Fast learning methods adapted to the user specificities: application to earth observation image information mining

Pierre Blanchart

École Doctorale d'Informatique,
Télécommunications et Électronique de Paris

PhD Thesis

defended on September 26th 2011

at Télécom ParisTech

Specialization : Signal and Images

Pierre Blanchart

Fast learning methods adapted to the user specificities: application to earth observation image information mining

Examination committee

Chi-Ren SHYU	Reviewers
Constantin VERTAN	
Isabelle BLOCH	Examiners
Arnold SMEULDERS	
Alain GIROS	
Mihai DATCU	Supervisor
Marin FERECATU	Co-supervisor

Summary

An important emerging topic in satellite image content extraction and classification is building retrieval systems that automatically learn high-level semantic interpretations from images, possibly under the direct supervision of the user. Indeed, because of the increased resolution of sensors, the content of satellite images has diversified enormously: it is not uncommon to see details such as cars, streets, technological artefacts, and sometimes people. Thus, retrieval techniques developed initially for multimedia databases are becoming increasingly more relevant to mining data from Earth Observation repositories. The goal of these techniques is to discover new and unexpected patterns, trends, and relationships embedded within large and diverse geographic data sets. Manual annotation is sometimes employed, but is extremely expensive and often subjective. Instead, systems which allow to perform visual content mining from large-scale image databases as well as indexing of high-dimensional database for fast relevant imagery retrieval have been proposed over the past few years. The main contributions of this work are inscribed in the direct continuation of the ideas developed in these systems. We envisage successively the two very broad categories of auto-annotation systems and interactive image search engine to propose our own solutions to the recurring problem of learning from small and non-exhaustive training datasets and of generalizing over a very high-volume of unlabeled data. We first look into the problem of exploiting the huge volume of unlabeled data to discover "unknown" semantic structures, that is, semantic classes which are not represented in the training dataset. We among others propose a semi-supervised algorithm able to build an auto-annotation model over non-exhaustive training datasets and to point out to the user new interesting semantic structures in the purpose of guiding him in his database exploration task. In our second contribution, we envisage the problem of speeding up the learning in interactive image search engines. Minimizing the number of iterations in the relevance feedback loop is indeed a crucial issue to build systems which are well-adapted to a human user. With this purpose in mind, we derive a semi-supervised active learning algorithm which exploits the intrinsic data distribution to achieve faster identification of the target category. Our last contribution deals with the problem of retrieving objects in large satellite image scenes. We describe an active learning algorithm which relies on a coarse-to-fine strategy to handle large volumes of data while keeping a satisfying level of accuracy. The proposed algorithm leads to a reduction by more than two orders of magnitude in the number computations necessary at each active learning iteration in standard state-of-the-art interactive image retrieval tools which do not allow to search for complex classes/objects in a really interactive way because of the computational overload inherent to multiple evaluations of the decision function of complex classifiers. We assess each time our results on Spot5 and QuickBird panchromatic imagery and we show that the methods we propose significantly outperform state-of-the-art techniques while adding interesting new features such as the "unknown" semantic structures discovery feature in the auto-annotation case or the interactive search scheme in the object retrieval part.

Contents

Chapter 1

Résumé français

1.1 Introduction

Cette thèse débutée en octobre 2008 a été effectuée sous la direction de Mihai Datcu et la co-supervision Marin Ferecatu au sein du centre de compétence pour l'extraction d'informations d'images de télédétection (Competence Center on Information and Image Understanding (COC)).

1.2 Contexte

Le traitement des images de télédétection a reçu ces dernières années une attention accrue du fait notamment de l'augmentation du nombre et de la résolution des instruments d'observation de la Terre. Avec l'apparition de capteurs de résolution métriques, les contenus informationnels des images se diversifient énormément et l'accroissement des capacités d'archivage permet de stocker de plus en plus d'images. Dans ce contexte, la nécessité de recourir à des systèmes d'indexation automatique d'images se fait nettement sentir car il n'est plus possible d'analyser toutes les images manuellement. Il existe donc un important travail de recherche dans lequel s'inscrit cette thèse pour développer des outils capables de remplacer un opérateur humain au niveau des tÃćches d'analyse et d'indexation des bases d'images satellitaires.

1.3 Objectifs

Le but des systèmes de recherche d'images est de diriger rapidement l'utilisateur vers des contenus qui sont pertinents par rapport à la requête qu'il a formulée. Après une présentation de la problématique et un état d'art du domaine, cette thèse présente nos contributions dans le cadre de l'apprentissage avec très peu d'exemples qui est propre à l'imagerie satellitaire. Ces contributions se situent principalement autour de l'utilisation de méthodes semi-supervisées pour exploiter l'information contenue dans les données non-labellisées et pallier en quelque sorte la faiblesse et la non-exhaustivité des bases d'apprentissage. Nous présentons deux scénarios d'utilisation de méthodes semi-supervisées. Le premier se place dans le cadre d'un système d'annotation
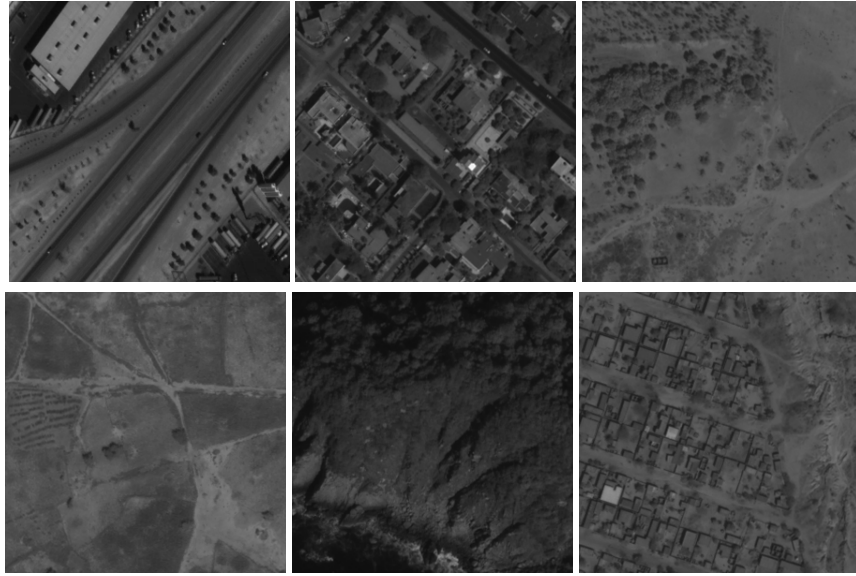
Figure 1.1:   Images panchromatiques QuickBird à une résolution de 61cm
©DigitalGlobe
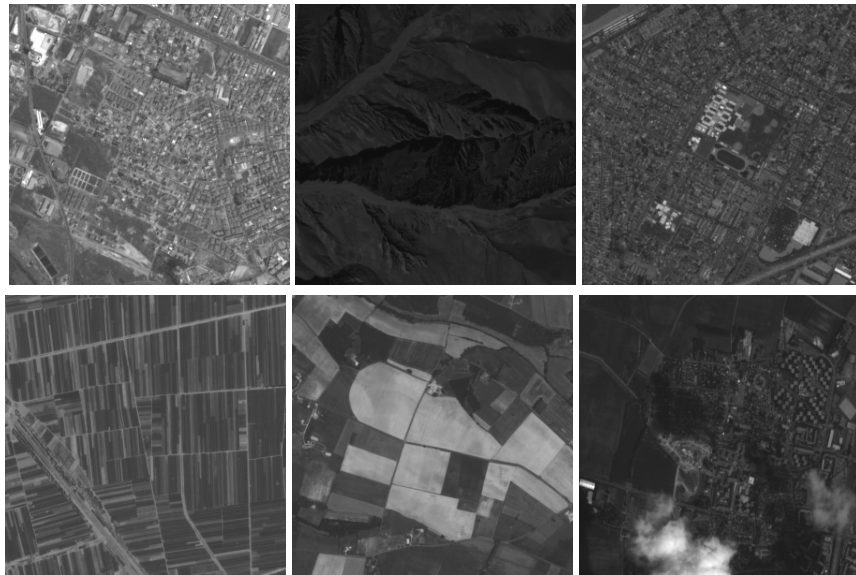
Figure 1.2: Images panchromatiques SPOT5 à une résolution de 2.5m ©CNES

automatique d'images. Le but est alors de détecter les structures inconnues, c'est à dire les ensembles cohérents de données qui ne sont pas représentées dans la base d'apprentissage et ainsi de guider l'utilisateur dans son exploration de la base. Le second scénario concerne les systèmes de recherche interactive d'images. L'idée est d'exploiter une structuration des données, sous la forme d'un clustering par exemple, pour accélérer l'apprentissage (i.e. minimiser le nombre d'itérations de feedback) dans le cadre d'un système avec boucle de pertinence. La nouveauté de nos contributions se situe autour du fait que la plupart des méthodes semi-supervisées ne per-

mettent pas de travailler avec de gros volumes de données comme on en rencontre en imagerie satellitaire ou alors ne sont pas temps-réel ce qui est problématique dans un système avec retour de pertinence où la fluidité des interactions avec l'utilisateur est à privilégier. Un autre problème qui justifie nos contributions est le fait que la plupart des méthodes semi-supervisées font l'hypothèse que la distribution des données labellisées suit la distribution des données non labellisées, hypothèse qui n'est pas vérifiée dans notre cas du fait de la non-exhaustivité des bases d'apprentissage et donc de l'existence de structures inconnues au niveau des données non labellisées. La dernière partie de cette thèse concerne un système de recherche d'objets à l'intérieur d'un schéma de type apprentissage actif. Une stratégie de type "coarse-to-fine" est introduite pour autoriser l'analyse de la base d'images à une taille de patch beaucoup plus "fine" tout en maintenant un nombre raisonnable d'évaluations de la fonction de décision du classificateur utilisé à chaque itération de la boucle d'apprentissage actif. L'idée est d'élaguer de grandes parties de la base de données à une échelle d'analyse dite "grossière", afin de réserver un traitement plus complexe et plus coÃżteux sur des zones restreintes et plus prometteuses des images.

Ce document est divisé en quatre parties. Les trois parties qui suivent donnent un aperçu de nos trois contributions. Le lecteur est invité est invité à se référer au manuscrit de thèse pour une description plus complète des solutions proposées. La quatrième et dernière partie regroupe les conclusions et quelques perspectives et présente le synopsis d'un système global unifiant nos trois contributions.

1.4 Méthodes semi-supervisées pour l'annotation automatique d'images et la découverte de structures sémantiques inconnues

1.4.1 Description de la méthode

Dans cette partie nous décrivons un système d'annotation automatique d'images fonctionnant avec une base d'apprentissage non-exhaustive. L'idée de non-exhaustivité est fondamentale en imagerie satellitaire où il est pour ainsi dire impossible de construire une base d'apprentissage résumant toutes les structures présentes à l'intérieur des images. En d'autres termes, il existe de nombreuses classes sémantiques pour lesquelles on ne possède pas d'exemple d'apprentissage. Dans la suite, nous proposons une méthode semi-supervisée qui permet d'étendre le processus d'annotation aux classes sémantiques inconnues avec des labels automatiques de type "classe inconnue 1", "classe inconnue 2" …Le principe n'est pas d'apporter une sémantique aux classes inconnues mais de les identifier en vue d'une annotation ultérieure par l'utilisateur. L'idée est donc de guider l'utilisateur dans sa tÃĄche d'exploration de la base et de l'orienter vers les structures non encore explorées. Un autre avantage lié à l'utilisation d'une méthode semi-supervisée réside dans la qualité des estimateurs statistiques obtenus: du fait de l'utilisation des données non labellisées, les méthodes d'estimation utilisées ne sont pas sujettes au "small sample size problem" Shahshahani and Landgrebe [1994] qui intervient dans les problèmes d'estimation statistique où le nombre de paramètres à estimer est du même ordre de grandeur que le nombre

d'échantillons d'apprentissage.
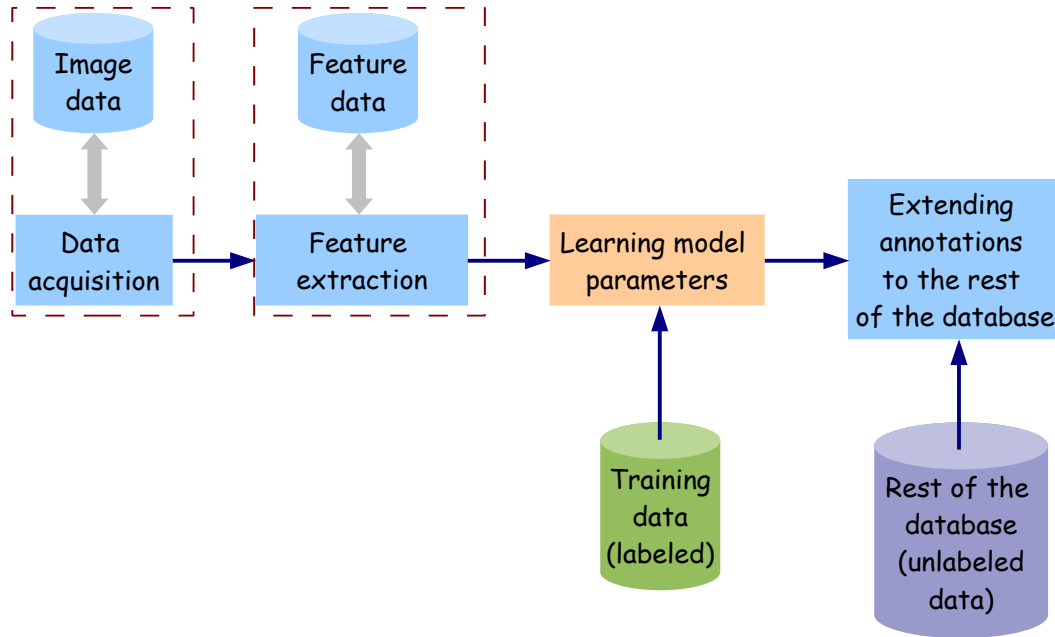
Figure 1.3:   Architecture générique de système pour effectuer l'annotation automatique de bases d'images satellitaires. La première étape consiste à extraire des descripteurs des images. Une base d'apprentissage constituée d'images annotées est ensuite utilisée pour calculer les paramètres du modèle d'annotation. Ce modèle est ensuite réutilisé pour étendre les annotations au reste de la base d'images.

