Recherche d'images par l'analyse factorielle des correspondances

Nguyen-Khang Pham[*,**] — **Annie Morin**[*] — **Patrick Gros**[*]

[*] *IRISA*
Campus de Beaulieu, F-35042 Rennes Cedex
{pnguyenk, amorin, pgros}@irisa.fr

[**] *Université de Cantho*
Campus III, 1 Ly Tu Trong, Ville de Cantho, Vietnam
pnkhang@cit.ctu.edu.vn

RÉSUMÉ. *On utilise l'analyse factorielle des correspondances (AFC) pour la recherche d'images par le contenu en s'inspirant directement de son utilisation en analyse des données textuelles (ADT). En ADT, l'AFC est appliquée à un tableau de contingence croisant mots et documents. La première étape consiste donc à établir des « mots visuels » dans les images (analogue des mots dans les textes). Ces mots sont construits à partir des descripteurs locaux (SIFT) des images. La méthode a été testée sur la base Caltech4 (Sivic* et al., *2005) et elle fournit de meilleurs résultats (qualité des résultats de recherche et temps d'exécution) que des méthodes plus classiques comme TF*IDF/Rocchio (Rocchio, 1971) ou pLSA (Hofmann, 1999b). Enfin, pour passer à l'échelle, nous proposons un nouveau prototype de recherche utilisant des fichiers inversés basés sur la contribution des images à l'inertie des axes à l'issue d'une AFC. Chaque fichier inversé est associé à une partie d'un axe et contient des images ayant une contribution forte à l'inertie de cet axe. Les tests réalisés montrent que ce nouveau prototype réduit le temps de recherche sans perte de qualité de résultat.*

ABSTRACT. *A new approach for content based image retrieval uses Factorial Correspondence Analysis (FCA), inspired from the use of FCA in textual analysis (TA). We get better results with FCA than with other classical methods such as: TF*IDF or pLSA. We already propose a new prototype for image retrieval using inverted files based on the contribution of images to the intertia of axes.*

MOTS-CLÉS : *Analyse factorielle des correspondances, SIFT, fichier inversé*

KEYWORDS: *Correspondence Analysis, SIFT, inverted file*

1. Introduction

L'utilisation des descripteurs locaux permet d'obtenir de bons résultats pour la reconnaissance d'images, la classification d'images et la recherche d'images par le contenu. Ces descripteurs sont robustes aux changements de contenu. Cette méthode a été proposée en 1997 par C. Schmid dans (Schmid *et al.*, 1997). Récemment, les méthodes développées originellement pour l'analyse des données textuelles (ADT) comme pLSA (probabilistic Latent Semantic Analysis) (Hofmann, 1999a), LDA (Latent Dirichlet Allocation) (Blei *et al.*, 2003) sont appliquées en analyse d'images, par exemple pour la classification des images (Willamowski *et al.*, 2004), la découverte des thèmes dans l'image (Sivic *et al.*, 2005), la classifications des scènes (Bosch *et al.*, 2006), et la recherche d'images (Lienhart *et al.*, 2007).

Dans ce travail, nous utilisons l'analyse factorielle des correspondances (AFC) pour la recherche d'images. Étant donné une image requête, le système doit retourner les images (dans la collection des images) les plus similaires à la requête. L'AFC permet ici de réduire l'espace pour représenter les images et calculer la similarité entre les images dans cet espace réduit. Les deux contributions principales de cet article sont l'utilisation de l'AFC pour la recherche d'images, et la proposition d'un prototype de recherche d'images utilisant des fichiers inversés basés sur la contribution des images à l'inertie des facteurs de l'AFC.

L'article est organisé de la façon suivante : nous décrivons brièvement les méthodes pLSA et l'AFC dans la section 2. La section 3 présente la recherche d'images par l'AFC. La section 4 est consacrée aux résultats expérimentaux. Dans la conclusion, nous présentons les perspectives de ce travail.

2. Méthodes

2.1. *Représentation des images*

Qu'il s'agisse des méthodes comme le pLSA ou l'AFC, il faut d'abord représenter le corpus sous forme d'une matrice d'occurrences F (un tableau de contingence) de dimension $M \times N$ où M désigne le nombre de documents et N indique le nombre de différents mots apparaissant dans le corpus. Chaque case de la matrice F_{ij} décrit le nombre de fois où le mot j (indice de colonne) est observé dans le document i (indice de ligne). Une telle représentation ignore l'ordre des mots dans un document et est appelée modèle sac-de-mots (bag-of-words).

Il n'y pas de mots au sens littéral du terme dans les images. Il faut donc les construire.

2.1.1. *Construction des mots visuels*

Les mots dans les images, appelés mots visuels, doivent être calculés pour constituer un vocabulaire de N mots. Chaque image sera donc représentée enfin par un

histogramme de mots. La construction des mots visuels se fait en deux étapes : (i) calcul des descripteurs locaux pour un ensemble d'images, (ii) classification (clustering) des descripteurs obtenus. Chaque cluster correspondra à un mot visuel. Il y aura donc autant de mots que de clusters obtenus à l'issue de l'étape (ii).

