Méthodes d'apprentissage pour la recherche de catégories dans des bases d'images

Philippe-Henri Gosselin

Université de Cergy-Pontoise

THÈSE

présentée pour obtenir le titre de DOCTEUR en Sciences
Traitement de l'Image et du Signal

Méthodes d'Apprentissage pour la Recherche de Catégories dans des Bases d'Images

Philippe-Henri GOSSELIN

Soutenance prévue le 7 décembre 2005 devant le jury composé de :

MM.	J.P. Cocquerez,	Examinateur
	M. Cord,	Directeur de thèse
	P. Gallinari,	Rapporteur
	F. Jurie,	Rapporteur
	S. Marchant-Maillet,	Examinateur
Mme	S. Philipp-Foliguet,	Directrice de thèse

Table des matières

Notations

Général

Variable	Tous les objets mathématiques (scalaires, vecteurs, matrices, fonctions, ...) sont représentés par un et un seul symbole. Par exemple ab, n'est pas l'objet "ab", mais l'objet "a" multiplié par l'objet "b".
Scalaire	Les scalaires sont représentés par des lettres minuscules. Exemple : $i, j, k, x, y, ...$
Vecteur	Les vecteurs sont représentés par des lettres minuscules en gras. Exemple : $\mathbf{x}$. La ième valeur d'un vecteur est representée avec un indice i. Exemple : x_i. La variable n'est plus en gras puisque x_i est un scalaire.
Matrice	Les matrices sont représentées par des lettres majuscules en gras. Exemple : $\mathbf{X}$. Le ième vecteur/colonne d'une matrice est représenté avec un indice i. Exemple : $\mathbf{x}_i$. La valeur à la colonne i et la ligne j est représentée avec un indice ij. Exemple : x_{ij}.
Ensemble	Les ensembles sont représentés par des lettres majuscules. Exemple : B.
Grand Ensemble	Les ensembles d'ensembles sont représentés par des lettres majuscules calligraphiées. Exemple : $\mathcal{B}$.
Intervalles	Les intervalles de réels sont notés $[a, b] \subset \mathbb{R}$. Les intervalles d'entiers sont notés $[a..b] \subset \mathbb{Z}$.
Transposée	L'opérateur de transposition est $\top$. Exemple : $\mathbf{X}^\top$.
Produit Scalaire	Le produit scalaire est noté $\langle ., . \rangle$. Exemple : $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_i x_i y_i$.
Distance	Une distance est notée $d(\mathbf{x}, \mathbf{y})$.
E	Matrice identité : $e_{ij} = \delta_{ij}$.
$\mathbf{e}_i$	Vecteur unitaire de $\mathbf{E}$.

Base d'images

Base	Une base d'image est un ensemble d'images munies d'une signature et d'une fonction de similarité.
n	Nombre d'images dans la base.
$\mathcal{S}$	Ensemble de signatures. Par exemple, si les signatures sont des vecteurs de dimension p, $\mathcal{S} = \mathbb{R}^p$.
$S_i \in \mathcal{S}$	Signature de l'image i.

$s(.,.)$ Fonction de similarité :

$$\begin{array}{rccc} s: & \mathcal{S}\times\mathcal{S} & \rightarrow & \mathbb{R} \\ & S, S' & \mapsto & s(S, S') \end{array}$$

$\mathbf{S}$ Matrice des similarités :

$$\forall i, j \in [1..n]\ s_{ij} = s(S_i, S_j)$$

$\mathcal{C}$ Ensemble des catégories recherchées par l'utilisateur.
q Nombre de catégories.
$C \in \mathcal{C}$ Catégorie.

Quantification

X Ensemble des vecteurs à quantifier.
p Dimension des vecteurs de X.
n Nombre de vecteurs dans X.
W Ensemble des vecteurs représentants ou centres (*codewords*).
m Nombre de représentants.
q_W Quantificateur, qui à un vecteur fait correspondre son représentant :

$$\begin{array}{rccc} q_W: & \mathbb{R}^p & \rightarrow & \mathbb{R}^p \\ & \mathbf{x} & \mapsto & q_W(\mathbf{x}) = \underset{\mathbf{w}\in W}{\operatorname{argmin}}\ d(\mathbf{x}, \mathbf{w}) \end{array}$$

Fonctions Noyaux

$\Phi(.)$ Fonction d'induction dans un espace hilbertien $\mathcal{H}$:

$$\begin{array}{rccc} k: & \mathbb{R}^p & \rightarrow & \mathcal{H} \\ & \mathbf{x} & \mapsto & \Phi(\mathbf{x}) \end{array}$$

$k(.,.)$ Fonction noyau :

$$\begin{array}{rccc} k: & \mathbb{R}^p\times\mathbb{R}^p & \rightarrow & \mathbb{R} \\ & \mathbf{x}, \mathbf{y} & \mapsto & k(\mathbf{x}, \mathbf{y}) = \langle\Phi(\mathbf{x}), \Phi(\mathbf{y})\rangle \end{array}$$

$\mathbf{K}$ Matrice $n \times n$ de Gram sur la base :

$$\forall i, j \in [1..n]\ k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$$

Apprentissage supervisé

X Ensemble des vecteurs à classifier.

p Dimension des vecteurs de X.

n Nombre de vecteurs dans X.

$\mathbf{y}$ Vecteur $n \times 1$ d'annotation. Chaque valeur y_i est l'annotation de l'image i :

$$y_i = \begin{cases} 1 & \text{si l'image } i \text{ est pertinente} \\ -1 & \text{si l'image } i \text{ n'est pas pertinente} \\ 0 & \text{si aucune information n'est disponible} \end{cases}$$

I Ensemble des indices des images annotées :

$$I = \{i \mid y_i \neq 0\}$$

$\bar{I}$ Ensemble des indices des images non annotées :

$$I = \{i \mid y_i = 0\}$$

I_c Ensemble des indices des images d'annotation c :

$$I_c = \{i \mid y_i = c\}$$

$f_{\mathbf{y}}(.)$ Fonction de pertinence après entraînement avec $\mathbf{y}$. Cette fonction renvoie à un vecteur $\mathbf{x}$ sa pertinence :

$$\begin{array}{rccc} f_{\mathbf{y}} : & \mathbb{R}^p & \rightarrow & [-1,1] \\ & \mathbf{x} & \mapsto & f_{\mathbf{y}}(\mathbf{x}) \end{array}$$

Selon le contexte, l'ensemble d'apprentissage $\mathbf{y}$ peut être omis.

Chapitre 1

Introduction

1.1 Présentation du problème

Avec la démocratisation des appareils multimédia, de plus en plus d'images numériques sont générées chaque jour. Entre les ordinateurs, les scanneurs, les webcams, les téléphones portables avec appareil photo, de plus en plus de personnes sont en mesure de diffuser sur les réseaux privés et publics des images numériques. La diminution du coût de stockage et la disponibilité de techniques de numérisation de haute qualité permettent aussi aujourd'hui de consistuer de très grandes bases d'images dans des domaines variés :

- Bases médicales ;
- Bases d'archives (patrimoine culturel, musées, ...) ;
- Bases d'agences photographiques, bases personnelles ;
- Bases d'images satellites et aériennes.

Ces bases sont souvent immenses et généralement sous-exploitées faute d'outils pour accéder à l'information qu'elles contiennent. Le traitement de ces bases a ouvert plusieurs champs de recherche :

- *Stockage.* Certaines bases d'images sont colossales, et un système matériel et logiciel est nécessaire pour pouvoir stocker sans dégradation toutes ces informations, tout en gardant à l'esprit qu'elle doivent rester accessibles.
- *Partage.* Plusieurs utilisateurs doivent pouvoir accéder aux informations des bases en même temps. Des problèmes de sécurisation des accès, de protection et d'intégrité des données se posent.
- *Recherche.* Une fois les images stockées et accessibles, elles restent toutefois noyées au sein d'une très grande quantité d'information. Bien que l'on puisse retrouver une image par un index informatique (identifiant, fichier, codage, ...), il est très difficile – voir impossible – de retrouver une image que l'on peut avoir en tête. Ce champ traite des techniques d'interrogation et de recherche dans les bases.

Dans cette thèse, nous nous intéressons aux techniques de recherche. Parmi ces techniques, on peut trouver deux grandes approches : la recherche par mots clefs, et la re-

cherche par le contenu. La première repose sur un ensemble de mots clefs qui ont été associés à chaque image de la base, et qui sont ensuite utilisés par une stratégie de recherche textuelle. Cette approche requiert un processus manuel de description de chaque image de la base par des mots clefs. La deuxième approche, à laquelle nous nous intéressons dans cette thèse, consiste à représenter chaque image de la base par un ensemble de caractéristiques visuelles, par exemple les couleurs, les textures ou les formes rencontrées dans une image. Ces caractéristiques visuelles, calculées de manière automatique, sont ensuite exploitées par le système pour comparer et retrouver des images.

Une stratégie de recherche d'images par le contenu (*Content-Based Image Retrieval*) dépend tout d'abord du type de base d'images que l'on souhaite traiter. On peut distinguer deux cas (Smeulders et al., 2000) : les bases spécialisées, contenant un type réduit d'images (par exemple les bases médicales, où des images d'un seul organe ont été acquises), et les bases généralistes, contenant des images de natures très variées. Les stratégies de recherche dans les bases spécialisées se distinguent de celle dans les bases généralistes par le fait que l'on connaît par avance le type d'images rencontré, ainsi que les recherches qui vont y être menées. Par exemple, pour les bases médicales on souhaite en général différencier les organes sains des organes malades. Ces précisions permettent la mise en place de techniques de caractérisation visuelles très performantes, ainsi qu'une stratégie de recherche bien adaptée aux requêtes des utilisateurs. La problématique pour ce type de base est avant tout la construction de ces outils dédiés.

Pour les bases généralistes auxquelles nous nous intéressons dans cette thèse, le problème est différent, puisque l'on ne connaît pas ou peu les requêtes des utilisateurs. On distingue généralement deux niveaux :

- le bas niveau dit « numérique », qui fait référence au contenu signal de l'image, autrement dit l'ensemble des pixels qui la constitue.
- le haut niveau dit « sémantique », qui fait référence au contenu interprétable de l'image, autrement dit l'ensemble des objets et significations que l'on peut y associer.

Il existe souvent entre la description bas-niveau et la sémantique d'une image une différence forte, que l'on appelle le *fossé sémantique/numérique*. Prenons l'exemple de recherche dans la figure 1.1, où l'on suppose qu'un utilisateur est à la recherche d'images de portes. L'utilisateur a utilisé une de ses images personnelles en guise de requête (celle en haut à gauche avec un petit carré vert), et le système a déterminé les images de la base les plus similaires. Le calcul de similarité se fait avec des descriptions bas-niveau, plus précisément la couleur et la texture. Les images dans la figure sont présentées par similarité décroissante, de gauche à droite et de haut en bas. Comme nous pouvons le constater, beaucoup d'images de portes ont en commun les mêmes caractéristiques visuelles (contours verticaux, lignes horizontales), que les caractéristiques visuelles calculés représentent aisément. En revanche, sur l'exemple de la figure 1.2, où l'utilisateur recherche des images de chiens, les caractéristiques visuelles calculés semblent beaucoup moins corrélées avec le type d'image que l'utilisateur a en tête.

Outre le fossé qui peut exister entre la caractérisation visuelle d'un objet et sa sémantique, une image peut aussi contenir plusieurs significations, dépendantes du contexte. Par exemple une peinture de portrait de Modigliani peut être placée selon le

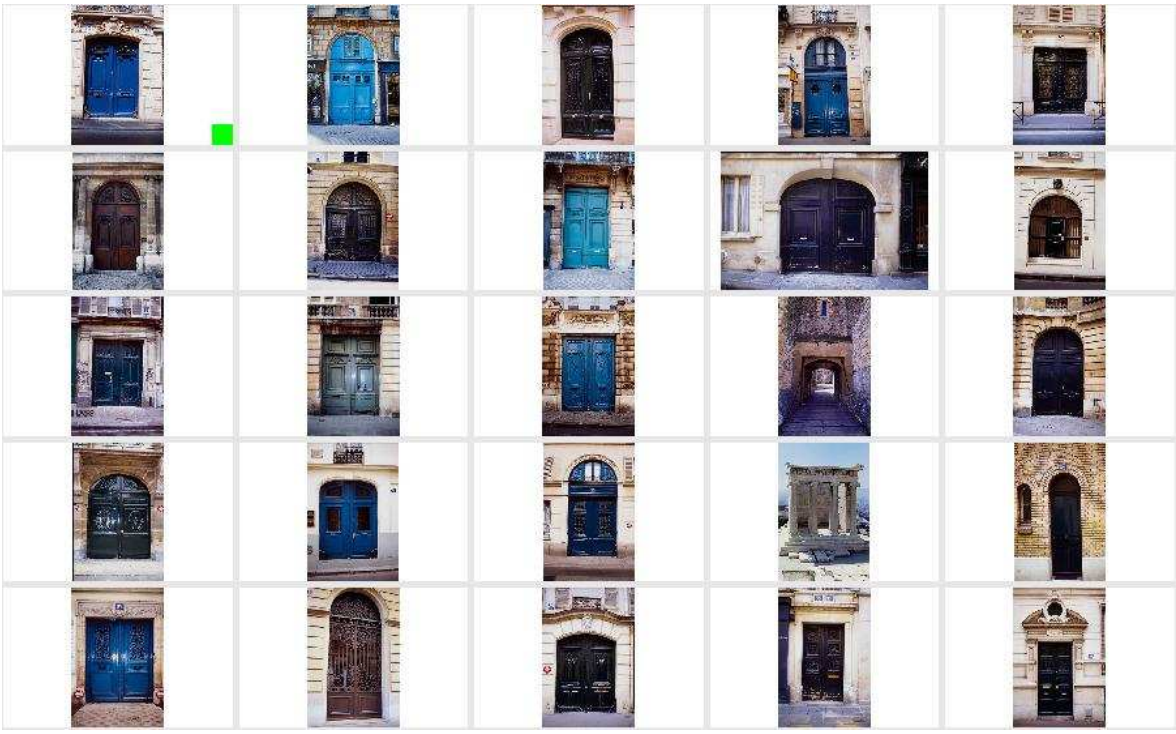


FIG. 1.1 – Exemple de recherche où les caractéristiques visuelles et la sémantique sont proches.

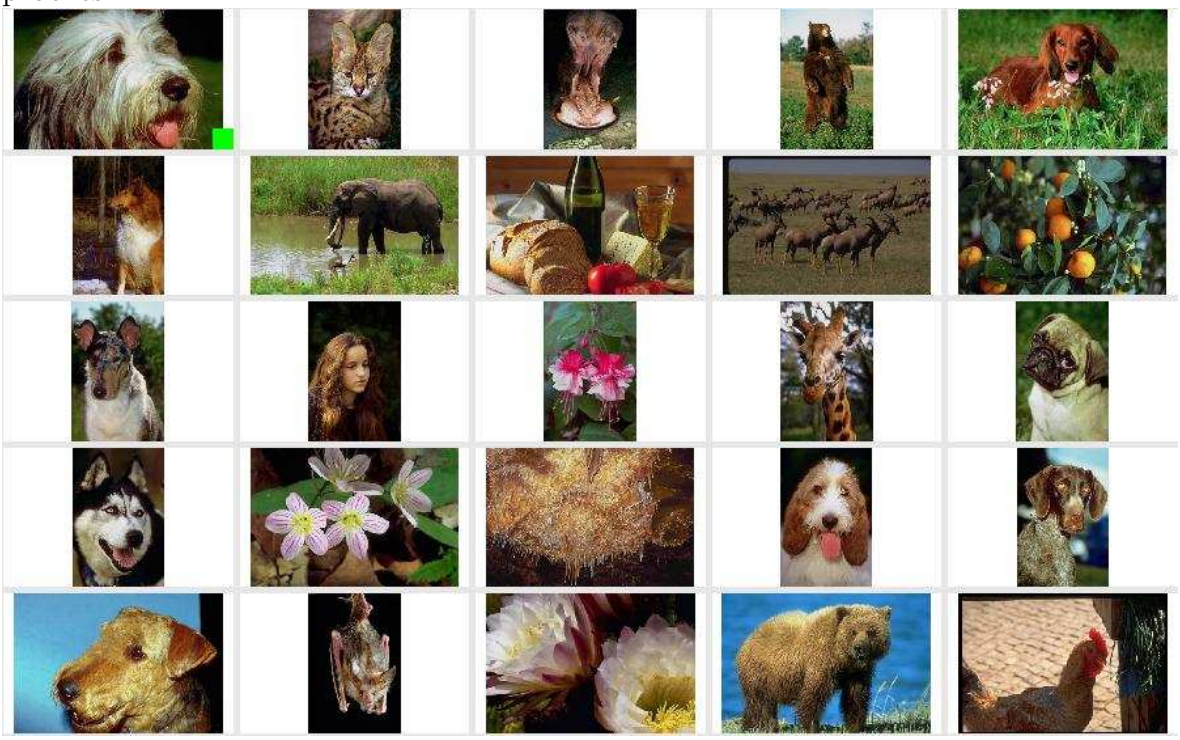


FIG. 1.2 – Exemple de recherche où les caractéristiques visuelles et la sémantique sont éloignées.

contexte dans la catégorie « peintures », ou dans la catégorie « portraits »(Santini et al., 2001). Ainsi, selon le contexte de la recherche, la similarité sémantique entre les images peut être très différente. Ceci montre l'importance du fossé qui existe entre le bas-niveau et le haut-niveau, puisqu'au delà de la difficulté à déterminer des caractérisations visuelles correspondant à la sémantique, le contexte est aussi un facteur important.

Dans le but de combler le fossé sémantique/numérique, le premier point à traiter concerne ce que nous appelerons la *Représentation de la base d'images*. Celle-ci est en général déterminée en amont de l'utilisation du système de recherche[1]. Concernant ce point, plusieurs outils sont utilisés :

- *Les primitives visuelles*. Cette première phase consiste à extraire les parties de l'images, par exemple des points d'intérêt, des régions, des zones d'intérêt, etc. Ces parties sont en général choisies de manière à extraire les zones les plus informatives de l'image. Un choix courant est aussi de simplement considérer tous les pixels de l'image.
- *Les caractéristiques visuelles* ou *descripteurs visuels* ou *attributs*. C'est la phase de caractérisation des couleurs, textures, formes, etc. présentes dans chaque primitive.
- *Les signatures* ou *index*. Les caractéristiques ne sont pas toujours directement exploitables, et nécessitent d'être traitées lors de la phase dite d'*indexation*. Un choix courant est de représenter chaque image par un vecteur, mais on rencontre aussi des systèmes où les signatures sont des ensembles de vecteurs, des graphes, etc.
- *La similarité*. Elle permet de comparer les images entre elles à partir de leur signatures. Le choix et le paramétrage de la similarité vont souvent de pair avec les signatures.

Une fois la représentation choisie, on peut aborder la stratégie de recherche en elle-même. Il existe différents systèmes que l'on peut classer en fonction du but de l'utilisateur. Dans cette thèse, nous nous intéressons à la *recherche interactive* de catégories. L'utilisateur fournit des informations au système dans le but de préciser la nature de sa recherche.

Enfin, le dernier point abordé dans cette thèse concerne l'*apprentissage long terme*. Ce type d'apprentissage vise à réutiliser toutes les informations qu'ont fournies les utilisateurs lors d'une session de recherche. Le but est d'améliorer le système à l'aide de cette connaissance, en modifiant la représentation de la base.

La problématique de la recherche d'images par le contenu est complexe, et les différents points que nous traitons ici (représentation, recherche interactive et long terme) sont liés. La modélisation du problème est une tâche difficile car tout choix dans l'un des points a des conséquences pour les autres, et outre les différentes méthodes que nous proposons dans les chapitres suivants, la principale contribution de cette thèse réside dans la présentation d'une architecture pour la recherche d'images et la mise en évidence des différents sous-problèmes qu'il y a à traiter.

[1]Cette étape d'initialisation du système est dite « hors-ligne », à la différence d'un système en relation avec des utilsateurs, qui est alors dit « en-ligne ».

1.2 Représentation d'une base d'images

1.2.1 Caractéristiques visuelles et signatures

Le contenu numérique brut des images ne peut être directement exploité pour la comparaison d'images. Il est nécessaire de passer par une opération d'*extraction de caractéristiques visuelles* (ou *attributs*) qui permet d'obtenir une représentation plus facilement manipulable. Idéalement, cette opération a pour but d'extraire le contenu sémantique d'une image. Autrement dit, extraire toutes les caractéristiques visuelles qui permettent de déterminer les objets ou significations présentes dans chaque image de la base. Les différentes caractéristiques visuelles sont exprimées dans un *espace de caractéristiques visuelles* (ou *espace des attributs*), par exemple le cube unitaire de $\mathbb{R}^3$ pour les couleurs RVB.

Pour certaines applications dédiées, il est possible de construire des caractérisations visuelles très performantes, si les catégories sont connues à l'avance et définissables à partir d'un contenu visuel très précis. Par exemple, une des problématiques au C2RMF (Centre de Restauration et de Recherche des Musées de France) est de classifier les paquetages des tableaux (lattes de bois qui soutiennent la toile) en fonction du nombre de lattes, leur orientation, la texture du bois, *etc.* Dans ce cas précis des descripteurs puissants ont été développés, et la classification a été très simple à réaliser. Cependant, l'objectif de cette thèse est de construire un système capable de s'affranchir des hypothèses fortes de ce type d'application.

Dans le contexte des bases généralistes, les caractéristiques visuelles sont très variées et vont de descripteurs génériques (couleur, texture, forme, *etc.*) à des caractérisations locales par zones d'intérêt (points, blocs, régions, *etc*).

Dans les deux cas, il est souvent nécessaire d'effectuer une étape dite d'*indexation*, qui permet le calcul d'une *signature* pour chaque image. En effet, certaines caractéristiques visuelles fournissent une description propre à l'image où elle a été calculée, et la comparaison avec la description d'une autre image n'est pas triviale. Dans d'autres cas, les descripteurs sont soit très grands soit très nombreux, et l'indexation est alors nécessaire pour réduire leur taille ou leur nombre.

1.2.2 Similarité

La similarité est à la base de tout système de recherche d'images, et permet de comparer les images à l'aide de leurs signatures calculées lors de l'indexation. Il existe de nombreuses fonctions de similarité, allant d'une simple distance euclidienne entre vecteurs, à des fonctions *ad hoc* dédiées à un type particulier de caractérisation visuelle. Parmi toutes ces techniques, l'une d'entre elles consiste à utiliser une fonction noyau (Schölkopf & Smola, 2002), qui ont été récemment introduites et popularisées par les Supports à Vaste Marge (SVM) (Boser et al., 1992). Ces fonctions offrent un cadre théorique qui permet la transformation d'un espace de signatures. Elles injectent cet espace initial dans

un espace linéaire de grande dimension. L'idée sous jacente à ce processus d'injection est de séparer le problème de l'apprentissage du problème de la représentation. Cette approche commence à apparaître dans la communauté de traitement de l'image, où l'idée est de voir une fonction noyau comme une fonction de similarité.

Une fois que les signatures et la fonction de similarité sont déterminées, nous obtenons une *représentation de la base* :

- Une signature S_i pour chaque image i, qui peut être sous une forme simple (un vecteur) ou plus complexe (un ensemble de vecteurs, un graphe, ...) ;
- Une fonction capable de mesurer la similarité visuelle $s(i, j)$ entre deux images i et j de la base.

1.3 Systèmes de recherche

1.3.1 Classement des systèmes par objectif

Il existe diverses méthodes de recherche d'images par le contenu proposées dans la littérature. Une première classification de ces méthodes consiste à considérer le but visé par l'utilisateur. Introduite par (Cox et al., 2000) et reprise par (Smeulders et al., 2000) elle distingue trois grandes catégories :

- la recherche associative ;
- la recherche de cible ;
- la recherche de catégorie ;

En recherche associative, l'utilisateur n'a pas de but précis. Il ne possède pas d'exemple de ce qu'il souhaite retrouver, et a seulement une vague idée de ce qu'il recherche. Dans ce type de recherche, c'est l'exploration de la base d'images qui permet à l'utilisateur de définir et d'affiner son objectif.

Contrairement à la recherche associative, la recherche de cible a un but clairement défini, à savoir retrouver une image particulière. Une stratégie utilisée pour ce faire est de procéder par élimination, en ignorant des parties de la base, jusqu'à ce qu'il ne reste plus que l'image désirée.

La recherche de catégorie s'intéresse à un sous-ensemble d'images de la base. On parle aussi de *classe* ou de *concept*. Dans ce cas, l'utilisateur souhaite retrouver des images possédant certaines caractéristiques, par exemple les images qui contiennent tel ou tel type d'objet.

Dans cette thèse, nous nous intéressons à la recherche de catégories. Pour ce faire, différentes méthodes sont proposées.

1.3.2 Recherche de catégories

Afin de déterminer la catégorie d'une image à partir de son contenu, plusieurs approches peuvent être distinguées.

Les premières techniques de catégorisation sont entièrement automatiques. Elle ont pour but de classifier la base d'images uniquement en s'appuyant sur des caractérisations visuelles. L'utilisateur fournit une image au système qui détecte alors la catégorie à laquelle elle appartient. Bien que les caractéristiques visuelles soient de plus en plus puissantes, cette stratégie reste insuffisante dans le cas où le fossé sémantique/numérique est important. En effet, une image peut avoir plusieurs interprétations différentes, et ainsi appartenir à plusieurs catégories à la fois.

Les autres techniques emploient un ensemble d'exemples. L'idée est de construire un détecteur pour chaque catégorie à partir d'exemples. Si l'on connaît la catégorie de quelques images, on peut optimiser la fonction de similarité sur la base. Le principe reste le même que dans le cas automatisé, à la différence que des détecteurs peuvent être construits pour toute catégorie particulière. De plus, cela permet une meilleure gestion des images qui appartiennent à plusieurs catégories. S'appuyer sur des exemples est l'approche qui nous semble la plus adaptée pour combler le fossé sémantique/numérique.

La construction de l'ensemble des exemples peut se faire de différentes manières. On peut procéder en deux temps, un temps pour la construction de l'ensemble des détecteurs, et un temps pour leur exploitation. Cela suppose de connaître à l'avance les catégories qui vont être recherchées. Une autre approche, sur laquelle nous nous sommes focalisés dans cette thèse, est de construire des détecteurs « à la demande », directement durant le processus de recherche, et non lors d'une phase postérieure, à partir d'une interaction entre de l'utilisateur et le système de recherche.

Dans ce type de recherche interactive, l'utilisateur est invité à donner des précisions sur sa recherche, le plus souvent sous la forme d'exemples. Les informations fournies par l'utilisateur sont alors exploitées pour améliorer le détecteur dans une phase dite de bouclage de pertinence (*relevance feedback*). Ainsi, pour chaque session de recherche d'un utilisateur, le sytème construit un détecteur personnalisé, entièrement dédié aux images recherchées.

1.3.3 Apprentissage long terme

Avec l'émergence de systèmes de recherche disponibles sur les réseaux, de plus en plus d'information sur l'utilisation de ces systèmes sont disponibles. Il devient alors intéressant d'exploiter ces informations pour améliorer la qualité des recherches futures. Ce processus de réutilisation des informations fournies par les utilisateurs est appelé *apprentissage long terme*, à la différence de la recherche interactive, qui fait partie de l'*apprentissage court terme* (Vasconcelos & Kunt, 2001).

Il existe plusieurs approches que l'on peut qualifier de long terme. Les méthodes d'apprentissage en deux temps que nous avions mentionnées au paragraphe précédent en font partie. Dans le cas où les utilisateurs spécifient la catégorie qu'ils recherchent à chaque session, par exemple en donnant un mot clef, on dispose alors d'un ensemble de lots d'annotations associées à différentes catégories. Il est alors envisageable de recouper les lots qui sont associés à la même catégorie, et d'obtenir ainsi un ensemble d'apprentissage pour chaque catégorie. Des détecteurs peuvent alors être construits de manière supervisée.

Dans le cadre généraliste où nous travaillons, il semble difficile de faire l'hypothèse que la catégorie recherchée soit spécifiée à chaque fin de session. En effet, cela suppose que l'utilisateur accepte de préciser la nature de sa recherche, et que toutes les catégories puissent facilement être décrites, par exemple par un mot clef. Dans le cas où cette information n'est pas disponible, le problème d'apprentissage long terme peut alors être qualifié de *faiblement supervisé*.

Ce type d'apprentissage est plus difficile, puisqu'il n'est pas possible de recouper trivialement les lots d'annotations pour en déduire ne serait-ce que le nombre de catégories recherchées. Dans cette thèse, nous nous intéressons à ce problème, où l'objectif est d'améliorer la qualité des sessions futures en s'appuyant sur des annotations concernant des catégories inconnues.

1.4 Contributions et plan de thèse

Nous présentons dans ce paragraphe les différents problèmes pour la recherche d'image que nous abordons dans cette thèse.

1.4.1 Signatures et similarité

Nous nous intéressons à la quantification vectorielle souvent nécessaire pour le calcul de signatures. La quantification vectorielle fait partie des problèmes d'apprentissage automatique ou *non supervisé*, *i.e.* qui ne s'appuie que sur l'ensemble des individus (ici tous les vecteurs de caractérisation visuelle) pour apprendre. La spécifité de la quantification vectorielle dans le cadre de l'indexation d'une base d'images provient du nombre conséquent de vecteurs à quantifier. Nous présentons en détail ce problème au chapitre 2, et proposons une méthode rapide et efficace pour quantifier un très grand nombre de vecteurs.

Nous avons opté pour une représentation de la similarité par fonctions noyaux. Ainsi, toutes les solutions auxquelles nous nous intéressons sont entièrement basées sur ce formalisme. Nous présentons quelques aspects de cette théorie dans notre contexte au chapitre 3. Nous y proposons une étude des différentes fonctions noyaux classiques, ainsi qu'une méthode d'analyse de leurs spectres pour l'aide au choix d'une fonction particulière.

1.4.2 Classification

Parmi les différentes approches d'apprentissage interactif, nous nous intéressons au bouclage de pertinence, avec une approche statistique. Nous modélisons la recherche d'une catégorie par un problème de classification binaire, où l'on doit construire un classifieur pour discriminer les images qui appartiennent à la catégorie (les *images pertinentes*) des autres (les *images non pertinentes*).

L'approche par classification offre la capacité à représenter des catégories complexes. La théorie de l'apprentissage dans ce domaine est très riche, et permet de se reposer sur des propriétés largement éprouvées. Nous présentons au chapitre 4 une étude des méthodes de classification utilisées pour la recherche d'image.

Un dernier point sur notre motivation quant à l'approche par classification concerne l'apprentissage actif, qui est souvent présenté avec des techniques de classification.

Ces travaux ont été publiés dans :
(Gosselin et al., 2004b)
(Gosselin et al., 2004a)
(Cord et al., 2005a)

1.4.3 Apprentissage actif

L'*apprentissage actif* est une approche qui s'intéresse au problème de la *sélection* des images à faire annoter par l'utilisateur. Les techniques qui lui sont apparentées, dites *actives*, vont déterminer les images qui, une fois annotées, donneront le meilleur résultat. Par exemple, un *classifieur actif* va *sélectionner* les images qui permettent de minimiser ensuite l'erreur de classification.

Dans le cas où l'on a la possibilité d'interagir avec un utilisateur, les exemples arrivent les uns après les autres, en fonction du résultat précédent. La suite d'exemples a un impact important sur la qualité du résultat final. En effet, une suite d'exemples peut donner de bien meilleurs résultats qu'une autre. Par exemple, si l'on demande à l'utilisateur d'annoter des images qui sont clairement dans l'une ou l'autre classe, compte tenu de la puissance des techniques d'apprentissages actuelles, le résultat restera inchangé.

Diverses solutions ont été proposées afin d'optimiser la sélection des images à faire annoter par l'utilisateur. Très sommairement, nous pouvons séparer ces techniques en deux catégories :

– La première catégorie, que nous qualifions de *pessimiste*, consiste à assurer un certain résultat pour une quantité d'exemples donnés. Ces techniques s'intéressent à l'ensemble des annotations possibles, et minimisent l'espace des possibilités. (Tong & Koller, 2001) proposent de diviser en deux un espace de classifieurs possibles pour chaque nouvel exemple, quelle que soit sa classe. Cette technique garantit qu'au bout d'un nombre

t d'exemples, l'espace des possibilités sera réduit d'autant. Cependant, ces techniques souffrent entre autres problèmes du fait que l'espace des possibilités est gigantesque, et une réduction suffisante de cet espace peut nécessiter beaucoup d'itérations.

– La deuxième catégorie, que nous qualifions d'*optimiste*, consiste à sélectionner les images qui, une fois ajoutées à l'ensemble d'apprentissage, amélioreront le plus le résultat, au sens d'un critère à optimiser. Par exemple, (Roy & McCallum, 2001) proposent de sélectionner les éléments qui maximiseront la généralisation du classifieur. Cependant, ces techniques tentent de minimiser un risque. En effet, elles sélectionnent l'image qui, sous réverve qu'elle prenne une certaine annotation, maximisera le critère. Or, si l'utilisateur ne donne pas l'annotation supposée, la classification n'évolue pas.

Nous présentons au chapitre 5 un état de l'art en classification active. Nous y présentons en détail les deux méthodes que nous venons de mentionner, sur lesquelles nous nous appuyons pour proposer des solutions au chapitre 6. Le contexte d'apprentissage actif en recherche d'image est particulier, et l'utilisation directe des techniques de classification n'est pas le meilleur choix. Nous proposons trois méthodes d'apprentissage actif, avec l'idée de trouver une approche qui corresponde aux contraintes particulières de la classification active pour la recherche d'images.

Ces travaux ont été publiés dans :
(Gosselin & Cord, 2004a)
(Gosselin & Cord, 2004b)
(Gosselin & Cord, 2005a)
(Cord et al., 2005b)

1.4.4 Apprentissage long terme

Enfin, nous nous intéressons au problème de l'apprentissage long terme dit *faiblement supervisé*, où les informations à propos des sessions passées sont imprécises. Nous commençons par décrire cette problématique en détail au chapitre 7, ainsi que les solutions de la littérature qui pourraient être utilisées.

Dans l'idée de rester dans l'approche globale par fonction noyau, nous proposons deux méthodes d'apprentissage long terme basées sur ce type de similarité. La première, présentée au chapitre 8, est une méthode qui modifie directement les similarités entre les images, tout en respectant les propriétés des fonctions noyaux. La deuxième, présentée au chapitre 9, est une méthode de modification indirecte des similarités, en travaillant sur un déplacement des vecteurs images dans l'espace des signatures. Nous présentons au chapitre 10 une étude expérimentale des deux méthodes, en fonction de différentes possibilités d'application.

Ces travaux ont été publiés dans :
(Gosselin & Cord, 2004c)
(Gosselin & Cord, 2005c)
(Gosselin & Cord, 2006)

Première partie

Représentation de la base d'images

Chapitre 2

Calcul d'histogrammes par quantification vectorielle

Suite à l'extraction des caractéristiques visuelles de chaque image, il est nécessaire de passer par un processus dit d'indexation, afin de former une signature pour chaque image. Cependant, même lorsqu'on dispose d'un ensemble de vecteurs caractérisant l'image, la comparaison avec un autre ensemble de vecteurs n'est pas triviale. Une solution pour pallier ce problème est le calcul d'un ensemble de vecteurs caractéristiques commun à toute la base ; par exemple, une palette des couleurs les plus utilisées dans la base. Ainsi, pour chaque image de la base, nous pouvons calculer un histogramme sur cette référence commune. Dans ce cas, ces histogrammes forment les signatures couleur des images de la base. Par la suite, comparer deux images reviendra à calculer la distance entre deux histogrammes.

Le processus de détermination de cette palette commune présenté dans ce chapitre est la quantification vectorielle de l'ensemble des vecteurs caractéristiques de toutes les images de la base. La quantification vectorielle est un important instrument utilisé en sciences et en ingénierie. Ses applications sont, par exemple : la reconnaissance d'objets, la segmentation, la vision informatique, la compression de données, etc.

Le but de ce chapitre est de justifier le choix d'un algorithme de quantification dans le cadre de l'indexation. Nous présenterons tout d'abord le problème de la quantification, puis les deux méthodes classiques (Nuées Dynamiques), les améliorations faites par Patanè (Patanè, 2001), et enfin son utilisation pour l'indexation.

2.1 Les caractéristiques visuelles

Dans cette thèse où l'apprentissage est notre problématique première, nous avons choisi deux caractéristiques simples : la couleur et la texture. Dans les deux cas, l'extraction fournit un ensemble de points pour chaque image, et l'ensemble de ces ensembles est quantifié pour calculer les signatures des images.

2.1.1 Couleur

La principale caractéristique utilisée dans un système d'indexation à vocation généraliste est la couleur.

Il existe de nombreux espaces couleur, par exemple RVB, YCrCb, HSV, $L^\star a^\star b^\star$. L'espace RVB (Rouge, Vert, Bleu) est l'espace colorimétrique le plus utilisé en informatique pour stocker les images. Bien qu'il soit adapté aux écrans d'ordinateur, cet espace ne dispose pas de propriétés intéressantes en terme d'invariance ou sur le plan cognitif. L'espace YCrCb est utilisé en compression d'images car il redistribue les couleurs en fonction de la réponse de l'œil humain. La modification est telle que la majeure partie de l'information perçue est placée sur l'axe Y, ce qui offre des avantages en terme de codage. Dans le souci de se rapprocher de la perception humaine de la couleur, l'espace HSV (Hue, Saturation, Value, c'est à dire Teinte, Saturation, Valeur) sépare les composantes d'intensité, de teinte (ou couleur pure) et de saturation (quantité de blanc). Enfin, l'espace $L^\star a^\star b^\star$ est proche de HSV, à la différence que les couleurs perçues par l'œil sont distribuées de manière uniforme dans un cube de $\mathbb{R}^3$ (Carrilero, 1999). Ce dernier espace, que nous avons utilisé pour nos expérimentations, est particulièrement intéressant lorsqu'un processus de quantification vectorielle basée sur une distance euclidienne est utilisé.

2.1.2 Texture

Même s'il n'existe pas de définition universelle de la texture, elle est généralement appréhendée comme « une structure spatiale constituée de l'organisation de primitives ayant chacune un aspect aléaloire » (Gagalowicz, 1983). On trouve de nombreuses méthodes référencées dans la littérature, dont des méthodes basées sur la décomposition paramétrique de Wold utilisé par (Liu & Picard, 1996) et (Stoica et al., 1998), les matrices de cooccurences (Aksoy & Haralick, 2001), les modèles autorégressifs multi échelle et les filtres de Gabor (Heinrichs et al., 2000; Manjunath & Ma, 1996; Rubner, 1999).

Pour le calcul des attributs de texture, nous utilisons un banc de filtre de Gabor (Gabor, 1946). En effet, outre son efficacité établie en reconnaissance de formes texturées et pour la modélisation des champs récepteurs des cellules du cortex visuel, différentes comparaisons réalisées dans le cadre de la recherche d'images placent cette approche parmi les techniques les plus efficaces (Ma & Manjunath, 1995). Ce banc de filtres réalise la décomposition spectrale du signal en fréquence et en orientation.

Nous utilisons 12 filtres :

- Quatre orientations : horitontale, verticale et les deux diagonales ;
- Trois bandes de fréquences : les basses, les moyennes et les hautes fréquences.

L'extraction s'effectue en passant les 12 filtres sur l'image. Pour chacun de ces filtres, nous obtenons une énergie pour chaque pixel de l'image. Nous obtenons ainsi un ensemble de vecteurs de dimension 12.

2.2 Signatures

Certaines techniques de caractérisation, bien qu'elles extraient une information visuelle, ne fournissent pas directement une signature pour comparer les images entre elles. Une étape d'indexation est nécessaire, qui consiste à construire une signature pour chaque image de la base.

Il existe plusieurs types de primitives pour l'indexation : le pixel, le bloc de pixel, la région, le point d'intérêt, *etc.* Deux approches courantes sont possibles pour construire les signatures. La première, l'approche globale, consiste en l'accumulation de l'information sur toute l'image pour la formation de la signature. La deuxième, l'approche locale, consiste à construire une signature sous la forme d'un ensemble de caractérisations pour chaque primitive de l'image, par exemple la couleur pour chaque région de l'image.

Un type de signature globale très courant est l'histogramme. A un facteur de normalisation près, l'histogramme constitue une approximation de la densité associée à l'image, vue comme une variable aléatoire. Cette approche statistique est proche de la recherche textuelle pour laquelle les documents sont couramment caractérisés par la fréquence d'apparition de mots-clef. Les signatures par histogrammes sont invariantes à des modifications globales de l'image : la translation, la rotation, et le changement d'échelle. Le calcul de ces signatures se fait en utilisant un dictionnaire de caractéristiques visuelles, par exemples toutes les teintes de rouge, tous les contours verticaux, *etc.* La détermination de ce dictionnaire se fait généralement par une quantification vectorielle de l'espace des caractéristiques visuelles.

Pour les techniques de caractérisation hors histogrammes, l'étape d'indexation peut être plus complexe. Par exemple, pour les techniques par points d'intérêt, chaque image est décrite par un ensemble de descripteurs (un descripteur par point d'intérêt) (Mikolajczyk et al., 2004). On peut être aussi amené à construire un dictionnaire de ces descripteurs, au même titre que pour les histogrammes (Jurie & Triggs, 2005).

Dans cette thèse, nous utilisons des signatures par histogramme. L'étape d'indexation se décompose alors en deux sous-étapes :

1. Calcul du dictionnaire de caractéristiques visuelles ;
2. Calcul des histogrammes (ou distributions).

Le dictionnaire est déterminé soit de manière statique, soit de manière dynamique. La quantification statique est un pré-découpage de l'espace des caractéristiques visuelles, par exemple l'ensemble des couleurs que les personnes peuvent distinguer. Au contraire, la quantification dynamique va déterminer de manière automatique l'ensemble des caractéristiques les plus pertinentes dans une base d'image, et construit ainsi un dictionnaire sur mesure.

Par exemple, un histogramme couleur[1] H_i peut être défini selon une palette B_i propre à l'image i. Cependant, certaines couleurs seront présentes dans l'image i mais

[1]L'exemple que nous présentons ici concerne les attributs couleurs, mais est aussi valable pour les autres types d'attributs.

pas nécessairement dans une autre image j. En quantification statique, afin de pouvoir détecter toutes les tonalités de couleurs, une très grande palette est nécessaire. Ceci pose deux problèmes. Le premier est sur le plan calculatoire, puisqu'il faut une grande quantité de ressources mémoire et processeur. De plus, une grande partie de ces ressources ne sont pas exploitées, puisque les histogrammes sont tous presque vides. Le deuxième problème est qu'une telle approche augmente de manière significative les problèmes liés aux très grandes dimensions (Hastie et al., 2001b).

La quantification dynamique permet de fabriquer une palette $\hat{B}$ commune à tous les histogrammes, et adaptée à la base d'image. Cela revient à déterminer les couleurs les plus présentes dans la base, puis calculer les histogrammes $\hat{H}_i$ pour la seule et unique palette $\hat{B}$. Ce problème de quantification se distingue par la quantité astronomique de vecteurs à quantifier, qui peut dépasser aisément le milliard d'individus. La signature $\hat{H}_i$ résultante a l'avantage d'être compacte et de faible dimension.

Dans cette thèse, nous utilisons des histogrammes avec un dictionnaire fabriqué à partir d'une quantification vectorielle. Nous présentons à la suite deux méthodes classiques de quantification, ainsi qu'une méthode optimisée (Patanè, 2001) pour réduire de manière significative les problèmes de minima locaux des deux autres méthodes. Enfin nous abordons le problème de la quantification vectorielle dans notre contexte d'indexation, où le nombre de vecteurs à quantifier est extraordinairement élevé. Nous proposons une méthode rapide et efficace pour traiter ce problème.

2.3 Quantification Vectorielle

Le but de la quantification vectorielle est de faire correspondre à un ensemble de vecteurs un ensemble de *représentants* (ou *codewords*) dont le nombre est très inférieur, par exemple, le passage de plusieurs dizaines de milliers de couleurs dans une image à quelques centaines.

2.3.1 Définitions

Soit $X = \{\mathbf{x}_1 ... \mathbf{x}_n\}$ l'ensemble des vecteurs de $\mathbb{R}^p$ à quantifier, $W = \{\mathbf{w}_1, ..., \mathbf{w}_m\}$, $m \leq n$, l'ensemble des représentants à déterminer, et $d : \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}$ une distance. Dans notre cas, l'espace des vecteurs ($\mathbb{R}^p$) est l'espace des caractérisations visuelles ($L^\star a^\star b^\star$ ou les sorties des 12 filtres de Gabor), et la distance utilisée est la distance euclidienne.

Définition 2.1 (Quantificateur) *Un quantificateur q_W est défini par :*

$$\begin{array}{rccl} q_W : & \mathbb{R}^p & \rightarrow & W \\ & \mathbf{x} & \mapsto & q_W(\mathbf{x}) = \underset{\mathbf{w} \in W}{\operatorname{argmin}}\ d(\mathbf{x}, \mathbf{w}) \end{array}$$

Pour un ensemble de représentant W donné, il est possible de déterminer une partition $\mathcal{C}^W$ de l'ensemble X constituée par les sous-ensembles $C^W_{\mathbf{w}}$ de $\mathbf{X}$.

Définition 2.2 (Classe) *La classe $C^W_{\mathbf{w}}$ d'un représentant $\mathbf{w} \in W$ sur $\mathbf{X}$ est définie par :*

$$C^W_{\mathbf{w}} = \{\mathbf{x} \in X | q_W(\mathbf{x}) = \mathbf{w}\}$$

L'ensemble des $\mathcal{C}^W = \{C^W_{\mathbf{w}}\}_{\mathbf{w}\in W}$ forme une partition de $\mathbf{X}$.

Le nombre de représentants étant inférieur au nombre de vecteurs à quantifier, nous avons en général une erreur de quantification entre un vecteur $\mathbf{x}$ et son représentant $q_W(\mathbf{x})$. Afin de mesurer la quantité d'information perdue, on utilise le critère de distortion.

Définition 2.3 (Distortion) *La distortion $D_W(E)$ de W sur un ensemble de vecteurs E est définie par :*

$$D_W(E) = \sum_{\mathbf{x}\in E} d(\mathbf{x}, q_W(\mathbf{x}))^2$$

Le log rapport d'erreur signal sur bruit (PSNR), une mesure de l'erreur basée sur la distortion, permet une lecture plus facile de cette quantité.

Définition 2.4 (PSNR) *Le PSNR de W sur un ensemble de vecteurs E est définie par :*

$$PSNR_W(E) = 10 \log_{10} \left(\frac{pnt^2}{D_W(E)} \right)$$

avec $n = |E|$ le nombre d'élément dans E, $t = \max_{\mathbf{x}\in E} ||\mathbf{x}||$ la plus grande norme dans E, et p la dimension des vecteurs de E.

Pour information, voici l'appréciation que l'on peut donner au PSNR pour une image :

PSNR	Qualité
0	nulle
10	Très mauvaise
20	Mauvaise
30	Bonne
40	Très bonne
$+\infty$	Parfaite

2.3.2 Quantification Optimale

Un ensemble de représentants $W^\star$ est optimal sur X lorsque, pour tout autre ensemble avec un même nombre de représentants, il minimise la distortion $D_W(X)$ pour tout W. Autrement dit :

$$W^\star = \underset{W}{\operatorname{argmin}}\, D_W(X)$$

La recherche de l'ensemble optimal peut se faire de manière itérative, en construisant une suite d'ensembles $(W_i)_i$ telle que :

$$\forall i \; D_{W_{i+1}}(X) < D_{W_i}(X)$$

L'ensemble résultat W est la limite de cette suite :

$$W = \lim_{i \to +\infty} W_i$$

L'algorithme 2.1 décrit ce procédé, les fonctions "initialiser" et "optimiser" étant propres à la méthode utilisée.

TAB. 2.1: Quantification

fonction $W, D = \text{quantifier}(X, m)$
Entrées : X *Ensemble de n vecteurs de* $\mathbb{R}^p$ *à quantifier* m *Nombre de représentants* Sorties : W *Ensemble des m représentants* D *Distortion*
$W = \text{initialiser}()$ $\mathbf{n}, \mathbf{d}, \mathbf{c} = \text{calculerclasses}(X, W)$ $D = \sum_{k=1}^{m} d_k$ **faire** $D1 = D$ $W = \text{optimiser}(X, W, \mathbf{n}, \mathbf{d}, \mathbf{c})$ $\mathbf{n}, \mathbf{d}, \mathbf{c} = \text{calculerclasses}(X, W)$ $D = \sum_{k=1}^{m} d_k$ **tantque** $\frac{\lvert D1 - D \rvert}{D} < \epsilon$

TAB. 2.2: Quantification (suite)

fonction $\mathbf{n}, \mathbf{d}, \mathbf{c} = \text{calculerclasses}(X, W)$

Entrées :
- X *Ensemble de n vecteurs de* $\mathbb{R}^p$ *à quantifier*
- W *Ensemble de m vecteurs de* $\mathbb{R}^p$

Sorties :
- n_k *Nombre de vecteurs dans la classe k*
- d_c *Distortion sur la classe k*
- c_i *Classe du vecteur* $\mathbf{x}_i$

$\mathbf{n} = 0; \mathbf{d} = 0$
pour $\mathbf{x}_i$ **dans** X
  $k = \underset{\mathbf{w}_j \in W}{\text{argmin}}\ d(\mathbf{x}_i, \mathbf{w}_j)$
  $n_k = n_k + 1$
  $d_k = d_k + d(\mathbf{x}_i, \mathbf{w}_k)^2$
  $c_i = k$
finpour

2.3.3 Paramètres

Nous avons vu qu'un quantificateur dépend de trois paramètres : p la dimension des vecteurs à quantifier, n le nombre de vecteurs à quantifier, et m le nombre de représentants à déterminer.

La dimension p des vecteurs d'entrée est un paramètre important et de nombreux problèmes surviennent généralement lorsque p augmente. Beaucoup de méthodes ne sont pas capables de faire face à ce genre de problèmes, et doivent être utilisées pour de faibles dimensions d'entrée.

Le nombre n de vecteurs d'entrée pose de sérieux problèmes lorsqu'il dépasse le milliard d'individus. C'est justement le cas en indexation puisqu'il nous faut quantifier plusieurs dizaines voir centaines de milliers d'images, chacune composée de millions de vecteurs. Nous détaillerons plus tard les solutions que nous proposons.

Enfin le nombre m de représentants à déterminer est un facteur important sur la distortion finale, puisque plus il est élevé plus la distortion sera faible, et ce de manière non triviale. Plusieurs travaux ont déjà été effectués à ce sujet (Patanè, 2001; Saux, 2003). Quelle que soit la méthode choisie pour déterminer le nombre m automatiquement, cela revient généralement à remplacer m par un autre paramètre, comme la distortion globale souhaitée. Dans le cadre de la quantification des attributs d'une base d'images, seul l'ordre de grandeur de m importe, le surcoût apporté par une détermination précise ne justifie pas son utilisation (Fournier, 2002). C'est la raison pour laquelle nous nous intéressons à la quantification sur un nombre m fixe de représentants.

2.4 Méthodes de Quantification

2.4.1 Nuées Dynamiques

Les Nuées Dynamiques est l'une des premières méthode de quantification vectorielle. Cet algorithme est aussi connue sous le nom K-Means ou LBG (Linde et al., 1980).

2.4.1.1 Version classique

La mise à jour du quantificateur est tout d'abord effectuée en calculant la partition $\mathcal{C}^W$ (foncion 'calculerclasses'). Puis, pour chaque classe $C^W_{\mathbf{w}}$ de $\mathcal{C}^W$, le représentant $\mathbf{w}$ est remplacé par le centre de gravité de $C^W_{\mathbf{w}}$:

$$\mathbf{w} \leftarrow \frac{1}{|C^W_{\mathbf{w}}|} \sum_{\mathbf{x}_i \in C^W_{\mathbf{w}}} \mathbf{x}_i$$

TAB. 2.3: Nuées Dynamiques

fonction W = initialiser(X)

W = ensemble de m vecteurs aléatoires

fonction W = optimiser($X, W, \mathbf{n}, \mathbf{d}, \mathbf{c}$)

W = nouveauxcentres($X, W, \mathbf{n}, \mathbf{c}$)

fonction W = nouveauxcentres($X, W, \mathbf{n}, \mathbf{c}$)

W = ensemble de m vecteurs nuls
pour $\mathbf{x}_i$ **dans** X
  $k = c_i$
  $\mathbf{w}_k = \mathbf{w}_k + \mathbf{x}_i$
finpour
pour $\mathbf{w}_k$ **dans** W
  $\mathbf{w}_k = \mathbf{w}_k / n_k$
finpour

2.4.1.2 Version avec découpage

La version classique nécessite une initialisation aléatoire de l'ensemble W des représentants. Cette initialisation rend le résultat final particulièrement instable, et parfois même certains représentants ne représentent aucun vecteur. Cela s'explique par le fait que certaines valeurs initiales, trop éloignées des principales concentrations ne représentent que peu de données.

Il existe une initialisation par découpage (Linde et al., 1980). A chaque étape de mise à jour du quantificateur, chaque représentant est divisé en deux autres représentants fruits de la quantification binaire de la classe à partager, et ce jusqu'à ce que le nombre voulu de centres soit obtenu. L'Algorithme 2.4 décrit ce processus pour le cas où $m = 2^k$. Pour les autres cas où $m = 2^k + t$, il suffit d'itérer ce processus tant que $r < 2^k$, puis de découper uniquement t classes lorsque $r = 2^k$.

TAB. 2.4: Nuées Dynamiques avec découpage

fonction $W = \text{initialiser}(X)$

$W =$ ensemble de m vecteurs aléatoires
$r = 1$

fonction $W = \text{optimiser}(X, W, \mathbf{n}, \mathbf{d}, \mathbf{c})$

$W = \text{nouveauxcentres}(X, W, \mathbf{n}, \mathbf{c})$
$W, r = \text{decouper}(X, W, r, \mathbf{c})$

fonction $W, r = \text{decouper}(X, W, r, \mathbf{c})$

pour j **dans** $[1, r]$
  $\hat{X} = \{\mathbf{x}_i \mid c_i = j\}$
  $\hat{W} = \text{quantifier}(\hat{X}, 2)$
  $\mathbf{w}_{2j} = \hat{\mathbf{w}}_1$
  $\mathbf{w}_{2j+1} = \hat{\mathbf{w}}_2$
finpour
$r = 2r$

2.4.2 Nuées Dynamiques Adaptatives

La méthode des Nuées Dynamiques Adaptatives entre dans le cadre des algorithmes d'apprentissage par compétition. Ces méthodes ne travaillent pas directement sur l'ensemble des données d'apprentissage, mais convergent par l'*injection* successive de chaque donnée, et ce dans un ordre de préférence le plus aléatoire possible. C'est la raison pour laquelle elles sont dites *adaptatives*. Cet algorithme est aussi appelée K-Means Adaptatif ou Centres Mobiles Adaptatifs.

2.4.2.1 Version classique

Cette méthode permet l'amélioration de l'ensemble des représentants W à l'aide de chaque nouveau vecteur $\mathbf{x}$ de l'ensemble X. Pour cet $\mathbf{x}$ la méthode détermine le représentant $\mathbf{w}_j$ le plus proche au sens de la distance d. Ce représentant (dit *gagnant*) est rapproché du vecteur $\mathbf{x}$ plus moins en fonction d'un poids ω_j. Ce poids est initialisé à zéro, puis incrémenté à chaque mise à jour du repésentant associé. Au fur et à mesure

que le poids d'un représentant augmente, la mise à jour de ce représentant est de moins en moins importante.

Une itération d'optimisation d'un ensemble W se fait par l'injection dans un ordre aléatoire des vecteurs de X. L'Algorithme 2.5 décrit ce processus.

TAB. 2.5: Nuées Dynamiques Adaptatives

fonction $W = \text{initialiser}(X)$

$W =$ ensemble de m vecteurs aléatoires
$\boldsymbol{\omega} = 0$

fonction $W = \text{optimiser}(X, W, \mathbf{n}, \mathbf{d}, \mathbf{c})$

pour $\mathbf{x}_i$ **dans** X **pris aléatoirement**
  $W = \text{injecter}(W, \mathbf{x}_i)$
finpour

fonction $W = \text{injecter}(W, \mathbf{x})$

$c = \underset{\mathbf{w}_j \in W}{\text{argmin}}\ d(\mathbf{x}, \mathbf{w}_j)$
$\mathbf{w}_c = \mathbf{w}_c + \frac{1}{1+\omega_c}(\mathbf{x} - \mathbf{w}_c)$
$\omega_c = \omega_c + 1$

2.4.2.2 Version avec découpage

Au même titre que les nuées dynamiques, l'initialisation pose problème. La solution classique de l'initialisation aléatoire résulte aussi d'une faible stabilité.

La version avec découpage est une solution pour se passer d'initialisation. Le principe est le suivant :

- On démarre avec une seule classe dont le représentant dont la valeur n'a pas d'importance. En effet, au démarrage le poids vaut zéro, et le représentant prendra la valeur du premier vecteur injecté.
- Ce représentant est mis à jour au fur et à mesure que des vecteurs sont injectés, jusqu'à ce que son poids ait atteint un certain seuil s. C'est à ce moment que le découpage apparaît : on ajoute une nouvelle classe dont le représentant est le même que que le premier, puis on répartit le poids accumulé entre les deux représentants. A ce stade de l'algorithme, W est composé de deux représentants identiques, de poids $\frac{s}{2}$. Le fait que W possède des représentatns identiques ne pose aucun problème, puisque, lors d'une injection, un et un seul représentant est gagnant.
- L'algorithme se poursuit dans le même esprit : dès que le poids d'un représentant est supérieur à s et que le nombre de représentants total est inférieur à celui voulu, on effectue le découpage.

Le seuil s de découpage doit être suffisamment grand pour que le découpage ait un

impact sur l'initialisation, et suffisamment petit pour que le nombre de représentant voulu soit atteint. Dans cette optique nous avons choisi $s = \frac{n}{2m}$.

L'Algorithme 2.6 décrit ce processus.

TAB. 2.6: Nuées Dynamiques Adaptatives avec découpage

fonction $W = \text{initialiser}(X)$

$W =$ ensemble de m vecteurs aléatoires
$\boldsymbol{\omega} = 0$
$r = 1$

fonction $W = \text{optimiser}(X, W, \mathbf{n}, \mathbf{d}, \mathbf{c})$

pour $\mathbf{x}_i$ **dans** X **pris aléatoirement**
  $W = \text{injecter}(W, \mathbf{x}_i)$
finpour

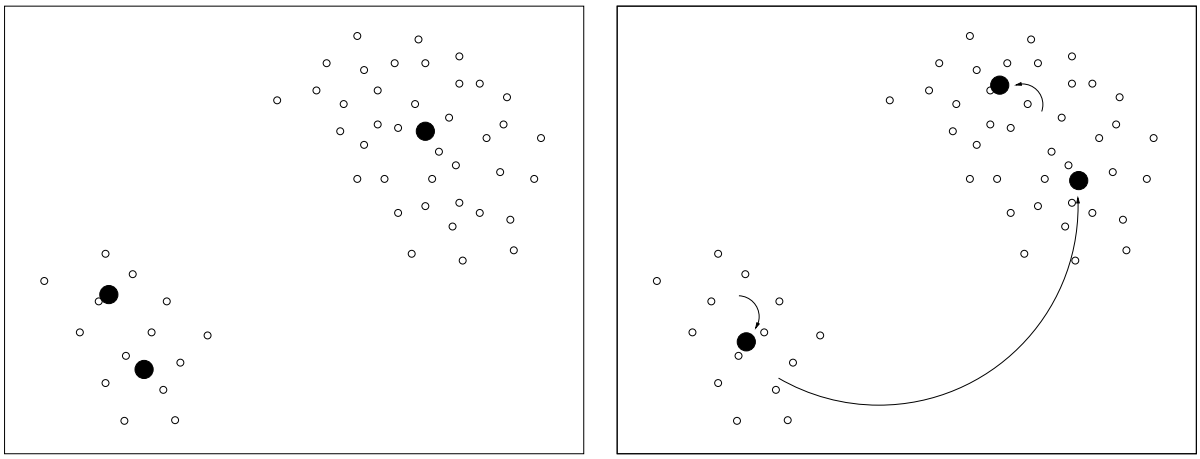
fonction $W = \text{injecter}(W, \mathbf{x})$

$c = \underset{\mathbf{w}_j \in W}{\text{argmin}}\ d(\mathbf{x}, \mathbf{w}_j)$
$\mathbf{w}_c = \mathbf{w}_c + \frac{1}{1+\omega_c}(\mathbf{x} - \mathbf{w}_c)$
$\omega_c = \omega_c + 1$
si $\omega_c == s$ **et** $r < m$ **alors**
  $\omega_c = \frac{s}{2}$
  $\omega_r = \frac{s}{2}$
  $\mathbf{w}_r = \mathbf{w}_c$
  $r = r + 1$
finsi

2.4.3 Nuées Dynamiques Améliorées

Les deux méthodes présentées précédemment ont un défaut commun : la distortion finale dépend beaucoup de l'intialisation. Bien que les initialisations par découpage améliorent ces méthodes, le problème n'est cependant pas résolu, et certains représentants sont souvent sous-exploités. Une approche pour tenter de rendre une méthode la plus indépendante possible de l'initialisation est d'introduire une notion d'utilité. Cette notion permet alors de juger si on est proche de la solution optimale, et dans le cas contraire, introduire une modification dans l'ensemble de ces représentant.

Plusieurs notions d'utilité ont été introduites (Fritzke, 1997; Patanè, 2001). Nous nous sommes intéressé à celle de (Patanè, 2001), qui repose sur un théroème de (Gersho, 1979) qui stipule que la distortion de chaque classes d'un ensemble optimal de représentant est la même. L'algorithme associée est aussi proposée par (Patanè, 2001), dont nous ne faisons qu'une description rapide, notre but étant la justification de son utilisation.



(a) (b)

FIG. 2.1 – Principe de l'utilité : les représentant dans le cas (a) ont une plus faible utilité que dans le cas (b).

2.4.3.1 Principe

Cette méthode est la même que celle des nuées dynamiques, à la différence qu'elle ajoute à chaque itération une remise en cause du quantificateur. L'idée est de répartir intelligemment les vecteurs X dans les différentes classes de $\mathcal{C}^W$.

Prenons l'exemple de la Figure 2.1(a), avec $p = 2$ et $m = 3$. Comme nous pouvons le constater, le nuage de points à gauche est représenté par deux vecteurs, et celui de droite, plus important que l'autre, ne dispose que d'un seul représentant. De toute évidence, il est plus judicieux de n'avoir qu'un représentant pour l'ensemble de gauche et de deux pour celui de droite, comme décrit dans la Figure 2.1(b).

Malheureusement, les nuées classiques ne sortiront pas de cette configuration.

On introduit l'*utilité* $u_{\mathbf{w}}$ d'une classe $C^W_{\mathbf{w}}$:

$$u_{\mathbf{w}} = \frac{D_W(C^W_{\mathbf{w}})}{\frac{1}{m} D_W(X)}$$

Ces valeurs permettent de mesurer la bonne répartition des vecteurs $\mathbf{x}$ dans les différentes classes de $\mathcal{C}^W$. L'idéal étant d'avoir une utilité de 1 pour chaque classe.

Une heuristique permet de choisir à partir des utilités deux représentants à optimiser. L'heuristique peut être par exemple de choisir le représentant de plus faible utilité et celui de plus forte utilité.