La figure 1.3 résume les différentes étapes suivies pour extraire le modèle des données complètes (labellisées et non labellisées). On commence par extraire le modèle M correspondant à la partie connue des données en utilisant uniquement la base d'apprentissage (qui, par définition, contient des exemples de toutes les classes sémantiques présentes dans la partie connue du modèle: sur le diagramme de la figure 1.3, il s'agit des classes "zones urbaines", "déserts", "forêts" et "champs"). Le modèle M est ensuite réutilisé pour estimer le modèle complet des données. étant donné que la partie non labellisée des données contient à la fois des classes sémantiques connues et inconnues, nous l'utilisons pour apprendre le modèle $\bar{M}$ correspondant aux classes sémantiques inconnues. On exploite la connaissance du modèle M des classes sémantiques connues pour déterminer lesquels des échantillons de la base appartiennent aux classes sémantiques inconnues. Ces échantillons sont ensuite utilisés pour apprendre la deuxième partie $\bar{M}$ du modèle et les échantillons non labellisés restant sont ajoutés aux échantillons labellisés pour améliorer l'estimation des paramètres du modèle M.

Notre modèle est similaire par nature au modèle LDA. On part de mots visuels (les attributs primitifs extraits des images dans notre cas) que l'on associe aux concepts sémantiques de haut niveau en utilisant des variables intermédiaires latentes qui sont dans notre cas des composantes d'un mélange de Gaussiennes définies sur l'espace des attributs primitifs. Les composantes du modèle de mélange ne sont donc pas estimées directement sur les données mais sont "déduites" à l'instar des modèles
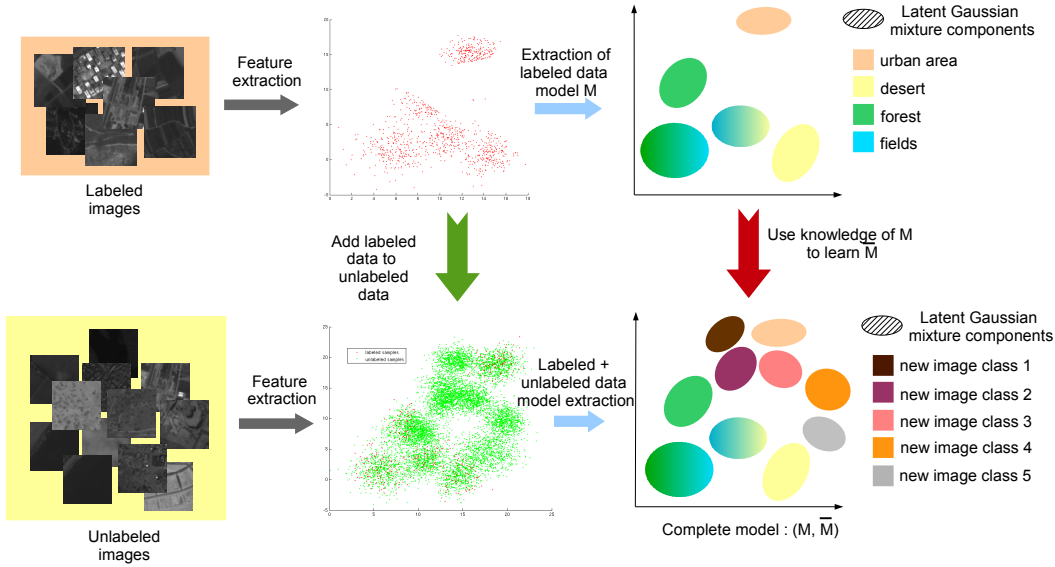
Figure 1.4: Synopsis des différentes étapes suivies pour extraire le modèle complet des données (i.e. le modèle des données labellisées et non-labellisées)

à variables latentes à partir des associations entre concepts sémantiques et attributs primitifs dans la base d'apprentissage. Ainsi chaque concept sémantique est défini par plusieurs composantes du modèle de mélange. Plus précisément, un concept sémantique sera représenté par un vecteur de probabilités caractérisant la probabilité qu'a le concept d'être associé à chaque composante du mélange. Cette idée est illustré par le diagramme de la figure 1.5. Les concepts sémantiques utilisés pour l'annotation sont appelés des concepts atomiques: on fait en effet l'hypothèse qu'il s'agit de concepts peu subjectifs qui ont des interprétations relativement similaires suivant les utilisateurs.
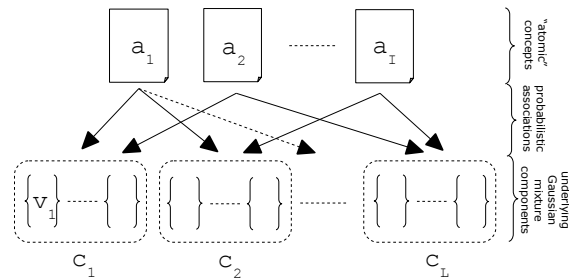
Figure 1.5: Modèle à variables latentes. Chaque variable observée, i.e. chaque concept atomique a_i est expliqué par une ou plusieurs composantes gaussiennes latentes c_l.

Pour les développements techniques et algorithmiques, le lecteur est invité à se référer au chapitre 5 du manuscrit. La partie qui suit donne un aperçu de quelques résultats obtenus sur une base d'images satellitaires SPOT5.
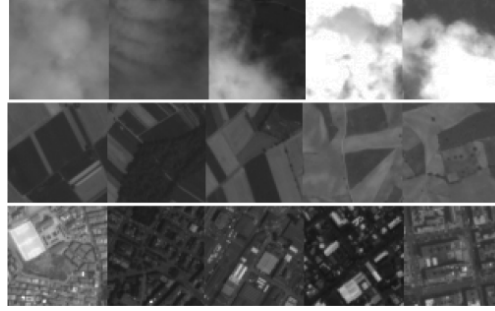
Figure 1.6:   Exemples d'images de la base d'apprentissage annotées à l'aide des trois concepts: "champs", "zones urbaines" et "nuages"

1.4.2   Exemple d'annotation automatique sur des images SPOT5

Nous avons testé notre système d'annotation automatique et de recherche de classes sémantiques inconnues sur une base d'images SPOT5 panchromatiques à 2.5 mètres de résolution. La base de test utilisée est composée de 64 images 3000 × 3000 fournies par le CNES [1]. Nous avons découpé chaque image en patchs plus petits de taille 64 × 64. Chaque patch est ensuite subdivisé en 4 "sous-patchs" sur lesquels sont extraits les vecteurs d'attributs primitifs qui sont une combinaison d'attributs de texture (attributs d'Haralick et filtres miroirs en quadrature) avec des attributs de couleur (histogrammes pondérés de niveaux de gris). Une réduction de dimensionnalité est effectuée à l'aide d'une analyse en composantes principales sur les attributs de texture et de couleur pris séparément. La base d'apprentissage que nous utilisons est constituée de 600 images 64 × 64 prises en dehors de la base de test. Trois concepts atomiques sont utilisés pour annoter ces images: "champs", "zones urbaines" et "nuages". La figure 1.6 montre des exemples d'images annotées prises dans la base d'apprentissage. Les images représentées sont relativement "pures", c'est à dire qu'un seul concept est généralement suffisant pour annoter ces images mais la base d'apprentissage utilisée comporte également des images annotées à l'aide de plusieurs concepts atomiques (par exemple, une image occupée partiellement par une zone urbaine et partiellement par des champs).

Pour appliquer notre procédure d'annotation, nous avons besoin d'une estimation du nombre de composantes du modèle de mélange à la fois pour les parties connues et inconnues du modèle. Nous avons utilisé un critère de type BIC (Bayesian Information Criterion) McQuarrie and Tsai [1998]. Les images de la figure 1.7 représentent les cartes a posteriori, c'est à dire les images obtenues en attribuant à chaque patch la classe $a_j = \mathrm{argmax}_{a_i} p(a_i | Img)$. Pour évaluer les performances de notre algorithme, nous nous sommes comparés à un algorithme SVM semi-supervisé proposé par Bruzzone et Al. Bruzzone et al. [2006]. Nous avons calculé dans chaque cas des matrices de confusion en assimilant les concepts les plus probables à des labels de classe (notre algorithme fournit en effet des résultats sous forme de vecteurs de probabilités qui sont inadaptés pour une évaluation à l'aide de matrices de confusion). Cinq structures inconnues ont été identifiées auxquelles on peut attribuer les étiquettes suivantes: zones montagneuses, zones boisées, désert, mer et structures

---

[1]Centre National d'études Spatiales

	(a)		
	annotations		
vérité	zones urbaines	champs	nuages
zones urbaines	**0.69**	0.21	0.1
champs	0.22	**0.68**	0.1
nuages	0.04	0.16	**0.8**

	(b)		
	annotations		
vérité	zones urbaines	champs	nuages
zones urbaines	**0.74**	0.17	0.09
champs	0.15	**0.77**	0.08
nuages	0.03	0.11	**0.86**

	(c)				
	annotations				
vérité	zones urbaines	champs	nuages	désert	mer
zones urbaines	**0.79**	0.07	0.05	0.06	0.03
champs	0.05	**0.81**	0.02	0.09	0.03
nuages	0.01	0.12	**0.8**	0.07	0
désert	0.06	0.13	0	**0.78**	0.03
mer	0.02	0.02	0.03	0.01	**0.92**

Table 1.1: Matrices de confusion. Table I(a): cas supervisé (obtenu en utilisant l'algorithme présenté dans la section 5.2.1 du manuscrit de thèse); Table I(b): SVM semi-supervisé; Table I(c): cas semi-supervisé avec détection de structures inconnues.

urbaines (notamment aéroports et ports). Il est bien entendu que les étiquettes des classes inconnues sont établies a posteriori par l'utilisateur, notre algorithme, lui, ne leur attribue que des labels automatiques sous la forme "classe inconnue 1, 2" .... Nous avons retenu les deux classes inconnues les plus représentées dans notre base de test (désert et mer) et nous les avons intégrées dans la troisième matrice de confusion qui illustre les performances de notre algorithme de détection de classes inconnues. On remarque que les performances sont supérieures à celles du SVM semi-supervisé pour les classes "zones urbaines" et "champs" mais pas pour la classe "nuages". Cela vient du fait que cette classe est presque monomodale du point de vie signal est donc qu'elle possède une bonne représentation au niveau de la base d'apprentissage. Dans ce cas de figure, les méthodes discriminantes comme les SVMs fonctionnent toujours mieux qu'un modèle génératif. Une des hypothèses principales sur lesquelles reposent les techniques SVMs, en effet, est que la distribution des données à l'intérieur de la base d'apprentissage reflète celle des données non labellisées ce qui est souvent le cas lorsque les classes sont monomodales. Par contre, dans le cas de classes multimodales, il peut arriver qu'il y ait des modes non représentés au niveau de la base d'apprentissage. C'est ce qui se produit notamment pour les classes "champs" et "zones urbaines". Les modes non représentés peuvent alors perturber l'apprentissage du SVM transductif en agissant comme du bruit lors de l'apprentissage du classificateur, c'est d'ailleurs un des problèmes principaux auxquels est sujet ce genre de méthode.