Le calcul des descripteurs locaux dans une image se fait aussi en deux étapes : il faut d'abord détecter des points d'intérêt dans l'image. Ces points d'intérêt sont, soit des maximums du Laplacien de Gaussien (Lindeberg, 1998), soit des extremums locaux 3D de la différence de Gaussien (Lowe, 1999), soit des points extraits par un détecteur Hessian-affine (Mikolajczyk *et al.,* 2004). Ensuite, le descripteur de ce point d'intérêt est calculé sur le gradient des niveaux de gris dans la région autour du point. On a sélectionné des descripteurs invariants à la rotation et au changement d'échelle, les descripteurs SIFT (Lowe, 2004). Chaque descripteur SIFT est un vecteur à 128 dimensions. La seconde étape consiste à former des mots visuels à partir des descripteurs locaux calculés à l'étape précédente. La plupart des travaux effectue un k-means sur les descripteurs locaux et prend les moyennes de chaque cluster comme mots visuels (Willamowski *et al.*, 2004, Sivic *et al.,* 2005, Bosch *et al.,* 2006). Après avoir construit le vocabulaire visuel, chaque descripteur est affecté au cluster le plus proche. Pour cela, on calcule dans $\mathbb{R}^{128}$ les distances de chaque descripteur aux représentants des clusters définis précédemment. Une image est ensuite caractérisée par la fréquence de ses descripteurs dans chaque cluster. On obtient ainsi un tableau de contingence croisant les images et les clusters.

Dans nos expérimentations, nous avons utilisée la méthode décrite dans (Mikolajczyk *et al.,* 2004) pour détecter des points d'intérêt. Le vocabulaire est construit en appliquant un k-means sur environ 300000 descripteurs tirés aléatoirement (un tiers pour chaque catégorie : faces, motorbikes, airplanes, background et cars). Le vocabulaire obtenu est de 2224 mots pour 4090 images. Ce choix de 2224 mots a été effectué par Sivic (Sivic *et al.,* 2005).

2.2. *PLSA*

Introduite par Thomas Hofmann (Hofmann, 1999a), le pLSA est une technique statistique qui s'inspire du LSA (Latent Semantic Analysis) (Deerwester *et al.,* 1990) pour l'analyse des tableaux de contingence. LSA est une méhode purement géométrique qui ressemble beaucoup aux méthodes factorielles ; dans pLSA, on introduit un modèle probabiliste : la distribution des mots dans une image est considérée comme multinomiale. La méthode se base sur une décomposition des mélanges dérivée d'un modèle de variables latentes.

pLSA introduit une variable latente $z \in Z = \{z_1, z_2, ..., z_K\}$ et modélise la probabilité jointe $P(d, w)$ par :

$$\begin{aligned} P(d,w) &= P(d)P(w|d), où \\ P(w|d) &= \sum_{z \in Z} P(w|z)P(z|d) \end{aligned} \quad [1]$$

Le logarithme de la vraisemblance du corpus est défini par :

$$\mathcal{L} = \sum_{d \in D} \sum_{w \in W} F(d,w) log(P(d,w)) \quad [2]$$

Où :

$D = \{d_1, d_2, ..., d_M\}$: l'ensemble des documents.

$W = \{w_1, w_2, ..., w_N\}$: l'ensemble des mots.

F : le tableau de contingence.

$\mathcal{L}$ est maximisé par un algorithme EM avec l'étape E :

$$P(z|d,w) = \frac{P(z)P(d|z)P(w|z)}{\sum_{z'} P(z')P(d|z')P(w|z')} \quad [3]$$

suivi par l'étape M :

$$P(d|z) = \frac{\sum_w F(d,w)P(z|d,w)}{\sum_{d',w} F(d',w)P(z|d',w)} \quad [4]$$

$$P(w|z) = \frac{\sum_d F(d,w)P(z|d,w)}{\sum_{d,w'} F(d,w')P(z|d,w')} \quad [5]$$

$$P(z) = \frac{1}{R} \sum_{d,w} F(d,w)P(z|d,w), R \equiv \sum_{d,w} F(d,w). \quad [6]$$

2.3. *Analyse factorielle des correspondances*

L'AFC est une méthode exploratoire classique pour l'analyse des tableaux de contingence. Elle a été proposée par J. P. Benzécri (Benzécri, 1973) dans le contexte de la linguistique, c'est-à-dire pour l'analyse de données textuelles. La première étude a été réalisée sur les tragédies de Racine. L'AFC sur un tableau croisant des mots et des documents permet de répondre aux questions suivantes : y a-t-il des proximités entre certains mots ? Y a-t-il des proximités entre certains documents ? Y a-t-il

des liens entre certains mots et certains documents ? L'AFC comme la plupart des méthodes factorielles utilise une décomposition en valeurs singulières d'une matrice particulière et permet la visualisation des mots, et des documents dans un espace de dimension réduit. Cet espace de dimension réduit a la particularité d'avoir un nuage de points projetés (mots et/ou documents) d'inertie maximale. Par ailleurs, l'AFC fournit des indicateurs pertinents pour l'interprétation des axes comme la contribution d'un mot ou d'un document à l'inertie de l'axe ou la qualité d'un mot et/ou d'un document sur un axe (Morin, 2004).
Soit un tableau de contingence $F = \{f_{ij}\}_{M,N}$ de taille $M \times N$ ($N < M$). On normalise F en $X = \{x_{ij}\}_{M,N}$ par :

