D'une pondération automatique des caractéristiques des graphèmes à la création des CodeBooks, un nouveau point de vue dédié aux applications CBIR

Daher Hani, Djamel Gaceb, Véronique Eglin, Stephane Bres, Nicole Vincent

D'une pondération automatique des caractéristiques des graphèmes à la création des CodeBooks, un nouveau point de vue dédié aux applications CBIR Sur l'écriture manuscrite

H. Daher[1], DJ. Gaceb[1], V. Eglin[1], S. Bres[1],N. Vincent [2]

[1] LIRIS (Laboratoire d'Informatique en Image et Systèmes d'information)
UMR 5205 CNRS, INSA de Lyon, F-69621 Villeurbanne cedex
[2] LIPADE (Laboratoire d'Informatique Paris Descartes)
Université Paris Descartes, F-75270 Paris cedex

{hani.daher, djamel.gaceb1veronique.eglin, stephane.bres}@insa-lyon.fr
Nicole.Vincent@mi.parisdescartes.fr

Résumé

Nous présentons dans cet article un nouveau mécanisme de construction des codebooks à partir des graphèmes issus de la décomposition de l'écriture manuscrite. Ces derniers sont importants pour simplifier ultérieurement l'automatisation de l'analyse, de la transcription de ces manuscrits et de la reconnaissance de styles ou de scripteurs. Notre approche apporte d'une part une sélection précise des descripteurs de graphèmes par algorithmes génétiques et d'autre part une méthodologie performante pour la catégorisation de la forme des graphèmes en utilisant la coloration de graphes. Nous montrons en quoi le couplage de ces deux mécanismes « sélection-classification » permet d'offrir une meilleure séparation des formes à catégoriser en exploitant leurs particularités grapho-morphologiques, leurs densités et leurs orientations significatives.

Mots Clef

Reconnaissance de forme, analyse des manuscrits anciens, algorithme génétique, CBIR de styles, poids génériques, coloration des graphes.

Abstract

We present in this paper a new mechanism for the construction of codebooks based on graphemes that are issued from the decomposition of manuscripts. These codebooks are important to simplify the automation of the analysis, the manuscripts transcription and the recognition of styles or writers. On one hand our approach provides a precise descriptors selection by genetic algorithms and on the other hand a high-performance methodology for the categorization of the shapes of graphemes by using graph coloring. We show how this coupling of these two mechanisms "selection-classification" can offer a better separation of the forms to categorize by exploiting their grapho-morphological, their density and their significant orientations particularities.

Keywords

Pattern recognition, manuscripts analysis, genetic Algorithm, CBIR of styles, generic weights, graph coloring.

1 Introduction

La valeur d'un manuscrit ne tient pas forcément à ses qualités esthétiques ou à un contenu exceptionnel. Aujourd'hui un nouveau regard est porté sur le manuscrit ancien comme témoin d'un ensemble de techniques et de pratiques comme la diversité des règles d'exécution des écritures, la mise en page, le style d'écriture, etc. Notre travail s'inscrit dans le cadre du projet ANR GRAPHEM. Il représente une contribution méthodologique applicable à l'analyse informatisée de l'écriture manuscrite latine du Moyen Âge et plus généralement à tout type d'écritures. Ce type d'analyse peut offrir un angle de vue plus large révélant sur un manuscrit le plus banal qui soit, une source d'informations très riche pour le paléographe.

Cet article présente un nouveau mécanisme de construction des codebooks à partir des graphèmes[1] issus d'un découpage adapté à l'écriture latine ancienne [2]. Cette catégorisation de la forme des graphèmes représente une base pour toute informatisation de l'analyse, de la reconnaissance de style ou de scripteur et même de la transcription.

Nous avons fait le choix de proposer une approche de construction de codebook[2] faisant usage d'une procédure de sélection de caractéristiques, non seulement pour réduire l'espace de stockage des descripteurs mais surtout pour améliorer les performances du classifieur et justifier le modèle donné de caractéristiques. Nous avons ensuite montré comment ces codebooks peuvent être utilisés, à leur tour, en tant que signatures témoignant une variabilité entre les styles d'écriture. Ce travail se présente en trois parties cohérentes et interactives :

a) Pondération non supervisée des caractéristiques pour

[1] Graphème : La plus petite forme élémentaire distinctive de l'écriture
[2] CodeBook : Listes des catégories des graphèmes : dictionnaire

une représentation pertinente et générique de la forme des graphèmes. Cette partie est essentiellement basée sur les algorithmes génétiques et l'évaluation non supervisée de résultats de la classification.

b) Classification robuste et modulable par coloration de graphe de l'ensemble de graphèmes obtenus sous la forme d'un codebook (table de similarités de graphèmes).

c) Application des codebooks dans la reconnaissance des styles basée sur la technique de CBIR[3]. Dans cette partie nous montrons le passage d'une description locale des graphèmes vers une description globale d'une page manuscrite (codebook). Nous montrerons également comment les deux premières parties peuvent coopérer ensemble pour offrir une meilleure description et classification de graphèmes. Nous présentons, dans la section 2, les méthodes existantes de sélection de caractéristiques et de construction de codebooks. Nous présentons ensuite notre nouveau mécanisme de construction de codebook ainsi que nos démarches de validation.