La partie suivante présente l'utilisation d'une méthode semi-supervisée dans la cadre d'un système de recherche interactive d'images. L'annotation automatique te la recherche interactive d'images sont en effet deux paradigmes qui peuvent être envisagés dans une optique complémentaire. Un des inconvénients majeurs des systèmes d'auto-annotation est en effet la nécessité d'avoir une base d'apprentissage annotée ce qui constitue une limitation en terme de diversité des requêtes possibles car ces dernières sont spécifiées à l'aide des mots du vocabulaire utilisé pour annoter les images dans la base d'apprentissage. C'est pour pallier ce problème que les systèmes de
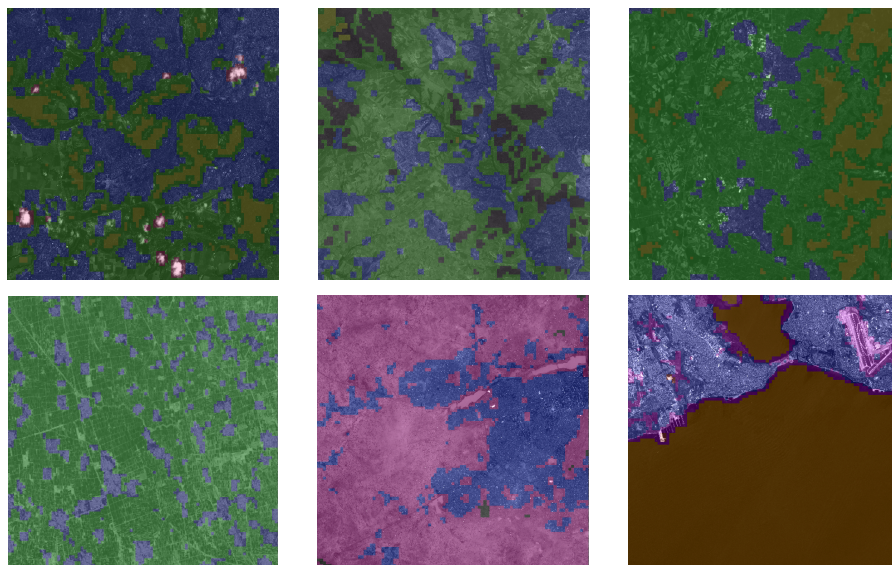
Figure 1.7: Cartes a posteriori obtenues en appliquant notre algorithme sur deux images SPOT5 panchromatiques: les différentes couleurs correspondent aux concepts atomiques avec la probabilité a posteriori la plus haute. Les concepts représentés ici sont: "zones urbaines" (en bleu), "champs" (en vert), "nuages" (en rouge grenat). Cinq structures inconnues ont été détectées correspondant respectivement à des zones montagneuses (en gris foncé sur l'image en haut au milieu), des zones boisées (en vert kaki sur les trois images du haut), des zones désertiques (en rose sur l'image en bas au milieu), de la mer (en marron-orangé sur l'image en bas à droite) et des structures urbaines (en violet sur l'image en bas à droite).

Figure 1.8: Détails des cartes a posteriori obtenues

recherche interactive d'images ont été mis au point. La requête y est spécifiée à l'aide d'une image exemple ("query by content") ce qui autorise a priori un nombre infini de requêtes possibles et permet à l'utilisateur de spécifier de manière naturelle et très précise ce qu'il recherche.

1.5   Intégration des données non-labellisées dans le cadre de la recherche interactive d'images

1.5.1   Description de la méthode

Dans ce chapitre, nous proposons une méthode d'apprentissage actif semi-supervisée qui se distingue des méthodes précédentes par sa capacité à gérer de gros volumes de données et sa facilité d'intégration dans un schéma d'apprentissage actif. Les données non-labellisées sont exploitées par l'intermédiaire d'une structuration de l'espace des données sous forme de clustering. Le schéma de la figure 1.9 présente un "synopsis" complet du système: l'utilisateur commence par pointer les prototypes des clusters qu'il juge pertinents par rapport à sa requête. Une fois cette phase d'initialisation effectuée, l'apprentissage se fait à l'intérieur d'une boucle d'apprentissage actif. Le but de la boucle est de construire itérativement une surface SVM approximant la catégorie recherchée par l'utilisateur. Pour ce faire, un SVM modifié permettant de travailler directement au niveau des clusters est défini. Le repositionnement de la surface SVM à chaque itération de la boucle se fait en ajustant de manière itérative les enveloppes convexes des clusters à partir du feedback de l'utilisateur et en ré-entraÃőnant le SVM modifié avec les nouvelles enveloppes. L'utilisateur se contente de donner un feedback sur des points précis appelés points critiques qui sont (re)définis à chaque itération comme les points des clusters les plus proches de la surface SVM courante.

Un concept alternatif permettant à l'utilisateur d'introduire progressivement au cours de l'apprentissage les clusters dont il juge les prototypes pertinents par rapport à sa requête est aussi présenté dans le chapitre 6 (cf. section 6.3.2). Le diagramme de la figure 1.10 présente un synopsis de ce système.

Le lecteur est invité à se référer au chapitre 6 du manuscrit pour les développements théoriques et algorithmiques associés à la méthode.

1.5.2   Validation expérimentale

Nous avons réalisé nos tests sur une base d'images satellite haute résolution Quick-Bird. Il s'agit d'images panchromatiques avec une résolution au sol de 61cm représentant des survols d'Acapulco, Las Vegas, Los Angeles, Londres et Ouagadougou. Nous avons 10 images de tailles approximatives 30000x30000. Nous extrayons des descripteurs à l'intérieur d'une fenêtre glissante de taille 200x200 que nous promenons sur toute l'image en la décalant d'un pas égal à la moitié de la fenêtre à chaque fois. Ainsi pour une image 30000x30000, nous obtenons environ 90000 vecteurs descripteurs ce qui fait une base de données contenant approximativement 900000 points (le terme "point" faisant référence aux vecteurs descripteurs).
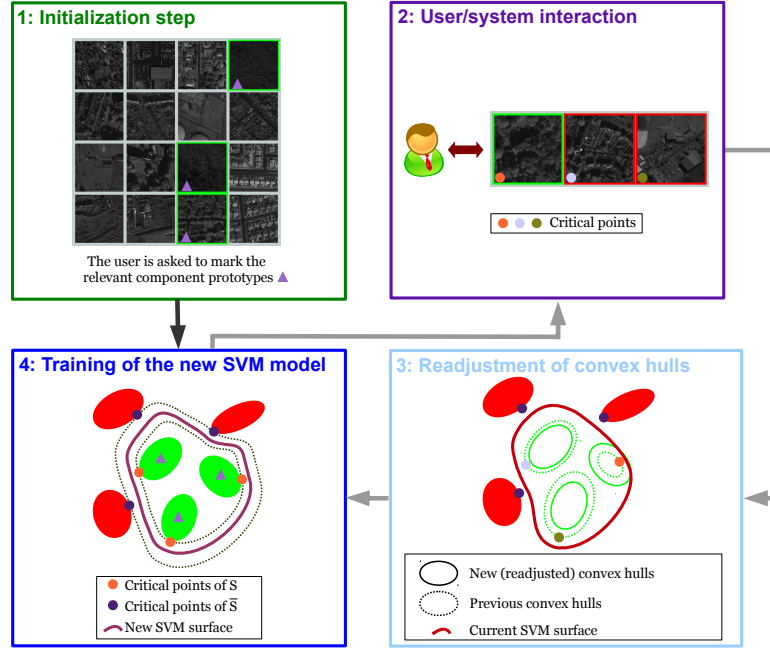
Figure 1.9: Synopsis du système complet.

Les diagrammes de la figure Fig. 1.11 représentent la précision, le rappel et le nombre de vecteurs supports du classificateur SVM courant en fonction du nombre d'itérations dans la boucle d'apprentissage actif. Nous voyons qu'avec notre méthode, nous arrivons beaucoup plus vite à saturation en terme d'apprentissage, c'est à dire que nous avons besoin de beaucoup moins d'itérations que la méthode dite "baseline" dans la boucle d'apprentissage actif pour arriver à classificateur qui n'évolue plus en terme de précision et de rappel.

1.6   Apprentissage actif en cascade pour la détection d'objets dans des bases d'image satellites haute résolution

1.6.1   Description de la méthode

Le lecteur est invité à se référer au chapitre 7 du manuscrit pour les développements théoriques et algorithmiques associés à la méthode dont nous donnons un aperçu ci-dessous.

La méthode que nous développons dans cette partie a pour objectif d'effectuer de la détection d'objets à l'intérieur d'un schéma de type apprentissage actif dans des bases d'images satellites haute résolution. L'avantage du schéma interactif est de permettre à l'utilisateur de rechercher tout objet qu'il juge intéressant et pas seulement les objets qui sont répertoriés dans la base d'apprentissage (qui est inexistante dans notre cas). La principale difficulté qui émerge est de maintenir un temps de calcul raisonnable entre chaque itération d'apprentissage actif afin de garder une interaction utilisateur-système relativement fluide. Nous proposons une stratégie multi-échelle
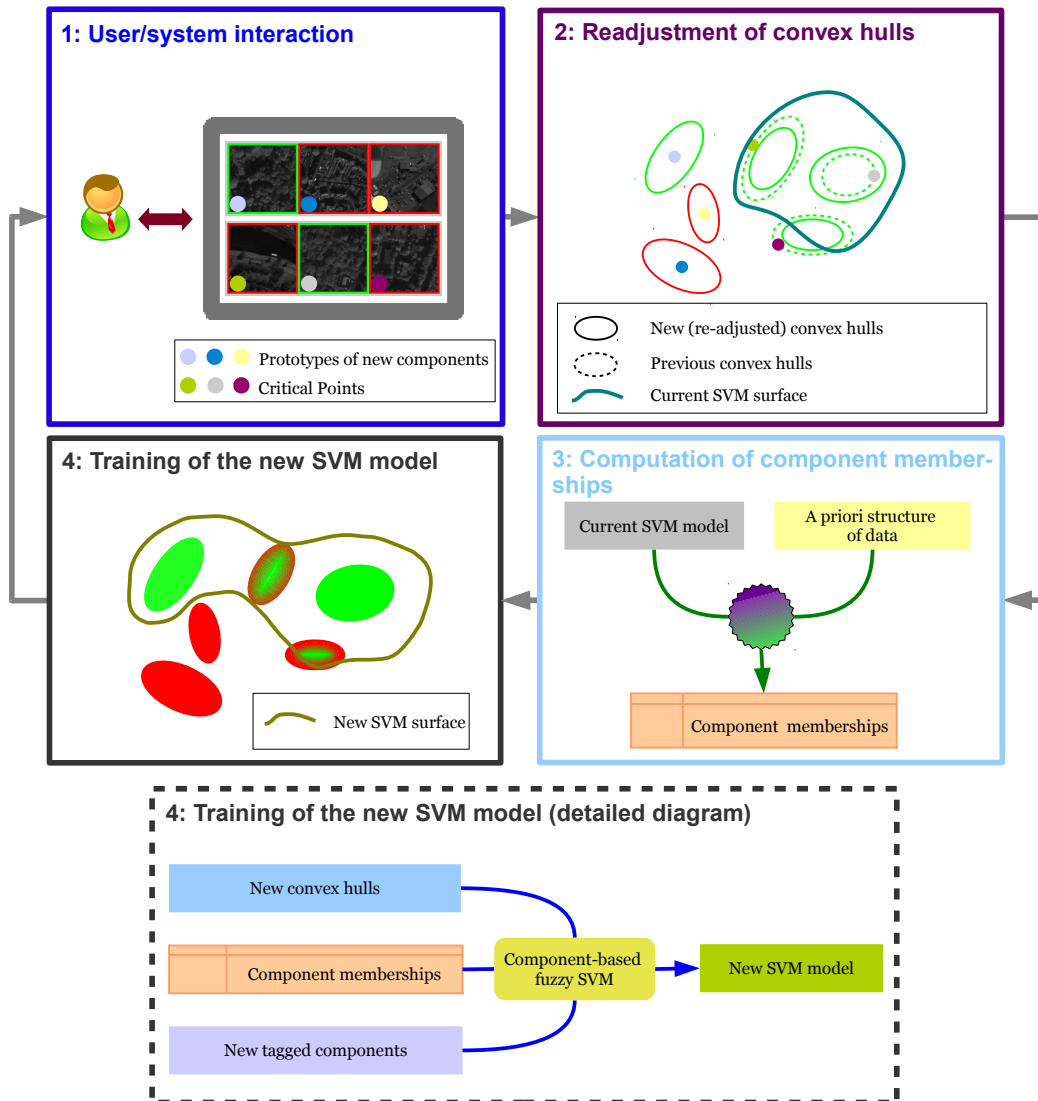
Figure 1.10: Concept proposé pour l'apprentissage "online" des composantes de mélange pertinentes par rapport à la requête utilisateur.
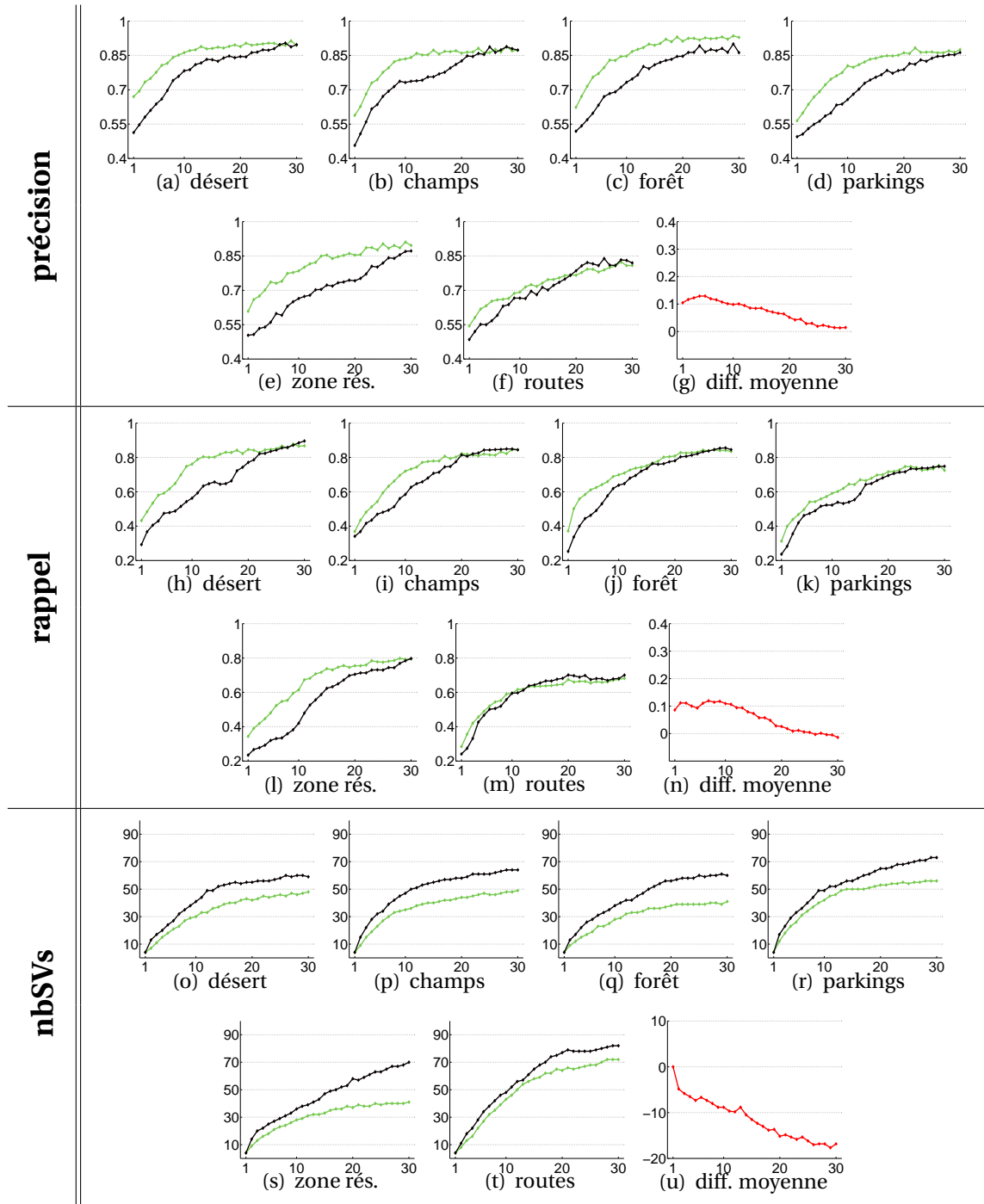
Figure 1.11: **QuickBird dataset:** comparaison en terme de **précision** (Fig. (a), (b), (c), (d), (e), (f), (g)), **rappel** (Fig. (h), (i), (j), (k), (l), (m), (n)) et **nombre de vecteurs supports** (Fig. (o), (p), (q), (r), (s), (t), (u)) de notre méthode (courbe verte) avec une "baseline" (courbe noire) pour six classes sémantiques. L'axe des abscisses représente le nombre d'itérations d'apprentissage actif. Les courbes rouges représentent les différences moyennes de performance entre les deux méthodes sur les six classes.