$$s = \sum_{i=1}^{M}\sum_{j=1}^{N} f_{ij} \quad [7]$$

$$x_{ij} = \frac{f_{ij}}{s} \ , \ \forall i = 1..M, j = 1..N \quad [8]$$

et note :

$$p_i = \sum_{j=1}^{N} x_{ij} \ , \ \forall i = 1..M \qquad q_j = \sum_{i=1}^{M} x_{ij} \ , \ \forall j = 1..N$$

$$P = \begin{pmatrix} p_1 & & & 0 \\ & p_2 & & \\ & & ... & \\ 0 & & & p_M \end{pmatrix} \qquad Q = \begin{pmatrix} q_1 & & & 0 \\ & q_2 & & \\ & & ... & \\ 0 & & & q_N \end{pmatrix}$$

Pour déterminer le meilleur sous-espace de projection des données, on calcule les valeurs propres et les vecteurs propres de la matrice V de taille $N \times N$:

$$V = X^{'} P^{-1} X Q^{-1} \quad [9]$$

où $X^{'}$ est la transposée de X.
On obtient alors les valeurs propres λ et les vecteurs propres μ :

$$\lambda = \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_N \end{pmatrix}, \ \mu = \begin{pmatrix} \mu_{11} & \mu_{12} & ... & \mu_{1N} \\ \mu_{21} & \mu_{22} & ... & \mu_{2N} \\ ... & ... & ... & ... \\ \mu_{N1} & \mu_{N2} & ... & \mu_{NN} \end{pmatrix}$$

On ne garde que les $K(K < N)$ premières valeurs propres les plus grandes et les vecteurs propres associés. Ces K vecteurs propres constituent une base orthonormée de l'espace réduit (appelé aussi espace des facteurs). Le nombre de dimensions du problème passe de N à K. Les documents sont projeté dans le nouvel espace réduit :

$$Z = P^{-1} X A \quad [10]$$

Dans la formule [10] $P^{-1}X$ représente les profils lignes et $A = Q^{-1}\mu$ est la matrice de transition associée à l'AFC. La projection des mots dans le sous-espace de dimension K est fournie par la formule suivante :

$$W = Q^{-1}X^{'}Z\lambda^{-\frac{1}{2}} \quad [11]$$

Un nouveau document (i.e. la requête) $r = [r_1\ r_2\ \cdots\ r_N]$ sera projeté dans l'espace des facteurs par la formule de transition [10] :

$$\begin{aligned} \hat{r}_i &= \frac{r_i}{\sum_{j=1}^{N} r_j},\ \forall i = 1..N \\ Z_r &= \hat{r}A \end{aligned} \quad [12]$$

2.4. *Mesures de similarité des images*

Après avoir calculé les coordonnées des images dans le nouvel espace (i.e. représentation probabiliste dans le cas de pLSA, sous-espace pour l'AFC), nous devons définir la similarité des images. Plusieurs métriques pour cette mesure de similarité sont disponibles, parmi les quelles les 4 mesures suivantes :

Distance de Manhattan (norme 1) :

$$L_1(a,b) = \sum |a_i - b_i|$$

Distance euclidienne (norme 2) :

$$L_2(a,b) = \sqrt{\sum (a_i - b_i)^2}$$

Distance cosinus :

$$L_{cos}(a,b) = cos^{-1}\left(\frac{a.b}{\|a\|.\|b\|}\right)$$

Divergence de Jensen-Shannon :

$$\begin{aligned} JS(a,b) &= \frac{1}{2}\left(KL\left(a, \frac{a+b}{2}\right) + KL\left(b, \frac{a+b}{2}\right)\right)\ o\grave{u} \\ KL(a,b) &= \sum a_i log\left(\frac{a_i}{b_i}\right) \end{aligned}$$

3. Recherche d'images par l'AFC

3.1. *AFC pour réduction de dimensions*

Un des avantages de l'AFC est de réduire la dimension du problème. Pour tenir compte du nombre d'images, il est préférable d'utiliser une structure d'indexation sous forme d'arbre comme un arbre *k-d* (Bentley, 1975). Cependant, une telle structure deviendra inefficace quand le nombre de dimensions est supérieur à 16 à cause de la malédiction de la dimension (Bellman, 1961). Enfin, l'indexation par l'arbre utilise la distance euclidienne la plupart du temps car on rencontre des difficultés lorsqu'on travaille avec d'autres distances.