Supposons que nous ayons choisi $C^W_{\mathbf{w}_i}$ et $C^W_{\mathbf{w}_p}$ avec $u_{\mathbf{w}_i} < 1$ et $u_{\mathbf{w}_p} > 1$ (Figure 2.2).

La première étape de l'échange va consister à quantifier les vecteurs de $C^W_{\mathbf{w}_p}$ sur deux repésentants $\mathbf{w}'_i$ et $\mathbf{w}'_p$ avec une Nuée Dynamique. Puis il faut réaffecter aux vecteurs de

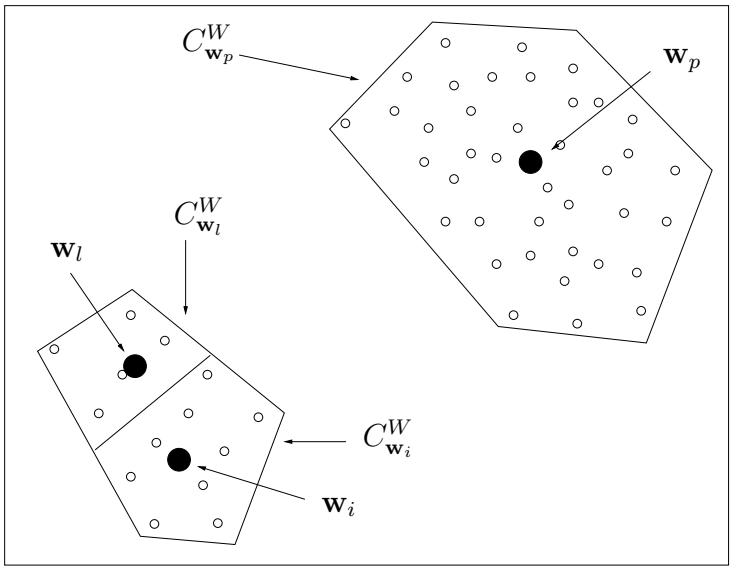


FIG. 2.2 – Algorithme ELBG : état avant l'échange

$C^W_{\mathbf{w}_i}$ un représentant. Pour cela on cherche le représentant $\mathbf{w}_l$ le plus proche de $\mathbf{w}_i$, et on calcule le centre de gravite $\mathbf{w}'_l$ de $C^W_{\mathbf{w}_i} \cup C^W_{\mathbf{w}_l}$. On peut alors former un nouvel ensemble W' de représentants en remplacant les représentants $\mathbf{w}_i$, $\mathbf{w}_p$ et $\mathbf{w}_l$ par les représentants $\mathbf{w}'_i$, $\mathbf{w}'_p$ et $\mathbf{w}'_l$ (Figure 2.3).

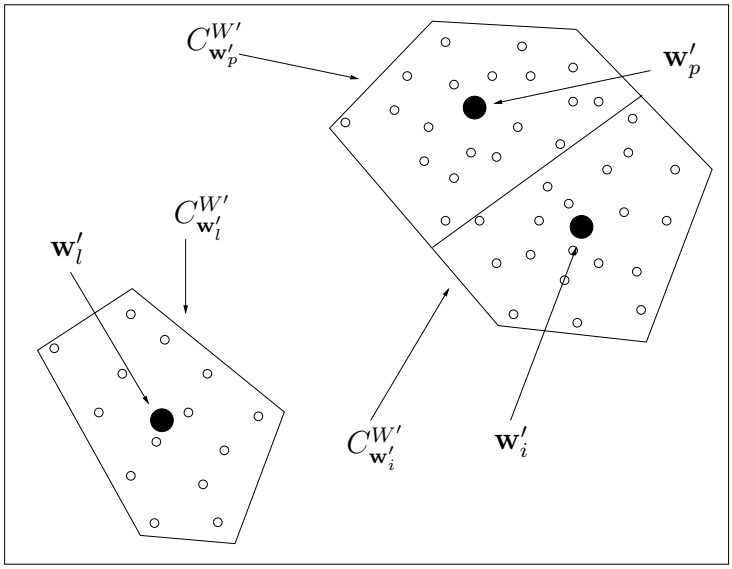


FIG. 2.3 – Algorithme ELBG : état après l'échange

Enfin il reste à vérifier que l'échange est intéressant. L'idéal pour cela serait de recalculer la distortion globale. Cependant le coût en calculs est très important. Patanè propose à la place de faire une estimation de cette distortion, en supposant que les distorsions des classes non concernées par l'échange restent inchangées. Il suffit alors de calculer les distortions $D^{W'}(C^{W'}_{\mathbf{w}'_i})$, $D^{W'}(C^{W'}_{\mathbf{w}'_i})$ et $D^{W'}(C^{W'}_{\mathbf{w}'_i})$ des nouvelles classes potentielles, et de

comparer leur somme à celle des classes i, j et p précédentes. Si la distortion des nouvelles classes est plus faible, l'échange est donc fructueux. Dans le cas contraire aucun changement n'est éffectué et on passe au couple suivant choisi par l'heuristique.

Cette étape d'échange est répétée autant de fois que nécessaire, puis on recalcule les centres comme pour les Nuées Dynamiques. L'algorithme 2.7 décrit cette méthode.

2.4.3.2 Version avec découpage

Puisque la méthode est basée sur les nuées dynamiques, il est aussi possible d'effectuer un découpage. Contrairement aux nuées classiques, les performances de la méthode améliorée avec découpage sont les mêmes qu'avec une initialisation aléatoire.

TAB. 2.7: Nuées Dynamiques Améliorées

fonction $W = \text{optimiser}(X, W, \mathbf{n}, \mathbf{d}, \mathbf{c})$

$W = \text{nouveauxcentres}(X, W, \mathbf{n}, \mathbf{c})$
$W = \text{ameliorer}(X, W, \mathbf{n}, \mathbf{d}, \mathbf{c})$

fonction $W = \text{ameliorer}(X, W, \mathbf{n}, \mathbf{d}, \mathbf{c})$

$\forall k \in [1, m]\ f_k = 0$ – *Marque si la classe k a été fusionnée*
$\forall k \in [1, m]\ s_k = 0$ – *Marque si la classe k a été scindée*
pour i **dans** $[1, m]$
  si $d_i < D/m$ **et** $s_i = 0$ **et** $f_i = 0$ **alors**
    $l = \text{trouverl}(i)$
    si $l \neq -1$ **alors**
      $p = \text{trouverp}(l)$
      si $p \neq -1$ **alors**
        $\text{echanger}(l, i, p)$
      finsi
    finsi
  finsi
finpour

fonction $l = \text{touverl}(i)$

$l = \underset{j \neq i}{\text{argmin}}\ d(\mathbf{w}_i, \mathbf{w}_j)$ – *Centre w_l le plus proche de w_i*
si $s_l = 0$ **ou** $f_l = 0$ **alors**
  $l = -1$ – *Invalide si déjà fusionné ou scindé*
finsi

fonction $p = \text{touverp}(l)$

$s = 0$
...

```
...
      – Le processus suivant permet de tirer aléatoirement une
classe p ; le tirage est pondéré par la distortion de chaque classe
éligible (utilité sup. à 1 et non fusionnée précédemment)
  pour p dans [1, m]
    si p ≠ l et d_p > D/m et f_p = 0 alors
      s = s + d_p
    finsi
  finpour
  si s ≠ 0 alors
    r = nombre aléatoire entre 0 et s
    s = 0
    pour p dans [1, m]
      si p ≠ l et d_p > D/m et f_p = 0 alors
        s = s + d_p
        si s ≥ r alors
          arreter
        finsi
      finsi
    finpour
  sinon
    p = −1
  finsi
```

fonction echanger(i, l, p)

```
  n'_l = 0
  w'_l = 0        – On calcule le centre de la fusion des classes i et l
  pour k dans [1, n]
    si c_k = i ou c_k = l alors
      w'_l = w'_l + x_k
      n'_l = n'_l + 1
    finsi
  finpour
  w'l = w'_l / n'_l
  d'_l = 0        – On calcule la distortion de la classe l'
  pour k dans [1, n]
    si c_k = i ou c_k = l alors
      d'_l = d'_l + d(x_k, w'_l)^2
    finsi
  finpour

      – On quantifie la classe p sur deux centres
  {w'_p, w'_i} = quantifier(C_{w_p}, 2)
      – On calcule la distortion des classes p' et i'
  d'_i = 0; d'_p = 0
...
```

```
...
  pour k dans [1, n]
    si c_k = p alors
      si d(x_k, w'_p) < d(x_k, w'_i) alors
        d'_p = d'_p + d(x_k, w'_p)^2
      sinon
        d'_i = d'_i + d(x_k, w'_i)^2
      finsi
    finsi
  finpour
     – L'échange est-il intéressant ?
  si d'_l + d'_i + d'_p < d_l + d_i + d_p alors
    w_l = w'_l; d_l = d'_l; f_l = 1
    w_p = w'_p; d_p = d'_p; s_p = 1
    w_i = w'_i; d_i = d'_i; s_i = 1
    pour k dans [1, n]
      si c_k = i ou c_k = l alors
        c_k = l
      sinon si c_k = p alors
        si d(x_k, w'_p) < d(x_k, w'_i) alors
          c_k = p
        sinon
          c_k = i
        finsi
      finsi
    finpour
  finsi
```

2.4.4 Réduction du temps de calcul

Le calcul des classes $C_{\mathbf{w}}^{W}$ est particulièrement coûteux. En effet, le calcul direct décrit dans l'Algorithme 2.1 nécessite $O(nmp)$ opérations. Ce problème est un sous-problème de la recherche rapide des plus proches voisins, qui constitue un domaine de recherche à part entière. De nombreuses méthodes sont proposées dans la littérature scientifique, dont un bon nombre sont présentées dans le mémoire de master de Chua (Chua, 2000). Etant donnée l'ampleur du problème et sa distance au sujet ce cette thèse, nous nous contentons de présenter une version simple appelée *TIEC-1* (Critère d'élimination basée sur l'inégalité triangulaire à l'ordre 1). En dépit de sa simplicité, cette méthode apporte néanmoins un gain non négligeable en temps de calculs.

Cette méthode repose sur une estimation $\hat{\mathbf{w}}$ du représentant le plus proche d'un vecteur $\mathbf{x}$, et limite la recherche dans un voisinage autour de $\hat{\mathbf{w}}$.

Soit $V_{\hat{\mathbf{w}},\mathbf{x}} \subset W$ l'ensemble des vecteurs de W dont la distance à $\hat{\mathbf{w}}$ est supérieure à $2d(\mathbf{x}, \hat{\mathbf{w}})$:

$$V_{\hat{\mathbf{w}},\mathbf{x}} = \{\mathbf{w} \in W | 2d(\mathbf{x}, \hat{\mathbf{w}}) \leq d(\hat{\mathbf{w}}, \mathbf{w})\}$$

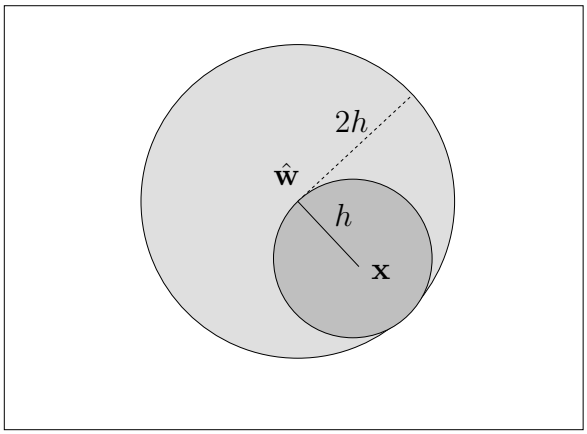


FIG. 2.4 – Principe du TIEC-1.

Il s'ensuit que, si $\mathbf{w} \in V_{\hat{\mathbf{w}},\mathbf{x}}$, alors

$$\begin{array}{rcl} 2d(\hat{\mathbf{w}},\mathbf{x}) & \leq & d(\hat{\mathbf{w}},\mathbf{w}) \\ 2d(\hat{\mathbf{w}},\mathbf{x}) & \leq & d(\hat{\mathbf{w}},\mathbf{x}) + d(\mathbf{x},\mathbf{w}) \\ d(\hat{\mathbf{w}},\mathbf{x}) & \leq & d(\mathbf{x},\mathbf{w}) \end{array}$$

Autrement dit, si un vecteur $\mathbf{w}$ appartient à $V_{\hat{\mathbf{w}},\mathbf{x}}$, alors il n'est pas un meilleur représentant que $\hat{\mathbf{w}}$. On peut donc éliminer sans erreur les représentants de $V_{\hat{\mathbf{w}},\mathbf{x}}$.

Dans le cas où le nouvel ensemble W est proche du précédent, cette méthode a une complexité de l'ordre de $O(np)$. C'est justement le cas pour toutes les itérations d'optimisations d'un quantifieur, mise à part la première. L'Algorithme 2.8 décrit ce processus.

TAB. 2.8: Calcul des classes par TIEC-1

fonction $\mathbf{n}, \mathbf{d}, \mathbf{c} =$ calculerclasses$(X, W, \mathbf{c})$

$A =$ matrice nulle de dimension $m \times m$
pour i, j **dans** $[1, m]$
  $A_{ij} = d(\mathbf{w}_i, \mathbf{w}_j)$
finpour
$\mathbf{n} = 0; \mathbf{d} = 0;$
pour i **dans** $[1, n]$
  $k = c_i$
  $h = d(\mathbf{w}_k, \mathbf{x}_i)$
  $k = \underset{j|A_{jk}<h}{\operatorname{argmin}}\ d(\mathbf{x}_i, \mathbf{w}_j)$
  $n_k = n_k + 1$
  $d_k = d_k + d(\mathbf{x}_i, \mathbf{w}_k)^2$
  $c_i = k$
finpour

2.4.5 Comparaison Expérimentale

Afin d'évaluer les performances de chaque méthode, nous avons calculé la distortion et le temps de la quantification en 256 couleurs RVB de chacune des 500 images de la base ANN (*cf.* annexe A).

Les résultats sont présenté en PSNR dans le tableau suivant :

Méthode	PSNR(dB)	Temps(sec)
N.D. Adaptatives	34.36 ± 1.35	13.24 ± 2.63
N.D. Adaptatives avec découpage	37.56 ± 2.89	14.76 ± 2.02
N.D.	37.87 ± 2.76	12.35 ± 1.55
N.D. avec découpage	37.90 ± 2.57	31.99 ± 11.03
N.D. Améliorées	38.69 ± 2.82	8.49 ± 1.27

Les Nuées Dynamiques Adaptatives donnent les plus mauvais résultats. La version avec découpage améliore nettement ceux-ci, confirmant que l'utilisation de Nuées Dynamiques Adaptatives efficaces ne serait se faire sans ce type d'initialisation. Sur le plan purement calculatoire, les temps de calculs sont parmi les plus élevés. Le contexte actuel d'utilisation de cette méthode ne lui est pas favorable sur ce plan. En effet, étant donné que les représentants sont modifiés lors de chaque injection, il n'est pas possible d'utiliser une optimisation comme le TIEC-1.

Les Nuées Dynamiques proposent des performances un peu meilleures que la version adaptative. Cependant, l'utilisation d'un découpage ne semble pas pertinent pour cette méthode, surtout sur le plan calculatoire. En effet, puisque lors de cette évaluation la quantification est sur 256 représentants, il est nécessaire d'effectuer au moins 8 itérations, alors que les autres méthodes peuvent parfois converger en 2 ou 3 itérations.

Les Nuées Dynamiques Améliorées sont inconstestablement la meilleure méthode, et ce à tous les niveaux. Cela s'explique par l'étape de recherche d'un optimum global, qui reduit le nombre d'itérations nécessaires à la convergence. Hormis une implantation difficile, cette méthode est actuellement un choix incontournable en matière de quantification.

2.5 Quantifier un grand nombre de données

La quantification vectorielle dans le cadre l'indexation diffère des problèmes classiques.

2.5.1 Contexte

Tout d'abord, nous souhaitons quantifier une quantité astronomique de vecteurs. Par exemple, pour le cas de l'attribut couleur, nous avons pour chaque image autant de vecteurs que de points (autour du million). Si la base comporte cent mille voir un million

d'images, il n'est pas consevable aujourd'hui d'injecter dans un quantificateur une telle quantité de données, et ce pour des raisons matérielles. Les 6000 images de la base COREL (*cf.* annexe A) utilisée pour les tests de performances constituent un ensemble de plus de 590 millions de vecteur RVB. Le simple chargement de cette base requiert près de 150 Go de mémoire RAM.

2.5.2 Méthode Adaptative

Fournier(Fournier, 2002) propose d'utiliser des nuées dynamiques adaptatives pour résoudre ce problème. En effet, puisque cette méthode peut mettre à jour le quantificateur vecteur par vecteur, le nombre total de vecteurs importe peu. Cependant, la méthode repose sur une hypothèse importante : il est nécessaire d'injecter les vecteurs dans un ordre totalement aléatoire. Malheureusement, ceci est matériellement non réalisable, puisque cela nécessite le décodage d'un fichier image pour chaque injection. L'approximation proposée par Fournier est d'injecter les vecteurs d'attributs par lot. Pour un fichier image décodé, on injecte aléatoirement une partie des vecteurs d'attribut de cette image. Mais cette méthode réduit les performances des nuées adaptatives déjà faibles. L'Algorithme 2.9 décrit cette méthode.

Un autre problème se pose : celui de l'ajout de nouvelles images dans la base. Etant donné la quantité de calculs, il n'est pas souhaitable d'avoir à re-effectuer tous les calculs sur toutes les images à chaque ajout. Une solution à l'aide des nuées adaptatives est de ne calculer les attributs que pour les nouvelles images, et d'injecter les nouveaux vecteurs dans le quantificateur précédent. Malheureusement, si les images contiennent des vecteurs attributs totalement nouveaux, ceci seront très mal représentés.

Tab. 2.9: Large Quantification : méthode adaptative

fonction $W = \text{quantifier}(B)$

Entrées :
  B *La base d'images*
Sorties :
  W *Les représentants*

$n =$ nombre d'images dans B
pour k **dans** $[1, 10n]$
  $X =$ image prise aléatoirement dans B
  $m =$ nombre de pixels dans X
  pour i **dans** $[1, m/10]$
    $\mathbf{x} =$ pixel pris aléatoirement dans X
    injecter($\mathbf{x}$)
  finpour
finpour

2.5.3 Méthode Proposée

La solution que nous proposons est de quantifier de manière indépendante les vecteurs attributs de chaque image, puis de quantifier l'ensemble des représentants de chaque image.

Ceci a deux avantages. D'une part, nous pouvons utiliser une méthode de quantification plus efficace. D'autre part, nous pouvons ajouter de nouvelles images à la base sans souffrir des inconvénients expliqués précedemment.

La méthode se décompose en deux étape : une quantification locale de chaque image, puis la quantification globale de tous les représentants obtenus.

Soit K le nombre d'images de la base, nous avons pour chaque image d'indice $k \in [1, K]$ un ensemble X_k de vecteurs attribut correspondants.

Après quantification de X_k sur m_k reprśentants nous obtenons un ensemble

$$W_k = \{\mathbf{w}_j^k | 1 \leq j \leq m_k\}$$

Nous réunissons ensuite les W_k dans un ensemble

$$X = \bigcup_{k=1}^{K} W_k$$

Il ne reste plus qu'à quantifier l'ensemble X sur m éléments. On obtient ainsi l'ensemble :

$$W = \{\mathbf{w}_j | 1 \leq j \leq m\}$$

La figure 2.5 décrit toutes ces étapes et l'Algorithme 2.10 décrit la méthode.

Cette première proposition accumule tous les réprésentants de chaque image sans prendre compte du nombre de vecteurs qu'ils représentent.

Afin de pallier ce problème, nous proposons d'utiliser le nombre z_j^k de vecteurs représentés par $\mathbf{w}_j$ pour l'image k :

$$z_j^k = |C_{\mathbf{w}_j}^{W_k}|$$

Autrement dit, la normalisation de $(z_j^k)_j$ est la distribution de W_k sur l'image k.

Nous obtenons ainsi pour chaque image un ensemble de représentants et leurs pondérations :

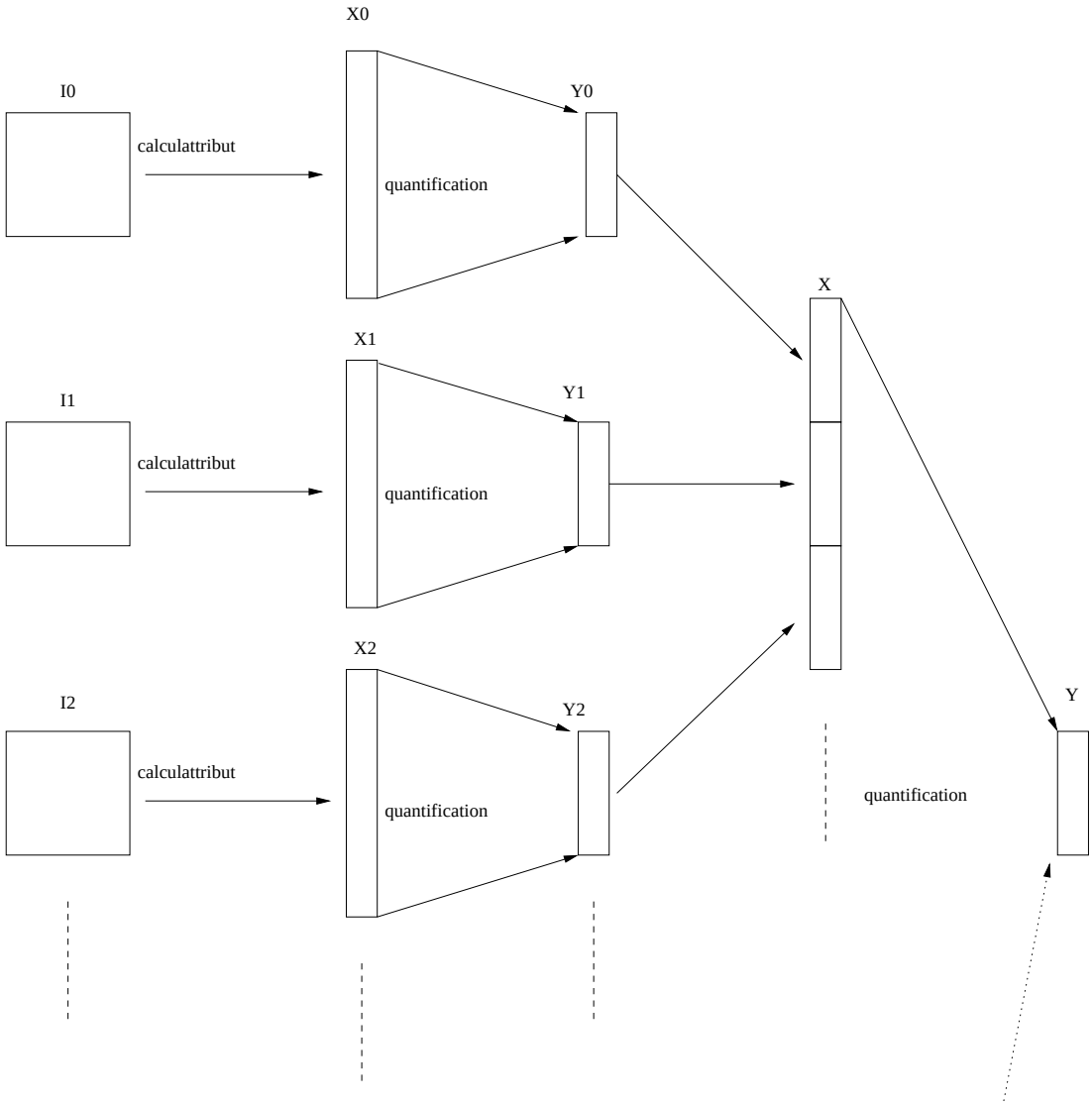


FIG. 2.5 – Quantification en 2 temps

$$\tilde{W}_k = \{(\mathbf{w}_j^k, z_j^k) | 1 \leq j \leq m_k\}$$

Nous les réunissons ensuite dans un ensemble :

$$\tilde{X} = \bigcup_{k=1}^{K} \tilde{W}_k$$

Il nous faut modifier quelque peu les Nuées Dynamiques afin de pouvoir prendre en compte les pondérations associées aux entrées. Il suffit pour cela de reproduire tous les calculs comme si nous avions certains vecteurs d'entrée répétés plusieurs fois.

Supposons que les vecteurs à quantificateur soient dans $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ de pondération associée $Z = \{z_1, ..., z_n\}$.

La distortion de X devient :

$$\tilde{D}(X) = \sum_{i=1}^{n} z_i d(\mathbf{x}, q(\mathbf{x}))^2$$

Le calcul du centre de gravité d'un ensemble devient pondéré :

$$\tilde{y}_j = \frac{1}{\sum_{i \in C_j} z_i} \sum_{i \in C_j} z_i \mathbf{x}_i$$

avec I_j l'ensemble des indices correspondant aux x_i de la classe C_j.

Hormis ces deux point les méthodes de quantification restent les mêmes.

TAB. 2.10: Large Quantification : méthode proposée

fonction $W = \text{quantifier}(B, m, t)$
Entrées : B *La base d'images* m *Nombre de représentants* t *Nombre de représentants par image* Sorties : W *Les représentants*
$n =$ nombre d'images dans B $\tilde{X} =$ ensemble vide **pour** k **dans** $[1, n]$ $X = k$ième image de B $\tilde{X} = \tilde{X} \cup \text{quantifier}(X, t)$ ...

```
...
  finpour
  W = quantifier(X̃, m)
```

2.5.4 Performances

Nous évaluons dans ce paragraphe les performances des signatures obtenues par les deux méthodes de large quantification. Contrairement aux expérimentations précédentes, nous nous intéressons ici à la qualité de classification que nous pouvons obtenir en utilisant les différentes signatures calculées. L'idée est l'évaluer les méthodes dans le cadre d'une utilisation pour la recherche de catégories.

Les expérimentations suivantes ont été effectuées sur la base COREL (*cf.* annexe A). La méthode de classification est les SVM (*cf.* chapitre 4), la fonction noyau est une gaussienne avec une distance du χ^2 (*cf.* chapitre 3). Les expérimentations suivantes évaluent les performances globales du système sur toutes les catégories de la base. Nous calculons 1000 classifications de la base. Pour chacune de ces classification, nous choisissons aléatoirement l'une des catégories, puis construisons aléatoirement un ensemble d'apprentissage de taille 100 pour la catégorie choisie. Suite à chaque classification de la base, nous déterminons la Précision Moyenne (*cf.* annexe A). Toutes ces valeurs sont ensuite moyennées sur les 1000 classifications.

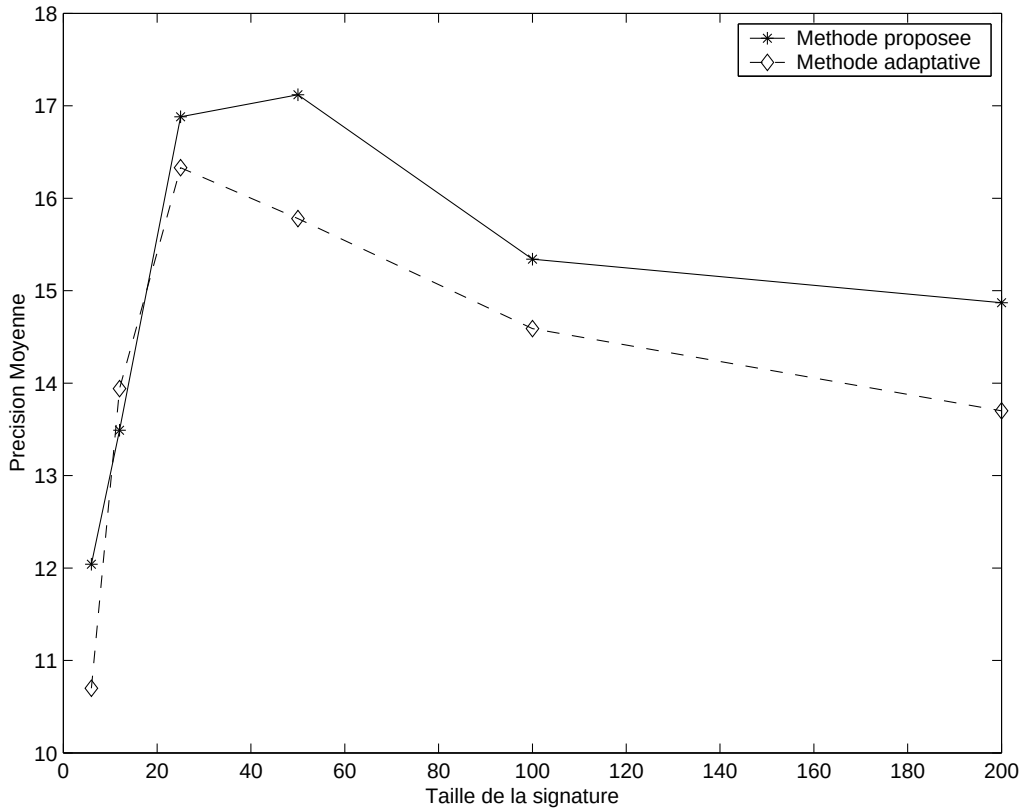
Pour la méthode adaptative, le calcul des centres est le suivant (avec n le nombre d'images dans la base) :

1. Initialisation des nuées dynamiques adaptatives pour m centres ;
2. Répété $10n$ fois :
   (a) Choix aléatoire d'une des images de la base ;
   (b) Calcul des vecteurs attributs pour l'image choisie ;
   (c) Injection de 1/10 des vecteurs attributs dans le quantifieur.

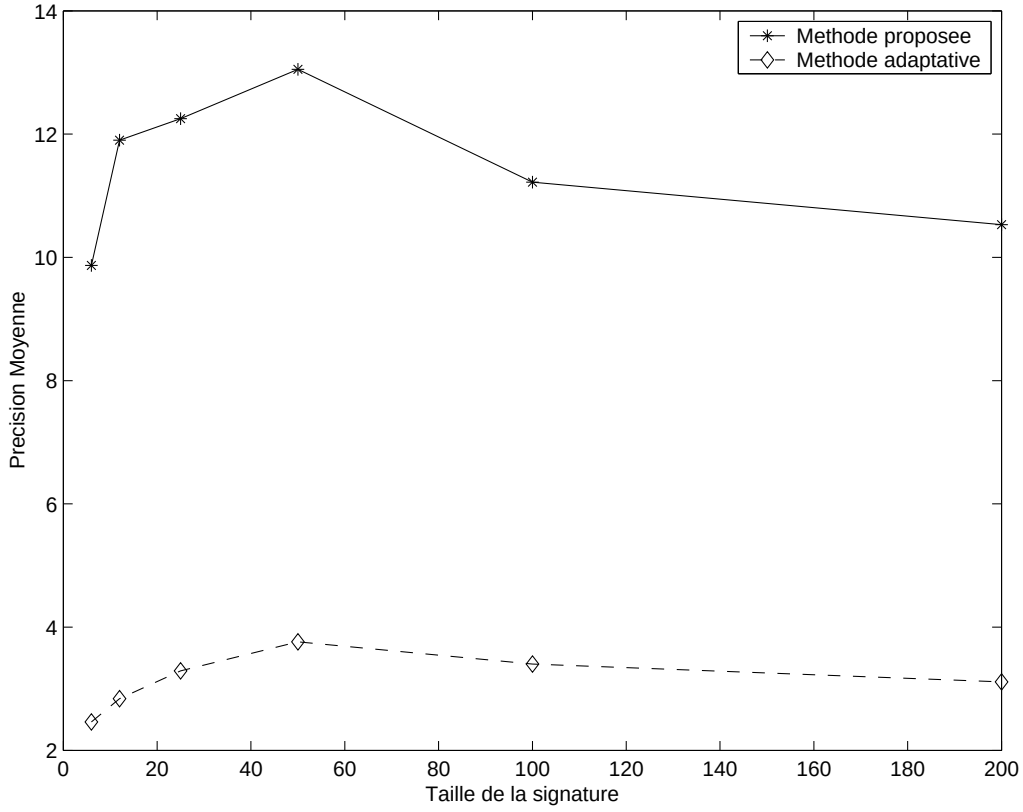
Pour la méthode proposée :

1. Pour chaque image de la base :
   (a) Calcul des vecteurs attributs ;
   (b) Quantification sur 256 centres de ces vecteurs par ELBG ;
   (c) Mémorisation des 256 centres.
2. Quantification sur m centres des $256 \times n$ centres mémorisés par ELBG.

Pour chaque méthode, nous calculons des signatures de différentes tailles m (6,12,25,50,100,200) pour l'attribut couleur $L^\star a^\star b^\star$ et pour l'attribut texture Filtres de Gabor. Notons que pour la méthode adaptative, tous les calculs doivent être refaits pour chaque taille de signature, alors que pour la méthode proposée, seule l'étape 2 doit être répétée.



(a) Pour les attributs couleurs ($L^\star a^\star b^\star$).



(b) Pour les attributs texture (Filtres de Gabor).

FIG. 2.6 – Précision Moyenne pour une classification SVM avec 100 exemples, en fonction de la dimension de la signature.

2.5.4.1 Temps de calcul

Les temps de calcul varient fortement en fonction de la méthode et de l'attribut. Pour l'attribut $L^\star a^\star b^\star$, qui est très rapide à calculer, le temps de calcul pour la méthode adaptative varie de 4 à 10 heures, en fonction du nombre de centres demandés. La méthode proposée a un temps de calcul assez comparable à la méthode adaptative dans ce cas. Pour l'attribut texture, qui est plus long à calculer, la différence entre les deux méthodes est assez importante. Pour la méthode adaptative, pour chaque taille de signature, il a fallu de 5 à 10 jours pour déterminer les centres. Au total, le calcul des centres pour toutes les tailles a pris environ un mois. Pour la méthode proposée, l'étape 1 identique pour toutes les tailles de signature prend environ une journée. Le temps de calcul de l'étape 2 ne prend que quelques minutes.

Une fois les centres calculés, pour chaque attribut et chaque taille de signature (6,12,25,50,100,200), nous avons calculé les signatures par histograme. Pour chaque image de la base, nous formons l'histogramme en comptant le nombre de vecteurs attributs dans chaque classe. Pour la méthode adaptative, il a fallu calculer de nouveau les attributs pour cette étape, soit environ une semaine de calculs supplémentaires.

2.5.4.2 Performances

Nous évaluons ensuite ces représentations de la base par une classification SVM avec 100 exemples, avec pour critère la Précision Moyenne. Les résultats sont présentés en figure 2.6. Concernant l'attribut $L^\star a^\star b^\star$, les deux méthodes ont des performances du même ordre, avec un léger avantage pour la méthode proposée. Nous pouvons remarquer que les deux méthodes ont un maximum global. La méthode adaptative est maximale pour 25 centres, et la méthode proposée pour 50 centres. Concernant l'attribut texture, la méthode proposée a des performances nettement supérieures à la méthode adaptative. Ceci est peut être dû au fait que la dimension des vecteurs attributs est plus grande (12 au lieu de 3), et que la méthode proposée résiste mieux aux problèmes survenant aux grandes dimensions. Nous retrouvons aussi un maximum global, qui est à 50 centres pour les deux méthodes.

Notons par ailleurs que la présence d'un maximum global dans tous les cas est un résultat intéressant. En effet, cela suggère l'existence d'un optimum pour la taille des signatures. De plus, cet aspect n'est pas étranger à un des outils de la théorie de l'apprentissage statistique, la dimension VC, que nous présentons au chapitre 3.

2.6 Conclusion

Dans ce chapitre, nous avons présenté les signatures par histogramme. Nous avons choisi une approche de calcul de ces signatures par quantification vectorielle. Nous avons mis en évidence les difficultés d'une telle opération dans notre contexte où le nombre des vecteurs à quantifier est astronomique. La méthode que nous proposons pour réaliser cette quantification a démontré son efficacité en terme de performance et de temps de calculs

lors d'expérimentations. Nous avons aussi mis en évidence le fait qu'il existe un optimum pour la taille des signatures sur la base de test utilisée.

Cette étude nous a permis de régler les dimensions des attributs pour les différentes évaluations que nous faisons dans la suite du mémoire. Compte tenu des résultats obtenus, nous avons choisi de mélanger les attributs couleurs et textures, en construisant des signatures avec 25 valeurs relatives à la couleur, et 25 valeurs relatives à la texture. Ce paramètrage, proche de l'optimal à la vue des résultats expérimentaux, nous permet d'avoir des signatures plus compactes et néanmoins efficaces.

Chapitre 3

Similarité et Fonctions Noyaux

Suite aux processus de calcul des signatures des images, il est nécessaire de disposer d'une notion de *similarité* afin de pouvoir comparer les images entre elles. Le choix de la similarité est très important, car elle est à la base de tout système de recherche.

Dans ce chapitre, nous présentons différentes approches de similarité qui sont proposées dans la littérature, ainsi que les outils qui permettent de les évaluer. Parmi toutes ces techniques, nous avons choisi de nous intéresser tout particulièrement à une similarité par fonction noyau, qui offre de nombreux avantages pour l'analyse et l'exploitation des signatures dans le cadre de l'apprentissage.

3.1 Similarité

3.1.1 Mesure de similarité

Définir une similarité revient à déterminer une mesure $s(i, j)$ entre deux images i et j en fonction des signatures. La similarité est par conséquent très dépendante de la nature des signatures.

Une mesure simple de similarité lorsque l'on dispose de signatures vectorielles est d'utiliser une distance géométrique. Les distances les plus courantes sont celles de Minkowski :

$$d(\mathbf{x}_i, \mathbf{x}_j) = \Big(\sum_{r=1}^{p} (x_{ri} - x_{rj})^p \Big)^{\frac{1}{p}}$$

Entre autres, nous avons la distance L_1 avec $p = 1$ et la distance L_2 (ou Euclidienne) avec $p = 2$.

L'un des types de signature les plus utilisés en indexation d'images est l'histogramme. A un facteur de normalisation près, l'histogramme constitue une approximation de la

densité associée à l'image, vue comme une variable aléatoire. Bien que ces signatures soient vectorielles, les distances géométriques ne sont pas nécessairement les plus intéressantes pour les histogrammes. Par conséquent, on trouve de nombreuses techniques de calcul de similarités dédiées aux histogrammes.

L'intersection d'histogrammes est une des techniques les plus anciennces, et est utilisée pour la couleur (Swain & Ballard, 1991). Notons que cette mesure est équivalente à une distance L_1 si les histogrammes sont normalisés. On peut aussi voir l'histogramme comme une distribution, et ainsi utiliser l'entropie comme similarité (Kullback, 1959). Une version symétrique de mesure par l'entropie est aussi proposée (Puzicha et al., 1997), et ainsi qu'une mesure probabiliste au sens du χ^2 (Sethi & Patel, 1995).

Ces techniques comparent indépendamment les valeurs des histogrammes. Ceci pose des problèmes en terme de robustesse, puisqu'elles sont sensibles à certaines variations sur l'image, par exemple, lorsqu'un histogramme couleur subit une translation résultant d'une modification des conditions de luminosité. Des mesures plus robustes sont proposées, comme les distances sur distributions cumulées (Stricker & Orengo, 1995), l'Earth Mother Distance (Rubner, 1999), ou les distances quadratiques généralisées ($d(\mathbf{x}_i, \mathbf{x}_i) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)}$).

Lorsque la signature n'est plus un simple vecteur mais un ensemble de vecteurs se rapportant aux différents espaces de caractéristiques (couleur, texture, forme, etc.), le problème de la fusion d'informations issues de modèles distincts se pose alors. Une technique courante est de concaténer les signatures, puis d'uniformiser les échelles avec une distance de Mahalanobis (Mahalanobis, 1930). On peut aussi traiter chaque espace de caractéristiques de manière indépendante, en utilisant un modèle hiérarchique (Rui & Huang, 2000). La fusion est classiquement réalisée par une simple combinaison linéaire (Heinrichs et al., 2000). Enfin, lorsque les vecteurs d'une signature se refèrent à des points d'intérêt (Mikolajczyk et al., 2004), on peut utiliser un système de vote en fonction des mises en correspondance (*matching*) (Schmid & Mohr, 1995; Gros, 1998).

3.1.2 Compromis généralisation/mémorisation

Dans le cadre généraliste, il est difficile de faire un choix parmi toutes les techniques de similarité. Chacune des mesures a des propriétés différentes, et peut aussi connaître des variations en fonction de la base et des signatures. L'intérêt d'une similarité par rapport à une autre réside dans sa capacité à favoriser au mieux l'apprentissage des catégories recherchées.

L'apprentissage repose principalement sur une corrélation entre la description des éléments et les catégories formées. Reprenons l'exemple de (Cristianini et al., 2002), qui discute de la forme de la matrice de toutes les similarités entre tous les éléments[1]. On peut distinguer deux extrêmes. Le premier est le cas où les images ont toutes la même similarité entre elles ($\forall i, j \; s(i,j) = 1$). Dans ce cas, toutes les images sont similaires à

[1]Plus précisément, Cristianini discute la forme de la matrice de Gram, qui peut être vue comme une matrice de similarité particulière.

toutes les autres, si bien qu'il devient impossible de les distinguer. C'est le cas extrême de *généralisation* ou *sous-apprentissage*. L'autre extrême est le cas où toutes les images sont non-similaires aux autres ($\forall i \neq j \; s(i,j) = 0$). Dans ce cas, aucune image n'est similaire à une autre. Il est certe alors possible de différiencier les images, par contre il devient impossible de généraliser. On ne peut en aucun cas déduire l'estimation de la classe d'une image connaissant la classe d'autres images. C'est le cas extrême de *mémorisation* ou *sur-apprentissage*.

Bien que ces deux cas extrêmes ne soient jamais atteints dans la pratique, un mauvais réglage peut facilement l'approcher. Par exemple, l'un des démons des très grandes dimensions (Hastie et al., 2001b) est de favoriser le deuxième cas extrême, où toutes les images sont non-similaires. Ceci montre que la résolution du problème d'apprentissage ne réside pas dans la recherche d'un extremum, mais dans la détermination d'un compromis entre les deux extrêmes.

La détermination de ce compromis est difficile à faire dans notre contexte où les catégories recherchées sont inconnues à l'avance. On peut cependant prévenir les cas extrêmes en passant par un processus d'*analyse des similarités*. (Brunelli & Mich, 2001) proposent une analyse basée sur la mesure de la capacité des histogrammes. Cette technique calcule la distribution des dissimilarités entre toutes les images de la base. L'observation de cette distribution sur un graphique permet de juger la capacité discriminante de la similarité choisie. Plus les dissimilarités sont concentrées, plus les images sont à la même distance les unes des autres, et plus il est difficile de les distinguer. Au contraire, plus les dissimilarités sont dispersées, plus les images sont à des distances différentes, plus il est facile de les discriminer.

3.1.3 Choix de l'approche noyau

Dans cette thèse, nous avons choisi d'utiliser une similarité basée sur des fonctions noyaux. Cette approche ne fournit pas une métrique particulière pour tel ou tel types de signatures, mais un cadre formel qui offre de nombreux avantages.

L'approche de la similarité par fonction noyau consiste à utiliser un produit scalaire comme fonction de similarité. Dans le cas où les signatures $\mathbf{x}_i$ des images sont des vecteurs, on utilise alors la similarité $s(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$. Dans le cas où les signatures ne sont pas des vecteurs, il est nécessaire de construire une fonction noyau adaptée au type des signatures. Par exemple, si l'on dispose d'une distance $d(\mathbf{x}_i, \mathbf{x}_j)$ entre les images, on peut transformer la distance d avec une gaussienne :

$$k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{d(\mathbf{x}_i, \mathbf{x}_j)}{2\sigma^2}}$$

La fonction $k(\mathbf{x}_i, \mathbf{x}_j)$ est alors le produit scalaire d'un certain espace hilbertien[2].

[2]Nous donnons cet exemple à titre illustratif, et nous n'énonçons pas les différentes propriétés nécessaires ou suffisantes que doit respecter la distance d.

D'une manière générale, cette approche propose de transformer un problème non-linéaire en un problème linéaire, ce qui simplifie l'étude des différentes problématiques. De plus, nous pouvons utiliser les nombreuses techniques linéaires d'analyse et d'apprentissage.

Nous présentons plus en détail cette approche dans les paragraphes qui suivent.

3.2 Fonctions Noyaux

Les méthodes à noyaux forment une nouvelle famille d'algorithmes aux propriétés intéressantes pour la recherche d'information (Vapnik, 1999; Cristianini & Shawe-Taylor, 2000; Schölkopf & Smola, 2002; Suykens et al., 2002). Leur simplicité liée à leur efficacité ont fait de ces techniques un outil incontournable pour certains chercheurs, et un sujet de recherche fondamentale en apprentissage.

Dans cette section, nous nous intéressons à l'ensemble des similarités $s(\mathbf{x}_i, \mathbf{x}_j)$ des images de la base. Nous présentons le fait que, si ces similarités ont certaines propriétés, alors elles correspondent aux valeurs d'un produit scalaire. Nous présentons ensuite le fameux *truc du noyau*, qui permet de transformer une méthode basée sur des produits scalaires en une méthode à noyau. Enfin, nous présentons des exemples de fonctions noyaux.

3.2.1 Similarité et fonctions noyaux

Dans ce paragraphe nous nous intéressons aux conditions sur la matrice $S_{ij} = s(\mathbf{x}_i, \mathbf{x}_j)$ de toutes les similarités $s(\mathbf{x}_i, \mathbf{x}_j)$ entre les images de la base pour que la fonction de similarité soit un produit scalaire, *i.e.* $s(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$.

Supposons que nous ayons une injection $\Phi : X \to \mathcal{H}$, qui a toute signature $\mathbf{x}_i$ fait correspondre un vecteur $\Phi(\mathbf{x}_i)$ dans un espace hilbertien $\mathcal{H}$, telle que la fonction de similarité soit le produit scalaire de $\mathcal{H}$:

$$s(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$$

Ce produit scalaire sur les images des signatures par une injection est une *fonction noyau* $k(\mathbf{x}_i, \mathbf{x}_j)$ sur X :

$$\begin{array}{rccl} k: & X \times X & \to & \mathbb{R} \\ & \mathbf{x}_i, \mathbf{x}_j & \mapsto & k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \end{array}$$

La matrice des similarités $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ est appelée *matrice de Gram* (Schölkopf & Smola, 2002) :

Définition 3.1 (Matrice de Gram) *Soit une fonction noyau k sur un ensemble X de n vecteurs de $\mathbb{R}^p$, alors la matrice $\mathbf{K}$ de $n \times n$ réels definie par :*

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$$

est appelée matrice de Gram *de k sur X.*

Connaissant une fonction noyau k, il est ainsi possible de construire une matrice de Gram. Il est toutefois aussi possible d'effectuer l'opération inverse, si, par exemple, la matrice de Gram est semi-définie positive :

Définition 3.2 (Matrice Semi-Définie Positive (*sdp*)) *Une matrice $n \times n$ $\mathbf{K}$ est semi-définie postive si elle satisfait l'une des propriétées (équivalentes) suivantes :*

(i) $\forall \mathbf{v} \in \mathbb{R}^n,\ \mathbf{v}^\top \mathbf{K} \mathbf{v} \geq 0$

(ii) Les valeurs propres de $\mathbf{K}$ sont positives ou nulles.

Nous utilisons la notation "matrice sdp" pour désigner une matrice semi-définie positive.

Lorsqu'une matrice $\mathbf{K}$ est *sdp*, alors elle est la matrice de Gram du produit scalaire sur un certain ensemble $\Phi(X) \subset \mathcal{H}$ (Schölkopf & Smola, 2002). $\mathbf{K}$ peut se factoriser sous la forme :

$$\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top = (\mathbf{V}\sqrt{\mathbf{\Lambda}})(\mathbf{V}\sqrt{\mathbf{\Lambda}})^\top = \Phi(X)^\top \Phi(X)$$

avec $\mathbf{V}$ la matrice $n \times p$ les vecteurs propres de $\mathbf{K}$, et $\mathbf{\Lambda}$ la matrice diagonale $p \times p$ des valeurs propres de $\mathbf{K}$.

Ainsi, une condition suffisante pour que la matrice des similarités soit l'ensemble des valeurs d'un produit scalaire est donc d'assurer qu'elle est semi-définie positive (*sdp*).

3.2.2 Le truc du noyau

Le *truc du noyau* établit que tout algorithme formulé avec une fonction noyau *sdp* peut être reformulé avec une autre fonction noyau *sdp*. Une démarche courante est d'exprimer l'algorithme avec un produit scalaire (qui est une fonction noyau *sdp*), puis de remplacer tous ces produits par une fonction noyau *sdp*. Le changement conserve toutes les propriétés de l'algorithme, hormis le fait que les opérations s'effectuent dans un autre espace $\mathcal{H}$. Par exemple, la projection sur un hyperplan de normale $\mathbf{w}$: $f(\mathbf{x}_i) = \langle \mathbf{w}, \mathbf{x}_i \rangle$, peut être « noyautée » par : $f(\mathbf{x}_i) = k(\mathbf{w}, \mathbf{x}_i)$.

On peut aussi appliquer le « truc » aux algorithmes basés sur une distance Euclidienne, sachant que :

$$\begin{aligned} d(\mathbf{x}_i, \mathbf{x}_j)^2 &= \langle \mathbf{x}_i - \mathbf{x}_j, \mathbf{x}_i - \mathbf{x}_j \rangle \\ &= \langle \mathbf{x}_i, \mathbf{x}_i \rangle + \langle \mathbf{x}_j, \mathbf{x}_j \rangle - 2\langle \mathbf{x}_i, \mathbf{x}_j \rangle \end{aligned}$$

D'où :

$$d(\mathbf{x}_i, \mathbf{x}_j)^2 = k(\mathbf{x}_i, \mathbf{x}_i) + k(\mathbf{x}_j, \mathbf{x}_j) - 2k(\mathbf{x}_i, \mathbf{x}_j)$$

Le principe est aussi valable pour tout algorithme formulé à partir de $\mathbf{X}^\top\mathbf{X}$. On peut alors remplacer $\mathbf{X}^\top\mathbf{X}$ par la matrice $\mathbf{K}$.

Par exemple, l'Analyse en Composantes Principales (PCA) revient à résoudre le problème de décomposition en valeur propres suivant :

$$(\mathbf{X}\mathbf{X}^\top)\mathbf{w} = \lambda\mathbf{w}$$

$\mathbf{w}$ est une combinaison linéaire des $\mathbf{x}_i$, *i.e.* $\exists\boldsymbol{\alpha}$ / $\mathbf{w} = \mathbf{X}\boldsymbol{\alpha}$, ce qui donne :

$$\begin{aligned} & (\mathbf{X}\mathbf{X}^\top)\mathbf{w} &&= \lambda\mathbf{w} \\ \Leftrightarrow \quad & (\mathbf{X}\mathbf{X}^\top)\mathbf{X}\boldsymbol{\alpha} &&= \lambda\mathbf{X}\boldsymbol{\alpha} \\ \Leftrightarrow \quad & \mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)\mathbf{X}\boldsymbol{\alpha} &&= \lambda\mathbf{X}^\top\mathbf{X}\boldsymbol{\alpha} \\ \Leftrightarrow \quad & (\mathbf{K}^\top)^2\boldsymbol{\alpha} &&= \lambda\mathbf{K}^\top\boldsymbol{\alpha} \\ \Leftrightarrow \quad & \mathbf{K}\boldsymbol{\alpha} &&= \lambda\boldsymbol{\alpha} \end{aligned}$$

Le problème revient donc à calculer la décomposition en valeurs propres de $\mathbf{K}$. $\mathbf{K}$ et $\mathbf{X}\mathbf{X}^\top$ ont les mêmes valeurs propres, et les vecteurs propres se déduisent de l'expression $\mathbf{w} = \mathbf{X}\boldsymbol{\alpha}$. On parle alors d'Analyse en Composantes Principales Noyautée (KPCA, *Kernel Principal Component Analysis*).

3.2.3 Exemple avec les monômes

Nous avons présenté le principe de l'approche par fonction noyau, cependant il reste à déterminer les fonctions en question. La détermination de ces fonctions, ainsi que les approches pour les construire sont actuellement un thème de recherche très actif. La technique la plus courante est d'en revenir à la définition première, en tentant de trouver une fonction $k(\mathbf{x}_i, \mathbf{x}_j)$ qui soit un produit scalaire. Nous présentons dans ce paragraphe un exemple de construction sur ce principe.

Supposons que nous nous intéressions à l'information contenue dans les produits d'ordre d (ou monômes) entre les valeurs x_r d'un vecteur $\mathbf{x} \in \mathbb{R}^p$:

$$x_{r_1} x_{r_2} \dots x_{r_d}$$

Ces monômes sont parfois appelés attributs produit (*product features*). Ce type d'attribut forme la base de nombreux algorithmes, par exemple en reconnaissance de forme basée sur les classifieurs polynomiaux (Schurmann, 1996). La motivation derrière ces attributs est de calculer toutes les corrélations possibles jusqu'à l'ordre d, puis de déceler la correlation qui caractérise l'objet ou la classe.

Prenons par exemple le cas des produits d'ordre $d = 2$ dans un espace de dimension $p = 2$, ce qui conduit à un espace de tous les produits de dimension 3 :

$$\begin{array}{rccc} \Phi & : & \mathbb{R}^2 & \rightarrow & \mathcal{H} = \mathbb{R}^3 \\ & & (x_1, x_2) & \mapsto & (x_1^2, x_2^2, x_1 x_2) \end{array}$$

On peut démontrer que la dimension de l'espace de tous les produits (non ordonnés) à l'ordre d pour des vecteurs de dimension p est :

$$C_{d+p-1}^{d} = \frac{(d+p-1)!}{d!(p-1)!}$$

Nous voyons très clairement ici que le calcul direct de cet espace devient rapidement impossible. Par exemple, avec des paramètres classiques en reconnaissance des formes, $d = 5$ et $p = 256$, la dimension de $\mathcal{H}$ est de l'ordre de 10^{10}.

Cependant, grâce à la théorie des fonctions noyaux, il est possible de travailler sur cet espace sans jamais le calculer. L'idée principale est de ne calculer que les produits scalaires $\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$ dans $\mathcal{H}$ entre les éléments de X. Nous faisons ainsi abstraction de la dimension de $\mathcal{H}$. Ce produit scalaire est évalué par une fonction noyaux k :

$$\begin{array}{rccc} k : & \mathbb{R}^p \times \mathbb{R}^p & \rightarrow & \mathbb{R} \\ & \mathbf{x}, \mathbf{x}' & \mapsto & k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle \end{array}$$

En effet, en reprenant l'exemple des monômes dans le cas ordonné, avec l'injection suivante :

$$\Phi(x_1, x_2) = (x_1^2, x_2^2, x_1 x_2, x_2 x_1)$$

le produit scalaire entre deux vecteurs $\mathbf{x}$ et $\mathbf{x}'$ est :

$$\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle = x_1^2 x_1'^2 + x_2^2 x_2'^2 + 2 x_1 x_2 x_1' x_2' = \langle \mathbf{x}, \mathbf{x}' \rangle^2 = k(\mathbf{x}, \mathbf{x}')$$

Nous voyons par cet exemple que le produit scalaire entre deux éléments de $\mathcal{H}$ est peu coûteux à calculer, en comparaison avec le calcul direct. Notons que ceci est aussi vrai pour tout ordre d, auquel cas la fonction noyau est $\langle \mathbf{x}, \mathbf{x}' \rangle^d$, ce qui nous donne le *noyau polynomial*.

3.2.4 Fonctions noyaux classiques

– **Linéaire** :

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

Cette fonction est la fonction "pas de noyau". Cela nous permet de comparer à une utilisation sans approche noyau.

– **Gaussien avec une distance Euclidienne** (Gaussien L2) :

$$k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}}$$

Nous utiliserons cette fonction sur des données normalisées, *i.e.* $\sigma = 1$.

– **Polynomiale de degré d** :

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle^d$$

– **Triangulaire** :

$$k(\mathbf{x}_i, \mathbf{x}_j) = 1 - \frac{1}{\sigma^2} ||\mathbf{x}_i - \mathbf{x}_j||$$

Ce noyau particulier (Berg et al., 1984) a été introduit pour la recherche d'images pour ses propriétés d'invariance aux échelles (Sahbi, 2003). Afin d'avoir un noyau *sdp*, les données sont placées dans une sphère de rayon $\sigma^2/2$. Dans les expériences qui suivent, nous avons pris $\sigma = 1$.

– **Gaussien avec une distance du χ^2** (Gaussien Chi2) :

$$k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{d(\mathbf{x}_i, \mathbf{x}_j)^2}{2\sigma^2}}$$

avec d une distance au sens du χ^2 :

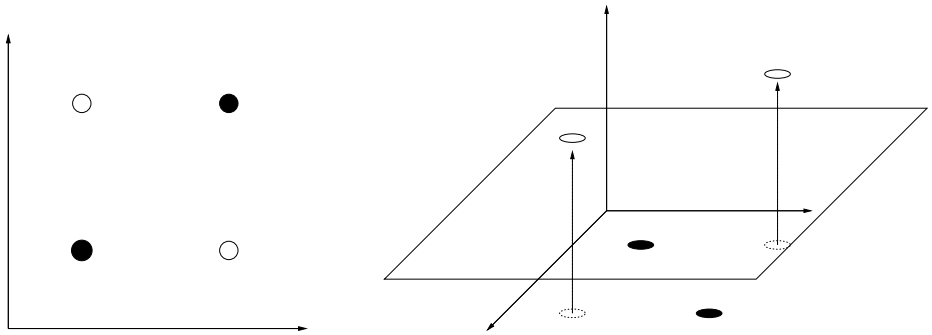


FIG. 3.1 – Dimension et pouvoir discriminant des classifieurs. Dans le cas à deux dimensions (a), il est impossible de séparer les points noirs et blancs à l'aide d'un hyperplan, alors que dans le cas à trois dimensions (b), cela devient possible.

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sum_{r=1}^{p} \frac{(x_{ri} - x_{rj})^2}{x_{ri} + x_{rj}}$$

Cette fonction est un exemple de « noyautisation » d'une technique de calcul de similarité sur les histogrammes.

3.3 Dimension et rang de la matrice de Gram

Nous nous intéressons à présent à l'intérêt de l'approche par fonctions noyaux dans le cadre de l'apprentissage. Dans le contexte généraliste, il est difficile de faire des *a priori* sur la nature des catégories qui vont être recherchées, mais il est cependant possible de régler l'un des paramètres les plus importants : la dimension de l'espace de représentation de la base. Comme nous avons pu le voir lors des expérimentations au chapitre 2, ce paramètre peut modifier de manière importante la capacité à apprendre du système.

Cette partie permet de mettre en valeur l'intérêt des fonctions noyaux pour ce type de réglage. Nous commençons par présenter les possibilités offertes par l'injection dans un espace de dimension plus grand que nous pouvons réaliser avec l'approche noyau. Nous présentons ensuite les liens théoriques qui existent entre la dimension de l'espace de représentation et la classification, et enfin les relations entre la dimension et le rang de la matrice de Gram.

3.3.1 Dimension de l'espace de représentation

Le principal problème de la recherche d'images dans les bases généralistes est de traiter le fossé sémantique/numérique. Comme nous l'avons exprimé précédemment, il est très

difficile de prédire les catégories recherchées par les utilisateurs. Il n'est donc pas possible d'envisager une forte généralisation sans *a priori*, et nous avons tout d'abord affaire avec des problèmes de mémorisation. En effet, étant donné qu'il existe un très grand nombre de catégories pouvant être recherchées, il est nécessaire que l'espace sur lequel s'appuie le classifieur lui permette de décrire des classes complexes (*i.e.* réparties en nombreux modes)[3].

Prenons l'exemple de la figure 3.1(a), où la base est formée de quatre images représentées par des vecteurs du plan, deux dans la classe recherchée (points blancs) et deux dans la classe non pertinente (points noirs). Supposons à présent que nous souhaitons les séparer à l'aide d'un hyperplan. On peut voir sur la figure 3.1(a) qu'il est impossible de les séparer avec une droite. Or, si nous les injectons dans un espace de dimension 3 comme sur la figure 3.1(b), il devient alors possible de les séparer.

Notons que pour l'exemple de la figure 3.1(a), l'infaisabilité réside dans le choix des classifieurs à hyperplans. Une des motivations relatives à l'utilisation d'injections est de ne plus concentrer tous les apports sur la méthode de classification, mais sur les propriétés de l'espace de représentation des images. Ceci permet l'utilisation de classifieurs plus simples à décrire, comme les classifieurs à hyperplans. Nous développons davantage cet aspect au paragraphe suivant, en présentant les liens avec la théorie de l'apprentissage proposée par (Vapnik, 1999).

Bien sûr, il ne suffit pas d'effectuer n'importe quelle injection pour obtenir le comportement voulu. Par exemple, l'injection suivante :

$$\begin{array}{rccl} \Phi & : & X = \mathbb{R}^2 & \rightarrow & \mathcal{H} = \mathbb{R}^3 \\ & & \mathbf{x} & \mapsto & \Phi(\mathbf{x}) = (x_1, x_2, 0) \end{array}$$

n'apporterait aucune différence, bien que les vecteurs soient de dimension 3. La dimension importante est celle de l'espace engendré par les vecteurs de $\Phi(X)$ et non la dimension de $\mathcal{H}$. Nous parlerons alors de *dimension utile*.

3.3.2 Dimension de Vapnik-Chervonenkis

Nous présentons dans ce paragraphe quelques éléments de la théorie de l'apprentissage statistique (Vapnik, 1999) importants dans notre contexte de classification binaire.

Cette théorie modélise le problème de l'apprentissage, qui consiste à trouver le compromis optimal déjà évoqué entre généralisation et mémorisation. Ce compromis est aussi appelé *bias-variance tradeoff*, *capacity control*, ou *over-fitting*. Nous nous intéressons ici à la classification binaire. La théorie repose sur une première hypothèse qui suppose que les données sont échantillonnées selon une distribution $P(\mathbf{x}, y)$. Sachant cela, le problème

[3]Notons que, bien que nous insistions sur le problème de la mémorisation dans ce paragraphe, la nécessité de généraliser reste toujours valable, et notre problème principal reste le compromis à trouver entre généralisation et mémorisation.

d'apprentissage s'exprime sous la forme de la minimisation du *risque*, fonction de tout classifieur f :

$$R_{\text{réel}}(f) = \int_{\mathbb{R}^p \times \{\pm 1\}} L(f(\mathbf{x}), y) dP(\mathbf{x}, y) \tag{3.1}$$

La fonction $L(f(\mathbf{x}), y)$ est une fonction de perte qui évalue l'erreur entre la prédiction $f(\mathbf{x})$ et la classe y. Par exemple, on peut choisir la fonction de perte tout ou rien, $L(f(\mathbf{x}), y) = 1 - \delta(f(\mathbf{x}), y)$.

