'autocorrélation

Les matrices d'autocorrélations permettent de mettre en évidence des périodicités dans une image et traduisent de ce fait les orientations principales de certaines textures, [4-1-10].

Leur principe de base est le suivant : comparer l'image originale avec une image décalée. Par rapport à la matrice de cooccurrence où on considérait un déplacement avec la fonction p, on va ici considérer des déplacements selon chaque axe. Etant donné cet ensemble de déplacements envisagés, la fonction d'autocorrélation centrée normée est définie de manière discrète et en dimension 2 par :

$$f_{MN}(\Delta x, \Delta y) = \sum_{x=1}^{M-\Delta x} \sum_{y=1}^{N-\Delta y} \left(\frac{I(x,y) \cdot I(x+\Delta x, y+\Delta y)}{(M-\Delta x)(N-\Delta y)} \right)$$

Pour peu que le voisinage exploré ne soit pas trop important, ces caractéristiques présentent l'avantage d'être compactes et rapidement calculables.

Pour caractériser les résultats, il est par exemple possible de calculer les moments du deuxième ordre.

L'autocorrélation permet de caractériser, de manière unique, l'anisotropie de grandes zones de texte.

1.3.4.1.4 La Rose de Directions :

Le calcul de la rose des directions de l'image a été proposé par Bres dans [4-1-13]. Il est basé sur le calcul de l'espérance mathématique déterminée à partir de la fonction d'autocorrélation.

La rose des directions est un diagramme polaire se basant sur l'étude de la réponse de la fonction d'autocorrélation lorsqu'elle est appliquée sur une image. Cette fonction a déjà été utilisée dans [4-1-14, 4-1-15] afin de caractériser des textures naturelles.

Elle indique de manière très précise les orientations présentes. Elle a en plus la particularité d'être peu sensible aux bruits de type tache d'encre.

A partir de la rose des directions, on peut calculer par exemple la valeur de l'orientation principale.

Du fait de sa définition mathématique issue de la fonction d'autocorrélation, la rose des directions permet de caractériser l'anisotropie ou de l'isotropie de la texture étudiée.

1.3.4.1.5 Longueur de plages

On appelle une plage un ensemble de pixels consécutifs et dans une direction donnée ayant le même niveau de gris. La longueur d'une plage est alors le nombre de pixels pour une plage donnée.

L'objet qui caractérise les textures dans cette méthode est alors un ensemble de matrices $P_\theta = (p_\theta(i,j))$. Chaque case (i , j)de la matrice P_θ contient le nombre de plages de longueur j et de niveau de gris i dans la direction θ. .

Les longueurs de plages peuvent être étudiées sur toute l'image ou bien sur des régions précises. Il est pour cela nécessaire d'en extraire quelques attributs :

Soit L le nombre de niveaux de gris dans l'image et n_θ la longueur de corde maximale dans la région étudiée et dans la direction θ.

-nombre de longueurs de plage :

$$SLP = \sum_{i=0}^{L-1} \sum_{j=1}^{n_\theta} p_\theta(i,j)$$

-proportion de petites plages :

$$RF_1 = \frac{1}{SLP} \sum_{i=0}^{L-1} \sum_{j=1}^{n_\theta} \frac{p_\theta(i,j)}{j^2}$$

-proportion de grandes plages :

$$RF_2 = \frac{1}{SLP} \sum_{i=0}^{L-1} \sum_{j=1}^{n_\theta} j^2 p_\theta(i,j)$$

-hétérogénéité des niveaux de gris :

$$RF_3 = \frac{1}{SLP} \sum_{i=0}^{L-1} (\sum_{j=1}^{n_\theta} p_\theta(i,j))^2$$

- hétérogénéité des longueurs de plage :

$$RF_4 = \frac{1}{SLP} \sum_{j=1}^{n_\theta} (\sum_{i=0}^{L-1} p_\theta(i,j))^2$$

-pourcentage de plages :

$$RF_5 = \frac{SLP}{K}$$

avec K le nombre total de pixels dans la région.

Tout comme pour les matrices de cooccurrences, cette méthode permet une bonne caractérisation des textures des régions. Cependant, elle reste relativement lente et est efficace sur des images comportant peu de niveaux de gris.

1.3.4.1.6 Histogramme de champs réceptifs

Dans [4-1-16, 4-1-17, 4-1-18], Schiele et al. ont proposé d'utiliser l'histogramme de champs réceptifs.

Cette représentation comme toutes les méthodes basées sur la technique de l'histogramme n'a pas besoin de segmentation ni de modélisation géométrique de l'objet. Les champs réceptifs sont en fait la première dérivée de Gaussienne, la magnitude et direction de la première dérivée et le Laplacien. L'effet de variation de l'intensité du signal est éliminé par la normalisation des réponses des filtres de champs réceptifs par l'énergie du signal.

1.3.4.1.7 Moments d'histogramme des luminances

L'histogramme normalisé est une estimation de la densité de probabilité des niveaux de gris dans l'image. A partir de cet histogramme, on peut définir les moments centrés d'ordre n.

- $\mu_n = \sum_{i=1}^{k} (a_i - m)^n p(a_i)$

avec : k : nombre de niveaux de gris dans l'image

a_i : niveau de gris i

$p(a_i)$: n_i / N^2 où n_i est le nombre de pixels de niveau a_i et N^2 le nombre de pixels de l'image.

$m = \sum_{i=1}^{k} a_i p(a_i)$ est la moyenne : donne l'apparence de l'image.

$\mu_2 = \sum_{i=1}^{k} (a_i - m)^2 p(a_i) = \sigma^2$: variance (carré de l'écart-type)

- faible pour les textures grossières (histogramme centré autour de m)
- grand pour les textures fines (histogramme étalé)

Le problème majeur de l'utilisation de l'histogramme des niveaux de gris pour l'étude de la texture réside dans le fait que la répartition spatiale des niveaux de gris n'est pas prise en compte.

1.3.4.1.8 Méthode de Tamura pour mesurer le contraste

La construction d'une matrice de cooccurrence pour obtenir une estimation du contraste peut se révéler coûteuse en temps de calcul.

Tamura et al [4-1-27] affirment que 4 facteurs sont supposés influencer les différences de contraste entre deux textures : la gamme dynamique des niveaux de gris, la polarisation de la distribution de noir et de blanc dans l'histogramme des niveaux de gris, l'acuité des frontières et la période de répétition des motifs. Cette méthode permet d'avoir une estimation précise et rapide.

Dans leur article, ils proposent une approximation du contraste à l'aide d'une mesure incluant les deux premiers facteurs. Pour obtenir une mesure de la polarisation, ils utilisent le kurtosis α_4 qui mesure la disposition des masses de probabilité autour de leur centre.

$$\alpha_4 = \frac{\mu_4}{\sigma^4}$$

Avec μ_4 le moment centré d'ordre 4 et σ^2 la variance.

Pour prendre en considération la gamme dynamique des niveaux de gris, ils combinent le kurtosis avec l'écart type de la façon suivante :

$$Contrast = \frac{\sigma}{\alpha_4^n}$$

avec n valeur positive. Dans leur article, Tamura et al présentent un ensemble de comparaisons entre des expérimentations psychologiques et leurs opérateurs. Ils concluent que la valeur n = 1=4 donne la meilleur approximation.

1.3.4.1.9 Dérivés gaussiennes orientées

Descripteur de texture basé sur les dérivés gaussien orientée (OGD) pour générer des vecteurs de caractéristique invariant à la rotation.

L'idée de base est de calculer l '«énergie» d'une région (un emplacement pour des caractéristiques locales et l'ensemble de l'image pour les fonctionnalités globales) en fonction orientable.

Cette énergie est calculée pour différentes «puissances» des canaux, qui sont le résultat de la convolution des images avec des filtres OGD d'un ordre précis.

Dans un certain sens, le premier calcule l'énergie de "bord", tandis que le second ordre calcule l'énergie de «ligne».

1.3.4.2 Extraction de textures par filtrage préalable

Une idée apparue assez rapidement dans le domaine de la texture est d'utiliser le domaine fréquentiel pour tenter de la caractériser. Ceci vient tout naturellement de l'aspect de périodicité qui pousse à étudier les fréquences spatiales pour retrouver une trace du motif. La transformée de Fourier fut le premier outil à émerger dans ce but. Comme on peut le remarquer dans un de ces travaux initiaux [4-1-19], un problème qui se pose est l'absence de localisation spatiale, ce qui impose de diviser l'espace au moyen d'une grille régulière (une fenêtre carrée passant sur chaque pixel de l'image étant jugée comme trop coûteuse). Si ce problème rend délicate la détection de texture sur une photo, l'étude de la transformée de Fourier d'une région de texture apporte néanmoins des informations particulièrement intéressantes. Ainsi Bajcy et Lieberman [4-1-20] ont calculé des spectres de puissance dans différentes fenêtres sur l'image. L'étude de la forme du spectre de puissance suivant le rayon et la direction permet de caractériser la texture par son orientation et sa périodicité. D'autres travaux (comme [4-1-21]) ont permis de dégager des propriétés de la texture telles que la régularité, la "directionnalité", la linéarité ou la finesse par l'étude de portions spécifiques du spectre.

On peut aussi utiliser un filtrage qui, en plus d'être sélectif selon une orientation, caractérise la périodicité et permet donc d'extraire localement des informations de même nature que celles qu'on pourrait extraire au moyen d'une transformée de Fourier. Ce principe est devenu

particulièrement populaire avec l'apparition des filtres par ondelettes et plus particulièrement des filtres de Gabor qui correspondent au produit d'une fonction gaussienne et d'une fonction cosinus. L'image est donc filtrée avec un jeu de fonctions qui vont isoler différentes orientations et différentes fréquences spatiales et une région de texture peut être par exemple caractérisée par la moyenne et la variance des coefficients de transformation.

Dans [4-1-22], les caractéristiques extraites sont le spectre de puissance de Fourier (mesure équivalente à l'énergie du spectre) pour Fourier, les magnitudes de réponses pour le filtre de Gabor et les variances (mesure type énergie) des sous-bandes issues de la transformée en ondelettes.

Dans [4-1-23], les caractéristiques extraites sont : l'énergie, la magnitude, le carré de sorties et la rectification, dont celle basée sur une fonction sigmoïde.

On peut aussi citer d'autre filtres : les cosinus locaux [4-1-24], les brushlets [4-1-25], les WaveAtoms [4-1-26], les filtres de Laws, les filtres en anneau et en wedge, la transformée cosinus discrète (DCT), les filtres miroirs en quadrature (QMF), les eigenfiltres et les bancs de filtres FIR les filtres multi-canal de type FIR (*finite impulse response*) et IIR (*infinite impulse response*), ect….

Chapitre II

La classification

Ce chapitre a pour objectif de présenter une vue d'ensemble de la classification. La classification représente un concept central des activités humaines, en informatique, c'est une méthode qui aide à la création de catégories afin de placer un objet dans l'une des catégories créées. Nous commencerons par présenter une introduction à la classification, puis nous aborderons les deux catégories d'apprentissage, l'apprentissage supervisé dont nous présenterons deux approches qui sont les réseaux de neurones artificiels (ANN, Artificial Neural Network) et les arbres de décision ; nous discuterons également d'une méthode de Boosting, AdaBoost que nous appliquerons aux arbres de décision. , ainsi que l'apprentissage non supervisé (ou Clustering) dont nous présenterons aussi deux approches qui sont les C-voisins (C-means) et les C-voisins flous (fuzzy C-menas).

2.1 Introduction

Nous avons vu dans le chapitre précédent un état de l'art sur les différents descripteurs existants . Afin de pouvoir les exploiter pour notre étude, nous devons effectuer une classification. Il y a différentes approches de classification basée sur deux types ; l'apprentissage supervisé et l'apprentissage non supervisé (clustering).
Les approches de classification sont diverses et appliquées à plusieurs domaines, la sécurité informatique, la biologie… dans notre cas, c'est à la reconnaissance de formes, plus précisément à la séparation texte – graphique en langue arabe, ce qui rend la tâche plus difficile. La classification en terme générale consiste à classer un objet dans une catégorie précise selon les caractéristiques de cet objet. Dans notre étude, la classification nous permettra de savoir quel descripteur définit (ou décrit) le mieux le texte ou le graphique donné, ce n'est pas aussi aisé à réaliser. Afin d'y parvenir, nous avons utilisé quatre approches sur lesquelles se baseront nos classifieurs ; les réseaux de neurones artificiels et AdaBoost avec les arbres de décision qui représentent deux approches qui utilisent un apprentissage supervisé, c'est-à-dire,qu' ils ont besoin de connaissances préalables sur les images données. Les deux autres approches utilisent le clustering, ces approches sont les C-Menas et les fuzzy C-means.
Nous n'allons aborder que les approches citées dans ce chapitre au vu du nombre important des approches de classification. Nous avons choisi deux types d'apprentissage afin de pouvoir comparer leurs performances au cours des expérimentations et voir lequel serait le plus adapté pour la résolution de notre problème (la séparation texte – graphique).

2.2 Introduction à la classification et à l'apprentissage

L'apprentissage automatique (machine-learning en anglais), un des champs d'étude de l'intelligence artificielle, est la discipline scientifique concernée par le développement, l'analyse et l'implémentation de méthodes automatisables qui permettent à une machine (au sens large) d'évoluer grâce à un processus d'apprentissage, et ainsi de remplir des tâches qu'il est difficile ou impossible de remplir par des moyens algorithmiques plus classiques.

Des systèmes complexes peuvent être analysés, y compris pour des données associées à des valeurs symboliques (ex: sur un attribut numérique, non pas simplement une valeur numérique, juste un nombre, mais une valeur probabilisée, c'est-à-dire un nombre assorti d'une probabilité ou associé à un intervalle de confiance) ou un ensemble de modalités possibles sur un attribut numérique ou catégoriel. L'analyse peut même concerner des données présentées sous forme de graphes ou d'arbres, ou encore de courbes (par exemple, la courbe d'évolution temporelle d'une mesure ; on parle alors de données continues, par opposition aux données discrètes associées à des attributs-valeurs classiques).

Le premier stade de l'analyse est celui de la ***classification,*** qui vise à *« étiqueter »* chaque donnée en l'associant à une classe.

Principe

Les algorithmes utilisés permettent, dans une certaine mesure, à un système piloté par ordinateur (un robot éventuellement), ou assisté par ordinateur, d'adapter ses analyses et comportements en réponse, en se fondant sur l'analyse de données empiriques provenant d'une base de données ou de capteurs.
La difficulté réside dans le fait que l'ensemble de tous les comportements possibles , compte tenu de toutes les entrées possibles, devient rapidement trop complexe à décrire (on parle d'explosion combinatoire) dans les langages de programmation disponibles. On confie donc à des programmes le soin d'ajuster un modèle permettant de simplifier cette complexité et de l'utiliser de manière opérationnelle. De plus, ce modèle est adaptatif, de façon à prendre en compte l'évolution de la base des informations pour lesquelles les comportements en réponse ont été validés, ce que l'on appelle apprendre ; ceci permet d'auto-améliorer le système d'analyse ou de réponse (commande adaptative…), ce qui est une des formes que peut prendre l'intelligence artificielle.

Ces programmes, selon leur degré de perfectionnement, intègrent éventuellement des capacités de traitement probabiliste des données, d'analyse de données issues de capteurs, de reconnaissance (reconnaissance vocale, reconnaissance de forme, d'écriture, etc.), de datamining, d'informatique théorique, etc.

2.3 Apprentissage supervisé

L'apprentissage supervisé est une technique d'apprentissage automatique où l'on cherche à produire automatiquement des règles à partir d'une base de données d'apprentissage contenant des « exemples » (en général des cas déjà traités et validés).

2.3.1 Les réseaux de neurones artificiels (ANN)

Les réseaux de neurones formels sont des structures (la plupart du temps simulées par des algorithmes exécutés sur des ordinateurs d'usage général, parfois sur des machines ou même des circuits spécialisés) qui prennent leur inspiration (souvent de façon assez lointaine) dans le fonctionnement élémentaire des systèmes nerveux. Ils sont utilisés essentiellement à résoudre des problèmes de classification, de reconnaissance de formes, d'association, d'extraction de caractéristiques, d'identification. Ils deviennent des compléments aux méthodes classiques, et sont même susceptibles de se substituer à celles-ci avec un taux de succès supérieur

- ***Le neurone biologique :*** Le neurone biologique est une cellule vivante spécialisée dans le traitement des signaux électriques. Les neurones sont reliés entre eux par des liaisons appelées axones. Ces axones vont eux-mêmes jouer un rôle important dans le comportement logique de l'ensemble, ils conduisent les signaux électriques de la sortie d'un neurone vers l'entrée (synapse) d'un autre neurone.

  Les neurones font une sommation des signaux reçus en entrée et en fonction du résultat obtenu vont fournir un courant en sortie. (Figure 2.1)

  La structure d'un neurone se compose de trois parties :

  i. La somma : ou cellule d'activité nerveuse, au centre du neurone.
  ii. L'axone : attaché au somma qui est électriquement actif, ce dernier conduit l'impulsion conduite par le neurone.
  iii. Dendrites : électriquement passives, elles reçoivent les impulsions d'autres neurones.

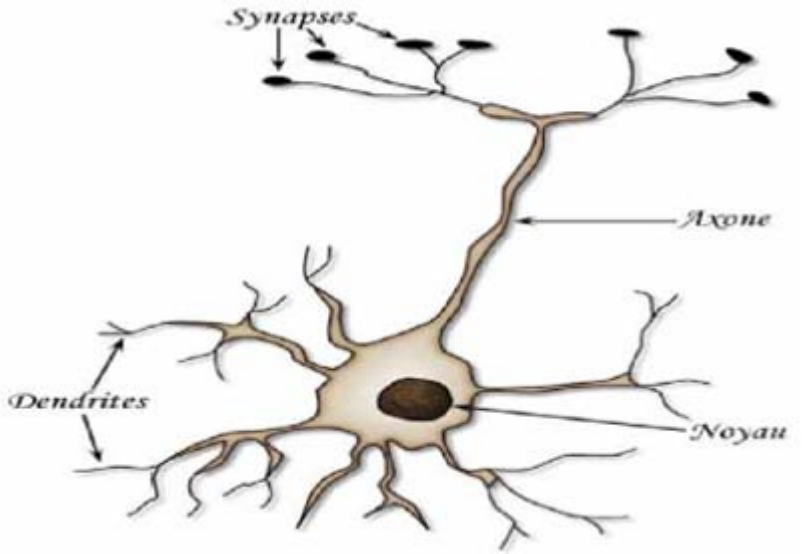


Figure 2.1. Le neurone biologique.

- ***Le neurone formel (artificiel)*** : le neurone artificiel (ou cellule) est un processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones appartenant

à un niveau situé en amont. A chacune des entrées est associé un poids w représentatif de la force de la connexion. Chaque processeur élémentaire est doté d'une sortie unique, qui se ramifie ensuite pour alimenter un nombre variable de neurones appartenant à un niveau situé en aval. A chaque connexion est associé un poids. (figure 2.2)

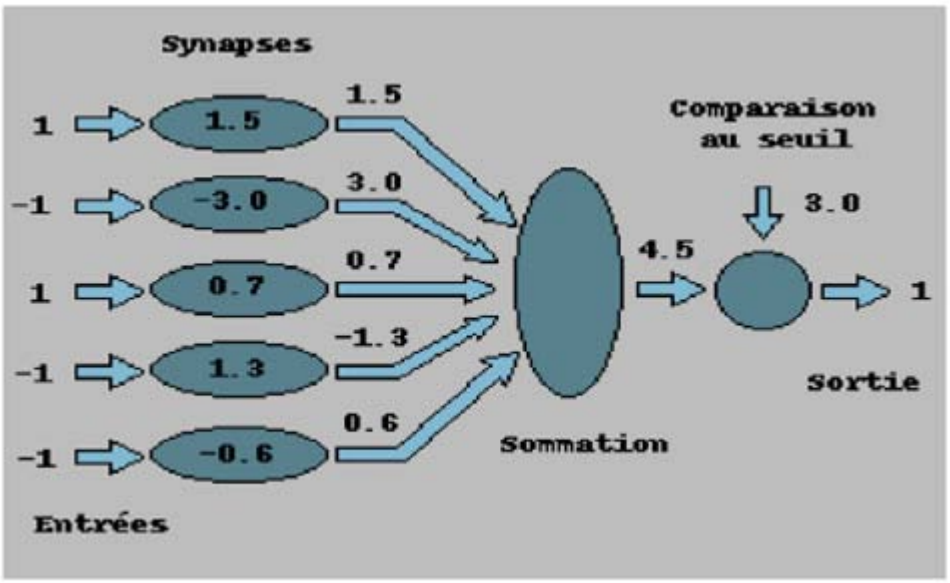


Figure 2.2. Le neurone formel.

L'idée principale derrière les modèles neuronaux provient d'une représentation graphique spéciale ressemblant à la complexité des éléments du cerveau. Ces techniques sont très utiles pour des extensions de modèles mathématiques à un contexte non-linéaire où il devient possible de traiter de nouvelles données et effectuer des modifications de données en temps réel.

- ***Modélisation d'un neurone formel :*** C'est en 1943 que, dans un article resté fameux, McCulloch et Pitts ont émis l'idée simplificatrice du neurone formel, c'est-à-dire un opérateur binaire interconnecté à ses semblables par des « synapses » excitatrices ou inhibitrices. Une assemblée de tels opérateurs en interaction devait avoir des propriétés collectives émergentes, c'est-à-dire capables de certains « calculs » que chacun d'eux séparément est incapable d'exécuter. En 1949, D.O.Hebb, dans un livre resté lui aussi fameux, introduisit la notion de « plasticité synaptique », c'est-à-dire le mécanisme de modification progressive des couplages entre neurones responsable de changements permanents de leurs propriétés collectives, ce que l'on peut appeler « l'apprentissage ». Son hypothèse, limitée à l'augmentation du coefficient de couplage entre deux neurones réels qui sont excités simultanément, a été étendue aux neurones artificiels comme une règle par laquelle les couplages se modifient proportionnellement aux corrélations entre neurones, que ces corrélations soient positives (activation) ou négatives (inhibition).

La modélisation consiste à mettre en œuvre un système de réseaux neuronaux sous un aspect non pas biologique mais artificiel, cela suppose que d'après le principe biologique on aura une correspondance pour chaque élément composant le neurone biologique, donc une modélisation pour chacun d'entre eux.

On pourra résumer cette modélisation par le tableau 1.1, qui nous permettra de voir clairement la transition entre le neurone biologique et le neurone formel.

Neurone biologique	**Neurone artificiel**
Synapses	Poids de connexion
Axones	Signal de sortie
Dendrites	Signal d'entrée
Somma	Fonction d'activation

Tableau 1.1. Analogie entre le neurone biologique et le neurone formel.

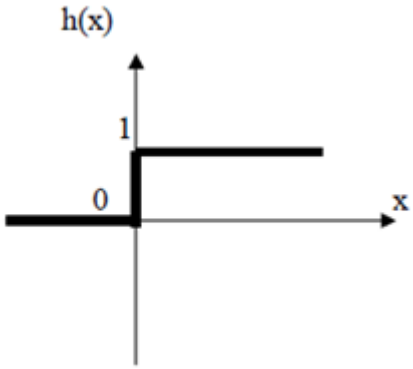
* **Les entrées :** Elles peuvent être :
  - Booléennes.
  - Binaires (0, 1) ou bipolaires (-1, 1).
  - Réelles.
* **Fonction d'activation :** Cette fonction permet de définir l'état interne du neurone en fonction de son entrée totale, citons à titre d'exemple quelques fonctions souvent utilisées :
  - **Fonction binaire a seuil**

Fonction Heaviside définie par

$$h(x) = \begin{cases} 1 & si\ x \geq 0 \\ 0 & sinon \end{cases}$$

Fonction Signe définie par

$$Sgr(x) = \begin{cases} +1 & si\ x \geq 0 \\ -1 & sinon \end{cases}$$



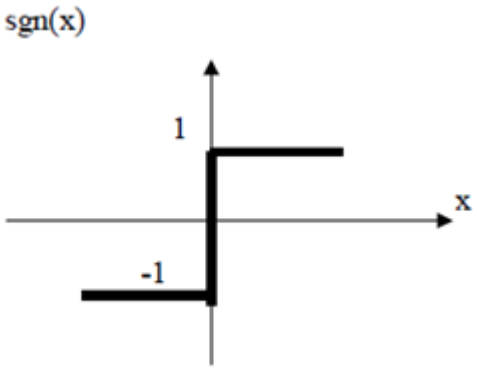


Figure 2.3. Fonction Heaviside. **figure 2.4. Fonction Signe.**

Le seuil introduit une non-linéarité dans le comportement du neurone, cependant il limite la gamme des réponses possibles à deux valeurs.

- ❖ **Fonction linéaire :** c'est l'une des fonctions d'activations les plus simples, sa fonction est définie par : F(x)=x.
- ❖ **Fonction linéaire à seuil ou multi-seuils :** cette fonction représente un compromis entre la fonction linéaire et la fonction seuil : entre ses deux barres de saturation, elle confère au neurone une gamme de réponses possibles. En modulant la pente de la linéarité, on affecte la plage de réponse du neurone.
- ❖ **Fonction sigmoïde :** elle est l'équivalent continu de la fonction linéaire. Etant continu, elle est dérivable, d'autant plus que sa dérivée est simple à calculer.

- ▪ **Fonction de sortie :** elle calcule la sortie d'un neurone en fonction de son état d'activation. En général, cette fonction est considérée comme la fonction identité. Elle peut être : binaire (0, 1) ou bipolaire (-1, 1), réelle. Elle représente le résultat désiré par le réseau de neurones.

Taxonomie et terminologie des réseaux usuels

Nous pouvons classer les réseaux en deux grandes catégories, selon la dépendance de l'évolution de ceux-ci en fonction explicite du temps. Dans le cas des **réseaux statiques**, le temps n'est pas un paramètre significatif. En d'autres termes, la modification de l'entrée n'entraîne qu'une modification stable de la sortie, mais n'entraîne pas de retour d'information vers cette entrée.

Les **réseaux dynamiques**, comme leur nom l'indique, contiennent des re-bouclages partiels ou totaux entre neurones, et ont donc une évolution dépendante du temps. Il faut bien distinguer la dépendance théorique, pour laquelle l'état du réseau à un certain instant dépend de son état à l'instant ou aux instants précédents, du temps nécessaire à obtenir une réponse, dans le cas d'une réalisation matérielle ou d'une simulation sur ordinateur, que le réseau soit statique ou dynamique. On peut ajouter que, formellement, le fonctionnement d'un réseau quelconque dépend de son histoire passée, par le biais de l'apprentissage. Mais cette dépendance est simplement causale, et non pas fonctionnelle, et n'a pas à être prise en compte

dans l'étude détaillée de leurs mécanismes. Le Perceptron multicouche ordinaire ou la carte auto-organisatrice sont des réseaux statiques. Par contre, le réseau de Hopfield ou le Perceptron avec rebouclage sont des réseaux dynamiques.

Type des réseaux de neurones artificiels

Les différent type de RNA sont distinguées par :

* La topologie du réseau.
* Le nœud, ou le type de neurone.
* La règle d'apprentissage associé au réseau.

Les quatre types de RNA les plus utilisés sont énumérés ci-dessous.

I. ***Perceptron a une ou plusieurs couches (PMC), ou le single or Multi-Layered Perceptrons (MLP)*** : Le réseau dans ce modèle est formé de trois couches : Une couche d'entrée (la rétine), fournissant des donnés à une couche intermédiaire, chargée des calculs, cela en fournissant la somme des impulsions qui lui viennent des cellules auxquelles elle est connectée, et elle répond généralement suivant une loi définie avec un seuil, elle-même connectée à la couche de sortie (couche de décision), représentant les exemples à mémoriser. Seule cette dernière couche renvoie des signaux à la couche intermédiaire, jusqu'à ce que leurs connexions se stabilisent.