2 Travaux antérieurs

2.1 Choix des caractéristiques

Choisir un ensemble de descripteurs discriminants d'un ensemble de formes en vue de leur reconnaissance ou de leur classification n'est pas aisé, du fait de la grande variété des choix présentés dans la littérature. C'est la raison pour laquelle plusieurs méthodes de sélection de caractéristiques ont été développées. Elles reposent, selon le critère de séparabilité des caractéristiques, sur deux types de techniques: les filtres [1] et les wrappers [3]. Dans l'approche de filtrage, la performance de chacune des caractéristiques est évaluée pour sélectionner, ensuite, un sous-ensemble des meilleures caractéristiques qui offrent les scores les plus élevés. Dans ce cas, l'évaluation des caractéristiques n'utilise pas forcément une étape de classification, elle repose le plus souvent sur l'analyse de différentes mesures comme la distance, l'entropie ou certains coefficients de corrélation. Les approches dites wrappers repose sur une évaluation qui dépend des taux d'erreur d'une classification et, parfois, d'un apprentissage. Différentes variétés existent pour ce type d'approche on peut citer: forward selection, backward selection, Forward-backward selection, tree harvesting et les algorithmes génétiques (AG). Ce type d'approche offre plus de fiabilité, par rapport au premier type, dans la mesure où la qualité d'une sélection est fortement justifiée par une meilleure classification et reconnaissance. C'est la raison pour laquelle nous nous sommes intéressés, dans notre étude, aux AG qui font partie de ces méthodes.

2.2 Techniques existantes pour extraire les tables de similarités

Schomaker et al [10] ont utilisé des codebooks qui listent les catégories de formes des composantes connexes des contours. Aucune démarche n'était faite pour réduire la sensibilité de la méthode à la dégradation d'encre très récurrente dans les manuscrits médiévaux et qui conduit à des contours discontinus. Dans [11] le codebook a été construit à partir de fragmentation implicite des traits en utilisant une fenêtre de taille fixe glissante et centrée sur le squelette. Aucune règle de décomposition n'était formulée, à savoir les points de jonction, de croisement et les minima locaux qui sont nécessaires pour toute adaptation à la dynamique de l'écriture. La majorité des méthodes présentées dans la littérature utilise une des variantes de la méthode k-means [4] pour construire le codebook. Khorsheed [5] a utilisé une technique différente basée sur les *MMC* pour former un codebook destiné à la transcription des manuscrits arabes. Gilliam et al, dans [6] ont utilisé les *KPPV* pour former un codebook à partir de manuscrits médiévaux à destination de l'identification des scripteurs. Toutes ces méthodes classiques nécessitent la connaissance a priori du nombre de classes et ne possèdent pas un module de sélection de caractéristiques. C'est la raison pour laquelle ces méthodes ne peuvent pas être utilisées sur une large variété de manuscrits comme celle traitée dans le cadre de notre projet. Le nombre de catégories de graphèmes significatifs est très variable d'un extrait manuscrit à un autre. Nous présentons dans la section suivante notre nouvelle méthode de construction des codebooks indépendante au nombre de classes et une auto-pondération des caractéristiques par AG interagissant avec une classification par coloration de graphe.

3 Notre approche de construction de codebooks

Les différentes étapes de notre démarche de clustering de graphèmes sont présentées par le synoptique la figure 1.

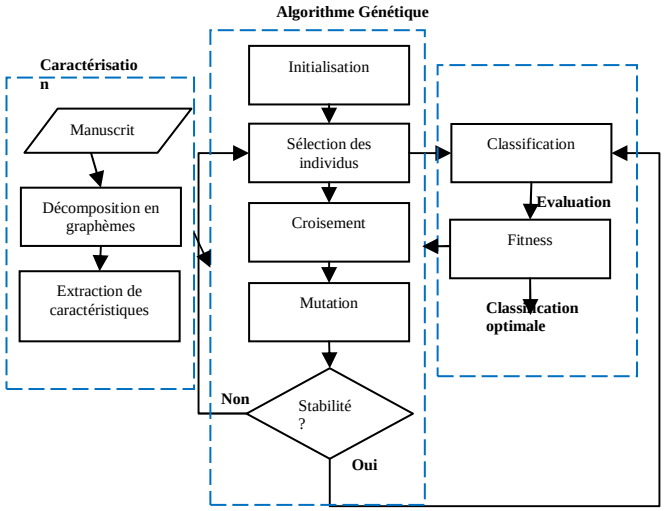

Figure 1. Processus de décomposition et sélection de caractéristiques

[3] CBIR :Content based image retrieval

Chaque graphème est représenté par un ensemble de caractéristiques. On utilise l'AG comme un modèle pour le choix automatique du meilleur sous ensemble de caractéristiques d'un graphème et le meilleur seuil de classification. Le seuil de clustering est une valeur qui contrôle le calcul des similarités entre un graphème et un cluster représenté par son graphème dominant (le graphème le plus éloigné de tous les autres clusters). Pendant l'évolution de l'optimisation (rendu visible par une fonction de fitness qui sera détaillée plus tard), le module AG coopère en permanence avec le module de classification de la façon suivante : a) En entrée, l'algorithme génétique envoie au classifieur différentes générations de paramètres (le sous ensemble de caractéristiques et le seuil de classification par coloration). b) A chaque génération de paramètres, le classifieur va classer les graphèmes en groupes induisant un codebook. c) La qualité du résultat de chaque classification est évaluée par une fonction de fitness. d) Cette fonction va à son tour alimenter l'algorithme génétique pour lui permettre de produire une nouvelle génération de paramètres qui sera acheminée à nouveau vers l'entrée du classifieur. e) Ce processus d'échanges mutuels entre les deux modules (AG/Classifieur) sera répété jusqu'à la stabilisation ou jusqu'à l'aboutissement d'un nombre d'itérations maximal (défini en général par l'utilisateur).