Cette quantité ne peut être évaluée, et dans la pratique elle est approximée par le *risque empirique*, connaissant la classe y_i de chaque individu $\mathbf{x}_i$:

$$R_{\text{emp}}(f) = \frac{1}{|I|} \sum_{i \in I} L(f(\mathbf{x}_i), y_i) \tag{3.2}$$

Le problème de la classification se formalise alors par la minimisation du risque réel ne connaissant que le risque empirique. Plusieurs outils d'apprentissage ont été proposés dans ce cadre, en particulier Vapnik propose une notion de capacité à généraliser, la dimension VC (Vapnik-Chervonenkis) qui est définie comme suit :

Définition 3.3 (Dimension VC) *La dimension VC d'une classe de fonctions $\mathcal{F}$ est définie comme le nombre maximum de points qui peuvent être exactement appris par une fonction de $\mathcal{F}$,*

$$\mathrm{h}_{\mathcal{F}} = max \left\{ |\hat{X} \mid tel\ que\ \hat{X} \subset X,\ \forall b \in \{-1, +1\}^{|\hat{X}|},\ \exists f \in \mathcal{F} \mid \forall \mathbf{x}_i \in \hat{X}, f(\mathbf{x}_i) = b_i \right\}$$

Avec cette mesure de capacité, on a le théorème suivant :

Théorème 3.1 (Compromis généralisation/mémorisation) *Soit $\mathcal{F}$ un ensemble de fonctions de décision sur X de dimension VC h, $\eta \in [0, 1]$, $\mathbf{y}$ des annotations sur X, $n = |\mathbf{y}|$, et :*

$$\epsilon(h) = \frac{2}{n} \Big(h \log \big(\frac{2n}{h}\big) + \log \frac{2}{\eta} \Big)$$

Alors :

$$P\Big(R_{réel}(f_{\mathbf{y}}) \leq R_{emp}(f_{\mathbf{y}}) + \epsilon(h) \Big) \geq 1 - \eta$$

Ce théorème exprime le compromis entre généralisation et mémorisation. Le but étant d'approcher au mieux le risque réel $R_{\text{réel}}(f_{\mathbf{y}})$, il faut minimiser le risque empirique $R_{\text{emp}}(f_{\mathbf{y}})$ et ϵ fonction de la dimension VC h. Ces deux valeurs sont liées et dépendent des classifieurs utilisés et de la dimension de l'espace de représentation.

Prenons par exemple les classifieurs à hyperplan sur $\mathbb{R}^p$, dont la dimension VC est $p+1$ (Sontag, 1998). Il est ainsi possible d'effectuer n'importe quelle séparation par hyperplan d'au plus $p + 1$ points pris dans X.

Théorème 3.2 (Dimension VC des hyperplans) *Soit $\mathcal{F}$ l'ensemble des hyperplans de $\mathbb{R}^p$,*

$$\mathcal{F} = \{\mathbf{x} \mapsto sgn(< \mathbf{w}, \mathbf{x} > +b), \mathbf{x}, \mathbf{w} \in \mathbb{R}^p, b \in \mathbb{R}\}$$

Alors la dimension VC de $\mathcal{F}$ est $p+1$.

Plus la dimension p est grande, plus la dimension VC h est importante, et plus $\epsilon(h)$ est élevé. Or, augmenter la dimension VC revient aussi à augmenter le pouvoir discriminant des classifieurs, donc leur capacité à mémoriser. En effet, dans ce cas, il est possible de trouver un hyperplan tel que $f(\mathbf{x}_i) = y_i$, et donc d'avoir un risque empirique nul.

Ainsi, augmenter la dimension p revient à augmenter la dimension VC, et par conséquent modifier de manière opposée le risque empirique et ϵ dans le théorème 3.1. Le but est alors de trouver la dimension p où le meilleur compromis sera réalisé.

Une technique pour régler au mieux la dimension de l'espace de représentation est d'utiliser les fonctions noyaux. En effet, en choisissant une fonction noyau, on choisit de manière implicite une injection qui va modifier la dimension de l'espace de représentation. Nous proposons au paragraphe suivant une méthode d'analyse du rang de la matrice de Gram afin de pouvoir évaluer cette nouvelle dimension.

3.3.3 Rang de la matrice de Gram

La dimension VC des hyperplans dans un espace de dimension p est $p+1$. Cela signifie que, dans un espace de dimension p, nous pouvons séparer par un hyperplan au plus $p+1$ points. Or, si nous supposons que nous disposons d'une matrice de Gram $\mathbf{K}$ (*i.e.* $\mathbf{K} = \Phi(X)^\top \Phi(X)$), alors la dimension des vecteurs du sous-espace de $\mathcal{H}$ engendré par $\Phi(X)$ est égale au rang r de la matrice $\mathbf{K}$.

Dans notre contexte de recherche d'images, de manière à faire face au fossé sémantique/numérique, nous avons besoin de former des classes très complexes, donc d'une grande dimension VC. Puisque nous disposons de tous les individus lors de l'utilisation du système, nous pouvons considérer directement le produit scalaire dans l'espace induit par la fonction noyau, et ainsi calculer la dimension des vecteurs dans cet espace *via* le rang de la matrice de Gram (à l'aide d'une KPCA). Cela nous permet de choisir les fonctions noyaux dont le rang de la matrice de Gram sur la base d'images est élevé, sans pour autant perdre toute capacité à généraliser.

L'étude du rang de la matrice de Gram peut toutefois poser quelques problèmes. Par définition, le rang est le nombre de valeurs propres non nulles d'une matrice. Or, pour beaucoup de fonctions noyaux, la matrice de Gram est toujours de rang plein. Cependant, on peut avoir des rapports très grand entre les plus fortes et les plus petites valeurs propres. Afin de pouvoir comparer la capacité à mémoriser de différentes fonctions noyaux, nous nous intéressons à la répartition des valeurs propres, et non au rang.

Si l'on calcule les valeurs propres $\lambda_1 < \lambda_2 < \cdots < \lambda_n$ de la matrice de Gram, on peut en déduire leur répartition énergétique :

$$\hat{\lambda}_r = \frac{\sum_{i=1}^{r} \lambda_i}{\sum_{i=1}^{n} \lambda_i}$$

Notons que nous faisons un abus de langage en utilisant le terme *rang fort* lorsque l'énergie des valeurs propres est bien répartie, et *rang faible* lorsque l'énergie des valeurs propres est concentrée. Ainsi, une rang fort correspond à une grande dimension VC, et donc à une grande capacité à décrire les classes complexes.

3.4 Expérimentations

3.4.1 Répartition de l'énergie des valeurs propres

Nous avons calculé les valeurs propres $\lambda_1 < \lambda_2 < \cdots < \lambda_{6000}$ de la matrice de Gram sur la base COREL (*cf.* annexe A) pour chaque fonction en utilisant une KPCA. Ces valeurs propres sont représentées sur la figure 3.2 par ordre décroissant. Notons que sur cette figure, nous ne représentons qu'un zoom sur la partie intéressante des courbes pour des raisons de lisibilité. Toutes les courbes (sauf dans le cas linéaire) se poursuivent naturellement vers la droite avec des valeurs non nulles extrêmement faibles.

Nous avons ensuite calculé la répartition énergétique $\hat{\lambda}_1, \ldots, \hat{\lambda}_{6000}$, qui est représentée sur la figure 3.3. Nous pouvons observer qu'avec les fonctions noyaux, l'énergie est plus répartie que dans le cas linéaire, ce qui implique une grande dimension VC, et donc la possibilité de décrire des classes complexes. Il est intéressant de noter qu'une grande partie de l'énergie reste localisée dans les premières valeurs propres. Pour tous les noyaux, on trouve plus de la moitié de l'énergie dans les 50 premières valeurs propres. Cela signifie qu'une grande capacité à généraliser est conservée, tout en offrant de nombreuses dimensions pour décrire les classes complexes.

3.4.2 Performances avec une classification par SVM

Nous avons évalué les performances en terme de Précision Moyenne à l'aide d'une classification par SVM (*cf.* chapitre 4, suivant le protocole présenté en annexe A. Les résultats sont présentés sur la figure 3.4.

Si nous faisons le parallèle entre ces résultats et la répartion de l'énergie des valeurs propres, nous pouvons obverser que plus l'énergie des valeurs propres est répartie, plus les performances sont élevées.

Ce comportement en terme de répartion énergétique n'est clairement pas l'unique facteur de réussite. Il est fort probable que, si nous construisions une représentation de la base de répartition énergétique identique au Gaussien Chi2, les performances soient très différentes. Ce point explique très certainement le saut plus important qui existe

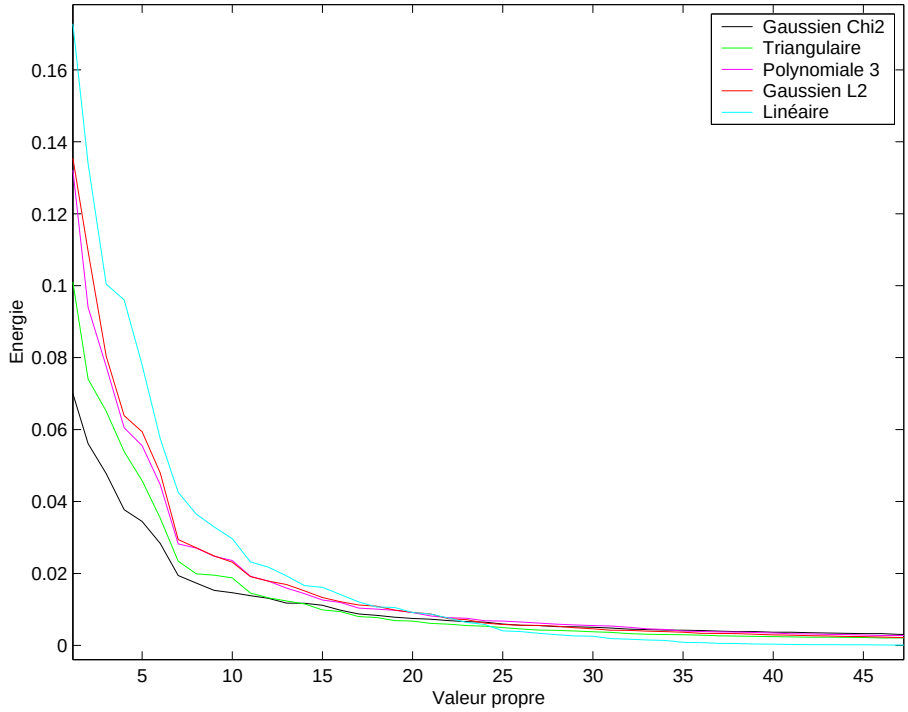


FIG. 3.2 – Valeurs propres pour différentes fonctions noyaux sur la base COREL.

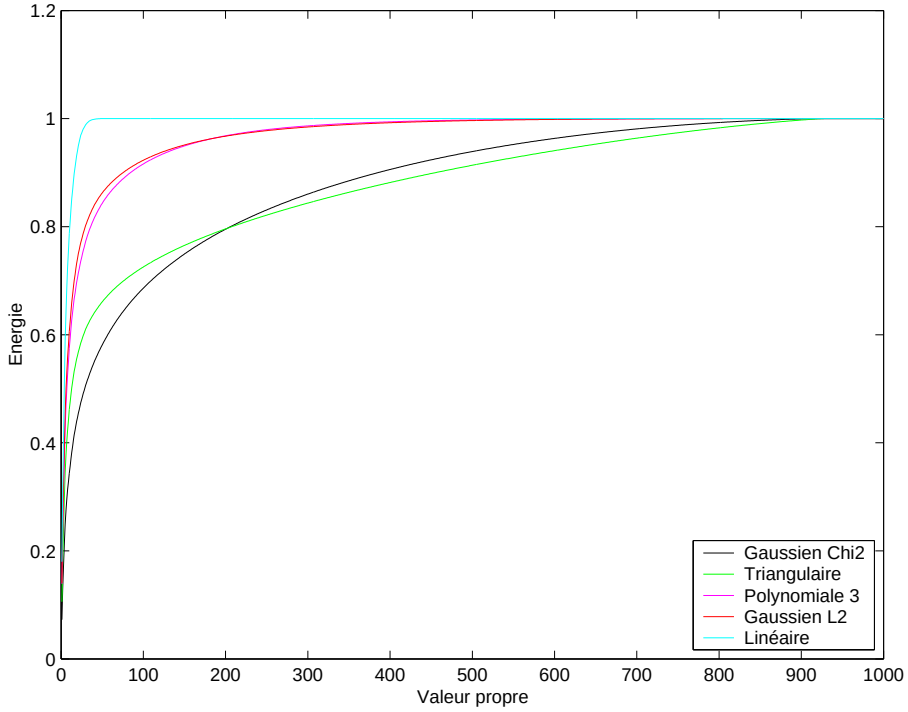


FIG. 3.3 – Répartition de l'énergie des valeurs propres pour différentes fonctions noyaux sur la base COREL.

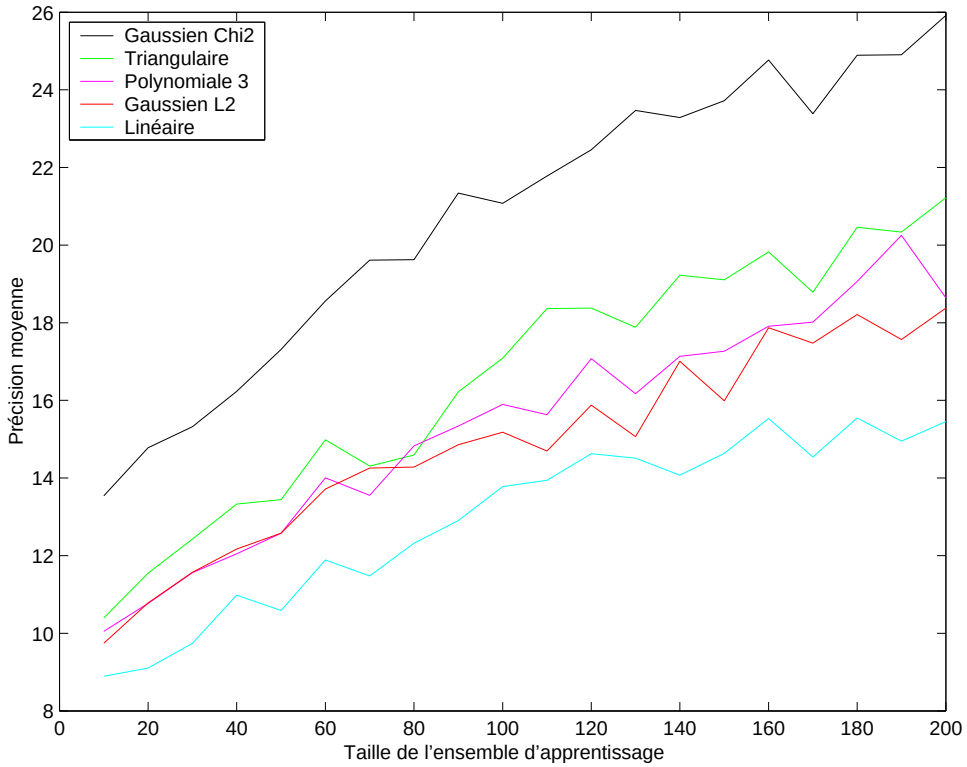


FIG. 3.4 – Précision Moyenne (en pourcentages) en fontion de la taille de l'ensemble d'apprentissage d'un SVM, pour différentes fonctions noyaux sur la base COREL.

entre Gaussien Chi2 et les autres, compte tenu du fait que ce noyau est fait pour des distributions, distributions qui sont utilisées pour la représentation initiale lors de ces expériences. La représentation via une fonction noyau reste très dépendante des espaces propres à forte énergie.

Cette méthode d'analyse est un outil pour avoir un idée de l'*a priori* que nous pouvons faire sur le comportement d'une fonction noyau sur une base donnée, ne connaissant pas à l'avance les classes recherchées. Par exemple, si nous savons que la base est susceptible d'être principalement divisée en 2 ou 3 catégories, une fonction noyau de rang plus faible sera préférable. A l'inverse, si nous savons qu'un très grand nombre de catégories seront recherchées, alors une fonction noyau de rang fort sera préférable.

3.5 Conclusion

Dans ce chapitre, nous avons présenté nos motivations et nos arguments quant à l'utilisation des fonctions noyaux en tant que fonction de similarité. Cette approche permet d'utiliser un grand nombre de techniques d'analyse et d'exploitation des données. Cela permet de s'appuyer sur la théorie de l'apprentissage statistique pour mieux appréhender le compromis à faire entre la *généralisation* et la *mémorisation* (ou *sur-apprentissage*). L'un des outils que nous pouvons utiliser est la dimension VC (Vapnik-Chervonenkis), qui per-

met d'exprimer les relations qui existent entre la dimension de l'espace de représentation et la capacité d'apprentissage. Cette dimension est liée au spectre de la matrice de Gram, dont l'analyse expérimentale permet d'orienter le choix d'une fonction noyau.

Outre les différents avantages en terme d'analyse et d'exploitation, l'approche noyau permet aussi de séparer le problème de la classification du problème de la représentation. Nous pouvons utiliser des classifieurs plus simples et donc plus faciles à manipuler, tout en ayant la possibilité de travailler avec des signatures complexes. Ainsi, toutes les techniques d'apprentissage que nous présentons dans cette thèse seront utilisables avec n'importe quel type de signatures, sous réserve d'avoir construit une fonction noyau adaptée.

Deuxième partie

Apprentissage interactif

Chapitre 4

Classification pour la recherche d'images

Les méthodes de recherche interactive d'images les plus rencontrées font souvent appel au bouclage de pertinence (*relevance feedback*), qui vise à combler le fossé sémantique/numérique à l'aide des annotations de l'utilisateur.

Nous présentons dans ce chapitre le principe du bouclage de pertinence, ainsi que différentes approches suivant ce schéma. Nous présentons ensuite l'approche que nous avons choisie qui s'appuie sur une classification binaire pour la recherche d'une catégorie d'images. Nous proposons ensuite une confrontation expérimentale des approches, afin de déterminer la méthode de classification la mieux adaptée à notre contexte. Le choix qui en découle nous permet ensuite de nous concentrer sur les autres aspects de notre problème, comme l'apprentissage actif au chapitre suivant, et l'apprentissage long terme dans la partie III.

4.1 Le bouclage de pertinence

4.1.1 Principe

Le but de la recherche est de retrouver les images appartenant à la catégorie recherchée par l'utilisateur. Le processus est démarré par une *requête*, en général une image que présente l'utilisateur. Cette requête permet un premier classement des images en fonction de leur pertinence, *i.e.* leur appartenance à la catégorie recherchée. L'utilisateur a la possibilité de fournir des précisions quant à la catégorie qu'il recherche, par exemple sous la forme d'annotations. A l'aide de ces précisions, le système peut calculer un nouveau classement et présenter de nouvelles images. L'utilisateur peut fournir de nouvelles annotations autant de fois qu'il le souhaite, et à chaque nouvelle mise à jour le système recalcule la pertinence des images.

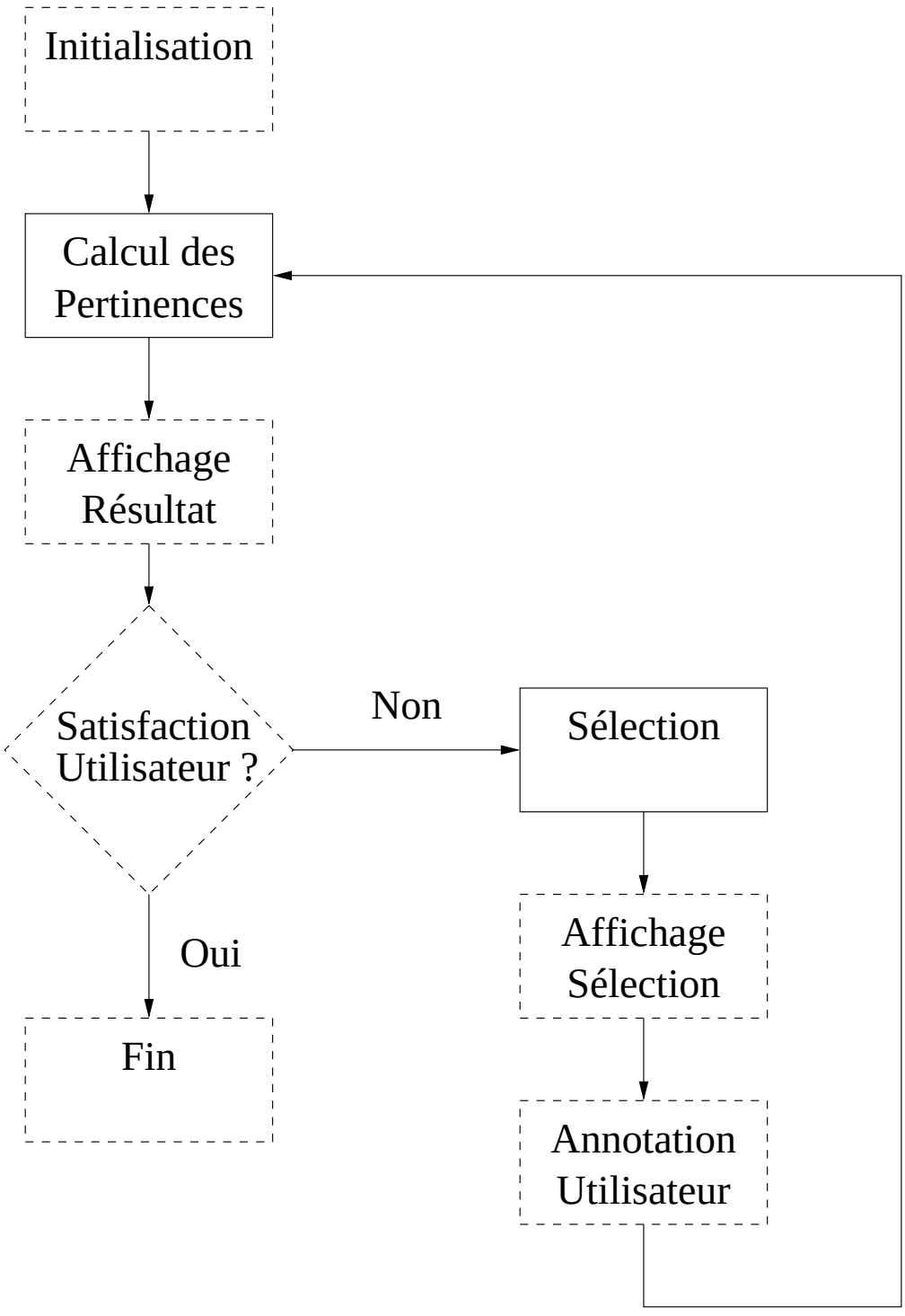


Fig. 4.1 – Achitecture du bouclage de pertinence

Nous présentons sur le schéma 4.1 une architecture de bouclage de pertinence pour la recherche de catégories d'images (Tong & Koller, 2001; Chang et al., 2003).

Deux étapes clefs sont à considérer au sein du schéma du bouclage de pertinence. L'étape du *Calcul des pertinences* estime la *pertinence* de chaque image, *i.e.* la probabilité d'appartenir à la catégorie recherchée. Il en résulte un classement qui est présenté à l'utilisateur, qui peut choisir de terminer la session si il est satisfait, ou bien de poursuivre. Dans le cas où l'utilisateur souhaite poursuivre sa session, le système propose, lors de l'étape de *sélection*, des images que l'utilisateur peut annoter. Une fois les images annotées, le système utilise ces nouvelles informations pour calculer de nouvelles pertinences, et ainsi de suite jusqu'à la satisfaction de l'utilisateur.

Il existe plusieurs manières de *sélectionner* les exemples. L'approche la plus naïve consiste à tirer aléatoirement des images dans la base. Cette technique est particulièrement inefficace puisque les concepts recherchés ont souvent très peu d'images au sein d'une même base, et la probabilité de tirer une image dans un concept donné est très faible. Il devient alors difficile de déterminer la catégorie avec peu d'annotations. Une technique très courante est de demander à l'utilisateur d'annoter les images les plus pertinentes, Cependant, cette solution n'est pas nécessairement la plus efficace. C'est le cadre d'étude de l'*apprentissage actif*, où d'autres critères que la pertinence peuvent être choisis pour sélectionner les images (Cohn, 1996; Tong & Koller, 2000).

4.1.2 Annotations

La manière la plus simple d'annoter est de spécifier si une image appartient ou non à la catégorie recherchée (annotation binaire). Nous dirons qu'une image est pertinente lorsqu'elle appartient à la catégorie recherchée, et non pertinente dans le cas contraire.

L'annotation binaire peut sembler assez limitée. En effet, certains chercheurs proposent une annotation plus fine, par exemple une valeur entre 0 et 1 (Rui & Huang, 2000). D'autres chercheurs proposent une interface graphique où les images sont disposées dans le plan selon leurs similarités (Rubner, 1999). Les images les plus proches de la requête sont présentées à l'utilisateur sous la forme d'une mosaïque bidimensionnelle. L'objectif est de traduire fidèlement la notion de similarité existant au sein de l'espace de recherche. L'utilisateur peut, à l'aide de la souris, entourer l'ensemble des images qu'il juge pertinentes. D'autres chercheurs proposent plusieurs formes de raffinement au sein de la même interface (Caenen et al., 2000). Trois fenêtres distinctes permettent à l'utilisateur d'interagir avec le système. Une première fenêtre affiche un échantillon de la base tiré aléatoirement, une deuxième affiche les images annotées dans la première, et une troisième affiche les images annotées dans un plan 2D. Dans cette dernière fenêtre, l'utilisateur peut déplacer les images de manière à rapprocher celles qui sont dans la même catégorie.

Une annotation plus fine que l'annotation binaire est discutable. En effet, il est difficile pour un utilisateur non expert du système d'évaluer précisement la pertinence d'une image. Comment dire si une image est à 60%, 80% ou 90% dans une catégorie particulière ? Le problème se pose également lorsqu'il faut déplacer des images sur un plan 2D afin de

révéler au système la similarité sémantique avec ses voisines. En outre, un argument fort en faveur des annotations binaires est leur succès dans le domaine de la recherche textuel. Il existe aujourd'hui de nombreuses approches très efficaces avec de solides fondements théoriques. En plus des techniques de classification, l'approche active de l'apprentissage permet aussi de proposer un affichage intéressant pour l'utilisateur.

4.2 Typologie des méthodes de bouclage de pertinence

4.2.1 Méthodes *ad hoc* issues de la recherche de documents

En recherche de documents, une stratégie consiste à s'intéresser au concept de *requête*. La requête est l'objet de la recherche, proposée par l'utilisateur. Par exemple, lorsque l'utilisateur fournit une image au système, elle constitue alors sa requête initiale. Le but du système est alors de modifier cette requête en fonction des annotations de l'utilisateur, de manière à retrouver ce qu'il recherche.

L'approche la plus simple pour la mise à jour de la requête consiste à calculer une nouvelle requête moyennant les signatures de l'ensemble des images pertinentes et de la requête initiale. Cette technique, souvent dénommée *query modification (QM)* dans la littérature, a été beaucoup utilisée en indexation de documents, et a été introduite plus récemment en indexation d'images (Rui et al., 1997).

L'autre façon d'aborder le problème consiste à adapter la fonction de similarité. Une heuristique est utilisée pour modifier les paramètres de la fonction de similarité, en général des coefficients sur les axes des attributs. Ces approches sont souvent rassemblées sous le terme de *query reweighting (QR)*. L'heuristique la plus répandue concerne la répartition, la dispersion ou la concentration des valeurs des attributs des images annotées. Par exemple, on peut considérer un axe comme pertinent si la variance des images pertinentes est faible sur cet axe (Rui et al., 1997). On peut aussi pondérer les axes en fonction du rapport entre l'écart type des images pertinentes et celui de toutes les images (Aksoy et al., 2000).

Une autre approche s'intéresse à la notion de rang : un attribut discriminant tend à classer les images pertinentes parmi les résultats les plus proches de la requête et inversement pour les images non pertinentes. (Heinrichs et al., 2000) proposent ainsi que les poids des attributs soient mis à jour à partir du rapport des rangs moyens calculés sur les ensembles d'images pertinentes et non pertinentes.

4.2.2 Méthodes basées optimisation

Les méthodes basées optimisation, au même titre que les méthodes *ad hoc* issues de la recherche de documents, s'intéressent au concept de requête, à laquelle elles ajoutent un critère mathématique à minimiser (Huang & Zhou, 2001).

Ces techniques considèrent la requête comme un point d'ancrage dans l'espace de recherche auquel une fonction de similarité permet de comparer n'importe quelle image. La pertinence d'une image est donc déterminée en fonction de la valeur renvoyée par la fonction de similarité, relativement à l'image requête.

Elles exploitent alors les annotations de l'utilisateur pour modifier les paramètres de cette fonction de similarité. Le problème est formulé par une optimisation des paramètres de la fonction sur un critère mathématique. Par exemple (Peng et al., 1999) mesurent la pertinence locale des attributs, estimée à partir d'un critère de réduction de l'erreur de classification bayésienne. Toujours dans le but d'optimiser la fonction de similarité, (Doulamis & Doulamis, 2001) minimisent l'erreur quadratique moyenne sur les images annotées, ou (Fournier et al., 2001a) rétropropagent l'erreur quadratique entre la similarité réelle et la similarité désirée.

4.2.3 Méthodes probabilistes

Dans le contexte des méthodes de recherche de cible, on trouve des techniques qui sont à l'origine d'autres techniques de recherche de catégories. Par exemple, l'approche qui consiste à estimer la probabilité $P(\mathbf{x} = \mathbf{x}^\star | H_t)$ pour chaque image $\mathbf{x}$ d'être l'image cible $\mathbf{x}^\star$, sachant les informations H_t fournies par l'utilisateur à l'itération t. L'utilisateur fournit en général des annotations relatives, *i.e.* "cette image est plus pertinente que celle-ci". L'approche repose sur la loi de Bayes, qui permet d'estimer la probabilité désirée en fonction des itérations précédentes.

Le système *PicHunter* (Cox et al., 2000) constitue le premier travail significatif sur les modèles bayesiens de bouclage de pertinence. Les hypothèses simplificatrices faites pour leur construction sont à l'origine de travaux complémentaires. Müller et al. (Müller et al., 1999) prennent en compte les changements de but de l'utilisateur au cours de la recherche. Selon eux, l'information renvoyée par le bouclage de pertinence est bruitée et parfois incohérente d'une itération à l'autre. La solution proposée consiste à pondérer les différents bouclages par un degré de confiance favorisant l'information la plus récente et la cohérence au sein des diverses comparaisons. Geman et Moquet (Geman & Moquet, 2000) considèrent le bouclage de pertinence plus comme un processus aléatoire que bruité. Ils remettent ainsi en cause l'existence et l'utilisation d'une métrique unique pour la comparaison des images. Selon eux, la métrique dépend de la cible et des images que le système propose à l'utilisateur. Le processus de modélisation de la pertinence repose sur une séquence de métriques indépendantes générées aléatoirement, correspondant à des pondérations différentes sur les attributs.

4.2.4 Méthodes par classification

La recherche de catégories peut aussi être vue comme un problème de classification binaire, *i.e.* à deux classes. La première classe est celle des images recherchées, appelée *classe pertinente*, et la deuxième classe est celle des images non recherchées, appelée

classe non pertinente. Le but est de construire une fonction capable de discriminer entre les images pertinentes et les images non pertinentes.

Compte tenu du fait que nous disposons d'exemples sous la forme d'annotations binaires, nous rentrons dans le cadre de la classification *supervisée*. Les méthodes supervisées s'appuient sur un *ensemble d'apprentissage* (les *exemples*, ici nos annotations) pour *entraîner* une fonction de classification (ou de discrimination). La fonction est utilisée pour déterminer la pertinence de chaque image de la base.

En recherche d'images, diverses techniques issues de l'apprentissage stastitique ont été proposées, comme la classification par critère de Bayes (Vasconcelos, 2000), les k-Plus Proches Voisins (Berrani et al., 2003), les Supports à Vaste Marge (SVM) (Chang et al., 2003; Chapelle et al., 1999; Tong & Koller, 2001; Saux, 2003, ...), ou encore les mélanges de gaussiennes (Najjar et al., 2003). Elles présentent des performances très intéressantes. D'une façon générale, plus on a d'exemples, plus il semble intéressant de travailler dans un contexte de classification.

4.3 Classification pour la recherche d'images

4.3.1 Définitions

La recherche d'images par classification binaire vise à construire une fonction de pertinence $f_{\mathbf{y}}$ sur un ensemble X en fonction d'un ensemble d'apprentissage $\mathbf{y}$. Dans notre cas, l'ensemble X des individus à classifier sont les n images de la base, chacune représentée par un vecteur $\mathbf{x}_i \in \mathbb{R}^p$.

Nous avons choisi d'utiliser une fonction de pertinence à valeurs dans $[-1, 1]$:

$$\begin{array}{rccc} f_{\mathbf{y}} : & X & \rightarrow & [-1,1] \\ & \mathbf{x}_i & \mapsto & f_{\mathbf{y}}(\mathbf{x}_i) \end{array}$$

Cela permet de distinguer trois cas différents : l'appartenance à la catégorie recherchée (proche de 1), la non appartenance à la catégorie recherchée (proche de -1), et l'incertitude (proche de 0). Ce formalisme est utile lors de l'étape de sélection pour l'apprentissage actif.

L'ensemble d'apprentissage est formé par les couples $(\mathbf{x}_i, y_i)$, où $\mathbf{x}_i$ est l'image d'indice i et y_i est l'annotation de l'image i :

$$y_i = \begin{cases} 1 & \text{si } \mathbf{x}_i \text{ est pertinente} \\ -1 & \text{si } \mathbf{x}_i \text{ est non pertinente} \\ 0 & \text{s'il n'y a pas d'annotation} \end{cases}$$

D'une manière générale, nous notons par $c = 1$ la classe des images recherchées ou *classe pertinente*, et par $c = -1$ la classe des images non recherchées ou *classe non pertinente*. Nous dirons qu'une image est *pertinente* si elle appartient à la catégorie recherchée, et qu'elle *non pertinente* dans le cas contraire.

Le processus de classification s'opère en deux temps :

1. *Entraînement*. L'entraînement permet la construction de la fonction de pertinence $f_{\mathbf{y}}$, en s'appuyant sur l'ensemble d'apprentissage $\mathbf{y}$.
2. *Calcul des pertinences*. La pertinence $f_{\mathbf{y}}(\mathbf{x}_i)$ de toutes les images de la base est calculée.

Nous employons aussi les écritures suivantes :

$$\begin{aligned} I &= \text{ensemble des indices des images annotées} \\ &= \{i \in [1..n] \mid y_i \neq 0\} \\ I_c &= \text{ensemble des indices des images annotées de la classe } c \\ &= \{i \in [1..n] \mid y_i = c\} \\ n_c &= \text{nombre d'images annotées de la classe } c \\ &= |I_c| \end{aligned}$$

Parmi les techniques de classification, nous pouvons distinguer plusieurs types d'approches :

- Classification *inductive* ou *supervisée*. Ce type de classification s'appuie uniquement sur les individus annotés pour l'entraînement.
- Classification *transductive* ou *semi-supervisée*. Ce type de classification s'appuie à la fois sur les individus annotés et non annotés.
- Classification *active*. Ce type de classification est une forme particulière de classification semi-supervisée, qui s'intègre dans un processus d'apprentissage dit *actif*. Elle permet d'intervenir dans le processus de construction de l'ensemble d'apprentissage dans le cadre d'une recherche interactive.

4.3.2 Risque

Nous présentons dans ce paragraphe une formalisation du but de la classification. Notre but premier est avant tout la satisfaction de l'utilisateur, qui est fonction du classement de la base. Les classifieurs ne sont pas construits dans ce but, mais une bonne classification conduit toutefois à un meilleur classement de la base. Il nous semble donc important de bien situer les objectifs formels de la classification dans le cadre de cette étude.

Le but de la classification binaire est de trouver la fonction $f(\mathbf{x})$ qui classifie au mieux tout individu de l'espace d'entrée (dans notre cas $\mathbb{R}^p$). Cela s'exprime par la minimisation du *risque réel* ou *espérance de l'erreur de classification* ou *erreur de généralisation*. Ce

risque est la somme des erreurs $L(f(\mathbf{x}), y)$ de classification entre l'annotation estimée $f(\mathbf{x})$ et la véritable annotation y sur tous les couples $(\mathbf{x}, y)$:

$$R_{\text{réel}}(f) = \int_{\mathbb{R}^p \times \{\pm 1\}} L(f(\mathbf{x}), y) dP(\mathbf{x}, y) \tag{4.1}$$

Minimiser le risque réel fait partie des problèmes de l'*induction* (Schölkopf & Smola, 2002).

La fonction $L(f(\mathbf{x}), y)$ est une fonction de perte qui évalue l'erreur entre la prédiction $f(\mathbf{x})$ et la classe y. Par exemple, on peut choisir la fonction de perte tout ou rien, $L(f(\mathbf{x}), y) = 1 - \delta(f(\mathbf{x}), y)$.

Lorsque des individus à classifier sont disponibles, par exemple l'ensemble $\bar{I}$ des images non annotées de la base d'images, on peut définir un risque sur l'ensemble de ces individus. On parle alors d'*erreur de classification sur* X ou d'*erreur de test sur* X ou *erreur de généralisation sur* X :

$$R_{\text{test}}(f) = \frac{1}{|\bar{I}|} \sum_{i \in \bar{I}} \sum_{c \in \{-1,1\}} L(f(\mathbf{x}_i), c) P(c|\mathbf{x}_i) \tag{4.2}$$

Minimiser le risque test fait partie des problèmes de la *transduction* (Schölkopf & Smola, 2002).

Le calcul de l'erreur de classification sur X se simplifie si l'on connaît la classe c_i de chaque individu $\mathbf{x}_i$. Par exemple, si $P(c = c_i|\mathbf{x}_i) = 1$, le calcul devient :

$$R_{\text{test}}(f) = \frac{1}{|\bar{I}|} \sum_{i \in \bar{I}} L(f(\mathbf{x}_i), c_i) \tag{4.3}$$

4.3.3 Particularités de la recherche d'images

Les méthodes de classification auxquelles nous nous intéressons ne sont applicables que si les deux types d'annotations sont disponibles (pertinents et non pertinents). Dans le cas contraire, nous utilisons une technique pour estimer la densité d'une classe. Par exemple, en début de session de recherche, il est courant de ne disposer que d'annotations positives. Dans ce cas, nous estimons la densité de la classe pertinente.

Dans tous les cas, le but est d'obtenir une fonction $f_{\mathbf{y}}$ d'appartenance à la catégorie recherchée, ou *fonction de pertinence*. Pour les estimateurs de densité, la fonction de pertinence est la probabilité d'appartenance. Pour les classifieurs, certaines méthodes sont initialement proposées pour discriminer, et requièrent une adaptation pour permettre un calcul d'appartenance à la catégorie recherchée. C'est la raison pour laquelle toutes les méthodes présentées auront pour expression finale le calcul de la fonction de pertinence.

La classification binaire pour la recherche interactive d'images est caractérisée par les points suivants, dont certains ont été relevées par (Chang et al., 2003) :

1. *Grande dimension.* Les vecteurs qui représentent les images sont souvent de grande dimension (de 10 à parfois plus de 1000). Il est important de traiter ce problème si l'on ne veut pas souffrir des effets de la « malédiction de la dimension ».
2. *Classes complexes.* La recherche étant totalement libre, il est impossible de faire des hypothèses fortes sur la structure des catégories recherchées. Celles-ci peuvent être aussi bien concentrées dans une zone de l'espace que réparties en de nombreux petits amas.
3. *Déséquilibre pertinent/non pertinent.* La taille de la catégorie recherchée est généralement très petite devant la taille de la base.
4. *Peu d'annotations.* Au démarrage d'une nouvelle recherche, le système doit fournir des résultats avec très peu de données. De plus, le système ne peut demander à l'utilisateur qu'un nombre limité d'annotations. Par exemple, on ne peut pas envisager de faire annoter l'utilisateur sur la moitié de la base.
5. *Apprentissage interactif.* L'ensemble d'apprentissage s'obtient morceau par morceau au cours des itérations. Ainsi tout résultat à une itération donnée dépend des précédents.
6. *Satisfaction de l'utilisateur.* La qualité d'un système de recherche de catégorie d'images est avant tout jugée par ses utilisateurs. Afin de modéliser cette satisfaction, on peut utiliser un critère particulier, comme la Précision Moyenne que nous présentons à la fin de ce chapitre.
7. *Rapidité.* Le but est de concevoir un système qui peut être utilisé rapidement et efficacement. Ainsi, on préférera une technique très peu coûteuse en calculs : il est difficile d'imaginer faire attendre plusieurs minutes entre chaque itération. En ce sens, nous nous astreignons à des techniques $O(n)$, sauf cas particulier d'étude.

Nous avons choisi de traiter les problèmes *Grande dimension* et *Classes complexes* par l'utilisation d'une fonction noyau. En effet, comme nous l'avons présenté au chapitre 3, cette approche permet d'injecter l'espace des signatures dans un espace linéaire aux propriétés intéressantes. Hormis les cas où ce n'est pas possible, toutes les méthodes présentées ont fait l'objet d'une « noyautisation »(*kernelization*) en utilisant le *truc du noyau*. Le truc du noyau, qui est plus amplement détaillé au chapitre 3, consiste à écrire un algorithme exclusivement à l'aide d'un produit scalaire $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$, puis de remplacer tous les produits scalaires par une fonction noyau $k(\mathbf{x}_i, \mathbf{x}_j)$.

4.4 Sélection de méthodes de classification supervisées pour la recherche d'images

Nous présentons dans cette section les techniques de classification supervisées les plus utilisées dans la communauté de recherche d'images, ainsi que des techniques qui nous semble, d'un point de vue théorique, adaptées au contexte :

– Critère de Bayes (Vasconcelos, 2000) ;
– Discriminant de Fisher (Mika et al., 1999) ;
– Support à Vaste Marge (Tong & Koller, 2001) ;
– k-Plus Proches Voisins (Berrani et al., 2003) ;
Nous avons implanté toutes ces méthodes au sein du système RETIN 2.

4.4.1 Méthode bayesienne

Cette méthode estime la densité $P_{\mathbf{y}}(\mathbf{x}|c)$ de chacune des deux classes ($c = 1$ ou $c = -1$), indépendamment l'une de l'autre. Ensuite, le critère de Bayes est appliqué afin de déterminer la classe de chaque individu (Vasconcelos, 2000) :

$$f_{\mathbf{y}}(\mathbf{x}) = \underset{c\in\{-1,1\}}{\operatorname{argmax}} P_{\mathbf{y}}(\mathbf{x}|c)P(c)$$

où $P_{\mathbf{y}}(\mathbf{x}|c)$ est la probabilité pour $\mathbf{x}$ d'appartenir à la classe c, et $P(c)$ la probabilité *a priori* de la classe c. En l'absence d'*a priori*, on pose $P(c) = \frac{1}{2}$.

Les méthodes d'estimation de densités que nous avons étudiées sont présentées en §4.5.

4.4.2 Discriminant de Fisher

Le Discriminant de Fisher est une méthode qui, au même titre que la précédente, s'intéresse plus particulièrement aux cœurs des classes (Mika et al., 1999). En revanche, le calcul des deux densités ne se fait pas indépendamment. Cette méthode s'intéresse à la moyenne et la variance de chaque classe en fonction d'un hyperplan. Cette méthode nous semble intéressante à étudier car elle est plus orientée vers les cœurs des classes, puisque nous recherchons une bonne estimation de la fonction de pertinence. De plus cette technique utilise un classifieur simple (l'hyperplan).

En effet, si nous considérons tous les hyperplans qui coupent la base en deux, alors, pour chacun d'entre eux, il est possible d'assigner une des deux classes à chaque individu. Nous sommes alors en mesure de calculer la moyenne et la variance de chaque classe. Le critère du Discriminant de Fisher pour distinguer l'hyperplan optimal de tous les autres est de choisir celui qui offre la plus petite variance pour chaque classe (chaque classe est bien dense) et la plus grande distance entre les deux moyennes (les deux classes sont bien séparées).

Mathématiquement, son but est de trouver la direction $\mathbf{w}$ de projection optimale telle que la distance entre les deux moyennes des deux classes projetées soit maximale alors que la variance de chaque classe est minimale. Ceci peut s'exprimer par la maximisation du coefficient de Rayleigh suivant :

$$J(\mathbf{w}) = \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}}$$

avec :

$$\begin{aligned}\mathbf{S}_B &= (\mathbf{m}_1 - \mathbf{m}_{-1})(\mathbf{m}_1 - \mathbf{m}_{-1})^\top \\ \mathbf{S}_W &= \sum_{c\in\{1,-1\}} \sum_{i\in I} (\mathbf{x}_i - \mathbf{m}_c)(\mathbf{x}_i - \mathbf{m}_c)^\top \\ \mathbf{m}_c &= \frac{1}{n_c} \sum_{i\in I_c} \mathbf{x}_i,\ n_c = |I_c|\end{aligned}$$

$\mathbf{S}_B$ et $\mathbf{S}_W$ sont respectivement les matrices d'inter- et d'intra-variance, et $\mathbf{m}_c$ est la moyenne de chaque classe. Le numérateur de $J(\mathbf{w})$ est la moyenne des classes projetées dans la direction $\mathbf{w}^\top$, et le dénominateur de $J(\mathbf{w})$ est la variance des classes dans la direction $\mathbf{w}$.

(Mika et al., 1999) proposent une version de cette optimisation dans un espace induit par le biais d'une fonction noyau. Le problème devient alors :

$$J(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^\top \mathbf{M} \boldsymbol{\alpha}}{\boldsymbol{\alpha}^\top \mathbf{N} \boldsymbol{\alpha}}$$

avec :

$$\begin{aligned}\mathbf{M} &= (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_{-1})(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_{-1})^\top \\ \mathbf{N} &= \mathbf{K}_I (\mathbf{K}_I)^\top - \sum_{c\in\{1,-1\}} n_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^\top \\ \mu_{jc} &= \frac{1}{n_c} \sum_{i\in I_c} k(\mathbf{x}_i, \mathbf{x}_j),\ n_c = |I_c| \\ \mathbf{K}_I &= \text{matrice de Gram sur les images annotées}\end{aligned}$$

La maximisation de $J(\boldsymbol{\alpha})$ peut se faire en calculant le vecteur propre correspondant à la plus grande valeur propre de la matrice $\mathbf{N}^{-1}\mathbf{M}$. Ce calcul peut être effectué très rapidement avec une itération de puissances, un algorithme classique du calcul matriciel (Golub & Van Loan, 1996). De plus, cette méthode ne souffre pas de problèmes de minima locaux.

Finalement, la fonction de pertinence est la projection de chaque point sur la direction optimale :

$$f_{\mathbf{y}}(\mathbf{x}) = \sum_{i\in I} \alpha_i k(\mathbf{x}, \mathbf{x}_i)$$

4.4.3 Supports à Vaste Marge

Les Supports à Vaste Marge (SVM) sont une technique de classification par hyperplan qui a été introduite dans la communauté d'apprentissage statistique (Vapnik, 1999).

Les classifieurs à hyperplan utilisent la projection sur la normale $\mathbf{w}$ d'un hyperplan comme fonction de discrimination :

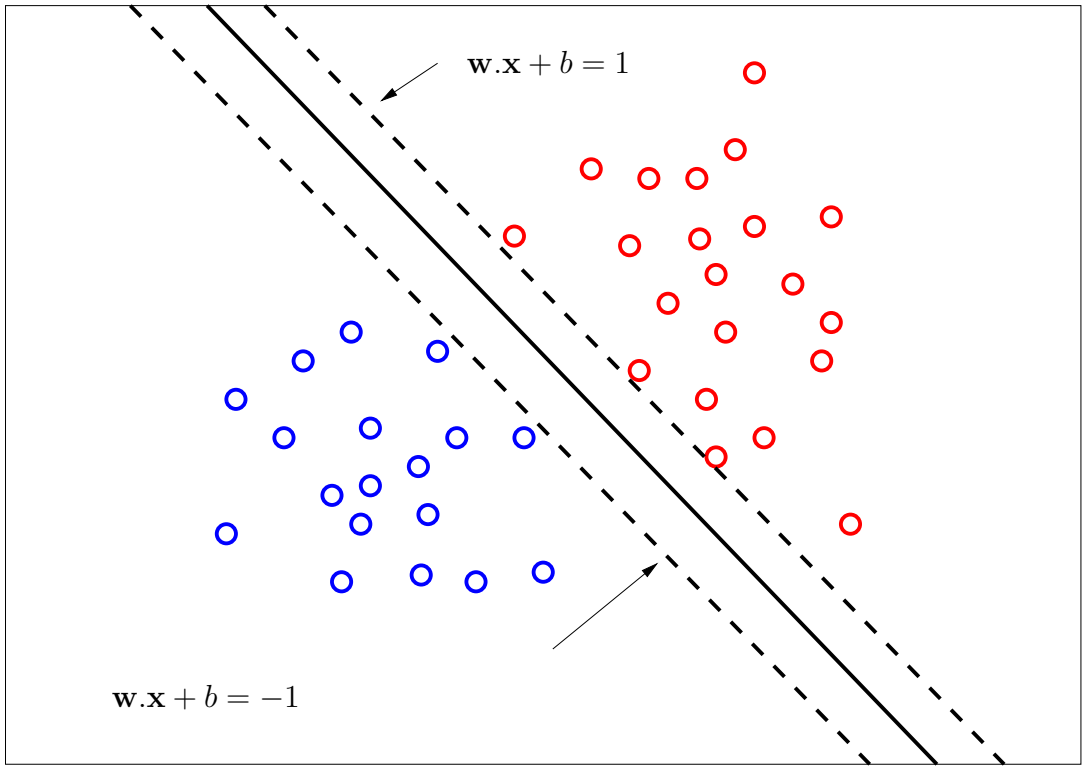


Fig. 4.2 – Marge des SVM

$$f_{\mathbf{y}}(\mathbf{x}_i) = \langle \mathbf{w}, \mathbf{x}_i \rangle + b$$

L'objectif est de trouver une fonction de discrimination telle que tous les éléments de même classe soient du même côté de l'hyperplan. Autrement dit, il faut trouver un vecteur $\mathbf{w}$ et un réel b tel que :

$$\forall i \in I \quad (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i > 0$$

Dans un cas séparable, il existe de nombreux hyperplans vérifiant cette équation. Les SVM se distinguent des autres techniques par un choix particulier d'hyperplan séparateur. L'approche choisit parmi tous les hyperplans séparateurs celui qui maximise la marge, *i.e.* tel que la plus petite distance entre les points et l'hyperplan soit maximale. Par exemple, la zone entre les deux lignes pointillées sur la figure 4.2.

Un tel hyperplan, en plus d'être unique, a de nombreux avantages sur le plan théorique (Vapnik, 1999). La fonction de décision étant invariante par changement d'échelle, on choisit de trouver l'hyperplan tel que $\mathbf{w}.\mathbf{x} + b = \pm 1$ pour les éléments les plus proches de la marge (cf figure 4.2), ce qui revient à minimiser $||\mathbf{w}||^2$ tel que :

$$\forall i \in I \quad y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$

En utilisant les Lagrangiens, le problème devient :

$$\underset{\boldsymbol{\alpha}}{\text{argmax}}\ W(\boldsymbol{\alpha}) = \sum_{i \in I} \alpha_i - \frac{1}{2} \sum_{i,j \in I^2} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

avec :

$$\sum_{i \in I} \alpha_i y_i = 0 \quad \text{et} \quad \alpha_i \geq 0,\ \forall i \in I$$

La fonction de pertinence s'écrit alors :

$$f_{\mathbf{y}}(\mathbf{x}) = \sum_{i \in I} y_i \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b$$

Cette première méthode suppose que les données sont linéairement séparables. Afin d'assouplir la discrimination, une marge souple peut être introduite, en acceptant la mauvaise classification de certains éléments. Ceci revient à majorer chacun des α_i par une constante C (Veropoulos, 1999). De plus, on peut facilement utiliser une fonction noyau avec cette méthode, qui est à l'origine de l'essort des ces fonctions.

Puisque la méthode s'écrit avec un produit scalaire $\langle \mathbf{x}, \mathbf{x}' \rangle$, on peut utiliser le truc du noyau :

$$\underset{\boldsymbol{\alpha}}{\text{argmax}}\ W(\boldsymbol{\alpha}) = \sum_{i \in I} \alpha_i - \frac{1}{2} \sum_{i,j \in I^2} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

avec :

$$\sum_{i \in I} \alpha_i y_i = 0 \quad \text{et} \quad 0 \leq \alpha_i \leq C,\ \forall i \in I$$

La fonction de pertinence est alors :

$$f_{\mathbf{y}}(\mathbf{x}) = \sum_{i \in I} \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i)$$

4.4.4 k-Plus Proches Voisins

Les k-Plus Proches Voisins sont une technique de classification très utilisée dans un nombre varié d'applications et qui présente souvent de bonnes performances.

Les kPPV déterminent l'appartenance d'un élément à une classe en fonction de ses k plus proches voisins[1]. Nous utilisons dans notre contexte une version avec noyau (Hastie et al., 2001a, Chapitre 6) :

$$f_{\mathbf{y}}(\mathbf{x}) = \frac{\sum_{i \in \mathrm{nn}_t(\mathbf{x})} y_i k(\mathbf{x}, \mathbf{x}_i)}{\sum_{i \in \mathrm{nn}_t(\mathbf{x})} k(\mathbf{x}, \mathbf{x}_i)}$$

avec $\mathrm{nn}_t(\mathbf{x})$ une fonction qui à un vecteur $\mathbf{x}$ fait correspondre l'ensemble des indices des t vecteurs les plus proches d'annotation non nulle.

4.4.5 Mélange de gaussiennes (EMiner)

Cette technique modélise chaque classe à l'aide d'un mélange de gaussiennes, en prenant compte à la fois des données annotées et non annotées (Najjar et al., 2003). Cette méthode se particularise par une détermination simultanée des paramètres des deux mélanges.

La méthode utilise un mélange de gaussienness g_r de moyenne $\boldsymbol{\mu}_r$ et de variance σ_r, avec $r \in R : g_r(\mathbf{x}) = e^{-\frac{(\mathbf{x}-\boldsymbol{\mu}_r)^2}{2\sigma_r^2}}$.

Chacune de ces gaussiennes est associée à l'une des deux classes : celles d'indices r dans R_1 sont associées à la classe pertinente, et celles d'indices r dans R_{-1} sont associées à la classe non pertinente. En sélectionnant les indices d'une des deux classes, on en déduit la probabilité d'appartenance de chaque image à une classe.

Cette technique se particularise par l'utilisation d'une variable supplémentaire $\mathbf{z}_i$ donnant la dimension semi-supervisée de la méthode. Il est défini comme suit :

- Pour i tel que $y_i = c$: $z_{ir} = \begin{cases} 1 & \text{si } r \in R_c \\ 0 & \text{sinon} \end{cases}$
- Pour i tel que $y_i = 0$: $z_{ir} = 1$

Il permet de sélectionner les gaussiennes de la classe c si l'élément annoté $\mathbf{x}_i$ appartient à la classe c. Pour les éléments $\mathbf{x}_i$ non annotés, il sélectionne toutes les gaussiennes.

La probabilité d'appartenance d'image $\mathbf{x}_i$ à la classe c est donnée par :

$$P_{\mathbf{y}}(\mathbf{x}_i|c) = \sum_{r \in R_c} p(r|x_i, z_i)$$

avec :

[1]La lettre k est ici le nombre de voisins, à ne pas confondre avec la fonction noyau $k(.,.)$

$$p(r|x_i, z_i) = \frac{z_{ir}\pi_r g_r(\mathbf{x}_i)}{\sum_{s\in R} z_{is}\pi_s g_s(\mathbf{x}_i)}$$

La fonction de pertinence est le rapport entre la probabilité de pertinence et la probabilité de non pertinence :

$$f_{\mathbf{y}}(\mathbf{x}_i) = \frac{P_{\mathbf{y}}(\mathbf{x}_i|1)}{P_{\mathbf{y}}(\mathbf{x}_i|-1)}$$

La détermination des paramètres ($\boldsymbol{\mu}_r$,σ_r) se fait par un algorithme Expectation-Maximisation.

4.4.6 SVM transductif

Les SVM transductif est, au même titre que les SVM, une technique de maximisation de la marge de l'hyperplan séparateur (Joachims, 1999). Cette technique se distingue de la version simplement supervisée par le fait qu'elle tente de trouver l'hyperplan qui sépare toutes les données, annotées et non annotées. Pour ce faire, il faut bien entendu disposer d'une annotation pour les données non annotées. L'approche consiste à sélectionner les annotations sur chaque élément de $\bar{I}$ qui vont fournir la plus vaste marge.

Cela revient à considérer un problème de SVM augmenté des variables $\mathbf{y}_i^\star$, $i \in \bar{I}$, soit les annotations des images non annotées. Autrement dit, il faut minimiser $||\mathbf{w}||^2$ sur $(\mathbf{y}^\star, \mathbf{w}, b)$ tel que :

$$y_i(< \mathbf{w}, \mathbf{x}_i > +b) \geq 1, \ \forall i \in I$$
$$y_i^\star(< \mathbf{w}, \mathbf{x}_i > +b) \geq 1, \ \forall i \in \bar{I}$$

Résoudre ce problème revient à trouver des annotations $\mathbf{y}^\star$ et un hyperplan $(\mathbf{w}, b)$ tels que l'hyperplan sépare les deux classes avec la plus grande marge.

Au même titre que les SVM inductifs, on peut assouplir la marge en introduisant une majoration paramétrée par une constante C. Entraîner un SVM transductif revient à résoudre un problème d'optimisation combinatoire. Pour un faible nombre de données non annotées, on peut tenter toutes les possibilités. Cependant, cela devient rapidement impossible en pratique. Joachims (Joachims, 1999) propose un algorithme sous optimal mais rapide. Cet algorithme commence par entraîner un SVM inductif, puis assigne une annotation pour chaque élément de $\bar{I}$ à partir de cette première classification. L'algorithme initialise une marge très souple (*i.e.* C très petit). Ensuite l'algorithme prend au hasard deux points d'annotations opposées qui ne respectent pas les contraintes, inverse leurs annotations, et re-entraîne le classifieur SVM avec toutes les données. Le processus est répété plusieurs fois, tout en prenant une marge de plus en plus dure. L'algorithme s'arrête lorsque C a atteint la valeur souhaitée par l'utilisateur.

Notons que cette technique requiert un paramètre supplémentaire, le nombre d'images dans la catégorie recherchée, qui est particulièrement difficile à estimer dans notre contexte.

Finalement la fonction de pertinence est donnée par :

$$f_{\mathbf{y}}(\mathbf{x}_i) = y_i^\star$$

4.5 Méthodes d'estimation de densité

Nous présentons dans cette section des techniques pour estimer la densité d'une classe, soit pour gérer le cas où seul un type d'annotation est présent, soit pour la classification bayesienne. L'objectif de ces techniques est d'estimer la probabilité $P(\mathbf{x}|c)$ d'une image $\mathbf{x}$ d'appartenir à la classe c, $c = 1$ pour la catégorie recherchée, et $c = -1$ pour les images non recherchées.

Nous avons aussi sélectionné des techniques utilisées en recherche d'images :
– Fenêtres de Parzen ;
– SVM à une classe (Chen et al., 2001).

4.5.1 Fenêtres de Parzen

Les fenêtres de Parzen permettent d'estimer simplement une densité, en modélisant la classe par une somme de gaussiennes :

$$P(\mathbf{x}|c) = \frac{1}{n_c} \sum_{i \in I_c} e^{-\frac{d(\mathbf{x},\mathbf{x}_i)^2}{2\sigma^2}}$$

Etant donné que nous utilisons surtout les noyaux gaussiens, cette méthode peut être utilisée avec une fonction noyau k :

$$P(\mathbf{x}|c) = \frac{1}{n_c} \sum_{i \in I_c} k(\mathbf{x}, \mathbf{x}_i)$$

4.5.2 Supports à Vaste Marge à une classe

Une approche similaire aux Supports à Vaste Marge (SVM) peut être utilisée pour estimer la densité d'un ensemble (Chen et al., 2001). Dans ce cas, ils sont dit *à une classe*. L'idée est de travailler avec des données distribuées sur une hypersphère[2] et de trouver l'hyperplan qui sépare le plus l'origine des individus. L'origine peut être vue, en quelque sorte, comme la deuxième classe.

[2]Notons qu'il est toutefois possible de ne pas travailler sur une hypersphère (Schölkopf et al., 2000).

Si la fonction de discrimination a pour expression :

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$$

alors le but est de maximiser $||\mathbf{w}||$ (la distance de l'hyperplan à l'origine) avec la contrainte que $f(\mathbf{x}_i) \geq 1$ sur I.

Le problème est alors pratiquement le même que celui des SVM à deux classes, ce qui conduit aux mêmes extensions, telles que la marge souple et l'utilisation des fonctions noyaux (*cf.* §4.4.3).

La fonction densité a alors pour forme :

$$P_{\mathbf{y}}(\mathbf{x}|c) = \sum_{i \in I_c} \alpha_{ic} k(\mathbf{x}, \mathbf{x}_i)$$

Le vecteur $\boldsymbol{\alpha}_c$ est déterminé en résolvant un problème d'optimisation quadratique pour chaque classe c.

4.6 Comparaison Expérimentale

Les techniques présentées ci-dessus ont été évaluées sur la base COREL décrite en annexe A. Les signatures sont des histrogrammes de 25 couleurs et 25 textures, qui ont été réduits par un processus de quantification vectorielle (*cf.* chapitre 2). La fonction noyau utilisée est une gaussienne avec une distance du χ^2 (*cf.* chapitre 3). Les expérimentations suivantes évaluent les performances globales du système sur toutes les catégories de la base. Pour un nombre m d'annotations donné, nous calculons 1000 classifications de la base avec l'une des méthodes présentées. Pour chacune de ces classifications, nous choisissons aléatoirement l'une des catégories, puis construisons aléatoirement un ensemble d'apprentissage de taille m pour la catégorie choisie. Les ensembles d'apprentissages ne sont pas nécessairement équilibrés, et la proportion d'annotations positives et négatives est la même que celle des images de la catégories dans la base. En général, les catégories ont 300 images au sein de cette base de 6000 images, et les ensembles d'apprentissage ont en moyenne 5% d'annotations positives. Suite à chaque classification de la base, nous déterminons la Précision Moyenne, le Top 100, l'erreur de classification et le temps de calcul (*cf.* annexe A). Toutes ces valeurs sont ensuite moyennées sur les 1000 classifications pour la même méthode et la même taille d'ensemble d'apprentissage.

Notons que pour chaque méthode de classification nous réalisons au total 20.000 simulations, étant donné que les ensembles d'apprentissages ont de 10 à 200 exemples par pas de 10. Nous avons fixé le nombre maximal d'annotations à 200 compte tenu de notre contexte, où le nombre d'annotations qu'un utilisateur est prêt à donner est faible devant la taille de la base. Le nombre de 200 annotations reste toutefois élevé, mais nous souhaitions évaluer le comportement des méthodes lorsque le nombre d'annotations est

très grand pour un utilisateur, dans l'éventualité d'une discontinuité importante dans l'évolution des performances en fonction de la taille de l'ensemble d'apprentissage. De plus, 1000 simulations par méthode et par taille d'ensemble d'apprentissage est un minimum pour obtenir des résultat stables. En dessous de ce nombre, les résultats d'une expérimentation à l'autre dans les mêmes conditions peuvent être variables.

Les méthodes suivantes ont été évaluées :

- Support à Vaste Marge ou SVM (section 4.4.3) ;
- Discriminant de Fisher ou KFD (section 4.4.2) ;
- Critère de Bayes (section 4.4.1) avec des fenêtres de Parzen (section 4.5.1) ;
- k-Plus Proches Voisins ou kPPV avec $k = 5$ (section 4.4.4) ;
- Raffinement de Similarité ou RS (Fournier et al., 2001b) ;
- Mélange de gaussiennes semi-supervisé ou EMiner (section 4.4.5).
- Support à Vaste Marge Transductifs ou TSVM (section 4.4.6) ;

Les méthodes RS et EMiner ne sont pas utilisées avec la fonction noyau.

Les SVM Transductifs (section 4.4.6) ne sont pas affichés pour des raisons de lisibilité : cette méthode apporte exactement les mêmes résultats qu'un SVM, mais avec un temps de calcul beaucoup plus important. Lorsqu'il n'y a qu'un seul type d'annotation, la densité de la classe est estimée avec un SVM à une classe (section 4.5.2).