de type "coarse-to-fine" implémentée sous la forme d'une cascade de classificateurs qui opèrent sur des tailles de patches de plus en plus petites. Le but de cette stratégie est de maintenir un nombre raisonnable de patches à chaque niveau de la hiérarchie afin de préserver la fluidité des interactions système / utilisateur au sein du processus d'apprentissage actif. En diminuant la taille du patch utilisé pour l'analyse, on provoque effectivement une explosion du nombre de patches à traiter: il est par conséquent nécessaire d'incorporer une stratégie visant à élaguer certaines parties de la base pour maintenir un nombre raisonnable d'évaluations de la fonction de décision du classificateur utilisé (un classificateur SVM dans notre cas) à chaque échelle. La figure 1.12 présente un résumé de cette idée.
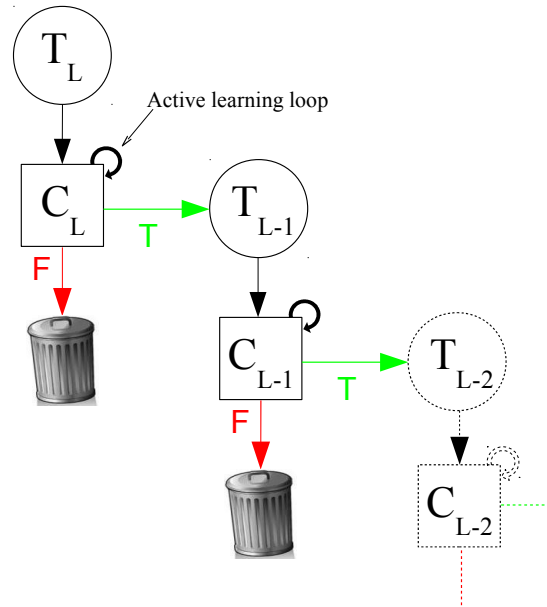
Figure 1.12: Représentation de la stratégie "coarse-to-fine" sous la forme d'une cascade de classificateurs. Le processus interactif est effectué de manière descendante à partir du haut de la cascade: on commence par construire le classificateur C_L à l'échelle L. L'ensemble T_{L-1} est défini comme l'ensemble de patches de E_{L-1} qui intersectent avec les patches classifiés positivement au sein de l'ensemble T_L. Cette procédure est répétée à chaque niveau de la cascade. Les classificateurs C_l sont construits en utilisant un processus d'apprentissage actif impliquant l'utilisateur.

1.6.2 Résultats

Nous utilisons pour les tests la base d'image QuickBird décrite dans la section 1.5.2 et dix classes d'objets présents dans les images (cf. figure 1.13). Nous nous comparons à une baseline décrite dans Ferecatu and Boujemaa [2007] qui opère à l'échelle la plus fine.

Nous voyons que notre méthode (courbes bleues) conduit à beaucoup moins d'évaluations de la fonction de décision du classificateur que la baseline SVM$_{baseline}$ (courbes vertes): sur la figure 1.14(a), nous voyons qu'il y a une différence d'environ deux ordres de
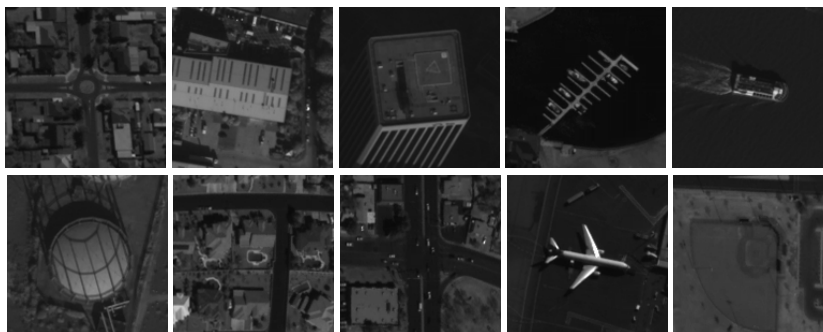
Figure 1.13:   Exemples d'objets appartenant aux dix classes cibles utilisées pour les tests.

magnitude en moyenne. Les courbes des figures 1.14(b) et 1.14(c) représentent respectivement la précision et le rappel moyennés sur les dix classes à chaque itération d'apprentissage actif. Notre méthode a des performances moyennes légèrement meilleures en terme de précision et le rappel est approximativement le même à la fin du processus d'apprentissage actif.

1.7   Conclusion

Nos contributions concernent trois aspects des systèmes de recherche d'image par analyse du contenu. Le premier est l'utilisation de méthodes semi-supervisées pour guider l'utilisateur dans son exploration de la base et l'orienter vers des structures non encore étiquetées et susceptibles de l'intéresser. Ce système repose sur l'existence d'une base d'apprentissage : un premier modèle des données labellisées est extrait et ensuite réutilisé pour apprendre un modèle complet des données labellisées et non-labellisées. Les structures sémantiques inconnues figurant dans les données non-labellisées sont annotées 'a l'aide de labels automatiques en vue d'être identifiées ensuite par l'utilisateur. Notre approche donne des résultats légèrement supérieurs à un SVM semi-supervisé dans le cas d'une base d'apprentissage très peu exhaustive. Elle possède en outre certains avantages intrinsèques comme la possibilité d'utiliser des exemples d'apprentissage possédant plusieurs annotations et celles d'identifier les structures inconnues dans les données.

   Le deuxième aspect concerne l'utilisation des données non-labellisées pour accélérer l'apprentissage dans un système de recherche interactive d'images. Nous avons proposé une méthode basée sur une structuration de l'espace sous forme de clustering. L'idée est de travailler avec une granularité de l'espace plus élevée afin de positionner très rapidement une surface SVM approximative. Nous avons également proposé une méthode de "raffinement" de cette surface basé sur l'ajustement interactif des enveloppes convexes des clusters. Les résultats obtenu sont prometteurs notamment en ce qui concerne la vitesse d'apprentissage : en moyenne, beaucoup moins d'itérations sont nécessaires à l'intérieur du processus d'apprentissage actif pour arriver à une approximation raisonnable du concept recherché par l'utilisateur. Le classificateur SVM que nous obtenons est aussi moins complexe en terme de nombre de vecteurs de support que celui obtenu avec la méthode classique proposée par Chang et Al. ce qui nous

laisse penser qu'il possède de meilleures capacités de généralisation (mais ceci reste à vérifier). Notre méthode intègre enfin certains des avantages des méthodes semi-supervisées comme l'exploitation des données non-labellisées sans les principaux inconvénients que sont le temps de calcul et l'occupation mémoire.

Le troisième aspect concerne la recherche d'objets dans les images en utilisant une approche de type apprentissage actif afin de pallier deux inconvénients majeurs des méthodes classiques de détection d'objets que sont la constitution d'une base d'apprentissage représentative pour les catégories d'objets que l'on souhaite rechercher et la limitation inhérente en terme de diversité des objets que l'on peut détecter. Notre approche présente des résultats concluant en terme de réduction du coÃżt calculatoire (à peu près deux ordres de magnitudes par rapport à la méthode dite "baseline") mais ne propose pas de solution concernant certains l'ajustement des paramètres du modèle comme le nombre d'itérations d'apprentissage actif nécessaires et la taille de patch utilisée à chaque échelle. Une perspective intéressante serait d'apprendre ces paramètres automatiquement en utilisant également le retour de l'utilisateur.

Nous avons envisagé de réunir nos différentes contributions à l'intérieur d'un système mixte combinant les avantages respectifs des systèmes d'annotation automatique et des systèmes de recherche interactive. L'idée serait d'avoir une approche permettant d'étendre les annotations de la partie connue de la base tout en autorisant la définition de nouvelles catégories via une boucle d'apprentissage interactif. Cela permettrait de définir un système complet d'exploration incrémentale de la base de données qui, à chaque session d'utilisation, incorpore dans le modèle d'annotation les nouvelles catégories définies par l'utilisateur via la composante de recherche interactive. Le diagramme de la figure 1.15 présente le synopsis d'un tel système.