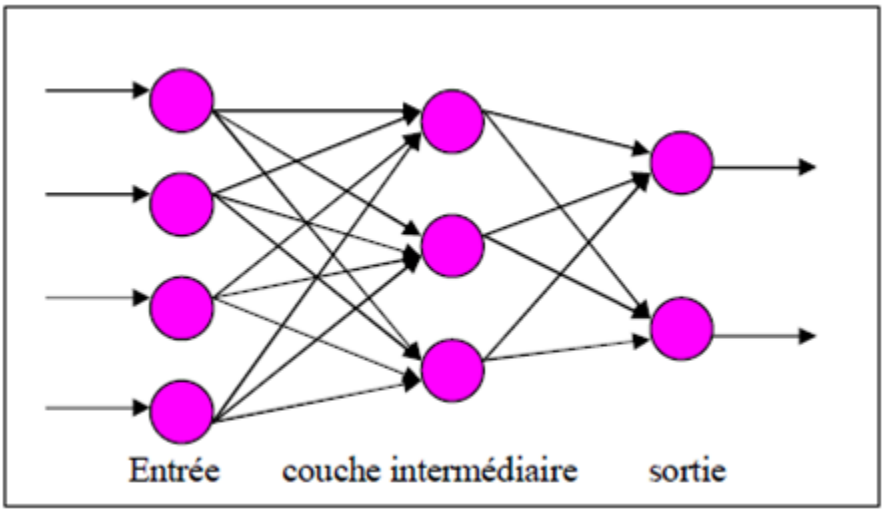


Figure 2.5. Le modèle du perceptron.

II. ***Réseaux à base de fonction radial ou Radial-basis function (RBF)*** : Ils utilisent des approximateurs locaux de fonctions et sont des Réseaux à une seule couche.

III. ***Réseaux de Hopfield*** : Le modèle de Hopfield utilise l'architecture des réseaux entièrement connectés et récurrents (dont les connexions sont non orientées et ou chaque neurone n'agit pas sur lui-même). Les sorties sont en fonction des entrées et du dernier état pris par le réseau.

IV. ***Les cartes de Kohonen ou Kohonen Self-organising Feature Maps (SOFM)*** : Ce modèle a été présenté par Kohonen en 1982 en se basant sur des constatations

biologiques. Il a pour objectif de présenter des données complexes et appartenant généralement à un espace discret de grandes dimensions dont la topologie est limitée à une ou deux dimensions. Les cartes de Kohonen sont réalisées à partir d'un réseau à deux couches, une en entrée et une en sortie. Notons que les neurones de la couche d'entrée sont entièrement connectés à la couche de sortie. Les neurones de la couche de sortie sont placés dans un espace d'une ou de deux dimensions en général, chaque neurone possède donc des voisins dans cet espace. Et qu'enfin, chaque neurone de la couche de sortie possède des connexions latérales récurrentes dans sa couche (le neurone inhibe les neurones éloignés et laisse agir les neurones voisins).

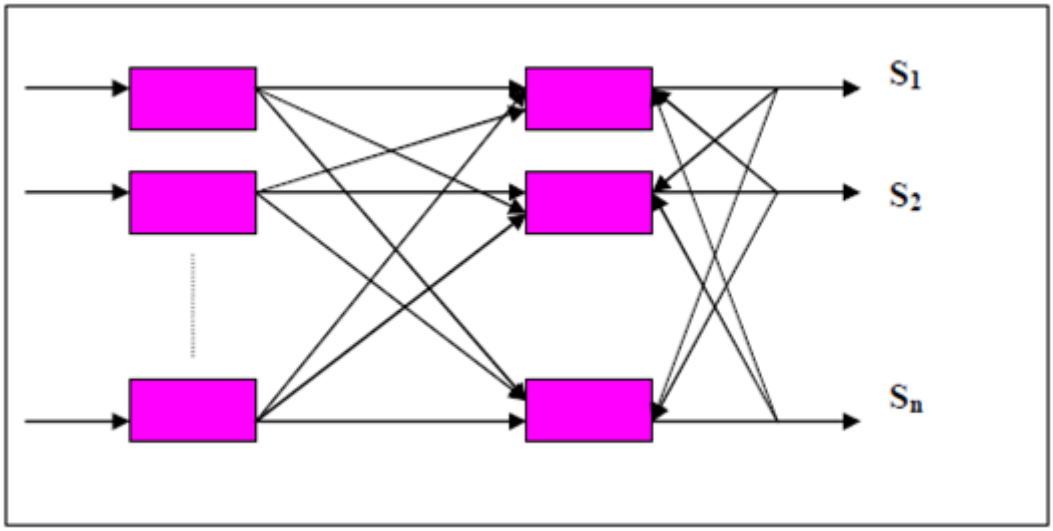


Figure 2.6. Le modèle de Kohonen.

Propriétés, motivation et limites des réseaux neuronaux

L'intérêt porté aujourd'hui aux réseaux de neurones tient sa justification dans les quelques propriétés intéressantes qu'ils possèdent et qui devraient permettre de dépasser les limitations de l'informatique traditionnelle, tant au niveau programmation qu'au niveau machine.

- Facilité d'application car ne nécessitant pas une compréhension approfondie.
- Le parallélisme : Cette notion se situe à la base de l'architecture des réseaux de neurones considérés comme ensembles d'entités élémentaires travaillant simultanément. Par l'étude du fonctionnement des réseaux de neurones. on pourrait aboutir de nouvelles techniques de formalisation de problèmes qui permettraient de les traiter en parallèle.
- La capacité d'adaptation :Celle-ci se manifeste par la capacité d'apprentissage qui permet de tenir compte de nouvelles contraintes ou de nouvelles données du

monde extérieur.

- La mémoire distribuée : Dans les réseaux de neurones, la mémoire correspond à une carte d'activation de neurones. Cette carte est en quelque sorte un codage du fait mémorisé ce qui attribue a ces réseaux l'avantage de résister aux bruits (pannes) car la perte d'un élément ne correspond pas a la perte d'un fait mémorisé.
- La capacité de généralisation : Cette capacité est importance surtout dans le cas ou la constitution de recueils d'expertise pour un système expert devient difficile (reconnaissance intuitive ou implicite). Les réseaux neuronaux peuvent apprendre à retrouver des règles a partir d'exemples.

Voici quelques unes des caractéristiques des problèmes bien adaptes a une résolution par les réseaux de neurones:

- Les règles qui permettent de résoudre le problème sont inconnues ou très difficiles à expliciter ou à formaliser. Cependant, on dispose d'un ensemble d'exemples qui correspondent a des entrées du problème et à des solutions qui leurs sont données par des experts.
- Le problème fait intervenir des données bruitées on incomplètes.
- Le problème peut évoluer.
- Le problème nécessite une grande rapidité de traitement.

Les principales limites d'utilisation des réseaux neuronaux sont :

- Difficulté d'utilisation du parallélisme inhérent aux réseaux de neurones: la plupart des réseaux sont simulés sur des machines séquentielles ce qui entraine des temps de calculs importants (surtout pendant l'apprentissage).
- Les performances des réseaux neuronaux dépendent de la qualité et de la pertinence du pré traitement effectué lors de l'apprentissage.
- L'un des principaux reproches fait aux réseaux neuronaux est leur incapacité à expliquer les résultats qu'ils fournissent. Ces réseaux se présentent comme des boites noires dont les règles de fonctionnement sont inconnues. La qualité de leurs performances ne peut être mesurée que par des méthodes statistiques, ce qui cause une certaine méfiance de la part des utilisateurs.
- En pratique, se pose dans tous les cas un problème essentiel, relatif a l'architecture du réseau neuronal car aucune procédure formelle n'est disponible pour répondre aux questions (dans le cas du PMC): Combien faut-il mettre de couches de neurones, et combien de neurones par couches?

On pourrait penser qu'il suffirait de mettre beaucoup de neurones. Mais, comme dans

toutes les techniques d'approximation, mettre trop de paramètres (chaque neurone introduit des paramètres supplémentaires, les poids synaptiques) n'est pas une bonne solution. Cela conduit a une très (trop) bonne approximation sur les exemples appris, mais a de très mauvaises sur des exemples non appris. On parle alors d'apprentissage par cœur, car les exemples appris sont tous parfaitement mémorisés, mais la relation générale entre entrées et sorties ne l'est pas.

Pour résoudre ce problème, deux alternatives possibles font actuellement l'objet de recherches actives :

L'une consiste à choisir initialement un réseau surdimensionné et à le simplifier progressivement en enlevant les paramètres les moins significatifs (élagage). La seconde possibilité consistes partir d'un réseau vide, et a ajouter des neurones en filtration des besoins (greffage).

2.3.2 Les arbres de décision

La construction des arbres de décision à partir de données est une discipline déjà ancienne. Les statisticiens en attribuent la paternité à Morgan et Sonquist [5.1.1] qui, les premiers, ont utilisé les arbres de régression dans un processus de prédiction et d'explication (AID – Automatic Interaction Detection). Il s'en est suivi toute une famille de méthodes, étendues jusqu'aux problèmes de discrimination et classement, qui s'appuyaient sur le même paradigme de la représentation par arbres [5.1.2] ; [5.1.3]. On considère généralement que cette approche a connu son apogée avec la méthode CART (Classification and Regression Tree) de [5.1.4] décrite en détail dans une monographie qui fait encore référence aujourd'hui.

Construire un arbre de décision

La popularité de la méthode repose en grande partie sur sa simplicité. Il s'agit de trouver un partitionnement des individus que l'on représente sous la forme d'un arbre de décision. L'objectif est de produire des groupes d'individus les plus homogènes possibles du point de vue de la variable à prédire. Il est d'usage de représenter la distribution empirique de l'attribut à prédire sur chaque sommet (nœud) de l'arbre. Pour mieux appréhender la démarche, nous allons reprendre et dérouler un exemple qui est présenté dans l'ouvrage de [5.1.5]. Le fichier est composé de 14 observations, il s'agit d'expliquer le comportement des individus par rapport à un jeu {jouer, ne pas jouer} à partir des prévisions météorologiques (Tableau 1.2.).

Numéro	Ensoleillement	Température (F°)	Humidité (%)	Vent	Jouer
1	Soleil	75	70	Oui	Oui
2	Soleil	80	90	Oui	Non
3	Soleil	85	85	Non	Non
4	Soleil	72	95	Non	Non

5	Soleil	69	70	Non	Oui
6	Couvert	72	90	Oui	Oui
7	Couvert	83	78	Non	Oui
8	Couvert	64	65	Oui	Oui
9	Couvert	81	75	Non	Oui
10	Pluie	71	80	Oui	Non
11	Pluie	65	70	Oui	Non
12	Pluie	75	80	Non	Oui
13	Pluie	68	80	Non	Oui
14	Pluie	70	96	Non	Oui

Tableau 1.2. Donner « Weather » (Quinlan 1993).

L'arbre de décision correspondant est décrit ci-dessous (Figure 2.7).

- Le premier sommet est appelé la « racine » de l'arbre. Il est situé sur le premier niveau. Nous y observons la distribution de fréquence de la variable à prédire « Jouer ». Nous constatons qu'il y a bien 14 observations, dont 9 « oui » (ils vont jouer) et 5 « non ».
- La variable « ensoleillement » est la première variable utilisée ; on parle de variable segmentation. Comme elle est composée de 3 modalités {soleil, couvert, pluie}, elle produit donc 3 sommets enfants.
- La première arête (la première branche), à gauche, sur le deuxième niveau, est produite à partir de la modalité « soleil » de la variable « ensoleillement ». Le sommet qui en résulte couvre 5 observations correspondant aux individus {1, 2, 3, 4, 5}, la distribution de fréquence nous indique qu'il y a 2 « jouer = oui » et 3 « jouer = non ».
- La seconde arête, au centre, correspond à la modalité « couvert » de la variable de segmentation « ensoleillement » ; le sommet correspondant couvre 4 observations, tous ont décidé de jouer (dans le tableau ce sont les individus n°6 à 9). Ce sommet n'ayant plus de sommets enfants, ce qui est normal puisqu'il est « pur » du point de vue de la variable à prédire, il n'y a pas de contre-exemples. On dit qu'il s'agit d'une feuille de l'arbre.

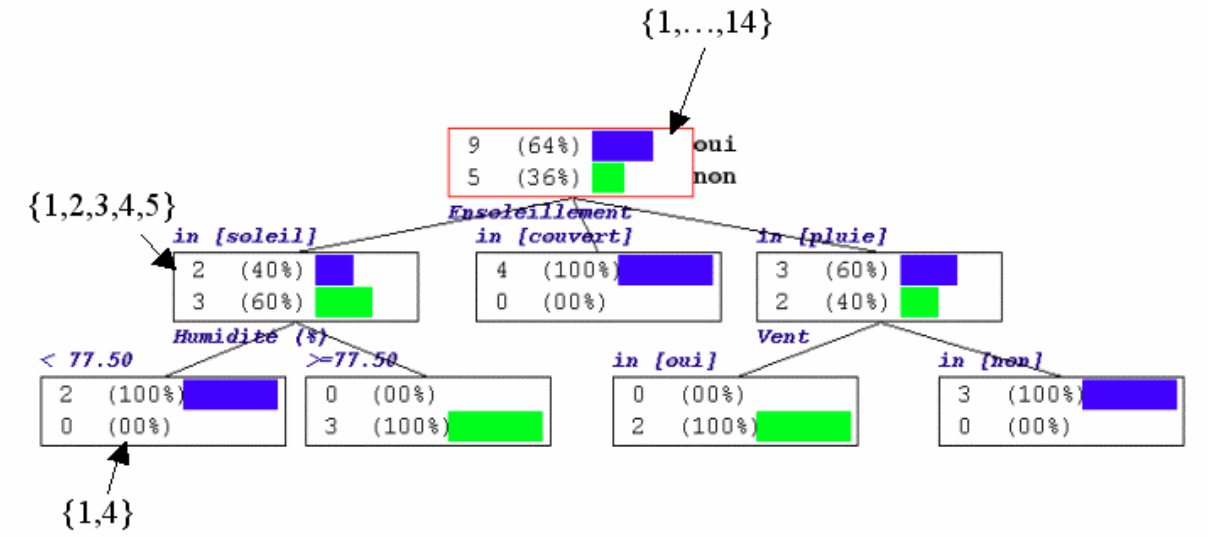


Figure 2.7. Arbre de décision sur le fichier "weather"

Reprenons le nœud le plus à gauche sur le deuxième niveau de l'arbre. Ce sommet, qui n'est pas pur, est segmenté à l'aide de la variable « humidité ». Comme le descripteur est continu, il a été nécessaire de définir un seuil dit de discrétisation qui permet de produire le meilleur partitionnement. Dans notre exemple, le seuil qui a été choisi est 77.5 %. Il a permis de produire deux feuilles complètement pures.

- Ce processus est réitéré sur chaque sommet de l'arbre jusqu'à l'obtention de feuilles pures. Il s'agit bien d'un arbre de partitionnement : un individu ne peut être situé dans deux feuilles différentes de l'arbre.
- Le modèle de prédiction peut être lu très facilement. On peut traduire un arbre en une base de règles sans altération de l'information. Le chemin menant d'un sommet vers la racine de l'arbre peut être traduit en une partie prémisse d'une règle de prédiction de type attribut-valeur « SI variable 1 = valeur 1 ET variable 2 = valeur 2 … ».
- Pour classer un nouvel individu, il suffit de l'injecter dans l'arbre, et de lui associer la conclusion attachée à la feuille dans laquelle il aboutit.

Le meilleur arbre de décision n'existe pas toujours, trouver le meilleur arbre de décision est, en général, un problème NP-complet, et enfin, l'erreur apparente est une vision très optimiste de l'erreur réelle.

Généralités sur l'apprentissage des arbres de décision

L'idée centrale est de diviser récursivement les exemples de l'ensemble d'apprentissage par des tests définis à l'aide des attributs jusqu'à ce que l'on obtienne des sous-ensembles d'exemples ne contenant (presque) que des exemples appartenant tous à une même classe.

Dans toutes les méthodes, on trouve les trois opérateurs suivants :

I. Décider si un nœud est terminal. Par exemple : tous les exemples sont dans la même classe, il y a moins d'un certain nombre d'erreurs, ...

II. Sélectionner un test à associer à un nœud. Par exemple : aléatoirement, utiliser des critères statistiques, ...
III. Affecter une classe à une feuille. On attribue la classe majoritaire sauf dans le cas où l'on utilise des fonctions coût ou risque.

Nous pouvons alors calculer un arbre de décision dont l'erreur apparente est faible, voire nulle. Un arbre de décision parfait est un arbre de décision tel que tous les exemples de l'ensemble d'apprentissage soient correctement classifiés. Un tel arbre n'existe pas toujours (s'il existe deux exemples tels que à deux descriptions identiques correspondent deux classes différentes). Le meilleur arbre de décision est le plus petit arbre de décision parfait, une mesure de complexité étant choisie.

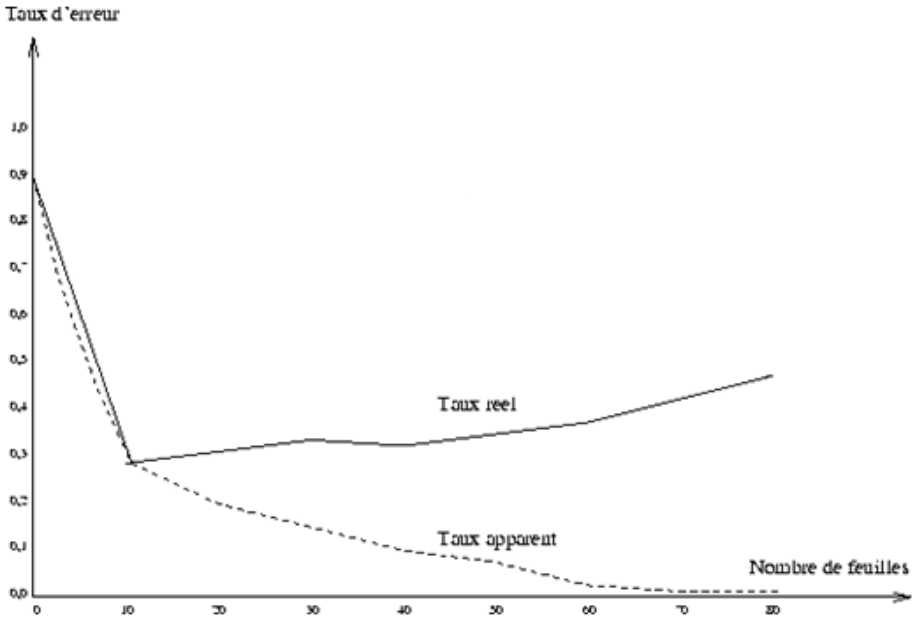


Figure 2.8. Erreur réelle et erreur apparente pour des données réelles de reconnaissance de caractères (Breiman et al.)

Donc il faut faire le choix d'une mesure de complexité et rechercher le meilleur compromis complexité / adéquation aux données. Les méthodes procèdent toujours en deux phases. Dans une première phase, on calcule récursivement, après avoir défini les trois opérateurs ci-dessous, un ``bon" arbre de décision (erreur apparente faible). Dans une seconde phase, on élague l'arbre obtenu pour essayer de faire diminuer l'erreur réelle. On procède en deux phases car il n'existe aucune heuristique satisfaisante permettant d'arrêter au ``bon" moment la croissance de l'arbre de décision. En effet, il se peut que l'on choisisse un test qui fasse augmenter temporairement l'erreur mais qui permettra, par la suite, de faire diminuer nettement cette erreur. De plus, le risque d'arrêter trop tôt la croissance de l'arbre est plus important que de l'arrêter trop tard (voir Figure 2.8.).

2.4 Clustering

Le principe du clustering est de diviser les données en plusieurs sous-ensembles. Comme le calcul de tous les sous-ensembles possibles n'est pas faisable, on utilise des heuristiques sous

la forme d'optimisation itérative de la fonction objective. Chaque algorithme possède un schéma de relocalisation qui réassigne itérativement les objets parmi les *K* classes. Dans la méthode du *k-means*, une classe est représentée par son centroïde, qui est une moyenne (souvent pondérée) des objets appartenant µa cette classe.

2.4.1 La méthode des k-means

Bien qu'elle ne fasse appel qu'à un formalisme limité et que son efficacité soit dans une large mesure attestée par les seuls résultats expérimentaux, la méthode des k-means est probablement la technique de partitionnement la mieux adaptée actuellement aux vastes recueils de données ainsi que la plus utilisée pour ce type d'application.

Description de la méthode

On considère l'espace de n points de dimension p suivant :

$$X \quad : \quad \begin{bmatrix} x_1^1 & \dots & x_1^j & \dots & x_1^p \\ \dots & \dots & \dots & \dots & \dots \\ x_i^1 & \dots & x_i^j & \dots & x_i^p \\ \dots & \dots & \dots & \dots & \dots \\ x_n^1 & \dots & x_n^j & \dots & x_n^p \end{bmatrix}$$

- On suppose que les n points peuvent être groupés en c clusters c < n
- Les clusters sont décrits par leurs centres :

$$\boldsymbol{V_k = \left[V_k^1, V_k^2, \dots., V_k^j, \dots, V_k^p\right], \quad 1 \leq K \leq c}$$

On note d (i,k) la distance entre le point X_i et le centre V_k

- Le point est affecté au cluster dont le centre est le plus proche (au sens de d)
- On note mk la moyenne des vecteurs dans le cluster k

Algorithme 1

Initialiser la position des centres :

$$\boldsymbol{V_k = \left[V_k^1, V_k^2, \dots., V_k^j, \dots, V_k^p\right], \quad 1 \leq K \leq c}$$

Calculer les mk

Jusqu'à ce qu'il n'y ait plus de changement sur les mk Faire

Chaque point Xi est affecté au cluster le plus proche
Calculer les nouveaux mk

Fin Jusqu'à

La figure suivante illustre le déroulement de l'algorithme pour :

P=2

c=4

$V_k = [0,0]$, $1 \le k \le 4$

Les cercles rouges représentent les positions successives des centres des 4 clusters.

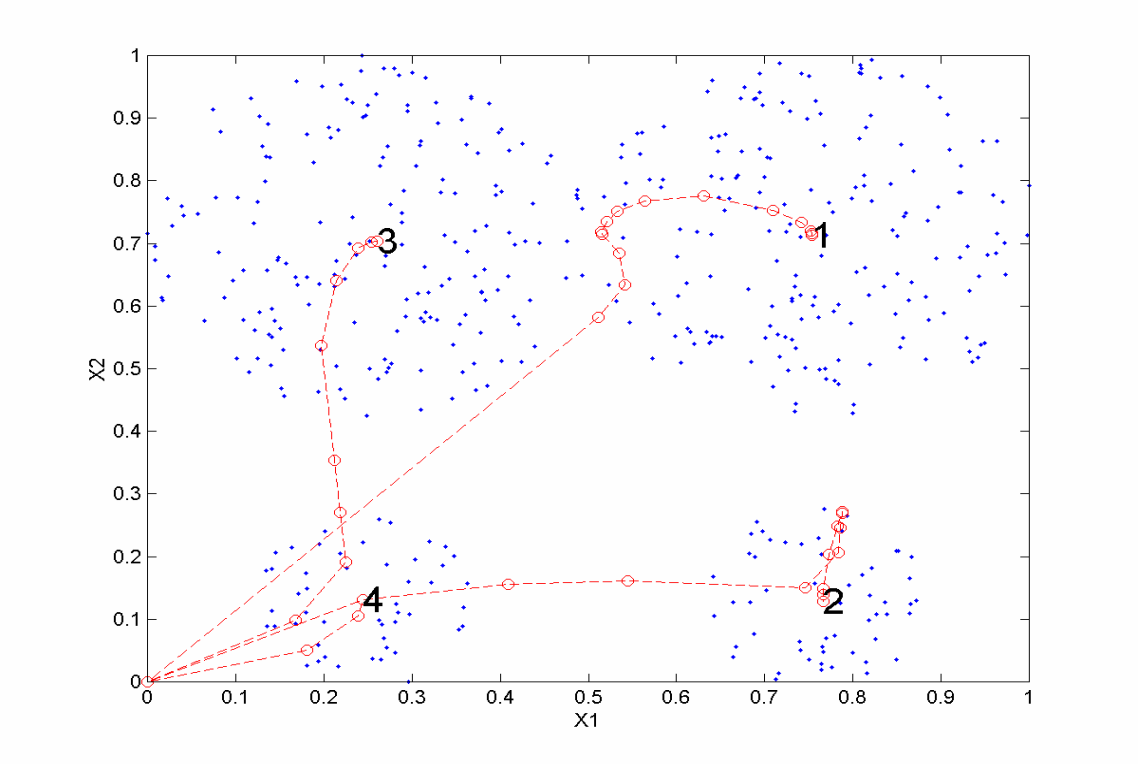


Figure 2.9. Exemple d'application du kmeans

2.4.2 Fuzzy C-means

En 1981, Besdek a proposé une version floue du K-means : l'algorithme du Fuzzy C-means (FCM).

Description de la méthode :

On considère l'espace de n points de dimension p suivant :

$$X \quad : \quad \begin{bmatrix} x_1^1 & \dots & x_1^j & \dots & x_1^p \\ \dots & \dots & \dots & \dots & \dots \\ x_i^1 & \dots & x_i^j & \dots & x_i^p \\ \dots & \dots & \dots & \dots & \dots \\ x_n^1 & \dots & x_n^j & \dots & x_n^p \end{bmatrix}$$

- On suppose que les n points peuvent être groupés en c clusters c < n
- Les clusters sont décrits par leurs centres :

$$\boldsymbol{V_k = [V_k^1, V_k^2, \dots, V_k^j, \dots, V_k^p], \quad 1 \leq K \leq c}$$

- On considère la matrice de proximité suivante :

$$U = \begin{bmatrix} u_{11} & \dots & u_{1k} & \dots & u_{1n} \\ \dots & \dots & \dots & \dots & \dots \\ u_{i1} & \dots & u_{ik} & \dots & u_{in} \\ \dots & \dots & \dots & \dots & \dots \\ u_{c1} & \dots & u_{ck} & \dots & u_{cn} \end{bmatrix}, \text{ avec k = 1,...,n et i = 1,...,c}$$

u_{ik} Représente le degré d'appartenance du point X_k au centre V_i

$$\boldsymbol{u_{ik} = \left[\sum_{j=1}^{c} \left(\frac{d_{ik}}{d_{jk}}\right)^{\frac{2}{m-1}}\right]}$$

d_{ik} Représente la distance entre V_i et V_k

$$\boldsymbol{u_{ik} = \left[\sum_{j=1}^{c} \left(\frac{\|X_k - V_i\|}{\|X_k - V_j\|}\right)^{\frac{2}{m-1}}\right]^{-1}}$$

si nous choisissons la distance euclidienne

Algorithme 2

---

1. Initialiser la position des centres :
   $\boldsymbol{V_k = [V_k^1, V_k^2, \dots, V_k^j, \dots, V_k^p], \quad 1 \leq K \leq c}$
   1 =0, initialiser la matrice $\boldsymbol{U^{(1)}}$
2. Calculer la position du centre :

$$V_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{cn} (u_{ik})^m}, \quad 1 \leq i \leq c$$

3. Calculer la matrice $U^{(}$

4. **Si** $\| U^{(l+1)} - U^{(1)} \|$

Arrêter l'algorithme

Sinon l =l +l et retourner en 2)

La figure suivante illustre l'application de cette méthode sur un nuage de points en deux dimensions (X, Y). A l'état initial les trois centres sont positionnés au centre du nuage. Puis après quelques itérations, ils convergent vers trois positions d'équilibre qui sont les barycentres des trois clusters constituant le nuage de points.

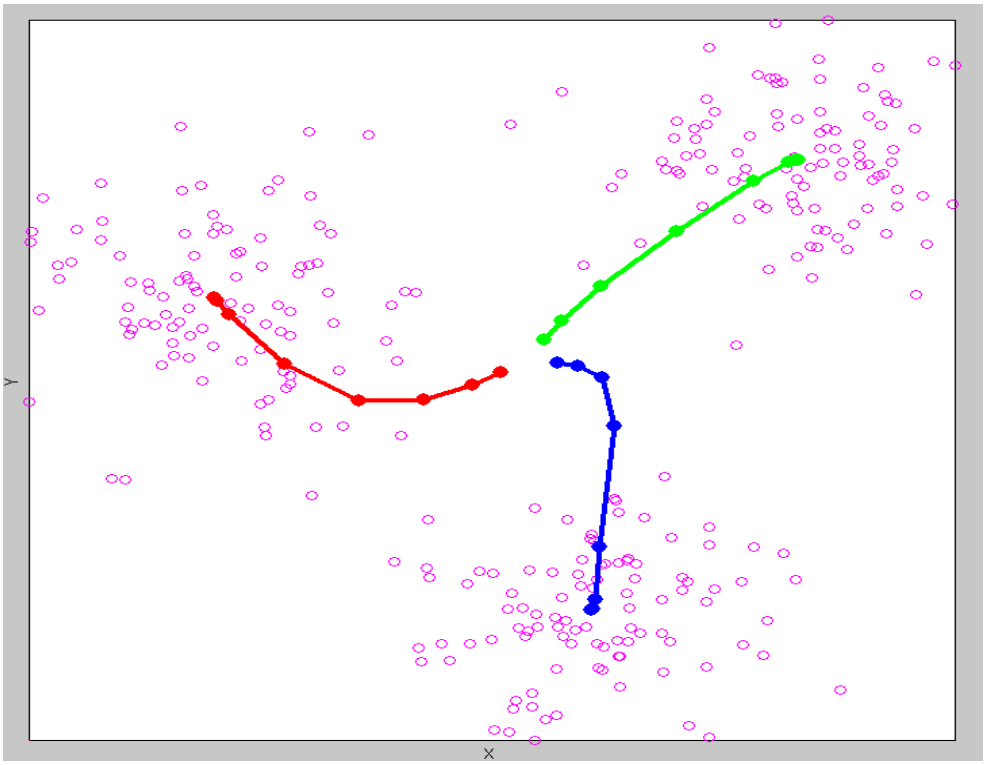


Figure 2.10. Exemple d'application du Fuzzy c-means

2.5 Boosting

Le mot boosting s'applique à des méthodes générales capables de produire des décisions très précises (au sens d'une fonction de perte) à partir d'un ensemble de règles de décision « faibles », c'est-à-dire dont la seule garantie est qu'elles soient un peu meilleures que le hasard.