4.6.1 Précision Moyenne et Top 100

Selon la Précision Moyenne (*cf* Fig. 4.3), nous pouvons voir que les SVM, le KFD, et Bayes sont les trois méthodes en tête, suivies de près par les kPPV. On peut voir que la méthode de Raffinement de la Similarité et la méthode semi-supervisée EMiner donnent de moins bons résultats, et n'augmentent pas aussi rapidement leurs performances que les autres. Ceci peut s'expliquer par le fait que ces méthodes ne permettent pas l'utilisation de fonctions noyaux, ce qui les pénalisent dans ce cas précis. Le comportement en terme de Top 100 (*cf* Fig. 4.4) est très similaire, à la différence que les kPPV font partie du peloton de tête.

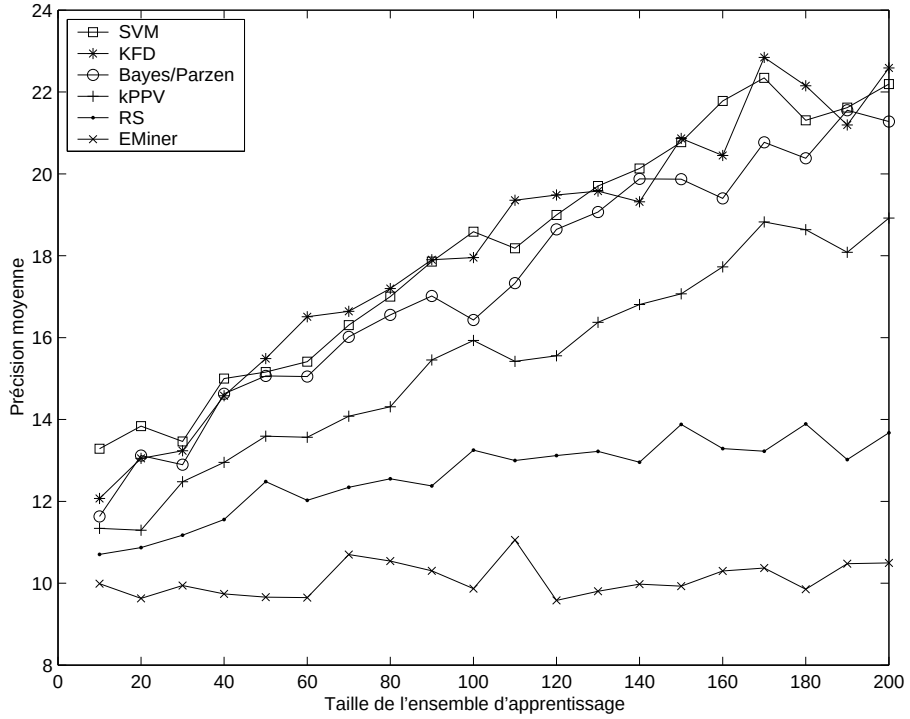


FIG. 4.3 – Précision Moyenne (en pourcentages) en fonction de la taille de l'ensemble d'apprentissage, pour chaque méthode de classification.

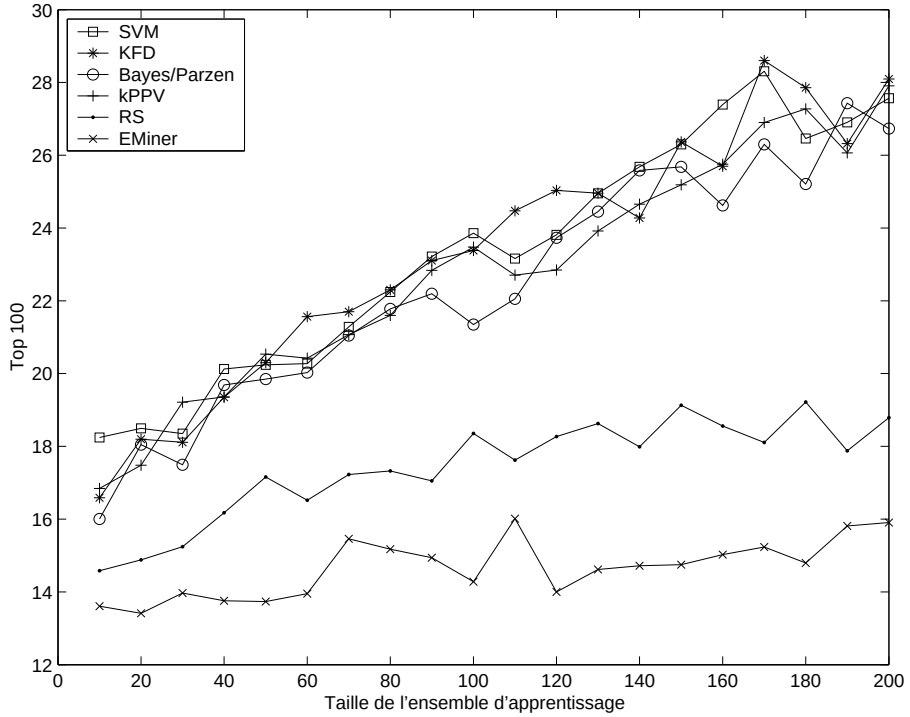


FIG. 4.4 – Top 100 (en pourcentages) en fonction de la taille de l'ensemble d'apprentissage, pour chaque méthode de classification.

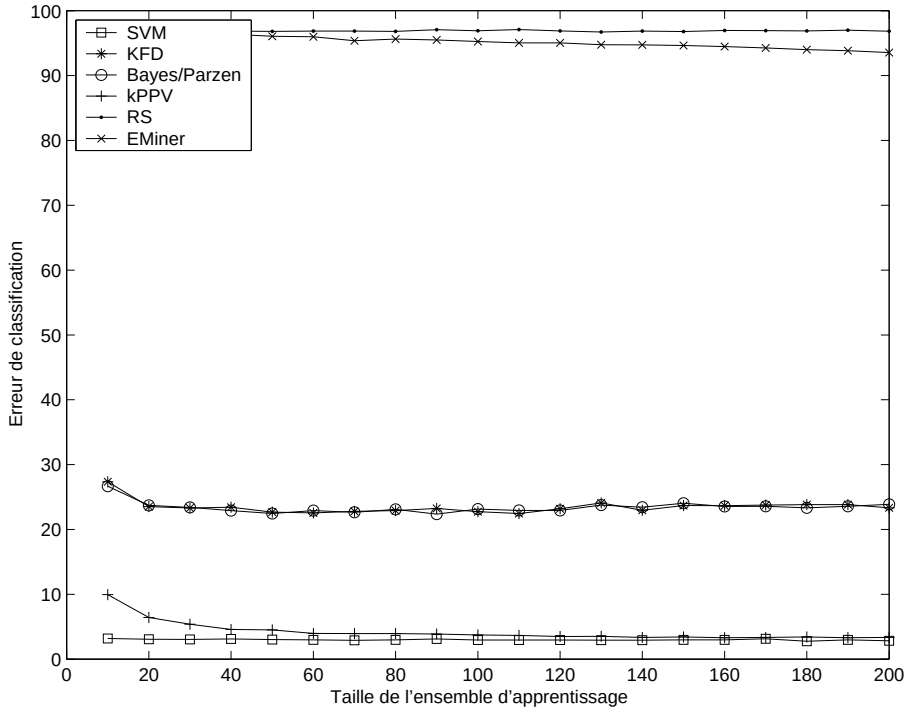


FIG. 4.5 – Erreur de classification (en pourcentages) en fonction de la taille de l'ensemble d'apprentissage, pour chaque méthode de classification.

4.6.2 Erreur de classification

D'un point de vue classification pure (*cf* Fig. 4.5), on peut voir 3 groupes de méthodes. Le premier et le moins performant, est formé par le Raffinement de Similarité et EMiner. Ces performances assez catastrophiques (plus de 95% d'erreur), peuvent aussi s'expliquer par le fait qu'aucune fonction noyau adaptée ne peut être utilisée. Dans ce cas précis, on peut supposer que l'espace des signatures est de trop faible dimension, ce qui pousse ces classifieurs à trop généraliser. Le deuxième groupe est formé par Bayes et KFD, qui offrent des performances intermédiaires, et le troisième groupe est formé par les SVM et les kPPV, qui offrent les meilleures performances. Ce résultat peut s'expliquer par le fait que ces deux dernières méthodes ont une capacité à s'adapter aux problèmes de densité, les kPPV de par leur nature même, et les SVM par un réglage automatisé de la souplesse de la marge.

On peut toutefois noter que l'erreur de classification n'est pas un bon critère pour comparer les performances des méthodes. En effet, il y a tout d'abord des résultats assez inattendus, comme pour EMiner qui donne plus de 95% d'erreur. De plus, l'erreur ne diminue pas ou peu lorsque l'ensemble d'apprentissage augmente. Cela est le fruit du deséquilibre qui existe entre la taille de la classe pertinente et la taille de la classe non pertinente. Une faible variation dans les paramètres d'un classifieur peut suffir à modifier la classe de centaines d'images, par exemple un faible déplacement de la frontière.

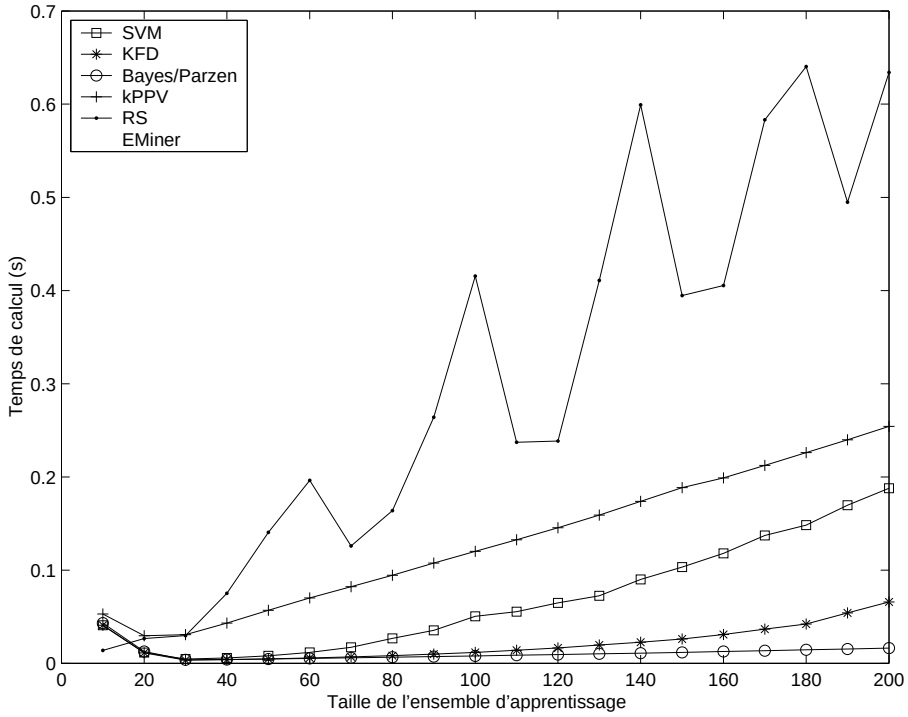


FIG. 4.6 – Temps de calcul (en secondes) en fonction de la taille de l'ensemble d'apprentissage, pour chaque méthode de classification.

4.6.3 Temps de calcul

Le dernier graphique de la figure 4.6 présente le temps de calcul nécessaire pour l'entraînement du classifieur et le classement de toutes les images. Comme EMiner requiert 100 fois plus de temps que les autres, elle n'a pas été représentée sur la figure. Cette méthode requiert environ 25 secondes de temps de calcul pour un ensemble d'apprentissage de 200 exemples. Toutes les méthodes voient leur temps de calcul augmenter régulièrement avec la taille de l'ensemble d'apprentissage, hormis RS (Raffinement de Similarité). Ceci peut s'expliquer par le fait que cette méthode effectue une descente de gradient, dont le nombre d'itérations peut varier.

4.6.4 Conclusion

En terme de Précision moyenne, ces résultats expérimentaux montrent la puissance qu'offrent les techniques de classification actuelles, tout particulièrement si on a la possiblité de les utiliser avec une fonction noyau adaptée. En se restreignant à une évaluation par précision/rappel, aucune méthode parmis Bayes, kPPV, KFD et SVM n'est plus intéressante qu'une autre. D'autre part, bien que l'erreur de classification ne soit pas un bon critère pour l'évaluation d'un système de recherche d'images, elle joue un rôle important pour les techniques de classification active, comme nous allons le voir dans le chapitre suivant. Sur ce plan, les kPPV et les SVM sont donc les méthodes les plus intéressantes.

Enfin, si l'on prends en compte le temps de calcul, la technique la plus intéressante sont les SVM, compte tenu des résultats obtenus et du protocole expérimental suivi. C'est la raison pour laquelle, dans les chapitres suivants, nous n'utiliserons plus que ce type de classification, et nous nous concentrerons sur les autres aspects du problème de la recherche interactive.

Chapitre 5

Classification active pour la recherche d'images

La classification dite *active* est une extension de la classification semi-supervisée. En plus d'exploiter les données non annotées, elle propose à l'utilisateur un choix particulier d'images à annoter. Elle se distingue de la classification simple par l'ajout d'une étape dite de *sélection*. Cette étape permet la sélection d'une image ou d'un lot d'images que l'utilisateur devra annoter. Cette approche est particulièrement intéressante en recherche interactive d'images de part le fait que seul un faible nombre d'annotations peut être demandé à l'utilisateur. L'ensemble d'apprentissage est de petite taille, il est donc indispensable de disposer des annotations qui donnent la meilleure classification. De plus, l'apprentissage actif offre un cadre formel plus riche que la sélection « simple »[1], et permet d'exprimer mathématiquement le problème de la sélection.

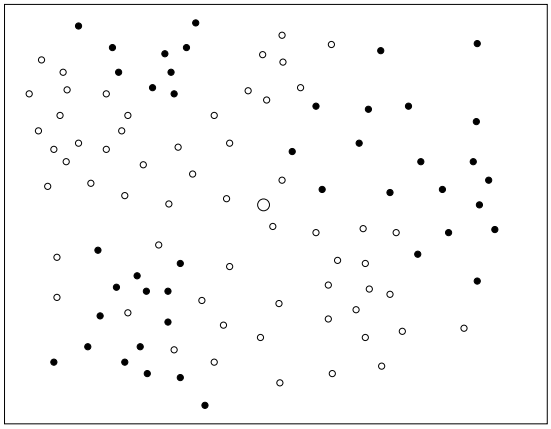
Dans ce chapitre, nous commençons par présenter les motivations relatives à cette approche, ainsi que la formalisation du problème. Nous proposons ensuite une vue d'ensemble des différentes approches actives de la littérature. Enfin, nous présentons les deux méthodes de sélection que nous avons retenues.

5.1 Présentation

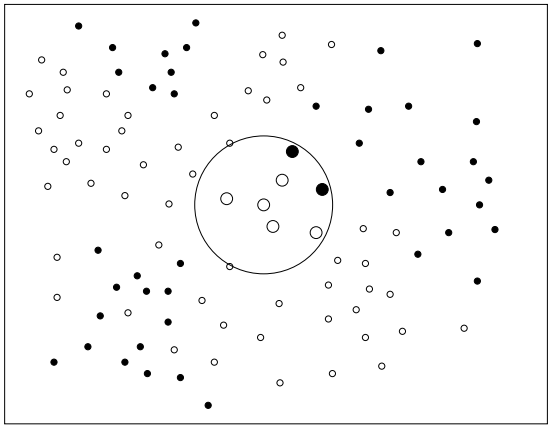
5.1.1 Un exemple de sélection

Illustrons le principe de la classification active par un exemple. Supposons que nous disposons d'un ensemble X de vecteurs de dimension $p = 2$, et qu'il existe une classe pour chacun de ces vecteurs. La figure 5.1(e0) présente cet ensemble sous la forme d'un nuage de points blancs pour la classe pertinente, et de points noirs pour la classe non pertinente. Supposons qu'au temps $t = 0$, l'utilisateur annote le point central comme appartenant à la catégorie recherchée. A ce stade, le système ne connaît que l'annotation de ce point

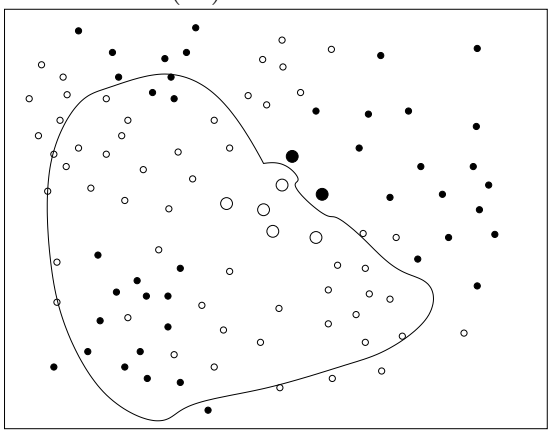
[1] Sélection « simple » : proposer à l'utilisateur d'annoter les images les plus pertinentes.



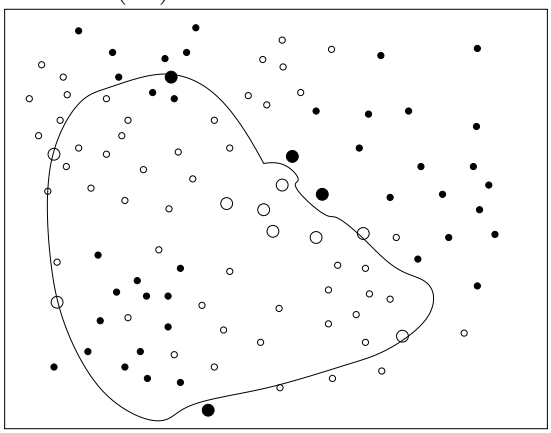
(e0) Etat à t=0



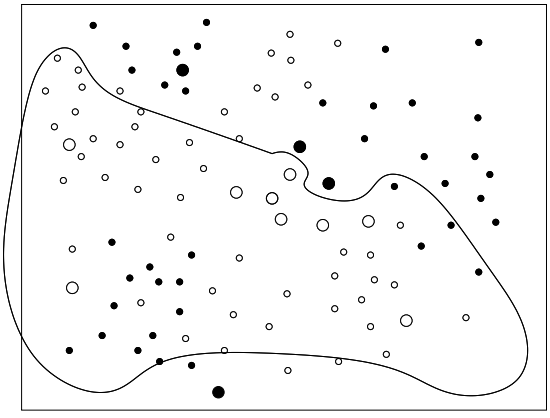
(a0) Annotations à t=0



(e1) Etat à t=1

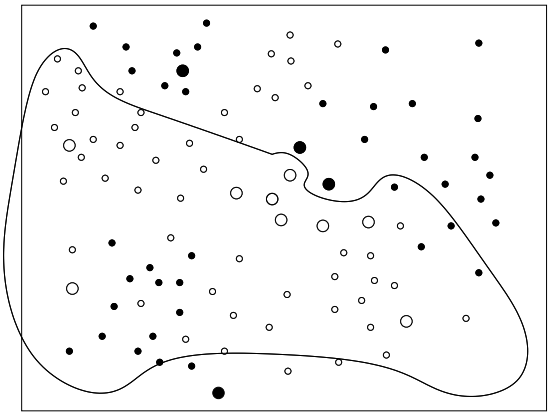


(a1) Annotations à t=1

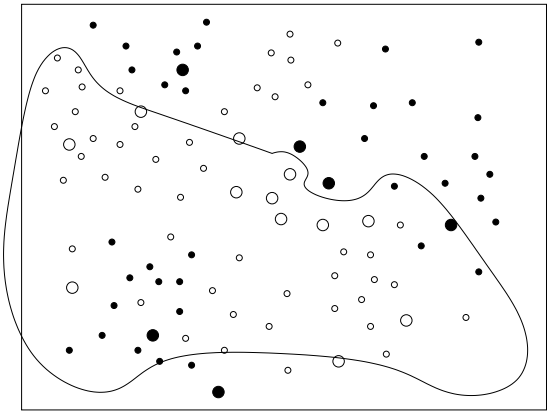


(e2) Etat à t=2

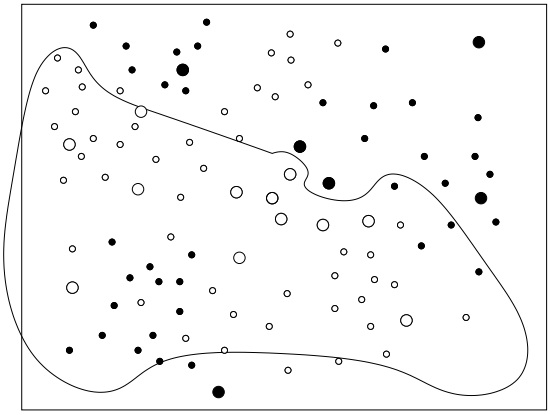
FIG. 5.1 – Exemple de classification active. Chaque point représente une image de la base. L'ensemble des points blancs forment la catégorie recherchée, et les points noirs ne font pas partie de cette catégorie. Les points plus larges (par exemple le point blanc central en e0) sont les points annotés. Le système ne connaît que la classe des points larges, la couleur des petits points lui est inconnue, et n'est affichée que pour des raisons de lisibilité.



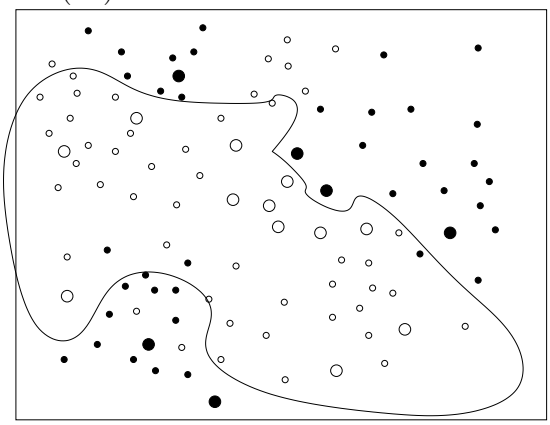
(e2) Etat à t=2



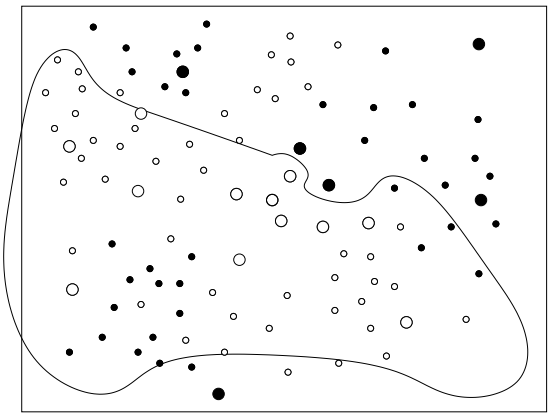
(a2) Choix d'annotations à t=2



(a2') Autre choix d'annotations à t=2



(e3) Etat à t=3



(e3') Autre état à t=3

Fig. 5.2 – Exemple de classification active (suite).

central, représenté par un point blanc plus large. La couleur des autres points n'est pas connue du système, et n'est représentée dans la figure qu'à titre indicatif.

Avec seulement un vecteur dans l'ensemble d'apprentissage, aucune classification n'est possible, le système se contente alors de calculer la pertinence de chaque vecteur en fonction de sa distance au vecteur annoté. Puis, le système propose d'annoter les images les plus proches de ce vecteur (Fig. 5.1(a0)), et on obtient ainsi une première classification (Fig. 5.1(e1)). Puis le système choisit certains vecteurs à annoter (Fig. 5.1(a1)). Il s'ensuit une classification meilleure que la précédente (Fig. 5.1(e2)). Ce processus se répète ainsi autant de fois que nécessaire, jusqu'à satisfaction de l'utilisateur. Cependant, l'utilisateur souhaite obtenir la meilleure classification avec le moins d'annotations possibles. Plusieurs classifications possibles peuvent être obtenues en fonction des annotations choisies. Partons de l'exemple à $t = 2$ (Fig. 5.2(e2)). Avec un certain choix d'annotations (Fig. 5.1(a2)), la classification est améliorée (Fig. 5.1(e3)), alors qu'avec un autre choix (Fig. 5.1(a2')), elle reste presque inchangée (Fig. 5.1(e3')).

Cet exemple montre que le choix des images annotées par l'utilisateur joue un rôle important pour la qualité de la classification. Avec deux ensembles d'apprentissage de même taille, nous avons deux qualités de classification différentes. Dans un premier cas davantage de points sont bien classés par rapport à l'itération précédente, alors que dans l'autre cas, il y a autant de points bien classés qu'à l'itération précédente.

5.1.2 Formalisation

Nous nous intéressons à la classification active dans le cadre des techniques transductives[2] qui ont pour but de minimiser l'erreur de classification sur un ensemble X disponible (ici la base d'images). A cela s'ajoute une notion de temps t, où t désigne l'itération de bouclage, ainsi que la présence d'un expert (l'utilisateur) capable d'annoter toute image de la base. L'expert peut être modélisé par une fonction $s : X \rightarrow \{-1, 1\}$, qui à toute image de la base fait correspondre sa classe.

En classification active, les images à faire annoter par l'utilisateur sont choisies de manière à obtenir la plus petite erreur de classification sur la base. Dans le cas où une seule image $\mathbf{x}_{i^\star}$ doit être sélectionnée, cela s'exprime par la minimisation de l'erreur de classification sur X sur toutes les fonctions $f_{\mathbf{y}_t + s(\mathbf{x}_i)\mathbf{e}_i}$ de classification avec l'ensemble d'apprentissage $\mathbf{y}_t$ à l'itération t auquel on a ajouté l'annotation $s(\mathbf{x}_i)$ de l'image $\mathbf{x}_i$ (et $e_{ij} = \delta_{i,j}$) :

$$i^\star = \underset{i \in \bar{I}}{\operatorname{argmin}} \; R_{\text{test}}(f_{\mathbf{y}_t + s(\mathbf{x}_i)\mathbf{e}_i}) \tag{5.1}$$

avec

[2]Il est possible de faire de la classification active de manière inductive, mais cela n'a pas de beaucoup de sens pour notre approche, sachant qu'il faudrait « présenter un vecteur » à l'utilisateur, ce qui n'entre pas dans le cas d'un système de recherche utilisable, à moins, bien sûr, d'être capable de reconstruire une image à partir de ce vecteur.

$$R_{\text{test}}(f_{\mathbf{y}}) = \frac{1}{|\bar{I}|} \sum_{i \in \bar{I}} \sum_{c \in \{-1,1\}} L(f_{\mathbf{y}}(\mathbf{x}_i), c) P(c|\mathbf{x}_i)$$

Ce problème ne peut être directement résolu, puisque nous ne connaissons pas l'annotation $s(\mathbf{x}_i)$ que l'utilisateur donnera pour chaque image $\mathbf{x}_i$. L'objectif des méthodes de sélection est d'approcher au mieux le résultat de cette minimisation.

5.2 Méthodes de sélection d'une image

5.2.1 Sélection basée sur l'incertitude

Une approche très répandue en apprentissage actif est la sélection basée sur l'incertitude du classifieur. Selon cette approche, l'individu le plus intéressant est celui que le classifieur a le plus de mal à classifier.

(Lewis & Gale, 1994) utilisent un classifieur probabiliste pour évaluer l'appartenance de chaque élément à la classe. Les éléments sélectionnées sont ceux dont la probabilité est la plus proche de $0,5$. Ce type de technique repose sur une bonne estimation de la probabilité de chaque élément, ce qui n'implique pas nécessairement une bonne classification. C'est la raison pour laquelle (Lewis & Catlett, 1994) proposent d'utiliser une technique pour approximer la probabilité de chaque élément, et une autre pour classifier.

Certaines techniques sont développées pour des classifieurs particuliers. Par exemple, pour les classifieurs SVM (Park, 2000) propose de sélectionner des vecteurs supports orthogonaux proches de la frontière, et (Tong & Koller, 2001) proposent trois techniques basées sur la minimisation de l'ensemble des hyperplans à vaste marge. Pour les classifieurs par kPPV, (Hasenjager & Ritter, 1996) et (Lindenbaum et al., 2004) présentent des techniques de sélection.

5.2.2 Sélection basée sur la contradiction de modèles des classes

Cette approche consiste à utiliser différents modèles pour représenter les classes, et à sélectionner les éléments sur lesquels les modèles se contredisent le plus.

Par exemple, (Cohn, 1996) propose d'entraîner plusieurs classifieurs avec le même ensemble d'apprentissage, puis de proposer les éléments dont la classification diffère le plus d'un classifieur à l'autre. (Cohn et al., 1994) proposent un schéma équivalent, mais avec des réseaux de neurones. Deux réseaux sont entraînés sur l'ensemble d'apprentissage, l'un généraliste, et l'autre plus spécifique. Les éléments sont alors sélectionnés si les deux réseaux se contredisent. (Krogh & Vedelsby, 1995) présentent la relation entre l'erreur de généralisation d'un ensemble de réseaux de neurones et son ambiguïté. Plus l'ambiguïté est forte, plus l'erreur est petite. Ils utilisent cette propriété afin de sélectionner des éléments.

Ces approches reposent principalement sur les modèles choisis, qui doivent être cohérents avec l'ensemble d'apprentissage. (Liere & Tadepalli, 1997) proposent d'utiliser 7 classifieurs pour la catégorisation textuelle. (Argamon-Engelson & Dagan, 1999) déplacent le problème en générant un ensemble de classifieurs en fonction de la distribution *a posteriori* des paramètres de ces classifieurs.

5.2.3 Sélection basée sur l'utilité d'une annotation

L'incertitude de classification peut ne pas être suffisante pour la sélection, et l'impact sur les autres points doit être pris en compte. Cette approche suggère de calculer pour chaque candidat un critère d'utilité, qui mesure l'impact de chaque annotation possible sur la classification. Par exemple, (Roy & McCallum, 2001) proposent de choisir l'élément qui minimise l'erreur de généralisation sur la base.

(Lindenbaum et al., 2004) présente un schéma global de la sélection basée sur l'utilité. Le problème est vu comme un jeu entre le système apprenti et l'utilisateur expert. L'algorithme construit un arbre des possibilités, aussi grand que le permettent les ressources matérielles. L'utilité est calculée sur chaque feuille de l'arbre, et la méthode sélectionne le point qui conduit à la plus grande utilité.

5.2.4 Pré-partitionnement (clustering)

Certains chercheurs proposent d'effectuer un pré-partitionnemment de la base avant la sélection. L'idée est de grouper les éléments similaires afin d'améliorer la sélection. (Engelbrecht & Brits, 2002) partitionnent la base, puis sélectionnent les éléments partition par partition. (Nguyen & Smeulders, 2003) partitionnent l'ensemble des éléments non annotés, puis choissisent les éléments qui sont proches de la frontière et représentatifs de partitions denses.

5.3 Présentation de deux méthodes actives de référence

Dans cette section, nous présentons deux méthodes de sélection d'une image. Tout d'abord, nous avons choisi la méthode de (Tong & Koller, 2001), SVM_{active} , qui est une méthode rapide, avec de bons fondements théoriques, et sert de référence dans de nombreux articles pour la comparaison en recherche d'images. Nous avons ensuite choisi la méthode de (Roy & McCallum, 2001), qui est une proposition d'appromixation directe de la minimisation à réaliser en classification active (*cf.* Eq. 5.1).

Ces deux méthodes font parties des briques de base de nos propres méthodes proposées au chapitre 6.

5.3.1 SVM_{active}

La méthode SVM_{active} , proposée par Tong (Tong & Chang, 2001), apporte une justification théorique au choix des éléments à faire annoter dans le cadre de la classification par SVM. Bien que cette technique ait pour but la minimisation de l'erreur de classification, au même titre que toute méthode de classification active, son approche est toutefois orientée vers la minimisation du nombre d'annotations.

Cette méthode propose de réduire de manière optimale l'espace des versions (Tong & Chang, 2001), à savoir dans notre contexte l'ensemble des hyperplans séparateurs pour un jeu d'annotations donné. Cet espace est isomorphe à l'ensemble des normales $\mathbf{w}$ qui classent correctement les données annotées :

$$V = \{\mathbf{w} \in \mathbb{R}^d |\ ||\mathbf{w}|| = 1,\ y_i \mathbf{w}^\top \mathbf{x}_i > 0\ \forall i \in I\}$$

(Tong & Chang, 2001) désignent cet espace V comme étant un sous-ensemble de l'*espace des versions*. Pour un candidat $\mathbf{x}$, on peut définir deux sous-ensembles de V suivant les annotations possibles $c = \pm 1$ de $\mathbf{x}$:

$$V_c = V \cap \{\mathbf{w} \in \mathbb{R}^d |\ c\mathbf{w}^\top \mathbf{x}^\star > 0\ \}$$

Tong montre que la manière optimale de minimiser V d'une itération à l'autre est de diviser son aire par deux. Autrement dit, il faut avoir $Aire(V_1) = Aire(V_{-1})$. Cependant, il n'est pas possible d'un point de vue pratique de calculer ces aires pour tous les points candidats. Il est donc nécessaire de faire une approximation.

Le raisonnement suivant se place dans l'espace des versions, où les images $\mathbf{x}$ sont des hyperplans, et les normales $\mathbf{w}$ sont des points. Sachant que l'algorithme SVM détermine le centre $\mathbf{w}$ de la plus grande sphère qui peut tenir dans V, on peut utiliser cette sphère comme approximation de V. Le but étant de diviser par deux V, le candidat idéal est donc un hyperplan qui coupe la sphère en deux. Le meilleur candidat (un hyperplan dans l'espace des versions) est donc celui dont la distance au centre de cette sphère est la plus petite, soit :

$$\mathbf{x}^\star = \underset{i \in I}{\text{argmin}}\ \mathbf{w}^\top \mathbf{x}_i$$

Cette présentation suppose que les données sont séparables et normalisées. Le passage aux fonctions noyaux est simple, et l'hypothèse de normalisation peut être contournée (Tong, 2001).

5.3.2 Minimiser l'erreur de généralisation

Cette méthode s'intéresse directement au critère de la classification active, à savoir sélectionner l'image qui, une fois ajoutée à l'ensemble d'apprentissage, minimisera l'erreur de généralisation. (Roy & McCallum, 2001) présentent leur approche dans le cadre plus général de la classification active par induction, qui tente de minimiser le risque réel. Nous présentons ici cette méthode dans le cadre de la transduction, pour des raisons de lisibilité.

Le but de cette technique est de déterminer l'individu $\mathbf{x}_{i^\star}$ qui, une fois ajouté à l'ensemble d'apprentissage avec l'annotation $s(\mathbf{x}_{i^\star})$ de l'utilisateur, minimise l'erreur de généralisation sur X :

$$i^\star = \underset{i \in \bar{I}}{\operatorname{argmin}} \; R_{\text{test}}(f_{\mathbf{y}+s(\mathbf{x}_i)\mathbf{e}_i})$$

Cette minimisation n'est pas calculable en pratique, compte tenu du fait que nous ne disposons pas de l'annotation $s(\mathbf{x}_i)$ de chaque individu $\mathbf{x}_i$. (Roy & McCallum, 2001) proposent une approximation de ce calcul.

Ils calculent une approximation $\hat{R}_{\text{test}}$ du risque pour les deux annotations possibles. Ils pondèrent par une approximation de la probabilité $\hat{P}_{\mathbf{y}}(c|\mathbf{x})$ d'être dans la classe c sachant $\mathbf{x}$:

$$i^\star = \underset{i \in \bar{I}}{\operatorname{argmin}} \sum_{c \in \{-1,1\}} \hat{R}_{\text{test}}(f_{\mathbf{y}+c\mathbf{e}_i}) \hat{P}_{\mathbf{y}}(c|\mathbf{x}_i)$$

La pondération permet d'augmenter la confiance en la classe c d'un individu $\mathbf{x}_i$ en fonction de son appartenance *a posteriori* pour le classifieur actuel (*i.e.* avec seulement les annotations $\mathbf{y}$) :

$$\hat{P}_{\mathbf{y}}(c|\mathbf{x}) = \frac{c}{2}(f_{\mathbf{y}}(\mathbf{x}) + c)$$

De même, pour estimer le risque $\hat{R}_{\text{test}}(f_{\mathbf{y}_t+c\mathbf{e}_i})$ supposant que la classe de $\mathbf{x}_i$ est c, on ne connaît toujours pas la classe des autres individus. Rappelons que la formulation générale du risque est :

$$R_{\text{test}}(f_{\mathbf{y}+c\mathbf{e}_i}) = \frac{1}{|\bar{I}| - 1} \sum_{j \in \bar{I} - \{i\}} \sum_{c' \in \{-1,1\}} L(f_{\mathbf{y}+c\mathbf{e}_i}(\mathbf{x}_j), c') P(c'|\mathbf{x}_j)$$

(Roy & McCallum, 2001) proposent d'approximer $P(c'|\mathbf{x}_j)$ en s'appuyant sur la probabilité *a posteriori* $\hat{P}_{\mathbf{y}+c\mathbf{e}_i}(c'|\mathbf{x}_j)$ pour un classifieur entraîné avec les annotations $\mathbf{y}$ et l'annotation c de $\mathbf{x}_j$.

Avec une fonction de perte L au sens du logarithme, l'estimation devient :

$$\hat{R}_{\text{test}}(f_{\mathbf{y}+c\mathbf{e}_i}) = \frac{1}{|\bar{I}|-1} \sum_{j\in\bar{I}-\{i\}} \sum_{c'\in\{-1,1\}} \hat{P}_{\mathbf{y}+c\mathbf{e}_i}(c'|\mathbf{x}_j) \log\left(\hat{P}_{\mathbf{y}+c\mathbf{e}_i}(c'|\mathbf{x}_j)\right)$$

Et avec une fonction de perte tout ou rien :

$$\hat{R}_{\text{test}}(f_{\mathbf{y}+c\mathbf{e}_i}) = \frac{1}{|\bar{I}|-1} \sum_{j\in\bar{I}-\{i\}} \left(1 - \max_{c'\in\{-1,1\}} \hat{P}_{\mathbf{y}+c\mathbf{e}_i}(c'|\mathbf{x}_j)\right)$$

La méthode repose principalement sur une bonne estimation $\hat{P}_{\mathbf{y}}(c|\mathbf{x})$ de la classe de chaque individu. C'est très certainement la raison pour laquelle elles est présentée dans (Roy & McCallum, 2001) avec des classifieurs par champs gaussiens qui, selon les auteurs, offre une très bonne estimation de $P_{\mathbf{y}}(c|\mathbf{x})$ (Zhu et al., 2003). Malheureusement cette méthode de classification a un coût calculatoire prohibitif ($O(n^3)$).

5.4 Méthodes de sélection d'un lot d'images

Les méthodes précédentes ont été construites pour choisir une seule image à faire annoter. Au sein d'un système de recherche d'images, on peut être amené à sélectionner plusieurs images à chaque itération, ne serait-ce que pour réduire les calculs.

Ce problème revient à déterminer un ensemble de q images, que nous notons par l'ensemble $I^\star$ des indices des images sélectionnées. L'objectif est de déterminer, parmi tous les lots d'images possibles, celui qui minimisera l'erreur de classification une fois annoté et ajouté à l'ensemble d'apprentissage. Cela s'exprime par la minimisation du risque sur toutes les fonctions de classification avec l'ensemble d'apprentissage $\mathbf{y}$ auquel on a ajouté la vraie classe de q images prises dans l'ensemble des images non annotées :

$$I^\star = \underset{\hat{I}\subset\bar{I},|\hat{I}|=q}{\operatorname{argmin}} R_{\text{test}}(f_{\mathbf{y}_t+\sum_{i\in\hat{I}} c_i\mathbf{e}_i})$$

Au même titre que pour la sélection d'une seule image, nous ne connaissons pas la vraie classe de chaque image. De plus, la complexité de ce problème est plus importante de par le nombre $C^q_{|\bar{I}|}$ de sous-ensembles $\hat{I}$ à tester.

Dans l'optique d'un système de recherche rapide, il n'est pas envisageable de calculer le risque pour tous ces sous-ensembles. Une pratique courante pour réduire le temps de calcul avec ce type de minimisation est de procéder de manière itérative. On calcule le critère de sélection d'une seule image, on sélectionne la meilleure, puis on l'ajoute à l'ensemble d'apprentissage avec une estimation de sa vraie classe. Puis on réitère, en calculant le critère de sélection d'une seule image, *etc.* Ce processus itératif repose sur l'estimée de la

vraie classe de chaque image sélectionnée. Or, un des critères que nous utilisons est de choisir précisément les images dont l'estimation de la vraie classe est la plus difficile.

Afin de pallier ce problème, un critère souvent utilisé est celui de la *diversité*. L'idée est de sélectionner un lot d'images différentes, tout en préservant leur intérêt à être sélectionnées. Par exemple, il n'est pas intéressant de sélectionner plusieurs images simplement en prenant les q images les plus proches de la marge. En effet, dans le cas où l'image optimale est entourée par d'autres images très proches, toutes ces images seront proposées à l'utilisateur. Etant donnée la puissance actuelle des classifieurs, annoter une de ces images donnera le même résultat que si une seule d'entre elles était annotée, d'où une perte en terme d'efficacité du système.

Nous présentons au paragraphe suivant une méthode de diversification qui repose sur la diversité des angles.

5.4.1 Diversité des Angles (Angle diversity)

Cette approche, proposée par (Brinker, 2003), choisit les images les plus orthogonales entre elles, tout en s'assurant qu'elles sont proches de la frontière. Cette technique s'incrit dans le cadre de la minimisation de l'espace des versions.

L'idée est la même que dans (Tong & Koller, 2001) : pour un candidat $\mathbf{x}$, on peut définir deux sous-ensembles de V suivant les annotations possibles c de $\mathbf{x}$:

$$V_c^1 = V \cap \{\mathbf{w} \in \mathbb{R}^d | \ c\mathbf{w}^\top \mathbf{x} > 0 \ \}$$

Pour la première sélection, on choisit le point $\mathbf{x}_{i_1^\star}$ qui divise au mieux en deux l'aire de V, à savoir le point le plus proche de la frontière.

Pour la suivante, on poursuit dans le même esprit : pour un candidat $\mathbf{x}$, on peut définir deux sous-ensembles de V_c^1 suivant les annotations possibles c de $\mathbf{x}$:

$$V_c^2 = V_c^1 \cap \{\mathbf{w} \in \mathbb{R}^d | \ c\mathbf{w}^\top \mathbf{x} > 0 \ \}$$

L'idée est aussi de choisir le point $\mathbf{x}_{i_2^\star}$ tel que $Aire(V_1^2) = Aire(V_{-1}^2)$. Dans l'espace des versions, V_c^1 n'est pas une sphère comme V, mais une demi-sphère. L'hyperplan qui divise au mieux en deux l'aire d'une demi-sphère est celui qui passe par son origine et est orthogonale à sa base. Cela revient à dire que le candidat $\mathbf{x}_{i_2^\star}$ recherché est celui qui est proche de la frontière (le centre de l'hypersphère) et dont l'angle avec $\mathbf{x}_{i_1^\star}$ (la base de la demi-sphère) est le plus petit possible.

Notons qu'il existe deux possibilités pour V_c^1 puisqu'on ne connaît pas l'annotation c de $\mathbf{x}_{i_1^\star}$. Cependant, comme on choisi $\mathbf{x}_{i_1^\star}$ de sorte à avoir $Aire(V_1^1) \simeq Aire(V_{-1}^1)$, les deux possibilités sont approximativement identiques. Selon (Brinker, 2003), cette approche est une bonne appromixation de la minimisation de l'espace des versions.

Pour les sélections suivantes, le raisonnement est le même. Il s'ensuit un choix de vecteurs qui sont proches de la frontière, et le plus possible orthogonaux entre eux, ce qui revient à dire que le cosinus de l'angle entre deux vecteurs est proche de zéro.

Le cosinus de l'angle entre deux points $\mathbf{x}_i$ et $\mathbf{x}_j$ dans l'espace induit par une fonction noyau est donné par :

$$\cos(\mathbf{x}_i, \mathbf{x}_j) = \frac{|\Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_j)|}{||\Phi(\mathbf{x}_i)||\ ||\Phi(\mathbf{x}_j)||} = \frac{|k(\mathbf{x}_i, \mathbf{x}_j)|}{\sqrt{k(\mathbf{x}_i, \mathbf{x}_i) k(\mathbf{x}_j, \mathbf{x}_j)}}$$

Pour le choix d'un nouveau point, la méthode minimise la somme pondérée de la distance à la frontière et du plus grand cosinus entre le point testé et un point annoté :

$$g_{I^\star}(\mathbf{x}_i) = \underbrace{\lambda |f(\mathbf{x}_i)|}_{\text{distance à la frontière}} + \underbrace{(1-\lambda) \max_{j \in I^\star} \frac{|k(\mathbf{x}_i, \mathbf{x}_j)|}{\sqrt{k(\mathbf{x}_i, \mathbf{x}_i) k(\mathbf{x}_j, \mathbf{x}_j)}}}_{\text{plus grand cosinus entre le vecteur testé } (\mathbf{x}_i) \text{ et un vecteur annoté } (\mathbf{x}_j)}$$

avec $I^\star$ initialisé à I, et λ un réel entre 0 et 1.

Une fois qu'un point a été choisi, on ajoute à $I^\star$ l'indice du point choisi, puis on minimise de nouveau $g_{I^\star}$ pour obtenir un nouveau point, à la fois éloigné (au sens des angles) des points de l'ensemble d'apprentissage et des points précédemment choisis. Ces opérations sont repetées autant de fois qu'il y a de points à choisir.

Le paramètre λ permet de doser entre minimiser $|f(\mathbf{x})|$ et diversifier les propositions. Ce paramètre permet d'adapter la méthode à un contexte plus ou moins difficile. Pour des catégories très difficiles, où les éléments pertinents sont rares, deux images très proches ne sont pas forcément dans la même classe. De plus, dans ce cas précis, une image non annotée proche d'une image annotée positivement a une faible probabilité de pertinence. Choisir des images diversifiées dans ce cas n'est pas intéressant, voire problématique. Dans le cas contraire, où la catégorie est facile à décrire, il préférable de choisir des images très éloignées.

Cependant, la détermination de ces deux cas n'est pas triviale. Ce genre de problème apparaît surtout en début de session, où peu d'annotations sont disponibles. Il est donc nécessaire d'avoir une information supplémentaire, comme par exemple l'utilisation de méta données pour régler cet *a priori*.

Une fois que l'on a effectué les premières itérations, et que l'ensemble d'apprentissage devient conséquent, il est envisageable de faire une estimation de la difficulté de la catégorie recherchée. En l'absence de toute estimation de λ, ce paramètre est fixé à $\frac{1}{2}$.

5.5 Synthèse

Nous avons présenté dans ce chapitre la problématique de la classification active, ainsi que des solutions proposées dans la littérature, puis nous avons détaillé deux approches de référence. Cependant, ces techniques ne répondent pas à toutes les exigences de la recherche interactive, et de nombreuses améliorations sont possibles.

Citons quelques points importants :

Les méthodes de classification classiques n'ont pas été initialement construites pour traiter deux classes de tailles très différentes. Elles ont en général pour but de diviser une base en deux parts de tailles égales. Dans notre contexte, la classe pertinente est beaucoup plus petite que la classe non pertinente, en général de 20 à 100 fois plus petite. Par conséquent, il est difficile d'obtenir une frontière de bonne qualité. En effet, lors des premières itérations, alors qu'il n'y a que peu d'exemples, la frontière est particulièrement instable. Nous proposons au chapitre suivant une méthode pour corriger la frontière en vue d'améliorer la sélection.

Le temps de calcul a une importance non négligeable dans notre contexte. Il n'est pas possible de faire patienter un utilisateur plusieurs heures entre chaque itération de bouclage. Pour des méthodes comme celle de (Roy & McCallum, 2001) où l'on teste toutes les possibilités, le temps de calcul est en $O(n^2)$. En effet, nous avons n images à tester, et pour chacune un calcul supplémentaire des n pertinences. Nous proposons au chapitre suivant une technique pour approximer ces calculs, et ainsi réduire la complexité à $O(n)$.

Un autre problème concerne l'objectif visé. Dans notre contexte, le critère prédominant est la Précision Moyenne, qui modélise la satisfaction de l'utilisateur pour un classement de la base. Or les techniques de classification active s'intéressent à la minimisation de l'erreur de classification. Bien que ce critère ne soit pas étranger à notre objectif, *i.e.* une faible erreur de classification entraîne une bonne Précision Moyenne, nous montrons au chapitre suivant qu'elles sont loin d'être équivalentes. Partant de ce constat, nous proposons une technique de sélection qui combine la minimisation de l'erreur et la maximisation de la Précision Moyenne.

Chapitre 6

Stratégie RETIN 2 d'apprentissage actif

Le chapitre précédent met en évidence l'intérêt de la classification active pour la recherche interactive. Cependant, les techniques présentées ne répondent pas à toutes les exigences de notre contexte. Nous proposons dans ce chapitre des solutions pour améliorer ces techniques.

Nous présentons l'architecture du système RETIN 2 d'apprentissage actif développée dans cette thèse, dont les différents modules sont ensuite détaillés. Cette partie permet d'avoir une vue d'ensemble des différentes contributions que nous proposons en apprentissage interactif. Nous terminons ce chapitre par un ensemble d'expérimentations.

6.1 Architecture RETIN 2 d'apprentissage actif

Le schéma 6.1 représente la partie « en-ligne » du système RETIN 2. Nos propositions pour améliorer la classification active pour la recherche d'images concernent le bloc *correction* et le bloc *sélection*. Les blocs non détaillés dans ce paragraphe sont les mêmes que ceux du schéma général de la figure 4.1.

6.1.1 Classification et Correction

La classification de la base est calculée à partir de l'ensemble d'apprentissage à l'itération actuelle, en utilisant l'une des méthodes présentées dans le chapitre 4 : les k-Plus Proches Voisins, les SVM, les mélanges de Gaussiennes, etc. On en déduit une fonction de pertinence f, qui à toute image de la base fait correspondre sa pertinence à la catégorie recherchée.

La qualité de la fonction de pertinence f est déterminante pour le processus de sélection. Lorsque l'on souhaite sélectionner les images les plus proches de la frontière, cela

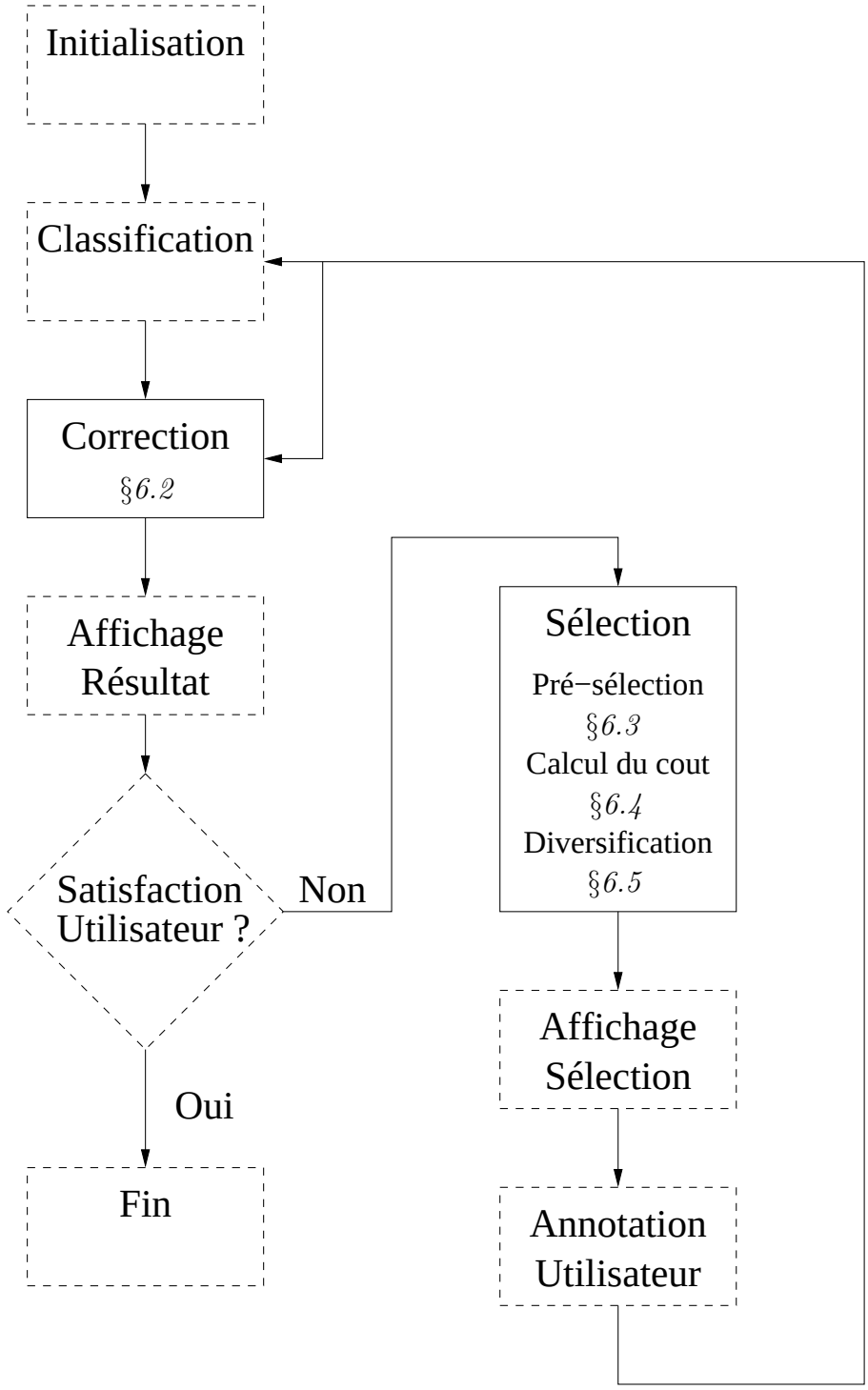


Fig. 6.1 – Achitecture RETIN 2 d'apprentissage actif

revient à s'intéresser aux images $\mathbf{x}_i$ telles que $|f(\mathbf{x}_i)|$ soit proche de zéro. Cela découle du formalisme que nous avons choisi, qui stipule qu'une image $\mathbf{x}_i$ est dans la classe pertinente si $f(\mathbf{x}_i)$ est proche de 1, dans la classe non pertinente si $f(\mathbf{x}_i)$ est proche de -1, et difficile à classifier si $f(\mathbf{x}_i)$ est proche de 0.

Il est donc important de disposer d'une bonne estimation de la fonction de pertinence, et tout particulièrement du passage de cette fonction par zéro. Cependant, nous travaillons dans un contexte où l'ensemble d'apprentissage est très petit, ce qui conduit à une mauvaise estimation. Cela est, semble-t-il, dû au fait que les techniques de classification ont tendance à diviser la base en deux parties de tailles égales. Lors de simulations, nous observons de grandes variations dans l'estimation du nombre d'images dans chaque classe selon f. En effet, plusieurs milliers d'images peuvent changer de classe d'une itération à l'autre.

Dans le but de contrecarrer ces effets, qui conduisent à une mauvaise sélection, nous proposons une solution au §6.2. L'idée est de travailler sur l'emplacement de la frontière, *i.e.* de *corriger* la fonction de pertinence de sorte que la sélection soit meilleure.

6.1.2 Sélection

Le temps de calcul a une importance non négligeable dans notre contexte. Pour toutes les méthodes qui testent l'influence de chaque image de la base sur la classification future, le temps de calcul est en $O(n^2)$. En effet, ces méthodes évaluent la classification qui pourrait être engendrée si l'utilisateur annote une image. Chacune de ces évaluations demande le recalcul de la pertinence des n images de la base, et doit être répétée pour chacune des n images de la base. C'est le cas de toute les méthodes basées « utilité », comme la méthode de (Roy & McCallum, 2001).

Afin de réduire ce temps de calcul, tout en préservant les performances, nous proposons une étape préliminaire à ces évaluations. L'idée est de *pré-sélectionner* un faible nombre d'images qui sont suceptibles de faire partie des images que la technique de sélection aurait choisies si toutes les images avaient été évaluées. Nous proposons une méthode pour ce faire au §6.3. Après étude de cette solution, il s'est avéré qu'elle peut être utilisée dans un autre but : combiner deux approches opposées de sélections. Nous discutons ce point au §6.3.

Les méthodes de sélection pour la classification active ont pour but la minimisation de l'erreur de généralisation. Or le critère que nous utilisons est la Précision Moyenne. Nous comparons au §6.4 deux techniques de sélections idéales basées sur ces critères. En utilisant la vérité terrain, l'une choisit l'image qui minimise l'erreur de généralisation, et l'autre choisit l'image qui maximise la Précision Moyenne. Les résultats montrent que les deux critères ne sont pas étrangers, mais il existe toutefois une différence non négligeable. C'est la raison pour laquelle nous proposons au §6.4 une technique de sélection qui augmente la Précision Moyenne.

Toujours dans l'idée de proposer un système de recherche interactive rapide, nous nous orientons sur un schéma de sélection d'un lot par diversification. Sélectionner des

images éloignées dans le but de prévenir l'annotation d'images proches est une solution sous-optimale de la sélection d'un lot, mais a le mérite d'être peu coûteuse en calculs. Ce schéma s'applique en deux temps :

1. *Calcul du coût.* Le premier temps est consacré au calcul de ce que nous appelons le coût, qui revient à tester chaque image avec un critère de sélection d'une seule image. Par exemple, SVM_{active} , (Roy & McCallum, 2001) ou encore la méthode proposée au §6.4. Chacune de ces méthodes renvoie une valeur pour chaque image de la base, par exemple pour SVM_{active} la distance à la frontière. Ceci peut se généraliser par le calcul d'une *fonction de coût* $g(\mathbf{x}_i)$ pour chaque image $\mathbf{x}_i$, par exemple pour SVM_{active} $g(\mathbf{x}_i) = |f(\mathbf{x}_i)|$. L'évaluation de cette fonction est limitée à l'ensemble des images présélectionnées.
2. *Diversification.* Le deuxième temps est consacré à la sélection du lot $I^\star$ d'images qui seront présentées à l'utilisateur pour annotation. L'idée est d'assurer que le lot sélectionné ne comportera pas d'images proches.

Nous proposons au §6.5 une méthode de diversification basée sur le clustering, ainsi qu'une généralisation de la diversité des angles.

6.1.3 Algorithme global

TAB. 6.1: Architecture RETIN 2

fonction mise_à_jour$(k, \mathbf{y} \rightarrow f, I^\star)$	
Entrées :	
$k(\mathbf{x}_i, \mathbf{x}_j)$	*Fonction noyau*
$\mathbf{y}$	*Annotations de l'utilisateur*
Sorties :	
f	*Fonction de pertinence*
$I^\star$	*Lot d'images à faire annoter*
Variables statiques :	
f_{t-1}	*Fonction de pertinence à l'itération précédente*
$I^\star_{t-1}$	*Lot d'images à l'itération précédente*
r_{t-1}	*Paramètre de correction à l'itération précédente*
$r = \text{retour}(r_{t-1}, f_{t-1}, I^\star_{t-1}, \mathbf{y})$	
$f = \text{classifier}(k, \mathbf{y})$	
$f = \text{corriger}(f, r)$	
$J = \text{présélectionner}(f)$	
$g = \text{sélectionner}(k, \mathbf{y}, f, J)$	
$I^\star = \text{diversifier}(k, g, J)$	
$f_{t-1} = f$	
$I^\star_{t-1} = I^\star$	
$r_{t-1} = r$	

6.2 Correction active de la frontière

Nous proposons dans cette section une technique de correction active de la frontière afin d'améliorer la qualité de la classification lorsqu'il y a très peu d'exemples, et par la même la qualité de la sélection active (Gosselin & Cord, 2004b).

6.2.1 L'importance de la frontière

La frontière joue un rôle très important lors de la sélection des images à faire annoter par l'utilisateur. Si nous prenons le cas de techniques qui sélectionnent les images les plus proches de celle-ci, comme SVM_{active} par exemple, il est clair que de grandes variations dans l'emplacement de la frontière peut totalement changer la sélection. De même, lors du calcul d'un critère comme celui de l'erreur de généralisation, des effets similaires apparaissent puisque la probabilité d'appartenance dépend fortement de la position d'une image par rapport à la frontière.

Or, lors des premières itérations d'une recherche, les classifieurs sont entraînés avec un nombre extrêmement faible d'exemples, de l'ordre de $0,1\%$ de la la base. Les classifieurs ont alors trop peu d'information pour ne serait-ce que déterminer l'ordre de grandeur de la taille de la classe pertinente. Leur tendance naturelle est, conformément à leur conception, de couper grossièrement la base en deux parties égales. A ce stade, chaque nouvel exemple entraîne de grands changements de classification. Plusieurs centaines, voir milliers d'images changent de classe à chaque itération. La sélection repose alors sur une frontière quasiment aléatoire. Pour le cas de la sélection des images les plus proches de la frontière, le comportement est pratiquement celui d'une classification non active, dont les exemples sont choisis aléatoirement.

Nous proprosons une approche heuristique afin de réduire ce type de comportement. Une approche statistique semble difficile dans ce contexte, étant donnés le manque d'informations et l'échec concernant les techniques semi-supervisées.

6.2.2 Schéma

La correction que nous proposons consiste à recaler la fonction de pertinence dans le but de la faire passer par zéro sur l'emplacement "idéal" de la fontière. La notion d'"idéal" dépend fortement du comportement que nous souhaitons obtenir, c'est à dire améliorer la sélection active, tout en gardant à l'esprit qu'il y a une interaction avec un utilisateur.

Nous souhaitons que la fonction corrigée estime mieux l'appartenance de chaque image à chaque classe. En d'autres termes, nous souhaitons qu'une image vue par l'utilisateur comme pertinente ait une appartenance positive, et qu'une image vue par l'utilisateur comme non pertinente ait une appartenance négative.

Plus que bien estimer la classification, nous souhaitons effectuer cette correction de manière à améliorer le classement des images, à l'image de la Précision Moyenne qui prévaut sur l'erreur de classification. Considérant le classement des images selon f, nous souhaitons déterminer la correction b_t telle que la fonction corrigée $\hat{f}$ passe par zéro lors du passage de la classe pertinente à la classe non pertinente :

$$\hat{f}(\mathbf{x}_i) = f(\mathbf{x}_i) - b_t$$

avec b_t la correction à l'itération t.

Avec une telle correction, si l'utilisateur observe les images une par une de la plus pertinente à la moins pertinente, alors nous souhaitons que $\hat{f}$ passe par zéro lorsque l'utilisateur a le sentiment que l'on est en train de sortir de la classe pertinente, autrement dit qu'il commence à y avoir beaucoup trop d'images qui ne l'intéressent pas.

La méthode que nous proposons ajoute une étape supplémentaire dans le processus d'apprentissage actif, avant l'étape de sélection. La technique remplace la fonction de pertinence f par une fonction de pertinence corrigée $\hat{f}$. L'étape de sélection qui suit reste inchangée.