(a)

(b)   (c)
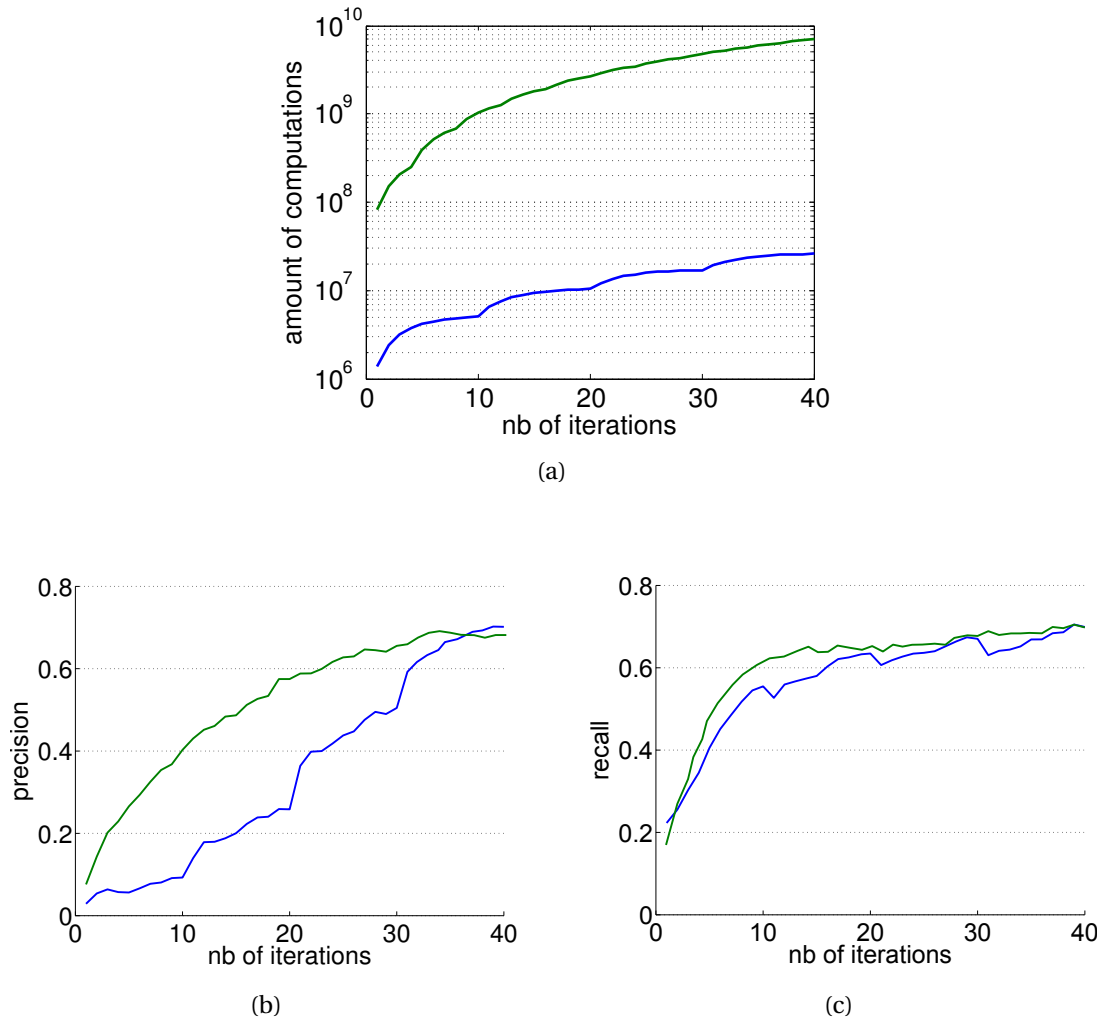
Figure 1.14:   **(a):** Nombre moyen d'évaluation de la fonction de décision du classificateur SVM à chaque itération d'apprentissage actif. **(b)** et **(c):** précision et rappel moyennés sur les 10 classes en fonction du nombre d'itérations dans la boucle d'apprentissage actif. Les courbes bleues représentent les résultats obtenus avec notre méthode et les courbes vertes les résultats obtenus avec SVM$_{baseline}$.
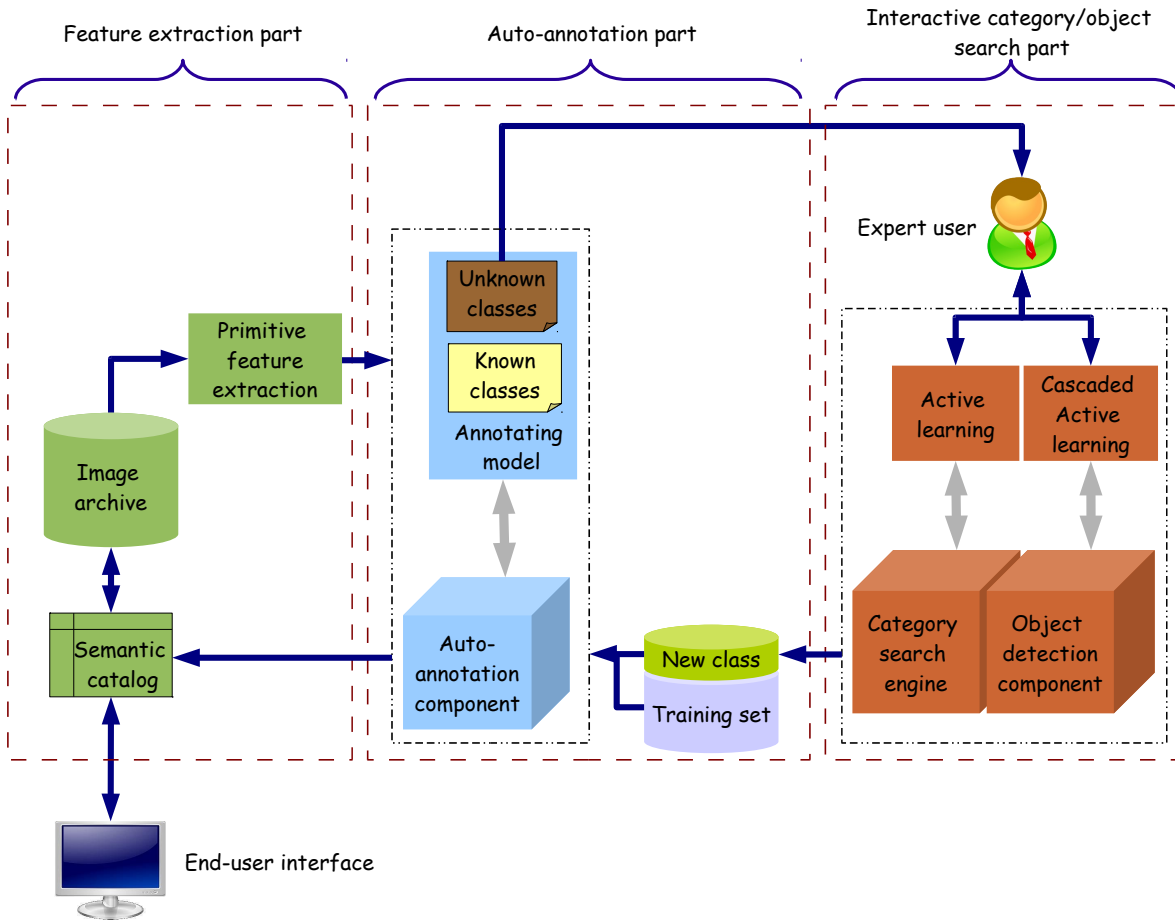
Figure 1.15: Synopsis d'un système général de fouille de données dans des bases d'images satellites combinant nos contributions par rapport aux deux principaux paradigmes de recherche d'images: l'annotation automatique et la recherche interactive de catégories / objets.

Chapter 2

Introduction

With the diversification of human activities in the multimedia domain and the technological progresses made in image acquisition devices, very large volumes of image data have become available, making obvious the need for efficient tools and methods to organize multimedia contents and ensure quick access to them. This problem is particularly sensitive in the Remote Sensing (RS) domain which is our main case of study though the methods we describe in this work could apply to a wider range of datatypes in the multimedia area. The last few years have indeed witnessed an increase in both the number and the resolution of Earth Observation (EO) imaging sensors, causing in turn the volume and the information content of EO image databases to grow exponentially during this period. In this context, it has quickly become an impracticable task for a human operator to extract manually relevant information from such databases and above all to search them exhaustively. Thus, techniques to automatically index and retrieve the information of interest have begun to emerge.

In the upcoming section, we describe the overall context associated with the acquisition and the treatment of remote sensing data. We then present an overview of remote sensor types and we mention several research topics which are considered as mainstreams by the remote sensing community. We conclude this part by positioning our contributions within these research topics.

2.1 Overall context of our work

As a result of the recent advances in sensor as well as in platform technologies, Earth Observation (EO) sensors onboard satellite platforms have been collecting increasingly large volumes of geospatial data over the past decades, principally remotely sensed imagery. Moreover, many countries have launched Earth Observation satellites, making available a large number as well as a wide variety of sensors. The geospatial domain has thus reached a point where the amount of collected EO data to manage grows by several terabytes per day. In this context, national space agencies and private entities around the world have been assigned the task of collecting, archiving, processing, disseminating and envisaging applications to the geospatial data coming from the satellites. Along with this, the information systems for Earth Observation have experienced major changes: the progresses in computer and information technologies in recent years have allowed the data to be more readily accessible through the internet and

many data centers have installed data servers allowing (controlled) web-based access to their data repositories. The apparition of such online data servers for the sharing and interoperability of geospatial data has in turn opened the way to the setting-up of various services and applications. The rising question is now how to use effectively the huge amount of geospatial data in applications. To be useful, the EO data must indeed be preprocessed to extract application-relevant information and knowledge. Traditional geographic information systems (GIS) used to rely on trained human experts for the information extraction and knowledge discovery components, rendering applications highly dependent on the availability of such experts. But the growing amount of GIS data available has rendered manual analysis an out-of-date and nearly impracticable technique. It has thus become an urgent research area to which this thesis is a modest contribution to evolve systems able to convert geospatial data into user/application specific knowledge. Such knowledge building systems can then be used to provide geospatial knowledge services for a wide range of applications. In this work, we focus on the problem of knowledge extraction from high-volume satellite image databases, letting aside the additional information which comes with the images in the GIS data.

2.1.1   Remote sensing imaging sensors and data acquisition techniques

Remote sensors are devices which acquire measurements of the earth surface. They are divided into three categories depending on the platform they are installed on: satellite, airborne and ground-based sensors. In the following, we focus on sensors onboard polar-orbiting Earth Observation satellites. These sensors provide a global coverage of the earth surface with varying revisit frequencies. They can be classified according to the number and the frequency range of the bands they can detect. We can roughly distinguish among the following categories:

- Panchromatic sensors cover a continuous range of bands in the visible or near infrared light spectrum.

- Multispectral sensors cover simultaneously several distinct bands with generally quite large spectral bandwidths.

- Hyperspectral sensors cover spectral bands which are much narrower than those covered by multispectral sensors. They allow the recording of several hundreds of bands at the same time leading to a much greater spectral resolution than in the multispectral case.

- Synthetic Aperture Radar (SAR) sensors

In our case, we principally dealt with panchromatic imagery provided by meter- and half-a-meter-resolution sensors. Among meter-resolution optical imaging sensors, we can mention SPOT5 which possesses a resolution of 2.5 meter in panchromatic mode and 4 spectral bands with a resolution of 10 meters each. This satellite was launched with the purpose of performing environmental monitoring tasks such as detecting and forecasting phenomena involving climatology and oceanography, monitoring human activities such as deforestation and urban expansion ... Recent years

Figure 2.1: Images centered on the same spot with three different resolutions. From left to right: 2m, 1m and 60cm.

have seen the emergence of sensors with sub-metric resolutions: this is the case of QuickBird for instance which provides a panchromatic mode with a ground resolution of 60 centimeters and a multispectral mode consisting of four bands at a resolution of 2.4 meters. At this resolution, man-made structures such as buildings or even houses are easily visible, making the image interpretation task even more complex due to the increased semantic content, that is, the fact that the number of semantic classes observable in the images is potentially much higher. Ikonos and WorldViews 1 and 2 are other examples of sub-metric sensors collecting both panchromatic and multispectral imagery. The images in the figure 2.1 represent the same ground portion taken by different sensors of increasing resolutions (we start with a resolution of 2 meters on the left and we go up to a resolution of 60 cm on the right). It gives an idea of how image semantics "behave" as the resolution increases: on the left image, we only observe urban structures whereas on the middle image, we clearly see vehicles and on the right image, we begin to distinguish people. Assessing automatically the semantic content of an image, i.e., estimating the number of semantic classes it contains is an open question which is still under investigation by many researchers.

The figures 2.2 and 2.3 show samples of respectively 60cm resolution panchromatic QuickBird images and 2.5m resolution panchromatic SPOT5 images. The semantic diversity we can observe in the images is decupled by the huge amount of data collected every day. To mention several characteristics of QuickBird, it possesses an onboard storage capacity of 128GB which is used to store about 57 scenes of size 30000 × 30000 per orbit, the orbital period lasting about 90 minutes. The acquired image data are indexed with their geographic position but with no semantic information, which is a big difference compared to multimedia databases that are most of the time (especially images coming from the internet) indexed semantically.