3.1 Description des graphèmes et seuil

Initialement, nous produisons une description vectorielle de chacun des graphèmes. Cette description sert à la mesure des similarités entre les graphèmes en vue de leur classification. *n* (*n=59)* caractéristiques sont utilisées pour décrire la forme des graphèmes et groupées en *11* descripteurs de la manière suivante : *{D1=Hauteur, D_2=Largeur, D_3=Excentricité, D_4=Densité Globale, D_5= Direction, D_6=Périmètre, D_7=Rapport fitness, D_8=Compacité, D_9 = 9Densités, D_{10}= Huit orientations, D_{11}=Moments de Zernike}*. Ces descripteurs ont été choisis car ils désignent des dimensions compréhensibles pour les experts paléographes et peuvent donc être directement interprétés. Cela va produire 2^n combinaisons pour les caractéristiques ou descripteurs.

L'algorithme de coloration de graphes [16] utilise un seuil d'adjacence. Ce dernier peut varier de 3 à 90, converti en binaire avec une représentation chromosomique sur 7 bits ce qui va produire 2^7 combinaisons possibles. Et c'est ici que l'algorithme génétique de sélection de caractéristiques joue pleinement son rôle pour résoudre ce problème de combinaison optimale.

3.2 Sélection et pondération de caractéristiques par algorithme génétique

Dans notre étude, on était confronté à deux problèmes: a) l'ajustement du seuil d'adjacence pour la coloration de graphes, b) La sélection non supervisée d'un sous-ensemble de caractéristiques calculées sur les graphèmes offrant la meilleure représentation du codebook des graphèmes. Le choix du seuil influence le nombre de classes produites par la coloration de graphes alors que la sélection des caractéristiques identifie les variables discriminantes qui nous permettent de réaliser une bonne séparation des graphèmes en classes. Nous présentons dans les paragraphes suivants nos démarches pour résoudre ces deux problèmes d'optimisation par l'algorithme génétique.

3.3 Représentation chromosomique

La première étape consiste à modéliser les variables entrant dans l'optimisation de la classification, à savoir les 59 caractéristiques et la variable seuil de coloration (avec $S \in [3,90]$). A chaque variable d'optimisation, nous faisons correspondre un bit binaire. Nous appelons chromosome une séquence de bits qui code l'ensemble de ces variables de la classification (voir la figure 2). Cette codification se fait en 66(= 59+7) bits pour la sélection de caractéristiques et le seuil. Ceci programme en deux parties indépendantes l'activation ou la désactivation de chacun des 59 premières caractéristiques ainsi que la variation de seuil de la coloration (sur les sept derniers bits). Sur les 59 premiers bits, un bit qui vaut 1 signifie que la caractéristique correspondante est sélectionnée pour la classification, quand il vaut 0 cette caractéristique ne sera pas sélectionnée. Nous avons fait le choix de sélectionner les caractéristiques avant de les pondérer afin d'évaluer dans un premier temps la contribution effective de chacune d'elles au sein d'une classification et relever des points de concordance éventuelle entre les résultats sur des images d'écriture fortement similaires (d'un point de vue morphologique).

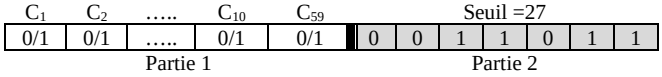

Figure 2.Codage chromosomique des paramètres en deux parties indépendantes.

Un meilleur choix de seuil d'adjacence qui influence la coloration de graphe conduit à une séparation optimale des classes (le nombre de classes le sera aussi) offrant une meilleure construction de codebook. Par conséquent, une faible valeur de ce seuil conduit à une sous-segmentation alors qu'une valeur élevée amène à une sur-segmentation des classes. Dans ces deux cas les taux de confusion et de fausse reconnaissance deviennent importants.

3.4 Initialisation et génération de la population initiale

Le choix de la population initiale (génération 0) est important car il peut rendre plus ou moins rapide la convergence vers l'optimum global de la classification de graphèmes. Le démarrage est initié avec l'a priori fort qu'on ne connait rien du problème à résoudre (pas d'a priori sur les résultats de classification) : nous avons donc

réparti la population initiale sur tout le domaine de recherche respectant la diversité entre les individus (caractéristiques et seuils). Cela permet de converger dans des temps raisonnables et ne pas privilégier certaines configurations de chromosomes. Chaque génération est donc composée de *50* chromosomes, avec une probabilité de croisement de *0.8* et une probabilité de mutation de *0.001*. Ces valeurs ont été fixées d'une manière expérimentale. Si deux chromosomes identiques existent dans la population initiale, un des deux sera régénéré.

3.5 Evaluation de la qualité de classification du codebook

La fonction d'évaluation de la qualité de la classification appelée fitness (notée *F*) est utilisée pour sélectionner et reproduire les meilleurs individus de la population (ensembles des caractéristiques et le seuil de coloration) qui maximisent la qualité de classification des graphèmes. A cette fin, nous avons utilisé la mesure proposée dans [16] qui combine l'uniformité intra-classes et la disparité inter-classes de la façon suivante :

$$F = M_{inter\text{-}groups} + M_{intra\text{-}groups}$$

Où $M_{inter\text{-}groups}$ représente la somme des dissimilarité entre les clusters C_i pondéré par leur mesure w_i. Soit :

$$M_{inter-classes} = \frac{\sum_{ci} wici}{\sum_{ci} wi}$$

Le poids associé à chaque cluster est évalué par le nombre de graphèmes dans chaque cluster.