6.2.3 Méthode

Afin d'obtenir le comportement souhaité, nous nous intéressons au classement de la base selon la pertinence au temps t :

$$O_t = \text{argsort} f_t(X)$$

avec argsort($\mathbf{v}$) une fonction qui renvoie les indices des éléménts triés de $\mathbf{v}$.

$$\underbrace{\mathbf{x}_{O_1}, \mathbf{x}_{O_2}, \dots}_{\text{Coeur de la catégorie}} \quad \underbrace{\dots, \mathbf{x}_{O_{r_t-1}}, \mathbf{x}_{O_{r_t}}, \mathbf{x}_{O_{r_t+1}}, \dots}_{\text{Zone d'incertitude}} \quad \underbrace{\dots, \mathbf{x}_{O_{n-1}}, \mathbf{x}_{O_n}}_{\text{Images les moins pertinentes}}$$

Puis nous déterminons le rang r_t tel que les r_t premières images selon O soient majoritairement pertinentes, puis que les $n - r_t$ images suivantes soient majoritairement des images non pertinentes. Nous cherchons le rang à partir duquel les images pertinentes et les images non pertinentes se mélangent équitablement. Ceci implique une correction du type :

$$b_t = f(\mathbf{x}_{O_{r_t}})$$

Afin de déterminer le rang r_t à l'itération t, nous proposons d'effectuer la mise à jour suivante :

$$r_{t+1} = r_t + h(f_t, I_t^\star, \mathbf{y}_{t+1})$$

avec f_t la fonction de pertinence à l'itération t, $I_t^\star$ les indices des images sélectionnées par l'apprentissage actif à l'itération t, $\mathbf{y}_{t+1}$ les annotations à l'itération $t+1$ (*i.e.* $\mathbf{y}_t$ auquel on a ajouté les nouvelles annotations des images de $I_t^\star$), et h une heuristique qui évalue le bon positionnement actuel de la frontière.

Le principe de cette approche est d'exploiter l'interaction avec l'utilisateur afin de déterminer le rang r_{t+1} sachant r_t. Lorsque l'utilisateur donne de nombreuses annotations positives sur les images qui ont été sélectionnées à l'itération t, nous supposons que toutes les images jusqu'au rang r_t sont majoritairement pertinentes. Nous pouvons en déduire que la zone problématique se trouve plus éloignée du coeur de la classe, donc qu'un rang r_{t+1} supérieur à r_t est plus intéressant. Inversement, lorsque de nombreuses annotations négatives sont données, nous pouvons en déduire que la zone problématique se trouve plus près du coeur de la classe, donc qu'un rang r_{t+1} inférieur à r_t est plus intéressant. Tout ceci dans l'idée que les images annotées autour du rang r_t ont une chance sur deux d'être pertinentes, donc que nous sommes dans la zone problématique.

6.2.4 Heuristique

Un premier choix pour l'heuristique h est de compter le nombre de nouvelles annotations positives (resp. négatives), puis d'augmenter (resp. diminuer) d'autant plus le rang qu'il y a d'annotations positives (resp. négatives) :

$$h(f_t, I_t^\star, \mathbf{y}_{t+1}) = \sum_{i \in I_t^\star} y_{i,t+1}$$

Un deuxième choix pour l'heuristique h est de pondérer le déplacement en fonction de la distance d'une image à la frontière. En effet, la méthode de correction a pour but de trouver la zone problématique, où les images des deux classes sont mélangées. Si nous supposons que le lot d'images sélectionnées à l'itération t sont les images les plus proches de la frontière, alors la première heuristique suffit. Or, si le lot a été sélectionné par une méthode qui ne choisit pas nécessairement les images les plus proches de la frontière, alors l'annotation de chacune de ces images n'a pas la même signification.

Par exemple, si l'utilisateur annote négativement une image qui se trouve dans le coeur de la classe, cela signifie que le coeur de la classe n'est pas encore bien défini. Il semble plus intéressant de reculer davantage le rang. Inversement, si l'utilisateur annote positivement une image qui se trouve loin du coeur de la classe, on peut alors supposer que la généralisation est bonne, et que le rang peut être avancé plus largement :

$$h(f_t, I_t^\star, \mathbf{y}_{t+1}) = \sum_{i \in I_t^\star} \big(y_{i,t+1} - f_t(\mathbf{x}_i)\big)$$

TAB. 6.2: Correction de la frontière

fonction corriger$(f, r \rightarrow \hat{f})$
Entrées : f *Fonction de pertinence* r *Rang* Sorties : $\hat{f}$ *Fonction de pertinence corrigée*
$O = \text{trier}(f)$ $\hat{f} = f - f(\mathbf{x}_{O_r})$

fonction retour$(r, f, I^\star, \mathbf{y} \rightarrow \hat{r})$
Entrées : r *Rang précédent* f *Fonction de pertinence précédente* Sorties : $I^\star$ *Lot d'images qui viennent d'être annotés* $\mathbf{y}$ *Nouvelles annotations* $\hat{r}$ *Nouveau Rang*
$\hat{r} = r + \sum_{i \in I^\star} \big(y_i - f(\mathbf{x}_i)\big)$

6.2.5 Comparaison avec la frontière non corrigée

La figure 6.2 présente le rang de passage à zéro avec ou sans correction. Cette évaluation est effectuée sur la base COREL présentée en introduction sur la catégorie « montagnes », avec 5 annotations par itération.

Nous pouvons voir sur cette figure que la frontière sans correction est particulièrement bruitée en début de session, et que son comportement est proche de l'aléatoire. Par contre, la frontière corrigée est stable en début de session. De plus, lorsque la frontière normale est stabilisée, la frontière corrigée a un comportement très proche de celle-ci. Cette expérience montre que la correction constitue une bonne approximation de la frontière recherchée.

6.3 Pré-sélection

Nous présentons dans cette section notre approche de pré-sélection, qui consiste à choisir rapidement un sous-ensemble J de la base auquel appartient le lot $I^\star$ de sélection final. L'idée première est de réduire les temps de calculs des méthodes de sélection basées « utilité », en réduisant les calculs de ces techniques sur le sous-ensemble présélectionné. Ce schéma est une approximation de la sélection, qui ne garantit pas que le lot de sélection final soit le même que sans pré-sélection. L'appromixation dépend de la taille du sous-ensemble pré-sélectionné : il est alors naturel de penser que plus l'approximation est forte,

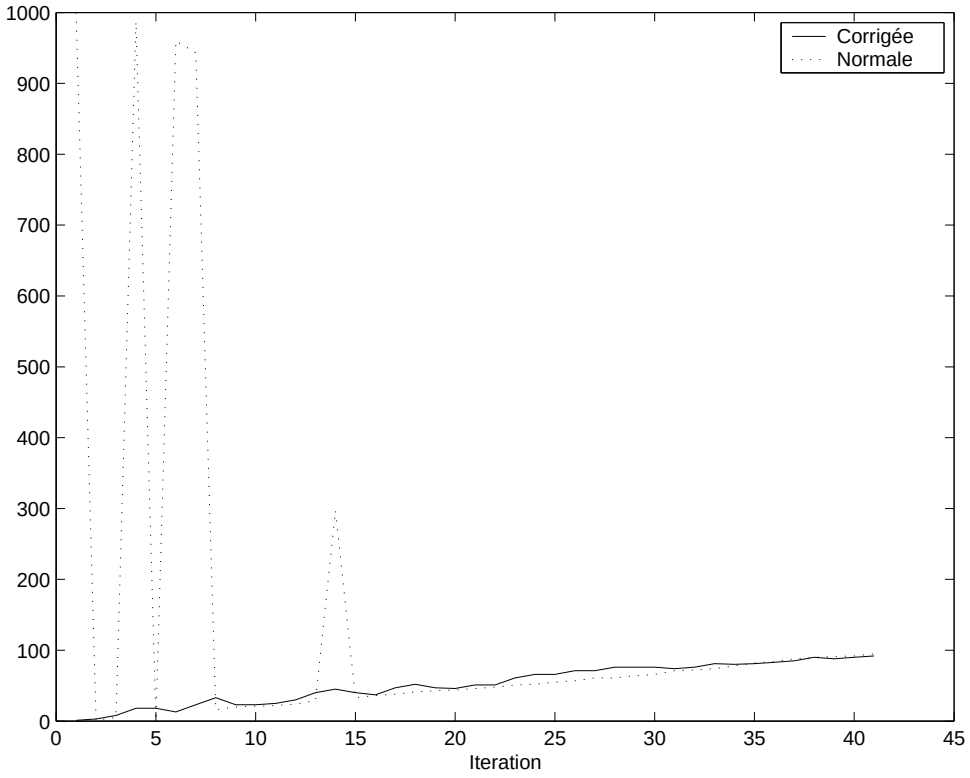


FIG. 6.2 – Comparaison entre le rang de passage à zéro de la fonction de pertinence avec (trait plein) ou sans (pointillés) correction.

les performances sont mauvaises. Or, dans notre contexte où les annotations sont pauvres et le critère premier n'est pas l'erreur de généralisation, il s'avère que l'effet est contraire. Ceci nous invite à proposer un schéma non plus uniquement pour la réduction du temps de calcul, mais dans le but de mettre en opposition deux approches de sélection.

6.3.1 Réduire les temps de calcul

Le temps de calcul a une importance non négligeable dans notre contexte. Par exemple, il n'est pas possible de faire patienter un utilisateur plusieurs heures entre chaque itération de bouclage.

Pour des méthodes basées « utilité », l'évaluation de l'intérêt de chaque image a un temps de calcul en $O(n)$. Par exemple, pour (Roy & McCallum, 2001), chaque évaluation nécessite le calcul d'une classification et de toutes les pertinences pour chaque annotation possible. Dans notre contexte où les annotations sont peu nombreuses, le calcul d'une nouvelle classification est négligeable, mais le calcul de la pertinences des n images de la base pour tout nouveau classifieur reste important.

Sans technique de réduction des calculs, les évaluations sont éffectuées sur toute la base, ce qui conduit à une complexité en $O(n^2)$. Nous proposons de limiter ces évaluation

sur un sous-ensemble J. Dans ce cas, si nous pré-sélectionnons m images, la complexité devient alors $O(nm)$, et si m est négligeable, alors la complexité est en $O(n)$.

Ce type de schéma n'a d'intérêt que si la méthode de pré-sélection est rapide, tout en gardant à l'esprit qu'il faut choisir des images intéressantes à faire annoter. Parmi les critères de sélection que nous avons pu étudier, celui de la distance à la frontière semble le plus approprié. En plus d'être un critère qui fonctionne bien dans notre contexte, il a le mérite d'avoir un coût quasi nul. En effet, puisqu'à chaque itération de bouclage nous devons recalculer la pertinence de toutes les images pour pouvoir les présenter à l'utilisateur, les valeurs de la fonction de pertinence f sur la base sont déjà calculées avant la sélection. Il ne reste plus qu'à calculer la valeur absolue de toutes ces pertinences, et à trier ces valeurs pour en déduire les m images les plus proches de la frontière :

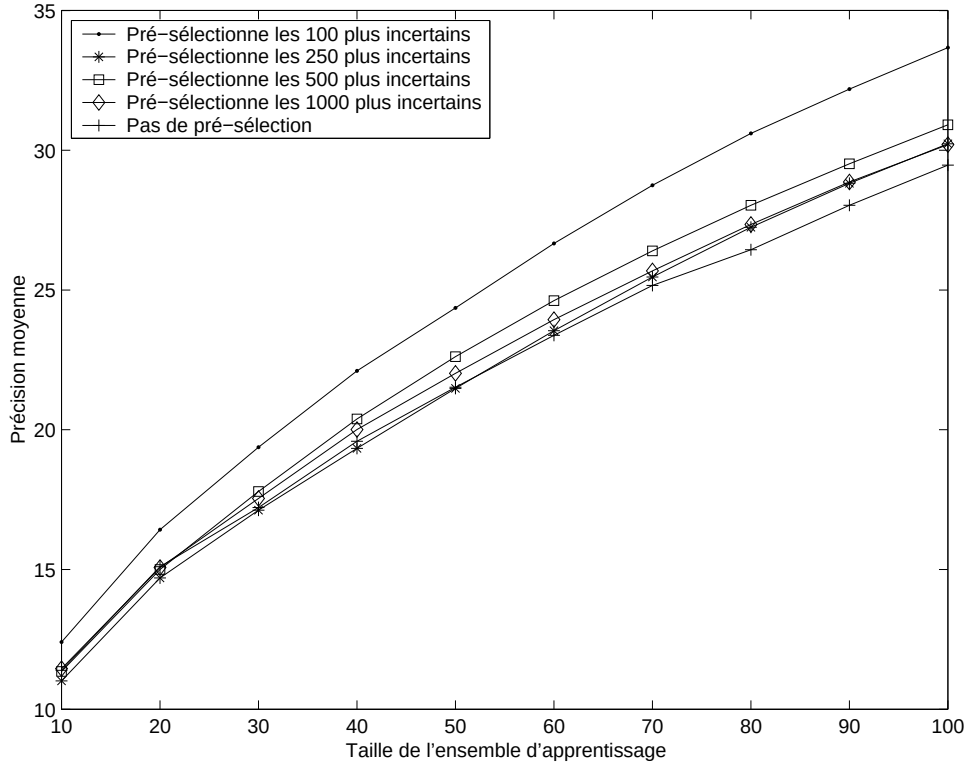
TAB. 6.3: Pré-sélection

fonction préselectioner$(f, m \rightarrow J)$
Entrées : m *Taille de la pré-sélection* f *Fonction de pertinence* Sorties : J *Pré-sélection*
$O = \text{argsort}(\lvert f \rvert)$ $J = O(1 : m)$

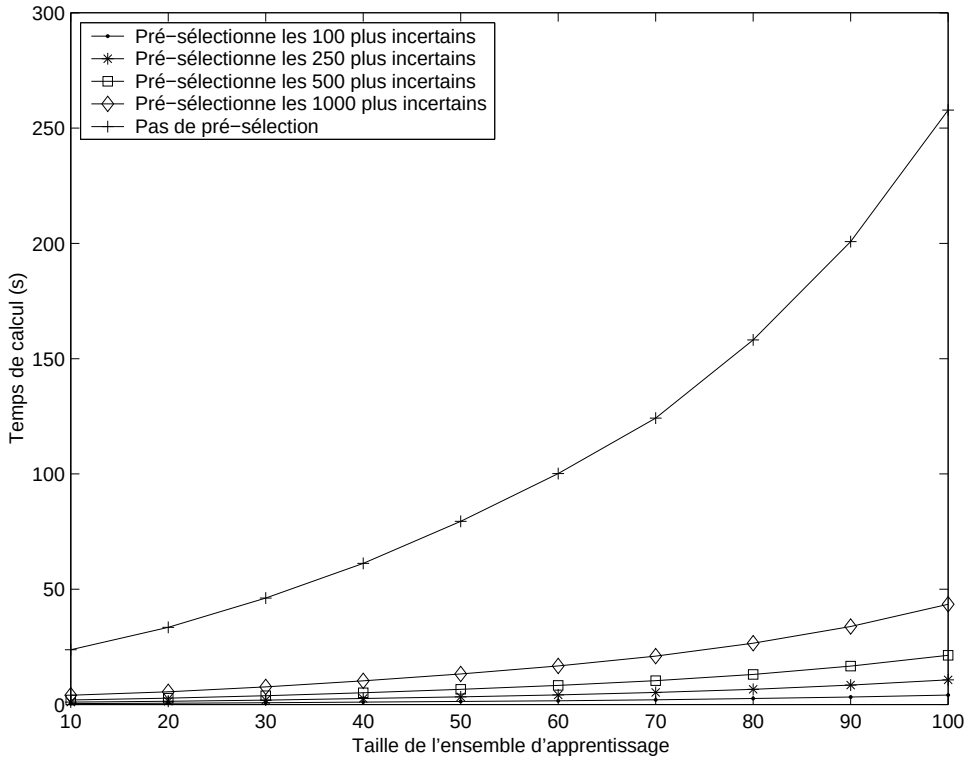
6.3.2 Influence du nombre d'images pré-sélectionnées

La pré-sélection constitue une approximation de la sélection, dont l'importance est fonction du nombre d'images pré-sélectionnées. Nous avons mené des expériences avec la méthode de (Roy & McCallum, 2001) dans le but d'observer cette influence. La technique de sélection, pour ces simulations, pré-sélectionne les m images les plus proches de la frontière, puis calcul le coût de ces images avec la méthode de (Roy & McCallum, 2001), et enfin conserve celles dont le coût est minimal.

Les résultats sont présentés sur la figure 6.3. Comme il était attendu, les temps de calculs sont réduits grâce à la pré-sélection : si nous observons la figure 6.3(b), nous pouvons constater que le temps de calcul passe de 250s avec la méthode de (Roy & McCallum, 2001) sans modifications à un temps de calcul de quelques secondes avec une pré-sélection de 100 images. Par contre, comme nous pouvons le voir sur la figure 6.3(a), les performances sont augmentées par l'approximation. Cela peut s'expliquer par le fait que le schéma combine deux approches différentes de la sélection. En effet, en commençant par pré-sélectionner les images les plus proches de la frontière, la technique s'oriente tout d'abord dans une première direction. Rappelons que nous sommes dans un contexte où la frontière est de mauvaise qualité, ce qui signifie que la frontière « idéale » se trouve quelque part autour de la frontière estimée. Si nous appliquons ensuite le critère de SVM_{active} , nous continuons à sélectionner avec la même quantité d'erreur. Cependant, en appliquant le critère de (Roy & McCallum, 2001), la technique sélectionne de manière différente.



(a) Précision Moyenne



(b) Temps de calcul moyen.

FIG. 6.3 – Précision Moyenne et temps de calcul en fonction de la taille de l'ensemble d'apprentissage avec la Méthode 2, pour différents nombres d'images pré-sélectionnées.

Ces résultats expérimentaux nous invitent à nous intéresser à la mise en opposition de deux critères de sélection, et plus particulièrement à un critère complémentaire à celui de la distance à la frontière. La frontière reste approximative, même si la correction permet d'avoir un meilleur ordre de grandeur de sa localisation. Le schéma de la pré-sélection nous permet ainsi de déterminer un ensemble d'images intéressantes à faire annoter, suivi d'une technique plus précise. La méthode que nous avons expérimentée dans ce paragraphe consiste à « sélectionner les images qui minimisent l'erreur de généralisation parmi les plus proches de la frontière ».

Nous proposons au paragraphe suivant une méthode affinant ce schéma, en utilisant la Précision Moyenne, de manière à « sélectionner les images qui maximisent la Précision Moyenne parmi les plus proches de la frontière ».

6.4 Maximiser la Précision Moyenne

Les techniques de classification actives ont été construites pour sélectionner les éléments qui améliorent la classification. Or, dans notre contexte de recherche interactive, le classement de la base en fonction de la pertinence est plus important.

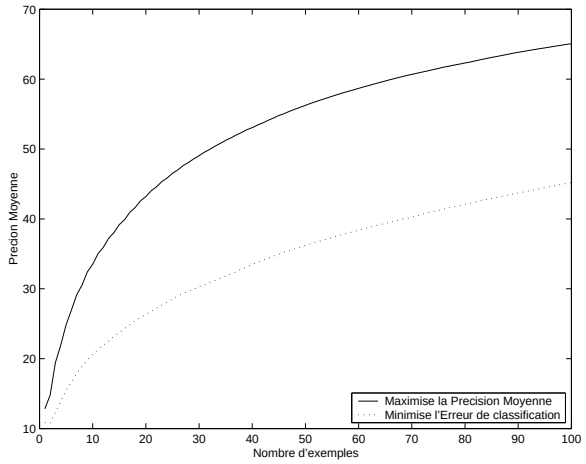
Nous proposons dans cette section de nous intéresser à la sélection des images qui améliorent directement le classement. Pour ce faire, nous proposons de formaliser la satisfaction de l'utilisateur en utilisant la Precision Moyenne, puis d'effectuer la sélection avec pour but de maximiser ce critère. Nous proposons tout d'abord d'observer les performances que l'on peut obtenir dans un cas où l'on peut calculer la véritable Précision Moyenne. Puis nous discutons des possibilités d'estimation de la Précision Moyenne, et enfin nous présentons une méthode de sélection active en ce sens.

6.4.1 Motivation

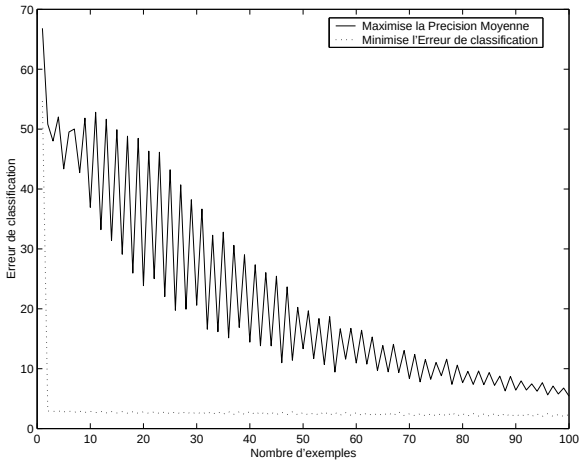
Comme il a été présenté en introduction, la Précision Moyenne est le critère que nous avons choisi pour évaluer les performances d'un système de recherche. Ce critère permet de quantifier la statisfaction de l'utilisateur, en se basant sur le classement de la base pour une catégorie donnée.

Nous proposons dans ce paragraphe d'observer le comportement d'un système si nous pouvons directement maximiser la vraie Précision Moyenne. Pour ce faire, nous supposons que nous disposons de l'appartenance de chaque image de la catégorie sous la forme d'un vecteur $\mathbf{c} = (c_1 \dots c_n)^\top$, avec $c_i = 1$ si l'image d'indice i est dans la catégorie, $c_i = -1$ sinon. Muni de cette connaissance, nous pouvons sélectionner l'image qui maximise la Précision Moyenne sur le classement avec la fonction de pertinence $f_{\mathbf{y}+c_i\mathbf{e}_i}$ entraînée sur l'ensemble d'apprentissage actuel $(\mathbf{y})$, auquel on ajoute l'annotation c_i de l'image i :

$$g_{\text{map}}(\mathbf{x}_i) = 1 - \text{précision}(f_{\mathbf{y}+c_i\mathbf{e}_i}, \mathbf{c})$$



Précision Moyenne



Erreur de classification

FIG. 6.4 – Précision Moyenne (en pourcentages) et Erreur de classification (en pourcentages) en fonction du nombre d'exemples, suivant que l'on minimise l'un ou l'autre.

Nous proposons de comparer cette sélection à une méthode de minimisation de l'erreur de classification, elle aussi dans un cas idéal, où l'on peut calculer la véritable erreur de classification :

$$g_{\text{err}}(\mathbf{x}_i) = \text{err}(f_{\mathbf{y}+c_i\mathbf{e}_i}, \mathbf{c})$$

Les deux approches sont testées sur la base COREL selon le protocole expérimental présenté en annexe A. Les résultats sont présentés sur la figure 6.4. Comme attendu, la maximisation de la Précision Moyenne donne la meilleure Précision Moyenne, et la minimisation de l'Erreur de classification donne les meilleurs taux d'erreur. Cependant, l'écart entre les résultats de ces deux approches est important. Bien que l'approche par classification donne de bons résultats, ils restent encore loin de ce qu'une technique de sélection active idéale peut fournir. C'est la raison pour laquelle nous proposons dans les sections suivantes une techniques de sélection active basée sur la maximisation de la Précision Moyenne.

Nous pouvons aussi remarquer, en l'abscence de correction et de minimisation directe de l'erreur, que l'erreur de classification est particulièrement chaotique lors de la maximisation de la Précision Moyenne, ce qui apporte une motivation supplémentaire quant à la correction de la frontière.

6.4.2 Estimer la Précision Moyenne

Construire une méthode de sélection active basée sur la maximisation de la Précision Moyenne revient à bien estimer cette Précision Moyenne. En effet, pour le schéma présenté dans la section précédente, nous disposions de la vérité terrain. Or, en utilisation réelle cette information n'est pas disponible, et doit être estimée.

Une telle estimation est particulièrement difficile. En effet, rappelons que le calcul de la Précision Moyenne nécessite deux informations :

– L'appartenance de chaque image à la catégorie ;
– La taille de la catégorie.

L'approche par classification offre une estimation de ces informations. La fonction d'appartenance peut nous donner une estimée de l'appartenance de chaque image, et le nombre d'images d'appartenance positive, la taille de la catégorie.

Il est donc naturel de penser que la Précision Moyenne peut s'estimer à l'aide de la fonction d'appartenance. Cependant, après expérimentation, une telle approche approxime très mal voire pas du tout la vériable Précision Moyenne. Cela peut s'expliquer par le fait que l'estimation de ce critère, à l'aide de la fonction d'appartenance, repose énormément sur le passage à zéro (problème que nous avons décrit précédemment)

6.4.3 Méthode

A défaut de pouvoir bien estimer la Précision Moyenne, nous proposons une méthode alternative dans le sens de la maximisation de la Précision Moyenne.

Cette méthode est basée sur la sélection des images les plus difficiles à classifier, *i.e.* celles qui sont les plus proches de la frontière. L'idée est la suivante : compte tenu de l'imprécision de la frontière, sélectionner l'image la plus proche relève d'une sélection aléatoire parmi les images les plus proches de la frontière. Nous proposons de biaiser cette sélection aléatoire en choisissant parmi les images les plus proches, celle qui maximise la Précision Moyenne.

Si nous considérons l'image la plus proche de la frontière, nous avons constaté qu'un faible changement dans le classifieur peut rapidement la changer. Rappelons que nous sommes dans un contexte où le nombre d'exemples est très faible. Ainsi, une faible modification dans les paramètres du classifieur aurait très problablement sélectionné une autre image. Cependant, à moins de changer très fortement les paramètres, l'image sélectionnée sera toujours parmi les images les plus proches pour un lot de paramètres donné. On peut voir ce type de sélection comme une sélection aléatoire parmi les images les plus proches, et dont l'estimée de la probabilité de sélection dépend de la distance à la frontière ($|f(\mathbf{x})|$).

Nous proposons de biaiser cette sélection aléatoire par un processus plus déterministe, en sélectionnant l'image qui est à la fois proche de la frontière et offre de bonnes chances d'augmenter la Précision Moyenne. Comme nous l'avons évoqué précédemment, estimer la Précision Moyenne pour une classification donnée est une tâche difficile. Cependant, nous pouvons facilement calculer une Précision Moyenne indépendemment de toute classification en considérant uniquement la sous-base formée par l'ensemble des images annotées.

En utilisant la similarité d'une image non annotée à toutes les images annotées, nous pouvons en déduire un classement de la "sous-base annotée", et compte tenu du fait que nous connaissons la classe de toute les images de cette sous-base, nous pouvons calculer la

Précision Moyenne. Cela nous donne une valeur qui estime la Précision Moyenne que nous pouvons avoir sur toutes les images de la base. Cela permet d'introduire une pondération dans la fonction de coût qui stabilise la sélection :

$$g(\mathbf{x}_i) = |f(\mathbf{x}_i)| \times (1 - \text{précision}_I(k(\mathbf{x}_i, .), \mathbf{y}))$$

avec map_I le calcul de la Précision Moyenne restreint à l'ensemble des images annotées.

TAB. 6.4: Sélection orientée Précision Moyenne

fonction sélectioner$(k, \mathbf{y}, f, J \rightarrow g)$
Entrées : k *Fonction noyau* $\mathbf{y}$ *Annotations* f *Fonction de pertinence* J *Pré-sélection* Sorties : g *Fonction de coût*
pour $i \in J$ $g(\mathbf{x}_i) = \|f(\mathbf{x}_i)\| \times (1 - \text{précision}_I(k(\mathbf{x}_i, .), \mathbf{y}))$ **finpour**

6.5 Diversification

Comme nous l'avons présenté au §6.1.2, nous avons choisi une approche de sélection du lot $I^\star$ d'images à faire annoter par l'utilisateur en s'appuyant sur la fonction de coût g et un critère de diversification. L'objectif est de choisir un ensemble d'images éloignées, tout en s'assurant qu'elles sont intéressantes à annoter.

Nous proposons ici deux techniques de diversification. La première repose sur un clustering de la pré-sélection J, et la deuxième est une généralisation de la méthode de (Brinker, 2003).

6.5.1 Clustering

Cette méthode commence par partionner la pré-sélection J en q clusters, puis sélectionne dans chaque cluster l'image qui minimise la fonction de coût g (Gosselin & Cord, 2004b) :

TAB. 6.5: Diversification par clustering

fonction diversifier$(k, g, J \rightarrow I^\star)$
...

...
Entrées :
$k(\mathbf{x}_i, \mathbf{x}_j)$ *Fonction noyau*
g *Fonction de coût*
J *Pré-sélection*
Sorties :
$I^\star$ *Lot d'images à faire annoter*
$\{Q_1, \ldots, Q_q\} = \text{quantifier}(k(J, J), q)$
$I^\star = \{\}$
pour $l \in [1..q]$
$i^\star = \underset{i \in Q_l}{\text{argmin}}\ g(\mathbf{x}_i)$
$I^\star = I^\star \cup \{i^\star\}$
finpour

Cela permet d'assurer une certaine diversité des images sélectionnées. Cependant, cette méthode souffre de quelques défauts. Premièrement, elle dépend du nombre d'images à répartir en clusters. Ceci ajoute un paramètre à régler pour que la méthode fonctionne. Deuxièmement, elle autorise la sélection d'images proches d'images déjà annotées, particulièrement dans le cas où la frontière est proche du coeur de la classe.

6.5.2 Angle diversity généralisé

Afin d'assurer une meilleure diversification du lot, la méthode de (Brinker, 2003) nous semble plus appropriée. Elle n'a pas les défauts que nous avons cités à propos de la méthode par clustering. Cependant, cette méthode telle qu'elle est proposée se limite à une sélection entièrement basée sur la distance à la frontière. Nous proposons de généraliser cette méthode en offrant la possibilité d'utiliser un autre critère. Dans l'algorithme de (Brinker, 2003), cela revient à remplacer $|f(\mathbf{x})|$ par une fonction de coût $g(\mathbf{x})$ (Gosselin & Cord, 2005a) :

TAB. 6.6: Angle diversity généralisé

fonction diversifier$(k, g, J \rightarrow I^\star)$
Entrées :
$k(\mathbf{x}_i, \mathbf{x}_j)$ *Fonction noyau*
g *Fonction de coût*
J *Pré-sélection*
Sorties :
$I^\star$ *Lot d'images à faire annoter*
$I^\star = \{\}$
pour $l \in [1..q]$
$i^\star = \underset{i \in J - I^\star}{\text{argmin}} \left(g(\mathbf{x}_i) + \max_{j \in I \cup I^\star} s(\mathbf{x}_i, \mathbf{x}_j) \right)$
...

```
...
    I* = I* ∪ {i*}
finpour
```

La fonction $s(\mathbf{x}_i, \mathbf{x}_j)$ est la fonction de similarité induite par la fonction noyau.

6.6 Trois méthodes d'apprentissage actif

Chacune des solutions que nous avons présentées s'intègrent dans l'un des blocs de l'architecture globale (*cf.* fig. 6.1). Ces solutions nous permettent de proposer plusieurs stratégies d'apprentissage actif.

6.6.1 Première méthode

La première méthode que nous proposons (Gosselin & Cord, 2004b) est une version améliorée de SVM_{active} :

- *Correction :* Corrige la frontière ;
- *Fonction de coût :* Distance à la frontière ;
- *Diversification :* Par clustering.

Nous utilisons le même critère de sélection que SVM_{active} , à savoir la distance à la frontière. Nous ajoutons à cela une correction de la frontière qui permet d'obtenir une meilleure frontière lors des premières itérations. La diversification utilisée est la méthode par clustering présentée en §6.5.1.

6.6.2 Deuxième méthode

La deuxième méthode que nous proposons (Gosselin & Cord, 2005a) est une version modifiée de (Roy & McCallum, 2001) :

- *Correction :* Corrige la frontière ;
- *Présélection :* Pré-sélectionne les 100 images les plus proches de la frontière ;
- *Fonction de coût :* Erreur de généralisation ;
- *Diversification :* Angle diversity généralisé.

Nous utilisons la correction de la frontière pour les mêmes raisons que précédemment. Cette méthode diffère principalement par l'étape de pré-sélection qui permet de donner une première orientation « proche de la frontière » avant de sélectionner les images qui minimisent l'erreur de généralisation. En plus de diminuer considérablement le temps de calcul, cela permet de proposer une alternative au choix délicat des images autour de la frontière. En effet, même avec une correction, la frontière reste très approximative. L'idée

de cette méthode est d'utiliser le critère de minimisation de l'erreur de généralisation pour sélectionner plus précisemment autour de cette frontière. Nous utilisons dans cette méthode la généralisation de la méthode de (Brinker, 2003)

6.6.3 Troisième méthode

La troisème méthode que nous proposons est dans le même esprit que la précédente, à la différence que nous avons remplacé le critère de minimisation de l'erreur de généralisation par celui de la maximisation de la Précision Moyenne :

- *Correction :* Corrige la frontière ;
- *Présélection :* Pré-sélectionne les 100 images les plus proches de la frontière ;
- *Fonction de coût :* Distance à la frontière pondérée par la Précision Moyenne ;
- *Diversification :* Angle diversity généralisé.

Nous proposons au paragraphe suivant une évaluation du système proposé ainsi qu'une comparaison des différentes méthodes.

6.7 Expérimentations

Nous présentons dans ce paragraphe des résultats d'utilisation du système RETIN 2, avec la configuration suivante :

- La base d'images utilisée est l'extrait de COREL présentée en annexe A ;
- Les signatures sont des histogrammes de 25 couleurs et 25 textures, qui ont été réduits par un processus de quantification vectorielle (*cf.* chapitre 2) ;
- La fonction noyau utilisée est une gaussienne avec une distance du χ^2 (*cf.* chapitre 3) ;
- Les SVM sont utilisées pour la classification (*cf.* chapitre 4) ;
- Nous utilisons les 3 méthodes actives que nous proposons, ainsi que les deux méthodes de référence (SVM_{active} et (Roy & McCallum, 2001)). Par défaut, la Méthode 3 est utilisée.

Notons que l'utilisateur du système a la possibilité à tout moment de changer un ou plusieurs de ces paramètres.

Nous présentons tout d'abord un exemple de fonctionnement d'une session de recherche. Le paramétrage du système est ensuite discuté. Enfin, nos stratégies actives sont comparées à plusieurs méthodes de références.

6.7.1 Fonctionnement d'une session de recherche

Un utilisateur peut rechercher une catégorie d'images à l'aide de l'interface graphique RETIN 2, dont une saisie d'écran est présentée en figure 6.5. L'interface se décompose en trois parties principales. La première, qui prend la majeure partie de l'écran, présente une partie du classement de la base par le système. Par exemple, sur la figure 6.5(a) nous pouvons voir les 25 images les plus pertinentes selon le système. Le classement est fait de gauche à droite puis de haut en bas. L'image de rose avec un petit carré vert est l'image la plus pertinente, puis celle à sa droite est la deuxième plus pertinente, etc. La partie à droite du classement de la base présente les informations relatives à toute image sélectionnée en cliquant avec le bouton central de la souris. Nous pouvons voir l'image en question sur le haut (en l'occurence, une rose vue de profil), ainsi qu'un ensemble d'informations relatives au système. Enfin, la partie inférieure présente les images sélectionnées par la technique d'apprentissage actif. Sur la figure 6.5, le système a été configuré pour sélectionner 5 images. L'utilisateur fournit ses annotations en cliquant sur les images. Les annotations sont représentées par des surimpressions de carrées vert (resp. rouge) pour les annotations positives (resp. négatives).

Nous présentons sur les figures 6.5, 6.6, 6.7, 6.8, 6.9 et 6.10 les différentes étapes d'une session de recherche de la catégorie *Roses* de la base COREL.

L'utilisateur a initialisé la requête en apportant une image de rose. Le système a alors annoté positivement l'image la plus proche dans la base, puis a classé les images en fonction de leur similarité visuelle à cette image requête. Sur la figure 6.5(a), nous pouvons voir ce classement : l'image en haut à gauche avec un petit carré vert est l'image annotée positivement, et les images qui suivent sont ses plus proches voisines.

L'utilisateur, afin de raffiner sa requête, annote les 5 images dans la partie inférieure de l'interface. Les annotations sont reportées automatiquement dans la partie supérieure de l'interface. Notons qu'à cette étape, ces annotations ne sont pas encore prises en compte pour le classement de la base, qui est le même qu'en figure 6.5(a). Ce double affichage permet cependant de voir que le système a sélectionné les cinq images les plus pertinentes de la base.

Une fois les annotations données, l'utilisateur demande une mise à jour du classement. Compte tenu du fait qu'il n'y a qu'un seul type d'annotation, RETIN estime dans ce cas la densité de la classe pertinente, puis calcule pour chaque image sa pertinence. Le nouveau classement est présenté en figure 6.6(a), où l'utilisateur a déjà donné ses annotations sur les 5 nouvelles images sélectionnées. De même, ces annotations se sont pas encore prises en compte, et sont affichées deux fois : une fois dans la barre de sélection, et une fois dans la partie principale. A ce stade, nous pouvons observer que le système a sélectionné des images parmi les plus pertinentes. La sélection active est limitée puisqu'il n'y a pas encore eu d'étape de classification.

Grâce à la première annotation négative que l'utilisateur a donné en deuxième itération, RETIN a calculé la première classification de la base. Cela permet tout d'abord d'éliminer des images qui ne sont pas des roses de la tête du classement. Nous pouvons voir sur la figure 6.6(b) que les 25 premières images du classement ne sont que des roses.

Dans le but d'illustrer davantage, nous présentons sur la figure 6.7 les 75 première images du classement. Nous retrouvons au début les 25 images de la figure 6.6(b), suivies d'images de moins en moins pertinentes. Nous pouvons voir que, bien qu'ayant trouvé le premier groupe d'images de la catégorie, nous sommes encore loin d'avoir en tête de classement les 500 images de roses que contient la base.

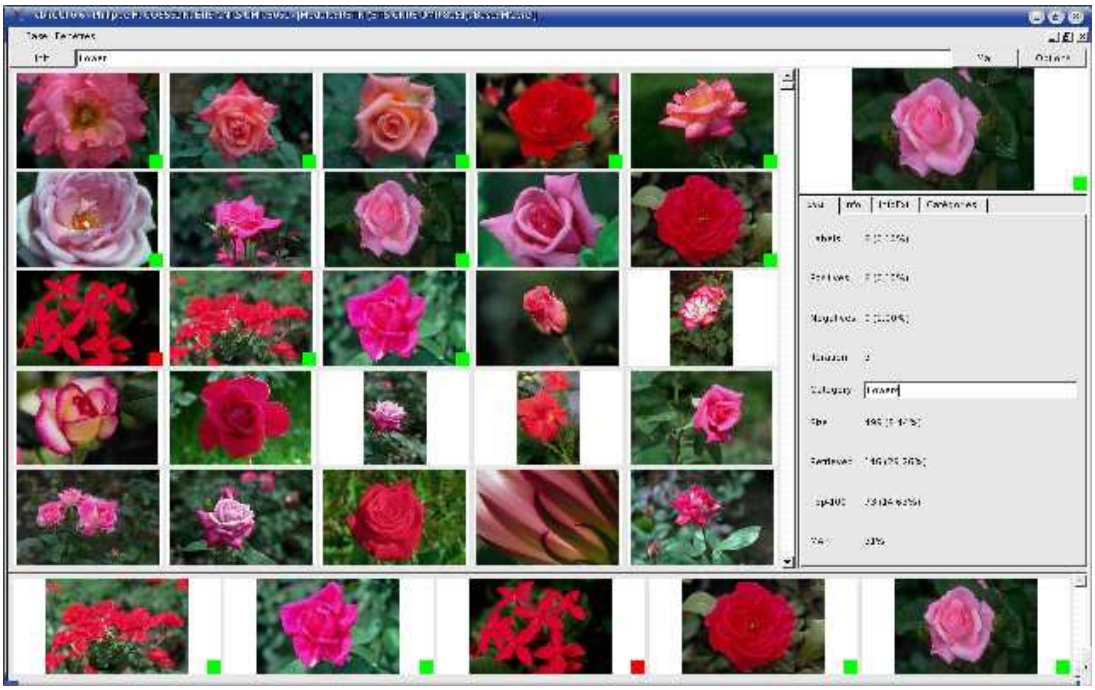
Cette figure permet aussi d'apprécier le fonctionnement de la sélection active. En effet, l'image en bas à droite de la figure 6.7 est une des images sélectionnées sur la figure 6.6(b). Nous ne voyons pas les autres images sélectionnées car elles sont au delà des 75 premières. Cela montre que la frontière calculée par RETIN est dans cette zone, relativement éloignée du cœur de la catégorie pour un début de session. Du fait des nombreuses annotations positives passées par l'utilisateur, le processus de correction a déplacé assez loin dans le classement le passage par zéro de la fonction de pertinence.

Les figures 6.8, 6.9 et 6.10 présentent le classement de la base aux itérations suivantes. De même, nous retrouvons les annotations positives prises en compte pour le classement tout en haut, et des annotations non prises en compte disséminées qui correspondent aux annotations de l'utilisateur dans la barre inférieure de l'interface. D'une manière générale, nous pouvons apprécier que le classement de tête s'améliore d'itération en itération.

(a) Initialisation avec une image requête.

(b) Annotations : les images sélectionnées par le système ont été annotées dans la fenêtre du bas.

Fig. 6.5 – Exemple d'une session de recherche. Première itération.



(c) Deuxième itération.

(d) Troisième itération.

FIG. 6.6 – Exemple d'une session de recherche. Deuxième et troisième itérations.

Fig. 6.7 – Exemple (suite) : Classement des 75 premières images à la troisième itération.

FIG. 6.8 – Exemple (suite) : Classement des 75 premières images à la quatrième itération.

Fig. 6.9 – Exemple (suite) : Classement des 75 premières images à la cinquième itération.

FIG. 6.10 – Exemple (suite) : Classement des 75 premières images à la sixième itération.

6.7.2 Paramétrage

Tous les choix de paramètres ou de méthodes associés au RETIN 2 ont été discutés et souvent évalués au fur et à mesure :

- *Taille de la base.* Pour toutes les méthodes de classification (non semi-supervisées) et pour toutes les méthodes de sélection hormis (Roy & McCallum, 2001), le temps de calcul ne dépend pas de la taille de la base, mais du nombre d'annotations dans l'ensemble d'apprentissage. Le principal calcul dans le processus de mise à jour est le calcul de la pertinence des n images, tout autre calcul est négligable, hormis dans les cas particuliers cités ci-dessus. Le système RETIN 2 est donc en mesure de traiter les très grandes bases avec une complexité globale en $O(n)$.
- *Taille des signatures.* Nous avons calculé l'erreur de classification d'un SVM en fonction de la taille des signatures au chapitre 2. Il en ressort qu'une taille de 50 fournit les meilleurs résultats.
- *Fonction noyau.* Nous avons calculé l'erreur de classification d'un SVM avec différentes fonctions noyaux au chapitre 3. Il s'est avéré qu'une gaussienne avec une distance du χ^2 est la plus adaptée à notre contexte.
- *Méthode de classification.* Nous avons calculé l'erreur de classification de plusieurs classifieurs au chapitre 4. Bien que la plupart des méthodes fournissent une Précision Moyenne comparable, il en ressort que la classification SVM est la plus intéressante compte tenu de sa rapidité et de la faible erreur de classification qu'elle fournit, aspect important pour la sélection active.
- *Pré-sélection.* Le nombre d'images pré-sélectionnées a été discuté au §6.3.
- *Méthode de sélection d'une image.* Cet aspect est le sujet du paragraphe suivant.
- *Méthode de diversification.* Nous avons comparé les deux méthodes (clustering et Angle Diversity généralisé) avec différentes méthodes de sélection d'une image. Dans l'ensemble, les deux méthodes ont une influence comparable, et contribuent toutes deux à améliorer les performances. Notons toutefois que la méthode par Angle Diversity généralisé est légèrement plus efficace.

Il reste à discuter des paramètres liés au protocole d'utilisation du système :

- Le nombre d'annotations par itération de bouclage ;
- Le nombre de bouclages par session de recherche.

La taille de l'ensemble d'apprentissage à la fin d'une session de recherche est un troisième paramètre fonction de ces deux premiers. Il est avant tout majoré par le nombre maximal d'annotations qu'un utilisateur est prêt à fournir au système. Nous pensons qu'une centaine d'annotations est un nombre raisonnable. D'une manière générale, plus ce paramètre est élevé, plus la Précision Moyenne est élevée, et le gain est augmenté de manière régulière (surtout avec l'approche active). Au cours des nombreuses expérimentations que nous avons ménées, nous n'avons jamais remarqué de discontinuitées importantes dans l'évolution de la Précision Moyenne en fonction du nombre d'annotations. Notons, bien que cet aspect ne soit pas important pour une application réelle, que si nous poussons le nombre d'annotations jusqu'à une valeur très élevée (1.000 annotations), alors la Précision Moyenne évolue de manière logarithmique, et finit par progresser avec

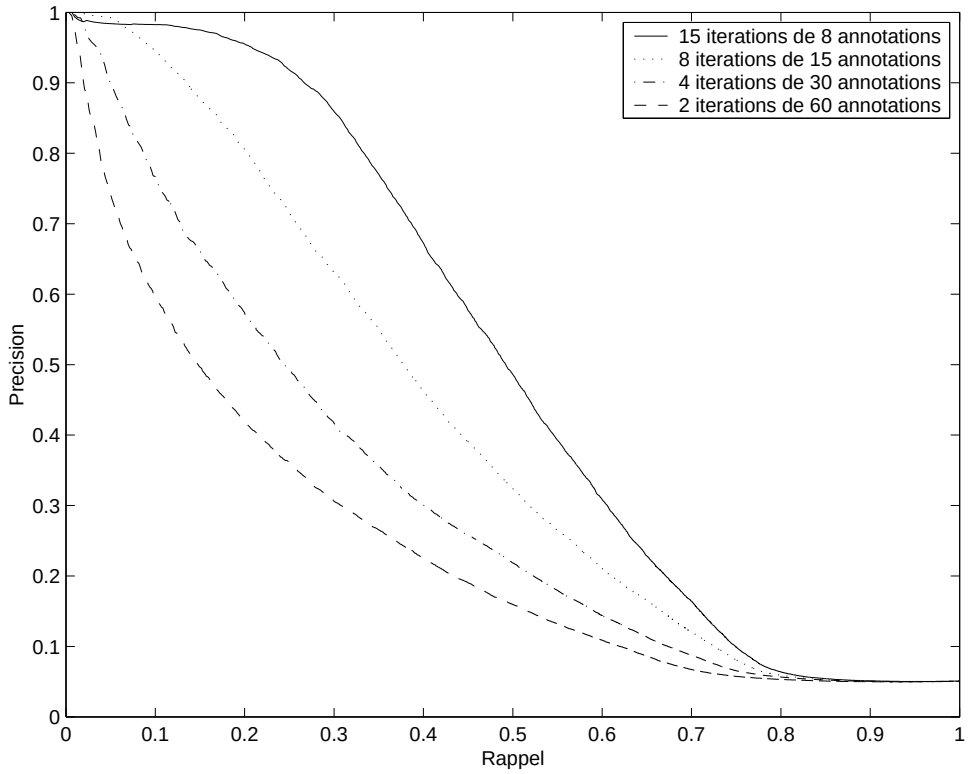


FIG. 6.11 – Courbes de Précision/Rappel pour la catégorie « savane », avec différents nombre d'annotations par itération. Chaque session débute avec une image de la catégorie, et dans tous les cas, à la fin de la session, il y a 121 annotations.

beaucoup plus de difficultées au dela de plusieurs centaines d'annotations. Il semble donc qu'il soit difficile pour le système de retrouver des images de la catégorie très éloignées du point de vue des signatures. Ce résultat est une des premières motivations quant à l'utilisation d'un apprentissage long terme que nous développons en partie III.

Concernant l'importance du nombre d'annotations que donne l'utilisateur à chaque itération de bouclage, nous avons simulé l'utilisation du système en suivant le protocole en annexe A, mais en changeant le nombre d'annotations par itération. Dans le but d'avoir des résultats comparables, nous avons imposé que la taille de l'ensemble d'apprentissage soit toujours la même en fin de session. Nous avons choisi les couples suivantes, qui fournissent toujours un ensemble d'apprentissage final de 121 images en comptant la requête initiale :

- 15 bouclages par session et 8 itérations par bouclage ;
- 8 bouclages par session et 15 itérations par bouclage ;
- 4 bouclages par session et 30 itérations par bouclage ;
- 2 bouclages par session et 60 itérations par bouclage ;

Nous avons ensuite calculé les courbes de précision/rappel pour chaque catégorie de la base dans toutes les configurations. Les résultats pour la catégorie « savane » sont présentés en figure 6.11 ; notons que toutes les catégories de COREL ont produit des courbes similaires. Le résultat est très clair : plus on fait de bouclage, tout en réduisant le nombre d'annotations par bouclage, meilleures sont les performances. Cela s'explique

par le fait que plus on multiplie le nombre de bouclages, plus la classification est remise en cause ; à chaque remise en cause, la correction est meilleure, la frontière est un peu plus précise et donc la sélection aussi. Ainsi, lorsque l'on sélectionne 2 fois 4 images, les 4 premières dépendent d'une première classification, alors que les 4 suivantes dépendent d'une deuxième classification, meilleure que la première. A l'inverse, lorsque l'on sélectionne 8 images en une seule fois, ces 8 images ne dépendent que de la première classification.

Bien que l'étape de diversification ait pour but de sélectionner un lot d'images, ses performances semblent limitées lorsque la taille du lot de sélection devient important. Dans un cas d'utilisation réelle, il est donc préférable de limiter le nombre d'annotations par bouclage. Cependant, il faut prendre en compte d'autres paramètres comme la puissance de calcul, pour ne pas lasser l'utilisateur.

6.7.3 Comparaisons

Nous présentons dans ce paragraphe les résultats des simulations ménées sur la base COREL. Nous simulons 1.000 utilisations du système pour chaque méthode d'apprentissage actif. A chaque itération de ces sessions de recherche simulées, nous calculons la Précision Moyenne pour le classement de la base, et le temps de calcul de la mise à jour (classification, calcul des pertinences et sélection). Ces résultats par itération et par méthode active sont ensuite moyennés sur les 1.000 simulations. Notons que les performances ne sont pas calculées catégorie par catégorie. Nous avons suivi un protocole où à chaque nouvelle session de recherche simulée, une catégorie est choisie aléatoirement, dans le but de simuler une utilisation réelle du système, où les utilisateurs arrivent les uns après les autres, chacun à la recherche de l'une des catégories. Les détails de ce protocole sont présentés en annexe A.

Cinq méthodes de classification active sont comparées :

- SVM_{active} (présentée au §5.3.1) : Sélection proposée par (Tong & Chang, 2001) ;
- Minimisation de l'erreur de généralisation (présentée au §5.3.2) : Sélection proposée par (Roy & McCallum, 2001) ;
- RETIN Méthode 1 (présentée au §6.6.1) : Correction de la frontière, sélection des plus proches de la frontière, et diversification par clustering ;
- RETIN Méthode 2 (présentée au §6.6.2) : Stratégie de (Roy & McCallum, 2001) modifiée par une pré-sélection, une correction de la frontière, et l'ajout d'une stratégie de diversification ;
- RETIN Méthode 3 (présentée au §6.6.3) : Correction de la frontière, sélection des images qui maximisent la Précision Moyenne, et diversification par *Angle Diversity* généralisé ;
- Méthode sans apprentissage actif : Approche non active, qui sélectionne aléatoirement des images de la base.

La figure 6.12 présente les résultats de ces simulations en terme de Précision Moyenne, et la figure 6.13 en terme de temps de calcul. Les temps de calcul pour la méthode par

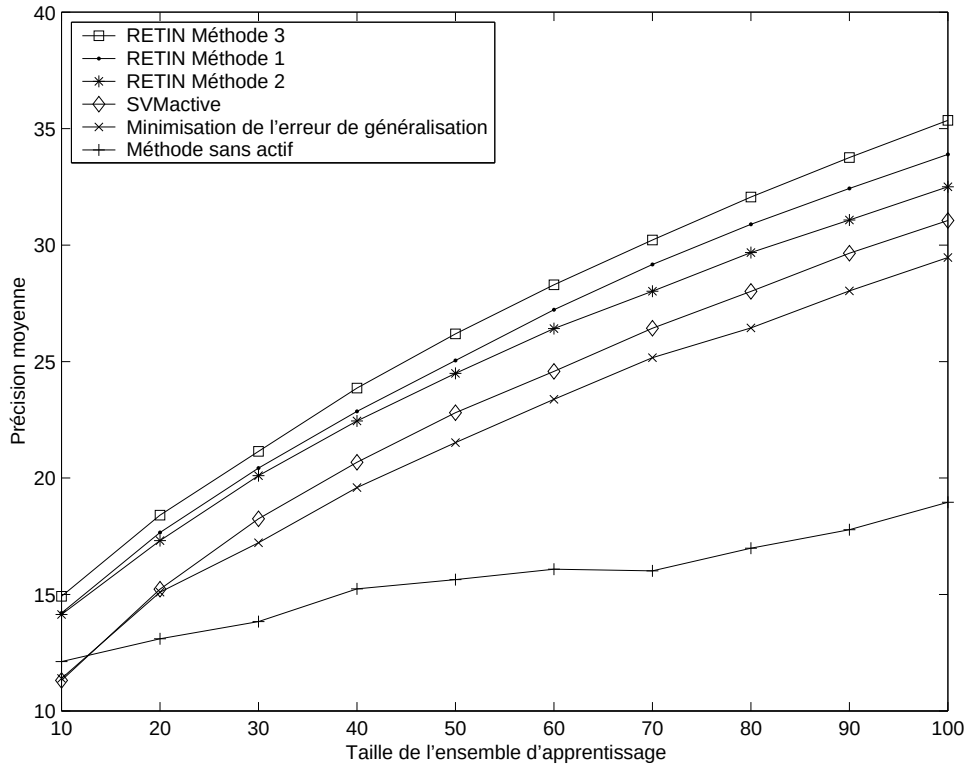


FIG. 6.12 – Précision Moyenne (en pourcentages) en fonction des différentes méthodes actives.

minimisation de l'erreur de généralisation et la Méthode 2 ne sont pas représentés sur la figure 6.13, car ils sont trop importants.

Les méthodes se distinguent clairement les unes des autres par leurs performances. Par ordre décroissant de Précision Moyenne, nous avons : la méthode 3, la méthode 1, la méthode 2, SVM_{active} , minimiser l'erreur et enfin pas de stratégie active.

Ces résultats mettent tout d'abord en valeur l'intérêt de l'approche active. En effet, comme nous pouvons le voir sur la figure 6.12, toutes les stratégies actives ont des performances bien meilleures que l'approche non active.

Les trois méthodes proposées sont au dessus des deux méthodes de référence que nous avons choisies, SVM_{active} et la méthode de (Roy & McCallum, 2001). La différence se situe principalement en début de session. En effet, pour 10 annotations, les méthodes proposées ont déjà 3-4% de Précision Moyenne de plus que les méthodes de référence. Cela résulte en partie de la correction active de la frontière, qui est présente dans nos trois méthodes. Sans cette correction, la frontière est mauvaise, et pénalise les méthodes de référence. L'autre facteur important est certainement le critère de sélection : la proximité à la frontière pour la Méthode 1, l'erreur de généralisation autour de la frontière pour la Méthode 2, et la Précision Moyenne autour de la frontière pour la Méthode 3. A la vue de ces résultats, le dernier critère semble être le plus pertinent.

Au niveau du temps de calcul, les deux méthodes orientées minimisation de l'erreur

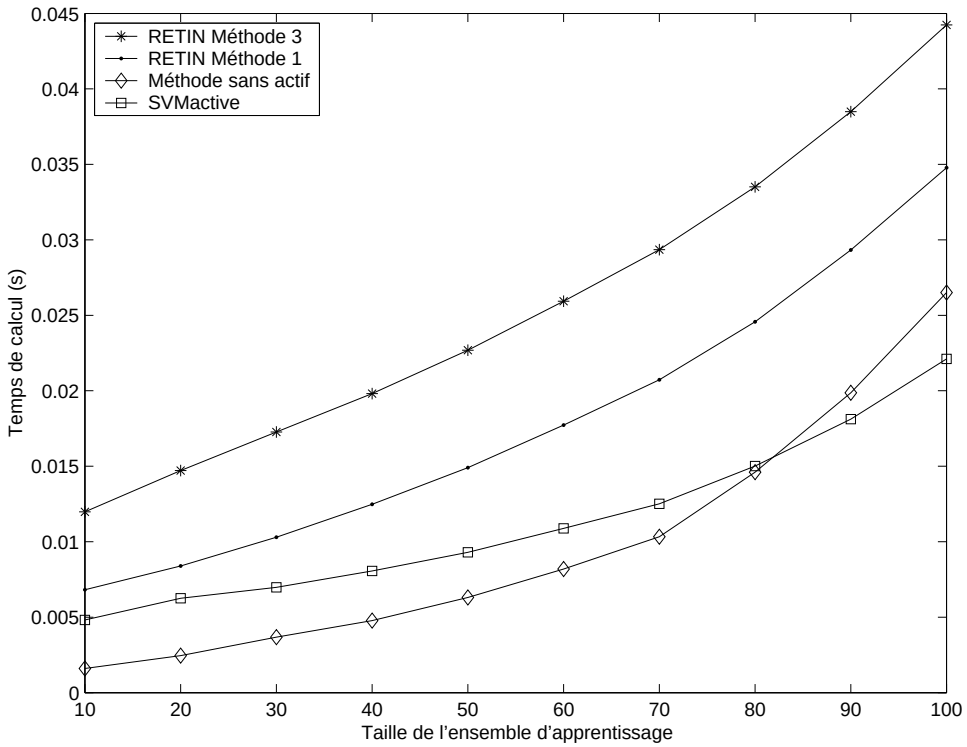


FIG. 6.13 – Temps de calcul (en secondes) en fonction des différentes méthodes actives. La méthode de (Roy & McCallum, 2001) et la Méthode 2 ne sont pas représentées pour des raisons de lisibilité. Le temps de calcul pour la dernière itération de la première méthode est de 250s, et de 4s pour la deuxième.

se distinguent des autres. Elles ont des temps de calcul de plus d'une seconde, alors que toutes les autres ont des temps de calculs de moins de 45 millisecondes. Notons que la méthode sans apprentissage actif a un temps de calcul qui augmente plus rapidement que les autres. Cela est le résultat des « effets de cache » : lors d'une session de recherche, les éléments précédemment annotés sont « cachés » (déjà en mémoire), alors que dans un processus de classification pour une taille fixe, aucun élément n'est « caché » avant le calcul.

6.8 Conclusion

Les résultats de ce chapitre montrent l'intérêt de la classification active pour la recherche interactive d'images. Nous avons présenté nos propositions pour pallier aux limitations des méthodes existantes. Ces solutions nous permettent de proposer des méthodes d'apprentissage actif rapides et efficaces, et tout particulièrement la Méthode 3 qui allie une correction active de la frontière, une sélection des images orienté Précision Moyenne, et une diversification de la sélection. Ces méthodes sont donc clairement utilisables dans un cadre réel sur de grandes bases d'images.

De plus, nous proposons une architecture et des techniques d'apprentissage afin d'adapter davantage la classification active au contexte de cette thèse :

1. *Fossé sémantique* et *Grande dimension*. Nous avons mis en avant l'intérêt des fonctions noyaux pour traiter les non-linéarités qui découlent de ces deux caractéristiques.
2. *Peu de données d'apprentissage*. Cette caractéristique est bien gérée par les techniques de classification actuelles qui, avec peu d'informations supervisées, permettent d'obtenir des résultats significatifs.
3. *Apprentissage interactif*. Cette partie montre l'intérêt de la classification active pour gérer l'apprentissage interactif entre le système et l'utilisateur. Les résultats expérimentaux montrent le gain que l'on peut obtenir en optant pour une telle stratégie. De plus, le modèle de diversification que nous proposons permet de sélectionner un lot d'images en utilisant différentes techniques de sélection simple.
4. *Déséquilibre pertinent/non pertinent*. Les techniques de classification binaire sont en général proposées par leurs concepteurs dans le cadre de la division en parts égales d'une base. Il en résulte une mauvaise estimation de la frontière. Nous avons proposé une méthode de correction active de la frontière qui permet de stabiliser la collecte des nouveaux exemples, tout en améliorant les performances du système.
5. *Satisfaction de l'utilisateur*. Les techniques de classification active « classiques » ont pour vocation la minimisation de l'erreur de classification. Or, dans notre contexte, la Précision Moyenne prévaut. Nous avons proposé une technique de sélection qui permet de sélectionner les images à faire annoter en favorisant les images qui augmentent la Précision Moyenne. Les résultats expérimentaux ont montré l'efficacité de cette méthode.
6. *Rapidité*. Le coût en calculs du système de recherche remplit le cahier des charges ($O(n)$), grâce à la techniques de pré-sélection que nous avons proposée, tout en conservant les performances. La majorité du calcul réside alors dans la détermination de l'appartenance de chaque image, tous les autres calculs étant négligeables devant celui-ci.

Troisième partie

Long terme

Chapitre 7

Apprentissage Long terme

Généralement, les systèmes de recherche d'images ne réutilisent pas les informations fournies par l'utilisateur lors de la recherche interactive. Cependant, avec la diffusion de systèmes de recherche sur les réseaux, une grande quantité d'informations relatives à leur utilisation devient disponible. A l'aide cette connaissance fournie par de nombreux utilisateurs, il devient envisageable d'exploiter ces informations passées afin d'améliorer la qualité des recherches. Par exemple, si plusieurs utilisateurs ont recherché une même catégorie, rassembler leurs annotations pour mieux décrire cette catégorie semble particulièrement intéressant pour tous les futurs amateurs de cette catégorie. Ces nouveaux utilisateurs pouraient en effet retrouver beaucoup plus vite les images qu'ils recherchent.

Il existe différentes manières d'envisager l'apprentissage long terme. Nous nous plaçons dans cette thèse dans un cadre généraliste. L'idée est de concevoir des solutions autonomes, qui améliorent la qualité du système sans aucune intervention des utilisateurs, sinon leurs contributions lors des sessions de recherche passées.

Nous présentons dans ce chapitre ce contexte d'apprentissage avant d'introduire dans les chapitres qui suivent deux méthodes d'apprentissage long terme.

7.1 Introduction

7.1.1 Présentation du contexte faiblement supervisé

L'objectif de l'apprentissage long terme que nous abordons est de retouver l'ensemble des catégories de la base qui sont recherchées par les utilisateurs. Chacune de ces catégories est un sous-ensemble de la base, formé d'images d'un même type, par exemple toutes les voitures, tous les paysages de montagne, *etc.*

Il est courant de s'intéresser à la détection de catégories qui forment une partition. Dans ce cas, chaque individu appartient exclusivement à une seule catégorie. On peut

toutefois donner un degré d'appartenance de chaque image à une catégorie, mais toujours dans l'idée que les images n'appartiennent au final qu'à une seule catégorie.

Dans cette thèse, nous nous intéressons à un problème d'apprentissage long terme de catégories d'images sur une base, mais nous ne contraignions pas ces catégories à former une partition stricte. Lorsque nous parlons de *catégories mélangées*, nous supposons que chaque image peut appartenir à plusieurs catégories. Un image avec un homme et un chien appartient à la catégorie « homme » et à la catégorie « chien », et peut aussi appartenir à la catégorie « un chien et son maître ». De plus, nous supposons qu'il y a une appartenance stricte pour chaque image à une catégorie aux yeux des utilisateurs : une image appartient ou n'appartient pas à une catégorie.

Lorsque l'on connaît le type de catégories qui vont être recherchées, il est possible de construire un détecteur pour chacune d'entre elles à l'aide de cette connaissance. Par exemple, on peut savoir que les utilisateurs vont rechercher des voitures, des paysages, des animaux, ... mais ne connaître l'appartenance à ces catégories que pour une partie de la base. Dans ce cas, il est possible de construire des classifieurs « un contre tous » pour chacune de ces catégories. On parle alors de *détection de concepts sémantiques*.

Dans notre contexte de travail, aucune connaissance sur les catégories n'est disponible, et même le nombre de ces catégories est inconnu. L'idée est alors d'exploiter les annotations fournies par les utilisateurs, au même titre que pour la recherche interactive, mais dans un autre but. Contrairement à la recherche interactive, on se concentre en apprentissage long terme non pas sur une seule catégorie, mais sur toutes les catégories de la base. De plus, on s'intéresse aux annotations fournies par plusieurs utilisateurs, et non un seul.