2.1.2 Research topics related to remote sensing and overview of the contributions

Research areas in remote sensing include standard topics in image interpretation, information mining and dynamic modeling. The first category covers generic domains such as remote sensing image classification, data/information fusion and change detection. The second one includes topics such as query by example and more generic

Figure 2.2: Panchromatic QuickBird images at a resolution of 61cm ©DigitalGlobe
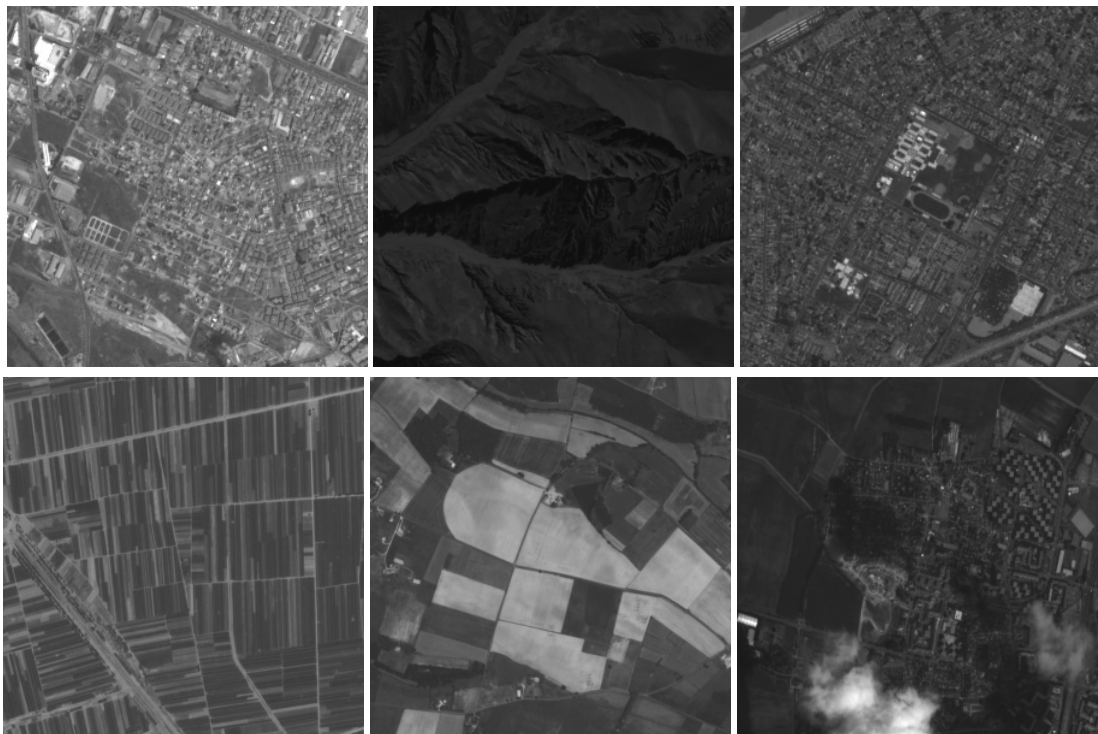
Figure 2.3: Panchromatic SPOT5 images at a resolution of 2.5m ©CNES

topics related to information mining systems such as content-based image query systems, interactive image search engines, auto-annotation systems. The third category refers to the understanding of how complex systems behave over time. It covers topics such as risk assessment, identification of potential regions using geospatial information systems, spatio-temporal modeling of dynamic phenomena in geospatial information systems etc. Our own contributions are situated between the image interpretation and the information mining category. We indeed address some problems related to mining large image databases inside the framework of interactive image search engines and auto-annotation systems but it cannot be done without resorting at some point to image interpretation tools such as unsupervised/supervised/semi-supervised classification methods.

In this thesis, we place ourselves in the framework of *auto-annotation systems* and *interactive image search engines* which are two rather different approaches towards information mining in image databases. Auto-annotation systems try to associate keywords belonging to a predefined vocabulary to the images in the database. Interactive image search engines can be assimilated to standard category search engines with the difference that the targeted concept is built using an interactive process involving the user: the latter is asked to assess the results provided by the system at each iteration of an interactive loop, rendering the building of the targeted category equivalent to an exchange of information between the system and the human agent operating on it.

In each of our contributions, we explore a number of issues related to one or the other of these two broad categories of systems. Our first two contributions are centered around the problem of taking advantage of both labeled and unlabeled data through the use of *semi-supervised* methods. Labeled data are indeed very expensive to obtain because they require a human agent to determine explicitly the corresponding semantic category they belong to. In this context, it is a natural idea to try to make the most out of the huge volume of unlabeled data, which, on the contrary, are almost costless to acquire.

In our first contribution, we propose a semi-supervised algorithm to perform auto-annotation of satellite image databases and *discover "unknown" semantic structures* inside the images. A setup which has indeed hardly been explored in auto-annotating systems, but, which is the rule rather than the exception, is the case when the training database used to learn the mapping function between images and semantic concepts is not exhaustive regarding the semantic content of images. In other words, there exists "unknown" image classes, i.e., classes for which there isn't any representative in the training database used to train the auto-annotating model. Consequently, we propose a semi-supervised method allowing to incorporate within the training process the unlabeled data which by definition contain the "unknown" image classes, the purpose being to identify in the end both the "known" and the "unknown" image classes. The latter are assigned automatic labels in the perspective of future annotation, the idea being to help the user in his database exploration task. We demonstrate the performance of our algorithm on a database of SPOT5 optical satellite images. We showed that beside the "unknown" classes discovery feature, our algorithm achieves superior performance (using confusion matrices as a performance indicator) compared to a semi-supervised Support Vector Machine (SVM).

In our second contribution, we introduce an active learning algorithm that incor-

porates the intrinsic data distribution modeled as a mixture of Gaussians *to speed up the learning* of the target class using an interactive relevance feedback process. The idea is to obtain a quick approximation of the targeted category by working at a low "granularity" determined by the convex hulls/equiprobable envelopes of the mixture components. Our scheme rely on a fundamental assumption of semi-supervised methods – the "cluster assumption" – which states that two elements of the same cluster are likely to belong to the same semantic class. The active learning process consists in re-adjusting iteratively the convex hulls to refine the definition of the target class. The underlying idea is that by adjusting the "extent" of the convex envelopes, we get closer to the cluster assumption, which allows us to obtain a better approximation of the target class. Further refinement can then be achieved by "bringing" back the learning at the original data points granularity. The proposed algorithm belongs to the class of semi-supervised algorithms in the sense that it implicitly uses the unlabeled data in the learning process through the unsupervised modeling of the intrinsic data distribution as a mixture of Gaussians. Results on a database of high-resolution QuickBird images and on a generalist database of color images show that our algorithm performs as well as other state-of-art interactive image search engines in terms of precision and recall while still achieving a noticeable diminution in the number of iterations needed in the active learning loop to arrive at a definition of the target class which satisfies the user.

In our third contribution, we propose an active learning scheme to perform object retrieval in high-resolution optical satellite image databases. Object retrieval is a fundamental challenge in machine learning but is often subject to the problem of gathering enough labeled examples of the target object, and, also, to the computational complexity inherent to the training and the evaluation of complex classifier functions on large databases. To cope with this, we propose a hierarchical top-down processing scheme to retrieve objects in high-volume image databases. We learn via a multistage active learning process a cascade of classifiers working each at a certain scale on a patch-based representation of images. The active learning component removes the dependence on the existence of a representative training set for each object class and avoids the need for an offline training phase. The underlying idea of our method is similar in nature to that behind coarse-to-fine testing, i.e., we seek to eliminate at each stage of the cascade large parts of images considered as non-relevant, the purpose being to set the focus at the finest scales on more promising and as spatially limited as possible areas. The whole scheme relies on the fact that by reducing the size of the analysis window (i.e. the size of the patch), we better capture the properties of the targeted object. The cascaded scheme is introduced to compensate for the extra computational burden incurred by diminishing the size of the patch (which causes an explosion of the number of patches to process). Tests on a database of 61cm resolution QuickBird panchromatic images show the validity of our approach in terms of precision and recall compared to a recent state-of-the-art method and, most importantly, we achieve a reduction of the number of computations (number of evaluations of the decision function) of two orders in magnitude compared to the aforementioned state-of-the-art method.

2.1.3   Structure of the document

This work is divided into five parts.

The first part gives an overview of the state-of-the-art in the domain of content-based image retrieval. We introduce the general problems of the field by giving an insight into the following topics:

- we start with the extraction of information from the images. We detail the principal methods of computing concise mathematical representations from the images and we discuss the ability of these representations to capture the underlying image semantics. We also discuss several ways of building image signatures from the extracted image descriptors. We conclude this part by presenting an overview of the state-of-the-art in the field of attribute selection which is one of the sub-domains of image retrieval which has concentrated the greatest efforts since the emergence of the field in the 80's.

- we then make a quick tour of the literature dealing with the three main learning paradigms, i.e., unsupervised, semi-supervised and supervised learning.

- we conclude the chapter by describing some state-of-the-art content-based image retrieval systems, making the distinction between auto-annotation systems and interactive image search engines. This allows us to position our contributions within these two broad categories of systems and to introduce the particular case of satellite imagery, satellite image databases being characterized by a huge semantic diversity along with quasi nonexistent training datasets, but, instead of that, very high volumes of unlabeled data.

The second part deals with some theoretical aspects which are necessary to fully understand our contributions to the field. We mainly review some techniques among the three learning paradigms mentioned above and we mention how these techniques can be adapted to different learning strategies, the three strategies we are interested in being "active learning", "multiple instance learning" and "cascaded learning".

The next three parts correspond to our contributions in the domain of auto-annotation systems and interactive image search engines. In chapter 5, we present a semi-supervised algorithm to perform auto-annotation of image databases and to discover "unknown structures" among these images. Then, in chapter 6, we introduce an active learning algorithm which exploits the intrinsic structure of the data to speed up the learning of the target category in the context of interactive image search engines. And last, in chapter 7, we describe a cascaded active learning scheme to perform object retrieval in large satellite image repositories. The figure 2.4 gives an hint about how our three contributions can be combined to perform information mining in large satellite image databases. Further explanations about the related concept are provided in section 3.5. We conclude this work by envisaging some perspectives and further developments to the ideas presented throughout it.
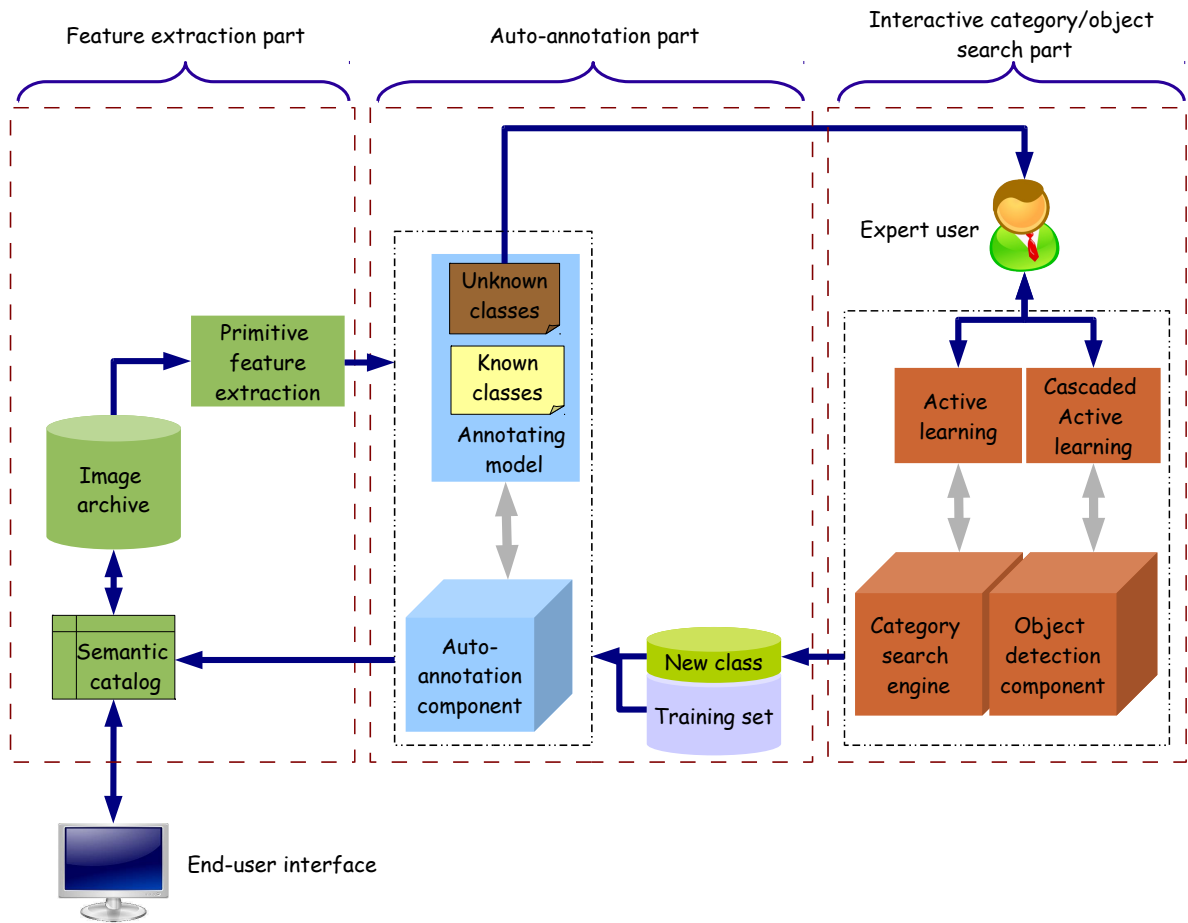
Figure 2.4: Concept for a general purpose system combining our contributions in the domains of auto-annotation, category search and object retrieval to perform information mining in large satellite image databases.