Et c_i est la disparité inter-classes pour le cluster C_i définie par: $c_i = \sum_{c_j \in c \mid j \neq i} p_{ij} c_{ij}$ où $p_{ij} = l_{ij} / l_i$

avec l_i nombre de graphèmes de cluster C_i et l_{ij} est le nombre de graphèmes qui forment la frontière commune entre les clusters C_i *et* C_j. On peut le déduire du nombre de graphèmes qui font partie de ces deux clusters mais éloignés par une faible distance (plus petite que *0.7*). Nous formulons le contraste entre les clusters C_i et C_j par:

$$c_{ij} = \frac{1}{l_i \bullet l_j} \sum_{v_x \in c_i} \sum_{v_y \in c_j} Ds\ (v_x, v_y)$$

Ds est la distance entre deux graphèmes v_x et v_y. $M_{intra\text{-}groups}$ calcule la somme des variances normalisées des classes (réduit à 1). Cette fitness nous permet d'obtenir une classification que nous qualifions d'optimale d'une manière non supervisée et de considérer comme optimal le jeu des paramètres (caractéristiques et seuil) qui a donné lieu à cette classification.

3.6 Sélection des individus, croisement et mutation

A chaque génération, des individus se reproduisent, survivent ou disparaissent de la population sous l'action d'un opérateur de sélection. Après avoir trié de manière décroissante selon la fitness, les individus de la population P_0, on utilise la technique de sélection par tournois. Cette méthode est celle avec laquelle on obtient les résultats les plus satisfaisants par rapport à la technique basée sur l'élitisme ou la roulette. On sélectionne par tournois les deux meilleurs individus de cette population dont on a besoin pour reproduire la nouvelle génération par croisement en deux points sur chacune des deux parties. En répète ce processus *n* fois de manière à obtenir les *n* individus des *P'* qui serviront de parents. L'opérateur de croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Le croisement est appliqué sur deux parents (P_1 et P_2) et génère deux enfants (E_1 et E_2) (figure3).

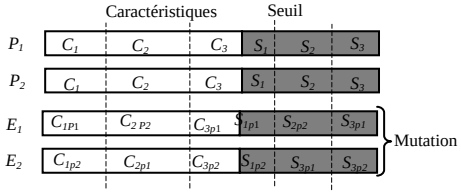

Figure 3.Principe de croisement entre deux chromosomes

Par la suite, on applique une mutation sur chacune des deux parties du chromosome (dans la pratique la probabilité de mutation utilisée est de *0.001*). L'opérateur de mutation a pour but de garantir l'exploration de l'espace d'états. Cela signifie que l'algorithme génétique sera susceptible d'atteindre tous les points de l'espace d'état, sans pour autant les parcourir tous dans le processus de résolution.

3.7 Critère d'arrêt

Le critère d'arrêt est atteint quand la fonction de fitness devient stable ou bien quand on atteint un nombre maximal de générations (fixé à *1000*). Dès la $n^{ème}$ génération, si la fonction de fitness reste stable encore durant *50* générations, le processus de calcul des générations suivantes est arrêté. On conserve alors l'ensemble des caractéristiques ayant conduit à la meilleure fonction de fitness.

4 Construction du codebook par coloration de graphes

La coloration de graphes constitue une branche très importante de la théorie des graphes. Ses applications sont nombreuses dans différents domaines scientifiques (optimisation des réseaux de transport ou de communication, formules chimiques,…). Les définitions de la coloration sont simples et de véritables problèmes de recherche peuvent être posés sous une forme bien structurée dont la formulation peut recouvrir de grandes difficultés pratiques. Ce modèle a été introduit la première fois dans le domaine des documents par Gaceb et Eglin [7] qui l'ont adapté à toutes les étapes d'analyse

des documents (de l'extraction de la structure physique et la localisation à la reconnaissance) pour consolider la coopération et assouplir les échanges d'information entre les différents modules. Grâce à sa simplicité et son potentiel en matière de classification, nous avons pu imaginer une méthode originale de construction de codebook représentatif de la distribution des graphèmes de l'écriture et de leurs fréquences d'apparition.

4.1 Type de colorations existantes

Il existe ainsi plusieurs types de colorations de graphe : la coloration de sommets à laquelle nous nous sommes intéressés, la coloration d'arêtes, la coloration par liste, etc. [7]. Une coloration de graphe *G(V,E)* est une fonction qui affecte une couleur à chaque sommet, et qui est telle que deux sommets reliés par une arête (adjacents ou voisins) n'ont pas la même couleur (contrainte de propreté). Les couleurs (ou entiers) attribuées aux sommets du graphe servent uniquement à regrouper les sommets en classes. Aussi Une seule valeur est utilisée comme paramètre de seuil de comparaison (mesure de similarité) dans l'algorithme de coloration de graphe, ce qui représente un avantage sur les autres algorithmes comme les *K-means* où le nombre de classes est demandé, et L'algorithme peut être appliqué sur différents niveaux et types de contenus: ici il a été utilisé pour construire des codebooks à partir de la classification des graphèmes, il peut être aussi appliqué sur les codebooks pour la classification des manuscrits.