3.2. *Passage à l'échelle*

Il y a deux indicateurs importants pour l'interprétation et l'évaluation en AFC. Ce sont la contribution des images à l'inertie d'un axe d'une part et la qualité de représentation des images sur un axe (facteur) d'autre part. Dans les expérimentations, nous avons constaté que la distance cosinus donnait de meilleurs résultats que la distance euclidienne. La distance cosinus est directement liée à la qualité de représentation des images sur les axes. Pour cette raison, nous avons proposé un nouveau prototype de recherche d'images utilisant des fichiers inversés basés sur la qualité de représentation. Cela permettra de réduire le nombre d'images à considérer lors du calcul de leur similarité avec la requête.

3.2.1. *Fichiers inversés basés sur la contribution*

Définition 1 (contribution) : *la contribution d'un point i (correspond à l'image i) à l'inertie de l'axe j est définie par :*

$$Cr(i,j) \quad = \quad p_i \frac{Z_{ij}^2}{\lambda_j} \qquad [13]$$

où Z_{ij} est la coordonnée du point i sur l'axe j.

Définition 2 (fichier inversé) : *étant donné un seuil $\epsilon > 0$, un fichier inversé F_j^+ (F_j^-) associé à la partie positive (négative) de l'axe j est une liste d'images ayant une contribution supérieure à ϵ et se trouvant dans la partie positive (négative) de l'axe j.*

$$F_j^+ \quad = \quad \{i \mid Cr(i,j) > \epsilon \; et \; Z_{ij} > 0\} \qquad [14]$$

$$F_j^- \quad = \quad \{i \mid Cr(i,j) > \epsilon \; et \; Z_{ij} < 0\} \qquad [15]$$

Pour les documents suplémentaires (i.e requête), puisqu'ils ne contribuent pas à l'inertie des axes, il n'y a pas donc leur contribution proprement dit. Pourtant nous pouvons

utiliser une analogue de la contribution, *pseudo-contribution*, à la place de la contribution.

Définition 3 (pseudo-contribution) : *la pseudo-contribution d'un point suplémentaire r (correspond à l'image $r = [r_1\ r_2\ \cdots\ r_N]$) à l'inertie de l'axe j est définie par :*

$$Cr(r,j) \quad = \quad p_r \frac{Z_{rj}^2}{\lambda_j},\ p_r = \frac{1}{s}\sum_{i=1}^{N} r_i \qquad [16]$$

où Z_{rj} est la coordonnée du point r sur l'axe j calculée de la formule [12], et s est fourni par [7].

Définition 4 (qualité de représentation) : *la qualité de représentation d'un point i (correspond à l'image i) sur l'axe j est le cosinus carré de l'angle entre l'axe j et le vecteur joignant le centre du nuage au point i :*

$$Cos_j^2(i) \quad = \quad \frac{Z_{ij}^2}{\sum_{k=1}^{K} Z_{ik}^2} \qquad [17]$$

Après avoir réduit la dimension des données à K, on construit, pour chaque axe, 2 fichiers inversés (un pour la partie positive et un autre pour la partie négative)[1]. Chaque fichier contient des images ayant une forte contribution à l'inertie sur la partie (positive, négative) de l'axe associé.

Le seuil ϵ est un paramètre qui contrôle la qualité de résultat et le temps de recherche. Dans les expérimentations, nous avons choisi un seuil égal à la contribution moyenne, à la moitié, et à un quart de la contribution moyenne. Puisque la somme des contributions est égale à 1 pour tous les axes, il est facile donc de calculer la contribution moyenne. Plus le seuil est grand, plus le nombre d'images dans un fichier inversé est petit, moins il faut de temps de recherche mais la qualité de résultat diminue.

3.2.2. *Prototype de recherche*

Algorithme pour rechercher une image requête r utilisant des fichiers inversés est donné dans le tableau 1. Cet algorithme se base sur le principe suivant : « Deux images similaires partagent certaines propriétés commune ». Avec ce principe, on rejettera les images qui n'ont aucune propriété commune à l'image requête. Dans ce travail, nous avons proposé d'utiliser la contribution à l'inertie des axes comme propriété pertinente. Le nombre d'images retrouvées après le filtrage sera beaucoup plus petit que le nombre d'images dans la base. Donc, le temps de recherche sera considérablement réduit. Le nombre *NF* est choisi empiriquement. Cependant, il est préférable que *NF* soit impair et inférieur à $K/2$ (pour le vote majoritaire dans l'étape de filtrage). Nous

1. Les images qui se trouvent à l'origine ne sont pas intéressantes. Elles sont souvent des fonds (backgrounds).

Entrée :	r – vecteur représentant l'image requête
1	Projeter r dans l'espace des facteurs par la formule [12] et trier les facteurs par contribution de la requête à ces facteurs
2	Prendre les (ex. *NF* = 3) fichiers inversés associés à *NF* (ex. *NF* = 3) premiers facteurs
3	Fusionner les fichier inversés ⇒ liste des images candidates Faire l'union des fichiers inversés Rejeter les images qui apparaissent au plus *NF*/2 fois
4	Calculer la distance entre la requête et les images conservées dans l'étape 3
Sortie :	liste des images les plus similaires à la requête

Tableau 1. *Algorithme pour recherche d'images se basant sur des fichiers inversés*

proposons 2 heuristiques qui peuvent être utilisées pour déterminer *NF* en fonction de l'image requête :

1) Heuristique 1 (pseudo-contribution) : on prend tous les axes sur lesquels la *pseudo-contribution* de l'image est supérieure à la contribution moyenne.