Ces méthodes s'appliquent aussi bien à l'estimation de densité qu'à la régression ou à la classification. Pour simplifier, nous nous concentrons ici sur la tâche de classification binaire.

Dans sa version « par sous-ensembles », cette technique fait produire à l'algorithme trois résultats selon la partie de l'ensemble d'apprentissage sur laquelle il apprend, puis combine les trois apprentissages réalisés pour fournir une règle de classification plus efficace. Examinons d'abord cette technique avant de voir comment la généraliser à l'aide de distributions de probabilités sur les exemples.

2.5.1 Boosting par sous-ensembles

Schapire [2.5.1] a développé le premier algorithme de boosting pour répondre à une question de Kearns : est-il possible de rendre aussi bon que l'on veut un algorithme d'apprentissage « faible », c'est-à-dire un peu meilleur que le hasard ? Shapire montra qu'un algorithme faible peut toujours améliorer sa performance en étant entraîné sur trois échantillons d'apprentissage bien choisis. Nous ne nous intéressons ici qu'à des problèmes de classification binaire.

L'idée est d'utiliser un algorithme d'apprentissage qui peut être de nature très diverse (un arbre de décision, une règle bayésienne de classification, une décision dépendant d'un hyperplan, etc.) sur trois sous-ensembles d'apprentissage.

1) On obtient d'abord une première hypothèse h1 sur un sous-échantillon S1 d'apprentissage de taille m1 < m (m étant la taille de S l'échantillon d'apprentissage disponible).
2) On apprend alors une deuxième hypothèse h2 sur un échantillon S2 de taille m2 choisi dans $S - S1$ dont la moitié des exemples sont mal classés par h1.
3) On apprend finalement une troisième hypothèse h3 sur m3 exemples tirés dans $S - S1 - S$ pour lesquels h1 et h2 sont en désaccord.
4) L'hypothèse finale est obtenue par un vote majoritaire des trois hypothèses apprises : H = vote majoritaire (h1, h2, h3) Le théorème de Schapire sur la « force de l'apprentissage faible » prouve que H a une performance supérieure à celle de l'hypothèse qui aurait été apprise directement sur l'échantillon S.

Une illustration géométrique du boosting selon cette technique de base est donnée dans les figures 2.11, 2.12 et 2.13.

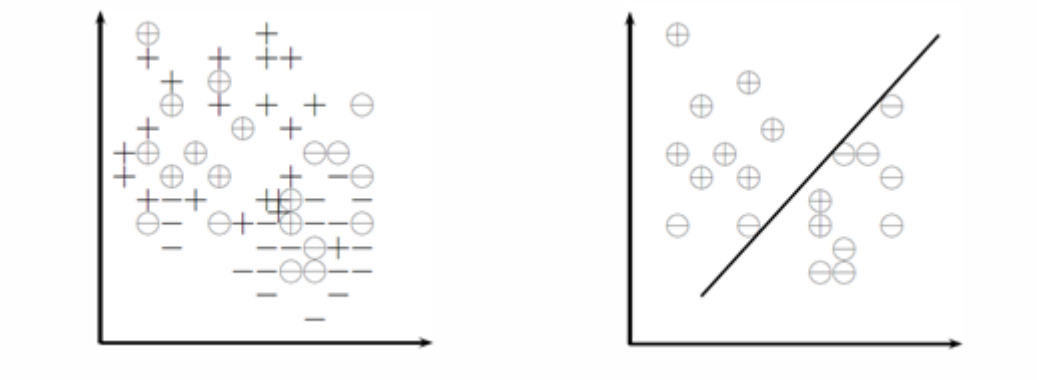


Figure 2.11. À gauche : l'ensemble d'apprentissage S et le sous-ensemble S_1 (points entourés). À droite : l'ensemble S_1 et la droite C_1 apprise sur cet ensemble.

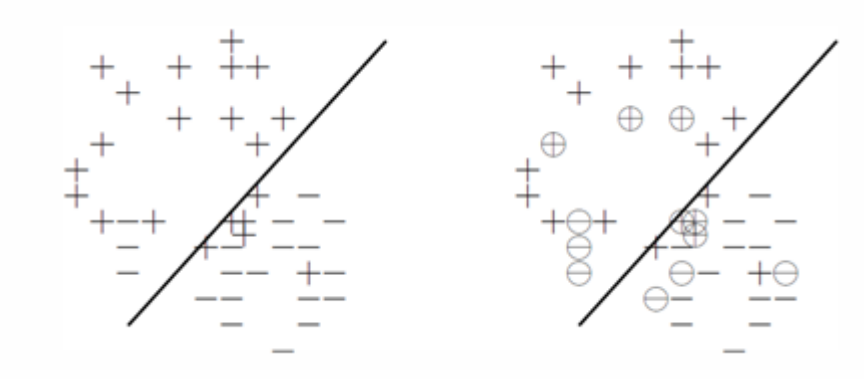


Figure 2.12. À gauche : l'ensemble S – S_1 et la droite C_1 apprise sur S_1. À droite : un ensemble S_2 inclus dans S – S_1 parmi les plus informatifs pour C_1 (points entourés)

Idéalement, les trois ensembles d'exemples extraits de S devraient le vider de tous ses exemples, ce qui revient à dire que la somme des valeurs m1, m2 et m3 doit approcher m. C'est la façon de tirer un profit maximal de S. Mais on conçoit que ce réglage ne soit pas forcément facile à obtenir en pratique : si l'algorithme A est performant sur S, m2 pourra être pris bien inférieur à m1, alors que la proportion pourrait être inverse si A est seulement un peu meilleur qu'un tirage de classe au hasard. En général, on règle empiriquement les proportions des trois ensembles en effectuant plusieurs essais, jusqu'à ce que tous les éléments de S ou presque participent au processus.

On peut utiliser récursivement la méthode et procéder avec neuf sous-ensembles, vingt-sept sous-ensembles, etc. Mais la meilleure généralisation est de faire glisser la notion de fonction caractéristique (qui vaut 1 sur les points d'un sous-ensemble et 0 partout ailleurs) vers celle de distribution de probabilités sur les points de l'ensemble d'apprentissage. Cette technique a déjà été employée pour les fenêtres de Parzen. C'est ce que réalise l'algorithme que nous présentons maintenant.

3 *Le boosting probabiliste et l'algorithme AdaBoost*

Trois idées fondamentales sont à la base des méthodes de boosting probabiliste :

1. L'utilisation d'un comité d'experts spécialisés que l'on fait voter pour atteindre une décision.

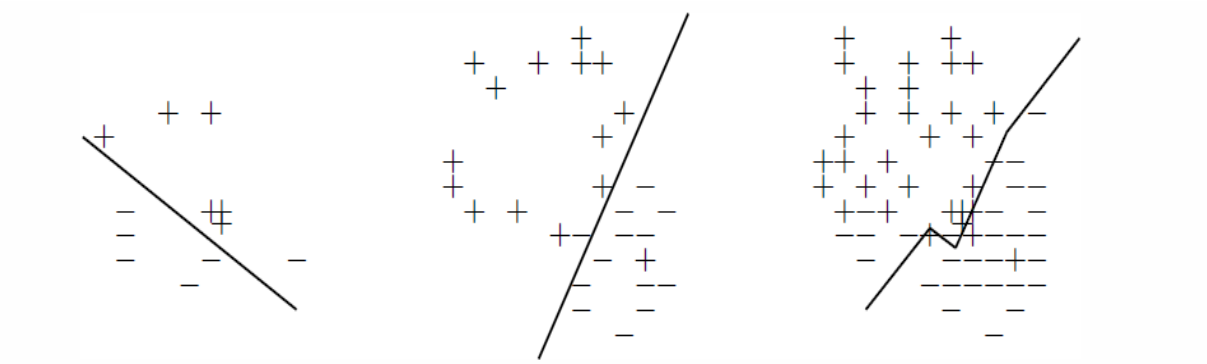


Figure 2.13. À gauche : l'ensemble S_2 et la droite séparatrice C_1 apprise sur cet ensemble. Au centre : l'ensembleS_3= S–S_1–S_2 et la droite séparatrice C_3 apprise sur cet ensemble. À droite : l'ensemble S et la combinaison des 3 droites séparatrices apprises sur cet ensemble.

2. La pondération adaptative des votes par une technique de mise à jour multiplicative.
3. La modification de la distribution des exemples disponibles pour entraîner chaque expert, en surpondérant au fur et à mesure les exemples mal classés aux étapes précédentes.

L'algorithme le plus pratiqué s'appelle AdaBoost (pour adaptive boosting). L'une des idées principales (voir l'algorithme 3) est de définir à chacune de ses étapes $1 \leq t \leq T$, une nouvelle distribution $D_t$de probabilités a priori sur les exemples d'apprentissages en fonction des résultats de l'algorithme à l'étape précédente. Le poids à l'étape t d'un exemple (x_i,u_i) d'indice i est noté $P_t(i)$. Initialement, tous les exemples ont un poids identique, puis à chaque étape, les poids des exemples mal classés par l'apprenant sont augmentés, forçant ainsi l'apprenant à se concentrer sur les exemples difficiles de l'échantillon d'apprentissage.

Algorithme 3 : AdaBoost dans le cas d'un apprentissage de concept.

Début

$S = \{(x_1, u_1), \ldots\ldots, (x_m, u_m)\}$, **avec** $u_i \in \{+1, -1\}$, ***i = 1, m***

Pour tous **les *i = 1, m*** faire $p_0(x_i) \leftarrow \frac{1}{m}$ $t \leftarrow 1$

Pour tout $t \leq T$ **faire**

Tirer un échantillonnage de S_t dans S selon les probabilités p_t

Apprendre une règle de classification de h_t sur S_t par l'algorithme *A*

Soit ε_t l'erreur apparente de h_t sur S_t, calculer $\alpha_t \leftarrow \frac{1}{2} \ln \frac{1-\varepsilon_t}{\varepsilon_t}$

Pour tous les $i = 1, m$ **faire**

$p_t(x_i) \leftarrow \frac{p_t(x_i)}{Z_t} e^{-\alpha_t}$ **si** $h_t = u_i$

$p_t(x_i) \leftarrow \frac{p_t(x_i)}{Z_t} e^{+\alpha_t}$ **si** $h_t \neq u_i$

(Z_t est une valeur de normalisation)

Fin

$t \leftarrow t + 1$

Fin

Fournir en sortie l'hypothèse finale $H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$

Fin

À chaque étape t, l'apprenant cherche une hypothèse h_t X → {−1, +1} bonne pour la distribution D_t sur X. La performance de l'apprenant est mesurée par l'erreur apparente :

$$\varepsilon_t = p_t[h_t(x) \neq u_i] = \sum_{i:h_t(x_i) \neq u_i} p_t(i)$$

On note que l'erreur est mesurée en fonction de la distribution D_t sur laquelle l'apprenant est entraîné. En pratique, soit les poids des exemples sont effectivement modifiés, soit c'est la

probabilité de tirage des exemples qui est modifiée et l'on utilise un tirage avec remise (bootstrap). Chaque hypothèse ht apprise est affectée d'un poids α_t mesurant l'importance qui sera donnée à cette hypothèse dans la combinaison finale. Ce poids est positif si $\varepsilon_t \leq 1/2$ (on suppose ici que les classes « + » et « − » sont équiprobables et donc que l'erreur d'une décision aléatoire est de1/2). Plus l'erreur associée à l'hypothèse ht est faible, plus celle-ci est dotée d'un coefficient α_t important.

L'examen des formules de mise à jour des poids des hypothèses dans l'algorithme 3 suggère que vers la fin de l'apprentissage, le poids des exemples difficiles à apprendre devient largement dominant. Si une hypothèse performante sur ces exemples peut être trouvée (c'est-à-dire avec $\varepsilon_t \approx 0$), elle sera alors dotée d'un coefficient αt considérable. L'une des conséquences possibles est que les exemples bruités, sur lesquels finit par se concentrer l'algorithme, perturbent gravement l'apprentissage par boosting. C'est en effet ce qui est fréquemment observé.

À la fin de cet algorithme, chaque règle de classification ht est pondérée par une valeur α_t calculée en cours de route. La classification d'un nouvel exemple (ou des points de S pour obtenir l'erreur apparente) s'opère en utilisant la règle :

$$H(x) = signe\left(\sum_{t=1}^{t=T} \alpha_t \, h_t\,(x)\right)$$

En un sens, on voit que le boosting construit l'hypothèse finale comme une combinaison linéaire d'une base de fonctions, dont les éléments sont les hypothèses ht. On retrouve là un thème fréquent dans les techniques d'apprentissage (par exemple les SVM, les méthodes d'approximation bayésiennes, etc.).

AdaBoost généralise le premier algorithme de boosting

Le premier algorithme de boosting utilisait un vote sur trois classificateurs appris sur trois sous-ensembles d'apprentissage. D'une certaine façon, il effectuait trois itérations d'AdaBoost avec des valeurs binaires pour les poids p_1, p_2 et p_3. De plus, il choisissait le second sous-ensemble comme des exemples pour lesquels le premier classificateur avait la performance d'un tirage aléatoire et le troisième comme des exemples pour lesquels le premier et le second classificateur étaient en désaccord. AdaBoost généralise cette idée : à chaque étape, le calcul de la nouvelle pondération est mené de manière à ce que le nouvel ensemble d'apprentissage soit mal classé par la combinaison linéaire des classificateurs précédents. Plus précisément, après t étapes, le classificateur est défini par :

$$H(x) = signe\left(\sum_{r=1}^{Q} \alpha_r\, h_r\,(x)\right)$$

Ensuite, l'algorithme calcule une nouvelle distribution de probabilités D_t+1, qui affecte à chaque élément de St la probabilité p_{t+1} (xi), selon la formule donnée dans l'algorithme 3. Schapire et Singer [2.5.2, 2.5.3] ont montré le résultat suivant, qui prouve en effet que les poids sont réactualisés d'une manière bien particulière :

h_t Classerait S pondéré par D_t+1 comme le hasard.

AdaBoost est un algorithme d'optimisation

Rappelons que d'une manière générale, l'apprentissage par minimisation du risque empirique (ERM) consiste à trouver l'hypothèse (dans une certaine famille) qui minimise une fonction de perte sur l'ensemble d'apprentissage. Il est cependant souvent difficile d'utiliser directement

L'erreur apparente (ou risque empirique) à la fin de AdaBoost est égale à :

$\varepsilon_T = \frac{1}{m}\,|\{i :\, H(x_i) \neq u_i\}|$ En notant: $f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$, il a été montré qu'elle est bornée de la façon suivante :

$$\varepsilon_t \leq \frac{1}{m}\sum_{i=1}^{m} \exp\left(-u_i f(x_i)\right) = \sum_{t=1}^{T} Z_t$$

Nous n'avons pas encore explicité Z_t, le facteur de normalisation introduit dans l'algorthme. Il est facile de calculer que :

$$Z_t = \sum_{i=1}^{m} D_t\,(i) \exp\left(-\alpha_t u_i h_t(x_i)\right)$$

La borne sur l'erreur apparente suggère donc de minimiser cette erreur par un algorithme glouton, en choisissant à chaque itération α_t et h_t de façon à minimiser Z_t. Et on peut démontrer que c'est ce que fait AdaBoost, en choisissant α_t égale à $\frac{1}{2}\ln\frac{1-\varepsilon_t}{\varepsilon_t}$.

Cet algorithme minimise donc une expression qui borne supérieurement l'erreur apparente et qui vaut, rappelons le :

$$\frac{1}{m}\sum_{i=1}^{m} D_t\,(i) \exp\left(-\alpha_t u_i h_t(x_i)\right)$$

AdaBoost est (presque) un algorithme bayésien

Il est également possible de relier AdaBoost à une décision bayésienne.

Notant $P_f[u = +1|x]$ la probabilité qu'un vecteur x soit classé +1. On peut montrer qu'une borne estimation de cette probabilité s'écrit :

$$P_f[u = +1|x] = \frac{exp(f(x))}{exp(f(x)) + exp(-f(x))}$$

Avec $f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

On peut pousser la raisonnement plus loin et modifier l'algorithme AdaBoost pour qu'il produise exactement une valeur de probabilité d'appartenance aux deux classes. C'est ce qui a été fait dans l'algorithme LogiBoost [2.5.3].

Les propriétés de l'algorithme AdaBoost

a. Bornes sur l'erreur apparente et sur l'erreur réelle

Commençons l'analyse de l'erreur apparente d'AdaBoost. Écrivons cette erreur ε_t de h_t sous la forme :

½ − γ_t, où $\gamma_t$mesure l'amélioration apportée par l'hypothèse ht par rapport à l'erreur de base 1/2. Freund et Shapire ont montré dans [2.5.4] que l'erreur apparente ε_T de l'hypothèse finale H est bornée par :

$$\prod_{t=1}^{T}\left[2\sqrt{\varepsilon_t(1-\varepsilon_t)}\right] = \prod_{t=1}^{T}\sqrt{1-4\gamma_t^2} \leq \exp(-2\sum_t \gamma_t^2)$$

Ainsi, si chaque hypothèse faible est légèrement meilleure que le hasard, ($\gamma_t \geq \gamma > 0$), alors l'erreur apparente diminue exponentiellement avec t. L'erreur en généralisation de l'hypothèse finale H peut être bornée par une expression faisant intervenir son erreur apparente ε_T , le nombre d'exemples d'apprentissage m, la dimension de Vapnik-Chervonenkis $d_{\mathcal{H}}$ de l'espace d'hypothèses :

$$R_{Réel}(H) = \varepsilon_T + O\left(\sqrt{\frac{T \cdot d_{\mathcal{H}}}{m}}\right.$$

Cette borne suggère que le boosting devrait tendre à sur-apprendre lorsque T devient grand, puisque le deuxième terme devient grand. Si cela arrive effectivement parfois, il a été observé

empiriquement que souvent cela ne se produit pas. De fait, il apparaît même fréquemment que le risque réel tend à diminuer même longtemps après que le risque empirique soit devenu stable, voire nul. Cette observation a priori énigmatique s'éclaircit si l'on établit un lien entre le boosting et les méthodes à vaste marge.

b. Adaboost et les marges

À la fin de AdaBoost, la marge d'un exemple (x, u), avec $u = \pm 1$ désignant la classe, s'exprime par :

$$marge\ (X,u) = \frac{u \sum_{t=1}^{T} \alpha_t h_t(x)}{\sum_{t=1}^{T} \alpha_t}$$

Ce nombre, sous sa forme normalisée donnée ici, est compris dans l'intervalle [−1, +1] et est positif seulement si H classifie correctement l'exemple. Nous savons que la marge peut être interprétée comme une mesure de confiance dans la prédiction. Il a été prouvé que l'erreur en généralisation peut être bornée avec une grande probabilité, pour tout $\theta > 0$, par :

$$R_{Réel} \leq P\,[marge\ (x,u) \leq \theta] + O\left(\sqrt{\frac{d\mu}{m\,\theta^2}}\right)$$

On note que cette borne est maintenant indépendante de T, le nombre d'étapes de boosting. De plus, il a été montré que le boosting cherche effectivement à augmenter la marge sur les exemples ; la raison en est qu'il se concentre sur les exemples difficiles à classer, c'est-à-dire sur les exemples dont la marge est la plus faible. Le fait que le risque réel tend à diminuer même longtemps après que le risque empirique soit devenu stable signifie que les marges continuent à croître, même si le classement de l'ensemble d'apprentissage reste inchangé.

c. AdaBoost est (presque) un SVM

AdaBoost produit un classificateur qui s'écrit

$$H(x) = \text{sign}\left(\sum_{t=1}^{t=T} \alpha_t \ \ h_t(x)\right)$$

Il est intéressant d'interpréter ce classificateur comme un hyperplan en dimension T, dont les paramètres sont le vecteur des valeurs α_t , pour **t** de **1** à **T**. Les **T** axes de ce nouvel espace de représentation sont calculés par la transformation de X par les T fonctions h_t . Sous cet angle, AdaBoost ressemble à un classificateur SVM. Il est connu que les SVM calculaient un hyperplan optimal dans un espace transformé (de grande dimension en général). Le type de transformation n'est pas le même, certes, mais l'hyperplan calculé par AdaBoost a-t'il quelque chose à voir avec le séparateur SVM? Comme on l'a vu au paragraphe précédent, AdaBoost est, comme les SVM, un algorithme de maximisation des marges. Plus précisément, en notant h le vecteur de dimension T composé des hypothèses (h_1 , . . . , h_t)

et $\boldsymbol{\alpha}$ le vecteur transposé de dimension **T** composé des valeurs (α_1 , . . . , α_t), on peut établir AdaBoost cherche la quantité :

$$\min_{\alpha} \max_{i} \frac{(\alpha \cdot h(x_i))u_i}{\|\alpha\|_1 . \|(h)(x)_i\|_\infty}$$

Avec $\|\alpha\|_1 = \sum_{i=1}^{T} |\alpha_i|$ et $\|(h)(x)_i\|_\infty = max_{t=1}^{T} |(h)(x)_i|$

La quantité recherchée par un SVM peut s'écrire de manière analogue :

$$\min_{\alpha} \max_{i} \frac{(\alpha \cdot h(x_i))u_i}{\|\alpha\|_2 . \|(h)(x)_i\|_2}$$

d. Généralisation à une classe, à plus de deux classes et à la régression

Quand toutes les données d'apprentissage ont la même classe, l'apprentissage se ramène à la classification non supervisée, qui peut s'aborder aussi comme un problème d'estimation de densité de probabilité à partir d'observations. Une approche consiste à estimer par séparation les quantiles de la distribution multidimensionnelle cherchée. Dans ce cas, on cherche dans une certaine famille une surface séparatrice H_μ telle que la probabilité d'appartenir à la classe unique soit supérieure à **μ** d'un côté de la surface et inférieure de l'autre côté. Pour AdaBoost, on se place du point de vue de l'espace de dimension **T** et on y cherche un hyperplan,

Quand on pose formellement le problème, on peut le ramener à un problème d'optimisation sous contraintes. Il est également possible d'étendre AdaBoost à l'apprentissage d'une règle de classification pour un nombre quelconque C de classes. Plusieurs solutions ont été exprimées. L'une des plus récentes propose de remplacer le calcul :

$$\alpha_t \leftarrow \frac{1}{2} \ln \frac{1-\varepsilon_t}{\varepsilon_t}$$

Opéré dans l'algorithme 3 par l'affectation suivante :

$$\alpha_t \leftarrow \frac{1}{2} \ln \frac{1-\varepsilon_t}{\varepsilon_t} + \log(C-1)$$

L'introduction de ce terme n'est pas qu'un artefact, mais a effectivement une signification en termes d'optimisation.

L'algorithme AdaBoost peut également être utilisé pour la régression : à partir d'une méthode de régression de base, on construit itérativement une combinaison pondérée d'occurrences de cette méthode en modifiant l'échantillon d'apprentissage. Une bibliographie est donnée dans [2.6.5].

L'utilisation du boosting

Le boosting, et particulièrement l'algorithme AdaBoost, a été employé avec succès avec de nombreux algorithmes d'apprentissage « faibles ». On l'utilise souvent par exemple avec des arbres de décision à profondeur volontairement limitée (dans le cas extrême, les decision stumps, ou « souches de décision » ne font qu'un seul test pour choisir une classe). On utilise aussi par exemple C4.5, un système d'apprentissage d'arbre de décision [2.5.5] ou Ripper, un système d'apprentissage de règles). Le boosting a été testé sur des domaines d'application variés. En général, l'utilisation du boosting a pour résultat d'améliorer souvent sensiblement les performances en apprentissage.

Les avantages du boosting et d'AdaBoost en particulier sont qu'il s'agit d'une méthode facile à programmer et aisée d'emploi. Elle ne nécessite pas de connaissance a priori sur l'algorithme d'apprentissage « faible » utilisé, et elle peut s'appliquer de fait à n'importe quel algorithme d'apprentissage faible. Les seuls paramètres à régler sont la taille de l'ensemble d'apprentissage **m** et le nombre total d'étapes **T**, qui peuvent être fixés par l'utilisation d'un ensemble de validation. De plus, des garanties théoriques sur l'erreur en généralisation permettent de contrôler l'apprentissage. Une autre propriété intéressante du boosting est qu'il tend à détecter les exemples aberrants (outliers) puisqu'il leur donne un poids exponentiellement grand en cours d'apprentissage. Cependant, la contrepartie de ce phénomène est que le boosting est sensible au bruit et ses performances peuvent être grandement affectées lorsque de nombreux exemples sont bruités. Récemment des algorithmes ont été proposés pour traiter ce problème (comme Gentle AdaBoost [2.5.6] ou BrownBoost [2.5.7]).

Nous avons vu au passage que l'adaptation aux problèmes multi-classes n'est pas immédiate, mais qu'elle a cependant fait l'objet d'études menant aussi à des algorithmes efficaces. De même, il existe des applications de boosting à la régression.

❖ **Boosting et théorie PAC**

Les premiers travaux sur le boosting sont historiquement issus du cadre de l'apprentissage PAC (probably approximately correct). Dans ce cadre, un algorithme PAC au sens fort (strong PAC) est défini ainsi :

- Pour toute distribution de probabilité D_x × sur l'espace des exemples *(x, u).*
- $\forall \varepsilon > 0, \delta$:
- Etant donné un nombre polynomial (fonction de 1 ,et 1) d'exemples suivant D_x ×.
- L'algorithme trouve une hypothèse d'erreur avec une probabilité ≥ 1 -

Les algorithmes d'apprentissage dits *faibles* ont une définition analogue, mais on leur demande seulement de trouver une hypothèse d'erreur $\varepsilon \geq \frac{1}{2} - \gamma$ avec γ strictement positif, donc éventuellement juste un peu meilleure que le hasard, en supposant une tâche de classification binaire avec la même proportion d'exemples positifs et négatifs.

La question qui est à l'origine du boosting est la suivante : « est-il est possible d'utiliser un algorithme faible pour obtenir un apprentissage de type fort ? » Shapire [2.5.8] a prouvé

que la réponse à cette question est positive et a conçu le premier algorithme de boosting par sous-ensembles. Freund [2.5.7] a ensuite produit un algorithme beaucoup plus efficace, également optimal, mais difficile à appliquer. En 1995, Freund et Shapire [2.5.4] ont proposé l'algorithme AdaBoost, efficace et pratique, qui est désormais la technique la plus employée pour améliorer les performances de n'importe quel algorithme d'apprentissage supervisé.
Dans le même temps, d'autres chercheurs ont analysé comment il est possible d'identifier les bons experts au sein d'une grande collection d'experts ou bien les bons attributs quand on a un grand nombre d'attributs (ces deux problèmes sont reliés). Les algorithmes développés, tels que Winnow [2.5.9] ont révélé l'intérêt dans ces problèmes de la mise à jour multiplicative des pondérations d'experts, comme le réalise le boosting pour la classification.

2.5 Conclusion

Nous venons de voir les différentes approches de classification que nous utiliserons par la suite. Ces approches ont fait leur preuve dans un certain nombre de domaine, que ce soit les réseaux de neurones, les arbres de décision ou les C-Means… Cependant, nous supposons que les C-Means ainsi que les fuzzy C-means ne seront pas d'une grande aide pour notre problème de séparation texte – graphique, sans connaissances au préalable. Nous tentons tout de même de les expérimenter car il représente l'une des approches les plus puissantes du clestering et nous ne pouvons être sûrs sans la mise en pratique.