4.2 Modélisation du problème de classification de graphèmes en termes de coloration

Le regroupement d'un ensemble $X=\{x_1,..,\ x_n\}$ de n graphèmes en plusieurs groupes homogènes se base sur le principe que chaque groupe doit réunir le plus de graphèmes similaires. Les regroupements portent sur un critère de similarité S. Ce critère spécifie que certaines paires de graphèmes $\{x_i\ ,\ x_j\}$ ne peuvent être fusionnées au sein d'un même groupe. Pour résoudre ce problème de partitionnement (ou de classification), on peut partir du point de vue inverse et formuler la question suivante : quel est le plus petit nombre de groupes homogènes que l'on peut former en respectant la contrainte *S ?* L'intérêt de formuler le problème de cette manière, est qu'il devient alors possible de le modéliser en termes de coloration de graphe. Le positionnement du problème est alors le suivant : nous représentons chaque graphème xi par un sommet $vi \in V$ d'un graphe simple *G* et nous ajoutons une arête $E(v_i\ ,v_j)$ entre chaque paire de graphèmes dissemblables (qui ne respectent pas la contrainte *S*). La coloration des sommets du graphe *G(V,E)* consiste alors à affecter à tous ses sommets une couleur de telle sorte que deux sommets adjacents (dissemblables) ne puissent pas porter la même couleur. Ces couleurs vont correspondre aux différents groupes homogènes qui constituent les différentes classes de graphèmes. Dans ce problème de regroupement, la question de la détermination du plus petit nombre de groupes homogènes, revient à rechercher le plus petit *k* pour lequel le graphe *G* correspondant admet une *k-coloration* : c'est donc précisément le nombre chromatique χ (G) du graphe *G* qu'il faut déterminer. En plus, cette modélisation présente l'avantage de gérer facilement plusieurs sortes d'ambiguïtés inhérentes à la forme des graphèmes par rapport aux mécanismes de regroupements classiques (Figure 4).

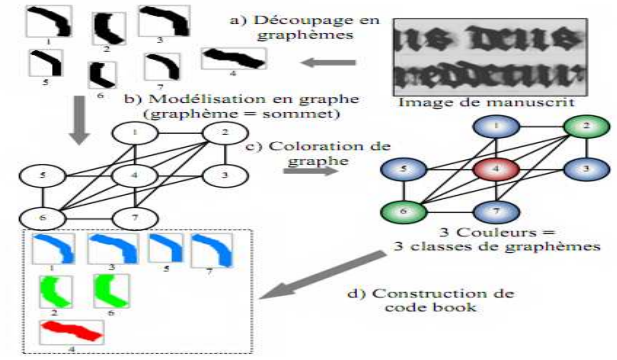

Figure 4.Etapes de construction de code book par coloration de graphes

4.3 Construction du codebook

A partir de l'ensemble des graphèmes extraits à partir de la méthode présenté dans [2], nous procédons à la construction du codebook de l'écriture (appelé aussi table de similarités).

4.3.1 Mesure de similarité

La dissimilarité entre v_i et v_j est donnée par la distance généralisée de Minkowski d'ordre α ($\alpha = 2$: distance euclidienne). $D_s = \left(\sum_{k=1}^{n} g_k(v_i^k, v_j^k)^{\alpha} \right)^{\frac{1}{\alpha}}$

$n=59$ est la longueur des vecteurs des caractéristiques. g_k est la fonction de dissemblance qui compare les caractéristiques deux à deux.

4.3.2 Construction du graphe

La construction d'un graphe *G* à colorer à partir d'un ensemble $X=\{x_1,\ldots,x_n\}$ de *n* graphèmes (où chaque sommet v_i correspond au vecteur descripteur du graphème x_i) est principalement basée sur le calcul de la matrice de distances MD_s . Cette matrice traduit les dissimilarités $Ds(x_i, x_j)$ entre les paires de graphèmes (x_i, x_j) données par la relation suivante : $MDS[v_i,v_j]=Ds(x_i, x_j)$ avec $i \in [1,n]$ et $j \in [1,n] \mid (i \neq j)$. Une fois *v* calculée, nous associons à *X* un graphe seuil supérieur $G_{\geq S} = (V=X, E_{\geq S})$ en utilisant la relation suivante :

$$E_{\geq S}[v_i, v_j] = \begin{cases} 1 & \text{si } Ds(x_i, x_j) = Ds(v_i, v_j) \geq S \\ 0 & \text{sinon} \end{cases}$$

Remarquons que le terme adjacence (ou voisinage) est différent du terme similarité, deux sommets sont adjacents s'ils ont une dissimilarité supérieure au seuil *S*. Le seuil *S* est également nommé seuil d'adjacence. Ce seuil peut être ajusté manuellement à l'aide des paléographes ou automatiquement en maximisant la qualité de classification ψ donné par [7], d'où :

$$S^{Optimal} = \arg\max(\psi(S_i))$$

5 Résultats et application

Nous proposons trois expérimentations pour illustrer les avantages de notre méthode. Dans la première, on valide notre approche de sélection de caractéristiques (par un apprentissage non-supervisé) et on montre comment on peut déduire un jeu de poids génériques sur la base d'apprentissage. Dans la seconde expérience on démontre la pertinence de ces poids sur une autre base (de test) qui se compose de manuscrits inconnus. En application de ces résultats, nous montrons dans une troisième expérience comment l'aspect hybride de notre approche de caractérisation (basée sur une analyse locale et globale en même temps) offre de meilleurs taux de reconnaissance de scripteurs par rapport aux méthodes existantes de CBIR globales y compris ceux qui sont déjà développées dans le cadre du projet GRAPHEM [12] (notamment la méthode basée sur les curvelettes).Nous utilisons pour les deux premières étapes de test une première base de *140* images de manuscrits médiévaux (disponible dans [9]) que nous avons décomposées en *600000* graphèmes. Nous divisons cette base en deux parties : B1 →100 images pour la première étape et B2→ *40* images pour la 2ème. Chaque page manuscrite sera représentée par un ensemble de graphèmes. Nous rappelons que le découpage de l'écriture en graphèmes est exécuté aux points suivants: de croisement, de jonction, de minimum local (figure 5).