2) Heuristique 2 (qualité de représentation) : cette heuristique se base sur une observation suivante : si un point est bien représenté sur quelques axes, le cosinus carré sur ces axes sera grand et le cosinus carré des autres axes sera petit car la somme des cosinus carrés est égale à 1. Donc, on peut prendre *NF* axes tels que la somme des cosinus carrés sur ces axes est supérieure à α (ex : $\alpha = 0.5$).

4. Résultats expérimentaux

Nous avons implémenté l'AFC en GNU Octave (Eaton, 1995). Sur un PC Pentium 4, 512MO avec système d'exploitation Linux, le temps d'exécution pour l'AFC sur un tableau de contingence de 4090 x 2224 est d'environ 3 minutes.

4.1. *Ensemble de tests*

Les tests ont été réalisés sur la base Caltech4 (Sivic *et al.*, 2005) tirée de la base Caltech101 (Fergus *et al.*, 2003). Cette base contient 4090 images réparties en 5 catégories pour un vocabulaire de 2224 mots. Le tableau 2 décrit cette la base et quelques images tirées de cette base se figurent dans le figure 1.

Catégories	Nombre d'images
faces	435
motorbikes	800
airplanes	800
backgrounds	900
cars (rear)	1155
Total	4090

Tableau 2. *Description de la base Caltech4*

Figure 1. *Images tirées de la base Caltech4*

4.2. *AFC vs d'autres méthodes*

Nous avons fait une AFC sur les données de la base Caltech et avons gardé les 7 premiers axes (pour la comparaison au pLSA, avec 7 modalités). Lorsqu'on augmente le nombre d'axes conservés, on constate que les premières images retournées (i.e. les 10 premières images) sont bonnes mais que la qualité se dégrade lorsqu'on retient un plus grand nombre d'images (voir le tableau 3). La distance euclidienne et la distance cosinus sont utilisées pour calculer les similarités dans les espaces de dimension réduite. Nous avons utilisé la courbe de précision - rappel pour comparer la performance des différentes méthodes.

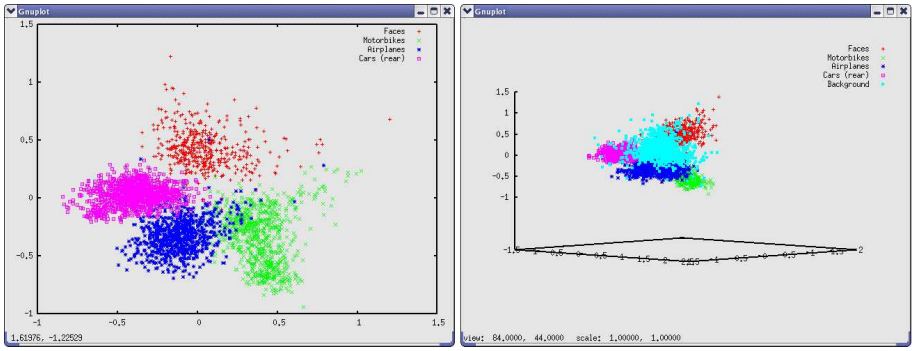

Figure 2. *Projection de la base Caltech4 sur les axes : à gauche, projection des images (sans background) sur les axes 1 et 2 ; à droite, projection des images (avec catégorie background) sur les axes 1, 2 et 3.*

4.2.1. *TF*IDF*

Avec cette méthode, chaque élément F_{ij} dans le tableau de contingence est normalisé à $tf(i,j)$ et pondéré par $idf(j)$ où $tf(i,j)$ est le nombre de mots j qui apparaît dans l'images i divisé par le nombre de mots dans l'image i et $idf(j) = ln(N/N_j)$ où N_i est le nombre d'images qui contiennent le mot j et N le nombre d'images dans la base. La distance euclidienne et la distance cosinus sont calculées sur les données pondérées.

4.2.2. *PLSA*

Nous avons appliqué un modèle pLSA avec 7 modalités (ce modèle donne les meilleurs résultats sur cette base (Sivic *et al.*, 2005). Chaque image dans la base est représentée par sa distribution $P(z|d)$. La dimension du problème est réduite à 7. Les nouvelles coordonnées servent à calculer la similarité entre la requête et les images dans la base (Hofmann, 1999b).

4.2.3. *Discussion*

Le nombre de thèmes (modalités de la multinomiale) dans pLSA et le nombre d'axes conservés en AFC sont des paramètres à régler. Le nombre d'axes conservés en AFC est difficile à choisir car les valeurs propres en AFC décroissent très lentement. Nous avons pris 7 axes en AFC pour la comparaison avec pLSA. La figure 3 montre les résultats quand on fait des tests sur 4 catégories (on ne tient pas compte de la catégorie « background ») et sur 5 catégories (avec la catégorie « background »). Le meilleur résultat est obtenu avec l'AFC quelle que soit la distance, euclidienne ou cosinus. Nous avons également testé TF*IDF avec la distance de Manhattan. Dans ce cas, on améliore le résultat mais ce n'est pas significativement supérieur aux résultats obtenus avec d'autres distances, ni avec pLSA et l'AFC.