Chapitre III : Conception

Ce chapitre se concentre sur la présentation détaillée des différentes étapes de la conception de notre système de classification. La première section concerne l'étape de l'acquisition et du prétraitement des images. Nous aborderons par la suite, les méthodes qui composent la partie segmentation telles que la morphologie mathématique, la recherche de composantes connexes…etc. Puis, nous discuterons de l'extraction de caractéristiques où seront utilisés les descripteurs. Dans la quatrième section, nous aborderons les différents classifieurs utilisés pour notre expérimentation, ainsi qu'une brève description des approches sur lesquelles ils se basent. Nous terminerons par présenter la discrimination texte – graphique afin de mieux expliquer le cadre de notre travail.

3.1. Introduction

La reconnaissance des formes est un domaine en plein essor depuis ces dix dernières années, il englobe un grand nombre de domaine d'application, tels que : l'industrie : évaluation de qualité sur une chaine de production, la télédétection, le multimédia : indexation, compression, la télésurveillance, l'imagerie satellitaire…etc. Le but principal de la reconnaissance des formes est de pouvoir distinguer un objet (texte, forme particulière, graphique, visage…) parmi un certain nombre d'autres objets.

L'objectif de notre travail est donc d'arriver à effectuer une séparation entre un texte en arabe et un graphique à partir d'une image donnée. Cette image peut contenir soit du texte (mot, sous mot, phrase, ou bloc de texte), soit du graphique (photo, illustration …). Ce qui suppose, qu'à partir d'une image donnée, nous pouvons dire si elle contient du texte ou si elle est composée de graphique. Afin d'y parvenir, nous débutons par la phase de construction de plusieurs classifieurs suivie de celle d'un apprentissage à partir d'images exemples pour ensuite pouvoir effectuer la séparation en texte et graphique.

Les classifieurs sont basés sur des approches différentes qui sont : les réseaux de neurones artificiels (ANN, Artificial Neural Network), CMEAN, Fuzzy CMEAN et AdaBoost avec les arbres de décisions. Nous utiliserons trois bases de données différentes, une base de données d'images contenant des objets textuels et graphiques séparés, une base de données à générer automatiquement par notre algorithme de segmentations, et une base de données contenant la fusion des deux bases de données citées. Nous utiliserons également un ensemble de descripteurs afin de trouver le ou les meilleurs descripteurs pour effectuer la classification.

3.2. Acquisition et prétraitement

L’acquisition de données est une étape nécessaire dans tout processus de classification, il n’est pas toujours évident de trouver l’ensemble de données adéquat ou de collecter des données pertinentes qui permettent d’atteindre l’objectif voulu. Pour notre expérimentation, nous avons acquis trois ensembles de données, le premier est un ensemble de données étiqueté (texte ou graphique), le second est un ensemble de données généré par un algorithme qui prend une image en entrée et qui renvoie comme résultat une séparation des objets, le troisième consiste en la fusion des ces deux ensembles de données.
Afin d’exploiter ces ensembles de données, un prétraitement est indispensable, il consiste à rendre les données en notre possession exploitables par nos systèmes de classification. L’opération que nous utiliserons pour le prétraitement consiste en une binarisation des données.

La binarisation est une opération qui produit deux classes de pixels qui sont en général représentés par des pixels noirs et des pixels blancs. La dynamique de l'image est alors réduite à deux luminosités. Elle permet d’identifier les points qui appartiennent à l’information véhiculée. Il existe deux type de binarisation, globe et locale.

3.2.1. Binarisation globale

La binarisation globale est possible lorsque les documents ne contiennent pas trop de bruit et qu’ils présentent un contraste convenable entre les données et le fond du document. Son principe est qu’après avoir déterminé une nuance seuil, tous les points de l’image numérisée sont parcourus. Les pixels appartenant au fond du document sont identifiés comme étant ceux ayant une nuance plus claire que la nuance seuil, et les autres comme étant l’information véhiculée par le document.

3.2.2. Binarisation locale

La binarisation globale n’apporte pas souvent des résultats satisfaisants. C’est notamment le cas lorsque l’image initiale est extrêmement bruitée. D’autres types de méthodes de binarisation basées sur une étude des propriétés locales de l’image produisent en général de meilleurs résultats. Ces méthodes sont qualifiées de binarisations locales ou encore de binarisation adaptatives. C’est le cas de la méthode proposée par Trier et Taxt [5.2.1], basée sur un seuillage local adaptatif, ainsi que le seuillage à Hystérésis [5.2.2] et celui de Niblack [5.2.3].

3.3. Segmentation

Une fois l’image binaire obtenue, l’information pertinente contenue correspond aux pixels noirs de cette image. On peut considérer que cette information correspond à la superposition d’un certain nombre de calques (*couches logiques*), chacun des calques contenant une partie de l’information globale.

L’étape de la segmentation est donc une étape importante et qui comporte elle-même d’autres étapes : la morphologie mathématique dont nous n’exploiterons que la méthode de dilatation et de l’érosion, la squelettisation qui utilise la méthode de la transformation de distance et la recherche de composantes connexes que nous allons présenter ci-dessous.

Pour la partie segmentation, nous avons élaboré nous même un algorithme qui permet une segmentation texte – graphique à partir d’une image contenant les deux. Cet algorithme comporte plusieurs étapes qui sont :

- Dilatation : dans cette étape, nous avons pris des images contenant soit du texte, soit du graphique, soit les deux ensembles, nous avons par la suite appliqué l’opération de la dilatation afin de ne pas avoir une séparation de sous-mots dans un même mot et afin de prendre également en considération au minimum un mot en entier et de ne pas perdre les points de ponctuations. Cette dilatation à un noyau qui a une largeur double à celle de sa longueur et qui devrait être adapté selon le type de texte.

- Erosion : cette étape a pour objectif d’atténuer légèrement les effets obtenus verticalement de la dilatation afin de séparer les lignes, les paragraphes, ou le graphique du texte.

- Transformé de distance et binarisation : dans ces étapes, nous nous n’effectuons que des calculs de la transformé puis une binarisation sur les résultats obtenus après le calcul.

- Recherche de composantes connexes : dans cette étape, nous effectuons soit la représentation du texte (c'est-à-dire un mot, une phrase ou un paragraphe) soit une représentation de graphique.

Dans ce qui suit, nous aborderons les différentes opérations et méthodes utilisées pour l’algorithme de décomposition texte/graphique plus en détails afin de comprendre leur concept.

3.3.1. Morphologie mathématique

La morphologie mathématique est un ensemble de méthodes qui offre un grand nombre d'outils très puissants de traitement et d'analyse d'images. Le principe de base de la morphologie mathématique est de comparer l'image à analyser par rapport à un ensemble de géométrie connue appelé élément structurant que l'on déplace de façon à ce que son origine passe par toutes les positions de l'image, pour mettre en évidence certaines de ses caractéristiques. Pour chacune des positions de x, nous nous posons une question relative à l'union ou à l'intersection de l'élément structurant avec les objets de l'image. L'ensemble des points correspondant à une réponse positive permet de construire une nouvelle image qui constitue l'image résultat.

L'érosion et la dilatation sont les opérateurs de base de la morphologie construite à partir de ces principes.

3.3.1.1. Dilatation

La dilatation est l'opération duale (ou inverse) de l'érosion. La transformation par dilatation se définit de manière analogue. Elle a été introduite par H.Minkowski [5.2.4] et développée par G Matheron [5.2.5] et J. Serra [5.2.6] . En prenant le même élément structurant, on pose pour chaque point x$\in \boldsymbol{R^2}$, la question : B_x touche-t-il l'ensemble **X** ?

L'ensemble des réponses positives forme un nouvel ensemble *appelé dilaté* de **X** *par* **B** :

$$\delta^B(X) = \{x : B_x \Uparrow X\} = \{x : B_x \cap X \neq \emptyset\}$$

La figure 3.1 montre l'ensemble initial et la figure 3.2 l'ensemble dilaté par un carré de taille 10. Le principe de la dilatation est illustré par la figure 3.3.

Figure 3.1. Ensemble X1

Figure 3.2. $\delta^B(X) = \{x : B_x \Uparrow X\} = \{x : B_x \cap X \neq \emptyset\}$

Lors d'une dilatation:

- Tous les objets vont "grossir" d'une partie correspondant à la taille de l'élément structurant,

- S'il existe des trous dans les objets, c'est à dire des "morceaux" de fond à l'intérieur des objets, ils seront comblés,
- Si des objets sont situés à une distance moins grande que la taille de l'élément structurant, ils vont fusionner.

3.3.1.2. Erosion

Son origine remonte à Hadwiger [5.2.7]; ce concept a été repris par Matheron [5.2.5] puis développé par Serra [5.2.6]. Pour définir la transformation par érosion dans un cadre ensembliste, considérons un ensemble **X**$\in \boldsymbol{R^2}$, (figure 2a). Soit **B** un élément structurant carré.

Pour chaque position x$\in \boldsymbol{R^2}$, on pose la question : est-ce que B_x centré en **x** est inclus dans **X**?

Les réponses positives forment un nouvel ensemble, (figure 3.3) *appelé érodé de* **X** *par* **B** :

$$\varepsilon^B(\mathrm{X}) = \{\mathrm{x} : B_x \subset X\}$$

Figure 3.3. Ensemble X1

Figure 3.4. $\varepsilon^B(\mathrm{X}) = \{\mathrm{x} : B_x \subset X\}$

L'érosion est anti extensive. De plus lors de cette transformation :

- Les objets de taille inférieure à celle de l'élément structurant vont disparaître,
- Les autres seront "amputés" d'une partie correspondant à la taille de l'élément structurant,
- S'il existe des trous dans les objets, c'est à dire des "morceaux" de fond à l'intérieur des objets, ils seront accentués,
- les objets reliés entre eux vont être séparés.

Une érosion de taille **n** peut se réaliser en répétant une érosion **n** fois avec un élément structurant de taille 1 ou en appliquant une seule érosion avec un élément structurant de taille **n**.

3.3.1.3. La transformée de distance

La transformée en distance (***distance transform***) d'une image binaire, parfois appelée fonction distance, permet de connaître la distance entre un pixel donné et le pixel le plus proche de l'image. De nombreuses méthodes exploitent cette transformée pour accélérer des calculs. Par exemple dans la localisation de motifs dans une image par ***Chamfer Matching*** ou pour l'obtention d'une distance de Hausdorff entre deux images, les calculs peuvent être considérablement accélérés. En effet le calcul de la transformée peut-être réalisé très rapidement, avec une très bonne approximation, en propageant un petit masque sur deux parcours de l'image.

La transformée de distance 3-4 est la base de la transformée de distance. Cette méthode permet de mieux traiter les points de jonction et de limiter le nombre de barbules et autres artefacts. Intuitivement, son principe est de considérer que le squelette de l'image est déterminé par le centre des cercles maximaux inscrits dans les traits de l'image.
L'implémentation est réalisée de la manière suivante :

1) Calcul de la transformée de distance. Chaque point noir d'une image binaire est étiqueté par la distance entre ce point et le point blanc le plus proche. Pour approcher au mieux la métrique euclidienne tout en gardant des temps de calcul faibles, des métriques de type *chanfrein* sont habituellement utilisées, comme par exemple la métrique dite 3–4 [5.2.8].

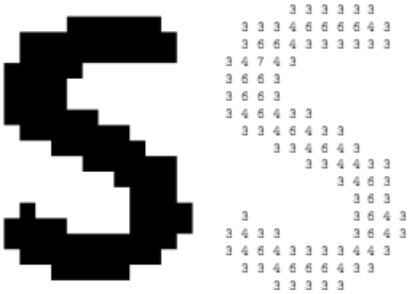


Figure 3.5. Image d'un caractère et sa TD 3-4

V_2	V_3	V_4
V_1	X	V_5
V_8	V_7	V_6

Figure 3.6. Voisinage utilisé pour le calcul de la TD3-4

2) Détection des maxima locaux. À partir de la transformée de distance, les pixels appartenant au squelette sont détectés en se basant sur un algorithme proposé par Arcelli et Sanniti di Baja [Arc 85, Arc 89].

3) Suivi du squelette. Il s'agit ensuite de prolonger le squelette en reliant les points ainsi marqués.

4) Réduction à l'épaisseur 1 sans altérer la connectivité du squelette, ni raccourcir ses branches.

5) Ébarbulage selon la technique proposée par Sanitti di Baja [5.2.8] .

Afin de gérer l'épaisseur initiale des traits, nous nous basons sur le squelette, étiqueté par la valeur de la transformée de distance. Cette valeur est multipliée par 2/3 afin d'obtenir l'épaisseur finale, la transformée de distance 3–4 ne correspondant pas à la distance euclidienne.

3.3.2. Etiquetage de composantes connexes

L'extraction des composantes connexes est largement utilisée en reconnaissance de formes pour étiqueter les régions détectées par segmentation. La technique consiste à regrouper les pixels voisins dans un ensemble appelé composante connexe. Chaque ensemble est disjoint des autres et peut ensuite être aisément isolé. Une composante connexe est un ensemble de pixels connexes entre eux. Par exemple, pour deux pixels appartenant à la composante connexe, il est possible de définir un chemin à l'intérieur de celle-ci.

Exemple de composantes connexes :

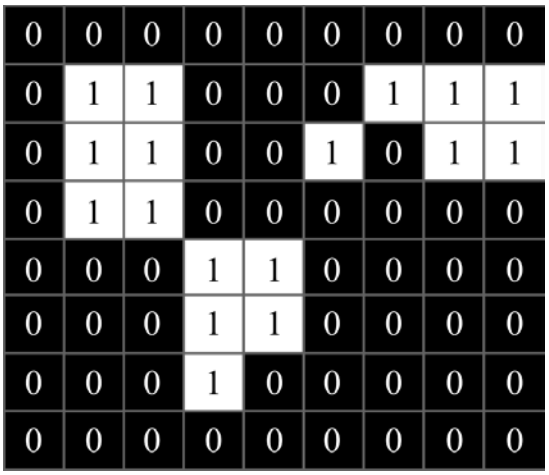


Figure 3.7. Image binaire

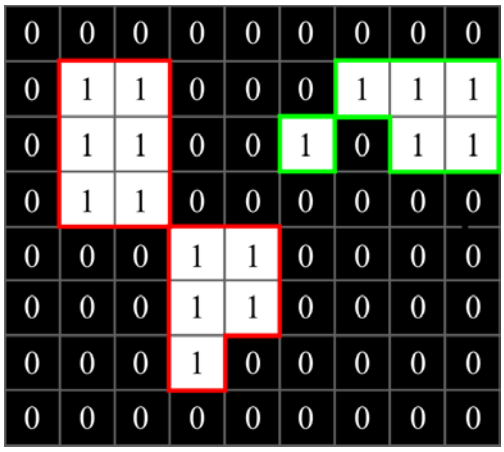


Figure 3.8. Composantes 4-connexes

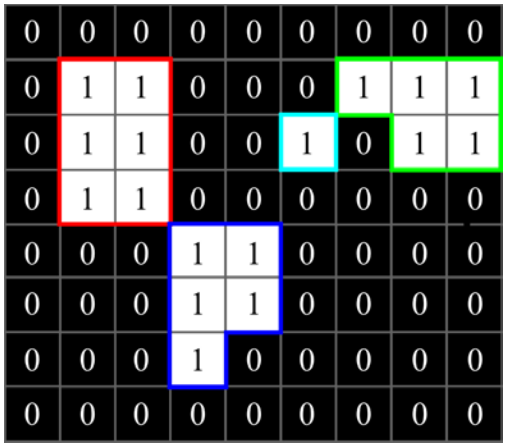


Figure 3.9. Composantes 8-connexes

Il existe deux approches pour effectuer l'étiquetage de composantes connexes qui sont l'approche par remplissage (flood fill) et l'approche par double parcourt.

3.4. Extraction des caractéristiques

En reconnaissance des formes, les caractéristiques (ou features en anglais) sont les propriétés mesurables d'un phénomène physique observé. L'extraction de caractéristiques discriminantes est une étape fondamentale du processus de reconnaissance, préalable à la classification. Les caractéristiques sont généralement numériques mais il peut s'agir de chaînes de caractères, de graphes ou d'autre quantité.

L'extraction de caractéristiques consiste à trouver le moyen de définir chaque forme (texte ou graphique dans notre cas) de telle sorte à pouvoir les classer en se basant sur leur définition ou plus exactement sur leur description, ce qui nous oriente vers l'utilisation de descripteurs. Dans tout processus de reconnaissance, la tâche qui s'avère la plus cruciale est celle de trouver un bon descripteur. Dans notre travail, il s'agit de trouver un descripteur pour la langue arabe ce qui rend cette tâche encore plus ardue. L'écriture arabe s'écrit de droite à

gauche. Elle est cursive, c'est-à-dire que les lettres sont liées généralement entre elles. Chaque caractère peut prendre quatre formes différentes, suivant sa position dans le mot. Un ensemble de pixels noirs adjacents les uns aux autres est appelé une composante connexe. Cette dernière, dans l'écriture arabe, ne représente pas forcément un mot entier, elle peut être seulement une partie du mot, car certains caractères ne doivent pas être attachés à leur successeur à gauche dans le mot. Par ailleurs, il existe des lettres différentes qui ont la même forme, mais qui se distinguent par la position et le nombre de points qui leur appartiennent. Les voyelles "a", "i" et "ou" ne sont pas utilisées systématiquement dans l'écriture arabe; des signes qui correspondent à des voyelles sont employés pour éviter des erreurs de prononciation.

Nous avons donc choisi d'implémenter un grand ensemble descripteurs différents afin de maximiser nos chances de trouver le meilleur ou la meilleure combinaison de descripteurs pour discriminer un texte et un graphique.

Comme mentionné dans le chapitre I, il y a différents types de descripteurs, de texture, de régions, de descripteurs contours ou de points d'intérêts. Nous avons choisi de tous les utiliser, ils vont du plus simple au plus complexe, excepté la représentation par point d'intérêt qui ne correspond pas à ce que nous recherchons (elle est surtout utilisée pour rapprocher deux formes), car nous désirons qu'un mot, un sous mot, une phrase, une ligne et un paragraphe soient classés dans la même catégorie (texte).

L'utilisation de ces descripteurs n'est pas évidente du fait que nous ignorons lequel de ces 87 descripteurs correspond ou décrit le mieux nos images afin de nous fournir une classification correcte. La solution à ce problème est de tester chaque descripteur, puis une combinaison d'un certain nombre de descripteurs pour chaque image (texte ou graphique). L'image sur laquelle nous calculons les descripteurs peut être l'image originale, l'image de contour, l'image de la projection horizontale, l'image de la projection verticale, l'image des deux projections ensemble, le squelette de l'image ou encore le log polar de l'image.

Nous allons dans ce qui suit donner une brève définition des types d'image que nous avons utilisés et cités précédemment.

Contour : un contour est une frontière entre deux objets dans une image. Mais plus largement, c'est une discontinuité de l'image (une variation brusque d'intensité)

Projection horizontale d'une image : la projection de l'image est faite sur un plan parallèle à l'horizon.

Projection horizontale du contour de l'image : la projection de l'image de contour est faite sur un plan parallèle à l'horizon.

Projection verticale d'une image : Le déplacement vers le haut de l'image à 90 ° suivie de la chute vers le bas de cette image.

Projection verticale du contour de l'image : Le déplacement vers le haut de l'image de contour à 90 ° suivie de la chute vers le bas de cette image.

Squelettisation : Le but de la *squelettisation* est de réduire les traits présents dans une image binaire à une ligne d'une épaisseur d'un pixel, appelé *squelette*. Cela permet d'extraire l'essentiel de l'information décrite par l'image : sa structure. À partir de cette nouvelle description de l'image, plus compacte et ainsi plus expressive sémantiquement, le calcul des vecteurs décrivant l'image se trouve facilité. Toutefois, si le squelette obtenu permet de traiter l'image plus facilement, il est impératif qu'il ne dénature pas l'information présente dans l'image initiale. Le calcul du squelette doit donc garantir que les propriétés topologiques et morphologiques de l'image initiale, telles que la connexité, les jonctions et les dimensions, seront préservées à l'issu du traitement.

3.5. Classifieurs

Nous venons de voir les différentes étapes qui précédent la classification, l'acquisition et le prétraitement, la segmentation et l'extraction de caractéristiques. Nous allons aborder maintenant la partie finale et la plus importante de notre travail qui est la classification.

Nous avons choisi d'utiliser quatre types de classifieur afin de comparer leurs performances et de trouver celui qui fournit les résultats les plus probants pour la résolution de notre problème, qui consiste en la classification d'images en texte arabe ou en graphique. Les quatre approches sur lesquelles se basent nos classifieurs sont les suivantes :

- ***Les réseaux de neurones artificiels (ANN)***
- ***Les C – voisins (C-MEAN)***
- ***Les C – voisins flous (Fuzzy C-MEAN)***
- ***AdaBoost avec les arbres de décisions***

Le choix de ces quatre approches se justifie par le fait qu'ils représentent de bons paradigmes pour la classification. Ils nous permettent de réaliser une comparaison entre une classification avec apprentissage supervisé lors de l'utilisation des classifieurs basé sur les réseaux de neurones artificiels et AdaBoost avec les arbres de décisions et le clustering lors de l'utilisation de C-MEAN et fuzzy C-MEAN.

AdaBoost a été utilisé avec les arbres de décision car nous supposons que ces derniers produiront de bons résultats de classification puisqu'ils emploient un apprentissage supervisé, alors que les C-MEAN et Fuzzy C-MEAN ne fourniront pas une classification correcte vu qu'ils ne peuvent faire de distinction entre le texte et le graphique. Ceci est dû au fait qu'ils se basent sur le clustering. Les classes résultantes seront forcément pêle-mêle car nous ne pouvons dire que la classe 1 représentera du texte et que la classe 2 représentera du graphique. Par conséquent, nous ne pouvons booster les classifieurs basés sur ces deux approches avec AdaBoost.

Dans la phase de test, chaque descripteur a été testé avec chacun des quatre classifieurs. Nous avons aussi testé différentes combinaisons de descripteurs pour chaque classifieur.

Pour chaque descripteur (ou combinaisons de descripteurs) avec un classifieur donné, nous avons effectué plusieurs fois l'apprentissage car le choix et l'ordre des exemples d'apprentissage choisis dans la base de données, ont été établis de manière aléatoire, pour ensuite faire la somme du pourcentage de réussite de ce descripteur.

Le choix et l'ordre des exemples d'apprentissage influence l'apprentissage, donc nous avons voulu voir les résultats de notre classification sans prendre trop en considération ce paramètre et essayer d'obtenir des résultats généraux.

3.6 Apprentissage et test

Cette sous-section concerne les phases d'apprentissage et de test de nos classifieurs, qui représentent des phases indispensables et très importantes. Nous allons décrire les bases de données utilisées vu que nous en avons plus d'une. Nous allons également décrire les variantes des systèmes de classifications que nous avons construits.

3.6.1 Les bases de données

L'apparition de bases publiques de tailles conséquentes comme la base IFN/ENIT [6.1.1] (24659 images, pour un vocabulaire de 937 de noms de villes tunisiennes, et 411 scripteurs), et l'organisation de compétitions en 2005 [6.1.2] et en 2007 [6.1.3] dans le cadre de la conférence ICDAR, ont rendu possibles les comparaisons entre systèmes et ont permis une progression rapide du domaine au cours des dernières années. Aujourd'hui, les performances obtenues par les meilleurs systèmes de reconnaissance de l'écriture manuscrite arabe semblent

proches de celles qu'on pourrait attendre sur du latin pour une taille de vocabulaire équivalente.

Pour notre expérimentation, nous avons utilisés trois bases de données différentes afin d'effectuer divers tests et comparaisons de nos quatre classifieurs.

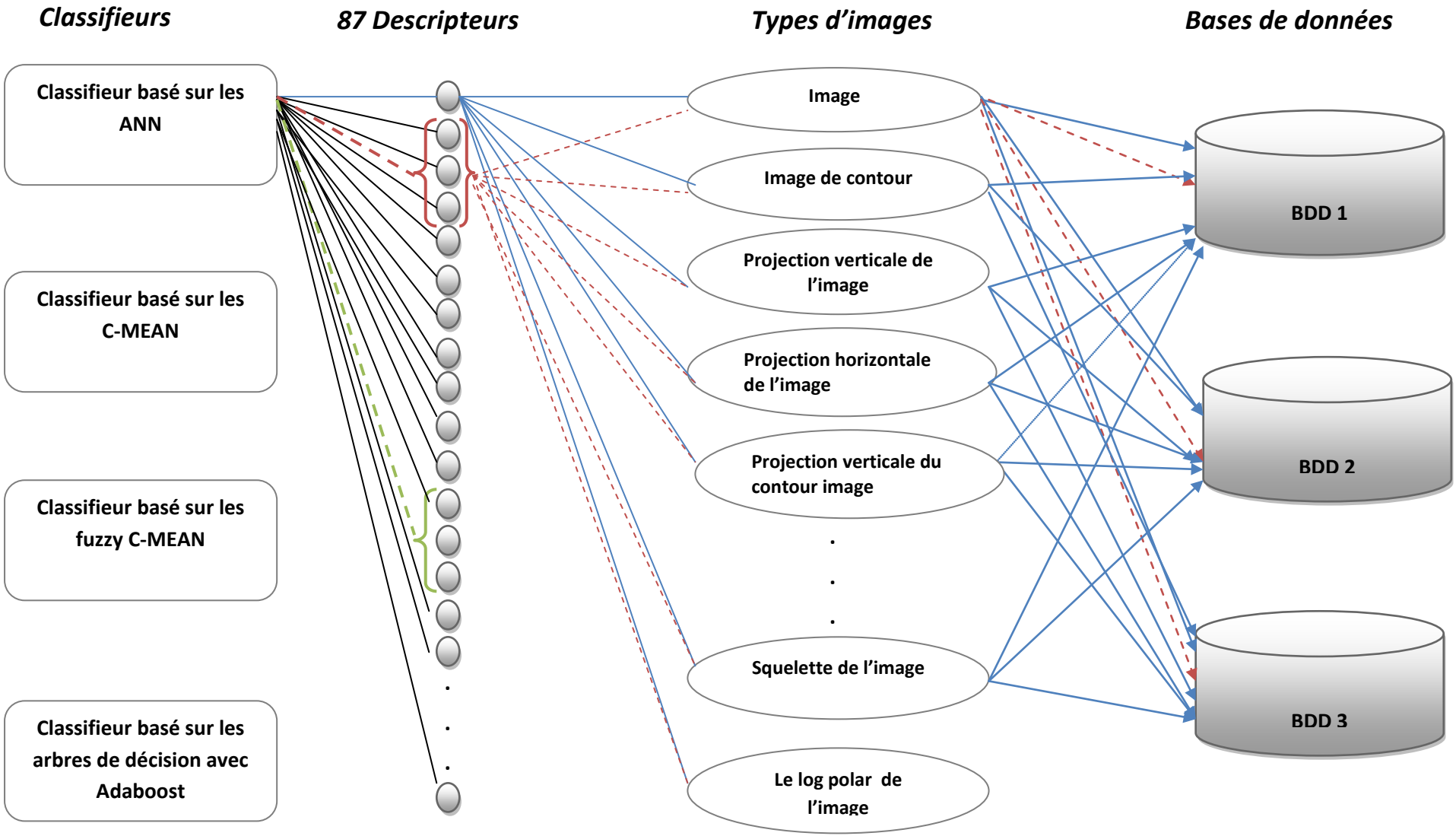
Les trois bases de données choisies sont les suivantes :

i. La première base de données est une base standard contenant des images classées en texte récupérée à partir de la base de données IFN/ENIT (c'est une base de référence qui contient du texte arabe) et des images en graphique récupérées sur internet [6.1.13].
ii. La seconde base de données a été générée par notre algorithme de décomposition texte / graphique (section 3.3) qui prend une image et effectue la séparation entre le texte et les graphiques qui composent cette image, nous obtenons une base classée également en texte et en graphique séparés.
iii. La troisième base de données constitue une fusion des deux bases que nous venons de décrire.

3.6.2 Les expérimentations

Les expérimentations s'effectueront comme suit ; étant donné un classifieur, nous prenons un type d'image précis (contour, projection verticale, projection horizontale, …) avec un descripteur ou une combinaison de descripteurs en utilisant une des bases de données pour effectuer l'apprentissage et le test. Ce processus se répétera jusqu'à ce que nous ayons effectué l'apprentissage et les tests sur toutes les combinaisons possibles entre classifieurs, type d'images, descripteurs et bases de données (voir le schéma ci-dessous). Ceci constitue un travail astronomique au vu du nombre de descripteurs (87) employés. L'objectif étant de découvrir lequel des ces quatre classifieurs fournit le résultat le plus intéressant.