Figure 5.Décomposition en traits:de haut en bas:Croisement, jonction, minimum local.

5.1 De la sélection à la déduction des poids génériques

Dans cette première étape, nous sélectionnons avec l'AG (voir la section 3) *pour chaque manuscrit* de la base *B1* un sous-ensemble de caractéristiques les plus pertinentes parmi l'ensemble de *59* caractéristiques de départ $\{a_1...a_{59}\}$. Chaque manuscrit conduit à une sélection de caractéristiques propre (on aura donc *100* sélections). Par la suite nous associons un poids à chaque caractéristique a_i calculé à partir des *100* sélections de la façon suivante : $w(a_i)=N(a_i)/T$; où $N(a_i)$ est le nombre de fois qu'une caractéristique a_i était sélectionnée sur les *100* manuscrits de la base *B1*. $T=\sum_{i=1}^{59} N(a_i)$ est un paramètre de normalisation des poids. La somme de *59* poids est égale à *1*.D'autre part, un seuil optimal de coloration est calculé aussi par *AG* pour chaque manuscrit de la base *B1*. Nous calculons, ainsi, le seuil générique de coloration comme la moyenne de tous les seuils optimaux obtenus sur les *100* manuscrits avec : $S_{générique} = \sum_{i=1}^{100} S_i^{optimal} / 100$

Les poids et le seuil génériques obtenus dans cette première étape sont utilisés pour calculer de nouveau la fonction de fitness sur chacun des manuscrits de *B1*. Nous comparons par la suite ces dernières avec les fonctions de fitness maximales de la sélection de caractéristiques. Cette comparaison va nous permettre de vérifier, pour chaque manuscrit, si les poids et le seuil génériques donnent des meilleures finesses (ou proches) que celles données par la sélection de caractéristiques (Figure 6).

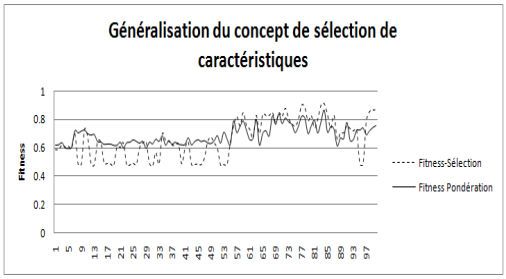

Figure 6.Courbes représentant les valeurs de la fonction de fitness qui sont calculés à partir des poids génériques et à partir de la sélection de caractéristiques sur les 100 manuscrits de la base d'apprentissage

A partir de la courbe précédente, on constate qu'avec un seul jeu de poids et seuil génériques on obtient sur tous les manuscrits une fitness nettement proche ou meilleure que celle obtenue par les différentes sélections de caractéristiques. Ce résultat montre la possibilité d'appliquer ces poids et seuil génériques sur de nouveaux manuscrits (inconnus) en préservant toujours une meilleure qualité de classification et par conséquence un codebook meilleur. C'est ce constat de faisabilité de la généralisation de ces poids et seuil qu'on va démontrer à travers l'expérience suivante.

5.2 Validation des poids génériques

Dans cette seconde étape, nous appliquons les poids et le seuil génériques issus de la première étape, cette fois ci, sur les manuscrits de la base *B2*. La courbe suivante montre que la fitness reste toujours performante devant toute sélection (même devant toutes les caractéristiques sans sélection). A partir de cette expérience on peut démonter la généralisation de ces poids et seuil (génériques) sur tout autre manuscrit inconnu sans avoir recours à une phase de sélection de caractéristiques (figure 7).

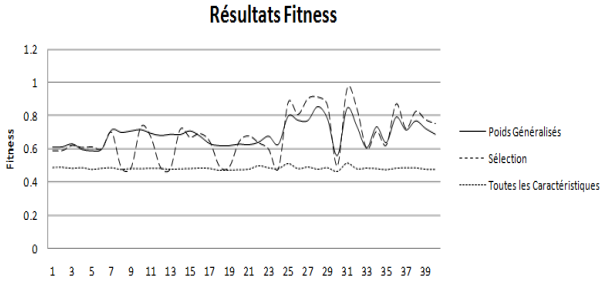

Figure 7.Résultats des 3 testes

On peut constater également à partir de la courbe suivante que les poids les plus élevés correspondent aux caractéristiques appartenant aux descripteurs suivant : *D_9 = 9 Densités, D_{10}= Huit orientations, D_{11}=Moments de Zernike}*. Cela signifie que ces dernières ont une force discriminatoire plus élevée que les autres (figure 8).