Part I

State of the art of content-based image retrieval, dicussion and theoretical concepts for classification and learning

Chapter 3

Content-based image retrieval: state of the art and discussion

The development of digital imaging in the 1980s and the 1990s marked a turning point in the way images were acquired and stored: traditional camera film reels were replaced by multimedia databases leading to increased browsing and searching facilities as well as augmented storage capabilities. But along with the numerous advantages of digital imaging such as the ease of acquisition and storage appeared the problem of handling the exponential growth in size and variety of digital image databases. In such a context and also because of the extensive needs associated with multimedia data search, evolving techniques for effective indexing and searching of high-volume multimedia databases has become in recent years a major and urgent research topic. It has been encouraged and supported at the same time by the rapid growth of computational power in computers, rendering realistic the processing of non-textual data such as images and videos in the framework of industrial applications. Earth Observation image databases have followed the same trend. The scope, coverage and volume of geospatial data sets have grown rapidly due to the progress in image acquisition and data processing technologies. Recent Earth Observation repositories have thus become vast heterogeneous data warehouses, created, processed, and disseminated by government as well as private-sector agencies. They include vast amounts of geo-referenced digital imagery acquired through high-resolution remote sensing systems and other monitoring devices, geographic and spatiotemporal data collected by global positioning systems as well as other position-aware devices, including cellular phones and in-vehicle navigation systems. Geospatial images are being used in many applications, including hazard monitoring, drought management, commercial land use planning, agricultural productivity, forestry, tropical cyclone detection, and other civil and intelligence applications. The range of users accessing remote-sensing images varies from expert users such as meteorologists to novices such as farmers, trying to access and interpret these images, especially satellite images.

In this chapter, we give an overview of state-of-art techniques in the domain of content-based image retrieval. We start with a general overview of content-based image retrieval systems, setting the accent on the underlying problematics. Next, we focus on methods related to the extraction and representation of information coming from images. Then, we present generic and widely adopted frameworks for image

retrieval and annotation. We conclude this chapter by presenting the problematics which have motivated this work and which are envisaged in the previously mentioned frameworks.

3.1 Searching and indexing image databases

Setting things to order for efficient retrieval is a preoccupation affecting many areas and which is rendered necessary by the permanent growing of databases, whether it is text, image, video databases ... Retrieval systems have been conceived in the purpose of assisting the user during his database exploration task and of guiding him towards relevant contents. This is a more and more crucial issue in many areas where the amount of information has become so huge that standard manual browsing has become impracticable, or, at least, much too long in contexts where fast access to information is necessary and an essential key to making the right decisions at the right time. Many methods have already been developed successfully to index information in the purpose of facilitating and accelerating searches. The most well-known example is perhaps with web search engines which index hundreds of millions of web pages, allowing users to find pertinent content in response to a textual request in less than 1 second. Keyword-based search such as used and implemented in web search engines remains the standard for most content mining commercial tools, regardless of the nature of data. Thus, for images, metadata under the form of side textual annotation is used to answer the user requests, leading often to imprecise or completely irrelevant answers. The observed lack of relevance could be explained by the fact that the human language is somewhat unadapted to account for the reality of things such as perceived by the human eye and it is often hard to characterize a scene using simply a few words. There is also the possibility that the side annotation is wrong or very imprecise. A second problem is that some kind of images such as those provided by remote-sensing sensors come with absolutely no annotations and the frequency of acquisition as well as the increased resolutions render the task of annotating these images simply impossible (example of QuickBird). In this context, we need efficient tools to learn automatically semantic from image data. This is specifically the problematic addressed by content-based image retrieval systems: organizing image databases by their visual content. This definition encompasses a wide range of systems, ranging from simple similarity-based retrieval systems to complex auto-annotation engines. Though "CBIR" is a kind of umbrella term designating a great variety of systems and a large panel of applications ranging from medicine to satellite imagery, we can identify a "common denominator" which is the reliance on visual similarity for assessing semantic similarity. This may prove problematic due to the semantic gap, that is, the fact that similarity at signal (i.e. descriptors) level does not necessarily mean similarity on a semantic point of view. All applications tackling directly or indirectly the problem of robust image understanding are thus specifically oriented towards bridging the semantic gap, that is, towards building a mapping function between low-level image descriptors and high-level semantic concepts which correspond to the human representation of reality. To achieve this task, CBIR systems are designed to solve two fundamental problems: **(a)** how to find a (good) mathematical description of the visual content of images and **(b)** how to assess the semantic similarity between two images

given their extracted mathematical representations (descriptors). A generic architecture of CBIR is presented in figure 3.1. Image signatures are first extracted from images most of the time as part of a preprocessing step. Signatures are generally obtained from the raw features by "organizing" them into vectors or distributions. Vectors are built by concatenating features of different natures (shape, color, texture ...) whereas distributions are obtained by computing histograms or by using region-based signatures. Both histograms and region-based signatures are weighting sets of vectors which can be normalized to form discrete distributions. A more complete overview of signature extraction is given in the next section, even though the building of the retrieval model in a CBIR system is often tightly related to the nature of extracted signatures and vice versa. The second step of CBIR generally consists in defining a measure of similarity at signature level. This is again strongly dependent on the type and nature of extracted signatures. Histogram signatures will benefit from the use of histogram distances, whereas region-based signatures may require the use of distances between distributions to plainly exploit the potential of discrete distributions used as signatures. Generally speaking, the choice of a similarity measure is motivated by the following factors:

- agreement with semantics, that is, how far similarity at signature level is repercuted at semantic level, or, in other words, to which extent the employed similarity measure allows to bridge the semantic gap.

- computational complexity, that is, the amount of computations required to evaluate the distance between two signatures. Some similarity measures are not suited for real time applications or cannot cope with large-scale databases

- robustness to noise and invariance to background

Among state-of-the-art CBIR systems relying on similarity measures between visual signatures, we can mention systems such as QBIC [Flickner et al., 1995], VisualSeek [Smith and Chang, 1996] and Photobook [Pentland et al., 1996]. In all these systems, signatures combining color, shape and textural information are extracted from images. The retrieval process is then entirely based on the notion of similarity to the user request, that is, the results are ordered according to the distance of their signature to the signature extracted from the query image. Some improvements to the above-mentioned basic matching scheme have been proposed from different prospects. The SIMPLIcity system [Wang et al., 2000], for instance, makes use of adaptive features: a preliminary categorization step separating images between textured/non-textured ... allows to automatically select the most appropriate set of features for each category before applying the matching scheme. In Zhang et al. [Zhang and Zhang, 2004], a region-based retrieval scheme is proposed which makes the assumption of the existence of an underlying generative model for regions in images. Images are first segmented into regions, then, using a vector quantization method, a region-based representation of the database is achieved. A probabilistic model based on a hidden-class assumption is computed on this representation, yielding a generative model of couples images/regions, the presence of a particular region inside an image being conditioned on a latent variable that reflects the underlying semantics. Instead of using a simple
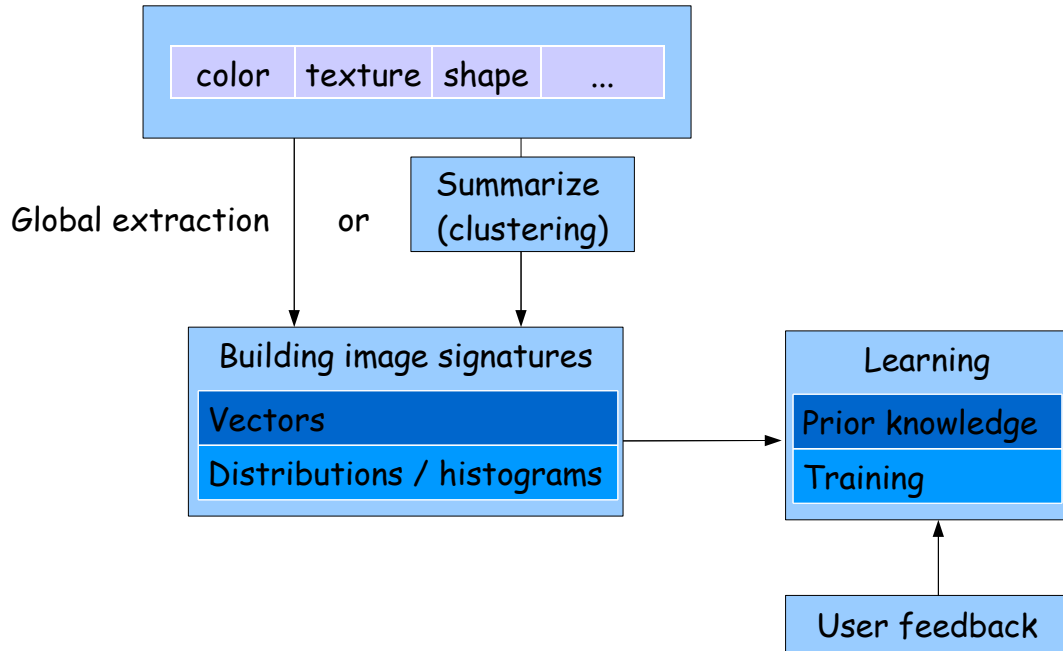
Figure 3.1: Paradigm of CBIR architecture (this diagram is directly inspired from the one in Datta et al. Datta et al. [2008])

matching criteria in the retrieval scheme, the system computes the posterior proba-bility of each discovered latent semantic concept given each image of the database. The same is done for the query image and, then, similarity is computed in the concept space using a cosine distance. The preceding approaches take the whole image into account when performing a query which might not be the best way to do things. In BLOBWORLD [Carson et al., 2002], the user is allowed to specify a portion of the im-age to be of interest: images are divided into blobs, that is, homogeneous color-texture segmented regions and, to perform a query, the user selects one or more blobs which, according to him, define the target class. A score of similarity is then defined between blobs and a fuzzy logic framework is used to handle compound queries consisting of several blobs. While this method allows a more precise formulation of user queries and a better understanding of the system responses, it may require to much involvement from the user on complex queries. A more recent approach in the domain of retrieval by similarity measures concerns the use of non-linear manifold to model the subspace on which image vectors lie. The standard Euclidean distance in the original linear fea-ture space is then replaced by a geodesic distance on a non-linear manifold.

It has been observed over time that making use of a simple similarity measure in the retrieval process wasn't sufficient to effectively bridge the semantic gap. More recent CBIR systems try to set the focus on the learning part to counterbalance the weakness of similarity measures when it comes to model semantics. Thus, machine learning techniques borrowed to the domains of clustering and classification have been suc-cessfully adapted to the problematics addressed by CBIR systems over the last few years. The learning component on the diagram of figure 3.1 has thus become the new area of investigation and up to now the most promising direction of improvement re-garding retrieval performance in CBIR systems.

Clustering techniques are useful in the unsupervised case, that is, when no labeled datas are available. Their purpose is generally to speed up the retrieval process in large-scale unstructured image repositories by precomputing an unsupervised index of the database. A query is then simply handled by returning the elements of the database having the same index as the query. Clustering can be performed on vector-based signatures but also on more complex signatures such as distributions: Li et al. [Li and Wang, 2003] for instance have proposed to extract region-based image signatures under the form of weighted sets of vectors and then to compute an index using a D2 clustering algorithm optimizing a functional similar to the one employed in K-means but working on discrete distributions instead of data points. The employed similarity measure in this system is the Mallows distance [Mallows, 1972].

Image classification techniques can be employed when a set of labeled training examples is available. They are used either for auto-annotation or for retrieval purposes. In the following, we make a distinction between generative and discriminative modeling approaches. Discriminative approaches try to estimate directly the posterior probabilities of classes or to build a classification boundary. Among well-known discriminative models, we can mention Support Vector Machines (SVMs) or nearest neighbors. Generative models attempt to estimate the joint probability of classes and elements to be classified. They require the computation of class-conditional probabilities and also the knowledge of a prior over each class. Gaussian Mixture Models (GMMS), mixture of experts, conditional random fields are examples of generative models. In the following, we briefly describe some systems falling under the category of classification-based CBIR systems.