Figure 3.10. Les phases d'apprentissage et de test sur les différentes combinaisons classifieur–descripteur(s)–type d'images–BDD



Ce schéma montre la complexité de notre travail, ces différentes combinaisons nous ont permis de trouver les meilleurs descripteurs ainsi que les meilleures combinaisons de descripteurs qui permettent le bon fonctionnement de nos quatre classifieurs. Il montre également la diversité de nos expérimentations, nous nous ne sommes pas arrêtés à un nombre restreint de classifieurs, ni à un type précis d'image, ni à une base de données spécifique. Ainsi, notre travail a englobé un certain nombre de variantes qui nous ont garanti un résultat de classification encourageant que nous aborderons dans le chapitre implémentation.

3.7 Conclusion

Ce chapitre conception a permis de passer en revue les étapes cruciales et préalables à la classification. A savoir:

1. L'acquisition et le prétraitement qui permettent de rendre les ensembles de données exploitables par nos classifieurs, c'est donc une étape que ne nous pouvions négliger.

2. La segmentation qui comporte elle-même d'autres étapes : la morphologie mathématique (méthode de dilatation et de l'érosion), la squelettisation (la transformée de distance) et la recherche de composantes. L'issue de cette étape est l'obtention d'images de texte et d'images de graphiques.

3. L'extraction de caractéristiques qui nous oriente vers l'utilisation de descripteurs afin de pouvoir catégoriser nos images en texte – graphique. Nous avons utilisé pour nos expérimentations plus de 100 descripteurs sur différentes types d'images (contour, projection verticale, projection horizontale…).

4. La classification réalisée grâce à l'emploi de quatre types de classifieurs basés sur des approches différentes (apprentissage supervisé et clustering) et l'utilisation de trois ensembles de données différents. ceci a permis de diversifier les expérimentations, en élaborant différentes combinaisons classifieur – descripteur – type d'image – base de données.

Le déroulement de ces expérimentations ainsi que leurs résultats seront abordés dans le chapitre suivant.

Chapitre IV : Implémentation

Ce chapitre concerne la partie implémentation de notre système de classification. Il consiste en la mise en œuvre de nos classifieurs en utilisant les différentes bases de données en notre possession. L'implémentation a été réalisée sous la plate forme Visual Studio 2008 avec Visual C++.Nous avons également eu recours à des librairies très connues comme OpenCV, ItiLib,…etc. Ce chapitre respecte la même structure que le chapitre Conception. Nous aborderons les même points essentiels qui sont ; l'acquisition et le prétraitement, la segmentation, la reconnaissance de caractéristiques et les classifieurs mais cette fois revus d'un point de vue implémentation. En outre, ce chapitre comporte une partie relative aux résultats fournis par nos classifieurs.

4.1. Introduction

Le développement de notre système a exigé l'utilisation d'un bon nombre d'algorithmes et de méthodes. Nous avons employé la morphologie mathématique (érosion, dilatation et squelettisation), la transformée de distance, la transformation d'images, la binarisation, la détection de composante connexe, 87 descripteurs ainsi que 4 types de classifieurs.

Par ailleurs, nous avons exploité un certain nombre de librairies disponibles (OpenCV, Qgar, itk, libFourier, ltiLib). Les algorithmes dont nous avons eu besoin ont dû être compilés manuellement pour qu'ils soient adaptés à notre configuration PC et qu'elles puissent fournir les meilleures performances lors de l'exécution. De plus, nous avons dû implémenter certains outils réalisant des tâches telles que la transformation de l'image ou la détection de composantes connexes.

4.2. Acquisition et prétraitement

Comme mentionné au niveau du chapitre 3, nous avons rassemblé trois bases de données utilisées pour la validation de la classification en texte – graphique. L'exploitation de ces données exige un prétraitement qui consiste en une binarisation.

4.3. Segmentation

Afin d'effectuer une décomposition texte – graphique, nous avons choisi des images (contenant soit du texte, soit du graphique soit les deux ensemble) et nous leur avons appliqué une dilatation avec un noyau rectangulaire de 8 pixels de large et de 4 pixels de hauteur, avec un centre au point (4,0) du noyau.

Une érosion est appliquée sur cette image dilatée avec un noyau en forme d'ellipse de 1 pixel de large et de 3 de hauteur avec un centre au point (0,1) du noyau.

Ensuite, la transformé de distance est calculée et une binarisation est effectuée avec un petit seuil. Nous avons utilisé la valeur de 5 après plusieurs tests.

Enfin, nous pouvons rechercher les composantes connexes qui représentent soit un texte- un mot, une phrase ou un paragraphe- soit un graphique.

Le noyau de la dilation et de l'érosion et le seuil de binarisation ont été obtenus à partir de différents tests sur les images qu'on possède. Ils sont dépendants les uns des autres et sont ajustables en fonction les caractéristiques de l'image.

4.4. Extraction de caractéristiques

Nous avons abordé l'extraction de caractéristiques au niveau de la conception dans le chapitre précédent. Dans cette section, nous allons l'aborder d'un point de vue programmation, nous allons implémenter les 87 descripteurs afin de les tester et voir lequel ou lesquels correspondent le mieux aux images de notre collection.

Les 87 descripteurs choisis ont été sélectionnés à partir de quatre librairies différentes et très largement utilisées dans le domaine de la reconnaissance de formes.

1. *Librairie ltiLib :*
2. *Librairie libFourier :*
3. *Librairie OpenCV :*
4. *librairie Itk :*

Nous avons donc exploité 48 descripteurs de la librairie ltiLib montrés par la table 4.1.

Descripteurs	**Dimensions**
Oriented gaussian derivatives based texture feature	40
qmf Energy of texture features	32
Curvature Feature	64
Orientation feature	36
Zernike moments magnitude of the coefficients feature	36
Zernike moments phase of the coefficients feature:	36

Zernike moments real part feature	36
Zernike moments imaginary part feature	36
MPEG-7 magnitude of the coefficients feature	36
MPEG-7 phase of the coefficients feature	36
MPEG-7 real part feature	36
MPEG-7 imaginary part feature	36
Radial Orthogonal magnitude of the coefficients feature	36
Radial Orthogonal phase of the coefficients feature:	36
Radial Orthogonal real part feature	36
Radial Orthogonal imaginary part feature	36
Area size	1
Border size	1
X-component of COG	1
Y-component of COG	1
Smallest x coordinate	1
Largest x coordinate	1
Smallest y coordinate	1
Largest y coordinate	1
Central moment	7
Inertia parallel to main axis	1
Inertia orthogonal to main axis	1
Orientation	1
Eccentricity	1
Compactness	1
Minimum distance to border	1
Maximum distance to border	1
Mean distance to border	1
Leftmost distance to main axis	1
Rightmost distance to main axis	1
Frontmost distance to COG, along main axis	1
Rearmost distance to COG, along main axis	1
hu Moments	7

hu Moments More xcog	1
hu Moments More ycog	1
hu Moments More eigen1	1
hu Moments More eigen2	1
hu Moments More orientation	1
hu Moments More m00	1
Border Signature Area	32
Border Signature Distance	32
Curvature Scale Space maxsigma	1
Curvature Scale Space maxrow	1

Table 4.1. Les descripteurs utilisés de la librairie ltiLib et leurs dimensions.

.

Nous avons également utilisé 27 descripteurs de la librairie libFourier montrés par la table 4.2.

Descripteurs	**Dimensions**
Pixel count	1
Aspect ratio	1
Centroid row	1
Centroid col	1
Triangularity	1
Ellipticity	1
Ellipse major axis length	1
Ellipse minor axis length	1
Ellipse aspect ratio	1
Ellipse eccentricity	1
Ellipse orientation	1
Equivalent diameter	1
Maximum Diameter	1
Solidity	1

Elliptic variance	1
Circularity	1
Rectangularity	1
Radial distance features mean	1
Radial distance features stdev	1
Radial distance features entropy	1
Radial distance features roughness	1
Chord length statistics mean	1
Chord length statistics variance	1
Chord length statistics skewness	1
Chord length statistics kurtosis	1
Fourier Descriptor	60
Flusser's affine invariant moments	4

Table 4.2. Les descripteurs utilisés de la librairie libFourier et leurs dimensions

Nous n'avons implémenté que 10 descripteurs de la librairie OpenCV qui sont montrés par la table 4.3.

Descripteurs	**Dimensions**
CV Hough Lines	1
CV Hough Cercles	1
cvMoments	17
cvContourMoments	17
prNoirImage	1
prNoirImageContour	1
prNoirImage/prNoirImageContour	11
prNoirHull	1
PrNoirVertical	1
prNoirHorizontal	1

Table 4.3. . Les descripteurs utilisés de la librairie OpenCv et leurs dimensions

Et enfin nous n'avons utilisé que deux descripteurs de la librairie Itk qui sont :

Descripteurs	Dimensions
itk histogram Frequency	16
itk Entropy	1

Table 4.4. . Les descripteurs utilisés de la librairie OpenCv et leurs dimensions

Chacune de ces librairies possède sa propre représentation de la structure d'une image, des contours, …etc. Nous avons donc pris les structures d'OpenCV comme référence pour le chargement et la représentation de toutes les images afin de faciliter le processus de classification qui utilisent ces images, nous n'avons pas à les traiter à chaque fois que nous les utiliserons, cela représente un gain de temps considérable.

En ce qui concerne les descripteurs, leurs calculs se fera sur l'image(ou l'une des transformations de l'image), ils seront stockés dans un vecteur dynamique dont la taille dépendra du nombre de descripteurs utilisés et de leurs dimensions.

Le calcul de chaque descripteur est organisé sous forme de fonction qui prend en entré l'image (structure de données OpenCV) et un vecteur de réel pour les valeurs du descripteur, puis si c'est nécessaire, transformer la structure de l'image dans la structure native de la librairie utilisé pour le calcul du descripteur, et renvoyer en retour le vecteur en ajoutant les valeurs du descripteur calculé.

On peut donc calculer autant de descripteur que l'on veut pour une image et qui seront donc contenu dans notre vecteur de façon contigu.

Et c'est ce vecteur qui représentera notre image dans la phase suivante de la reconnaissance.

Nous avons utilisé AdaBoost avec l'algorithme REAL et avec 100 arbres de décisions.

Nous avons utilisé 2 couches cachées pour les réseaux de neurones contenant chacune 100 neurones, un neurone (soit texte, soit graphique). Les entrées dépendent de la dimension du descripteur.

La reconnaissance est précédée automatique de la phase d'apprentissage. Nous avons utilisé pour cette dernière entre 50% et 50% de notre base de données. Nous avons commencé par la base de données déjà séparée en mot et graphique, puis celle générée pour ensuite testé avec le mélange des deux.

Dans nos tests, chaque descripteur ou combinaisons de descripteurs choisis, a été utilisé avec les 4 types de classifieurs à tour de rôle, et pour chacun d'entre eux.

4.5. Test et résultats

Nous avons utilisé 3131 images contenant des objets textuels et 680 objets graphiques dans la 1ere base. La seconde base comporte 627 objets textuels et 383 objets graphiques.

L'utilisation de ces trois bases de données s'effectuera selon une règle très employée, 1/2 (50%) des données pour la phase apprentissage et 1/2 pour la phase test (50%).

Pour les testes, nous avons utilisé 22 descripteurs simple et en combinaison.

1) Descripteur axOGDFeature:
   Dérivés gaussien orientée (OGD).

2) Descripteur qmfEnergy:
   Energie de texture.

3) Descripteur curvatureFeature:
   Curvature Scale Space

4) Descripteur orientationFeature:
   Calcule de l'orientation.

5) Descripteur regionShapeFeaturesZernike:
   Zernike moments magnitude of the coefficients feature, Zernike moments phase of the coefficients feature, Zernike moments real part feature et Zernike moments imaginary part feature.

6) Descripteur regionShapeFeaturesMPEG7:
   MPEG-7 magnitude of the coefficients feature, MPEG-7 phase of the coefficients feature, MPEG-7 real part feature et MPEG-7 imaginary part feature.

7) Descripteur regionShapeFeaturesOrthoRadial:
   Radial Orthogonal magnitude of the coefficients feature, Radial Orthogonal phase of the coefficients feature, Radial Orthogonal real part feature et Radial Orthogonal imaginary part feature.

8) Descripteur geometricFeatures:
   Area size, Border size, X-component of COG, Y-component of COG, Smallest x coordinate, Largest x coordinate, Smallest y coordinate, Largest y coordinate, Central moment, Inertia parallel to main axis, Inertia orthogonal to main axis, Orientation, Eccentricity, Compactness, Minimum distance to border, Maximum distance to border, Mean distance to border, Leftmost distance to main axis, Rightmost distance to main axis, Frontmost distance to COG, along main axis, Rearmost distance to COG, along main axis.

9) Descripteur huMoments:
   Les moments de hu, Xcog, eigen1, eigen2, orientation, Moment d'ordre 00.

10) Descripteur borderSignatureArea:

11) Descripteur borderSignatureDistance:

12) Descripteur curvatureScaleSpace:
    Curvature Scale Space max sigma et Curvature Scale Space max row

13) Descripteur F08_basic_shape:
    Pixel count, Aspect ratio, Centroid row, Centroid col, Triangularity, Ellipticity, Ellipse major axis length, Ellipse minor axis length, Ellipse aspect ratio, Ellipse eccentricity, Ellipse orientation, Equivalent diameter, Maximum Diameter, Solidity, Elliptic variance, Circularity, Rectangularity, Radial distance features mean, Radial distance features stdev, Radial distance features entropy, Radial distance features roughness, Chord length statistics mean, Chord length statistics variance, Chord length statistics skewness, Chord length statistics kurtosis, Descripteur F08_Fourier:

14) Descripteur F08_zernike_moments:

15) Descripteur F08_Contour_Sequence_Moments:

16) Descripteur F08_hu_moments_contours:

17) Descripteur F08_Flusser_moments:

18) Descripteur itkImageEntropy:

    Histograme de Frequence et Entropy.

19) Descripteur CV:
    Les Moments m00, m10, m01, m20, m11, m02, m30, m21, m12, m03.
    Les moments centrés mu03, mu11, mu12, mu20, mu21, mu30.
    Les Moments de contour m00, m10, m01, m20, m11, m02, m30, m21, m12, m03.
    Les moments de contour centrés mu03, mu11, mu12, mu20, mu21, mu30.
    Les Moments de hu.
    Pourcentage de pixels Noir dans l'image.
    Pourcentage de pixels Noir dans l'image de Contour.
    Pourcentage de pixels Noir dans l'image / Pourcentage de pixel Noir dans l'image de Contour.
    Pourcentage de pixels Noir dans l'enveloppe convexe.
    Pourcentage de pixels Noir dans l'enveloppe convexe de la projection Verticale.
    Pourcentage de pixels Noir dans l'enveloppe convexe Horizontale.

20) Descripteur Hough Lines:

21) Descripteur Hough Cercles:

Les meilleurs résultats obtenus avec AdaBoost utilisant les arbres de décisions :

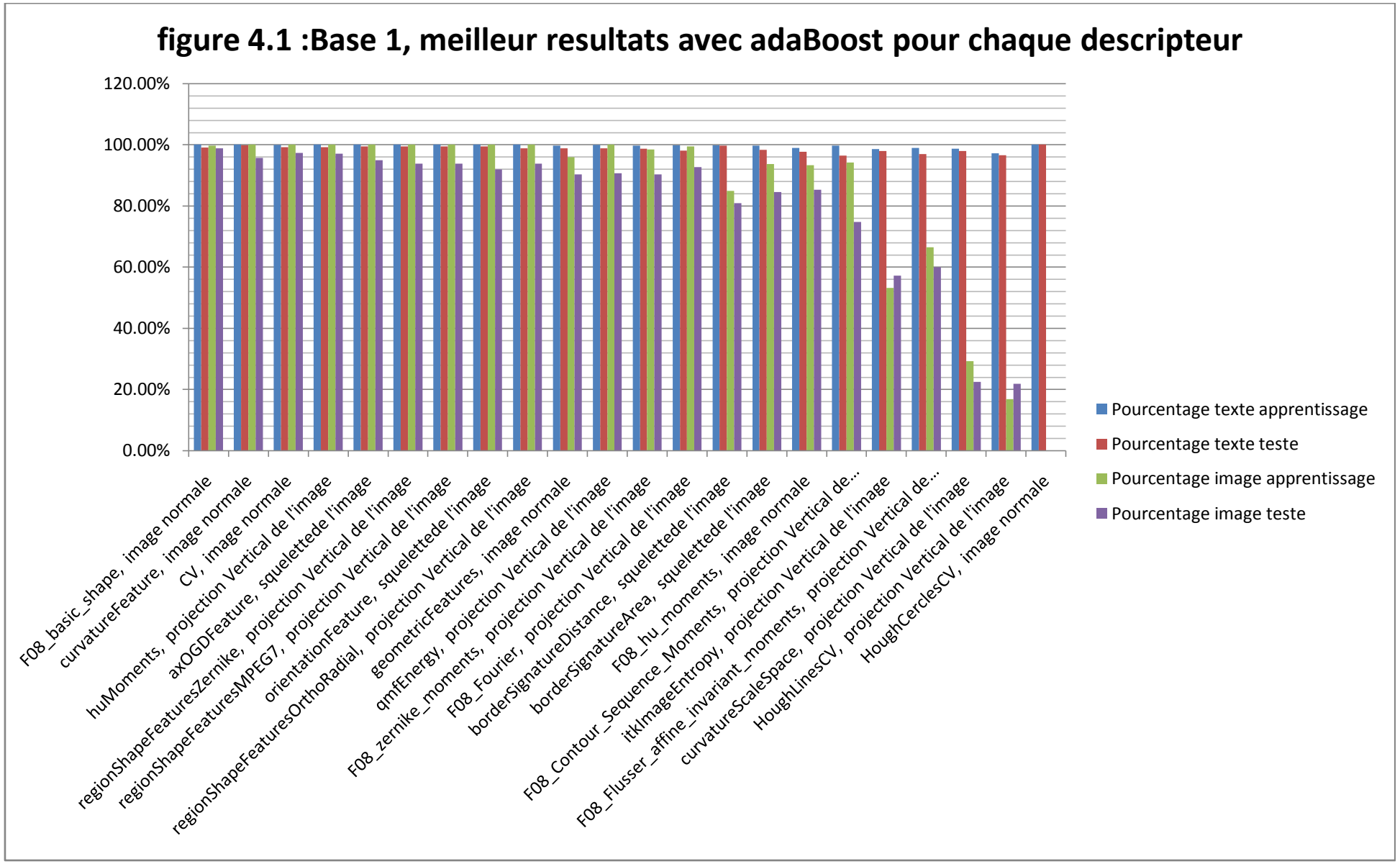


figure 4.1 :Base 1, meilleur resultats avec adaBoost pour chaque descripteur
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
F08_basic_shape, image normale
curvatureFeature, image normale
CV, image normale
huMoments, projection Vertical de l'image
axOGDFeature, squelettede l'image
regionShapeFeaturesZernike, projection Vertical de l'image
regionShapeFeaturesMPEG7, projection Vertical de l'image
regionShapeFeaturesOrthoRadial, projection Vertical de l'image
orientationFeature, squelettede l'image
geometricFeatures, image normale
qmfEnergy, projection Vertical de l'image
F08_zernike_moments, projection Vertical de l'image
F08_Fourier, projection Vertical de l'image
borderSignatureDistance, squelettede l'image
borderSignatureArea, squelettede l'image
F08_hu_moments, image normale
F08_Contour_Sequence_Moments, projection Vertical de...
itkImageEntropy, projection Vertical de l'image
F08_Flusser_affine_invariant_moments, projection Vertical de...
curvatureScaleSpace, projection Vertical de l'image
HoughLinesCV, projection Vertical de l'image
HoughCerclesCV, image normale
Pourcentage texte apprentissage
Pourcentage texte teste
Pourcentage image apprentissage
Pourcentage image teste

	F08_basic_shape	huMoments	CV
Temps d'apprentissage (secondes)	6.562	3.063	11.282
Transformation de l'image	image normale	projection Verticale de l'image	image normale
Nombre d'image d'apprentissage	342	310	342
Nombre de texte d'apprentissage	1563	1595	1563
Nombre d'images bien classées	675	669	671
Nombre d'images mal classées	5	11	9
Nombre de texte bien classés	3117	3119	3118
Nombre de texte mal classés	14	12	13
Taux global de reconnaissance de textes	99.5529%	99.6167%	99.5848%
Taux global de reconnaissance d'images	99.2647%	98.3824%	98.6765%
Taux de reconnaissance de texte dans l'apprentissage	100.0%	100.0%	99.936%
Taux de reconnaissance de texte dans les testes	99.1071%	99.2188%	99.2347%
Taux de reconnaissance des images dans l'apprentissage	99.7076%	100.0%	100.0%
Taux de reconnaissance des images dans les testes	98.8166%	97.027%	97.3373%
Taux global de reconnaissance dans l'apprentissage	99.9475%	100.0%	99.9475%
Taux global de reconnaissance dans les testes	99.0556%	98.7933%	98.8982%

Tableau 4.1 : les meilleurs résultats pour la 1ere base de données

Au niveau du Tableau 4.1, nous constatons que F08_basic_shape donne, de manière globale, les meilleurs résultats durant les tests. Concernant, la phase d'apprentissage, nous considérons que le descripteur les moments de hu génère de meilleurs résultats puisqu'il permet une meilleure reconnaissance des textes. Par ailleurs, il offre un apprentissage deux plus rapide que les deux autres.

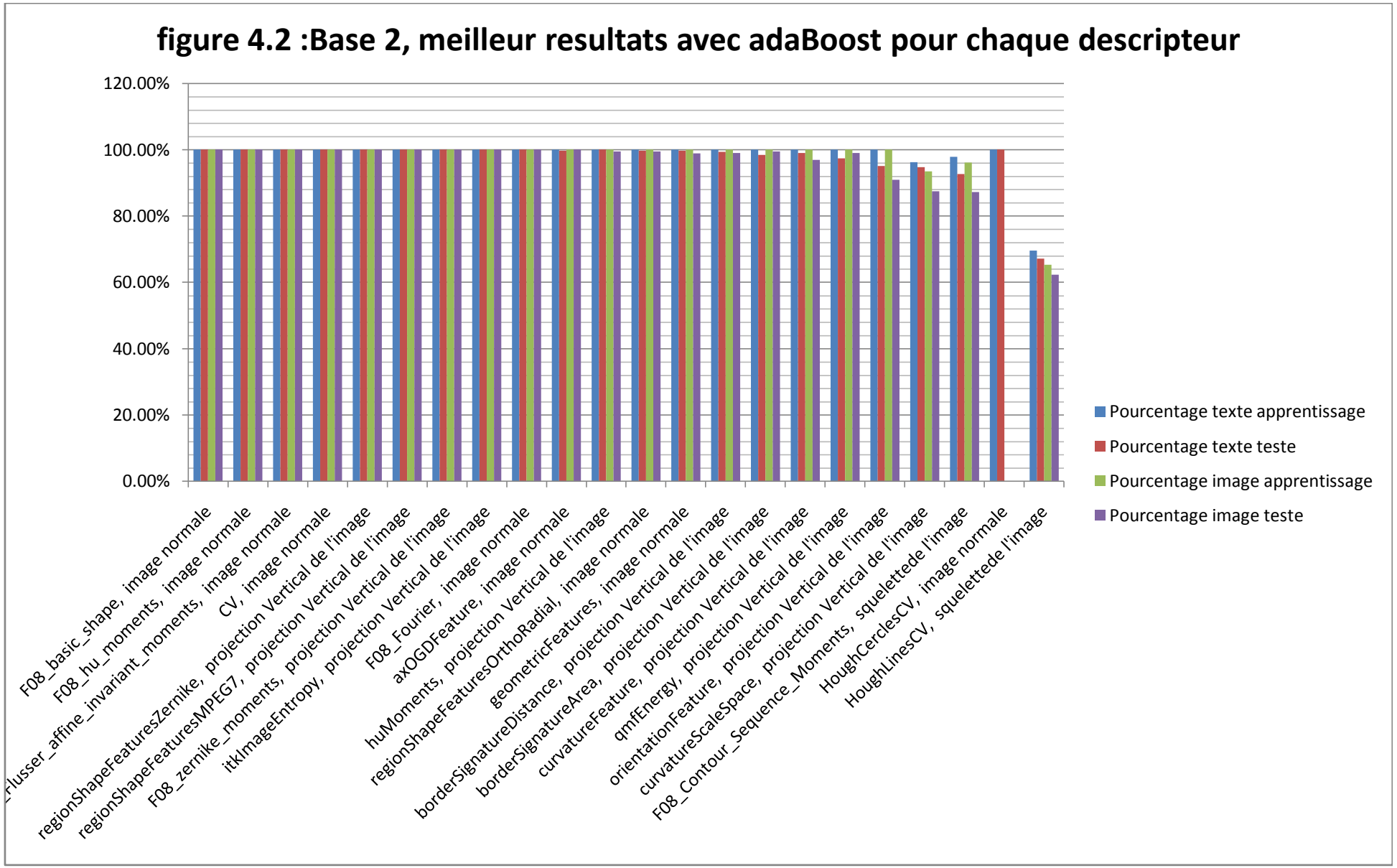


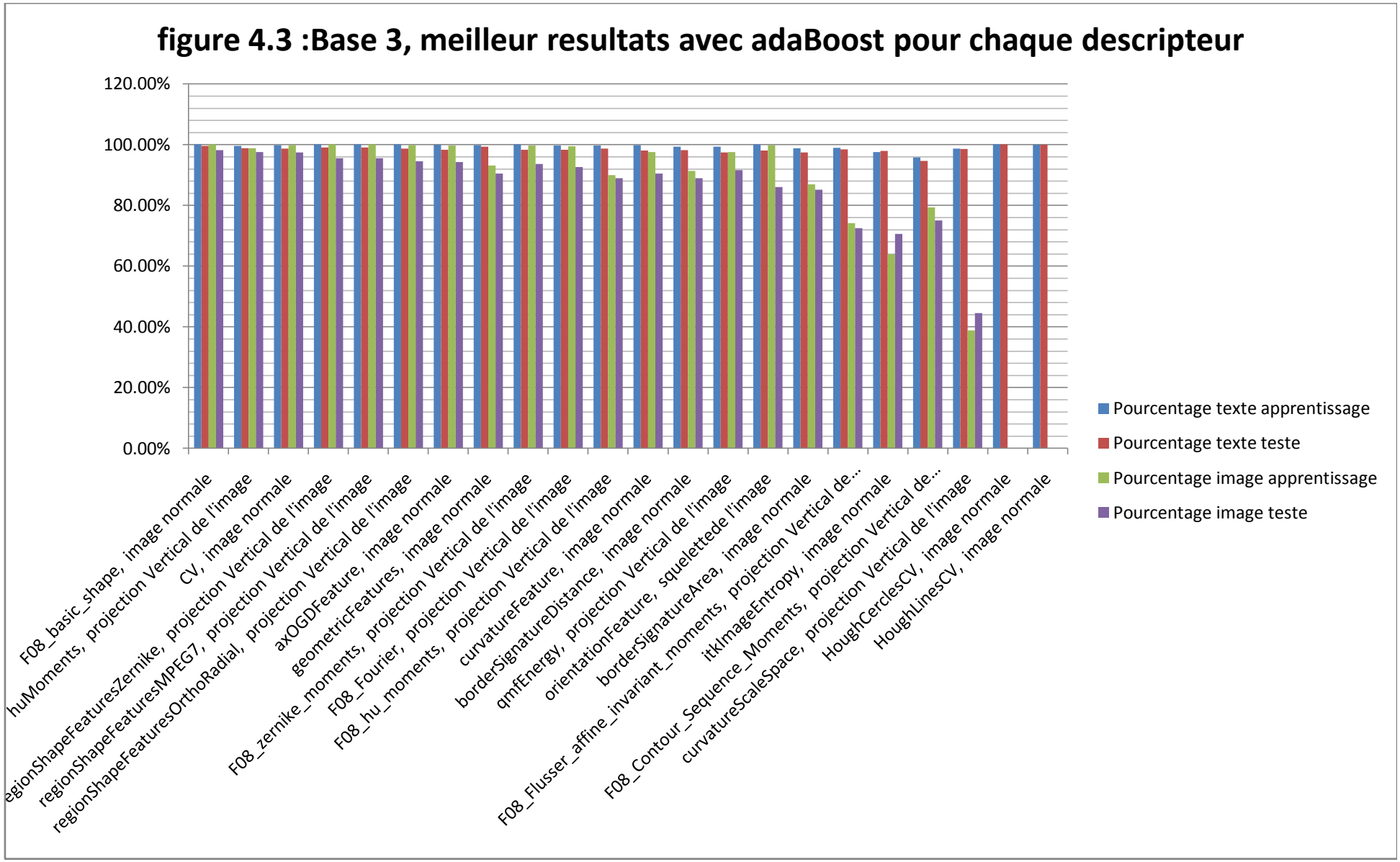
figure 4.2 :Base 2, meilleur resultats avec adaBoost pour chaque descripteur
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
F08_basic_shape, image normale
F08_hu_moments, image normale
rlusser_affine_invariant_moments, image normale
CV, image normale
regionShapeFeaturesZernike, projection Vertical de l'image
regionShapeFeaturesMPEG7, projection Vertical de l'image
F08_zernike_moments, projection Vertical de l'image
itkImageEntropy, projection Vertical de l'image
F08_Fourier, image normale
axOGDFeature, image normale
huMoments, projection Vertical de l'image
regionShapeFeaturesOrthoRadial, image normale
geometricFeatures, image normale
borderSignatureDistance, projection Vertical de l'image
borderSignatureArea, projection Vertical de l'image
curvatureFeature, projection Vertical de l'image
qmfEnergy, projection Vertical de l'image
orientationFeature, projection Vertical de l'image
curvatureScaleSpace, projection Vertical de l'image
F08_Contour_Sequence_Moments, squelettede l'image
HoughCerclesCV, image normale
HoughLinesCV, squelettede l'image
Pourcentage texte apprentissage
Pourcentage texte teste
Pourcentage image apprentissage
Pourcentage image teste