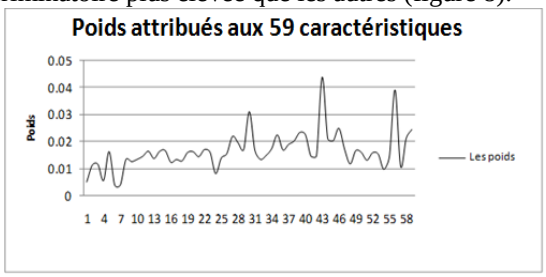

Figure 8. *Relation entre poids et caractéristiques*

Dans la section suivante, nous allons voir comment on peut appliquer ces poids et le seuil génériques d'une manière efficace dans la reconnaissance de style par la technique *CBIR*. Pour faire une comparaison par le contenu de l'image, chaque manuscrit est décrit par son propre codebook. Nous rappelons que ce codebook est construit par une coloration des graphèmes en utilisant les poids et le seuil génériques issus de l'étape1. La figure suivante montre un exemple d'un codebook où on voit clairement que les graphèmes qui dérivent du même mouvement de la plume se regroupent dans la même classe.

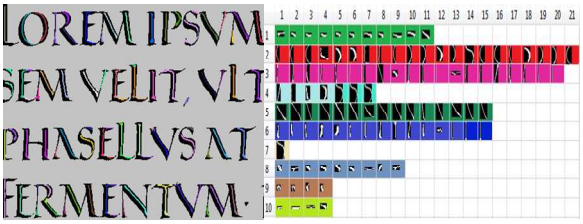

Figure 9. Résultat de codebook en utilisant la coloration de graphes et la pondération de caractéristiques

5.3 Application à la reconnaissance de styles basée sur la technique CBIR

Dans cette troisième étape, nous effectuons les différentes expériences sur deux autres bases : B3 → *426* images de manuscrits médiévaux [13] et B4 →*132* images de manuscrits de la base d'oxford [9]. Chaque manuscrit p_i des deux bases est représenté par son codebook *$cb(p_i)$* de k_i classes de graphèmes (k_i nombre chromatique de la coloration de graphes). Avec $cb(p_i) = \{p_i|_{r=1...ki}\}$ Chaque classe c_r^i est représentée par le centroïde des caractéristiques x_s^r de tous ces graphèmes avec $c_r^s = \{x_r^s|_{s=1..59}\}$. La dissimilarité entre deux manuscrits *(p_i, p_j)* est basée sur la distance globale de Schaefer [8] (notée par *dp*) entre leurs codebooks *{cb((*p_i *), cb(*p_j*)}*. Cette distance est une amélioration de la distance de Hausdorff. Cette mesure peut être formulée comme suit:

$$d_p[cb(p_i), cb(p_j)] = \max\{d[cb(p_i), cb(p_j)], d[cb(p_i), cb(p_j)]\}$$

$$d[cb(p_i), cb(p_j)] = \frac{1}{k_i}\sum_{r=1}^{k_i} \min_{t=1}^{kj} Ds[c_r^i \in cb(p_i), c_t^j \in cb(p_j)]$$

D_s est une mesure de dissimilarité locale entre chaque paire de graphèmes centroïdes. Pour tester cette représentation nous comparons une image requête d'un style donné à tous les manuscrits de la base à laquelle il appartient. La précision **P** et le rappel **R** des images correctement retournées sont obtenus directement à partir de la matrice de confusion des réponses pertinentes et non pertinentes retournées. Top n : signifie les n premiers manuscrits de styles les plus proches du manuscrit requête. C'est sur ces manuscrits proches que la précision et le rappel sont calculés, voir le tableau suivant :

Tableau 1.Résultat de précision et rappel sur les deux bases de manuscrits (P : précision, R : rappel).

	B4 (Notre méthode)		**B3 (Curvelets [12])**		**B3 (Notre méthode)**	
	P	**R**	**P**	**R**	**P**	**R**
Top 5	0.91	0.15	0.294	0.03	0.37	0.04
Top 10	0.90	0.25	0.292	0.07	0.35	0.06
Top 15	0.83	0.73	0.298	0.1	0.33	0.09
Top 20	0.77	0.85	0.292	0.15	0.22	0.18

D'une part les résultats sur la base d'Oxford montrent qu'en moyenne on a une bonne précision de récupération, pour le Top *15* on a une précision de *83%* avec un rappel de *73%*. Ce qui signifie que *73%* des manuscrits de l'image requête ont été donnés par l'application CBIR avec une F-Mesure de *77.6%*. Mais d'autre part on n'a pas eu de bons résultats pour les manuscrits du moyen âge, pourtant ils sont un peu meilleurs que ceux de la méthode de curvelettes. La courbe ROC et le graphe *PR* sont présentés dans la figure suivante avec *ASC* (Air Sous la Courbe) = *87.5%*.

Dans *B3* les mauvais résultats apparaissent dans les méthodes (Curvelets et la notre):cela est du à la vérité de terrain proposée par les paléographes qui est différente de la vérité visuelle (qui est la nôtre) et qui est sujette à des interprétations qu'il n'est pas possible de quantifier avec des descripteurs visuels comme ceux que nous proposons. Pour cela qu'il faut considérer le tableau résultat sans comparer les résultats de *B3* avec *B4*.