Comme le remarquent (Vasconcelos & Kunt, 2001), l'apprentissage long terme est un problème *faiblement supervisé*. En effet, l'objectif est de retrouver les catégories sachant que :

- on ne dispose que de lots d'annotations contenant quelques annotations positives et négatives ;
- on ne connaît pas la catégorie associée aux annotations positives ;
- le nombre d'annotations est très faible ;
- les images d'annotations négatives n'appartiennent pas à une même catégorie ;
- les catégories de la base sont mélangées.

Compte tenu de ces contraintes, une approche est d'utiliser un grand nombre de lots d'annotations, et de modifier la représentation de la base (signatures et fonction de similarité) de sorte que l'information sémantique dans les lots d'annotations $\mathbf{y}$ soient représentée. Par exemple, si les utilisateurs ont souvent mis ensemble certaines images, l'optimisation doit faire en sorte que ces images soient similaires pour la représentation optimisée de la base, et inversement pour les images séparées par les utilisateurs.

7.1.2 Notations

Nous proposons dans ce paragraphe une définition formelle du problème.

Nous supposons que les utilisateurs sont à la recherche d'un nombre q fini de catégories sur la base d'image. Cette information peut être écrite sous la forme de vecteurs $\mathbf{c}_j$, tels que $c_{ij} = 1$ si l'image i appartient à la catégorie j, et $c_{ij} = -1$ sinon. Ces vecteurs peuvent être regroupés dans une matrice $\mathbf{C}$:

$$\mathbf{C} = \left(\begin{array}{c|ccccccccc} & \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & \mathbf{c}_4 & \mathbf{c}_5 & \mathbf{c}_6 & \mathbf{c}_7 & \dots & \mathbf{c}_q \\ \hline \mathbf{x}_1 & 1 & 1 & -1 & -1 & -1 & 1 & 1 & \dots & 1 \\ \mathbf{x}_2 & 1 & 1 & 1 & 1 & -1 & -1 & 1 & \dots & 1 \\ \mathbf{x}_3 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & \dots & -1 \\ \mathbf{x}_4 & 1 & -1 & 1 & 1 & -1 & -1 & -1 & \dots & -1 \\ \mathbf{x}_5 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & \dots & 1 \\ \mathbf{x}_6 & 1 & -1 & -1 & 1 & 1 & 1 & -1 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\ \mathbf{x}_n & 1 & -1 & -1 & 1 & -1 & 1 & -1 & \dots & -1 \end{array}\right)$$

La matrice $\mathbf{C}$ reflète l'ensemble des catégories que les utilisateurs ont en tête, et par conséquent **$\mathbf{C}$ n'est pas disponible**. La matrice $\mathbf{C}$ est appelée la *vérité terrain*. Notons que cette représentation de la vérité terrain permet d'exprimer le fait que les catégories se mélangent, *i.e.* qu'une image peut appartenir à plusieurs catégories à la fois. Par exemple, dans la matrice $\mathbf{C}$ ci-dessus, l'image $\mathbf{x}_1$ appartient à au moins 5 catégories.

L'objectif du long terme est, en autres, de déterminer cette matrice $\mathbf{C}$, et ce à l'aide des informations fournies par les utilisateurs lors des sessions précédentes.

Il existe plusieurs moyens de récolter ces informations. Nous avons fait le choix suivant. Nous stockons toutes les annotations des utilisateurs lors des sessions précédentes sous la forme de vecteurs $\mathbf{y}_s$ pour chaque session passée s. Ces vecteurs sont tels que $y_{is} = 1$ si l'image i appartient à la catégorie recherchée lors de la session s, $y_{is} = -1$ si l'image i n'appartient pas à la catégorie recherchée lors de la session s, et $y_{is} = 0$ si l'utilisateur n'a pas annoté l'image i. Ces vecteurs peuvent être regroupés dans une matrice $\mathbf{Y}$:

$$\mathbf{Y} = \left(\begin{array}{c|cccccccccc} & \mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_3 & \mathbf{y}_4 & \mathbf{y}_5 & \mathbf{y}_6 & \mathbf{y}_7 & \dots & \mathbf{y}_s & \dots \\ \hline \mathbf{x}_1 & 1 & 1 & 0 & 0 & -1 & 1 & 0 & \dots & 0 & \dots \\ \mathbf{x}_2 & 1 & 1 & 1 & 1 & -1 & 0 & 1 & \dots & 0 & \dots \\ \mathbf{x}_3 & 1 & 0 & 1 & -1 & 0 & 0 & 0 & \dots & -1 & \dots \\ \mathbf{x}_4 & 0 & -1 & 1 & 0 & 0 & -1 & 0 & \dots & 0 & \dots \\ \mathbf{x}_5 & -1 & 0 & 0 & 1 & 1 & -1 & 0 & \dots & 1 & \dots \\ \mathbf{x}_6 & 0 & 0 & -1 & 0 & 1 & 0 & -1 & \dots & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \\ \mathbf{x}_n & 0 & 0 & 0 & 0 & 0 & 0 & -1 & \dots & 0 & \dots \end{array}\right)$$

La matrice $\mathbf{Y}$ contient des informations **partielles** à propos de catégories **indéterminées** issues de $\mathbf{C}$. Si l'on suppose que les utilisateurs donnent leur annotation sur des images choisies au hasard, chaque vecteur de $\mathbf{Y}$ peut être vu comme le choix

aléatoire d'un des vecteurs de $\mathbf{C}$ dont on a annulé presque toutes les valeurs. La matrice $\mathbf{Y}$ contient autant de vecteurs que de sessions passées.

L'objectif est ainsi d'optimiser la représentation de la base (signatures $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, $\mathbf{x}_i \in \mathbb{R}^p$ et fonction de similarité $s(\mathbf{x}_i, \mathbf{x}_j)$) de manière à retrouver les catégories de $\mathbf{C}$ en ne disposant que des informations de $\mathbf{Y}$.

7.2 Approches existantes

7.2.1 Optimisation de la fonction de similarité

La fonction de similarité d'un système peut être optimisée pendant une session de recherche, mais aussi en propageant l'information sur toutes les sessions. Cela revient à optimiser les paramètres de la fonction, le plus souvent des poids sur les axes de l'espace des signatures. Par exemple, (Müller et al., 2000) propose une méthode pour modifier ces poids en fonction du comportement des utilisateurs.

Une forme plus souple de modification de la fonction de similarité vise à optimiser une distance quadratique entre les images, de manière à mieux refléter la sémantique. Dans ce cas, on s'interesse à l'optimisation de la matrice $\mathbf{A}$ semi-définie positive pour la distance $d_{\mathbf{A}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)\mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)}$. Notons que ce changement de distance euclidienne est équivalent à la transformation $\mathbf{X} \rightarrow \mathbf{A}^{1/2}\mathbf{X}$. (Schultz & Joachims, 2003) proposent une méthode pour déterminer $\mathbf{A}$ en fonction d'annotations relatives. Selon notre formalisme, ce type d'annotations est comparable à l'utilisation de vecteurs $\mathbf{y}_t$ dont deux valeurs seraient positives et une négative. Le problème est formulé sous la forme d'une optimisation quadratique, dont l'expression est la même que pour celle des SVM, ce qui conduit à un algorithme rapide. (Xing et al., 2002) proposent une autre méthode pour déterminer $\mathbf{A}$, qui repose sur l'ensemble des similarités $S_{ij}^{\text{sém}}$ et dissimilarités $D_{ij}^{\text{sém}}$ sémantiques entre tous les couples d'images ij de la base. $S_{ij}^{\text{sém}}$ est égale à 1 si les deux images i et j ont été annotées comme appartenant à la même catégorie, et $D_{ij}^{\text{sém}}$ est égale à 1 si les deux images i et j ont été annotées comme différentes. Ce protocole n'envisage pas la présence de catégories mélangés. L'algorithme proposé effectue la minimisation sur $\mathbf{A}$ de la distance entre les couples ij tels que $S_{ij}^{\text{sém}} = 1$, et la maximisation de la distance entre les couples ij tels que $D_{ij}^{\text{sém}} = 1$. Il fait appel à la programmation semi-définie (Lanckriet et al., 2002), et a une complexité en $O(n^2)$.

Ces techniques travaillent sur les axes de l'espace des signatures. On peut à l'inverse travailler sur les individus eux-mêmes en modifiant directement la similarité entre chaque couple.

7.2.2 Optimisation de la matrice des similarités

D'autres approches visent à modifier directement la matrice des similarités entre toutes les images.

(Han et al., 2003) proposent de modifier les valeurs de cette matrice en fonction du nombre de fois qu'un couple d'images possède ou non la même annotation. (Fournier & Cord, 2002) proposent une autre méthode basée sur le même principe. Cette approche autorise l'apprentissage de classes plus complexes, et permet même une remise en cause totale des similarités. Cependant, elle requiert une grande quantité de mémoire ($O(n^2)$) pour le stockage de la matrice des similarités. De plus, une technique d'apprentissage en ligne dédiée à la technique d'apprentissage long terme est nécessaire. En effet, les modifications *ad hoc* proposées ne respectent aucune propriété particulière en terme de métrique. En particulier, elles ne conservent pas les propriétés des distances ou des produits scalaires, ce qui interdit l'utilisation de techniques puissantes d'apprentissage interactif comme celles que nous avons présentées en partie II.

D'autres chercheurs proposent des techniques pour modifier une matrice des similarités, mais en conservant des propriétés qui autorisent l'utilisation de classifieurs généraux. L'idée est de voir la matrice de Gram comme une matrice des similarités. Dans (Cristianini et al., 2002), une méthode est proposée pour construire une matrice de Gram à partir d'un ensemble de signatures, pas nécessairement visuelles. Elle utilise le principe du *Latent Sematic Indexing* en prenant pour signatures les lignes de $\mathbf{Y}$, puis approxime par une KPCA la matrice $\mathbf{Y}\mathbf{Y}^\top$ de manière à former une matrice de Gram optimisée pour $\mathbf{Y}$. Ces noyaux sont utilisés en recherche d'image par (Heisterkamp, 2002).

On trouve aussi des méthodes basées sur l'alignement du noyau (Cristianini et al., 2001). L'alignement du noyau est une forme de corrélation entre deux matrices de Gram $\mathbf{K}_1$ et $\mathbf{K}_2$:

$$A(\mathbf{K}_1, \mathbf{K}_2) = \frac{\langle \mathbf{K}_1, \mathbf{K}_2 \rangle_F}{\sqrt{\langle \mathbf{K}_1, \mathbf{K}_1 \rangle_F \langle \mathbf{K}_2, \mathbf{K}_2 \rangle_F}}$$

avec $\langle \mathbf{K}_1, \mathbf{K}_2 \rangle_F$ le produit scalaire Frobenius[1] de deux matrices.

L'alignement du noyau est utilisé en transduction, où l'on s'intéresse à l'alignement du noyau $\mathbf{K}$ avec les annotations $\mathbf{y}$. En effet, la matrice $\mathbf{y}\mathbf{y}^\top$ est une matrice de Gram, permettant le calcul de l'alignement entre un noyau et un ensemble d'apparentissage :

$$A_{\mathbf{K}}(\mathbf{y}) = A(\mathbf{K}, \mathbf{y}\mathbf{y}^\top) = \frac{1}{\sqrt{\langle \mathbf{K}, \mathbf{K} \rangle}} \frac{\mathbf{y}^\top \mathbf{K} \mathbf{y}}{\mathbf{y}^\top \mathbf{y}}$$

(Cristianini et al., 2006) proposent une méthode pour aligner un noyau avec l'ensemble d'apprentissage en modifiant les valeurs propres de la matrice de Gram. Ils proposent aussi une méthode dans le même esprit pour rassembler les individus de même annotation. Des applications successives de ces méthodes avec différents ensembles d'apprentissage, ceux de la matrice $\mathbf{Y}$, pourraient permettre un apprentissage long terme pour plusieurs catégories. Cependant, ces méthodes sont particulièrement coûteuses en calculs, souvent de l'ordre de $O(n^2)$.

[1] Somme des produits terme à terme de deux matrices.

Il existe différentes extensions à ces travaux sur l'optimisation de la similarité. (Amari & Wu, 1999; Heisterkamp et al., 2001) utilisent des fonctions noyaux *quasiconformelles*. Bien que les techniques présentées dans ces papiers soient proposées pour la recherche en ligne, leur protocole d'apprentissage est aussi adapté à l'apprentissage long terme. Les fonctions noyaux quasiconformelles sont des fonctions noyaux modifiées par une fonction réelle positive $c(\mathbf{x})$: $\hat{k}(\mathbf{x}_i, \mathbf{x}_j) = c(\mathbf{x}_i)c(\mathbf{x}_j)k(\mathbf{x}_i, \mathbf{x}_j)$. Elles permettent de modifier localement la résolution de l'espace des signatures *via* la fonction $c(\mathbf{x})$. Ainsi, la métrique est étendue dans les zones riches en images de même catégorie, et réduite autour des images isolées. Cela permet de prédire les zones où l'on peut davantage généraliser, et celles plus problématiques, où les catégories se mélangent.

On trouve aussi diverses méthodes basées sur la programmation semi-définie (Lanckriet et al., 2002). Ces techniques utilisent l'alignement du noyau (Cristianini et al., 2001) pour optimiser les paramètres d'une combinaison de différentes fonctions noyaux. Cela permet, entre autres, de sélectionner la fonction noyau qui convient le mieux.

Une dernière approche consiste à faire du *clustering semi-supervisé*, qui détermine des clusters au même titre que les Nuées Dynamiques (*K-Means*), mais sous contraintes. (Grira et al., 2005) proposent une méthode de ce type avec le même protocole que (Xing et al., 2002), où l'information supervisée est un ensemble de couples d'images à mettre en correspondance.

7.2.3 Synthèse et choix pour RETIN

Notre objectif pour l'apprentissage long terme est tout d'abord d'optimiser la représentation de la base d'images (signatures $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, $\mathbf{x}_i \in \mathbb{R}^p$ et fonction de similarité $s(\mathbf{x}_i, \mathbf{x}_j)$) en fonction des annotations passées de $\mathbf{Y}$. L'optimisation que nous souhaitons est celle qui améliore la représentation des catégories recherchées par les utilisateurs ($\mathbf{C}$), et qui a pour conséquence d'améliorer la qualité des recherches futures. Nous présentons à la suite quelques points clef de notre contexte d'apprentissage, qui ont orienté nos choix de méthodes par rapport aux approches existantes.

Compte tenu du fait que nous travaillons sur des bases généralistes, il est nécessaire de disposer de techniques d'apprentissage long terme capables d'apprendre les catégories les plus complexes, largement multimodales, et mélangées. Dans ce contexte, les techniques d'optimisation de la fonction de similarité ne semblent pas adaptées. De même, les techniques de déformation locale de la similarité ne nous semblent pas non plus suffisantes.

Un autre point concerne le cadre d'exploitation de l'apprentissage long terme que nous souhaitons utiliser. En effet, notre objectif est de proposer des solutions qui s'intègrent dans un système de recherche global. Il n'est pas envisageable de restreindre les techniques de recherche interactive suite à l'intégration d'un apprentissage long terme. Sous ces contraintes, il est nécessaire que les solutions proposées permettent l'utilisation de toutes les techniques de recherche interactive présentées en partie II. Ces techniques reposent essentiellement sur une représentation de la base par une fonction noyau. La contrainte

que nous nous imposons pour l'utilisation conjointe de la recherche interactive et du long terme est donc de se restreindre aux approches qui optimisent une fonction noyau.

Nous nous intéressons aux systèmes de recherche qui s'appliquent dans un cadre temps réel. Le système doit être disponible en permanence et se mettre à jour au fur à et à mesure des sessions de recherche. Le meilleur exemple du cadre opérationnel recherché est celui d'un système diffusé sur un réseau, constamment alimenté en images et en annotations par les utilisateurs. Cette contrainte applicative a plusieurs conséquences. La première concerne la quantité de temps de calculs disponibles : il est nécessaire que les méthodes soient rapides, au sens où le système doit pouvoir maintenir une activité constante. En d'autres termes, il est nécessaire que les nouveaux lots d'annotations soient traités aussi vite qu'ils arrivent. Afin de formaliser ceci, nous utilisons une contrainte courante qui consiste à fixer un ordre de grandeur linéaire avec la taille de la base ($O(n)$). Une deuxième contrainte qui va de pair avec la première concerne la capacité du système à gérer les grandes bases d'images. Les bases peuvent atteindre des tailles très élevées, et il nous semble difficile d'envisager un besoin en mémoire au delà de $O(n)$. Ces deux contraintes exluent un certain nombre d'approches. Les techniques d'optimisation coûteuse en calculs ne sont pas envisageable, comme certaines techniques basées sur la programmation semi-définie que l'on peut trouver dans la communauté de l'apprentissage statistique. De même, les techniques qui nécessitent le stockage complet de la matrice des similarités ne peuvent s'appliquer directement.

Nous avons donc choisi d'orienter notre recherche vers une approche par optimisation de la similarité entre toutes les images par des méthodes à noyau à faible coût en calcul et en mémoire.

7.3 Stratégies RETIN

Pour réaliser cette optimisation, il existe plusieurs manières de procéder, l'une étant de modifier directement les similarités, et une autre étant de déplacer des vecteurs[2] de sorte que leurs produits scalaires reflètent les similarités désirées. Nous proposons une méthode pour chacune de ces deux approches :

- *Méthode basée matrice de Gram* (RETIN SL) (Gosselin & Cord, 2004c; Gosselin & Cord, 2005c). Cette première méthode modifie directement la matrice des similarités entre toutes les images. Cependant, nous nous imposons de travailler non pas sur une matrice des similarités quelconques, mais sur une matrice de Gram. Ceci nous permet d'utiliser toutes les techniques d'apprentissage interactif basées noyaux. De plus, nous verrons qu'une approximation par KPCA et un algorithme rapide assurent une complexité en $O(n)$.

- *Méthode basée vecteurs* (RETIN CVL) (Gosselin & Cord, 2005b; Gosselin & Cord, 2006). Cette deuxième méthode déplace les vecteurs de $\mathbf{X}$ de manière à les regrouper

[2]Ces vecteurs sont soit directement les signatures (si elles sont vectorielles), soit le vecteur résultant d'une injection Φ (*cf.* chapitre 3).

par catégories. Nous proposons un critère original qui permet une grande stabilité et une gestion des catégories mélangées. Compte tenu du fait que la méthode modifie des vecteurs, l'utilisation conjointe avec les techniques de recherche basées vecteurs (et donc basées noyaux) est possible. Nous proposons dans le chapitre suivant un algorithme très rapide qui offre une complexité négligeable devant la taille de la base ($O(1)$).

L'apprentissage long terme nécessite avant tout la collecte des informations sur les sessions passées, ainsi que leur exploitation pour l'optimisation du système. Dans le cadre de cette thèse, nous avons fait le choix d'un scénario adaptatif : à chaque fin de session de recherche, les informations sont immédiatement utilisées pour optimiser la représentation. Ce scénario adaptatif est difficile à paramétrer, étant donné que nous sommes face à une double boucle d'adaptation si nous prenons en compte l'apprentissage interactif. En effet, la nouvelle représentation dépend des sessions passées, et chaque lot d'annotations pour chaque session dépend lui même des choix de la sélection active, qui elle même dépend de la représentation précédente, *etc.*

Un scénario adaptif peut s'écrire sous la forme d'une mise à jour $\hat{X}$ des signatures X de la base en fonction des annotations $\mathbf{y}$:

$$\hat{X} = \text{update}(X, \mathbf{y}, \rho)$$

où ρ est un réel entre 0 et 1 qui quantifie l'impact de la mise à jour. Nous appelons ρ la *vigilance*. Ce paramètre peut être réglé en fonction de l'utilisateur. Si l'utilisateur est un expert (par exemple un médecin sur une base médicale), on peut largement prendre en compte ses annotations, en utilisant une valeur forte de ρ. Dans le cas contraire, on peut limiter la mise à jour en utilisant une valeur faible de ρ.

7.4 Conclusion

Nous avons exposé le problème de l'apprentissage long terme dans notre contexte généraliste. Il s'agit d'un problème faiblement supervisé, où les données d'apprentissage sont un ensemble de vecteurs $\mathbf{y}_t$ qui ne contiennent que très peu d'annotations, ni ne permettent d'identifier la catégorie associée. Après avoir présenté quelques méthodes existantes, nous avons exposé notre approche du problème, ainsi que le principe de nos deux méthodes. Notre approche est basée sur une optimisation de la représentation de la base, soit en travaillant directement sur les similarités entre les images, soit en modifiant les vecteurs signatures.

Bien que dans cette thèse nous n'appliquions ces deux méthodes que dans le but d'améliorer un système de recherche interactive, il est aussi possible de les utiliser à d'autre fins. Par exemple, en détection de concepts sémantiques, le but est de construire un détecteur pour chaque catégorie de la base. Chacun de ces détecteurs permet de déterminer à quel point une image (pas nécessairement dans une base) appartient à la catégorie associée au détecteur. Ces outils reposent en général sur des ensembles d'apprentissages, un

par catégorie, qui sont utilisés pour entraîner les déctecteurs, en général des classifieurs « un contre tous ». Ces techniques nécessitent par conséquent de disposer d'un ensemble d'apprentissage pour chaque catégorie. Or, dans un contexte faiblement supervisé, ces ensembles ne sont pas disponibles, et les informations dont on dispose sont très partielles (*cf.* 7.1.1). C'est à ce niveau que les méthodes d'apprentissage long terme que nous proposons peuvent intervenir. En effet, une fois que l'on a appliqué l'une de ces méthodes, les images sont alors regroupées en clusters, un cluster par catégorie. On peut alors détecter ces clusters (par exemple avec un K-Means) pour en déduire des ensembles d'apprentissage. Ainsi, les solutions que nous proposons peuvent être une étape préliminaire aux techniques de détection de concepts sémantiques dans le cas où l'information est faiblement supervisée.

Chapitre 8

Méthode basée matrice de Gram

Nous présentons dans ce chapitre une approche basée sur une optimisation d'une matrice de similarité particulière, la matrice de Gram $\mathbf{K}$ de dimension $n \times n$. Cette matrice est l'ensemble des produits scalaires des injections $\Phi(\mathbf{x}_i)$ des signatures $\mathbf{x}_i \in X$ dans un espace induit (*cf.* chapitre 3) :

$$\mathbf{K} = \Phi(X)^\top \Phi(X)$$

Dans un souci de lisibilité, nous ne considérerons que la matrice $\mathbf{X} = \Phi(X)$ des signatures dans l'espace induit. La matrice de Gram peut s'écrire comme :

$$\mathbf{K} = \mathbf{X}^\top \mathbf{X} \tag{8.1}$$

La méthode que nous proposons est une mise à jour de cette matrice qui respecte la propriété *semi-définie positive* de la matrice de Gram. De plus, nous présentons une approximation par KPCA et un algorithme en $O(n)$ qui permet l'utilisation de cette méthode pour les grandes bases d'images.

Cette méthode, notée *RETIN Semantic Learner*, a fait l'objet de publications (Gosselin & Cord, 2004c; Gosselin & Cord, 2005c).

8.1 Motivations

L'idée initiale de cette méthode est similaire à celles des approches par modification des poids sémantiques tel que (Han et al., 2003) ou (Fournier & Cord, 2002). Ces techniques modifient les valeurs de la matrice des similarités, mais ne conservent pas de propriétés métriques, et ont un besoin en mémoire en $O(n^2)$.

Nous proposons d'imposer des contraintes sur le cadre de déformation de la matrice des similarités. Comme nous l'avons déjà remarqué, la matrice de Gram peut être vue comme une matrice des similarités, mais avec la propriété d'être semi-definie positive (*sdp*) (cf. 3.2.1). Nous nous efforçons d'effectuer toute modification de la matrice en respectant cette propriété.

Deuxièmement, les méthodes citées ont un besoin en mémoire en $O(n^2)$, ce qui limite leur utilisation à de petites bases. Nous proposons d'exploiter la propriété *sdp* afin de réduire le besoin en $O(n)$. En effet, avec cette propriété, la matrice peut toujours être décomposée en vecteurs et valeurs propres :

$$\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$$

avec $\mathbf{V}$ la matrice des vecteurs propres et $\mathbf{\Lambda}$ la matrice diagonale des valeurs propres.

Ainsi, sous réserve d'un faible rang de cette matrice, la matrice peut être décrite en ne conservant que les espaces propres à forte variance.

8.2 Construction du noyau

Nous présentons dans ce paragraphe une approche constructive du noyau. L'idée est d'effectuer la somme des noyaux qui résultent de chaque session de recherche (Cristianini et al., 2002). La question est de savoir si l'on peut construire un noyau $\mathbf{K}$ à partir d'un ensemble $\mathbf{Y}$ d'annotations tel qu'en entraînant ensuite un classifieur sur $\mathbf{K}$ avec n'importe quel $\mathbf{y}_t \in \mathbf{Y}$, les images annotées de $\mathbf{y}_t$ soient correctement classées. Nous nous limitons dans cette étude aux classifieurs par hyperplan.

Une fonction de décision par hyperplan est le signe de la projection sur la normale $\mathbf{w} \in \mathbb{R}^n$ de l'hyperplan[1] :

$$f(\mathbf{x}) = \langle \mathbf{x}, \mathbf{w} \rangle = \mathbf{x}^\top \mathbf{w}$$

Nous travaillons sur une matrice $\mathbf{X}$, ce qui nous permet d'écrire l'image par f de cet ensemble de manière matricielle : $f(\mathbf{X}) = \mathbf{X}^\top \mathbf{w}$, et nous nous restreignons aux hyperplans dont la normale est une combinaison linéaire de $\mathbf{X}$, *i.e.* il existe $\boldsymbol{\alpha} \in \mathbb{R}^n$ tel que $\mathbf{w} = \mathbf{X}\boldsymbol{\alpha}$: $f(\mathbf{X}) = \mathbf{X}^\top \mathbf{X}\boldsymbol{\alpha}$. D'où, en utilisant l'équation 8.1 :

$$f(\mathbf{X}) = \mathbf{K}\boldsymbol{\alpha}$$

[1]Dans tous ce chapitre, nous travaillons dans l'espace induit par une fonction $\Phi()$. Par exemple, la normale $\mathbf{w}$ est égale à $\Phi(w)$ avec w une signature. De plus, étant donné que nous travaillons sur un ensemble fini de n individus, l'espace induit est donc $\mathbb{R}^n$.

Considérons à présent un lot d'annotations $\mathbf{y} \in \mathbf{Y}$. Assurer qu'il existe un hyperplan séparateur selon les annotations $\mathbf{y}$ revient à dire qu'il existe un $\boldsymbol{\alpha}$ tel que $\forall i \in [1,n] \quad y_i f(\mathbf{x}_i) > 0$, soit en écriture matricielle $\text{diag}(f(\mathbf{X})\mathbf{y}^\top) > 0$ ou encore $\text{diag}(\mathbf{K}\boldsymbol{\alpha}\mathbf{y}^\top) > 0$.

La question que nous nous posons s'exprime alors de la manière suivante : un noyau $\mathbf{K}$ a « appris » les lots d'annotations dans $\mathbf{Y}$ s'il assure qu'il existe un hyperplan séparateur pour tout lot d'annotations dans $\mathbf{Y}$:

$$\forall \mathbf{y} \in \mathbf{Y} \; \exists \boldsymbol{\alpha} \in \mathbb{R}^n \; \text{diag}(\mathbf{K}\boldsymbol{\alpha}\mathbf{y}^\top) > 0$$

Intéressons-nous tout d'abord à la construction d'un noyau à partir d'un seul lot d'annotations. Considérons une classe 1 sur un ensemble de vecteurs $\mathbf{X}$ décrite par un vecteur $\mathbf{y}_1$, tel que $y_{i1} = 1$ si $\mathbf{x}_i$ est dans la classe, et $y_{i1} = -1$ sinon.

Etudions la séparabilité de la classe 1 avec le noyau suivant :

$$\mathbf{K}_1 = \mathbf{y}_1\mathbf{y}_1^\top$$

Le problème revient à trouver $\boldsymbol{\alpha} \in \mathbb{R}^n$ tel que :

$$\begin{aligned}
& \text{diag}(\mathbf{K}_1\boldsymbol{\alpha}\mathbf{y}_1^\top) > 0 \\
\Leftrightarrow \quad & \text{diag}(\mathbf{y}_1\mathbf{y}_1^\top\boldsymbol{\alpha}\mathbf{y}_1^\top) > 0 \\
\Leftrightarrow \quad & \text{diag}(\mathbf{y}_1 \underbrace{\mathbf{y}_1^\top\boldsymbol{\alpha}}_{\beta\in\mathbb{R}}\mathbf{y}_1^\top) > 0 \\
\Leftrightarrow \quad & \beta\,\text{diag}(\mathbf{y}_1\mathbf{y}_1^\top) > 0
\end{aligned}$$

$\text{diag}(\mathbf{y}_1\mathbf{y}_1^\top) > 0$ est toujours vrai, il suffit donc de prendre un $\boldsymbol{\alpha}$ tel que $\beta = \mathbf{y}_1^\top\boldsymbol{\alpha}$ soit positif. Par exemple, $\boldsymbol{\alpha} = \mathbf{y}_1$ est solution.

Regardons à présent si le noyau $\mathbf{K}_1$ est capable de séparer d'autres classes que la classe 1. Considérons une classe 2 différente de la classe 1 et décrite par un vecteur $\mathbf{y}_2$. Nous considérons que deux classes sont différentes si leurs vecteurs de descriptions ne sont pas colinéaires.

Etudions la séparabilité de la classe 2 avec le noyau $\mathbf{K}_1$:

$$\begin{aligned}
& \text{diag}(\mathbf{K}_1\boldsymbol{\alpha}\mathbf{y}_2^\top) > 0 \\
\Leftrightarrow \quad & \text{diag}(\mathbf{y}_1\mathbf{y}_1^\top\boldsymbol{\alpha}\mathbf{y}_2^\top) > 0 \\
\Leftrightarrow \quad & \text{diag}(\mathbf{y}_1 \underbrace{\mathbf{y}_1^\top\boldsymbol{\alpha}}_{\beta\in\mathbb{R}}\mathbf{y}_2^\top) > 0 \\
\Leftrightarrow \quad & \beta\,\text{diag}(\mathbf{y}_1\mathbf{y}_2^\top) > 0
\end{aligned} \tag{8.2}$$

Etant donné que les classes 1 et 2 sont différentes, il existe un i tel que $y_{i1}y_{i2} > 0$ et un j tel que $y_{j1}y_{j2} < 0$. Donc l'équation 8.2 ne peut être vérifiée, et par conséquent $\mathbf{K}_1$ n'assure pas la séparabilité escomptée.

Intéressons-nous maintenant à un autre noyau :

$$\mathbf{K}_2 = \mathbf{y}_1\mathbf{y}_1^\top + \mathbf{y}_2\mathbf{y}_2^\top$$

Etudions la séparabilité de la classe 1 avec le noyau $\mathbf{K}_2$:

$$\begin{array}{ll} & \mathrm{diag}(\mathbf{K}_2\boldsymbol{\alpha}\mathbf{y}_1^\top) > 0 \\ \Leftrightarrow & \mathrm{diag}((\mathbf{y}_1\mathbf{y}_1^\top + \mathbf{y}_2\mathbf{y}_2^\top)\boldsymbol{\alpha}\mathbf{y}_1^\top) > 0 \\ \Leftrightarrow & \mathrm{diag}(\mathbf{y}_1 \underbrace{\mathbf{y}_1^\top\boldsymbol{\alpha}}_{\beta_1\in\mathbb{R}}\mathbf{y}_1^\top + \mathbf{y}_2 \underbrace{\mathbf{y}_2^\top\boldsymbol{\alpha}}_{\beta_2\in\mathbb{R}}\mathbf{y}_1^\top) > 0 \\ \Leftrightarrow & \beta_1\mathrm{diag}(\mathbf{y}_1\mathbf{y}_1^\top) + \beta_2\mathrm{diag}(\mathbf{y}_2\mathbf{y}_1^\top) > 0 \end{array} \tag{8.3}$$

Puisque les classes 1 et 2 sont différentes, *i.e.* $\mathbf{y}_1$ et $\mathbf{y}_2$ ne sont pas colinéaires, alors il existe un $\boldsymbol{\alpha}$ colinéaire à $\mathbf{y}_1$ ($\beta_1 > 0$) et perpendiculaire à $\mathbf{y}_2$ ($\beta_2 = 0$). Avec le noyau $\mathbf{K}_2$, il est donc toujours possible de séparer les individus de la classe 1 des autres. Un raisonnement similaire permet d'affirmer que ce même noyau sépare aussi la classe 2.

Nous pouvons poursuive cette démarche de construction pour un ensemble de $\{\mathbf{y}_t\}_{i\in[1,q]}$ différents (*i.e.* non colinéaires deux à deux) :

$$\mathbf{K}_q = \sum_{t=1}^{q} \mathbf{y}_t\mathbf{y}_t^\top \tag{8.4}$$

Etudions la séparabilité de la classe i *versus* les autres à l'aide de ce noyau :

$$\begin{array}{ll} & \mathrm{diag}(\mathbf{K}_q\boldsymbol{\alpha}\mathbf{y}_i^\top) > 0 \\ \Leftrightarrow & \mathrm{diag}(\sum_{t=1}^{q} \mathbf{y}_t \underbrace{\mathbf{y}_t^\top\boldsymbol{\alpha}}_{\beta_t\mathbb{R}}\mathbf{y}_i^\top) > 0 \\ \Leftrightarrow & \sum_{t=1}^{q} \beta_t\mathrm{diag}(\mathbf{y}_t\mathbf{y}_i^\top) > 0 \\ \Leftrightarrow & \beta_i\mathrm{diag}(\mathbf{y}_i\mathbf{y}_i^\top) + \sum_{t\neq i} \beta_t\mathrm{diag}(\mathbf{y}_t\mathbf{y}_i^\top) > 0 \end{array} \tag{8.5}$$

La condition de différence (les lots d'annotation sont non colinéraires deux à deux) n'est pas suffisante pour nous permettre d'assurer une solution. Cependant, nous pouvons remarquer que si les $\mathbf{y}_t$ sont linéairement indépendants, alors l'equation 8.5 est vérifiée si nous prenons un $\boldsymbol{\alpha}$ colinéaire à $\mathbf{y}_i$ ($\beta_i > 0$) et perpendiculaire à tous les autres $\mathbf{y}_t$ ($\beta_t = 0,\ t \neq i$).

Ainsi, une condition suffisante pour que le noyau 8.4 sépare q classes est d'assurer que les vecteurs d'annotations sont linéairement indépendants. Cela signifie qu'un tel noyau

apprend autant de classes qu'il y a de dimensions dans l'espace engendré par l'ensemble des $\{\mathbf{y}_t\}_{t=[1,q]}$. Or, les lots d'annotations sont généralement peu corrélés, par exemple la génération aléatoire d'une matrice Y de q lots partiels d'annotations a un rang proche de q. Ce type de construction va donc mémoriser tous ces lots un à un, et ne va pas recouper les différentes sessions précédentes afin de faire émerger les catégories. Cette approche constructive mémorise trop et ne généralise pas suffisamment.

Cette condition suffisante d'indépendance des $\mathbf{y}_t$ est intéressante si nous faisons le rapprochement avec les vecteurs propres. En effet, la décomposition en valeurs et vecteurs propres de $\mathbf{K}$ est une somme similaire à celle de l'équation 8.4 :

$$\mathbf{K} = \sum_{t=1}^{q} \lambda_t \mathbf{v}_t \mathbf{v}_t^\top \tag{8.6}$$

Les vecteurs propres $\mathbf{v}_t$ peuvent être vus comme des vecteurs d'annotations, et étant donné que les valeurs propres λ_t sont positives, nous avons bien une somme équivalente à celle de l'équation 8.4. Ce résultat, qui n'est pas étranger à la dimension VC des hyperplans, assure que nous pouvons séparer par des hyperplans au moins q catégories mélangées dans une matrice de Gram de rang q.

Notons que les relations existant entre le spectre de la matrice de Gram et la capacité d'apprentissage qui en découle est un sujet de recherche encore peu exploré, et est actuellement une problématique importante au sein de la communauté de l'apprentissage statistique (Scholkpof et al., 1999; Shawe-Taylor et al., 2002).

Nous proposons au paragraphe suivant une méthode similaire basée sur cette approche constructive, mais avec des modifications qui permettent une meilleure généralisation.

8.3 RETIN SL

La méthode que nous proposons, *RETIN Semantic Learner*, s'intègre dans le scénario adaptatif que nous avons présenté au chapitre précédent. Le but est donc de mettre en place une mise à jour de la représentation de la base à la fin de chaque session de recherche :

$$\hat{X} = \text{update}_{\text{gram}}(X, \mathbf{y}, \rho) \tag{8.7}$$

Compte tenu du fait que nous travaillons avec une approche noyau, nous nous intéressons à l'image des signatures de X par une injection Φ, et étant donné que nous travaillons sur un ensemble fini d'individus (la base d'image), nous pouvons considérer la matrice $\mathbf{X} = \Phi(X)$ de l'image des signatures X par Φ. La mise à jour de l'équation 8.7 est donc équivalente à :

$$\hat{\mathbf{X}} = \text{update}_{\text{gram}}(\mathbf{X}, \mathbf{y}, \rho) \tag{8.8}$$

Enfin, compte tenu du fait que nous souhaitons travailler sur les similarités, donc sur les valeurs de la matrice de Gram, le formalisme que nous allons utiliser pour mettre en place la méthode sera une mise à jour de la matrice de Gram elle-même :

$$\hat{\mathbf{K}} = \text{update}_{\text{gram}}(\mathbf{K}, \mathbf{y}, \rho) \tag{8.9}$$

Cette dernière équation 8.9 est équivalente aux deux précédentes (8.7,8.8) si nous posons $\mathbf{K} = \mathbf{X}^\top \mathbf{X} = \Phi(X)^\top \Phi(X)$.

Notre méthode se base sur l'approche constructive présentée au paragraphe précédent, mais en ajoutant des modifications qui permettent une meilleure généralisation. De plus, afin de répondre aux contraintes de temps de calcul, nous proposons un algorithme de mise à jour en $O(n)$.

Une des difficultés dans la mise en place d'une méthode basée matrice de Gram est que nous devons conserver la propriété de *dsp* (*semi-définie positive* : les valeurs propres de la matrice existent et sont positives). Ainsi, partant d'une matrice $\mathbf{K}$ *sdp*, nous devons nous assurer que le résultat de la mise à jour $\hat{\mathbf{K}}$ est aussi *sdp*.

Dans un souci de lisibilité, nous supposons que les indices des images ont été permutés de telle sorte que les indices I_1 des annotations positives correspondent aux premières images de la base, et que les indices I_{-1} des annotations négatives correspondent aux images suivantes :

$$\begin{array}{lcll} I_1 & = & \text{Indices des images annotées positivement} & = \{1, \ldots, n_1\} \\ I_{-1} & = & \text{Indices des images annotées négativement} & = \{n_1+1, \ldots, n_1+n_{-1}\} \\ I & = & \text{Indices des images annotées} & = \{1, \ldots, n_1+n_{-1}\} \\ \bar{I} & = & \text{Indices des images non annotées} & = \{n_1+n_{-1}+1, \ldots, n\} \end{array} \tag{8.10}$$

avec n_1 le nombre d'images d'annotations positives, et n_{-1} le nombre d'images d'annotations négatives. Ce changement d'écriture n'entraîne aucune perte de généralité.

8.3.1 Gérer le multi-classe

Nous démarrons la construction de notre méthode en nous appuyant sur la méthode constructive présentée au paragraphe précédent. Cette méthode permet de construire un noyau à partir d'un ensemble de q vecteurs d'annotations :

$$\mathbf{K}_q = \sum_{t=1}^{q} \mathbf{y}_t \mathbf{y}_t^\top \tag{8.11}$$

Nous nous intéressons à une approche adaptative, et une mise à jour correspondant à cette méthode est la suivante :

$$\text{update}_{\text{build}}(\mathbf{K}, \mathbf{y}, \rho) = (1-\rho)\mathbf{K} + \rho\mathbf{y}\mathbf{y}^\top$$

Cependant une telle mise à jour ne prend pas en compte la nature des annotations dans $\mathbf{y}$: elle suppose que les images annotées négativement sont dans la même catégorie[2]. Si nous observons les modifications faites par cette mise à jour :

$$\left((1-\rho)\mathbf{K} + \rho\mathbf{y}\mathbf{y}^\top\right)_{ij} = (1-\rho)K_{ij} + \rho \begin{cases} 1 & \text{si } i,j \in I_1 \times I_1 \\ -1 & \text{si } i,j \in I_1 \times I_{-1} \\ -1 & \text{si } i,j \in I_{-1} \times I_1 \\ 1 & \text{si } i,j \in I_{-1} \times I_{-1} \\ 0 & \text{sinon} \end{cases}$$

Nous pouvons voir que la similarité K_{ij} entre les couples i, j d'images annotées négativement est augmentée (la vigilance ρ est toujours comprise entre 0 et 1).

Or, autant les images d'annotations positives sont dans une même catégorie, autant les images d'annotations négatives ne sont pas forcément dans la même catégorie. Nous savons juste qu'elles ne sont pas dans la même catégorie que les images positives. Nous souhaitons donc renforcer les similarités entre les images positives, diminuer les similarités entre les images positives et négatives, et ne pas trop modifier les similarités entre les images négatives, ce qui est réalisé par l'opération suivante :

$$(1-\rho)\mathbf{K} + \rho\mathbf{u}\mathbf{u}^\top$$

avec

$$u_i = \begin{cases} 1 & \text{si } y_i > 0 \\ -\gamma & \text{si } y_i < 0 \\ 0 & \text{sinon} \end{cases} \qquad et \quad \gamma > 0$$

.

Cette modification, qui conserve la propriété de *sdp*[3], a pour effet (pondéré par ρ) :

- d'augmenter de 1 les similarités entre les images positives ;
- de diminuer de γ les similarités entre les images positives et négatives ;
- d'augmenter de γ^2 les similarités entre les images négatives.

Ainsi, avec une valeur de $\gamma < 1$ les similarités entre les images négatives sont peu augmentées (γ^2 devient petit devant 1), tout en diminuant suffisamment les similarités entre les images positives et négatives.

[2]Les noyaux du type $\mathbf{y}\mathbf{y}^\top$ ont été introduits pour des problèmes à deux classes.

[3]La somme de deux matrices *sdp* est *sdp*.

8.3.2 Généraliser

Nous proposons de généraliser davantage en ajoutant une opération qui va propager les modifications sur les similarités/produits scalaires des images non annotées selon $\mathbf{y}$.

L'idée est de faire émerger les catégories après plusieurs mises à jour successives. Si nous nous contentons d'une mise à jour du type de l'approche constructive, les modifications dans la matrice restent locales. En effet, seules les similarités/produits scalaires entre les images annotées sont affectées.

Propager l'information contenue dans $\mathbf{y}$ aux images non annotées est la solution que nous avons choisie pour mieux généraliser. L'idée est d'homogénéiser les similarités entre les images annotées positivement et les autres images. Ainsi, si dans la suite des mises à jour, des lots d'annotations partielles pour une même catégorie sont fournis, *i.e.* certaines images de cette catégorie sont annotées positivement dans chaque lot, alors la similarité de toutes les images de cette catégorie finiront par s'homogénéiser.

Pour ce faire, nous utilisons une opération qui conserve la propriété de *sdp*, tout en obtenant l'effet désiré. Dans cette optique, nous avons choisi l'opérateur suivant :

$$\mathbf{K} \leftarrow \mathbf{T}\mathbf{K}\mathbf{T}^\top$$

avec :

$$\mathbf{T} = \begin{pmatrix} \overbrace{\begin{matrix} \frac{1}{n_1} & \cdots & \frac{1}{n_1} \\ \vdots & & \vdots \\ \frac{1}{n_1} & \cdots & \frac{1}{n_1} \end{matrix}}^{n_1 \text{ colonnes}} & & \\ & 1 & \\ & & \ddots \\ & & 1 \end{pmatrix}$$

Cet opérateur conserve la propriété de *sdp*, puisque pour toute matrice $p \times n$ $\mathbf{A}$ et toute matrice $n \times n$ $\mathbf{B}$ *sdp*, alors $\mathbf{A}\mathbf{B}\mathbf{A}^\top$ est *sdp*.

Cette opération moyenne les similarités entre les images d'annotation positive et toutes les autres. Cela revient à remplacer dans $\mathbf{K}$ chaque ligne/colonne d'indice dans I_1 par la moyenne des lignes/colonnes d'indice dans I_1.

Cette opération a un effet plus lisible si l'on observe son équivalent en vectoriel. En effet, puisque $\mathbf{K} = \mathbf{X}^\top\mathbf{X}$, nous avons :

$$\begin{aligned} \mathbf{T}\mathbf{K}\mathbf{T}^\top &= \mathbf{T}\mathbf{X}^\top\mathbf{X}\mathbf{T}^\top \\ &= (\mathbf{T}^\top\mathbf{X})^\top(\mathbf{T}^\top\mathbf{X}) \end{aligned}$$

Cela revient, d'un point de vu vectoriel, à effectuer l'opération suivante :

$$\mathbf{X} \leftarrow \mathbf{T}^\top \mathbf{X}$$

Soit, plus précisément :

$$\begin{array}{ll} \forall i \in I_1 & \mathbf{x}_i \leftarrow \frac{1}{n_1} \sum_{j \in I_1} \mathbf{x}_j \\ \forall i \notin I_1 & \mathbf{x}_i \leftarrow \mathbf{x}_i \end{array}$$

L'opération a aussi pour effet de diminuer le rang de $\mathbf{K}$ puisque, suite à cette opération, toutes les lignes/colonnes[4] d'indice dans I_1 sont linéairement dépendantes (elles sont identiques).

Notons qu'une opération similaire n'est pas appliquée aux images annotées négativement, puisque ces images n'ont pas de raison d'être dans une même catégorie. Nous savons juste qu'elles ne sont pas dans la catégorie des images annotées positivement.

8.3.3 Schéma de mise à jour

Nous fusionnons les deux modifications en l'opérateur suivant :

$$\mathbf{K}^\star = (1-\rho)\mathbf{K} + \rho a(\mathbf{T}\mathbf{K}\mathbf{T}^\top + b\mathbf{u}\mathbf{u}^\top)$$

La mise à jour finale est une approximation de ce calcul :

$$\text{update}_{\text{gram}}(\mathbf{K}, \mathbf{y}, \rho) = \hat{\mathbf{K}} = \text{approximation de } \mathbf{K}^\star$$

Les deux réels a, b sont calculés de manière à stabiliser la mise à jour :
- a est calculé de sorte que $\sum_{ij} \hat{K}_{ij} = \sum_{ij} K_{ij}$. Cette opération permet d'assurer que les valeurs ne dépasseront pas la limite des réels flottants autorisée par les ordinateurs.
- b est calculé de sorte que les diagonales d'indice dans I_1 de $\mathbf{T}\mathbf{K}\mathbf{T}^\top + b\mathbf{u}\mathbf{u}^\top$ soient égales à 1. L'opération $\mathbf{T}\mathbf{K}\mathbf{T}^\top$ diminue la valeur de ces diagonales, alors que $\mathbf{u}\mathbf{u}^\top$ les augmente. Cela évite un sur-apprentissage des catégories souvent recherchées par les utilisateurs. Par exemple, si l'on effectue un grand nombre de fois la mise à jour avec le même lot d'annotations sans cette correction, alors les diagonales d'indices dans $\bar{I}$ seront nulles, *i.e.* toute l'information à propos des images non annotées serait totalement effacée.

[4]Nous travaillons sur une matrice symétrique, donc la ligne d'indice i est égale à la colonne d'indice i.

$$\mathbf{TKT}^\top + b\mathbf{uu}^\top =$$

$$\left(\begin{array}{ccc|ccc|c} \mu & \dots & \mu & v_{n_1+1} & \dots & \dots & v_n \\ \vdots & & \vdots & \vdots & & & \\ \mu & \dots & \mu & v_{n_1+1} & \dots & \dots & v_n \\ \hline v_{n_1+1} & \dots & v_{n_1+1} & & & & \\ \vdots & & \vdots & & & & \\ & & & & & & \\ \hline & & & & & & \\ \vdots & & \vdots & & & & \\ v_n & \dots & v_n & & & & \end{array}\right) + b \left(\begin{array}{ccc|ccc|ccc} 1 & \dots & 1 & -\gamma & \dots & -\gamma & 0 & \dots & 0 \\ \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 1 & \dots & 1 & -\gamma & \dots & -\gamma & 0 & \dots & 0 \\ \hline -\gamma & \dots & -\gamma & \gamma^2 & \dots & \gamma^2 & & & \\ \vdots & & \vdots & \vdots & & \vdots & & & \\ -\gamma & \dots & -\gamma & \gamma^2 & \dots & \gamma^2 & & & \\ \hline 0 & & 0 & & & & & & \\ \vdots & & \vdots & & & & & & \\ 0 & \dots & 0 & & & & & & \end{array}\right)$$

avec $\mu = \frac{1}{n_1^2} \sum_{i,j \in I_1} K_{ij}$ et $v_i = \frac{1}{n_1} \sum_{j \in I_1} K_{ij}$

8.4 Algorithme

Le calcul direct de la mise à jour s'effectue en $O(n^2)$. Nous proposons dans ce paragraphe un algorithme qui permet une mise à jour en $O(n)$ en nous appuyant sur une factorisation, ainsi qu'une approximation de la matrice de Gram par décomposition spectrale.

8.4.1 Approximation de la matrice de Gram

Le but de la méthode est de calculer une appromixation $\hat{\mathbf{K}}$ de la matrice $\mathbf{K}^\star$:

$$\mathbf{K}^\star = (1-\rho)\mathbf{K} + \rho a(\mathbf{TKT}^\top + b\mathbf{uu}^\top)$$

L'approximation se fait par décomposition en vecteurs et valeurs propres, ce qui revient à déterminer la matrice $n \times n$ orthogonale $\mathbf{V}^\star$ et la matrice $n \times n$ diagonale $\mathbf{\Lambda}^\star$ telles que :

$$\mathbf{K}^\star = \mathbf{V}^\star \mathbf{\Lambda}^\star (\mathbf{V}^\star)^\top$$

Nous utilisons une technique classique du calcul matricielle qui consiste à écrire $\mathbf{K}^\star$ sous la forme :

$$\mathbf{K}^\star = (\text{une matrice orthogonale})(\text{une petite matrice})(\text{une matrice orthogonale})^\top$$

La petite matrice peut être décomposée rapidement en vecteurs et valeurs propres, en utilisant une des techniques standard de décomposition spectrale.

Pour ce faire, nous commencons par factoriser $\mathbf{K}^\star$ de la manière suivante :

$$\mathbf{K}^\star = \mathbf{A}\mathbf{B}\mathbf{A}^\top$$

où $\mathbf{A}$ et $\mathbf{B}$ sont deux matrices, avec $\mathbf{B}$ de petites dimensions. Les détails de cette décomposition sont présentés plus loin.

En calculant la décomposition QR de $\mathbf{A} = \mathbf{Q}\mathbf{R}$, *i.e.* telle que $\mathbf{Q}$ soit orthogonale, nous avons :

$$\begin{aligned} \mathbf{K}^\star &= \mathbf{A}\mathbf{B}\mathbf{A}^\top \\ &= \mathbf{Q}\mathbf{R}\mathbf{B}(\mathbf{Q}\mathbf{R})^\top \\ &= \mathbf{Q}(\mathbf{R}\mathbf{B}\mathbf{R}^\top)\mathbf{Q}^\top \end{aligned}$$

Une astuce détaillée au paragraphe suivant permet d'avoir $\mathbf{Q}$ de dimension $n \times (p+m)$ et $\mathbf{R}$ de dimension $(p+m) \times (2p+1)$, avec $m = |I|$ le nombre d'annotations dans $\mathbf{y}$. Cela permet une décomposition spectrale rapide de $\mathbf{R}\mathbf{B}\mathbf{R}^\top$ qui a pour dimensions $(p+m) \times (p+m)$. En notant $\mathbf{W}$ la matrice orthogonale des vecteurs propres et $\mathbf{M}$ la matrice diagonale des valeurs propres de $\mathbf{R}\mathbf{B}\mathbf{R}^\top$, nous avons :

$$\begin{aligned} \mathbf{K}^\star &= \mathbf{Q}(\mathbf{R}\mathbf{B}\mathbf{R}^\top)\mathbf{Q}^\top \\ &= \mathbf{Q}(\mathbf{W}\mathbf{M}\mathbf{W}^\top)\mathbf{Q}^\top \\ &= (\mathbf{Q}\mathbf{W})\mathbf{M}(\mathbf{Q}\mathbf{W})^\top \\ &= \mathbf{V}^\star\mathbf{\Lambda}^\star(\mathbf{V}^\star)^\top \end{aligned}$$

Nous obtenons ainsi la décomposition de $\mathbf{K}^\star$ par la matrice des vecteurs propres $\mathbf{V}^\star = \mathbf{Q}\mathbf{W}$ et celle des valeurs propres $\mathbf{\Lambda}^\star = \mathbf{M}$ sachant que $\mathbf{Q}$ et $\mathbf{W}$ sont des matrices orthogonales. L'approximation se situe à ce niveau de l'algorithme : nous ne conservons que les p plus fortes valeurs propres de $\mathbf{\Lambda}^\star$. En notant J l'ensemble des indices des diagonales de plus grandes valeurs dans $\mathbf{\Lambda}^\star$, l'approximation de $\mathbf{K}^\star$ est donnée par :

$$\hat{\mathbf{K}} = \hat{\mathbf{V}}\hat{\mathbf{\Lambda}}\hat{\mathbf{V}}^\top$$

avec $\forall j \in [1..p]\ \hat{\Lambda}_{jj} = \Lambda^\star_{J(j),J(j)}$ et $\forall i \in [1..n], j \in [1..p]\ \hat{V}_{ij} = V^\star_{i,J(j)}$.

Pour un système de recherche nécessitant une décomposition spectrale de la matrice de Gram, ce résultat est intéressant puisque la décomposition a déjà été calculée par notre méthode d'apprentissage long terme. Il est aussi possible d'obtenir une représentation vectorielle sachant que :

$$\hat{\mathbf{X}}^\top = \hat{\mathbf{V}}\sqrt{\hat{\mathbf{\Lambda}}}$$

Le nombre p d'espaces propres à conserver lors de l'approximation dépend surtout du nombre de catégories que vont rechercher les utilisateurs. Comme nous l'avons fait remarquer à la fin de la discussion autour de la construction d'une fonction noyau, le nombre de valeurs propres non nulles est une limite au nombre de catégories qui peuvent être apprises (*cf* Eq. 8.6). Ainsi, si nous avons un ordre d'idée du nombre de catégories qui vont être recherchées sur la base, cela suggère de fixer un nombre p d'espaces propres à conserver (supérieur au nombre de catégories). Lors d'expérimentations, nous avons pu observer qu'en fixant un nombre p largement supérieur au nombre de catégories, le nombre de valeurs propres significatives à la fin de l'optimisation était légèrement supérieur au nombre de catégories. Par exemple, avec une approximation $p = 100$ sur la base COREL avec 50 catégories, il y a environ 53 valeurs propres significatives lorsque la méthode se stabilise.

8.4.2 Factorisation en $\mathbf{ABA}^\top$

Afin de factoriser $\mathbf{K}^\star$ en $\mathbf{ABA}^\top$, nous utilisons une astuce qui permet de transformer une somme pondérée de matrices factorisables.

En effet, sachant que $\mathbf{K} = \mathbf{X}^\top\mathbf{X}$, nous pouvons factoriser chaque élément de la somme :

$$\mathbf{K}^\star = (1-\rho)\mathbf{K} + \rho a(\mathbf{TKT}^\top + b\mathbf{uu}^\top)$$

en :

$$\begin{array}{ccccc} \mathbf{K} & = & \mathbf{X}^\top\mathbf{X} & = & (\mathbf{X}^\top)(\mathbf{X}) \\ \mathbf{TKT} & = & \mathbf{TX}^\top\mathbf{XT}^\top & = & (\mathbf{TX}^\top)(\mathbf{TX}^\top)^\top \\ \mathbf{uu}^\top & = & \mathbf{uu}^\top & = & (\mathbf{u})(\mathbf{u})^\top \end{array}$$

Alors, si nous placons tous les parties de ces factorisations dans une matrice $\mathbf{A}$:

$$\mathbf{A} = \begin{pmatrix} \mathbf{X}^\top & \mathbf{TX}^\top & \mathbf{u} \end{pmatrix} \tag{8.12}$$

et tous les coefficients dans une matrice $\mathbf{B}$:

$$\mathbf{B} = \begin{pmatrix} (1-\rho)\mathbf{E}_p & 0 & 0 \\ 0 & \rho a\mathbf{E}_p & 0 \\ 0 & 0 & \rho ab \end{pmatrix} \tag{8.13}$$

Nous obtenons la décomposition désirée, avec $\mathbf{E}_p$ la matrice identité de dimension $p \times p$:

$$\begin{aligned}\hat{\mathbf{K}} &= (\mathbf{X}^\top)(1-\rho)(\mathbf{X}^\top)^\top + (\mathbf{T}\mathbf{X}^\top)\rho a(\mathbf{T}\mathbf{X}^\top)^\top + \mathbf{u}\rho ab\mathbf{u}^\top \\ &= \begin{pmatrix} \mathbf{X}^\top & \mathbf{T}\mathbf{X}^\top & \mathbf{u} \end{pmatrix} \begin{pmatrix} (1-\rho)\mathbf{E}_p & 0 & 0 \\ 0 & \rho a\mathbf{E}_p & 0 \\ 0 & 0 & \rho ab \end{pmatrix} \begin{pmatrix} \mathbf{X}^\top & \mathbf{T}\mathbf{X}^\top & \mathbf{u} \end{pmatrix}^\top \\ &= \mathbf{A}\mathbf{B}\mathbf{A}^\top\end{aligned}$$

La matice $\mathbf{A}$ a pour dimensions $n \times (2p+1)$ et $\mathbf{B}$ a pour dimensions $(2p+1) \times (2p+1)$, avec p la dimension des vecteurs de $\mathbf{X}$.

8.4.3 Calcul de la décomposition QR de A

Nous présentons ici un algorithme pour la décomposition QR de $\mathbf{A}$ en $O(n)$.

Le but de cet algorithme est de permettre la décomposition avec le moins de calcul possible. Nous ne faisons pas une décomposition QR au sens strict, nous souhaitons avant tout faire apparaître une matrice orthogonale que multiplie une matrice de petite taille. Dans la présentation qui suit, nous supposons que les indices ont été permutés de sorte que les indices des images annotées soient les premiers (*cf* Eq. 8.10). Ceci n'entraîne aucune perte de généralité.

Si nous observons la matrice $\mathbf{A}$ (*cf.* Eq. 8.12), nous pouvons voir que les deux grandes colonnes $\mathbf{X}^\top$ et $\mathbf{T}\mathbf{X}^\top$ ne diffèrent que sur les n_1 lignes auxquelles correspondent les indices des images positives. De même, $\mathbf{u}$ est nul sur les indices des images non annotées. Si nous notons $\mathbf{X}_l$ la matrice de l'ensemble des vecteurs annotés, et $\mathbf{X}_u$ la matrice de l'ensemble des vecteurs non annotés, nous pouvons alors écrire que :

$$\mathbf{A} = \begin{pmatrix} \mathbf{X}_l^\top & \mathbf{T}_l\mathbf{X}_l^\top & \mathbf{u} \\ \mathbf{X}_u^\top & \mathbf{X}_u^\top & 0 \end{pmatrix}$$

Si nous introduisons la décomposition QR de $\mathbf{X}_u^\top = \mathbf{Q}_u\mathbf{R}_u$, avec $\mathbf{Q}_u$ de dimension $(n-m) \times p$ et $\mathbf{R}_u$ de dimension $p \times p$, compte tenu du fait que $p < n$, nous obtenons :

$$\mathbf{A} = \begin{pmatrix} \mathbf{X}_l^\top & \mathbf{T}_l\mathbf{X}_l^\top & \mathbf{u} \\ \mathbf{Q}_u\mathbf{R}_u & \mathbf{Q}_u\mathbf{R}_u & 0 \end{pmatrix}$$

Ce qui nous permet d'obtenir la décomposition suivante :

$$\mathbf{A} = \underbrace{\begin{pmatrix} \mathbf{E}_m & 0 \\ 0 & \mathbf{Q}_u \end{pmatrix}}_{\mathbf{Q}} \underbrace{\begin{pmatrix} \mathbf{X}_l^\top & \mathbf{T}_l\mathbf{X}_l^\top & \mathbf{u} \\ \mathbf{R}_u^\top & \mathbf{R}_u^\top & 0 \end{pmatrix}}_{\mathbf{R}}$$

$\mathbf{Q}$ est orthogonale de dimension $n \times (p+m)$, et $\mathbf{R}$ de dimension $(p+m) \times (2p+1)$.

8.4.4 Calcul de a et b

Le paramètre b est calculé de sorte que les diagonales d'indice dans I_1 de $\mathbf{TKT}^\top + b\mathbf{u}\mathbf{u}^\top$ soient égales à 1. La matrice $\mathbf{TKT}^\top$ a une seule et unique valeur aux indices des images positives, qui est une moyenne de toutes les similarités de $\mathbf{K}$ pour ces mêmes indices :

$$\forall i,j \in I_1^2 \quad (\mathbf{TKT}^\top)_{ij} = \frac{1}{n_1^2} \sum_{i',j' \in I_1} K_{i'j'}$$

Sachant que $\mathbf{u}\mathbf{u}^\top$ a une valeur de 1 aux mêmes indices, alors il suffit de poser :

$$b = 1 - \frac{1}{n_1^2} \sum_{i',j' \in I_1} K_{i'j'}$$

pour assurer que la matrice $\mathbf{TKT}^\top + b\mathbf{u}\mathbf{u}^\top$ ait des valeurs de 1 aux indices I_1.

Le paramètre a est calculé de sorte que $\sum_{ij} \hat{K}_{ij} = \sum_{ij} K_{ij}$. Nous commençons par calculer une et une seule fois la somme s_0 des K_{ij} lors de l'initialisation de l'algorithme. Ce calcul n'est plus effectué par la suite, et nous le remplaçons par le calcul suivant.