	F08_basic_shape	F08_hu_moments	F08_Flusser_affine_invariant_moments
Temps d'apprentissage (secondes)	0.954	0.282	0.234
Transformation de l'image	image normale	image normale	image normale
Nombre d'image d'apprentissage	183	183	183
Nombre de texte d'apprentissage	322	322	322
Nombre d'images bien classées	383	383	383
Nombre d'images mal classées	0	0	0
Nombre de texte bien classés	627	627	627
Nombre de texte mal classés	0	0	0
Taux global de reconnaissance de textes	100.0%	100.0%	100.0%
Taux global de reconnaissance d'images	100.0%	100.0%	100.0%
Taux de reconnaissance de texte dans l'apprentissage	100.0%	100.0%	100.0%
Taux de reconnaissance de texte dans les testes	100.0%	100.0%	100.0%
Taux de reconnaissance des images dans l'apprentissage	100.0%	100.0%	100.0%
Taux de reconnaissance des images dans les testes	100.0%	100.0%	100.0%
Taux global de reconnaissance dans l'apprentissage	100.0%	100.0%	100.0%
Taux global de reconnaissance dans les testes	100.0%	100.0%	100.0%

Tableau 4.2 : les meilleurs résultats pour la 2eme base de données

Nous constatons que plusieurs descripteurs nous ont donnés 100% de succès. Toutefois, d'un point de vue temps d'apprentissage, nous remarquons que les moments de hu sont les plus performants.

figure 4.3 :Base 3, meilleur resultats avec adaBoost pour chaque descripteur



	F08_basic_shape	huMoments	CV
Temps d'apprentissage (secondes)	7.401	3.821	12.868
Transformation de l'image	image normale	projection Verticale de l'image	image normale
Nombre d'image d'apprentissage	520	495	540
Nombre de texte d'apprentissage	1890	1915	1870
Nombre d'images bien classées	1053	1043	1048
Nombre d'images mal classées	10	20	15
Nombre de texte bien classés	3750	3727	3729
Nombre de texte mal classés	8	31	29
Taux global de reconnaissance de textes	99.7871%	99.1751%	99.2283%
Taux global de reconnaissance d'images	99.0593%	98.1185%	98.5889%
Taux de reconnaissance de texte dans l'apprentissage	100.0%	99.53%	99.7861%
Taux de reconnaissance de texte dans les testes	99.5717%	98.8063%	98.6758%
Taux de reconnaissance des images dans l'apprentissage	100.0%	98.7879%	99.8148%
Taux de reconnaissance des images dans les testes	98.1584%	97.5352%	97.3231%
Taux global de reconnaissance dans l'apprentissage	100.0%	99.3776%	99.7925%
Taux global de reconnaissance dans les testes	99.2534%	98.5068%	98.3824%

Tableau 4.3 : les meilleurs résultats pour la 3eme base de données

Durant les tests, F08_basic_shape est de loin meilleur que les autres, mais il est aussi deux fois plus lent lors de l'apprentissage.

Les 3 meilleurs résultats obtenus avec les réseaux de neurones:

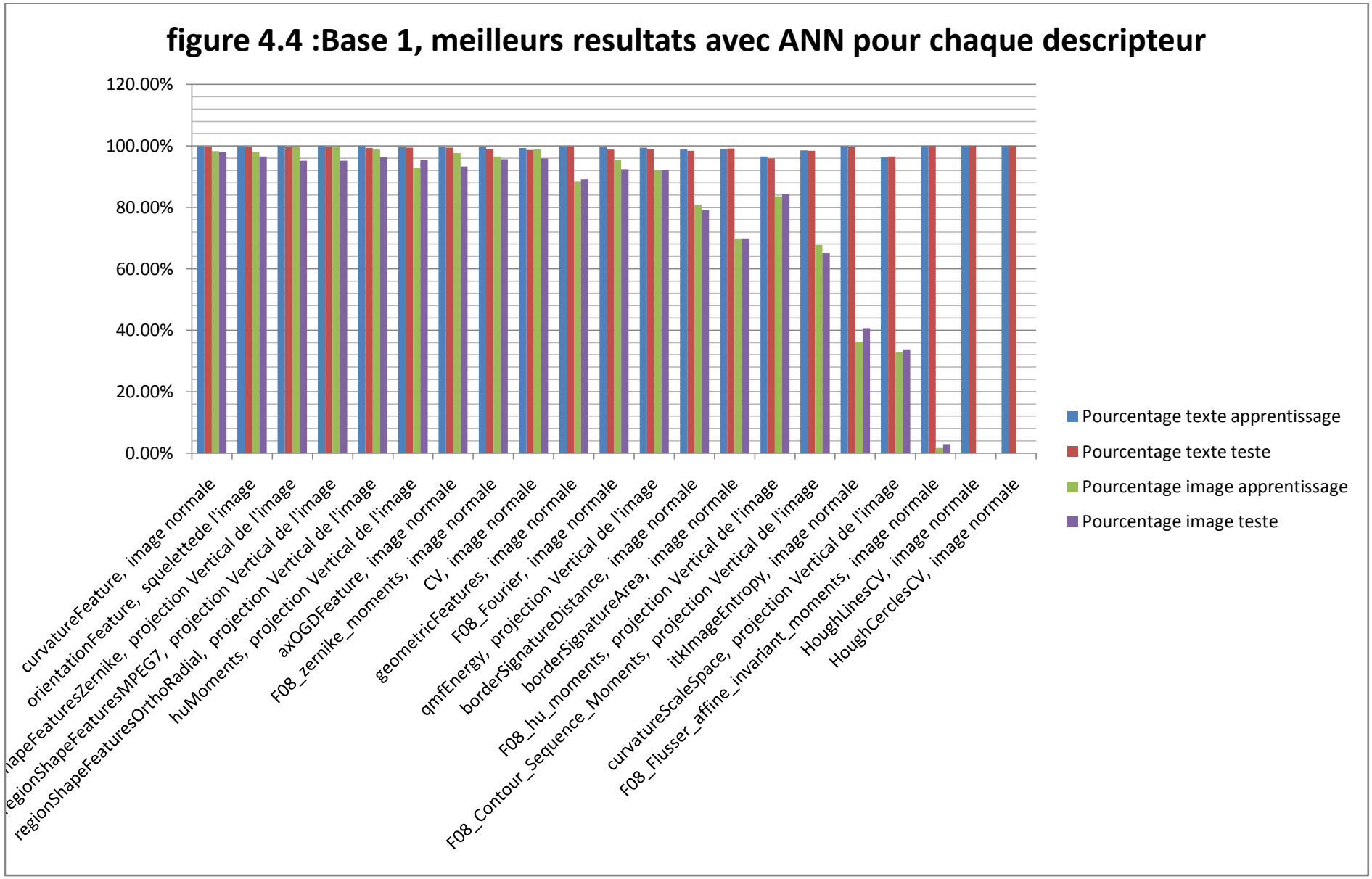


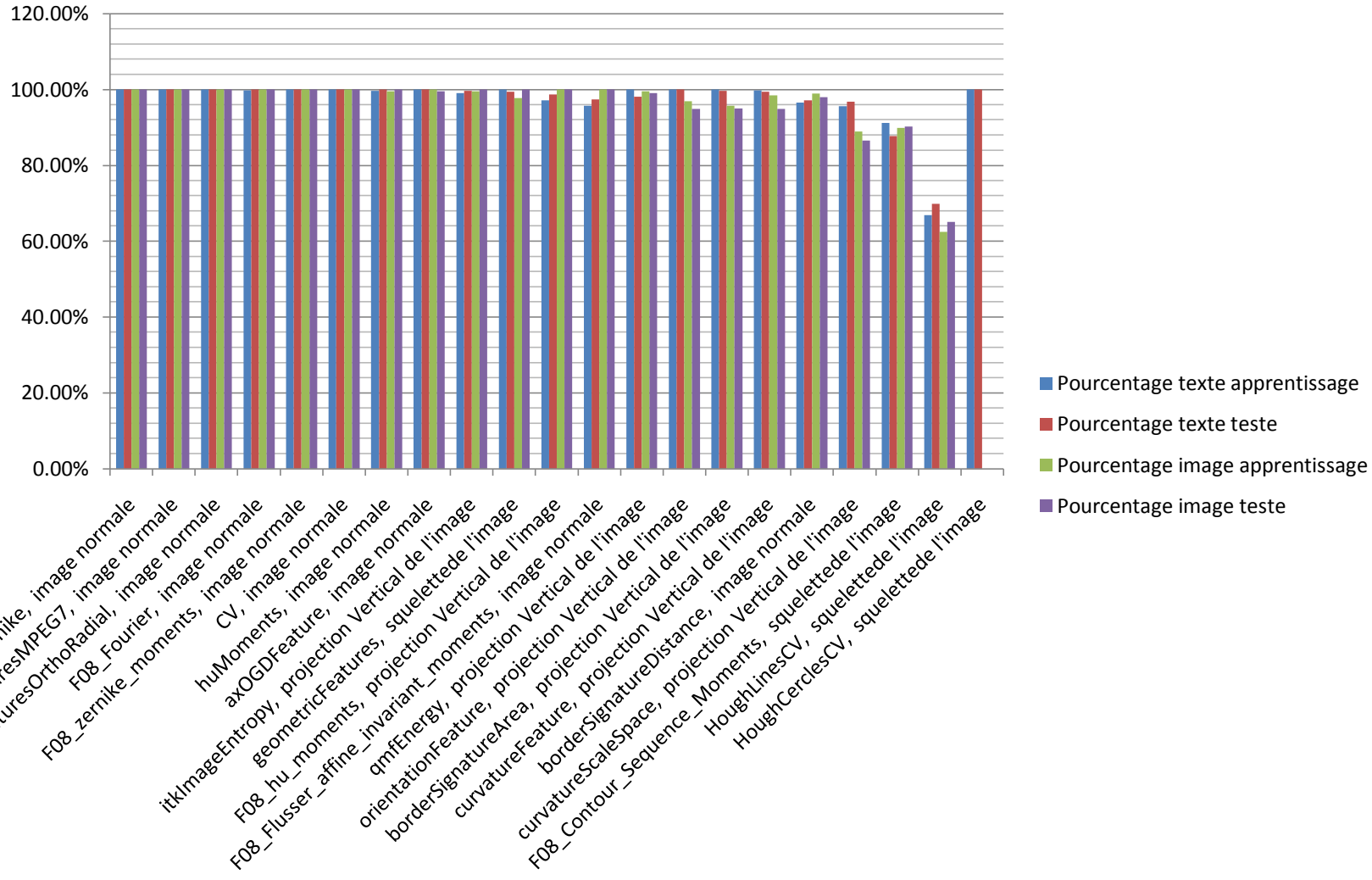
figure 4.4 :Base 1, meilleurs resultats avec ANN pour chaque descripteur
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
curvatureFeature, image normale
orientationFeature, squelettede l'image
regionShapeFeaturesZernike, projection Vertical de l'image
regionShapeFeaturesMPEG7, projection Vertical de l'image
regionShapeFeaturesOrthoRadial, projection Vertical de l'image
huMoments, projection Vertical de l'image, image normale
F08_zernike_moments, image normale
axOGDFeature, image normale
CV, image normale
geometricFeatures, image normale
F08_Fourier, image normale
qmfEnergy, projection Vertical de l'image
borderSignatureDistance, image normale
borderSignatureArea, image normale
F08_hu_moments, projection Vertical de l'image
F08_Contour_Sequence_Moments, projection Vertical de l'image
itklmageEntropy, image normale
curvatureScaleSpace, projection Vertical de l'image
F08_Flusser_affine_invariant_moments, image normale
HoughLinesCV, image normale
HoughCerclesCV, image normale
Pourcentage texte apprentissage
Pourcentage texte teste
Pourcentage image apprentissage
Pourcentage image teste

	curvatureFeature	orientationFeature	regionShapeFeaturesZernike
Temps d'apprentissage (secondes)	290.515	259.843	392.625
Transformation de l'image	image normale	squelette de l'image	projection Verticale de l'image
Nombre d'image d'apprentissage	342	306	328
Nombre de texte d'apprentissage	1563	1599	1577
Nombre d'images bien classées	667	661	662
Nombre d'images mal classées	13	19	18
Nombre de texte bien classés	3126	3122	3123
Nombre de texte mal classés	5	9	8
Taux global de reconnaissance de textes	99.8403%	99.7126%	99.7445%
Taux global de reconnaissance d'images	98.0882%	97.2059%	97.3529%
Taux de reconnaissance de texte dans l'apprentissage	99.872%	99.9375%	100.0%
Taux de reconnaissance de texte dans les testes	99.8087%	99.4778%	99.4852%
Taux de reconnaissance des images dans l'apprentissage	98.2456%	98.0392%	99.6951%
Taux de reconnaissance des images dans les testes	97.929%	96.5241%	95.1705%
Taux global de reconnaissance dans l'apprentissage	99.5801%	99.6325%	99.9475%
Taux global de reconnaissance dans les testes	99.4753%	98.8982%	98.6884%

Tableau 4.4 : les meilleurs résultats pour la 1ere base de données

Nous constatons que le taux de reconnaissance des textes/images durant les tests est meilleur de 0.4% comparé à adaboost mais au prix d'un apprentissage beaucoup plus lent. Cependant, nous remarquons que le taux de reconnaissance dans l'apprentissage n'est pas parfait comme dans adaboost.

figure 4.5 :Base 2, meilleurs resultats avec ANN pour chaque descripteur

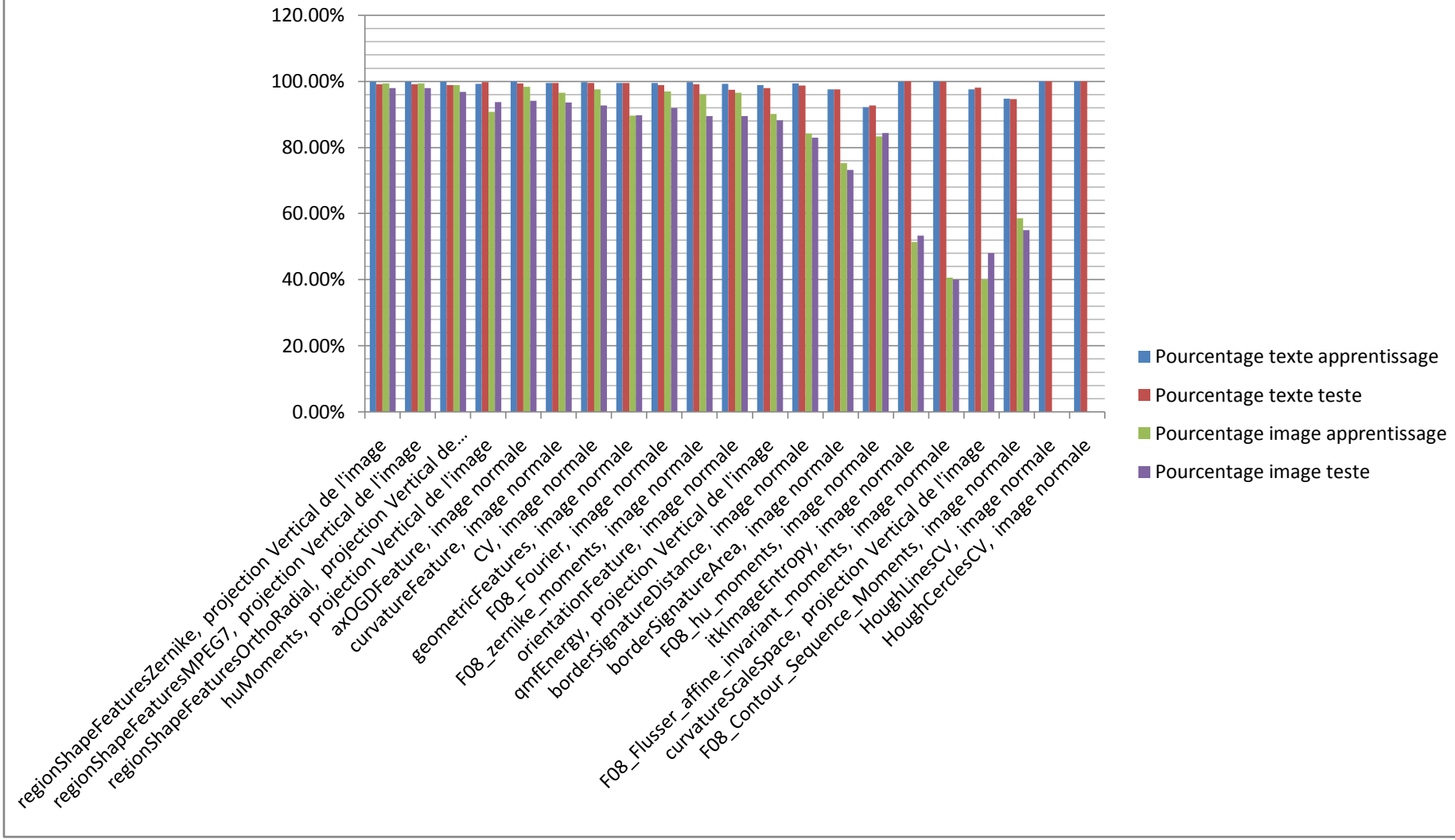


	regionShapeFeaturesZernike	F08_zernike_moments	huMoments
Temps d'apprentissage(secondes)	119.969	77.781	69.281
Transformation de l'image	image normale	image normale	image normale
Nombre d'image d'apprentissage	183	183	209
Nombre de texte d'apprentissage	322	322	296
Nombre d'images bien classées	383	383	382
Nombre d'images mal classées	0	0	1
Nombre de texte bien classés	627	627	626
Nombre de texte mal classés	0	0	1
Taux global de reconnaissance de textes	100.0%	100.0%	99.8405%
Taux global de reconnaissance d'images	100.0%	100.0%	99.7389%
Taux de reconnaissance de texte dans l'apprentissage	100.0%	100.0%	99.6622%
Taux de reconnaissance de texte dans les testes	100.0%	100.0%	100.0%
Taux de reconnaissance des images dans l'apprentissage	100.0%	100.0%	99.5215%
Taux de reconnaissance des images dans les testes	100.0%	100.0%	100.0%
Taux global de reconnaissance dans l'apprentissage	100.0%	100.0%	99.604%
Taux global de reconnaissance dans les testes	100.0%	100.0%	100.0%

Tableau 4.5 : les meilleurs résultats pour la 2eme base de données

Comme avec adaBoost, nous avons obtenu des résultats parfaits mais avec peu de descripteurs comparés a AdaBoost. Toutefois, les ANN présentent un temps d'apprentissage relativement élevé.

figure 4.6:Base 3, meilleurs resultats avec ANN pour chaque descripteur



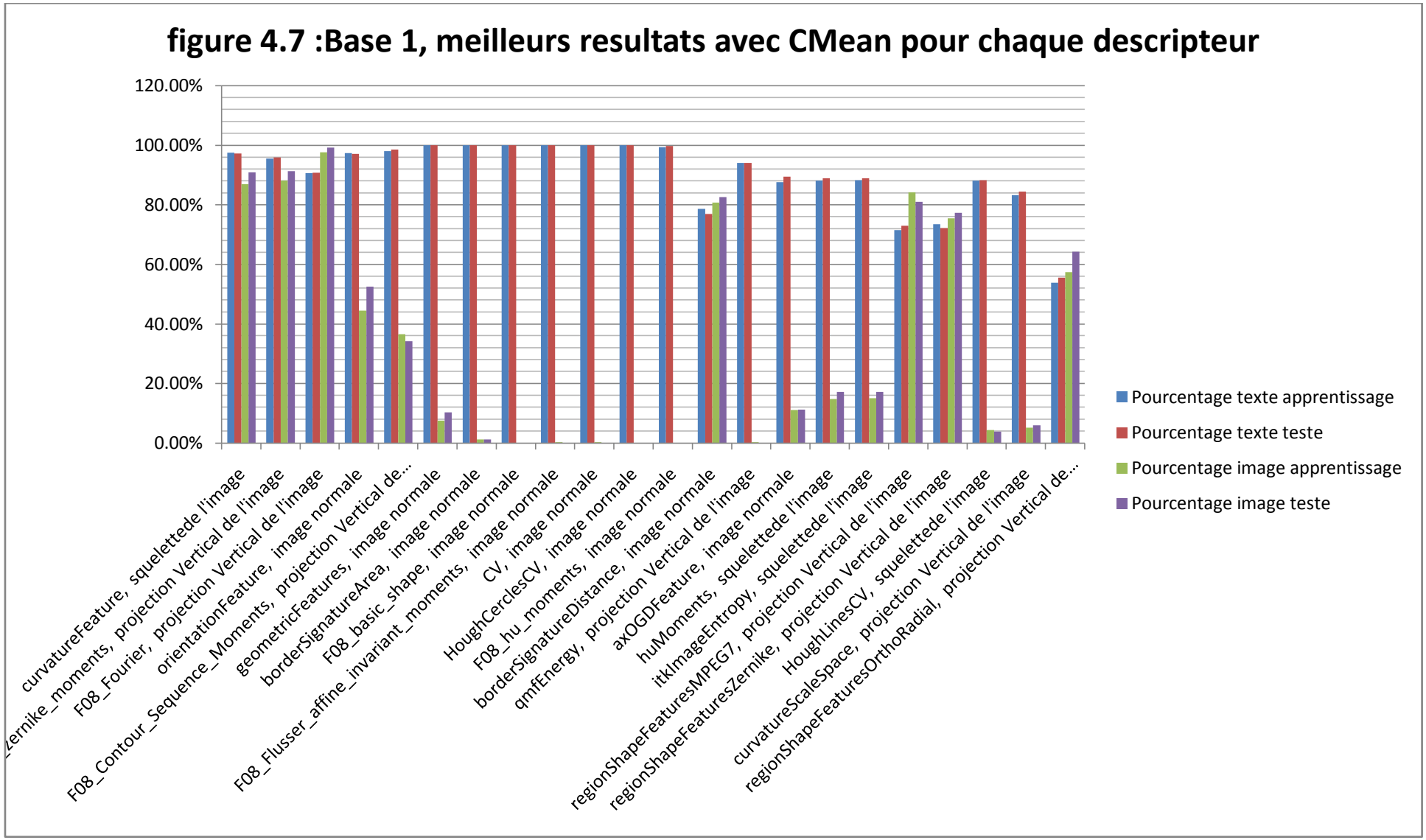
	regionShapeFeaturesZernike	regionShapeFeaturesOrthoRadial	huMoments
Temps d'apprentissage (secondes)	498.45	497.625	284.498
Transformation de l'image	projection Verticale de l'image	projection Verticale de l'image	projection Verticale de l'image
Nombre d'image d'apprentissage	495	495	495
Nombre de texte d'apprentissage	1915	1915	1915
Nombre d'images bien classées	1048	1039	981
Nombre d'images mal classées	15	24	82
Nombre de texte bien classés	3740	3732	3738
Nombre de texte mal classés	18	26	20
Taux global de reconnaissance de textes	99.521%	99.3081%	99.4678%
Taux global de reconnaissance d'images	98.5889%	97.7422%	92.286%
Taux de reconnaissance de texte dans l'apprentissage	99.8956%	99.7911%	99.2167%
Taux de reconnaissance de texte dans les testes	99.1319%	98.8063%	99.7287%
Taux de reconnaissance des images dans l'apprentissage	99.3939%	98.7879%	90.7071%
Taux de reconnaissance des images dans les testes	97.8873%	96.831%	93.662%
Taux global de reconnaissance dans l'apprentissage	99.7925%	99.5851%	97.4689%
Taux global de reconnaissance dans les testes	98.8387%	98.3409%	98.2995%

Tableau 4.6 : les meilleurs résultats pour la 3eme base de données

Cette expérimentation a généré de bons résultats, mais adaboost a produit de meilleurs résultats avec un temps d'apprentissage beaucoup plus réduit.