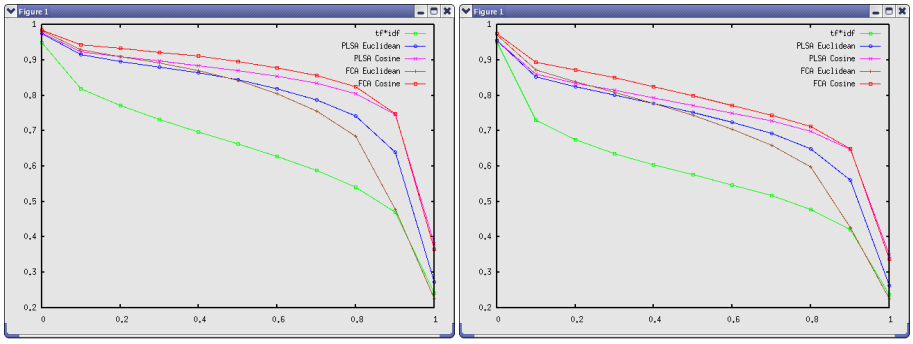

Figure 3. *Courbe de précision – rappel : à gauche, résultat pour 4 catégories (sans background) et à droite, résultat pour 5 catégories (avec background).*

Méthodes	#imgs	5 imgs	10 imgs	50 imgs	100 imgs	temps	
E : K = 7	–	94.87%	93.48%	90.54%	89.08%	4.24	(4.6)
E : K = 15	–	95.92%	94.61%	**91.35%**	**89.64%**	4.99	(3.9)
E : K = 30	–	**96.30%**	**95.03%**	91.22%	89.23%	6.56	(3.0)
N : K = 7	888	94.55%	93.23%	90.24%	88.76%	**1.22**	**(15.9)**
N : K = 15	791	95.85%	94.47%	91.01%	88.93%	1.32	(14.7)
N : K = 30	741	96.18%	94.88%	90.38%	87.05%	1.56	(12.4)
TF*IDF	–	88.24%	84.81%	77.52%	73.72%	19.36	(1.0)

Tableau 3. *Taux de précision sur les premières images retournées, NF = 3, $\epsilon = \frac{1}{4}moyenne$; E : est la méthode exhaustive et N : est la nouvelle méthode ; #imgs : le nombre d'images restées après filtrage ; temps : le temps de réponse pour une image (millisecondes/image) ; (x) : taux d'accroissement par rapport à TF*IDF*

4.3. *AFC avec fichiers inversé*

Pour comparer la performance de la nouvelle méthode de recherche avec la méthode exhaustive, nous avons calculé la précision sur les 5, 10, 50 et 100 premières images retournées. Le paramètre K est pris égal à 7, 15 et à 30 ; le seuil ϵ pour des fichiers inversés est mis à un quart de la contribution moyenne ; *NF* est calé à 3. Les résultats sont figurés dans le tableau 3. La nouvelle méthode limite le nombre d'images pour lesquelles on calcule la similarité avec la requête. Cela diminue le temps de réponse. En général, la nouvelle méthode est 4 fois plus rapide que la méthode exhaustive (on gagne 75% du temps) avec une perte inférieure à 1% du taux de précision. Enfin, dans tous les cas, la nouvelle méthode est meilleure que TF*IDF en qualité de résultat (plus de 10%) ainsi que le temps de recherche (14 fois plus rapide). Les expérimentations suivantes nous montrent l'influence du seuil et celle du para-

$Seuil\ \epsilon$	#imgs	5 imgs	10 imgs	50 imgs	100 imgs	temps	
$moyenne$	229	94.41%	92.95%	87.22%	80.62%	**0.59**	**(32.8)**
$\frac{1}{2}moyenne$	494	95.64%	94.28%	90.45%	87.45%	0.90	(25.5)
$\frac{1}{4}moyenne$	791	**95.85%**	**94.47%**	**91.01%**	**88.93%**	1.32	(14.7)

Tableau 4. *Influence du seuil, $NF = 3$, $K = 15$; #imgs : le nombre d'images restées après filtrage ; temps : le temps de réponse pour une image (millisecondes/image) ; (x) : taux d'accroissement par rapport à TF*IDF*

NF	#imgs	5 imgs	10 imgs	50 imgs	100 imgs	temps	
1	1134	95.82%	94.50%	90.99%	88.76%	1.64	(11.8)
3	791	**95.85%**	94.47%	91.01%	88.93%	1.32	(14.7)
5	607	95.71%	94.38%	90.94%	88.91%	1.10	(17.6)
7	474	95.55%	94.24%	90.73%	88.54%	**0.98**	**(19.8)**
auto1 (3.7)	786	95.83%	94.47%	91.05%	88.88%	1.32	(14.7)
auto2 (2.8)	922	95.82%	**94.52%**	**91.12%**	**89.23%**	1.42	(13.6)

Tableau 5. *Influence du paramètre NF, $K = 15$, $\epsilon = \frac{1}{4}moyenne$; auto1 : NF est calculé d'après la heuristique 1, auto2 : NF est calculé d'après la heuristique 2 ; #imgs : le nombre d'images restées après filtrage ; temps : le temps de réponse pour une image (millisecondes/image) ; (x) : taux d'accroissement par rapport à TF*IDF*

mètre *NF*. Nous faisons varier ce paramètre avec des valeurs : $moyenne$, $\frac{1}{2}moyenne$, $\frac{1}{4}moyenne$ en fixant $K = 15$, *NF* = 3. Le tableau 4 montre les résultats.