Sachant que l'opération $\mathbf{TKT}^\top$ ne change pas la somme des K_{ij}, seules les valeurs ajoutées par $b\mathbf{u}\mathbf{u}^\top$ doivent être prises en compte. Il suffit donc de calculer la somme s des $u_i u_j$ pour déterminer a, sachant que :

$$\begin{array}{rccc}
 & \sum_{ij} K_{ij} & = & \sum_{ij} K^\star_{ij} \\
\Leftrightarrow & \sum_{ij} K_{ij} & = & \sum_{ij} \Big(1-\rho)\mathbf{K} + \rho a(\mathbf{TKT}^\top + b\mathbf{u}\mathbf{u}^\top) \Big)_{ij} \\
\Leftrightarrow & \sum_{ij} K_{ij} & = & (1-\rho) \sum_{ij} K_{ij} + \rho a(\sum_{ij} (\mathbf{TKT}^\top)_{ij} + b \sum_{ij} u_i u_j) \\
\Leftrightarrow & s_0 & = & (1-\rho) s_0 + \rho a(s_0 + bs) \\
\Leftrightarrow & 0 & = & -\rho s_0 + \rho a(s_0 + bs) \\
\Leftrightarrow & a & = & \frac{s_0}{s_0 + bs}
\end{array}$$

avec :

$$\begin{array}{rcl}
s & = & \sum_{i,j \in I^2} u_i u_j \\
 & = & \sum_{i,j \in I_1^2} u_i u_j + \sum_{i,j \in I_{-1}^2} u_i u_j + \sum_{i,j \in I_1 \times I_{-1}} u_i u_j + \sum_{i,j \in I_{-1} \times I_1} u_i u_j \\
 & = & n_1^2 + \gamma^2 n_{-1}^2 - 2\gamma n_1 n_{-1} \\
 & = & (n_1 - \gamma n_{-1})^2
\end{array}$$

La somme s_0 des K_{ij} peut être calculée à moindre frais lors de l'initialisation de l'algorithme :

$$\begin{aligned}\sum_{ij} K_{ij} &= \sum_{ij}\sum_{l} x_{li}x_{lj} \\ &= \sum_{l} \underbrace{\sum_{i} x_{li}}_{z_l} \underbrace{\sum_{j} x_{lj}}_{z_l} \\ &= \mathbf{z}^\top \mathbf{z}\end{aligned}$$

avec $z_l = \sum_i x_{li}$

8.4.5 Complexité

La plupart des opérations de l'algorithme ont une complexité négligeable devant la taille de la base (n), sous réserve que le nombre m d'annotations non nulles dans $\mathbf{y}$ et la dimension p des vecteurs le sont aussi. Deux opérations consomment la plus grande partie du temps de calcul :

- La décomposition QR de $\mathbf{X}_u^\top$: complexité $O(n)$ puisque $\mathbf{X}_u$ a pour dimensions $(n-m) \times p$;
- La multiplication matricielle $\mathbf{QW}$: complexité aussi $O(n)$ puisque $\mathbf{Q}$ et $\mathbf{W}$ ont pour dimensions respectives $n \times (p+m)$ et $(p+m) \times p$

8.4.6 Algorithme

L'algorithme est initialisé en effectuant une approximation du calcul de la KPCA (Schlkopf et al., 1998), puis l'équation $\hat{\mathbf{X}}^\top = \hat{\mathbf{V}}\sqrt{\hat{\mathbf{\Lambda}}}$ est appliquée.

TAB. 8.1: Mise à jour sémantique basée matrice de Gram.

fonction $\text{update}_{\text{gram}}(\mathbf{X}, \mathbf{y}, \rho \rightarrow \hat{\mathbf{X}})$

$I = (\mathbf{y} \neq 0); \bar{I} = (\mathbf{y} == 0)$

– Calcul des paramètres

$\forall l \in [1..p]\ z_l = \sum_i x_{I(l),i}$

$b = 1 - \frac{1}{n_1^2}\mathbf{z}^\top\mathbf{z}$

$s = (n_1 - \gamma n_{-1})^2$

$a = \frac{s_0}{s_0 + bs}$

– Calcul de la QR de A

$\mathbf{X}_u = \mathbf{X}(:, \bar{I})$

$\mathbf{Q}_u, \mathbf{R}_u = \text{qr}(\mathbf{X}_u^\top)$

$\mathbf{Q} = \begin{pmatrix} \mathbf{E}_m & 0 \\ 0 & \mathbf{Q}_u \end{pmatrix}$

...

...

$$\mathbf{R} = \begin{pmatrix} \mathbf{X}_l^\top & \mathbf{T}_l\mathbf{X}_l^\top & \mathbf{u} \\ \mathbf{R}_u^\top & \mathbf{R}_u^\top & 0 \end{pmatrix}$$

– *Calcul des valeurs propres*

$$\mathbf{B} = \begin{pmatrix} (1-\rho)\mathbf{E}_p & 0 & 0 \\ 0 & \rho a\mathbf{E}_p & 0 \\ 0 & 0 & \rho ab \end{pmatrix}$$

$$\mathbf{W}, \mathbf{M} = \text{eig}(\mathbf{R}\mathbf{B}\mathbf{R}^\top)$$

$$J = \text{argsort}(\text{diag}(\mathbf{M}))$$

$$\hat{\mathbf{\Lambda}} = \mathbf{M}(J, J)$$

$$\hat{\mathbf{V}} = \mathbf{Q}\mathbf{W}(:, J)$$

– *Calcul final*

$$\hat{\mathbf{X}} = \hat{\mathbf{V}}\sqrt{\hat{\mathbf{\Lambda}}}$$

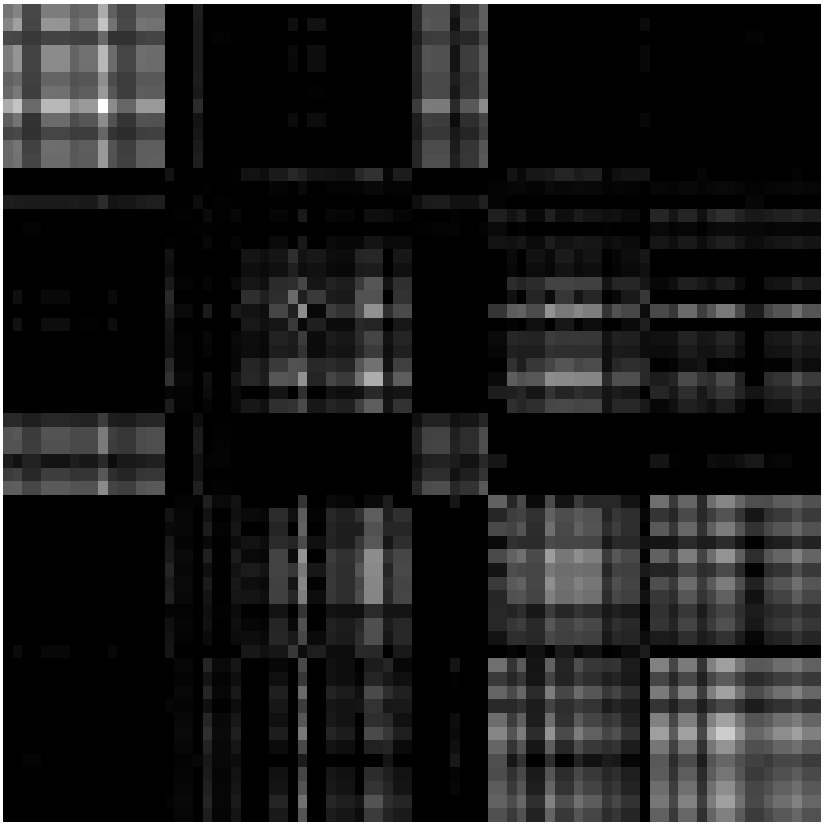
8.5 Exemple sur données synthétiques

Nous présentons dans ce paragraphe un résultat d'optimisation sur une base synthétique. Nous avons construit une base de 120 vecteurs répartis en 3 catégories mélangées de tailles différentes. Certains vecteurs appartiennent exclusivement à une catégorie, et d'autres appartiennent à deux catégories à la fois. Lors de la fabrication de cette base, les vecteurs ont été distribués selon 6 gaussiennes : 3 pour les vecteurs à une seule classe, 3 pour les vecteurs à deux classes. Chaque lot de vecteurs correspondant à une même gaussienne ont des indices consécutifs. Les centres des gaussiennes ont été placés aléatoirement. La matrice des similarités avant optimisation est présentée en figure 8.1(a). Les points blancs représentent une forte similarité, et les points noirs une faible similarité.

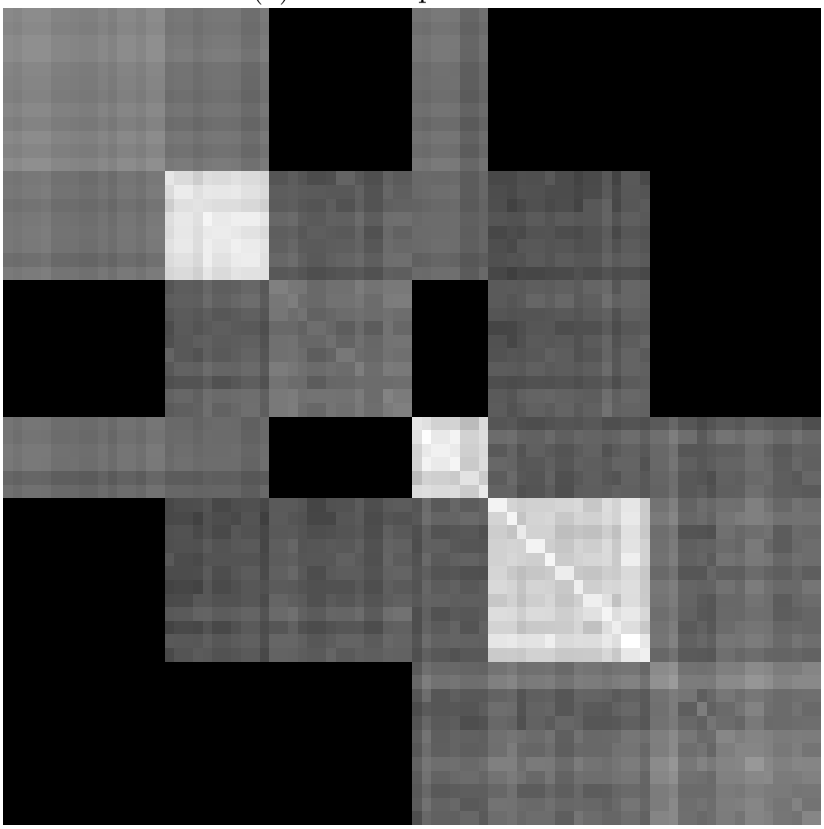
Nous avons ensuite construit aléatoirement un ensemble $\mathbf{Y}$ de 100 vecteurs d'annotations $\mathbf{y}_s$. Chaque vecteur annotation correspond à l'une des trois catégories, et comporte 10 valeurs non nulles tirées aléatoirement dans la catégorie à laquelle il appartient. La matrice des similarités après optimisation est présentée en figure 8.1(b). Nous observons sur cette figure que les vecteurs de même type (*i.e.* appartenant exclusivement à une catégorie, ou appartenant à deux catégories) ont été regroupés sous la forme de blocs autour de la diagonale. Nous pouvons aussi observer les blocs plus sombres hors de la diagonale qui représentent les similarités entre catégories.

8.6 Conclusion

La méthode inspirée de la discussion de (Cristianini et al., 2002) permet d'apprendre une matrice de noyau sémantique dans le contexte d'apprentissage faiblement supervisé



(a) Avant Optimisation



(b) Après Optimisation

Fig. 8.1 – Exemple d'optimisation avec la méthode basée Gram

présenté au chapitre 7. Elle offre la possibilité de gérer l'asymétrie entre les annotations positives et négatives tout en offrant une capacité à généraliser à partir de peu de données d'apprentissage. Un point important de la méthode concerne sa capacité à effectuer ces opérations tout en conservant la propriété *sdp* de la matrice des similarités. Enfin, l'algorithme présenté permet une mise à jour en $O(n)$.

Le cadre de travail de cette méthode nous permet de poser de nouvelles questions quant aux relations qui existent entre le spectre de la matrice de Gram et la capacité d'apprentissage qui en découle. Nous avons fait remarquer que le nombre de catégories sur la base semble lié au nombre d'espaces propres, fait que nous avons pu observer lors d'expérimentations.

Chapitre 9

Méthode basée vecteurs

Nous présentons dans ce chapitre une méthode alternative à celle basée matrice de Gram, qui travaille dans un espace de signatures sémantiques. Elle vise à offrir un paramétrage plus intuitif des règles de mise à jour en s'affranchissant des manipulations matricielles qui doivent conserver la propriété de *sdp*.

La méthode, baptisée *RETIN Concept Vector Learner*, a fait l'objet de publications (Gosselin & Cord, 2005b; Gosselin & Cord, 2006).

9.1 Motivations

L'idée initiale de cette méthode repose sur l'analyse de l'opérateur introduit dans le chapitre 8 qui à $\mathbf{K}$ fait correspondre $\mathbf{TKT}^\top$ Cet opérateur est équivalent à la modification suivante :

$$\begin{array}{ll} \forall i \in I_1 & \mathbf{x}_i \leftarrow \frac{1}{n_1} \sum_{j \in I_1} \mathbf{x}_j \\ \forall i \notin I_1 & \mathbf{x}_i \leftarrow \mathbf{x}_i \end{array}$$

avec I_1 l'ensemble des indices des images annotées positivement selon $\mathbf{y}$, et $n_1 = |I_1|$.

Cette modification revient à déplacer chaque image annotée positivement vers le barycentre des images annotées positivement. Nous nous sommes alors intéressés à l'espace de signatures (induit par une fonction noyau), travaillant directement sur des déplacements de vecteurs. L'idée est de déplacer chaque vecteur $\mathbf{x}_i$ dans une certaine direction, de sorte que la similarité[1] entre les images reflète l'information contenue dans la matrice $\mathbf{Y}$.

Le premier objectif de cette approche est de regrouper les vecteurs d'une même catégorie de sorte qu'ils forment un cluster. Toute la difficulté réside dans le fait que

[1]La notion de similarité que nous avons choisie ici est basée sur la distance euclidienne qui sépare deux vecteurs. De plus, si tous les vecteurs ont une même norme, la distance euclidienne ne dépend plus que du produit scalaire. Dans ce cas, la notion de similarité est la même que pour la méthode basée Gram.

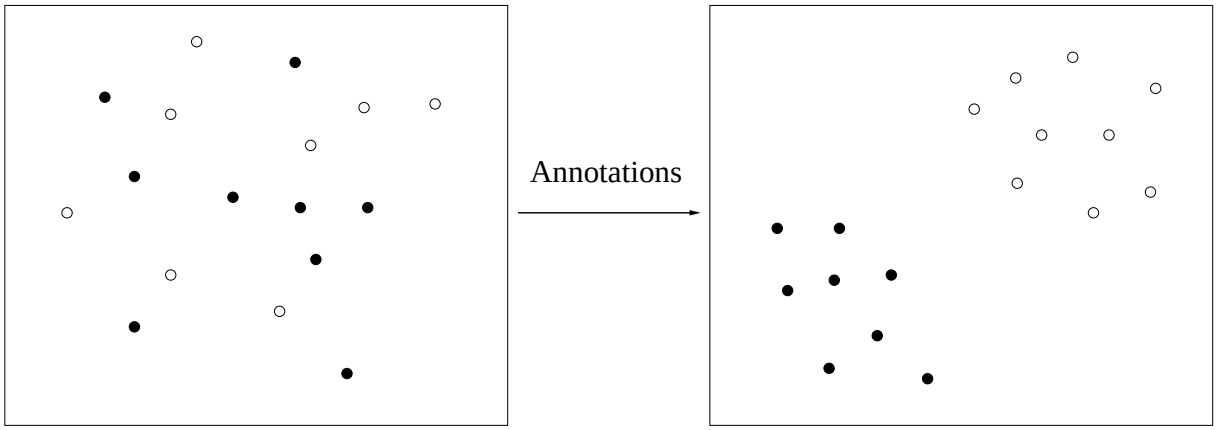


FIG. 9.1 – Principe de l'apprentissage basé vecteurs : les vecteurs sont déplacés en fonction des annotations, de sorte à les regrouper par catégorie.

nous n'avons qu'une information partielle à propos des catégories. En effet, dans chaque lot d'annotations $\mathbf{y}$, nous avons une information du type « ces images sont dans une certaine catégorie »et « ces autres images ne sont pas dans cette catégorie ». Autrement dit, avec une approche adaptative, nous ne pouvons atteindre le regroupement final que par morceaux. Par exemple, dans la figure 9.1, nous aurons une information du type « ces deux points blancs sont dans la même catégorie »et « ce point noir n'est pas dans cette catégorie ». Nous avons à regrouper tous les points blancs et tous les points noirs à l'aide seulement d'un ensemble de relations de ce type.

Afin de formaliser cet objectif, nous introduisons la notion de *centre*. Chaque centre $\mathbf{g}_j$ est le représentant de la catégorie j qui existe par hypothèse, mais n'est jamais explicité. Si nous supposons que la méthode est effective et a convergé, *i.e.* que les images sont regroupées en clusters par catégories, alors les centres sont les barycentres de chacun de ces clusters. Ainsi, la méthode converge avec succès si les vecteurs sont déplacés de telle sorte qu'en appliquant une quantification vectorielle, on obtient autant de clusters que de catégories, et que chaque *centre* est le représentant d'un cluster. Chaque image $\mathbf{x}_i$ a alors pour représentant un des centres $\mathbf{g}_j$. Par exemple, sur la figure 9.1, les vecteurs après optimisation sont regroupés en deux clusters, et une quantification vectorielle de ces points donnerait deux représentants, un pour chaque cluster.

Cependant, il n'est pas possible de déterminer ces centres avant que la méthode n'ait convergée. De plus, les regroupements ne peuvent se faire que par lots. Il nous faut donc estimer, à chaque mise à jour, à quel centre chaque image peut correspondre. Cela revient à déterminer pour chaque $\mathbf{x}_i$ un vecteur $\hat{\mathbf{g}}_i$ le plus proche possible du centre $\mathbf{g}_j$ auquel correspond $\mathbf{x}_i$, et ce sans jamais connaître $\mathbf{g}_j$.

Cette approche peut s'exprimer comme suit :

$$\forall i \in I \quad \mathbf{x}_i \leftarrow (1-\rho)\mathbf{x}_i + \rho\hat{\mathbf{g}}_i \tag{9.1}$$

Toute la difficulté réside dans la détermination des $\hat{\mathbf{g}}_i$ de sorte que l'objectif soit atteint.

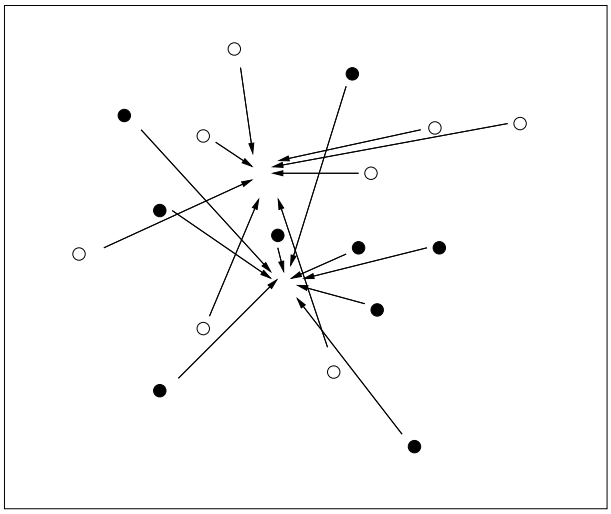


FIG. 9.2 – Première approche : les vecteurs sont déplacés vers leur barycentre.

A partir de ce schéma de mise à jour, nous allons maintenant présenter et discuter des méthodes de calcul des $\hat{\mathbf{g}}_i$.

9.1.1 Première approche

Une première approche consiste à déplacer les images de même annotation vers leur barycentre (cf. Fig. 9.2). Ainsi, les images positives sont déplacées vers le barycentre positif, et les images négatives vers le barycentre négatif :

$$\begin{array}{ll} \forall i \in I_1 & \hat{\mathbf{g}}_i = \hat{\mathbf{g}}^+ = \frac{1}{n_1} \sum_{j \in I_1} \mathbf{x}_j \\ \forall i \in I_{-1} & \hat{\mathbf{g}}_i = \hat{\mathbf{g}}^- = \frac{1}{n_{-1}} \sum_{j \in I_{-1}} \mathbf{x}_j \end{array}$$

Cette première approche ne fonctionne que dans certains cas, et lors des simulations elle s'est avérée inefficace sur des catégories largement mélangées. Tout d'abord, rien ne garantit que les centres de gravités soient à des endroits différents. Par exemple sur la figure 9.2, nous pouvons voir que les deux centres de gravités sont relativement proches. Lors des simulations à partir de données synthétiques générées aléatoirement, nous avons observé de nombreux cas où les deux centres de gravités sont très proches La méthode tente alors de regrouper des vecteurs de catégories différentes au même endroit, ce qui empêche toute discrimination entre les catégories. D'autres problèmes surviennent lorsque les catégories se mélangent, *i.e.* certaines images appartiennent à plusieurs catégories à la fois. Dans ce cas, les images qui appartiennent à plusieurs catégories sont rapprochées des images qui appartiennent à ces catégories, et ces mêmes images sont rapprochées des images multi-catégories. Au final, tous les vecteurs se rapprochent les uns des autres, et finissent par converger vers un unique point. Lors des simulations sur données synthétiques, ce comportement est systématique.

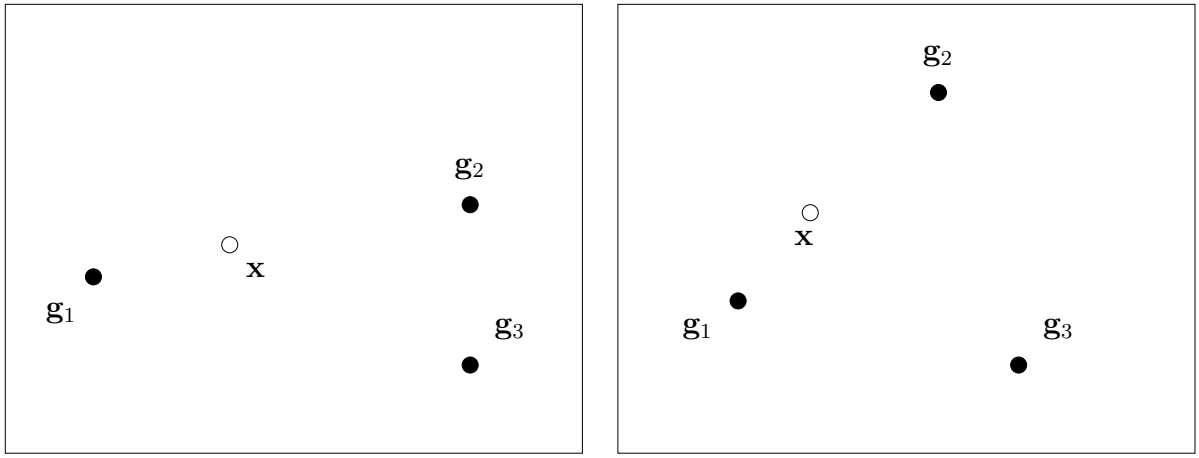


FIG. 9.3 – Centres non-équidistants(a) et équidistants(b).

Cette première méthode traite de la même manière les annotations positives et négatives. Or, comme nous l'avons déjà remarqué, donc notre contexte, le regroupement des exemples négatifs n'est pas pertinent. Nous avons proposé dans le chapitre 8 une adapation de la fonction noyau ($\mathbf{u}\mathbf{u}^\top$) prenant en compte cette asymétrie. Nous développons à la suite cette idée en proposant des déplacements spécifiques pour les annotations négatives.

9.2 RETIN CVL

Nous proposons d'introduire une contrainte sur la répartition des centres, en imposant que les centres soient équidistants. Nous présentons à la suite cette propriété, puis le calcul des centres pour les annotations positives et les annotations négatives en s'appuyant sur ce principe.

9.2.1 Equidistance des centres

Nous proposons d'utiliser la contrainte d'équidistance des centres, autrement dit que les points $\mathbf{g}_j$ soient à égale distance les uns des autres. Cela apporte un certain nombre d'avantages qui permettent le bon fonctionnement de la méthode.

Cette répartition des centres est tout d'abord intéressante pour représenter l'appartenance de chaque image à chacune des catégories. Sur la figure 9.3, nous présentons deux configurations avec 3 centres $\mathbf{g}_j$, dans un cas non équidistant (a) et dans un cas équidistant (b). Si nous considérons une image, par exemple le vecteur $\mathbf{x}$ sur la figure 9.3, l'appartenance à chacune des catégories correspondant aux 3 centres est la distance euclidienne qui sépare $\mathbf{x}$ à l'un des centres $\mathbf{g}_j$. Dans les deux cas de figure, nous pouvons représenter toutes

les appartenances possibles. Par exemple sur les figures 9.3(a) et (b), l'image $\mathbf{x}$ est entre les catégories 1 et 2, et est plus éloignée de la catégorie 3. Cependant, les déplacements que nous ferons pour modifier les appartenances dans le cas non équidistant dépendront de la répartition des centres, alors que dans le cas équidistant cette opération est plus simple. En effet, pour tout centre considéré, les autres centres sont toujours répartis de la même manière : ainsi un unique type de déplacement peut être considéré.

Une propriété qui découle de cette contrainte permet de travailler plus facilement avec le fait que les centres sont inconnus. En effet, si nous considérons $p+1$ vecteurs équidistants et de norme 1, alors la distance entre deux de ces vecteurs est toujours $\sqrt{2(1+\frac{1}{p})}$. Sachant cela, il suffit de déplacer les vecteurs vers des centres de même norme et séparés d'une certaine distance pour que les points s'agglutinent toujours autours de points équidistants. Cela permet d'assurer à la fois que les agglomérats soient en des lieux différents, et que les vecteurs appartenant à plusieurs catégories soient placés entre les points d'agglomération correspondants.

Cette propriété d'unicité de la distance découle du théorème suivant :

Théorème 1 *Soit* $G = \{\mathbf{g}_1, \ldots, \mathbf{g}_q\}$ *un ensemble de vecteurs* $\mathbf{g}_j \in \mathbb{R}^p$.

Si :

$$\forall j \in [1..q],\ ||\mathbf{g}_j|| = 1,$$

$$\exists d > 0\ \forall j, j' \in [1..q],\ ||\mathbf{g}_j - \mathbf{g}_{j'}||^2 = d$$

alors :

$$\textit{si } d = 2(1+\tfrac{1}{q-1}),\ \textit{alors } p \geq q-1$$

$$\textit{sinon } p \geq q.$$

Preuve du Théorème 1. Dans cette preuve nous utilisons le principe de dualité en analysant le rang de la matrice de Gram.

Soit $\mathbf{K} = \mathbf{G}^\top\mathbf{G}$ la matrice $q \times q$ de l'ensemble des produits scalaires entre tous les couples de vecteurs de $\mathbf{G} = (\mathbf{g}_1 \ldots \mathbf{g}_q)$. Etant donné que $\forall j, j' \in [1..q],\ ||\mathbf{g}_j|| = 1$ et $||\mathbf{g}_j - \mathbf{g}_{j'}||^2 = d$, il s'en suit que :

$$\langle \mathbf{g}_j, \mathbf{g}_{j'} \rangle = \frac{1}{2}(||\mathbf{g}_j||^2 + ||\mathbf{g}_{j'}||^2 - ||\mathbf{g}_j - \mathbf{g}_{j'}||^2) = 1 - \frac{d}{2}$$

Si nous posons $h = 1 - \frac{d}{2}$, alors $\mathbf{K}$ peut s'écrire :

$$\mathbf{K} = \begin{pmatrix} 1 & h & h & \dots & h \\ h & 1 & h & \dots & h \\ h & h & \ddots & & \vdots \\ \vdots & \vdots & & 1 & h \\ h & h & \dots & h & 1 \end{pmatrix}$$

$\mathbf{K}$ est la matrice des produits scalaires entre q vecteurs de dimension p, d'où $\operatorname{rang}\mathbf{K} \leq p$.

A présent regardons comment nous pouvons minimiser le rang de $\mathbf{K}$. Afin de calculer ce rang, nous nous intéressons au polynôme caractéristique de $\mathbf{K}$:

$$\det(\mathbf{K} - \lambda\mathbf{E}_q) = \det \begin{pmatrix} 1-\lambda & h & h & \dots & h \\ h & 1-\lambda & h & \dots & h \\ h & h & \ddots & & \vdots \\ \vdots & \vdots & & 1-\lambda & h \\ h & h & \dots & h & 1-\lambda \end{pmatrix}$$

avec $\mathbf{E}_q$ la matrice $q \times q$ identité.

Soit $\lambda = \lambda' - h + 1$, alors :

$$\det(\mathbf{K} - \lambda\mathbf{E}_q) = \det \begin{pmatrix} h-\lambda' & h & h & \dots & h \\ h & h-\lambda' & h & \dots & h \\ h & h & \ddots & & \vdots \\ \vdots & \vdots & & h-\lambda' & h \\ h & h & \dots & h & h-\lambda' \end{pmatrix} = \det(h\mathbf{u}\mathbf{u}^\top - \lambda'\mathbf{E}_q)$$

avec $\mathbf{u} = (1 \dots 1)^\top$.

Le polynôme caractéristique de la matrice $h\mathbf{u}\mathbf{u}^\top$ de rang 1 est :

$$\det(h\mathbf{u}\mathbf{u}^\top - \lambda'\mathbf{E}_q) = (-\lambda')^{q-1}(hq - \lambda')$$

Alors, sachant que $\lambda' = h - 1 + \lambda$:

$$\det(\mathbf{K} - \lambda\mathbf{E}_q) = \big((1-h) - \lambda\big)^{q-1}\big((1 + (q-1)h) - \lambda\big)$$

Les racines sont $1 - h$ and $1 + (q-1)h$.

Le rang de $\mathbf{K}$ est minimal si nous avons le maximum de racines nulles. Sachant que $d > 0$, alors $h \neq 1$, et la première racine ne peut être nulle.

Dans le cas où $d = 2(1 + \frac{1}{q-1})$, $h = -\frac{1}{q-1}$, et $\operatorname{rang} \mathbf{K} = q - 1$. Ainsi $p \geq q - 1$.

Dans le cas où $d \neq 2(1 + \frac{1}{q-1})$, $\operatorname{rang} \mathbf{K} = q$ et $p \geq q$.

Fin de la preuve du Théorème 1.

Le théorème 1 montre, en autres, que pour tout ensemble de $p+1$ vecteurs équidistants de dimension p et de norme 1, alors la distance qui sépare deux de ces vecteurs est toujours $\sqrt{2(1 + \frac{1}{p})}$:

Théorème 2 *Soit $G = \{\mathbf{g}_1, \ldots, \mathbf{g}_{p+1}\}$ un ensemble de vecteurs $\mathbf{g}_j \in \mathbb{R}^p$ de norme 1. Alors les assertions suivantes sont équivalentes :*

(i) les vecteurs de G sont équidistants

(ii) $\forall j, j' \in [1..p+1],\ ||\mathbf{g}_j - \mathbf{g}_{j'}||^2 = 2(1 + \frac{1}{p}) = d$

(iii) $\forall j, j' \in [1..p+1],\ \langle \mathbf{g}_j, \mathbf{g}_{j'} \rangle = -\frac{1}{p} = h$

Preuve du Théorème 2.

$(i) \Leftrightarrow (ii)$: Ceci ce déduit du Théorème 1 sachant que $q = p + 1$.

$(ii) \Leftrightarrow (iii)$: Ceci découle du fait que les vecteurs sont de norme 1, sachant que $||\mathbf{g}_j - \mathbf{g}_{j'}||^2 = \langle \mathbf{g}_j, \mathbf{g}_j \rangle + \langle \mathbf{g}_{j'}, \mathbf{g}_{j'} \rangle - 2\langle \mathbf{g}_j, \mathbf{g}_{j'} \rangle$.

Fin de la preuve du Théorème 2.

9.2.2 Estimation des centres positifs

Compte tenu du fait que les vecteurs d'annotation positive (*i.e.* $\mathbf{x}_i$ tels que $y_i > 0$) sont tous dans une même catégorie, nous proposons de les diriger vers un même centre. Cependant, nous ne proposons pas de les diriger vers leur centre de gravité, mais vers le centre de gravité de tous les vecteurs annotés :

$$\forall i \in I_1 \quad \hat{\mathbf{g}}_i = \mathbf{g} = \frac{\sum_{j \in I} y_j \mathbf{x}_j}{\left\| \sum_{j \in I} y_j \mathbf{x}_j \right\|}$$

Le centre de gravité est normalisé de manière à satisfaire le théorème 1. Nous proposons ce centre de gravité de manière à éviter que les vecteurs se déplacent tous dans la même

zone, ce qui conduit aux effets de la méthode naïve. Notons que si nous dirigeons les vecteurs négatifs vers l'opposé de ce centre, la méthode se comporte déjà beaucoup mieux que l'approche naïve.

Une autre motivation quant au choix de ce centre est lié à l'alignement du noyau (Cristianini et al., 2001). Dans un cas à deux catégories, rapprocher les vecteurs positifs de $\mathbf{g}$ et éloigner les vecteurs négatifs de $\mathbf{g}$ peut s'exprimer comme la minimisation sur $\mathbf{X}$ suivante :

$$\min_{\mathbf{X}} \sum_{i \in I_1} y_i ||\mathbf{x}_i - \mathbf{g}||^2 \tag{9.2}$$

Si nous supposons que tous les vecteurs sont normalisés, alors Eq. 9.2 est équivalente à :

$$\max_{\mathbf{X}} \sum_{i \in I_1} y_i \langle \mathbf{x}_i, \mathbf{g} \rangle \tag{9.3}$$

Sachant que $\mathbf{g} = cte \sum_j y_j \mathbf{x}_j$ on a :

$$\begin{array}{ll} & \max_{\mathbf{X}} \sum_i y_i \langle \mathbf{x}_i, \mathbf{g} \rangle \\ \iff & \max_{\mathbf{X}} \sum_i y_i \langle \mathbf{x}_i, \sum_j y_j \mathbf{x}_j \rangle \\ \iff & \max_{\mathbf{X}} \sum_{i,j} y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ \iff & \max_{\mathbf{X}} A_{\mathbf{X}}(\mathbf{y}) \end{array} \tag{9.4}$$

où $A_{\mathbf{X}}(\mathbf{y})$ est l'alignement entre le noyau linéaire $\mathbf{X}^\top \mathbf{X}$ et $\mathbf{y}$.

Ceci montre que la proposition liée aux centres de gravité est en rapport avec l'alignement du noyau. Cependant, cette approche est valable pour un problème à deux catégories : elle agglomère aussi les vecteurs négatifs. C'est la raison pour laquelle nous proposons une autre méthode pour le caclul des centres négatifs.

9.2.3 Estimation des centres négatifs

Notre seule hypothèse est que les vecteurs annotés négativement (*i.e.* $\mathbf{x}_i$ tels que $y_i < 0$) ne sont pas dans la catégorie des vecteurs annotés positivement. Nous ne cherchons donc pas à les regrouper dans une catégorie unique, mais proposons de les diriger vers des centres estimés.

Pour ce faire, nous nous reposons sur le théorème 1. Chaque vecteur négatif $\mathbf{x}_i$ est considéré indépendamment des autres, et est dirigé vers le centre estimé $\hat{\mathbf{g}}_i$ dans la direction $\mathbf{x}_i - \mathbf{g}$, de telle sorte que la distance au carré qui sépare $\mathbf{g}$ et $\hat{\mathbf{g}}_i$ soit $d = 2(1 + \frac{1}{p})$. Le calcul suivant repose sur la construction d'une base orthognale du plan engendré par

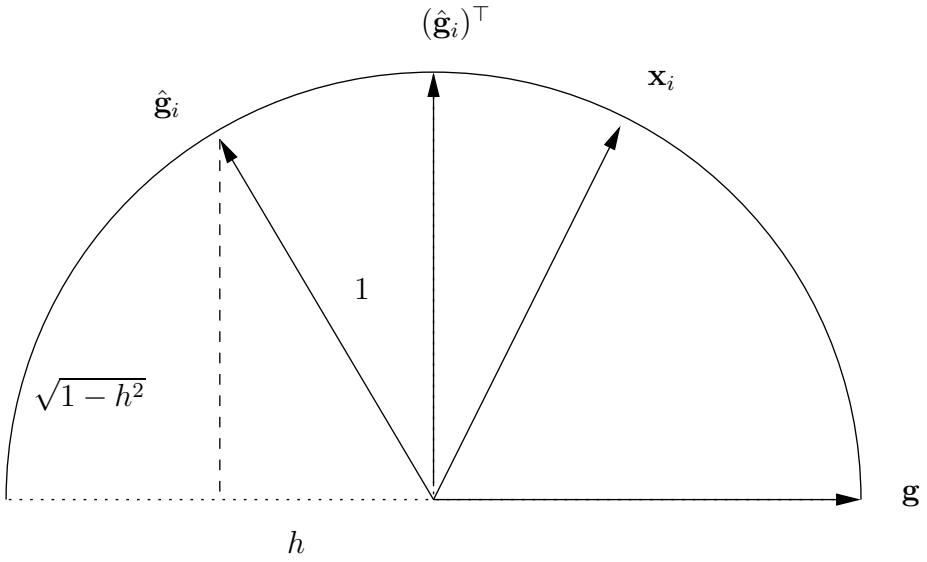


FIG. 9.4 – Centre possible $\hat{\mathbf{g}}_i$ pour un vecteur négatif $\mathbf{x}_i$, par rapport au barycentre des vecteurs annotés $\mathbf{g}$.

$\mathbf{x}_i$ et $\mathbf{g}$ (*cf.* figure 9.4). Une fois la base déterminée, il suffit alors de calculer $\hat{g}_i$ avec les coordonnées souhaitées.

Nous commençons par calculer une base $\{\mathbf{g}, (\hat{\mathbf{g}}_i)^\perp\}$ du plan engendré par $\mathbf{g}$ et $\mathbf{x}$:

$$\begin{aligned}(\hat{\mathbf{g}}_i)^\perp &= \frac{\mathbf{x}_i - < \mathbf{x}_i, \mathbf{g} > \mathbf{g}}{||\mathbf{x}_i - < \mathbf{x}_i, \mathbf{g} > \mathbf{g}||} \\ &= \frac{\mathbf{x}_i - < \mathbf{x}_i, \mathbf{g} > \mathbf{g}}{\sqrt{< \mathbf{x}_i, \mathbf{x}_i >^2 - < \mathbf{x}_i, \mathbf{g} >^2}}\end{aligned}$$

Puisque nous voulons $||\mathbf{g} - \hat{\mathbf{g}}_i||^2 = d$, alors $< \mathbf{g}, \hat{\mathbf{g}}_i >= h = -\frac{1}{p}$, et :

$$\begin{aligned}\hat{\mathbf{g}}_i &= h\mathbf{g} + \sqrt{1-h^2}(\hat{\mathbf{g}}_i)^\perp \\ &= h\mathbf{g} + \sqrt{\frac{1-h^2}{< \mathbf{x}_i, \mathbf{x}_i >^2 - < \mathbf{x}_i, \mathbf{g} >^2}}(\mathbf{x}_i - < \mathbf{x}_i, \mathbf{g} > \mathbf{g}) \\ &= \left(h - < \mathbf{x}_i, \mathbf{g} > \sqrt{\frac{1-h^2}{< \mathbf{x}_i, \mathbf{x}_i >^2 - < \mathbf{x}_i, \mathbf{g} >^2}}\right)\mathbf{g} \\ &\quad + \sqrt{\frac{1-h^2}{< \mathbf{x}_i, \mathbf{x}_i >^2 - < \mathbf{x}_i, \mathbf{g} >^2}}\mathbf{x}_i\end{aligned}$$

9.2.4 Algorithme

L'algorithme de mise à jour final est présenté sur la table 9.1. La complexité de cet algorithme repose sur le nombre m d'annotations non nulles dans le vecteur $\mathbf{y}$. Chaque mise à jour est un calcul vectoriel dont la complexité dépend de la dimension des vecteurs p. Si p et m sont négligeables devant la taille de la base, alors la complexité de la mise à jour est négligeable devant la taille de la base.

TAB. 9.1: Mise à jour sémantique basée vecteurs.

fonction $\text{update}_{\text{vector}}(\mathbf{X}, \mathbf{y}, \rho)$

$h = -\frac{1}{p}$
$\mathbf{g} = \frac{\sum_j y_j \mathbf{x}_j}{\|\sum_j y_j \mathbf{x}_j\|}$
pour $\mathbf{x}_i$ **dans** $\mathbf{X}$
  si $y_i > 0$ **alors**
    $\mathbf{x}_i = \mathbf{x}_i + \rho |y_i| (\mathbf{g} - \mathbf{x}_i)$
  sinon si $y_i < 0$ **alors**
    $r = < \mathbf{x}_i, \mathbf{g} >$
    $s = \sqrt{\frac{1-h^2}{<\mathbf{x}_i, \mathbf{x}_i>^2 - r^2}}$
    $\mathbf{x}_i = \mathbf{x}_i + \rho\big((h - rs)\mathbf{g} + (s-1)\mathbf{x}_i\big)$
  finsi
finpour

9.3 Exemple sur données synthétiques

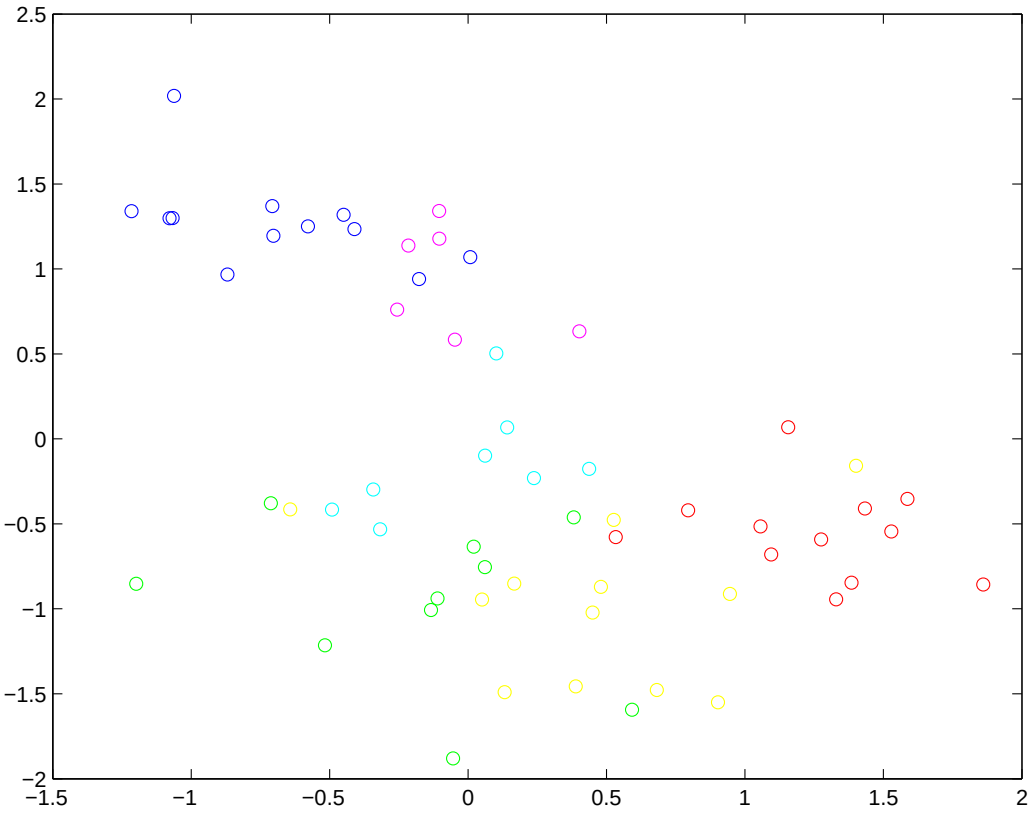
Nous présentons dans ce paragraphe un résultat d'optimisation sur une base synthétique. Nous utilisons la même base qu'au chapitre précédent de 120 vecteurs répartis en 3 catégories mélangées de tailles différentes. Certains vecteurs appartiennent exclusivement à une catégorie, et d'autres appartiennent à deux catégories à la fois.

Nous avons ensuite construit aléatoirement un ensemble $\mathbf{Y}$ de 100 vecteurs d'annotations $\mathbf{y}_s$. Chaque vecteur annotation correspond à l'une des trois catégories, et comporte 10 valeurs non nulles tirées aléatoirement dans la catégorie à laquelle il correspond.

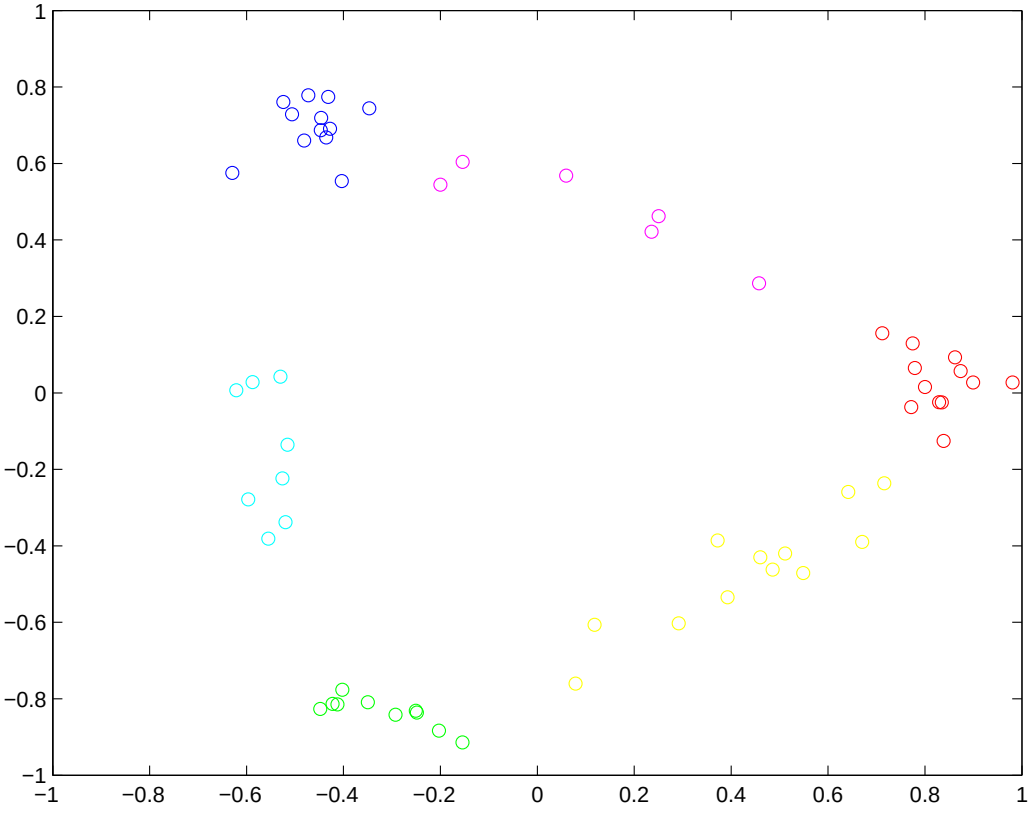
Dans les figures 9.5 et 9.6, les points bleus (resp. rouges et verts) appartiennent exclusivement à la catégorie 1 (resp. 2 et 3). Les points magentas appartiennent aux catégories 1 et 2, les points jaunes aux catégories 2 et 3, et les points cyans aux catégories 1 et 3.

9.3.1 Cas où $p = q - 1$

Nous présentons tout d'abord un exemple où les conditions du théorème 2 sont respectées, autrement dit où la dimension de l'espace engendré par les vecteurs (p) est égale

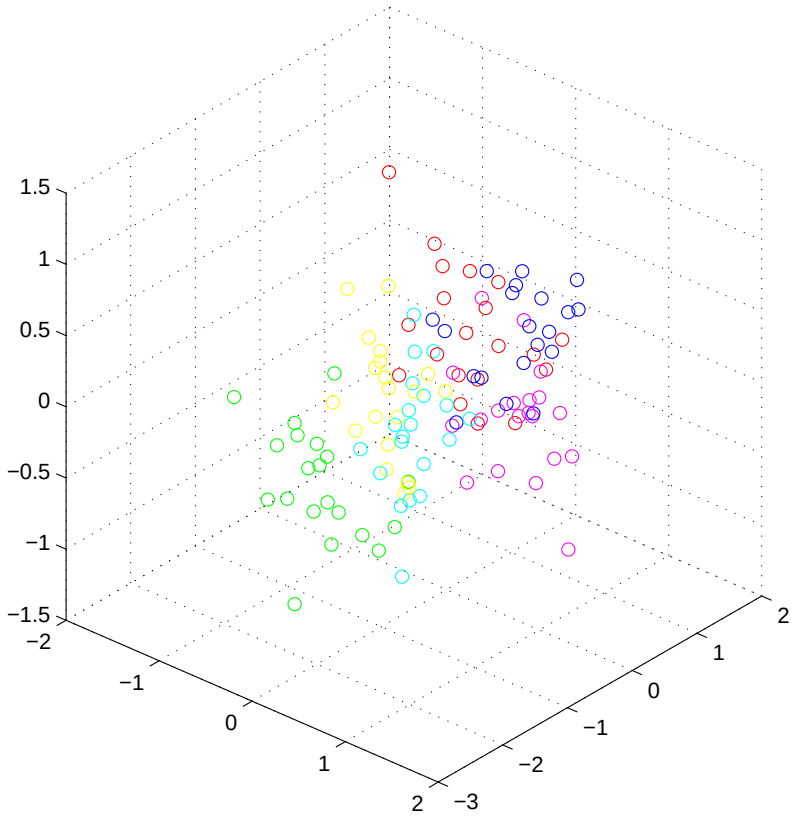


(a) Avant Optimisation

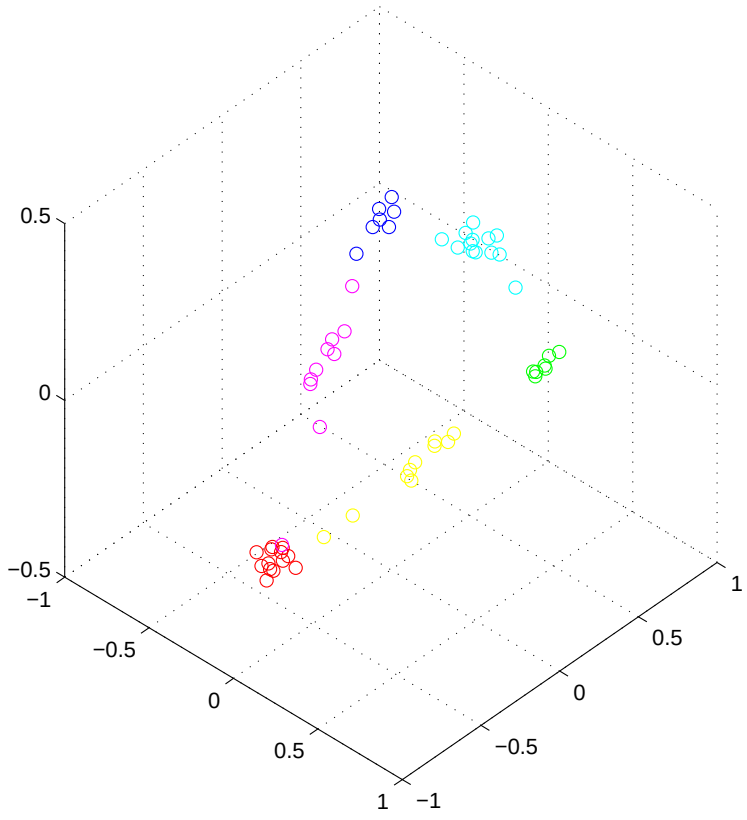


(b) Après Optimisation

Fig. 9.5 – Exemple d'optimisation avec la méthode basée vecteurs, dans un espace dont la dimension (2) est égale au nombre de catégories(3) moins un.



(a) Avant Optimisation



(b) Après Optimisation

FIG. 9.6 – Exemple d'optimisation avec la méthode basée vecteurs, dans un espace dont la dimension (3) est supérieur au nombre de catégories(3) moins un.

au nombre de catégories (q) moins 1. Dans le cas de la figure 9.5, $p = 2$ et $q = 3$.

Les vecteurs avant optimisation sont présentés en figure 9.5(a) et après optimisation sont présentés en figure 9.5(b).

Nous pouvons observer sur la figure 9.5(b) que les vecteurs appartenant exclusivement à une catégorie ont été regroupés en clusters équidistants. Les vecteurs appartenant à deux catégories ont été regroupés entre les clusters, à une distance qui dépend du nombre de fois qu'ils ont été annotés dans l'une ou l'autre des catégories.

9.3.2 Cas où $p > q - 1$

Nous présentons ici un exemple où la dimension de l'espace engendré par les vecteurs (p) est strictement supérieure au nombre de catégories (q) moins 1. Dans le cas de la figure 9.6, $p = 3$ et $q = 3$. Dans ce cas, les conditions du théorème 2 ne sont pas respectées.

Les vecteurs avant optimisation sont présentés en figure 9.6(a) et après optimisation sont présentés en figure 9.6(b).

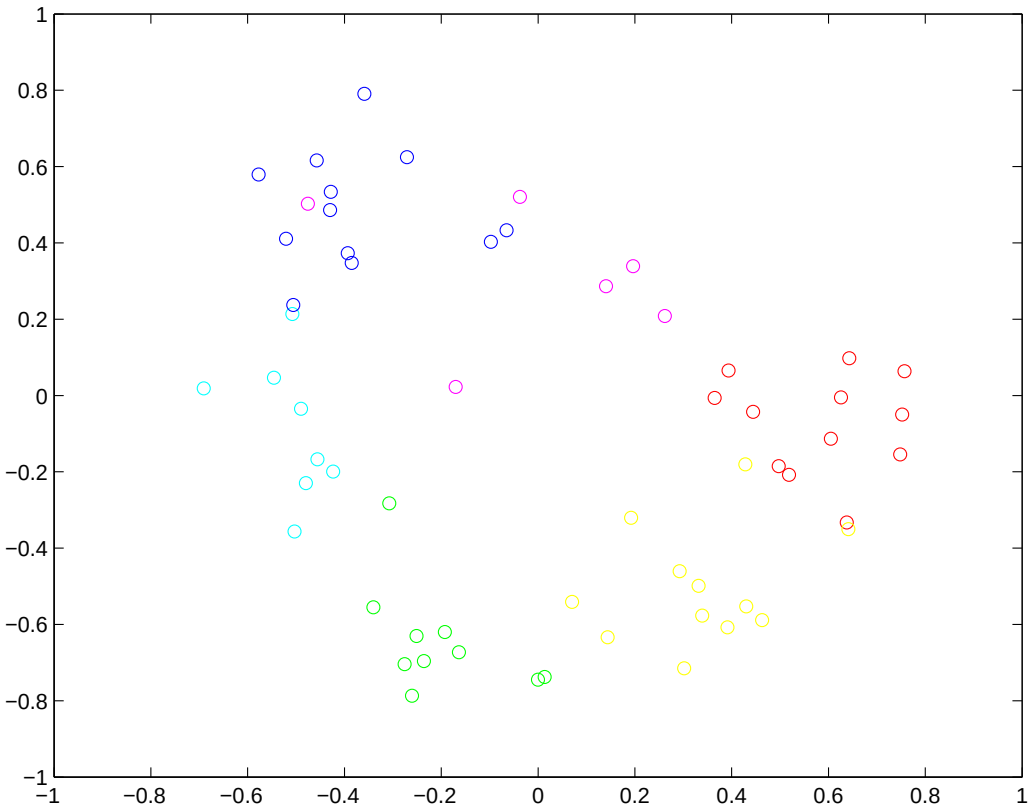
Nous pouvons observer sur la figure 9.6(b) que les vecteurs se regroupent dans un plan au même titre que dans le cas $p = q - 1$. Cela est dû au fait que des vecteurs équidistants dans un espace sont aussi équidistants dans un sous-espace. De plus, étant donné que l'algorithme tente de regrouper les vecteurs de mêmes catégories, ils finissent par se regrouper en agglomérats équidistants. En quelque sorte, l'algorithme se comporte comme s'il y avait 4 catégories, mais dont une est vide.

9.3.3 Cas où $p < q - 1$

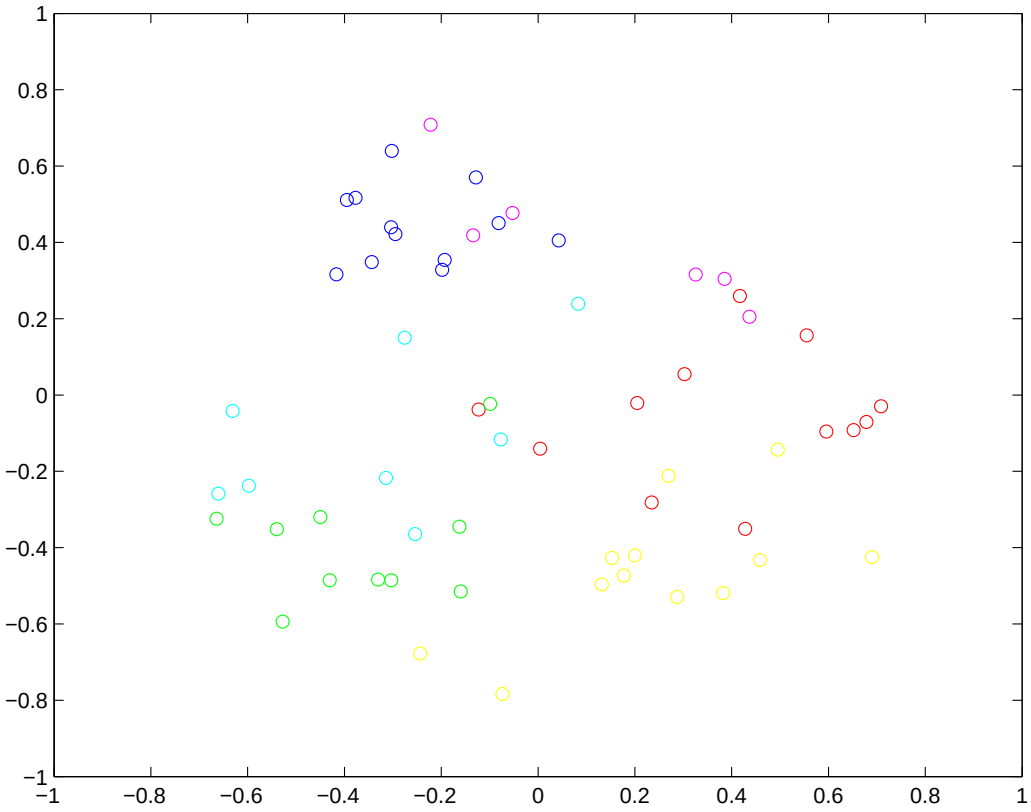
Dans le cas où la dimension de l'espace engendré par les vecteurs (p) est strictement inférieure au nombre de catégories (q) moins 1, l'algorithme est incapable de converger. Les vecteurs sont déplacés indéfiniment, et le résultat final ressemble à un brassage aléatoire des vecteurs. L'incapacité à gérer ce cas consitue le principal défaut de cet algorithme. Il est donc impératif que le nombre de dimensions de l'espace engendré par les vecteurs de $\mathbf{X}$ soit supérieur au nombre de catégories, ce qui est souvent le cas dans notre contexte où les signatures sont souvent très grandes.

9.3.4 Annotations erronées

Dans les expérimentations précédentes, nous avons supposé qu'il n'y avait pas d'erreur dans la matrice $\mathbf{Y}$. Cependant, dans une application réelle, des annotations erronées peuvent apparaître lorsque les données sont collectées au travers de l'interaction avec les utilisateurs.



(a) Après Optimisation avec 10% d'annotations erronées



(b) Après Optimisation avec 20% d'annotations erronées

Fig. 9.7 – Exemple d'optimisation avec la méthode basée vecteurs, dans le cas où l'ensemble d'apprentissage contient un certain nombre d'annotations erronées.

De manière à simuler ces erreurs, nous avons construit de nouvelles matrices d'annotations $\mathbf{Y}^{10\%}$, $\mathbf{Y}^{20\%}$ à partir de la matrice précédente $\mathbf{Y}$, mais contenant de fausses annotations. Nous avons initialisé chaque nouvelle matrice à l'aide de $\mathbf{Y}$, puis avons permuté l'une des annotations non nulles dans chaque $\mathbf{y}_s$ de $\mathbf{Y}^{10\%}$, et deux des annotations non nulles dans chaque $\mathbf{y}_s$ de $\mathbf{Y}^{20\%}$. Etant donné qu'il y a 10 valeurs non nulles dans chaque $\mathbf{y}_s$, permuter une annotation représente un taux d'erreur de 10%, et permuter deux annotations représente un taux d'erreur de 20%.

Nous présentons les ensemble optimisés en figure 9.7, soit en utilisant $\mathbf{Y}^{10\%}$ (a), soit en utilisant $\mathbf{Y}^{20\%}$ (b). A mesure que le taux d'erreur augmente, les vecteurs sont moins bien rassemblés en clusters, et tendent à être éparpillés dans l'espace. Cependant, la méthode est toujours capable de rassembler les vecteurs autant que faire ce peut.

9.4 Conclusion

Nous avons présenté dans ce chapitre une méthode d'apprentissage long terme faiblement supervisé qui repose sur un déplacement des vecteurs et une représentation des catégories par clusters équidistants. Ce cadre de travail nous permet de proposer un algorithme rapide, dont la complexité de la mise à jour est négligeable devant la taille de la base. En dépit de sa simplicité, elle est capable de généraliser au même titre que la méthode basée matrice de Gram, sous réserve que la dimension « utile »(dimension de l'espace engendré par les vecteurs de $\mathbf{X}$) soit supérieure au nombre de catégories.

La méthode repose essentiellement sur l'une des conséquences du théorème 1, qui stipule que le nombre maximal de catégories qui peuvent être apprises suivant le schéma proposé est de $p+1$ dans un espace de dimension p. Cela n'est pas étranger aux remarques que nous avons fait au chapitre 8 à propos du spectre de la matrice de Gram, qui suggérait qu'il est suffisant de disposer de p valeurs propres pour apprendre p catégories. En effet, si nous mettons à part l'approximation, la méthode basée matrice de Gram ne souffre pas de la limitation de la méthode basée vecteurs ; cette méthode va faire grossir la dimension de l'espace « utile » (le rang de la matrice de Gram) autant que nécessaire. Cela est dû au fait que cette méthode travaille dans un espace de dimension infinie. Or, si nous appliquons la méthode basée vecteur dans un espace infini, la limitation disparaît, et le théorème 2 suggère que dans un espace infini, des vecteurs de même norme sont équidistants si et seulement si il sont orthogonaux. Autrement dit, dans un tel espace les deux méthodes vont travailler sur une base orthogonale pour représenter les catégories. Enfin, la méthode basée vecteur regroupera dans un espace infini p catégories dans p clusters, chacun représenté par un centre orthogonal à tous les autres, au même titre que la méthode basée Gram, et la dimension « utile » sera p.

Chapitre 10

Expérimentations

Nous présentons dans cette partie les résultats de simulations avec la méthode basée matrice de Gram (chapitre 8) et la méthode basée vecteur (chapitre 9). Le paramétrage du système RETIN lors de ces expérimentations est le suivant :

- La base d'images utilisée est l'extrait de COREL présentée en annexe A ;
- Les signatures (initiales, avant apprentissage long terme) sont des histogrammes de 25 couleurs et 25 textures, qui ont été réduits par un processus de quantification vectorielle (*cf.* chapitre 2) ;
- La fonction noyau est celle que produit la méthode d'apprentissage long terme ;
- Les SVM sont utilisées pour la classification (*cf.* chapitre 4) ;
- La technique d'apprentissage active est RETIN Méthode 3 (*cf.* chapitre 6).

10.1 Evaluation en fonction de différents scénarios

Les méthodes sont évaluées en suivant différents scénarios. Dans tous les cas, ces scénarios mènent toujours à 9 nouvelles représentations de la base, et diffèrent uniquement par la procédure de mise à jour. Chacune de ces nouvelles représentations est évaluée en suivant le protocole présenté en annexe A.

Pour chacune des optimisations, il est nécessaire de disposer de vecteurs d'annotations $\mathbf{y}$ qui correspondent aux sessions de recherche passées. Dans tous les scénarios, le nombre d'annotations positives dans ces vecteurs est d'au moins 10.