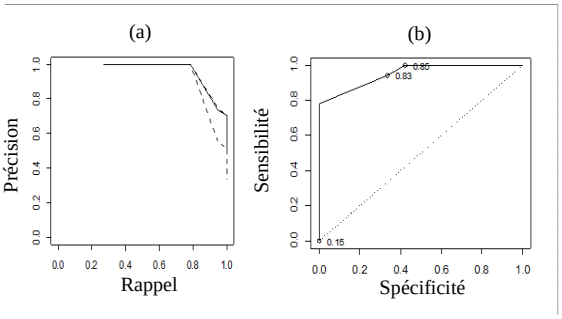

Figure 10.(a) PR et (b) ROC pour la base d'Oxford.

6 Conclusion

Nous avons élaboré à travers ce papier une nouvelle approche de caractérisation mixte (locale et globale) de l'écriture manuscrite. Nous avons montré comment à partir des graphèmes issus d'une analyse locale de la dynamique de l'écriture on peut obtenir une signature globale pour toute une page manuscrite caractérisant d'une manière discriminante les différents styles d'écriture. Cette signature a été calculée à partir d'un codebook représentant en différentes catégories, toute la diversité morphologique des graphèmes. A ce niveau, nous avons montré comment la coloration de graphe, engagée dans la construction des codebooks, offre une meilleure représentation des classes sans avoir recours à la moindre connaissance a priori du nombre de ces catégories. Afin de donner à notre approche une plus grande adaptabilité à la grande variabilité de la forme des graphèmes et plus de souplesse vis-à-vis de certaines ambiguïtés, nous avons joint à la construction des codebooks, une phase d'extraction automatique des poids et des seuils génériques. Cette démarche nous a permis d'atteindre de meilleures classifications des graphèmes et par conséquent d'offrir une construction des codebooks plus efficace. Nous avons démontré comment tous ces éléments ont participé à l'amélioration des taux de reconnaissance des styles de l'écriture manuscrite. Cette contribution représente un nouveau point de vue dédié aux applications CBIR, nous l'avons développé dans le cadre du projet GRAPHEM mais la méthodologie est applicable beaucoup plus largement comme l'ont montré les applications développées.

L'approche de décomposition des écritures manuscrites en graphèmes peut se généraliser (nos travaux sur la base *IAM* et la base des manuscrits clandestins d'Europe du $18^{ème}$ siècle l'attestent) à un grand panel de documents textuels. Plus généralement, la décomposition en traits (strokes) est très développée dans les images de documents et schémas graphiques, les partitions musicales et plus généralement les images de traits. Des travaux portant sur la décomposition en fragments linéaires (à partir d'une extraction de squelettes des formes) sur des images naturelles(ou des images de peintures comme dans [14]) permettent d'envisager des pistes intéressantes d'application de nos travaux. Nous pouvons envisager une extension de l'application *CBIR* à des approches de Word spotting ou shape-spotting portant sur un matching de formes guidés par l'adjacence des graphèmes le long de la formation du mot ou de la forme. Enfin la proposition d'exploiter le système de génération de poids à n'importe quelle autre application de reconnaissance de formes peut être formulée en continuité de nos travaux dès lors que celle-ci nécessite une caractérisation en petites composantes (de type *strokes*).

Bibliographie

[1] M. Dash and H. Liu, Consistency-based search in feature selection, *Artif. Intell*, vol. 151, pp. 155–176, 2003.

[2] H. Daher, D. Gaceb,V. Eglin, S. Bres, N. Vincent, Ancient Handwritings Decomposition Into Graphemes and Codebook Generation Based on Graph Coloring, *ICFHR*,pp. 119-124,2010.

[3] R. Kohavi and G. H. John,Wrappers for feature subset selection, *Artif. Intell*, vol. 97, pp. 273–324, 1997.

[4] H. B. Kekre, Tanuja K. Sarode, An Efficient Fast Algorithm to Generate Codebook for Vector Quantization, *ICETET*, 2008.

[5]M.S.Khorsheed, Recognising handwritten Arabic manuscripts using a single hidden Markov model, *Pattern Recogn.Lett*, Vol.24, pp. 2235-2242, 2003.

[6] Tara Gilliam, Richard C. Wilson, and John A. Clark, Scribe Identification in Medieval English Manuscripts. *ICPR*,pp.1880-1883, 2010.

[7] D. Gaceb and V. Eglin, Improvement of postal mail sorting system, *IJDAR*, Vol. 2, pp. 67-80, 2008.

[8] G. Schaefer,Compressed domain image retrieval by comparing vector quantization codebooks, *Proc. SPIE*, vol. 4671, p.959 , 2002.

[9] http://image.ox.ac.uk/list?collection=bodleian.

[10] L. Schomaker, K. Franke, and M. Bulacu. 2007, Using codebooks of fragmented connected-component contours in forensic and historic writer identification. *Pattern Recogn. Lett*, Vol. 28, pp. 719-727, 2007.

[11] I. Siddiqi, N. Vincent. Text Independent Writer Recognition Using Redundant Writing Patterns with Contour-Based Orientation and Curvature Features, *Pattern Recognition Journal*, Vol.43, pp. 3853-3865, 2010.

[12] G. Joutel, V. Eglin, S. Bres, and H. Emptoz. Curvelets Based Queries for CBIR Application in Handwriting Collections, *ICDAR*, Vol. 2, pp. 649-653, 2007.

[13] http://www.bl.uk

[14]Animating Chinese Paintings through Stroke-Based Decomposition, Songhua Xu, Yingqing Xu,, Sing Bing Kang,David H. Salesin DAVID H, ACM Transactions on Graphics, Vol. 5, pp. 1–31, 2005.