Pour étudier l'influence du paramètre *NF*, nous avons pris $K = 15$, $seuil = \frac{1}{4}moyenne$ et testé avec *NF* = 1, *NF* = 3, *NF* = 5, *NF* = 7 et *NF* calculé automatiquement (*cf.* Section 3.2.2) pour chaque image requête. Le tableau 5 figure les résultats.

Quand *NF* est petit (ex. *NF* = 1), un seul axe ne suffit pas en terme d'information récupéré et donc le résultat n'est pas bon. Quand on prend plus d'axes (ex. *NF* = 3) la qualité augmente et le filtre devient également trop contraignant (dans le cas *NF* = 7, une image sera gardée si elle apparaît au moins 4 fois dans les fichiers inversés). Par conséquent, cela dégrade la qualité du résultat. C'est pour cela qu'il faut choisir le paramètre *NF* en se basant sur la pseudo-contribution ou sur la qualité de représentation (*cf.* Section 3.2.2). Sur la base Caltech4, *NF* moyen est égale à 3.7 d'après la heuristique 1 et à 2.7 d'après heuristique 2 et notre proposition est justifiée a posteriori.

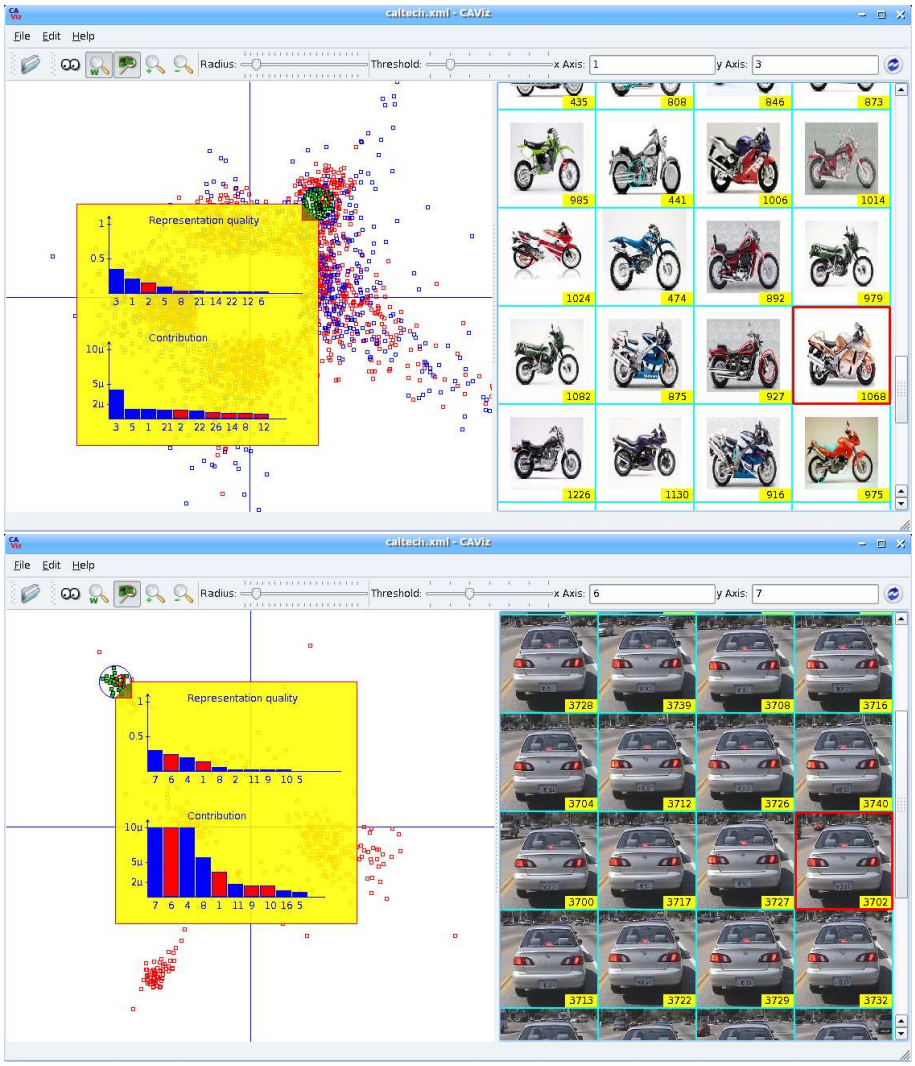

Figure 4. *Qualité de représentation et contribution des images aux axes, μ : contribution moyenne.*