Les 3 meilleurs résultats obtenus avec CMean :

figure 4.7 :Base 1, meilleurs resultats avec CMean pour chaque descripteur

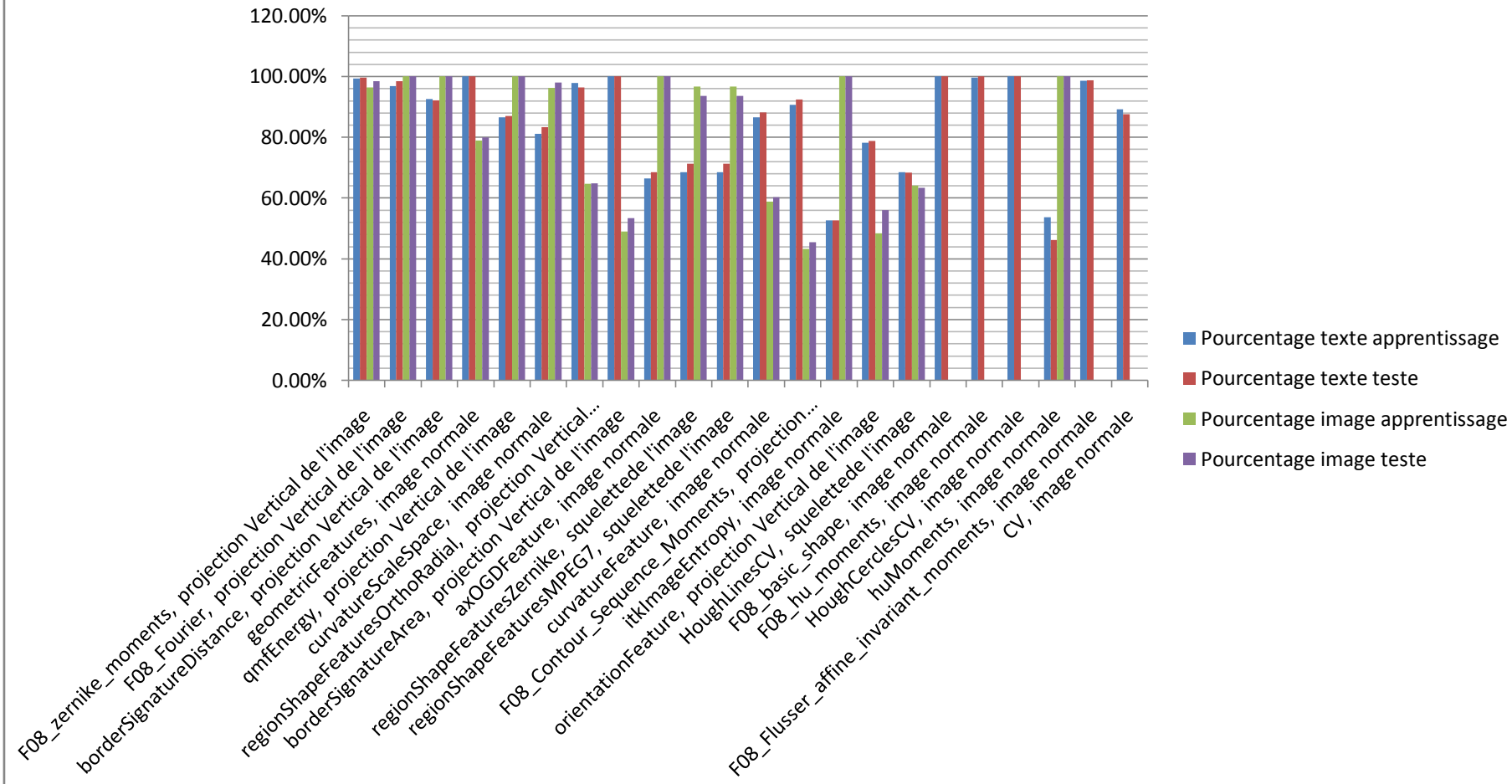


	curvatureFeature	F08_zernike_moments	F08_Fourier
Temps d'apprentissage (secondes)	1.14	0.797	0.891
Transformation de l'image	Squelette de l'image	projection Verticale de l'image	projection Verticale de l'image
Nombre d'image d'apprentissage	299	327	328
Nombre de texte d'apprentissage	1606	1578	1577
Nombre d'images bien classées	606	610	669
Nombre d'images mal classées	74	70	11
Nombre de texte bien classés	3048	2996	2838
Nombre de texte mal classés	83	135	293
Taux global de reconnaissance de textes	97.3491%	95.6883%	90.642%
Taux global de reconnaissance d'images	89.1176%	89.7059%	98.3824%
Taux de reconnaissance de texte dans l'apprentissage	97.4471%	95.5006%	90.6151%
Taux de reconnaissance de texte dans les testes	97.2459%	95.8789%	90.6692%
Taux de reconnaissance des images dans l'apprentissage	86.9565%	88.0734%	97.561%
Taux de reconnaissance des images dans les testes	90.8136%	91.2181%	99.1477%
Taux global de reconnaissance dans l'apprentissage	95.8005%	94.2257%	91.811%
Taux global de reconnaissance dans les testes	95.9601%	95.0157%	92.235%

Tableau 4.7 : les meilleurs résultats pour la 1ere base de données

A l'issue de cette expérimentation, nous observons des résultats moins performants qu'avec les deux premiers classifieurs et ce quelque soit le descripteur utilisé.

figure 4.8 :Base 2, meilleurs resultats avec CMean pour chaque descripteur

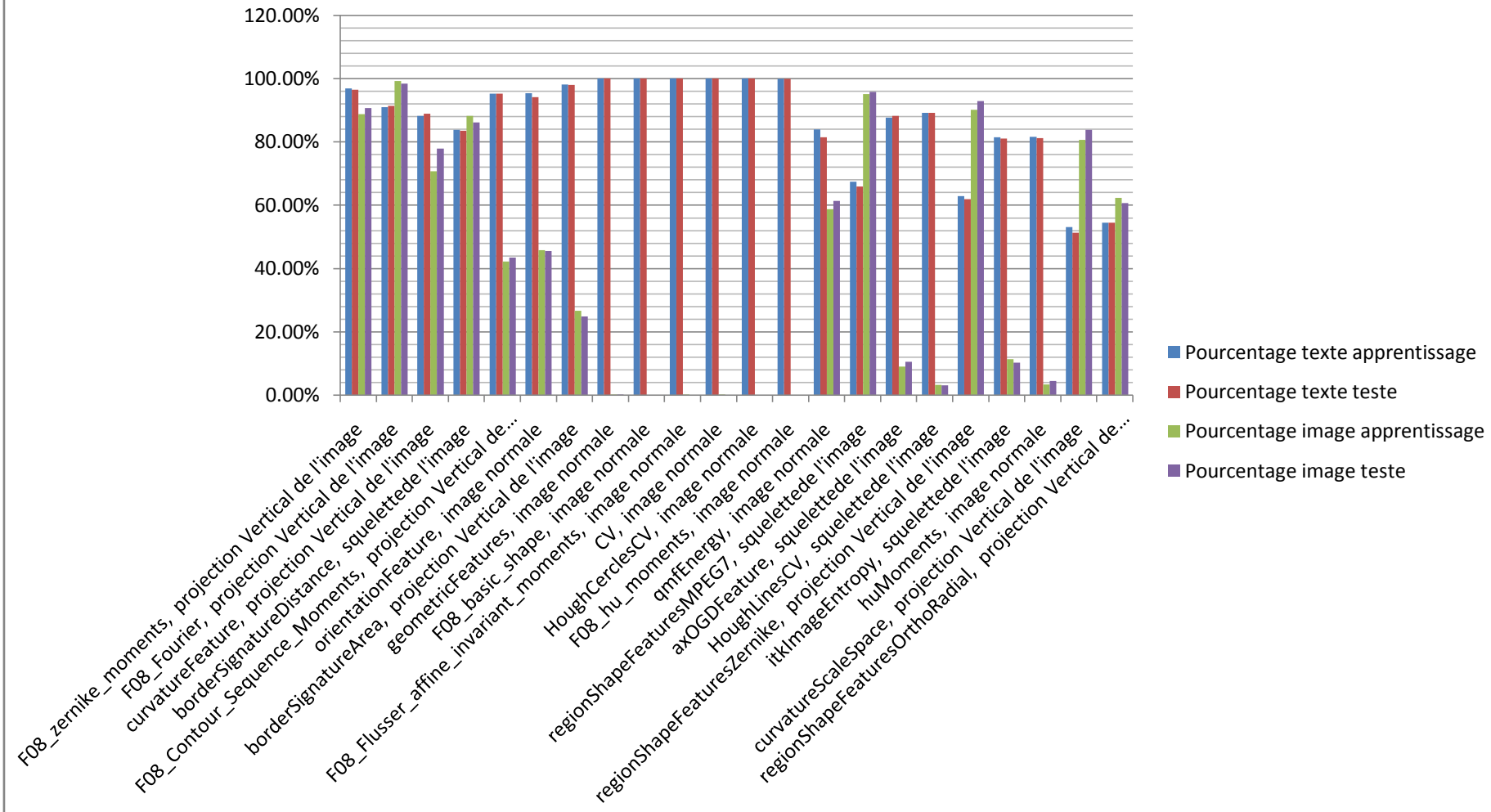


	F08_zernike_moments	F08_Fourier	geometricFeatures
Temps d'apprentissage(secondes)	0.125	0.156	0.187
Transformation de l'image	projection Verticale de l'image	projection Verticale de l'image	image normale
Nombre d'image d'apprentissage	190	190	209
Nombre de texte d'apprentissage	315	315	296
Nombre d'images bien classées	373	383	304
Nombre d'images mal classées	10	0	79
Nombre de texte bien classés	624	612	627
Nombre de texte mal classés	3	15	0
Taux global de reconnaissance de textes	99.5215%	97.6077%	100.0%
Taux global de reconnaissance d'images	97.389%	100.0%	79.3734%
Taux de reconnaissance de texte dans l'apprentissage	99.3651%	96.8254%	100.0%
Taux de reconnaissance de texte dans les testes	99.6795%	98.3974%	100.0%
Taux de reconnaissance des images dans l'apprentissage	96.3158%	100.0%	78.9474%
Taux de reconnaissance des images dans les testes	98.4456%	100.0%	79.8851%
Taux global de reconnaissance dans l'apprentissage	98.2178%	98.0198%	91.2871%
Taux global de reconnaissance dans les testes	99.2079%	99.0099%	93.0693%

Tableau 4.8 : les meilleurs résultats pour la 2eme base de données

Cette expérimentation a généré de bons résultats pour les 2 premiers descripteurs. Toutefois, ils ne sont pas aussi parfaits qu'avec les ANN ou adaboost. Le temps d'apprentissage est néanmoins plus rapide.

figure 4.9 :Base 3, meilleurs resultats avec CMean pour chaque descripteur



	F08_zernike_moments	F08_Fourier	F08_Contour_Sequence_Moments
Temps d'apprentissage (secondes)	0.658	1.012	0.258
Transformation de l'image	projection Verticale de l'image	projection Verticale de l'image	projection Verticale de l'image
Nombre d'image d'apprentissage	495	495	529
Nombre de texte d'apprentissage	1915	1915	1881
Nombre d'images bien classées	954	1050	455
Nombre d'images mal classées	109	13	608
Nombre de texte bien classés	3634	3426	3578
Nombre de texte mal classés	124	332	180
Taux global de reconnaissance de textes	96.7004%	91.1655%	95.2102%
Taux global de reconnaissance d'images	89.746%	98.777%	42.8034%
Taux de reconnaissance de texte dans l'apprentissage	96.9191%	90.9661%	95.2153%
Taux de reconnaissance de texte dans les testes	96.4731%	91.3728%	95.2051%
Taux de reconnaissance des images dans l'apprentissage	88.6869%	99.1919%	42.155%
Taux de reconnaissance des images dans les testes	90.669%	98.4155%	43.4457%
Taux global de reconnaissance dans l'apprentissage	95.2282%	92.6556%	83.5685%
Taux global de reconnaissance dans les testes	95.1058%	93.0319%	83.7412%

Tableau 4.9 : les meilleurs résultats pour la 3eme base de données

Cette expérience a généré aussi de moins bons résultats.

Les 3 meilleurs résultats obtenus avec fuzzy CMean:

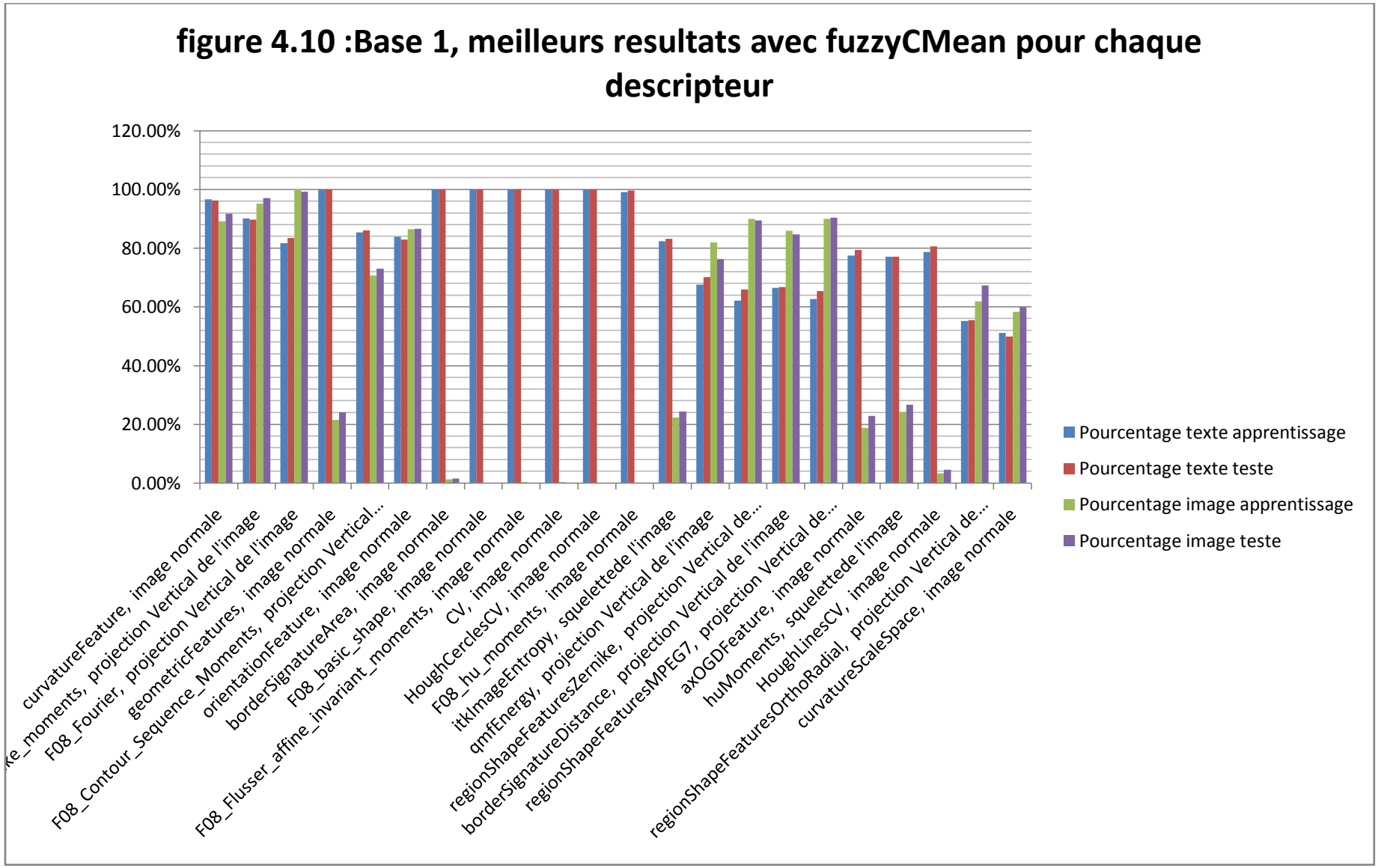


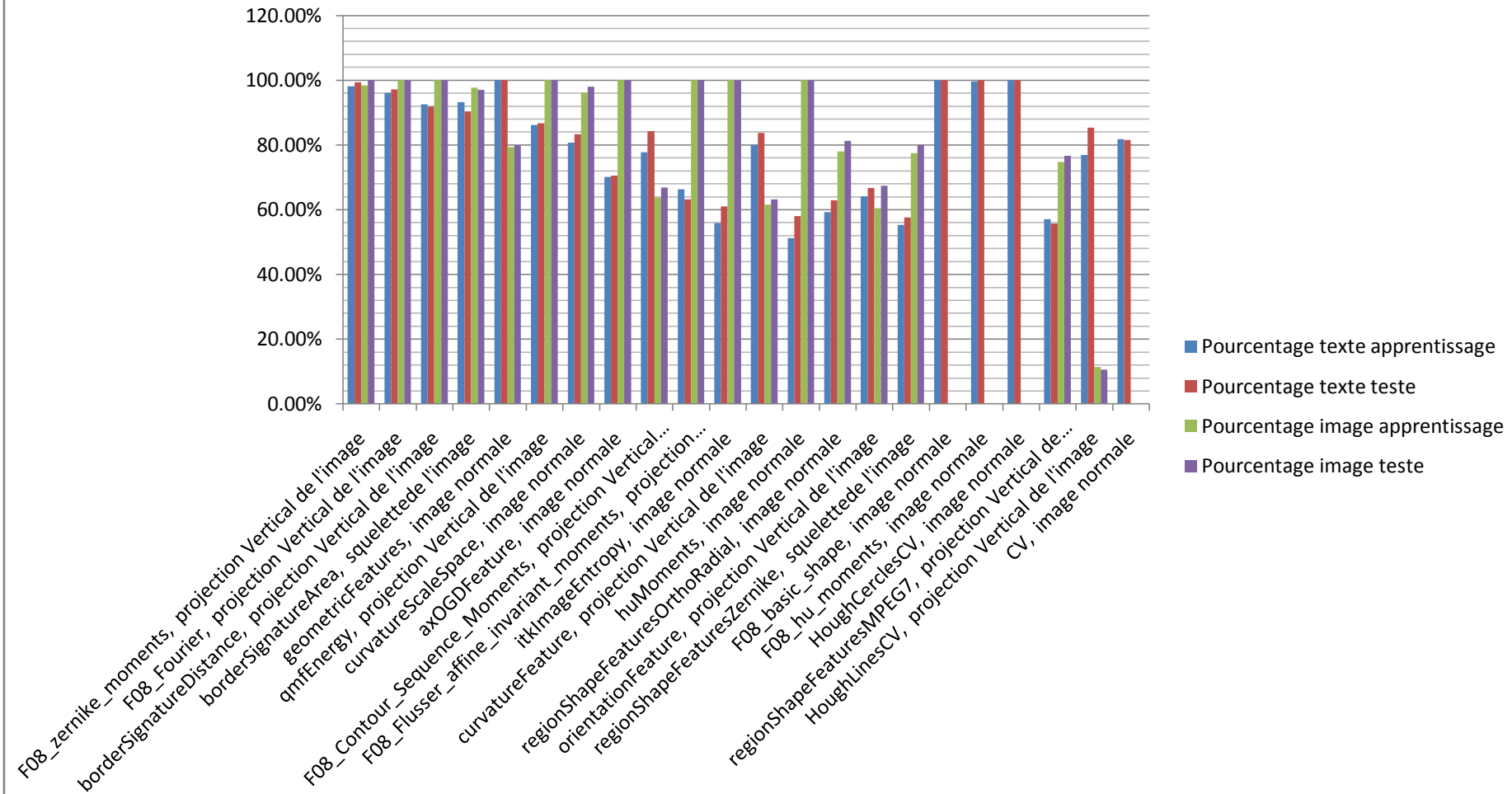
figure 4.10 :Base 1, meilleurs resultats avec fuzzyCMean pour chaque descripteur
120.00%
100.00%
80.00%
60.00%
40.00%
20.00%
0.00%
curvatureFeature, image normale
ke_moments, projection Vertical de l'image
F08_Fourier, projection Vertical de l'image
F08_Contour_Sequence_Moments, projection Vertical...
geometricFeatures, image normale
orientationFeature, image normale
borderSignatureArea, image normale
F08_basic_shape, image normale
F08_Flusser_affine_invariant_moments, image normale
CV, image normale
HoughCerclesCV, image normale
F08_hu_moments, image normale
itklmageEntropy, squelettede l'image
qmfEnergy, projection Vertical de l'image
regionShapeFeaturesZernike, projection Vertical de...
borderSignatureDistance, projection Vertical de l'image
regionShapeFeaturesMPEG7, projection Vertical de...
axOGDFeature, image normale
huMoments, squelettede l'image
HoughLinesCV, image normale
regionShapeFeaturesOrthoRadial, projection Vertical de...
curvatureScaleSpace, image normale
Pourcentage texte apprentissage
Pourcentage texte teste
Pourcentage image apprentissage
Pourcentage image teste

	curvatureFeature	F08_zernike_moments	F08_Contour_Sequence_Moments
Temps d'apprentissage (secondes)	1.297	0.609	0.265
Transformation de l'image	image normale	projection Verticale de l'image	projection Verticale de l'image
Nombre d'image d'apprentissage	331	310	328
Nombre de texte d'apprentissage	1574	1595	1577
Nombre d'images bien classées	615	654	489
Nombre d'images mal classées	65	26	191
Nombre de texte bien classés	3020	2815	2682
Nombre de texte mal classés	111	316	449
Taux global de reconnaissance de textes	96.4548%	89.9074%	85.6595%
Taux global de reconnaissance d'images	90.4412%	96.1765%	71.9118%
Taux de reconnaissance de texte dans l'apprentissage	96.6328%	90.094%	85.2885%
Taux de reconnaissance de texte dans les testes	96.2749%	89.7135%	86.036%
Taux de reconnaissance des images dans l'apprentissage	89.1239%	95.1613%	70.7317%
Taux de reconnaissance des images dans les testes	91.6905%	97.027%	73.0114%
Taux global de reconnaissance dans l'apprentissage	95.3281%	90.9186%	82.7822%
Taux global de reconnaissance dans les testes	95.4355%	91.1333%	83.6306%

Tableau 4.10 : les meilleurs résultats pour la 1ere base de données

Comme avec Cmean, fuzzy Cmean produit aussi de moins bon résultats.

figure 4.11 :Base 2, meilleurs resultats avec fuzzyCMean pour chaque descripteur

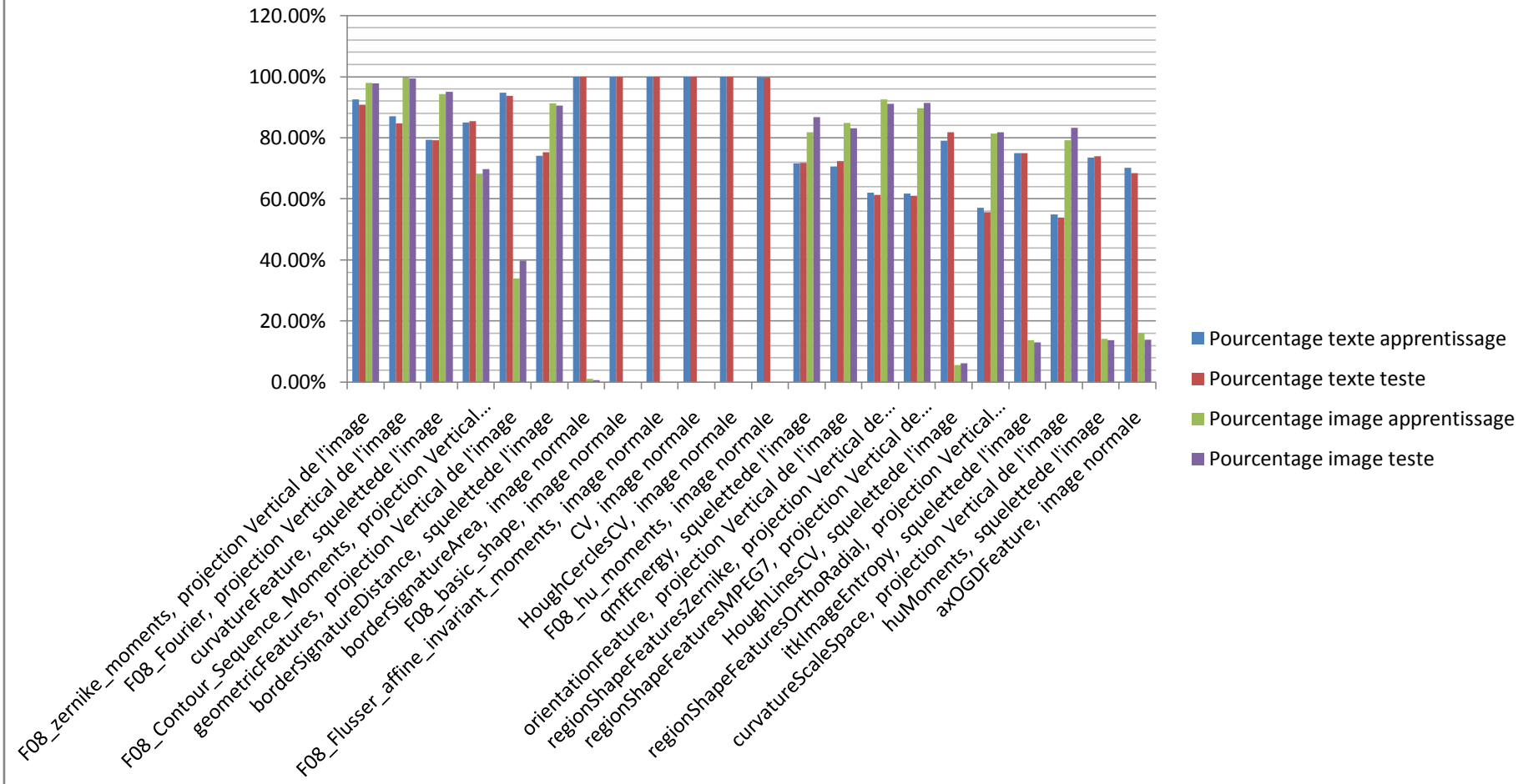


	F08_zernike_moments	F08_Fourier	borderSignatureDistance
Temps d'apprentissage (seconds)	0.141	0.656	0.203
Transformation de l'image	projection Verticale de l'image	projection Verticale de l'image	projection Verticale de l'image
Nombre d'image d'apprentissage	190	194	184
Nombre de texte d'apprentissage	315	311	321
Nombre d'images bien classées	380	383	383
Nombre d'images mal classées	3	0	0
Nombre de texte bien classés	619	606	578
Nombre de texte mal classés	8	21	49
Taux global de reconnaissance de textes	98.7241%	96.6507%	92.185%
Taux global de reconnaissance d'images	99.2167%	100.0%	100.0%
Taux de reconnaissance de texte dans l'apprentissage	98.0952%	96.1415%	92.5234%
Taux de reconnaissance de texte dans les testes	99.359%	97.1519%	91.8301%
Taux de reconnaissance des images dans l'apprentissage	98.4211%	100.0%	100.0%
Taux de reconnaissance des images dans les testes	100.0%	100.0%	100.0%
Taux global de reconnaissance dans l'apprentissage	98.2178%	97.6238%	95.2475%
Taux global de reconnaissance dans les testes	99.604%	98.2178%	95.0495%

Tableau 4.11 : les meilleurs résultats pour la 2eme base de données

Cette expérimentation a généré de bons résultats qui restent loin d'être aussi parfaits que ceux des ANN ou adaboost.

figure 4.12 :Base 3, meilleurs resultats avec fuzzyCMean pour chaque descripteur



	F08_zernike_moments	F08_Fourier	F08_Contour_Sequence_Moments
Temps d'apprentissage (secondes)	1.326	2.449	0.281
Transformation de l'image	projection Verticale de l'image	projection Verticale de l'image	projection Verticale de l'image
Nombre d'image d'apprentissage	529	529	520
Nombre de texte d'apprentissage	1881	1881	1890
Nombre d'images bien classées	1040	1058	733
Nombre d'images mal classées	23	5	330
Nombre de texte bien classés	3445	3228	3201
Nombre de texte mal classés	313	530	557
Taux global de reconnaissance de textes	91.6711%	85.8968%	85.1783%
Taux global de reconnaissance d'images	97.8363%	99.5296%	68.9558%
Taux de reconnaissance de texte dans l'apprentissage	92.504%	87.0813%	84.9735%
Taux de reconnaissance de texte dans les testes	90.8364%	84.7096%	85.3854%
Taux de reconnaissance des images dans l'apprentissage	97.9206%	99.6219%	68.0769%
Taux de reconnaissance des images dans les testes	97.7528%	99.4382%	69.7974%
Taux global de reconnaissance dans l'apprentissage	93.6929%	89.834%	81.3278%
Taux global de reconnaissance dans les testes	92.3683%	87.9718%	81.8747%

Tableau 4.12 : les meilleurs résultats pour la 3eme base de données

Cette expérimentation a généré des résultats moins bons que les 2 premiers classifieurs.

Synthèse :

A l'issue de cet ensemble de tests, nous notons qu'avec la 2eme base de données[6.1.1], nous obtenons un taux de reconnaissance parfait (100%), ce qui n'est pas le cas des deux autres bases. Ceci est dû au fait que la 1ere et la 3eme bases ne contiennent pas seulement des mots mais aussi des sous mots, des phrases et des paragraphes. Cette diversité engendre une reconnaissance plus compliquée.

Du point de vue descripteur, nous constatons que les moments de hu, Zernike et les moments en général ainsi que la combinaison de descripteurs simples nous donnent en général les meilleurs résultats quelque soit le type de classifieur.

Concernant le classifieur, nous observons qu'adaboost et les réseaux de neurones offrent de meilleurs résultats. Cmean et fuzzy Cmean produisent aussi de bons résultats (moins bons que les précédents) mais uniquement avec un ou deux descripteurs (les moments de zernike et fourrier). Avec les réseaux de neurones, le nombre de descripteurs qui nous donnent de bons résultats est plus réduit qu'avec AdaBoost.

En définitive, nous concluons qu'adaboost est plus performant que les réseaux de neurones puisqu'il produit de bons résultats pour la majorité des descripteurs utilisés. De plus, il offre un temps d'apprentissage beaucoup plus réduit qu'avec les réseaux de neurones.

Conclusions et perspectives

Bien que les domaines de traitement d'images et de reconnaissance de formes aient fait des progrès considérables dans différents champs d'application : reconnaissance de visages et d'empreintes digitales, le tri automatique du courrier postal et plus récemment la recherche d'information multimédia, les étapes de localisation et de caractérisation adéquates d'objets demeurent des processus très délicats et non triviaux. L'objectif de notre travail était de faire une étude bibliographique et expérimentale des différentes manières de caractériser des objets qu'on rencontre dans les images. Nous nous intéressons néanmoins à deux classes uniquement qui sont les objets **textuels** et **non-textuels**. Nous avons étudié et implémenté un très grand nombre de descripteurs de formes très connues dans la littérature qu'on a testé dans le domaine de la séparation texte/graphique. Trois jeux de test de taille très importante et de diversité considérable ont été réunis et utilisés pour la validation. Afin de mieux apprécier la robustesse des algorithmes implémentés, nous avons également codé une méthode de segmentation automatique d'image basée sur la transformée de distance. Le résultat étant l'un des trois jeux de test composé d'objets textes et objets graphiques mais séparés automatiquement évidemment avec quelques erreurs. Les résultats de validation prouvent encore une fois la robustesse de la méthode AdaBoost avec un temps d'apprentissage très appréciable. De point de vue descripteurs, nous avons noté que ce sont les moments en général qui ont donné les meilleurs résultats mais aussi la combinaison de descripteurs basiques.