En terme de comparaison, il est difficile de trouver des techniques qui répondent aux contraintes de notre contexte. Par exemple, nous travaillons avec des fonctions noyaux, ce qui élimine toutes les méthodes *ad hoc*. De même, comme nous nous intéressons à des bases où les catégories sont mélangées, certaines méthodes ne fonctionnent pas du tout. C'est le cas, par exemple, des méthodes de déformation globale de l'espace. Nous avons testé les méthodes de (Schultz & Joachims, 2003) et (Xing et al., 2002) en utilisant le code que les auteurs ont publié, mais leur application n'a aucun effet sur la Précision Moyenne.

Enfin, les autres techniques que nous avons étudiées sont beaucoup trop calculatoires, et par conséquent incompatibles avec une utilisation temps réel.

10.1.1 Scénario adaptatif

Nous testons ici les méthodes dans un cas de mise à jour permanente du système. L'idée est de mettre le système à la disposition des utilisateurs, et de prendre immédiatement en compte toute session de recherche. Ainsi, à chaque fois qu'un utilisateur termine sa session, les annotations qu'il a fournies sont immédiatement utilisées pour une mise à jour long terme.

Le protocole que nous avons suivi pour simuler ce scénario combine les mises à jour et une évalution du système toutes les 100 sessions de recherche :

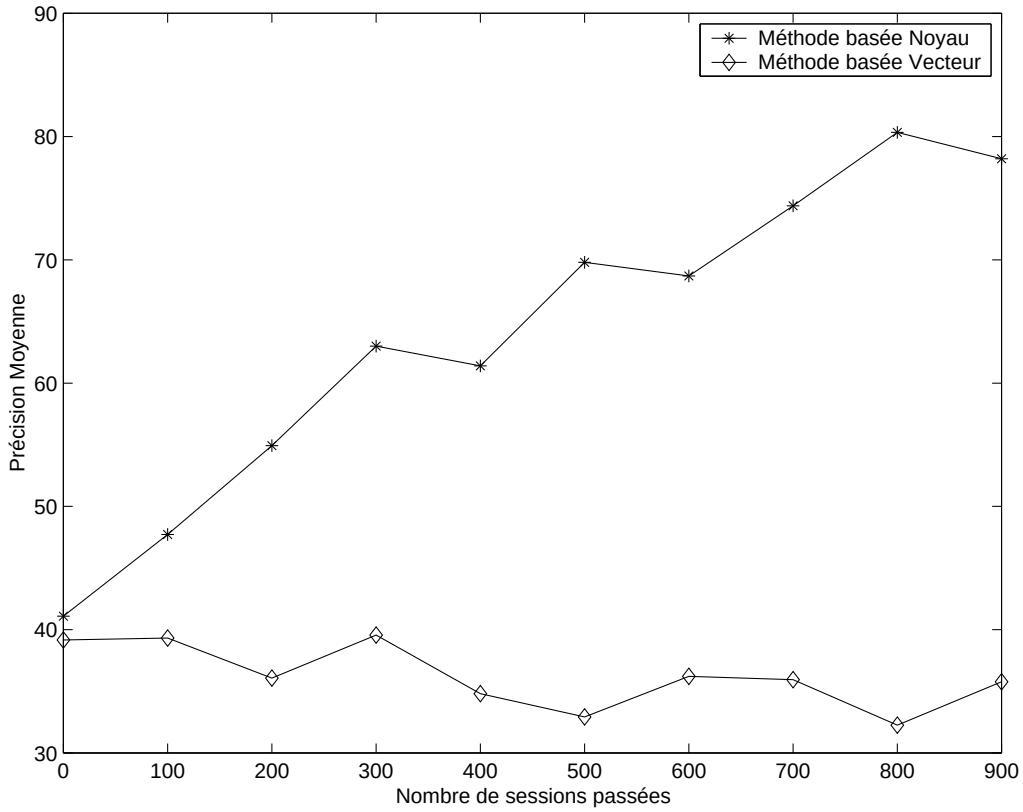
1. Pour m de 1 à 9 :
   (a) On répète 100 fois :
      i. Simulation de l'utilisation du système : on choisit aléatoirement une catégorie, annote dix images qui sont dans cette catégorie, puis annote les dix images proposées par la technique active sur neuf itérations de bouclage ;
      ii. Les annotations à la fin de la session de recherche simulée sont utilisées pour la mise à jour long terme, avec une vigilance ρ de $0,1$ ou $0,01$.
   (b) On mémorise la nouvelle représentation dans X_m.

Cette procédure conduit à 9 nouvelles représentations X_m qui sont évaluées. Les résultats de l'évaluation sont présentés en figure 10.1(a) pour une vigilance élevée ($\rho = 0,1$), et en figure 10.1(b) pour une vigilance faible ($\rho = 0,01$).

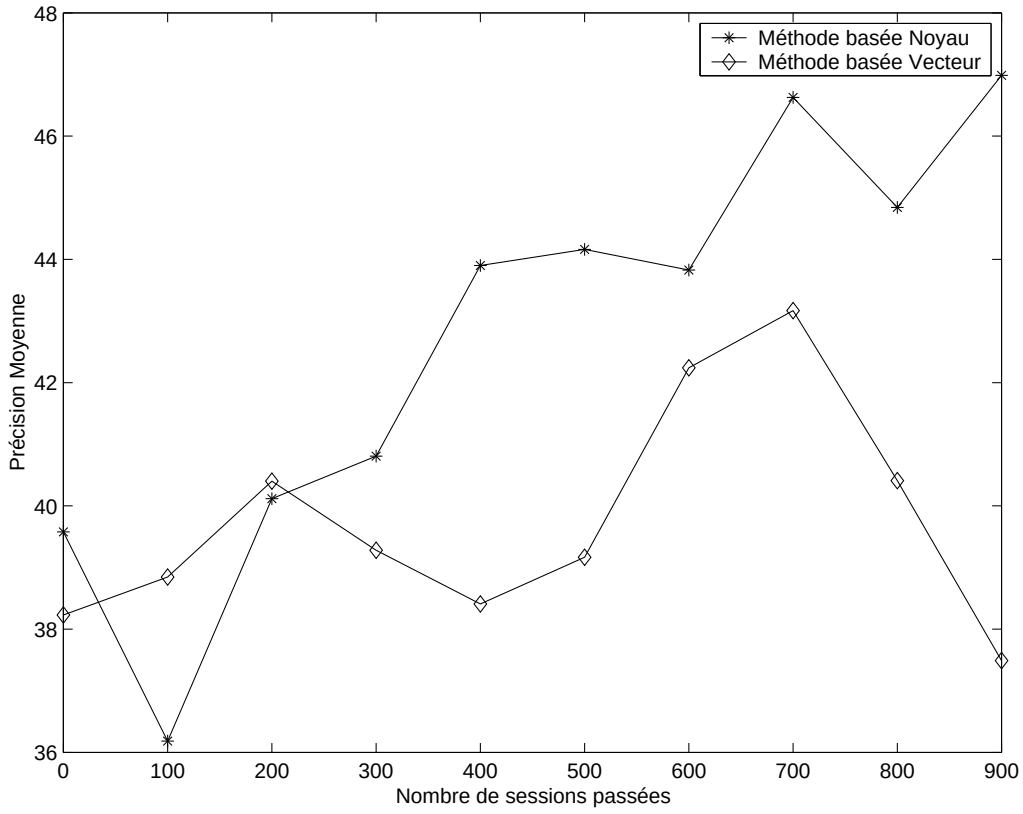
Nous pouvons voir sur ces deux figures que l'évolution de la Précision Moyenne est assez irrégulière. Cela est dû au fait que ce scénario n'exécute pas des mises à jour jusqu'à stabilisation, *i.e.* à la fin de chaque simulation seulement 900 mises à jour ont été appliquées. Alors que la méthode basée Gram améliore les performances du système, la méthode basée vecteur n'est pas efficace pour ce type de scénario.

10.1.2 Scénario en deux temps

Nous évaluons dans ce paragraphe les performances des deux méthodes pour un scénario en deux temps avec des ensembles d'apprentissage synthétiques. Le but est de donner une idée de la capacité des méthodes indépendamment du problème de la collecte des annotations par l'utilisation du système. Nous supposons donc que les ensembles d'apprentissage fabriqués synthétiquement sont issus d'une utilisation idéale du système, où la technique active fournit des lots d'annotations équilibrés, *i.e.* où les annotations positives sont aussi nombreuses que les annotations négatives. Nous présentons aussi à

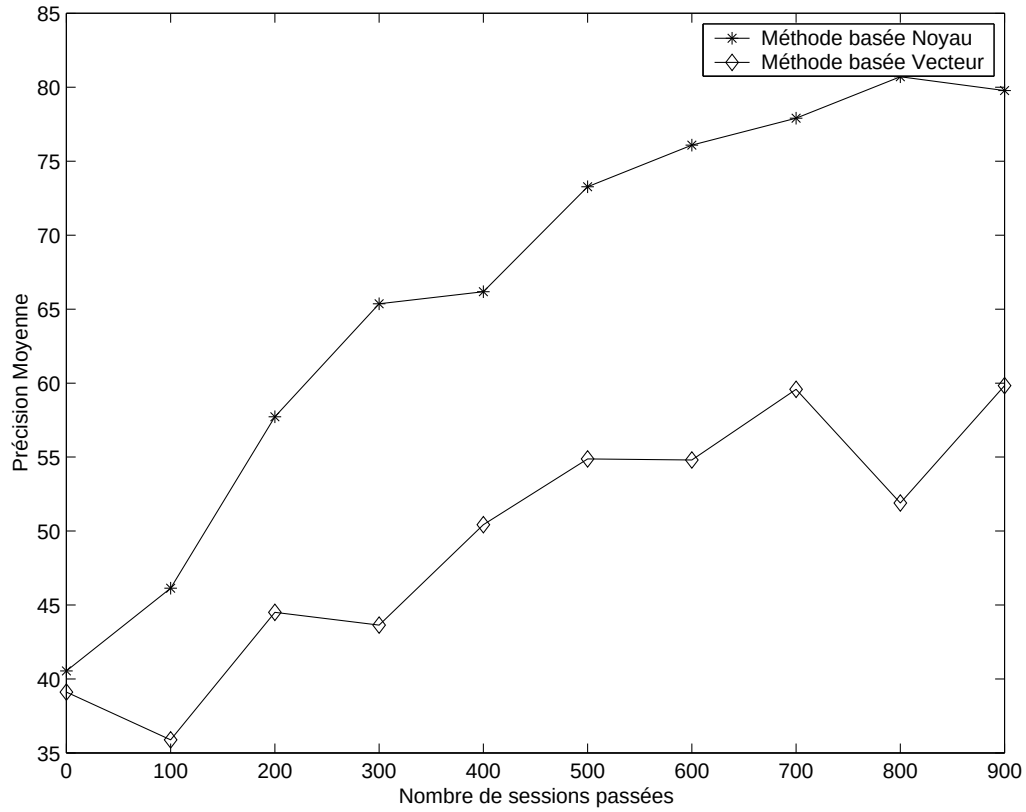


(a) Avec une vigilance élevée ($\rho = 0.1$).

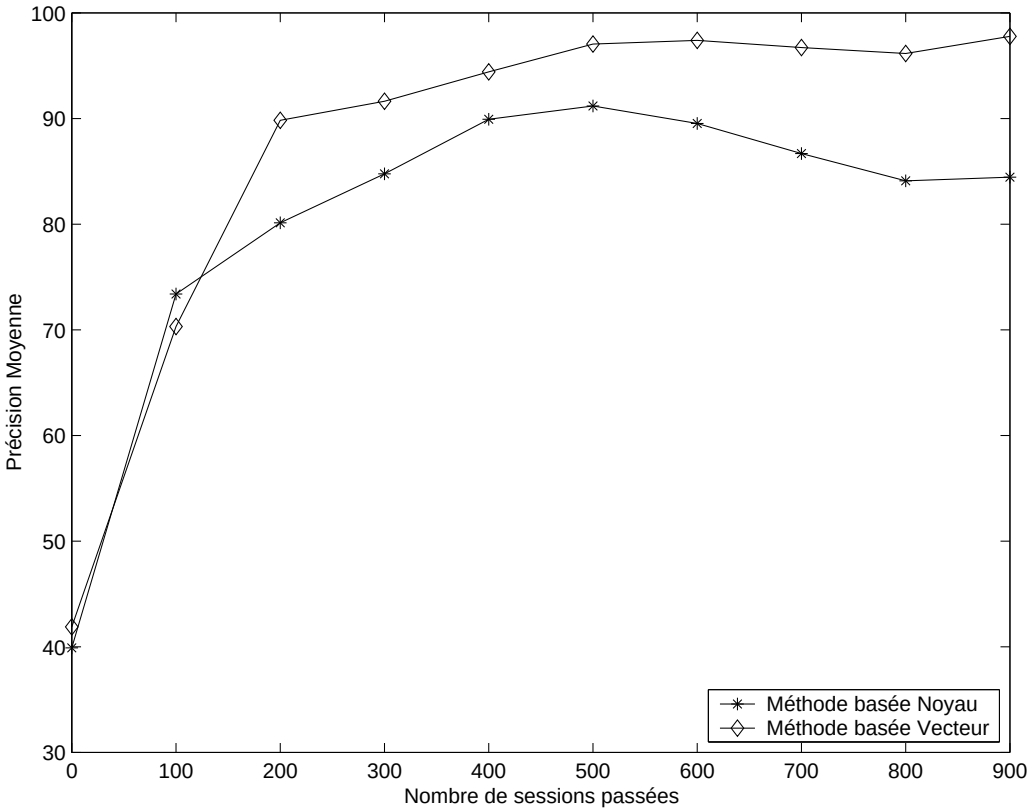


(b) Avec une vigilance faible ($\rho = 0.01$).

FIG. 10.1 – Scénario adaptatif.



(a) Avec des ensembles d'apprentissage non équilibrés.



(b) Avec des ensembles d'apprentissage équilibrés.

FIG. 10.2 – Scénario en deux temps, avec des ensembles d'apprentissage synthétiques.

titre de comparaiton des résultats avec des ensembles d'apprentissage synthétiques non équilibrés.

Le protocole est le suivant, pour m de 1 à 9 :

1. *Fabrication d'un ensemble d'apprentissage synthétique.* On construit une matrice $\mathbf{Y}_m$ de taille $n \times (100m)$. Chaque vecteur colonne $\mathbf{y}$ de ces matrices est généré de la manière suivante :
   (a) On choisit aléatoirement une catégorie c de $\mathbf{C}$;
   (b) On construit aléatoirement un vecteur d'annotation $\mathbf{y}$ pour la catégorie c :
   - Pour le cas non équilibré : on assure qu'il y a au moins 10 annotations positives dans $\mathbf{y}$; pour les 90 autres on donne l'annotation d'une image choisie aléatoirement dans la base.
   - Pour le cas équilibré : on assure qu'il y a 50 annotations positives et 50 annotations négatives dans $\mathbf{y}$.
2. *Optimisation.* On optimise la représentation X de la base en injectant dans un ordre aléatoire les vecteurs de $\mathbf{Y}_m$ avec une vigilance ρ de $0,01$.

Une fois ces calculs terminés, nous avons neuf nouvelles représentations X_m de la base correspondant aux neuf matrices $\mathbf{Y}_m$. Les résultats de l'évaluation de ces représentations sont présentés en figure 10.2(a) pour les ensembles non équilibrés, et en figure 10.2(b) pour les ensembles équilibrés.

Ces premiers résultats mettent en évidence la capacité de deux méthodes. Nous pouvons voir que pour le cas non équilibré, la méthode basée Gram fonctionne mieux que la méthode basée vecteur, et que nous avons un résultat inverse pour le cas équilibré. La méthode basée Gram semble donc mieux gérer des ensembles d'apprentissage de moins bonne qualité, mais ne dispose pas de la capacité à apprendre de la méthode basée vecteur dans un cas favorable.

10.1.3 Scénario intermédiaire

Dans ce paragraphe nous nous intéressons à un scénario intermédiaire entre les deux précédents. L'idée est de mettre à jour au cours de la vie du système, mais par étapes. Le système est utilisé pendant un certain nombre de sessions, et les annotations des utilisateurs sont mémorisées à la fin de chaque session. Puis, la méthode long terme est appliquée avec les annotations mémorisées jusqu'à stabilisation. Ce processus permet une remise en cause progressive du système, tout en offrant une procédure de stabilisation des méthodes.

Nous simulons ce scénario par le protocole suivant, en appliquant les méthodes toutes les 100 sessions de recherche :

1. $Y = \{\}$;

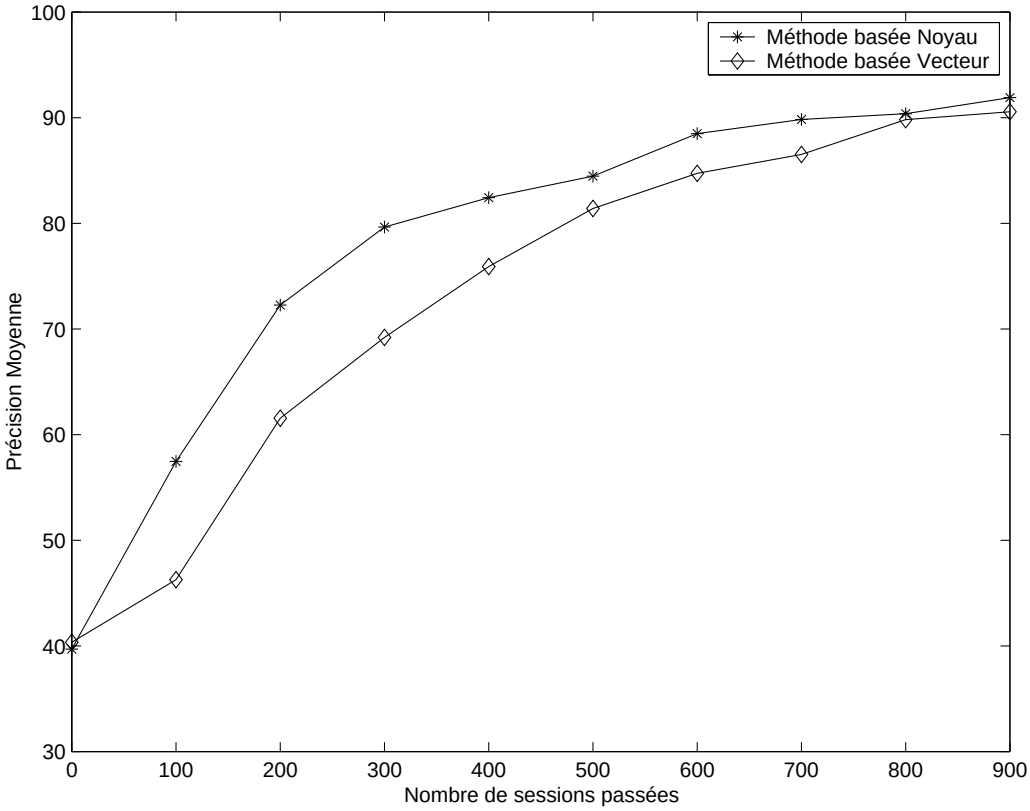


FIG. 10.3 – Scénario par étape, avec des ensembles d'apprentissage construits par simulation du système de recherche

2. Pour m de 1 à 9 :
   (a) On répète 100 fois :
      i. Simulation de l'utilisation du système : on choisit aléatoirement une catégorie, annote 10 images qui sont dans cette catégorie, puis annote les 10 images proposées par la technique active sur 9 itérations de bouclage ;
      ii. On ajoute le vecteur annotation $\mathbf{y}$ à $\mathbf{Y}$.
   (b) On optimise la représentation X de la base en injectant dans un ordre aléatoire les vecteurs de Y avec une vigilance ρ de $0,01$;
   (c) On mémorise la représentation actuelle X dans X_m.

Cette procédure conduit à 9 nouvelles représentations X_m qui sont évaluées. Les résultats de l'évaluation sont présentés en figure 10.3.

Comme nous pouvons le voir sur la figure 10.3, les deux méthodes sont très efficaces. Dans les deux cas, la Précision Moyenne augmente très rapidement. La méthode basée vecteur augmente cependant moins vite que la méthode basée Gram aux premières sessions, mais fini par la rattraper à la fin de la simulation. Ces courbes mettent en évidence le fait que la méthode basée vecteurs a besoin de davantage de données d'apprentissage pour fonctionner.

10.2 Influence de la technique d'apprentissage actif

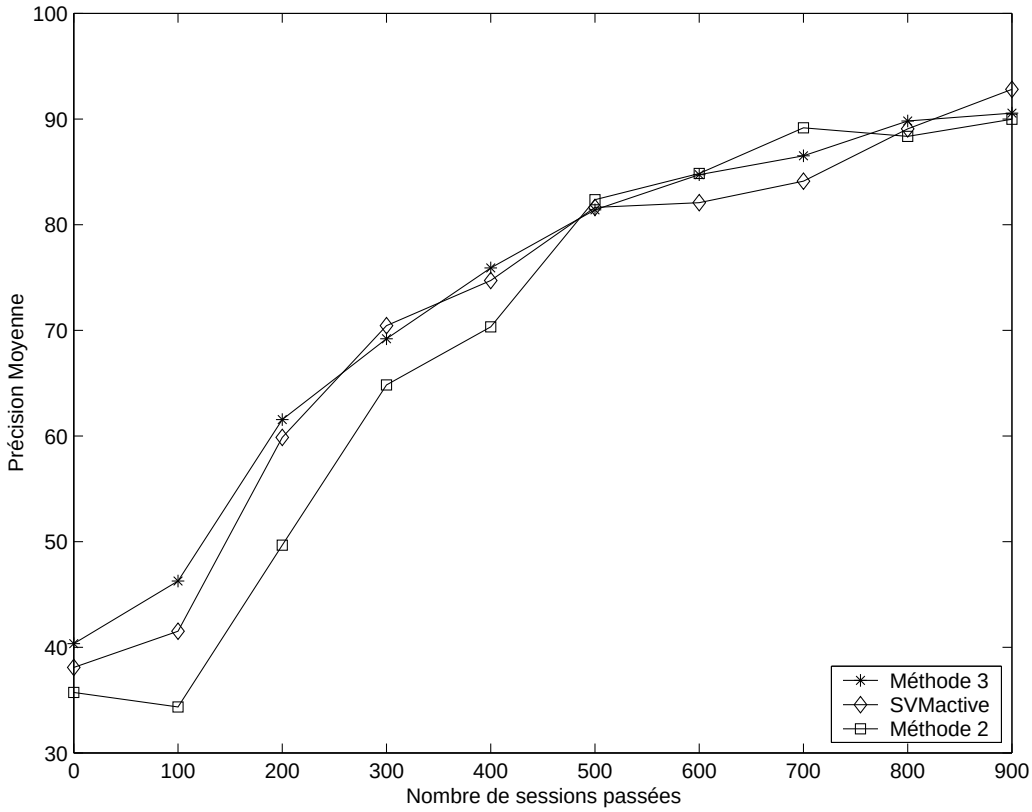
Nous nous intéressons dans ce paragraphe à l'influence de la technique d'apprentissage actif utilisée lors de la recherche interactive. Le scénario suivi est le scénario intermédiaire que nous avons utilisé au paragraphe précédent, mais avec des méthodes de classification active différentes.

Nous avons choisi les trois méthodes suivantes :

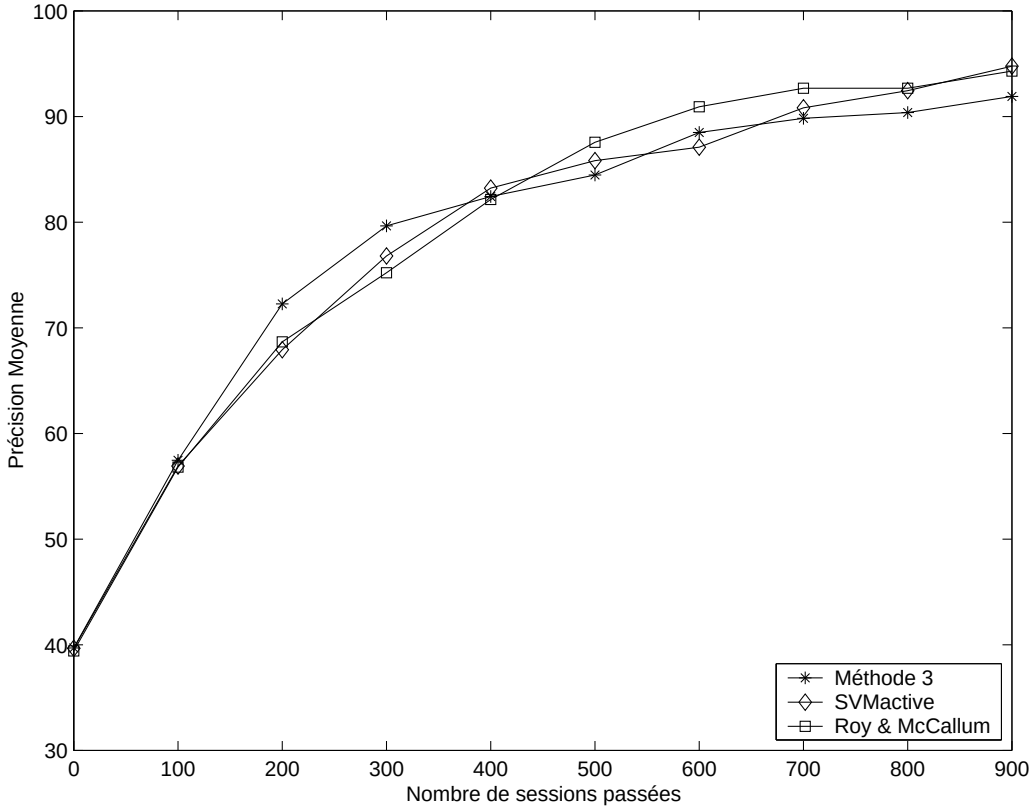
- RETIN Méthode 3 (présentée au §6.6.3) ;
- SVM_{active} (présentée au §5.3.1) ;
- RETIN Méthode 2 (présentée au §6.6.2).

Notons que nous souhaitions comparer la meilleure de nos méthodes actives aux méthodes de référence SVM_{active} et minimiser l'erreur de généralisation. Cependant, compte tenu du coût calculatoire de la méthode de (Roy & McCallum, 2001), nous avons préféré notre deuxième méthode, qui en est une version modifiée mais beaucoup plus rapide.

Les résultats des simulations sont présentés en figure 10.4(a) pour la méthode basée vecteur, et en figure 10.4(b) pour la méthode basée Gram. Globalement, le choix de la technique active n'a pas une grande influence sur l'apprentissage long terme pour le protocole que nous avons suivi. On peut toutefois noter que des différences existent aux premières sessions de recherche avec la méthode basée vecteur. Cette méthode étant plus sensible à la qualité des lots d'annotations, une technique active moins performante semble conduire à un apprentissage plus difficile.



(a) Avec la méthode basée vecteur.



(b) Avec la méthode basée matrice de Gram.

FIG. 10.4 – Influence de la technique d'apprentissage actif.

10.3 Conclusion

Ces expérimentations permettent de mettre en évidence les capacités des deux méthodes proposées.

Tout d'abord, la méthode basée vecteur a plus de difficultés à améliorer la représentation du système dans un cas défavorable, *i.e.* où les lots d'annotations sont peu nombreux, ou bien lorsque les lots d'annotations ont peu d'annotations positives. C'est le cas tout particulièrement lors d'un scénario adaptatif, où une seule mise à jour est appliquée à la fin de chaque session de recherche. Cependant, lorsqu'un scénario intermédiaire est appliqué, c'est à dire lorsque des mises à jour du système sont effectuées de manière périodique (lors des expérimations toutes les 100 sessions de recherche), les deux méthodes ont des performances comparables. Dans ce cas, les deux méthodes sont particulièrement efficaces, puisque la Précision Moyenne augmente de manière significative après quelques centaines de sessions de recherche, et monte à plus de 90% avec 800 lots d'annotations. Ces valeurs de Précision Moyenne sont très élevées, sachant qu'elles ne se lisent pas comme un pourcentage d'images bien catégorisées. Une Précision Moyenne dépassant les 75% constitue déjà un très bon score, et une valeur dépassant les 90% est extrêment élevée : à ce niveau de résultat on peut considérer que les catégories sont entièrement apprises, et les utilisateurs retrouveront très rapidement leurs catégories.

Nous avons aussi évalué l'influence de la technique de recherche interactive utilisée sur l'apprentissage long terme. Comme nous pouvions nous y attendre, ce paramètre influe surtout au début de l'apprentissage long terme, puisque lorsque la représentation est bien optimisée, toutes les techniques actives sont comparables. Ce résultat nous invite à nous interesser à la problématique de la sélection active dans le cadre d'un apprentissage long terme. Hors de ce contexte, la sélection s'effectue uniquement dans le but d'améliorer la session actuelle. Cependant, en prenant en compte l'évolution de la représentation de la base, une technique de sélection pourrait sélectionner les images à faire annoter par les utilisateurs de manière à favoriser l'apprentissage long terme, tout en améliorant la classification actuelle.

Chapitre 11

Conclusion

La recherche d'images par le contenu est une problématique majeure pour la gestion des grandes bases d'images, et gagne en importance avec l'augmentation du nombre d'images numériques. Dans cette thèse, nous nous sommes intéressés à l'application de techniques d'apprentissage dans le cadre de la recherche de catégories d'images dans de grandes bases généralistes.

Afin de pouvoir traiter le caractère généraliste du problème, nous nous sommes tournés vers les techniques d'apprentissage statistique. Actuellement, de nombreux chercheurs de la communauté du traitement de l'image se tournent vers ce type d'apprentissage. L'apprentissage statistique s'intéresse à des problèmes d'apprentissage théoriques et généraux, et la contribution principale de cette thèse est d'apporter une brique supplémentaire dans la construction du pont entre le traitement de l'image et l'apprentissage statistique.

11.1 Bilan

Nous avons tout d'abord traité dans une première partie le problème de la *représentation de la base*. Nous avons utilisé un schéma courant, en considérant un ensemble de signatures pour décrire les images, et une fonction de similarité pour les comparer.

Concernant les signatures, nous avons proposé une technique pour calculer de manière automatique des histogrammes par quantification vectorielle. Nous avons présenté les difficultés qu'engendre la quantification d'un très grand nombre de vecteurs, ainsi que l'importance de la taille des histogrammes. Ainsi, nous avons vu qu'il existe une taille optimale, et qu'augmenter la taille des histogrammes n'apporte pas toujours une amélioration des signatures. De plus, la méthode de quantification pour un très grand nombre de vecteurs que nous avons proposée permet de calculer des signatures de tailles différentes en une seule passe. Ainsi, une fois la méthode appliquée, il est possible de déterminer autant de signatures de tailles différentes que souhaité, pour un temps de calcul négligeable, tout en offrant de très bonnes performances. Cela permet, entre autres, de modifier facilement

la taille des signatures si elle s'avère non optimale, mais aussi de rajouter de nouvelles images dans la base sans avoir à tout recalculer.

Concernant la fonction de similarité, nous avons opté pour une approche par fonctions noyaux. Nous avons présenté cette approche qui permet de formaliser plus facilement les problèmes d'apprentissage, aspects sur lesquels nous nous appuyons pour la recherche interactive et l'apprentissage long terme. Elle permet notamment une analyse plus fine de la représentation de la base. Nous avons mis en évidence les relations qui existent entre le spectre de la matrice de Gram et la capacité d'apprentissage qui en résulte. Bien que l'étude des vecteurs et valeurs propres de la matrice de Gram reste un problème ouvert et encore peu exploré, l'analyse que nous proposons permet de sélectionner des fonctions noyaux adaptées.

Le deuxième point important que nous avons exploré dans la partie II concerne l'apprentissage pour la *recherche interactive*. Nous avons opté pour une approche par classification active. Nous avons mis en évidence les différentes caractéristiques de l'apprentissage interactif pour la recherche d'images, puis avons proposé des solutions pour améliorer les approches existantes.

Nous avons traité le problème de la recherche de catégories par une classification binaire, *i.e.* discriminer entre la classe des images recherchées et les autres images. Nous avons présenté un ensemble de méthodes de classification supervisées et semi-supervisées adaptées à notre contexte d'apprentissage. Toutes ces méthodes ont été appliquées dans le cadre de l'utilisation d'une fonction noyau. Nous avons ensuite implanté et comparé ces différentes méthodes. Dans l'ensemble, les méthodes à noyau ont toutes des performances comparables en terme de Précision Moyenne. Cependant, en considérant d'autres aspects, comme le temps de calcul et la qualité de la classification, il s'est avéré que l'approche par Supports à Vaste Marge est la meilleure méthode pour tous les critères.

Nous avons choisi de traiter le problème de l'interaction par une approche par apprentissage actif. Nous avons présenté cette approche qui permet d'optimiser la sélection des images à faire annoter par l'utilisateur, par conséquent la qualité de l'ensemble d'apprentissage du classifieur. Nous avons mis en évidence l'intérêt de cette approche, dont les performances sont significativement meilleures qu'avec une approche non active.

Nous avons ensuite mis en évidence les différentes limitations des techniques de classification active que nous avons présentées. Dans le contexte de la recherche interactive d'images, l'apprentissage est soumis à un ensemble de contraintes particulières. Dans le but d'améliorer l'approche active, nous avons proposé un ensemble de solutions pour répondre aux différentes contraintes. Parmi ces solutions, citons la correction active de la frontière qui permet d'améliorer la qualité de la sélection dans notre contexte où la taille des classes (les images recherchées *versus* les autres images) sont très différentes. Nous avons aussi proposé une technique pour réduire de manière significative le temps de calcul des techniques de sélection basées sur l'« utilité ». Nous avons montré l'importance du critère à optimiser lors de la sélection, par exemple, en classification active, ce critère est l'erreur de classification. Or dans notre contexte, la Précision Moyenne prévaut, et nous

avons proposé une méthode de sélection pour maximiser ce critère. Enfin, la combinaison de ces différentes solutions nous a permis de proposer une technique d'apprentissage active très efficace, et pour un temps de calcul très faible.

Le dernier point important que nous avons étudié en partie III concerne l'*apprentissage long terme*. Ce type d'apprentissage permet la ré-utilisation des annotations que les utilisateurs fournissent lors des sessions de recherche.

Nous avons présenté le caractère *faiblement supervisé* de l'apprentissage long terme dans notre contexte, où les informations fournies par les utilisateurs sont incomplètes. Notre approche générale s'appuie sur une modification des similarités entre les images dans le cadre de l'utilisation de fonctions noyaux. Ce schéma nous permet d'utiliser les résultats de l'apprentissage long terme pour la recherche interactive.

Suivant ce schéma général, nous avons proposé deux méthodes d'apprentissage long terme. Dans les deux cas, ces méthodes améliorent la similarité entre les images en travaillant sur le produit scalaire. La première méthode modifie directement ces produits scalaires, en optimisant les valeurs de la matrice de Gram, tout en respectant ses propriétés. Notre méthode approxime de manière effective cette matrice, permettant une réduction importante des besoins en mémoire. Cette approximation est justifiée par une discussion autour du rôle des valeurs et vecteurs propres de la matrice de Gram pour l'apprentissage des catégories. De plus, un algorithme rapide permet une mise à jour de même complexité. La deuxième méthode modifie les similarités de manière indirecte, en travaillant dans l'espace des signatures. Elle s'appuie sur un théorème d'équidistance des centres de catégories qui permet un paramètrage efficace. L'algorithme est extrêmenent rapide, puisque la complexité d'une mise à jour est négligeable devant la taille de la base. Les deux méthodes ont été testées sur une base généraliste, et ont démontré leur capacité à améliorer la représentation de la base avec peu d'information supervisée.

Toutes les méthodes que nous avons proposées s'intègrent dans l'architecture du système RETIN 2. C'est un système peu gourmand en ressources machines, puisque toutes nos techniques ont des besoins en mémoire et processeur au plus linéraires devant la taille de la base. Ce système est en mesure de traiter de très grandes bases d'images. Une application sur des bases de photographies de peintures contenant plus de 20,000 images est en cours. Notons aussi que le système RETIN 2 a peu de paramètres à régler, une fois que les fonctions noyaux et les méthodes d'apprentissage ont été choisies. Enfin, notre schéma offre de multiples possibilités d'extensions. Par exemple, l'utilisation d'autres caractérisations visuelles et/ou signatures peuvent profiter de toutes les techniques d'apprentissage, sous réserve que l'on construise une fonction noyau adaptée.

11.2 Perspectives

Le travail de cette thèse se situe à l'interface du traitement des images et de l'apprentissage statistique.

Concernant l'apprentissage interactif, nous avons mis en évidence l'importance du critère à maximiser. Bien que les techniques de classification soient efficaces pour la recherche d'images, elles restent néanmoins basées sur la minimisation de l'erreur de classification. Or, dans notre contexte, c'est la satisfaction de l'utilisateur qui est déterminante. Bien cerner les critères qui modélisent au mieux cette satisfaction (par exemple, la Précision Moyenne) serait un atout pour construire des méthodes d'apprentissage plus adaptées au contexte de la recherche d'images.

Nous avons aussi pu observer l'importance du spectre de la matrice de Gram pour l'apprentissage. En particulier, la répartition des valeurs propres nous semble très importante. Cette problématique est actuellement un thème de recherche très investi par la communauté de l'apprentissage. Dans notre contexte de recherche d'images où nous avons souligné le rôle déterminant du spectre de la matrice de Gram dans nos techniques d'apprentissage, il serait intéressant de caractériser davantage les relations qui existent entre ces espaces propres et la capacité d'apprentissage qui en résulte.

Nous avons travaillé sur des bases fermées, et l'extention des différentes approches à la problématique de l'ajout de nouvelles images dans une base reste à traiter. Ces nouvelles images sont susceptibles de modifier les propriétés de la représentation de la base. Il serait intéressant d'étudier l'impact de ces modifications sur les techniques d'apprentissage. Par exemple, la capacité à généraliser peut être perturbée, et donc entraîner des modifications nécessaires dans le paramétrage du système. De même, l'apprentissage long terme doit pouvoir optimiser la représentation de la base, tout en intégrant ces nouvelles images dépourvues de toute annotation.

Nous nous sommes concentrés sur l'utilisation de signatures vectorielles. Bien que l'approche noyau nous permette d'assurer la pérennité de nos techniques d'apprentissage, un travail de construction de fonctions noyaux reste nécessaire pour tout changement du type de signature. De plus, nous avons utilisé une représentation de la base où chaque image possède une signature. L'apprentissage que nous réalisons est principalement une classification des individus (les images). Il serait encore plus intéressant de propager l'apprentissage à un plus bas niveau, jusqu'à la caractérisation visuelle. Les extracteurs de caractérisques visuelles sont pour la plupart automatisés. L'apprentissage pourrait ne plus exclusivement servir à classifier les individus, mais aussi à apprendre une caractérisation des images. En d'autres termes, l'objectif serait de faire de la caractérisation visuelle supervisée, en s'appuyant sur les informations que fournissent des utilisateurs à un système de recherche.

Enfin, la problématique de l'apprentissage long terme faiblement supervisé présenté

dans cette thèse peut s'étendre à un cadre plus vaste que celui de la recherche de catégories d'images. Cette problématique s'appuie sur un ensemble d'apprentissage particulier. Généralement, les ensembles d'apprentissage sont des étiquettes associées à chaque individu, par exemple « cette image appartient à la catégorie 2 », « cette image n'appartient pas à la catégorie 3 ». Or, les individus n'ont pas d'étiquette dans l'ensemble d'apprentissage que nous utilisons pour le long terme. Cet ensemble est constitué de ce que nous pourrions appeler des « relations ». En recherche d'images, ces « relations » sont des appartenances à une même catégorie, par exemple « ces images appartiennent à la même catégorie, et ces autres images non ». Il est toutefois possible d'imaginer d'autres relations, comme « ces images contiennent le même objet, que ces autres images ne contiennent pas ». Ce type d'information peut être généralisé sous la forme « ces individus ont une propriété commune, que ces autres individus n'ont pas ». L'apprentissage à effectuer consiste alors à retrouver ces relations, ne connaissant ni leur nombre, ni leur taille, ni leur structure.

Annexes

Annexe A

Protocole d'évaluation

Nous présentons dans ce paragraphe les bases, les critères et le protocole d'évaluation que nous utilisons pour évaluer les performances des différentes techniques implantées.

A.1 Bases

Nous utilisons les bases d'images suivantes :

- *Extrait de la base COREL.* La galerie commerciale COREL est composée de photographies couleur regroupées autour d'un grand nombre de thèmes d'une centaine d'images chacun. Cette base contient des photos d'animaux, de lieux, de paysages, d'objets. Les tailles des images (384×256), leurs qualités et luminosités sont homogènes. Nous avons construit un extrait de 6.000 images parmi les 50.000 photographies de la base COREL, en choisissant aléatoirement 77 des thèmes de la base. Nous avons réduit la taille de cette base afin d'avoir des temps d'évaluation résonnables.
- *Base ANN.* Cette base a été construite par l'université de Washington[1] Elle comporte 500 photographies du même type que la base COREL, réparties en 12 catégories distinctes.

Afin de permettre une évaluation sur ces bases, nous avons besoin d'une *vérité terrain.* Cette vérité est l'ensemble des catégories qui sont susceptibles d'être recherchées par des utilisateurs. Pour chaque base, nous avons procédé comme suit :

- *Extrait de la base COREL.* Elle est divisée en de nombreux thèmes comprenant une centaine d'images environ. Afin de pouvoir évaluer le système face au problème des catégories mélangées, nous avons regroupé manuellement 2 ou 3 thèmes pour former 50 catégories. Ainsi, chaque image de la base n'appartient pas exclusivement à une

[1] Annotated groundtruth database, Department of Computer Science and Engineering, University of Washington, www.cs.washington.edu/research/imagedatabase.

catégorie, mais peut appartenir à plusieurs d'entre elles. Plus précisément, chaque image peut appartenir de 1 à 10 catégories à la fois.
– *Base ANN.* Cette base est proposée avec un ensemble de 12 catégories distinctes que nous avons conservé.

A.2 Critères

Evaluer un système de recherche d'images revient à estimer la satisfaction d'un utilisateur recherchant une certaine catégorie pour un classement de la base d'images. Nous sommes dans un cadre où l'appartenance stricte des images à la catégorie recherchée n'est pas le critère prédominant. Par exemple, en reconnaissance de forme un critère souvent utilisée est le taux de bonne catégorisation de chaque image. Or, dans notre contexte, nous nous intéressons à un utilisateur recherchant une seule catégorie. Lors de cette évaluation, les autres catégories n'entrent pas en compte. De plus, le classement des images a une importance. Il est en effet très intéressant pour l'utilisateur de retrouver des images parmi les images les plus pertinentes.

L'un des plus utilisés est la *Précision* et le *Rappel*. Si nous notons par A l'ensemble des images appartenant à la catégorie, et par $B(i)$ l'ensemble des i images les plus pertinentes selon le système pour le classement actuel, alors :

$$\begin{aligned} \text{Précision} &= \frac{|A \cap B(i)|}{i} \\ \text{Rappel} &= \frac{|A \cap B(i)|}{|A|} \end{aligned}$$

Pour un classement de la base par le système, on peut ainsi calculer autant de couples de Précision/Rappel que de nombre d'images que l'utilisateur souhaite retrouver, de $i = 1$ à au nombre $i = n$ d'images dans la base. L'ensemble de ces points permet de tracer une courbe de Précision/Rappel, qui permet d'évaluer la qualité du classement actuel de la base.

Notons qu'un critère dérivé de la Précision est le *Top*. On s'intéresse alors au nombre d'images retrouvées parmi les premières renvoyées par le système. Par exemple, pour le Top 100 est le nombre d'images retrouvées parmi les 100 premières du classement, ce qui est équivalent à une Précision pour $i = 100$ images demandées par l'utilisateur.

Les courbes de Précision/Rappel permettent d'évaluer les performances du système sur une seule catégorie, mais sont plus difficiles à lire si l'on souhaite avoir une mesure scalaire de la performance. Pour ce faire, nous utilisons la *Précision Moyenne* (*Mean Average Precision*[2]) qui est définie par l'intégrale de la courbre de Précision/Rappel.

[2] *cf.* . TREC VIDEO conference : http ://www-nlpir.nist.gov/projects/trecvid/

A.3 Protocole

De nombreuses évaluations sont présentées tout au long de ce mémoire. Sauf mention contraire, le protocole expérimental pour l'evaluation des performances globales du système est le suivant :

Chaque évaluation est effectuée pour 1.000 simulations de sessions de recherche. Pour chacune des simulations, une valeur de Précision Moyenne est calculée, et la mesure finale est la moyenne de ces Précisions Moyennes sur les 1.000 simulations.

Chaque session de recherche est simulée à l'aide d'un utilisateur virtuel ou robot. Le robot est conçu pour agir comme un utilisateur réel. Il commence par choisir aléatoirement une des catégories de la base considérée. Par exemple, pour l'extrait de COREL, une des 50 catégories que nous avons construites. Puis nous supposons que le robot apporte une image personnelle qui n'est pas dans la base. Le système annote alors l'image de la base la plus proche. Afin de simuler cette initialisation, une image de la catégorie est choisie aléatoirement. Ainsi, l'ensemble d'apprentissage débute avec une annotation positive. Puis le système sélectionne 10 images non annotées dans la base, en fonction de la technique d'apprentissage utilisée. Le robot annote alors ces 10 images en fonction de leur appartenance à la catégorie. Le processus de sélection/annotation est répété 10 fois, ce qui nous donne un ensemble d'apprentissage final avec 101 exemples, soit $1,68\%$ de la taille de la base pour la base COREL. A l'aide de cet ensemble d'apprentissage, le système calcule un classement des images par pertinence, et en déduit la Précision Moyenne pour cette session de recherche.

Ce protocole experimental se résume de la facon suivante : 1 image initiale, 10 bouclages, 10 annotations par bouclage.

Bibliographie

Aksoy, S., & Haralick, R. (2001). Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern Recognition Letters*, *22*, 563–582.

Aksoy, S., Haralick, R., Cheikh, F., & Gabbouj, M. (2000). A weighted distance approach to relevance feedback. *IAPR International Conference on Pattern Recognition* (pp. 812–815). Barcelona, Spain.

Amari, S., & Wu, S. (1999). Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6) :783–789.

ANN. Annotated groundtruth database (ann). Department of Computer Science and Engineering, University of Washington, www.cs.washington.edu/research/imagedatabase/.

Argamon-Engelson, S., & Dagan, I. (1999). Commitee-based sample selection for probabilistic classifiers. *Journal of Artificial Intelligence Research*, 11 :335–360.

Berg, C., Christensen, J. P. R., & Ressel, P. (1984). Hamonic analisys on semigroups. *Springer-Verlag*.

Berrani, S.-A., Amsaleg, L., & Gros, P. (2003). Recherche approximative de plus proches voisins : application la reconnaissance d'images par descripteurs locaux. *Technique et Science Informatiques*, 22(9) :1201–1230.

Boser, B. E., Guyon, I., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Workshop on Computational Learning Theory* (pp. 144–152).

Brinker, K. (2003). Incorporating diversity in active learning with support vector machines. *International Conference on Machine Learning* (pp. 59–66).

Brunelli, R., & Mich, O. (2001). Histograms analysis for image retrieval. *Pattern Recognition*, *34*, 1625–1637.

Caenen, G., Frederix, G., Kuijk, A., Pauwels, E., & Schouten, B. (2000). Show me what you mean ! PARISS : A CBIR-interface that learns by example. *International Conference on Visual Information Systems (Visual'2000)* (pp. 257–258).

Carrilero, A.-C. (1999). *Les espaces de représentation de la couleur* (Technical Report 99D006). ENST, Paris.

Chang, E., Li, B. T., Wu, G., & Goh, K. (2003). Statistical learning for effective visual information retrieval. *IEEE International Conference on Image Processing*. Barcelona, Spain.

Chapelle, O., Haffner, P., & Vapnik, V. (1999). Svms for histogram based image classification. *IEEE Transactions on Neural Networks*, *9*.

Chen, Y., Zhou, X., & Huang, T. (2001). One-class svm for learning in image retrieval. *International Conference in Image Processing (ICIP)* (pp. 34–37). Thessaloniki, Greece.

Chua, J. (2000). Fast full-search equivalent nearest-neighbour search algorithms. Master's thesis, School of Computer Science and Software Engineering, Monash University, Australia 3168.

Cohn, D. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*, *4*, 129–145.

Cohn, D., Atlas, L., & Ladner, R. (1994). Improving generalization with active learning. *Machine Learning*, 15(2) :201–221.

Cord, M., Fournier, J., Gosselin, P. H., & Philipp-Foliguet, S. (2005a). Interactive exploration for image retrieval. *EURASIP Journal on Applied Signal Processing*.

Cord, M., Gosselin, P. H., & Philipp-Foliguet, S. (2005b). Stochastic exploration and active learning for image retrieval. *Image and Vision Computing*.

Cox, I., Miller, M., Minka, T., Papathomas, T., & Yianilos, P. (2000). The bayesian image retrieval system, PicHunter : Theory, implementation and psychophysical experiments. *IEEE Transactions on Image Processing*, *9*, 20–37.

Cristianini, N., Kandola, J., Elisseff, A., & Shawe-Taylor, J. (2006). On optimizing kernel alignment. *Journal of Machine Learning Research*.

Cristianini, N., Lodhi, H., & Shawe-Taylor, J. (2002). Latent semantic kernels. *Journal of Intelligent Information Systems*.

Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines. *Cambridge University Press*.

Cristianini, N., Shawe-Taylor, J., Elisseff, A., & Kandola, J. (2001). On kernel target alignement. *Neural Information Processing Systems*. Vancouver, Canada.

Doulamis, N., & Doulamis, A. (2001). A recursive optimal relevance feedback scheme for cbir. *International Conference in Image Processing (ICIP'01)*. Thessaloniki, Greece.

Eichhorn, J., & Chapelle, O. (2004). *Object categorization with svm : kernels for local features* (Technical Report).

Engelbrecht, A. P., & Brits, R. (2002). Supervised training using an unsupervised approach to active learning. *Neural Processing Letters*, 15(3) :247–260.

Fournier, J. (2002). *Content based image indexing and interactive retrieval*. Doctoral dissertation, UCP, Paris, France. Written in French.

Fournier, J., & Cord, M. (2002). Long-term similarity learning in content-based image retrieval. *International Conference in Image Processing (ICIP)*. Rochester, New-York, USA.

Fournier, J., Cord, M., & Philipp-Foliguet, S. (2001a). Back-propagation algorithm for relevance feedback in image retrieval. *International Conference in Image Processing (ICIP'01)* (pp. 686–689). Thessaloniki, Greece.

Fournier, J., Cord, M., & Philipp-Foliguet, S. (2001b). Retin : A content-based image indexing and retrieval system. *Pattern Analysis and Applications Journal, Special issue on image indexation*, *4*, 153–173.

Fritzke, B. (1997). The lbg-u method for vector quantization - an improvement over lbg inspired from neural netword. *Neural Processing Letters*, *5*, 35–45.

Gabor, D. (1946). Theory of communication. *The Journal of the Institute of Electrical Engineers*, *93*, 429–457.

Gagalowicz, A. (1983). *Vers un modèle de textures*. Thse d'etat, Universit Pierre et Marie Curie, Paris VI.

Geman, D., & Moquet, R. (2000). A stochastic feedback model for image retrieval. *RFIA'2000* (pp. 173–180). Paris, France.

Gersho, A. (1979). Asymptotically optimal block quantization. *IEEE Transaction Information Theory*, *IT-25*, 373–380.

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations*. The John Hopkins University Press.

Gosselin, P. H., & Cord, M. (2004a). A comparison of active classification methods for content-based image retrieval. *International Workshop on Computer Vision meets Databases (CVDB), ACM Sigmod*. Paris, France.

Gosselin, P. H., & Cord, M. (2004b). RETIN AL : An active learning strategy for image category retrieval. *IEEE International Conference on Image Processing*. Singapore.

Gosselin, P. H., & Cord, M. (2004c). Semantic kernel updating for content-based image retrieval. *IEEE International Workshop on Multimedia Content-based Analysis and Retrieval*. Miami, Florida, USA.

Gosselin, P. H., & Cord, M. (2005a). Active learning techniques for user interactive systems : application to image retrieval. *International Workshop on Machine Learning for MultiMedia (In conjonction with ICML).*

Gosselin, P. H., & Cord, M. (2005b). Méthodes d'apprentissage smatiques : application la recherche d'images. *Conférence sur l'Apprentissage.* Nice, France.

Gosselin, P. H., & Cord, M. (2005c). Semantic kernel learning for interactive image retrieval. *IEEE International Conference on Image Processing.* Genova, Italy.

Gosselin, P. H., & Cord, M. (2006). Feature based approach to semi-supervised similarity learning. *Pattern Recognition, Special Issue on Similarity-Based Pattern Recognition.*

Gosselin, P. H., Najjar, M., Cord, M., & Ambroise, C. (2004a). Discriminative classification *vs* modeling methods in CBIR. *IEEE Advanced Concepts for Intelligent Vision Systems (ACIVS).* Brussel, Belgium.

Gosselin, P. H., Najjar, M., Cord, M., Ambroise, C., & Philipp-Foliguet, S. (2004b). Mthodes d'apprentissage pour la recherche d'images par le contenu. *Proceedings of Reconnaissance des Formes et Intelligence Artificielle (RFIA).* Toulouse, France.

Grira, N., Crucianu, M., & Boujemaa, N. (2005). Semi-supervised image database categorization using pairwise constraints. *IEEE International Conference on Image Processing.* Genova, Italy.

Gros, P. (1998). *De l'appariement à l'indexation des images.*

Han, J., Li, M., Zhang, H., & Guo, L. (2003). A memorization learning model for image retrieval. *IEEE International Conference on Image Processing.* Barcelona, Spain.

Hasenjager, M., & Ritter, H. (1996). Active learning of the generalized high-low game. *International Conference on Artificial Neural Networks* (pp. 501–506). Springer-Verlag.

Hastie, T., Tibshirani, R., & Friedman, J. (2001a). *The element of statistical learning.* Springer.

Hastie, T., Tibshirani, R., & Friedman, J. (2001b). *Elements of statistical learning : Data mining, inference and prediction.* Springer-Verlag, New York.

Heinrichs, A., Koubaroulis, D., Levienaise-Obadia, B., Rovida, P., & Jolion, J. (2000). Image indexing and content based search using pre-attentive similarities. *RIAO2000* (pp. 1616–1631).

Heisterkamp, D. R. (2002). Building a latent semantic index of an image database from patterns of relevance feedback. *International Conference on Pattern Recognition* (pp. (4) :132–137). Quebec City, Canada.

Heisterkamp, D. R., Peng, J., & Dai, H. K. (2001). Adaptive quasiconformal kernel metric for image retrieval. *International Conference on Computer Vision and Pattern Recognition*.

Huang, T., & Zhou, X. (2001). Image retrieval with relevance feedback : From heuristic weight adjustment to optimal learning methods. *International Conference in Image Processing (ICIP'01)* (pp. 2–5). Thessaloniki, Greece.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. *Proc. 16th International Conference on Machine Learning* (pp. 200–209). Morgan Kaufmann, San Francisco, CA.

Jurie, F., & Triggs, B. (2005). Creating efficient codebooks for visual recognition. *International Conference on Computer Vision and Pattern Recognition*.

Krogh, A., & Vedelsby, J. (1995). Neural network ensembles, cross validation, and active learning. *Advances in Neural Information Processing Systems* (pp. 7 :231–238).

Kullback, S. (1959). *Information theory and statistics*. Wiley (New York).

Lanckriet, G. R. G., Cristianini, N., Bartlett, N., El Ghaoui, L., & Jordan, M. I. (2002). Learning the kernel matrix with semi-definite programming. *International Conference on Machine Learning*. Sydney, Australia.

Lewis, D. D., & Catlett, J. (1994). Heterogeneous uncertainly sampling for supervised learning. *International Conference on Machine Learning* (pp. 148–56). W.W. Cohen and H. Hirsh, editors.

Lewis, D. D., & Gale, W. A. (1994). A sequential algorithm for training text classifiers. *International ACM-SIGIR Conference on Research and Developement in Information Retrieval* (pp. 3–12). W.B. Croft and C.J. Van Rijsbergen, editors.

Liere, R., & Tadepalli, P. (1997). Active learning with committees for text categorization. *National Conference on Artificial Intelligence* (pp. 591–596).

Linde, Y., Buzo, A., & Gray, R. (1980). An algorithm for vector quantizer design. *IEEE Transaction on Communication*, *28*, 84–94.

Lindenbaum, M., Markovitch, S., & Rusakov, D. (2004). Selective sampling for nearest neighbor classifiers. *Machine Learning*, 54(2) :125–152.

Liu, F., & Picard, R. W. (1996). Periodicity, directionality, and randomness : Wold features for image modeling nad retrieval. *IEEE Trans. Pattern Anal. Machine Intell.*, *18*, 722–733.

Ma, W., & Manjunath, B. (1995). Image indexing using a texture dictionnary. *SPIE Conference on Image Storage and Archiving System* (pp. 288–298). Philadelphia, Pennsylvania.

Mahalanobis, P. (1930). On tests and measures of groups divergence. *Journal of the Asiatic Society of Benagal*.

Manjunath, B., & Ma, W. (1996). Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence (Special Issue on Digital Libraries)*, *18*, 837–842.

Mika, S., Rätsch, G., Weston, J., Schölkopf, B., & Müller, K.-R. (1999). Fisher discriminant analysis with kernels. *Neural Networks for Signal Processing IX* (pp. 41–48). IEEE.

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., & Gool, L. V. (2004). A comparison of affine region detectors. *Submitted to International Journal of Computer Vision*. Submitted in August 2004.

Müller, H., Müller, W., Squire, D. M., Marchand-Maillet, S., & Pun, T. (2000). Learning feature weights from user behavior in content-based image retrieval. *MDM/KDD2000 Workshop on Multimedia Data Mining in conjunction with the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Boston, USA.

Müller, W., Squire, D. M., Müller, H., & Pun, T. (1999). *Hunting moving targets : an extension to bayesian methods in multimedia databases*Technical report 99.03). Computer Vision Group, Computing Science Center, University of Geneva, Genve, Switzerland.

Najjar, N., Cocquerez, J., & Ambroise, C. (2003). Feature selection for semi supervised learning applied to image retrieval. *IEEE ICIP*. Barcelena, Spain.

Nguyen, H. T., & Smeulders, A. (2003). Active learning using pre-clustering. *International Conference on Machine Learning*.

Park, J. M. (2000). Online learning by active sampling using orthogonal decision support vectors. *IEEE Workshop on Neural Networks for Signal Processing*.

Patanè, G. (2001). *Unsupervised learning on traditional and distributed systems*. Doctoral dissertation, Palermo.

Peng, J., Bhanu, B., & Qing, S. (1999). Probabilistic feature relevance learning for content-based image retrieval. *Computer Vision and Image Understanding*, *75*, 150–164.

Puzicha, J., Hofmann, T., & Buhmann, J. (1997). Non-parametric similarity measures for unsupervised texture segmentation and image retrieval. *IEEE Conference on Computer Vision and Pattern Recognition* (pp. 267–272).

Roy, N., & McCallum, A. (2001). Toward optimal active learning through sampling estimation of error reduction. *International Conference on Machine Learning*.

Rubner, Y. (1999). *Perceptual metrics for image database navigation*. Doctoral dissertation, Stanford University.

Rui, Y., & Huang, T. (2000). Optimizing learning in image retrieval. *Conf on Computer Vision and Pattern Recognition (CVPR)* (pp. 236–243). Hilton Head, SC.

Rui, Y., Huang, T., Mehrotra, S., & Ortega, M. (1997). A relevance feedback architecture for content-based multimedia information retrieval systems. *IEEE Workshop on Content-Based Access of Image and Video Libraries* (pp. 92–89).

Sahbi, H. (2003). Machnies a vecteur de support pour une dtection hirarchiques des visages.

Santini, S., Gupta, A., & Jain, R. (2001). Emergent semantics through interaction in image databases. *IEEE Transactions on Knowledge and Data Engineering*, *13*, 337–351.

Saux, B. L. (2003). *Classification non exclusive et personnalisation par apprentissage : Application à la navigation dans les bases d'images*. Doctoral dissertation, INRIA.

Schmid, C. (2004). Weakly supervised learning of visual models and its application to content-based retrieval. *International Journal of Computer Vision*, *56*, 7–16.

Schmid, C., & Mohr, R. (1995). *Matching by local invariants*Technical report RR-2644). INRIA.

Schölkopf, B., & Smola, A. (2002). *Learning with kernels*. MIT Press, Cambridge, MA.

Schölkopf, B., Williamson, R., Smola, A., Shawe-Taylor, J., & Platt, J. (2000). Support vector method for novelty detection. *Advances in Neural Information Processing Systems*.

Scholkpof, B., Shawe-Taylor, Smola, A., & Williamson, R. (1999). *Generalization bounds via eigenvalues of the gram matrix* (Technical Report). NeuroColt.

Schultz, M., & Joachims, T. (2003). Learning a distance metric from relative comparisons. *Neural Information Processing Systems*.

Schurmann, J. (1996). *Pattern classification : a unified view of statistical and neural approaches*. Wiley, New York.

Schlkopf, B., Smola, A., & Mller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, *10*, 1299–1319.

Sethi, I., & Patel, N. (1995). Statistical approach to scene change detection. *Storage and Retrieval for Image and Video Databases* (pp. 329–338).

Shawe-Taylor, J., Williams, C., Cristianini, N., & Kandola, J. (2002). On the eigenspectrum of the gram matrix and its reliationship to the operator eigenspectrum. *ALT* (pp. 23–40).

Smeulders, A., Worring, M., Santini, S., Gupta, A., & Jain, R. (2000). Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*, 1349–1380.

Sontag, E. (1998). VC dimension of neural networks. *Neural Networks and Machine Learning (C.M. Bishop, ed.)*, 69–94.

Stoica, R., Zerubia, J., & Francos, J. M. (1998). *Indexation et recherche dans une base de données multimedia grâce á une modélisation paramétrique de texture utilisant la décomposition de wold 2d*RR 3594). INRIA.

Stricker, M., & Orengo, M. (1995). Similarity of color images. *SPIE, Storage and Retrieval for Image Video Databases III* (pp. 381–392).

Suykens, J. A. K., Van Gestel, T., De Brabanter, J., & De Moor, B. (2002). Least square support vector machines. *World Scientific Publishing Co., Pte, Ltd., Singapore*.

Swain, M., & Ballard, D. (1991). Color indexing. *International Journal of Computer Vision*, *7*, 11–32.

Tong, S. (2001). *Active learning : Theory and applications*. Doctoral dissertation, Stanford University.

Tong, S., & Chang, E. (2001). Support vector machine active learning for image retrieval. *ACM Multimedia*.

Tong, S., & Koller, D. (2000). Support vector machine active learning with applications to text classification. *International Conference on Machine Learning*, 999–1006.

Tong, S., & Koller, D. (2001). Support vector machine active learning with application to text classification. *Journal of Machine Learning Research*, 2 :45–66.

Vapnik, V. N. (1999). *Statistical learning theory*. Wiley-Interscience, New York.

Vasconcelos, N. (2000). *Bayesian models for visual information retrieval*. Doctoral dissertation, Massachusetts Institute of Technology.

Vasconcelos, N., & Kunt, M. (2001). Content-based retrieval from image databases : current solutions and future directions. *International Conference in Image Processing (ICIP'01)* (pp. 6–9). Thessaloniki, Greece.

Veropoulos, K. (1999). Controlling the sensivity of support vector machines. *International Joint Conference on Artificial Intelligence (IJCAI)*. Stockholm, Sweden.

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russell, S. (2002). Distance metric learning, with application to clustering with side-information. *Neural Information Processing Systems*. Vancouver, Bristish Columbia.

Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. *International Conference on Machine Learning*.