5. Conclusion et perspectives

Nous avons présenté dans cet article une nouvelle approche pour la recherche d'images par le contenu en utilisant l'AFC. Cette méthode est testée sur la base Caltech4 et comparée aux méthodes classiques comme : TF*IDF et pLSA. Les expérimentations ont montré que dans tous les cas, l'AFC donne le meilleur résultat. Nous avons aussi proposé une nouvelle approche utilisant des fichiers inversés basés sur la contribution des images à l'inertie des axes. La nouvelle méthode réduit le temps d'exécution sans perte de qualité de résultat. Comme la plupart des méthodes de réduction de dimensions, une question concerne le nombre de dimensions à conserver. Dans la littérature, on conserve souvent 100 dimensions pour LSA et 30 pour l'AFC. On peut suggérer l'utilisation de la méthode Bayésienne de (Teh *et al.*, 2004) pour déterminer ce nombre.

Pour le passage à l'échelle, les méthodes basées sur la décomposition de la base d'images en sous-bases sont prometteuses. La base peut être décomposée en clusters et l'AFC est appliquée sur les clusters. Un algorithme heuristique sera utilisé pour sélectionner les clusters dans lesquels il faudra chercher la réponse à la requête. Une autre amélioration des résultats est de combiner l'AFC avec d'autres méthodes comme CDM (Contextual Dissimilarity Measure) (Jegou *et al.*, 2007) et/ou des méthodes d'apprentissage pour les rangs (Chu *et al.*, 2005, Burges, 2005).

Remerciements

Nous remercions A. Zisserman et ses collègues pour leurs données de Caltech4.

6. Bibliographie

Bentley J. L., « Multidimensional binary search trees used for associative searching », *Communications of the ACM*, vol. 18, n° 9, p. 509-517, 1975.

Benzécri J. P., *L'analyse des correspondances*, Paris : Dunod, 1973.

Blei D. M., Ng A. Y., Jordan M. I., « Latent dirichlet allocation », *Journal of Machine Learning Research*, vol. 3, p. 993-1022, 2003.

Bosch A., Zisserman A., Munoz X., « Scene Classification via pLSA », *Proceedings of the European Conference on Computer Vision*, 2006.

Burges C. J. C., « Ranking as Learning Structured Outputs », *Proceedings of the NIPS 2005 Workshop on Learning to Rank*, 2005.

Chu W., Ghahramani Z., « Extensions of Gaussian Processes for Ranking : Semi-Supervised and Active Learning », *Proceedings of the NIPS 2005 Workshop on Learning to Rank*, 2005.

Deerwester S., Dumais S., Furnas G., Landauer T., Harsman R., « Indexing by latent semantic analysis », *Journal of the American Society for Information Science*, 1990.

Fergus R., Perona P., Zisserman A., « Object Class Recognition by Unsupervised Scale-Invariant Learning », *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, p. 264-271, June, 2003.

Hofmann T., « Probabilistic latent semantic analysis », *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI'99)*, 1999a.

Hofmann T., « Probabilistic latent semantic indexing », *In Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR'99)*, 1999b.

Jegou H., Harzallah H., Schmid C., « A contextual dissimilarity measure for accurate and efficient image search », *Proceedings of CVPR'07*, 2007.

Lienhart R., Slaney M., « pLSA on large scale image databases », *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2007.

Lindeberg T., « Feature detection with automatic scale selection », *International Journal of Computer Vision*, vol. 30, n° 2, p. 79-116, 1998.

Lowe D. G., « Object recognition from local scale-invariant features », *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, p. 1150-1157, 1999.

Lowe D. G., « Distinctive image features from scale-invariant keypoints », *International Journal of Computer Vision*, p. 91-110, 2004.

Mikolajczyk K., Schmid C., « Scale and affine invariant interest point detectors », *Proceedings of IJC V*, vol. 60, n° 1, p. 63-86, 2004.

Morin A., « Intensive Use of Correspondence Analysis for Information Retrieval », *Proceedings of the 26th International Conference on Information Technology Interfaces, ITI2004*, p. 255-258, 2004.

Rocchio J., « Relevance feedback in information retrieval », *In G. Salton (ed.) : The SMART retrieval system : experiments in automatic document processing*, Prentice Hall, p. 313-323, 1971.

Schmid C., Mohr R., « Local grayvalue invariants for image retrieval », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, p. 530-535, 1997.

Sivic J., Russell B. C., Efros A. A., Zisserman A., Freeman W. T., « Discovering Object Categories in Image Collections », *Proceedings of the International Conference on Computer Vision*, 2005.

Teh Y. W., Jordan M. I., Beal M. J., Blei D. M., « Hierarchical dirichlet processes », *Proceedings of NIPS*, 2004.

Willamowski J., Arregui D., Csurka G., Dance C., Fan L., « Categorizing nine visual classes using local appearance descriptors », *Workshop Learning for Adaptable Visual Systems (ICPR 2004) Cambridge, United Kingdom*, 2004.