Nous espérons que ce travail constituera une plate-forme de validation des caractéristiques employées en reconnaissance de forme et plus précisément en reconnaissance de l'écriture arabe manuscrite. Plusieurs points font que ce travail est original en soit :

1- Il réunit dans un même outil la majorité des descripteurs de formes qui existent.
2- Il est souple et facile à faire évoluer en ajoutant d'autres descripteurs.
3- Dans un même système, AdaBoost avec arbres de décision, RNA, C-means et Fuzzy C-means sont implantés.
4- Beaucoup de test ont été effectués, les classifieurs ont été testés sur chacun des descripteurs indépendamment et en combinaison.
5- Trois bases de données importantes d'objets textuels et graphiques ont été construites.

Toutefois, plusieurs points peuvent être enrichis dans ce travail :

- Implémenter une méthode automatique de sélection de caractéristiques.
- Combiner les différents classifieurs.
- Réaliser des tests de l'outil sur d'autres données : visage, empreintes, …

- Incorporer notre outil dans un système de recherche d'image sur le web.

Bibliographie :

[1-2-07] S. Derrode, *Représentation de formes planes à niveaux de gris par différentes approximations de Fourier-Mellin analytique en vue d'indexation de bases d'images,* Thèse de Doctorat, Université de Rennes 1, 1999.

[1-2-08] F. Ghorbel, « A complete invariant description for gray level images by the harmonic analysis approach », *Pattern Recognition Letters*, **15**, 1994, pp. 1043-1051.

[1-2-09] G. Ravichandran, M. Trivedi, « Circular-Mellin features for texture segmentation », *IEEE Trans. Image Processing*, **4**, 1995, pp. 1629-1640.

[1-2-10] E. PERSOON et K.S. FU : Shape discrimination using Fourier descriptors. *IEEE Trans. Systems, Man and Cybernetics*, SMC-7(3), mars 1977.

[1-2-13] D. ZHANG et G. LU : Study and evaluation of different Fourier methods for image retrieval. *Image and Vision Computing*, 23(1):33–49, janvier 2004.

[1-2-11] G. H. GRANLUND : Fourier Preprocessing for Hand Print Character Recognition. *IEEE Trans. on Computers*, C–21(2):195–201, 1972.

[1-2-12] C. T. ZAHN et R. Z. ROSKIES : Fourier descriptors for plane closed curves. *IEEE Trans. Computers*, C-21:269–281, 1972.

[1-2-01] Bracewell,R. N. (1978). The Fourier Transform and its Applications. Electronical and Electronic Engeneering Serie, McGraw-Hill, New York.

[1-2-05] Casasent, D. Psaltis, D. (1980). Hybrid processor to compute invariant memonts for pattern recognition. Optics Lett., 5:395-397.

[1-2-02] Grace, A. E. et Spann, M. (1991). A comparison between fourier-mellin descriptor and moment based features for invariant object recognition using neural networks. Pattern Recognition Letters, 12(10):635-643.

[1-2-03] Reddy, B. S. et Chatterji, B. (1996). An fft-based technique for translation, rotation, and scale-invariant image registration. IEEE Transactions on Image Processing, 5(8):1266-1271.

[1-2-04] Sheng, Y. et Lejeune, C. (1991). Invariant pattern recognition, San Francisco, California, Etats-Unis, pages 532-537.

[1-2-06]. Wood, J. (1996). Invariant pattern recognition : a review. Pattern Recognition, 29 (1) :1-17.

[1-2-17] T.W. RAUBER et A.S. STEIGER-GARÇÃO : Shape description by UNL Fourier features-an application to handwritten character recognition. *In 11th IAPR International Conference on Pattern Recognition, Jerusalem, Israël*, 1992.

[1-2-16] D.S Zhang and G. Lu. Generic Fourier descriptor for shape-based image retrieval. In Proceedings of the IEEE International Conference on Multimedia and Expo, volume 1, pages 425–428, 2002.

[1-2-14] F. P. Kuhl and C. R. Giardina. Elliptic Fourier Features of Closed Contours. Computer Vision, Graphics and Image Processing, 18 :236–258, 1982.

[1-2-15] T. Taxt, J. B. Olafsdottir, and M. Daehlen. Recognition of Handwritten Symbols. Pattern Recognition, 23 :1155–1166, 1990.

[1-2-18] Gabor, D. (1946). Theory of communication. Journal of the Inst. Elec. Eng., 93(26) :429-457.

[1-2-19] Wu, X et Bhanu, B. (1995). Gabor wavelets for 3d object recognition. Dans Proceedings of the 5th International Conference on Computer Vision, Cambridre, Massachussets, Etats-Unis, page 537-542.

[1-2-20] Sanger, T. (1988). Stereo disparity computation using gabor filters. Biological Cybernetics, 59 :405-418.

[1-2-21] Fleet, D. J.,Jepson, A. D., et Jenkin, M. R. M. (1991). Phase-based measurement, Computer Vision, Graphics, and Image processing. Image Undertanding, 53(2) :198-210.

[1-2-22] Grossmann, A. et Morlet, J. (1984). Decomposition of Hardy function into square integrable wavelets of constant shape. SIAM Journal on Mathematical Analysis, 15(4):723-736.

[1-2-23] Malat, S. (1989). A theory for multiresolution signal decomposition : The wavelet representation. IEE Transactions on Pätern Analysis and Machine Intelligence, 11(7) :674-693.

[1-2-24] Malat, S. (1999). A Wavelet Tour of Signal Processing. Academic Press, 1999.

[1-2-25] S. R. Deans. Applications of the Radon Transform. Wiley Interscience Publications, New York, 1983.

[1-2-26] S. Tabbone and L. Wendling. Technical Symbols Recognition Using the Two-dimensional Radon Transform. In Proceedings of the 16th International Conference on Pattern Recognition, Quebec (Canada), volume 2, pages 200–203, August 2002.

[1-2-27] S. Tabbone and L. Wendling. Binary shape normalization using the Radon transform. In Proceedings of 11th International Conference on Discrete Geometry for Computer Imagery, Naples (Italy), volume 2886 of Lecture Notes in Computer Science, pages 184–193, November 2003.

[1-2-28] K.FALCONER, "Fractal geometry: Mathematical foundation and applications", Wiley, New York, USA, 1990.

[1-4-1] A. Jain and A. Vailaya, *Image retrieval using color and shape*, Pattern Recognition **29** (1996), 1233–1244.

[1-5-01] A. Evans, N. Thacker, and J.Mayhew, *The use of geometric histogram for model-based object recognition*, in British Machine Vision Conference, 1993.

[1-5-02] E. Huet, B. & Hancock, *Structurally gated pairwise geometric histograms for shape indexing*, in British Machine Vision Conference, 1997.

[1-5-03] N. A. Thacker *Solving shapebased object recognition from a computational standpoint - practical and physiological constraints*, 2002.

[1-7-01]Duda, R. O. et Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. Commun. ACM, 15(1):11–15.

[1-7-02] D. Ballard, *Generalizing the hough transform to detect arbitrary shapes*, Pattern Recognition **13** (1981), 111–122.

[1-7-03] L. Davis, *Hierarchical generalized hough transforms and line-segment based generalized hough transforms*, PR **15** (1982), 277–285.

[1-9-01] R.C. Gonzalez, R.E. Woods, Digital Image Processing, Addison-Wesley, Reading, MA, 1992, pp. 502–503.

[1-10-01] Z. Zhang. A comparative study of fourier descriptors for shape representation and retrieval. In fifth Asian Conference on Computer Vision, volume 3, pages 646–651, 2002.

[1-11-01] S. Yang. Symbol Recognition via Statistical Integration of Pixel-Level Constraint Histograms : A New Descriptor. IEEE Transactions on PAMI, 27(2) :278–281, February 2005.

[1-12-01] F. Mokhtarian, A.K. Mackworth, “A theory of multiscale, curvature-based shape representation for planar curve”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, pp. 789–805, 1992.

[1-13-01] Garello, R. (2001). *Analyse de signaux bidimensionnels*. Collection IC2. Hermés, Paris.

[1-13-02] Alata, O. et Cariou, C. (2001). Modélisation stochastique linéaire 2-D. In Garello, R., editeur, *Analyse de signaux bidimensionnels*, chapter 2, pages 67–116. Hermès - Collection IC2, Paris.

[1-14-01] M. Zuliani, S. Bhagavathy, B. S.Manjunath, and C. S. Kenney, *Affine-invariant curve matching*, in IEEE International Conference on Image Processing, Oct 2004.

[1-14-02] A. Chalechale, A. Mertins, and G. Naghdy, *Edge image description using angular radial partitioning*, IEE Proceedings-Vision Image and Signal Processing **151** (2004), 93–101.

[1-14-03] O. Carmichael and M. Hebert, *Object recognition by a cascade of edge probes*, in BMVC02, 2002, p. Matching/Recognition.

[1-14-04] O. Carmichael and M. Hebert, *Shape-based recognition of wiry objects*, IEEE Trans. on Pattern Analysis and Machine Intelligence **26** (2004), 1537–1552.

[1-15-01] S. Belongie and J. Malik. Matching with shape contexts. In *IEEE Workshop on Content-Based Access of Image and Video Libraries*, June 2000.

[2-2-06] Abu-Mostafa, Y. S. and Psaltis, D., Image Normalisation by Complex Moments, *IEEE Trans. on PAMI*, **7**, pp. 46–55, 1985.

[2-2-07] Y.S. ABU-MOSTAFA et D. PSALTIS : Recognitive aspects of moment invariants. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 6(6):698–706, novembre 1984.

[2-2-08] R. Bamieth, R. De Figueiredo, « A general moments invariants/attributed graph method for the three dimensional object recognition from a single image », *IEEE Journal of Robotics Automation*, **2**, 1986, pp. 240-242.

[2-2-09] S. O. Belkasim, M. Shridar, and M. Ahmadi. Pattern Recognition with Moment Invariants : A Comparative Study and New Results. Pattern Recognition, 24 :1117–1138, 1991.

[2-2-10] Boyce, J. F. and Hossack, W. J., Moment Invariants for Pattern Recognition, *Pattern Recog. Lett.*, **1**, pp. 451–456, 1983.

[2-2-11] M.E. Celebi and Y.A. Aslandogan, "A Comparative Study of Three Moment-Based Shape Descriptors", *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing*, 1, p. 788--793, 2005.

[2-2-01] Y. Chen, N.A. Langrana, A.K. Das, « Perfecting vectorized mechanical drawings », *Computer Vision and Image Understanding*, **63** (2), 1996, pp. 273-286.

[2-2-12] J. FLUSSER : On the independence of rotation moment invariants. *Pattern Recognition*, 33(9):1405–1410, septembre 2000.

[2-2-13] J. FLUSSER et T. SUK : Construction of complete and independent systems of rotation moment invariants. *In 10th International Conference on Computer Analysis of Images and Patterns, Groningen, Pays-Bas*, pages 41–48, 2003.

[2-2-01] M. K. Hu.: “Visual pattern Recognition by Moment Invariants”. In IRE Transactions on Information Theory, IT-8:179-187, 1962.

[2-2-15] H.K. KIM et J.D. KIM : Region-based shape descriptor invariant to rotation, scale and translation. *Signal Processing : Image Communication*, 16(1-2):87–93, septembre 2000.

[2-2-04] A. Khotanzad, Y. H. Hong, « Rotation invariant image recognition using features selected via a systematic method », *Pattern Recognition*, **23**, 1990, pp. 1089-1101.

[2-2-05] Khotanzad, A. and Hong, Y. H., Invariant Image Recognition by Zernike Moments, *IEEE Trans. on PAMI*, **12**, pp. 489–498, 1990.

[2-2-16] S.X. Liao, M. Pawlak, « On the accuracy of Zernike moments for image analysis », *IEEE Trans. on PAMI*, **20** (12), 1998, pp. 1358- 1364.

[2-2-17] S. X. Liao, M. Pawlak. “Image Analysis with Moment Descriptor”, 1995.

[2-2-03] M. Teague, « Image analysis via the general theory of moments », *Journal of Optical Society of America*, **70**, 1980, pp. 920-930.

[2-2-18] C.H. TEH et R.T. CHIN : On image analysis by the methods of moments. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10(4):496–513, juillet 1988.

[2-2-19] D. Zhang: “Image Retrieval Based on Shape”. In thèse de doctorat Monash University,

[2-2-20] F. ZERNIKE : Diffraction theory of the cut procedure and its improved form, the phase contrast method. *Physica*, 1:689–704, 1934.

[2-4-01] Kavallieratou, E., Fakotakis, N. et Kokkinakis, G. (2002). Handwritten character recognition based on structural characteristics. icpr, 03:30139.

[2-4-02] A. Goshtasby, *Description and discrimination of planar shapes using shape matrices*, IEEE Trans. On Pattern Analysis and Machine Intelligence **7** (1985), 738–743.

[2-4-03] A. Taza and C. Suen, *Discrimination of planar shapes using shape matrices*, IEEE Trans. on Systems, Man and Cybernetics **19** (1989), 1281–1289.

[2-4-04] M. Zuliani, S. Bhagavathy, B. S.Manjunath, and C. S. Kenney, *Affine-invariant curve matching*, in IEEE International Conference on Image Processing, Oct 2004.

[2-4-05] I. Sirovich and M. Kirby. Low-dimensional procedure for the characterisation of human faces. *J. Opt. Soc. Am*, 4(3) :519.524, 1987.

[2-4-07] J. Coutaz, J. L. Crowley, F. Berard, and D. Salber. Eigenspace coding as a means to support privacy in computer mediated communication. *Interact*, 1997.

[2-4-08] O. Chomat, V. Colin de Verdiere, and J. L. Crowley. Recognizing gold_sh ? or local scale selection for recognition technique. In *International Symposium for Intelligent Robotics System*, pages 197.206, 1999.

[2-4-06] L. Paletzta, M. Prantl, and A. Pinz. Reinforcement learning for autonomous threedimensional object recognition. In *Symposium on Intelligent Robotics Systems, SIRS 98*, pages 63.81, Edinburgh, United Kingdom, 1998.

[2-4-09] F. Pourraz. Estimation de position d'un robot mobile par projection dans un espace de composantes principales. Master's thesis, ENSIMAG, 1998.

[2-4-10] N. Winters and J. Santos-Victor. Omni-directional visual navigation. In *International Symposium for Intelligent Robotics System*, pages 109.118, 1999.

[2-4-11] H. Murase and S. Nayar. Visual learning and recognition of 3d objects from appearance. *International Journal on Computer Vision*, 1995.

[2-4-12] W.-Y. Kim and Y.-S. Kim. A new region-based shape descriptor. In TR 15-01, Pisa (Italy), December 1999.

[2-4-13] A.K. JAIN : Fundamentals of digital image processing. *In Prentice Hall*, 1989.

[3-1-01] M. Brady. Seeds of perception. Dans Proceedings of the 3rd Alvey Vision Conference, pages 259{265, 1987.

[3-1-02] L. Dreschler et H.-H. Nagel. Volumetric model and 3D trajectory of a moving car derived from monocular tv frame sequences of a street scene. Computer Graphics and Image Processing, 20: 199{228, 1982.

[3-1-03] Z. Zhang, R. Deriche, O. Faugeras et Q.-T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. Arti_cial Intelligence, 78: 87{119, 1995.

[3-1-08] Florack et al., 1994 General intensity transformations and differential invariants. Journal of Mathematical Imaging and Vision, 4(2):171-187.

[3-1-05] Freeman et Adelson, 1991 the design and use of steerable filters. IEEE Transactions on pattern Analysis and Machine Intelligence, 13(9):891-906.

[3-1-09] ter Haar Romeny, 1996 Introduction to scale-space theory: multiscale geometric image analysis. Rapport technique ICU-96-21, Utrecht University.

[3-1-04] Koenderink et Van Doorn, 1987 Representation of local geometry int the visual system. Biological Cybernetics, 55 :367-375.

[3-1-06] Rao et Ballard, 1995 Object indexing using an Iconi Sparse Distributed Memory. Proceedings of 5th International Conference on Computer Vision, pages 24-31.

[3-1-07] Salden et al., 1992 A complete and irreductible set of local orthogonally invariant features of 2-dimensional images. Proceedings of 11th International Conference on Pattern Recognition, pages 180-184.

[3-1-10] D. Gabor. Theory of communication. Proceedings of Inst. Elec. Eng., 93(26): 429{441, 1946.

[3-1-11] A. Grossmann et J. Morlet. Decomposition of Hardy functions into square integrable wavelets of constant shape. SIAM J. Math., 15: 723{736, 1984.

[3-1-12] Y. Meyer. Ondelettes et fonctions splines. Dans Sem. Equations aux Derivees Partielles, Ecole Polytechnique, Paris, 1986.

[3-1-13] S. G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 11(7): 674{693, 1989.

[3-1-14] L. Jacobson et H. Wechsler. Derivation of optical ow using a spatio temporal frequency approach. Computer Vision, Graphics and Image Processing, 38: 29-65, 1991.

[3-1-15] Lowe D.G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2) :91–110, 2004.

[3-1-16] Lindeberg, T. (1998). Feature detection with automatic scale selection. Int. J. Comput. Vision, 30(2):79–116.

[3-1-17] Matas, J., Chum, O., Urban, M. et Pajdla, T. (2004). Robust wide baseline stereo from maximally stable extremal regions. IVC, 22(10):761–767.

[3-1-18] S. Lazebnik, C. Schmid and J. Ponce, "A Sparse Texture Representation Using Local Affine Regions", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27, 8, p. 1265--1278, 2005.

[3-1-19] Bay H., Tuytelaars T., and Van Gool L.J. Surf : Speeded up robust features. In *European Conference on Computer Vision*, volume 1, pages 404–417, 2006.

[3-1-20] Brown M. and Lowe D. Invariant features from interest point groups. In *Proceedings of the BritishMachine Vision Conference*, 2002.

[3-1-21] Ros J. and Laurent C. Description of local singularities for image registration. In

ICPR, volume 4, pages 61–64, 2006.

[3-1-22] Mallat S. Foveal Approximations for Singularities. *Applied and Computational*

Harmonic Analysis, 14(2) :133–180, 2003.

[4-1-01] J. Zhang and T. Tan, “Brief review of invariant texture analysis methods”, *Pattern Recognition*, vol. 35, pp. 735-747, 2002.

[4-1-05] Pentland, A. (1984). Fractal based description of natural scenes. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 6 :661–674.

[4-1-06] R. M. Haralick, "Statistical and structural approaches to texture", *Proceedings of the IEEE*, 67, 5, p. 786-804, 1979.

[4-1-07] J.A. McLaughlin and J. Raviv, "Nth-order autocorrelations in pattern recognition", *Information and Control*, 12, 2, p. 121--142, 1968.

[4-1-08] Haralick, R. M., Shanmugam, K. and Dinstein, I., Textural Features for Image Classification, *IEEE Trans. on Systems, Man and Cybernetics*, **2**, pp. 610–621, 1973.

[4-1-09] Sandra Herlidou. Caracterisation tissulaire en Imagerie par Resonnance Magnetique Nucleaire par l'analyse de texture : etude du tissu musculaire et de tumeurs intracraniennes. PhD thesis, Universite de Rennes 1, 1999.

[4-1-10**]** BRUCE, GREEN. *« La perception Visuelle : Physiologie, psychologie et écologie »*. Presse universitaire de Grenoble, 1993, 431p.

[4-1-13] S Bres, Contributions à la quantification des critères de transparence et d’anisotropie par une approche globale. PhD Thesis, 1994.

[4-1-14] W.K. PRATT. *Digital Image Processing (Book : First Edition)*. Wiley, 1978.

[4-1-15] David HARWOOD, Timo OJALA, Matti PIETIK, Shalom KELMAN, and Larry DAVIS. Texture classification by center-symmetric auto-correlation, using kullback discrimination of distributions. *Pattern Recogn. Lett.*, 16(1):1-10, 1995

[4-1-16] B. Schiel and J. L. Crowley. Object recognition using ultidimensional receptive field histograms. In *Proc. of ECCV'96*, 1996.

[4-1-17] B. Schiel and J. L. Crowley. Probabilistic object recognition using multidimensional receptive field histograms. In *Proc. of ICPR'96*, 1996.

[4-1-18] B. Schiele and J. L. Crowley. Object recognition without correspondence using multidimensional receptive field histograms. *International Journal of Computer Vision*, 36(1) :31.50, 2000.

[4-1-19] S.L. Tanimoto, "An Optimal Algorithm for Computing Fourier Texture Descriptors", *IEEE Trans. Comput*, 27, 1, p. 81--84, 1978.

[4-1-20] R. Bajcsy and Lieberman, "Texture gradient as a depth cue", *Computer graphics and image processing*, 5, 1976.

[4-1-21] M. E. Jernigan and F. D'Astous, "Entropy-based texture analysis in the spatial frequency domain", *IEEE Trans. Pattern Anal. Mach. Intell*, 6, 2, p 237--243, 1984.

[4-1-22] C.C. Chen and C.C. Chen, “Filtering methods for texture discrimination”, *Pattern Recognition Letters*, vol. 20, pp. 783-790, 1999.

[4-1-23] T. Randen and J.H. Husøy, “Multichannel filtering for image texture segmentation”, *Optical Engineering*, vol. 33, pp. 2617-2625, 1994.

[4-1-24] J. L. Starck, M. Elad & D. L. Donoho. *Redundant multiscale transforms and their application for morphological component analysis*. AIEP, page 132, 2004.

[4-1-25] F. G. Meyer & R. R. Coifman. *Brushlets : a tool for directional image analysis and image compression*. Applied and Computational Harmonic Analysis, pages 147–187, 1997.

[4-1-26] L. Ying & L. Demanet. *Wave atoms and sparsity of oscillatory patterns*. Applied and Computational Harmonic Analysis, pages 368–387, 2007.

[4-1-27] H. Tamura, S. Mori, and T. Yamawaki. Texture features corresponding to visual perception. IEEE transaction on Systems, Man, Cybernetics, 8(6) :460–473, 1978.

[5.1.1] : Morgan J., Sonquist J.A., *Problems in the Analysis of Survey Data, and a Proposal*, Journal of the American Statistical Association, 58:415-435, 1963.

[5.1.2] ; Morgan J., Messenger R., *THAID-a sequential analysis program for the analysis of nominal scale dependent variables*, Survey Research Center, U of Michigan, 1973.

[5.1.3] : Kass G., *An exploratory technique for investigating large quantities of categorical data*, Applied Statistics, 29(2), 119-127, 1980.

[5.1.4] : Breiman L, Friedman J., Olshen R., Stone C., *Classification and Regression Tree*, California: Wadsworth International, 1984.

[5.1.5] : Quinlan R., *C4.5: Programs for Machine Learning*, Morgan Kaufman, 1993.

[5.2.1] : Ø. Due Trier et T. Taxt. Improvement of “Integrated Function Algorithm” for Binarization of Document Images. *Pattern Recognition Letters*, 16:277–283, mars 1995. (p. 23)

[5.2.2] R. deriche, "Using Canny’s criteria to derive a recursively implemented optimal edge detector", Int. Journal of Computer Vision, vol. 1, n°2, pp.167-187, 1987.

[5.2.3] W. Niblack. *An Introduction to Digital Image Processing*, pages 115–116. Englewood Cliffs, N.J. : Prentice Hall, 1986.

[5.2.4] : Minkowski H. “Volumen und Oberfläche” *Math. Ann.* 1903, 57, 447-495.

[5.2.5] : G. Matheron « Les variables régionalisées et leur estimation » Masson Paris 1965

[5.2.6] : J. Serra « Introduction à la morphologie mathématique » Cahiers du Centre de Géostatistique et Morphologie Mathématique. Ecole des Mines de Paris. Fontainebleau 1969, N°3.

[5.2.7] : H. Hadwiger « Vorslesungen über Inhalt, Oberfläche und Isoperimetrie » Springer Verlag Berlin 1957.

[5.2.8] : G. Sanniti di Baja. Well-Shaped, Stable, and Reversible Skeletons from the (3,4)-Distance Transform. *Journal of Visual Communication and Image Representation*, 5(1):107–115, 1994. (pp. 50, 51)

[5.2.9] : C. Arcelli et G. Sanniti di Baja. A Width-Independent Fast Thinning Algorithm. *IEEE Transactions on PAMI*, 7(4):463–474, 1985. (p. 50)

[5.2.10] : C. Arcelli et G. Sanniti di Baja. A One-Pass Two-Operation Process to Detect the Skeletal Pixels on the 4-Distance Transform. *IEEE Transactions on PAMI*, 11(4):411–414, 1989. (p. 50)

[6.1.1] PECHWITZ M., MADDOURI S. S., MÄERGNER V., ELLOUZE N., , AMIRI H., IFN/ENIT-DATABASE OF HANDWRITTEN ARABIC WORDS, *CIFED*, 2002.

[6.1.2] MÄRGNER V., PECHWITZ M., , ABED H. E., Arabic Handwriting Recognition Competition, *ICDAR*, 2005, pp. 70–74.

[6.1.3] MÄRGNER V., ABED H. E., Arabic Handwriting Recognition Competition, *ICDAR*, 2007, pp. 1274-1278.

[6.1.4] L. A. Fletcher and R. Kasturi. A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images. IEEE Transactions on PAMI, 10(6):910_918, 1988.

[6.1.5] J. M. Gloger. Use of Hough Transform to Separate Merged Text/Graphics in Forms. In Proceedings of 11th International Conference on Pattern Recognition, Den Haag (The Netherlands), volume 2, pages 268_271, 1992.

[6.1.6] T. Kaneko. Line Structure Extraction from Line-Drawing Images. Pattern Recognition, 25(9):963_973, 1992.

[6.1.7] D. X. Le, G. R. Thoma, and H.Wechsler. Classi_cation of binary document images into textual or nontextual data blocks using neural network models. Machine Vision and Applications, 8:289_304, 1995.

[6.1.8] Z. Lu. Detection of Text Regions From Digital Engineering Drawings. IEEE Transactions on PAMI, 20(4):431_439, April 1998.

[6.1.9] H. Luo and I. Dinstein. Using Directional Mathematical Morphology for Separation of haracter Strings from Text/Graphics Image. In Shape, Structure and Pattern Recognition (Post-proceedings of IAPR Workshop on Syntactic and Structural Pattern Recognition, Nahariya, Israel), pages 372_381. World Scienti_c, 1994.

[6.1.10] Huizhu Luo and Rangachar Kasturi. Improved Directional Morphological Operations for Separation of Characters from Maps/Graphics. In K. Tombre and A. K. Chhabra, editors, Graphics Recognition_Algorithms and Systems, volume 1389 of Lecture Notes in Computer Science, pages 35_47. Springer-Verlag, April 1998.

[6.1.11] T. Pavlidis and J. Zhou. Page Segmentation and Classi_cation. CVGIP: Graphical Models and Image Processing, 54(6):484_496, November 1992.

[6.1.12] K. Y. Wong, R. G. Casey, and F. M. Wahl. Document Analysis System. IBM Journal of Research and Development, 26(6):647_656, 1982.

[6.1.13] http://www.lems.brown.edu/~dmc/main.html.

[2.5.1] R.E. Schapire. The strength of weak learnability. Machine Learning, 5(2):197–227, 1990.

[2.5.2] R. E. Schapire and Y. Singer. Improved Boosting Algorithms Using Confidencerated Predictions. In Proceedings of the llth Annual Conference on Computational Learning Theory, 1998.

[2.5.3] Friedman J, Hastie T, Tibshirani R (1998). Additive Logistic Regression: a Statistical View of Boosting. Stanford University.

[2.5.4] Y. Freund et R.E. Schapire : A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(1):119–139, 1997.

[2.5.5] J. R. Quinlan. C4-5: Programs for Machine Learning. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1993.

[2.5.6] Hastie, T., Tibshirani, R., Friedman, J. H. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer Series in Statistics. 2001.

[2.5.7] Y. Freund, An adaptive version of the boost by majority algorithm, in Proc. of the Twelfth Annual Conference on Computational Learning Theory, 1999.

[2.5.8] Y. Freund and R. Schapire. A short introduction to boosting. Journal of Japanese Society for Artificial Intelligence, 14(5) :771–780, 1999.

[2.5.9] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. Information and Computation, 108(2):212–261, 1994.