Latent variables models rely on Latent Dirichlet Allocation (LDA [Blei et al., 2003]) to perform automatic annotation of image databases. The original LDA model is a generative model of text documents. Each document is considered to be a mixture of a small number of latent topics and the creation process of words inside this document is supposed to be explainable through these latent topics. Fei-Fei et al. [Fei-Fei and Perona, 2005] have proposed an extension of LDA to perform classification of image collections. In this extension, textual words have been replaced with visual words resulting from a vector quantization of the descriptor space. Thus, each image is considered to be a bag of visual words. Like in the LDA model for text, images are interpreted as mixtures of latent themes and the creation process of visual words inside images is conditioned on the themes present in the images. An application of this idea to satellite imagery has been proposed in Lienou et al. [Lienou et al., 2010].

Generative models are widely used in auto-annotation systems: Fan et al. [Fan et al., 2008] have proposed a generative hierarchical model which has some similarities to GMM-based systems. The gap between "atomic image concepts" (that is, the concepts with the smallest intra-concept variation on visual properties) and low-level descriptors is bridged through the use of an algorithm called Product of Mixture Experts (POM). This algorithm exploits a modeling of the underlying data density under the form of several GMMs (one per each kind of image descriptor: texture, color, shape, ...). "Experts" are then computed under the form of three-level associative models linking the atomic image concepts to each mixture component of each GMM through an intermediate level of co-appearance patterns (such as "sand and water", "water, tree and sky", ...). Thus, estimating the whole model parameters consists in estimating the

parameters of each GMM plus the probabilistic associations between Gaussian mixture components and co-appearance patterns and the probabilistic associations between co-appearance patterns and atomic image concepts. Barnard et al. [Barnard et al., 2003] have proposed an extension to the basic LDA idea developed by Blei under the form of a generative model of words and pictures. More precisely, images are segmented into regions and the notion of "document" such as introduced by Blei makes reference here to sets of image regions and words. The generative model is then constructed by considering that inside a document, words on one hand and image regions on the other hand can be explained through the intermediary of latent topics. Among other well-known auto-annotation systems, we can mention ALIPR [Li and Wang, 2003] (Automatic Linguistic Indexing of Pictures - Real Time). In this system, signatures under the form of color and texture distributions are extracted from images. Then, for each semantic concept, statistical models called "profiles" are learned in the space of signatures. To annotate a new image, the system starts by extracting a signature from this image and then compute its likelihood regarding each profile. Using Bayes law, it is then possible to compute the probability for an image to be associated with a concept of the training database, which defines a model for automatic annotation.

Classification-based retrieval systems suffer from several problems. In the unsupervised case, we have to estimate both the number of clusters and the clusters themselves, which often leads to poor indexation performance. The clusters indeed may not be visually consistent, that is, they may contain different semantic concepts. Or, on the contrary, clusters may not be representative of a semantic notion and just correspond to a signal class with no proper semantic associated with it. Even though the purpose of supervised classification methods is to remedy the two major drawbacks of unsupervised clustering techniques mentioned above, these methods are strongly dependent on the existence of comprehensive training sets, which is problematic in practice. User annotations are besides often very subjective, rendering more complex the problem of constituting a ground truth.

To cope with the non-existence of training sets (which is the rule rather than the exception), recent Content Based Image Retrieval (CBIR) systems make use of a "relevance feedback" feature to iteratively learn a model of the query target. The idea behind relevance feedback is to involve the user in the learning process by asking him to assess the relevance of the results which are fed back to him by the system at each iteration. The system then uses the information about whether or not theses results are relevant to update the current query and perform a new one. Through this user-system interaction component, systems based on relevance-feedback help the user to better characterize his search and do not require to have a training set at one's disposal to build the retrieval model. In the following, we only mention a few well-known state of the art systems making use of a relevance feedback component. A more comprehensive review is done in section 3.4.3. Some learning methods can be adapted very easily in a relevance feedback framework: Tong et al. have shown how SVMs can be used to learn iteratively a model of the target class and how they provide at the same time a natural way of determining which are the most informative examples to feed back to the user at each iteration of the feedback loop so as to maximize the transfer of information between the user and the system and thus to minimize the number of

iterations necessary to arrive at a suitable definition of the target class. The very simple idea exploited here consists in asking the user to assess the relevance on the most ambiguous images defined as the images whose corresponding signature is the closest from the SVM separating surface. The current most ambiguous elements are then included in the training set used to build the SVM classifier at the next iteration. This basic idea has been re-used in many recent CBIR systems. Costache et al. [Costache et al., 2006] have proposed an adaptation of this strategy in the context of remote sensing on both SAR and high-resolution optical satellite images. A more complex system, RETIN, has been proposed by Gosselin et al. [Gosselin et al., 2008] still relying on the initial idea of Chang et al. but with a supplementary strategy of user-system adaptation through a modification of the ranking function. The system proposed by Chang et al. [Tong and Chang, 2001] indeed yield good results on condition that the separating surface is already approximately well positioned, which is not the case at the beginning of the learning process. To alleviate this problem, an adaptive ranking function is proposed which do not rely on the SVM separating surface and thus is not subject to its fluctuations at the beginning of the learning process. Further improvements have been added to the basic scheme by making use of a transductive SVM and of a clustering in the signature space to ensure sparsity of the examples fed back to the user at each iteration of the learning loop. The system IKONA proposed by Boujemaa et al. [Boujemaa et al., 2001] allows the user to specify his request using a whole image or a region of an image and then enters a relevance feedback loop. An extra criterion for sparsity of examples displayed to the user has been added: instead of displaying most ambiguous elements as in Chang et al., the system looks for the most ambiguous most orthogonal, that is, the elements among the most ambiguous ones yielding the smallest pairwise scalar products in the high-dimensional SVM feature space. The system also includes a feature for mental image discovery which allows to find in a few iterations an image corresponding to what the user has in mind. This is particularly useful to solve the page zero problem which often occurs in relevance-feedback like systems: due to the absence of any training set, there are no example images associated with semantic concepts to initiate a request with. IKONA is a generic system working on different kind of image datas, ranging from generalist color image databases to databases of high-resolution optical satellite images.

A problematic which has hardly been explored but which is an emerging problem in the actual CBIR community is how to handle databases containing hundred of terabytes of images such as EO image databases. Much machine learning algorithms do not scale well to such high-volume databases. Among the very few ones which have been successfully implemented and deployed over very high-volume EO repositories in the purpose of collecting and analyzing data with minimal human intervention, we mention the GeoIRIS system [Shyu et al., 2007] which is a content-based high-resolution satellite imagery retrieval system supporting Query-By-Example (QBE) using either image regions (patches) or anthropogenic objects, Geospatial Queries, and Geospatial-enabled QBE queries. The KIM system [Datcu et al., 2002] is also a working solution deployed over very large remote sensing image databases. In this system, high-level user concepts are linked to low-level primitive features trough the use of a bayesian network.

3.2   Overview and discussion of basic image descriptors

Computing informative and concise mathematical representations/descriptors from images is one of the fundamental bricks of CBIR systems. In the following chapter, we give an overview of different kinds of image descriptors. Among the major types of image features, we can roughly distinguish between those describing color, texture and shape. The use of a particular kind of feature rather than another is determined by the nature of images inside the database on which we want to perform searches. Many approaches use combinations of different kind of features to try to exploit the potential of each feature regarding different visual aspects of the image. We discuss this particular issue at the end of this chapter.

3.2.1   Color information

Color is perhaps the most commonly used feature to describe the content of digital color images. The simplest expression of a color descriptor is obtained by calculating a color histogram in the original RGB space (Swain et al. [Swain and Ballard, 1991]) or by summarizing the corresponding discrete distribution with its first and second orders statistical moments (Stricker et al. [Stricker and Orengo, 1995]). These simple descriptors possess the advantage of being very fast to compute and also rotation and scale invariant, allowing to describe two images differing only by a geometric transformation with the same color descriptor. But these representations do not include any spatial information, which means that two different images can lead to the same color histogram. To remedy the lack of efficiency of color histograms in this prospect, color coherence vectors have been introduced by Pass et al. [Pass et al., 1997]. They start from the observation that color information in uniform color regions should not be matched against color information in regions with scattered pixel color values. They propose thus to build two separate histograms for coherent and incoherent pixels respectively. Still with the idea of integrating spatial information within color descriptors, Huang et al. [Huang et al., 1997] have proposed to use color correlograms which analyze the correlations between pairs of pixels at different distance steps. Vertan et al. [Vertan and Boujemaa, 2000] have proposed the use of weighted color histograms which embed local information about the statistical and visual relevance or importance of each pixel. In practice, the authors propose to adaptively weight each pixel contribution into the color distribution. They suggest several weighting schemes based respectively on the edge strength (Laplacian) and on probabilistic and fuzzy measures such as entropy and fuzzy entropy.

Another approach consists in envisaging color space transformations. The Lab space is generally considered to better coincide with human perception than the RGB space and to be perceptually uniform, i.e., distances between color triplets in that space directly reflect perceptual color differences. Put in other words, the perceived amount of difference between two colors will be directly related to the distance between these two colors in the Lab color space. A set of color descriptors using such transformations is described within the MPEG-7 standard [Manjunath et al., 2002], even though the "power" and the attractiveness of the MPEG-7 color descriptor relies more on a compact representation of the color information than on the color space transformation itself. A special emphasis has also been set on features which remain

invariant under varying imaging conditions. Gevers et al. [Gevers and Smeulders, 2000] have proposed to extract color models independent of objects geometry, pose and illumination. Then, from these models, "color invariant edges" are derived which allow in turn the computation of shape invariant features. In the same vein, De Weijer et al. [Van De Weijer et al., 2006] have described a robust photometric invariant feature based on the computation of a color tensor describing the local differential structure of images.

In panchromatic optical satellite imagery, color descriptors rely on a single channel representing the satellite-measured radiometry. Most descriptors described above can be adapted on grey level images. Specific descriptors are also employed such as the Weber Local Descriptors (WLD) proposed by Chen et al. [Chen et al., 2009]. These descriptors rely on a perceptual law, the Weber law, which relates the physical magnitude of a stimuli to its perceived intensity. This law states that the smallest noticeable difference on a perceptual point of view is proportional to the intensity level of the background (a whisper will thus be easily perceived in a quiet environment whereas, in a noisy place, speaking loudly is necessary to be heard). The idea underlying Weber's law is used in WLD descriptors to compute in each pixel of the image a quantity called differential excitation and which is defined to be the ratio of the value of the pixel over the value of the background (the latter being computed as the mean value over a small neighborhood around the current pixel). An orientation component is also computed in each pixel as the gradient orientation. The intensity and orientation informations are then combined into a joint histogram which is then reshaped into a 1D histogram. Thus, both the intensity and the orientation information are captured by the descriptor, the difference with the commonly used gray level-gradient orientation two-dimensional histogram being that the WLD descriptor is insensitive to changes in radiometry. This is due to the fact that we do not look directly at the intensity levels but at the amplitude of the changes in intensity relatively to the background intensity level. Thus, WLD histograms are particularly well adapted to high-resolution optical satellite imagery where insensitiveness of descriptors to radiometry is a crucial point due to extremely varying imaging conditions (which cause in turn important differences in brightness between different scenes taken by the same sensor, considerably complexifying the task of retrieving similar objects/categories across different scenes).

Color information extraction from multispectral images requires the use of specific techniques. The simple fact that multispectral images possess more than three bands (Landsat has 7) which are besides not all located in the visible light spectrum renders common color-space transforms inapplicable. The data acquisition techniques used in multispectral imagery are indeed quite different from those used in common camera devices: in multispectral imagery, acquisition is performed by radiometers which measure the radiant flux (power) of electromagnetic radiation whereas the CCD of a standard digital camera measures the light intensity. Multispectral image analysis is usually done by exploiting different spectral band combinations: the green-red-infrared combination is used to detect vegetation and camouflage, vegetation being highly reflective in the near infrared. The well known Normalized Difference Vegetation Index (NDVI [Kriegler et al., 1969]) is a numerical indicator computed on the red and infrared bands which allows to say whether or not the observed target contains live green vegetation (NDVI is mainly sensitive to chlorophyll). The enhanced vegetation index (EVI)

is an other vegetation spectral index which is this time responsive to canopy structural variations, canopy type and plant physiognomy. The blue-nearIR-midIR combination is used to observe the water depth, the soil moisture content and the eventual presence of fires. There exists many other combinations in use. Baraldi et al. [Baraldi et al., 2006] have proposed a "fuzzy rule-based per-pixel classifier" for preliminary classification of Landsat multispectral images directly exploiting spectral band values in the classification process. Their algorithm uses kernel fuzzy spectral rules implemented as logical expressions composed of scalar variables which are combined with relational and logical operators (the scalar variables being directly the Landsat spectral band numerical values). The system is able to detect a set of "low-level" land cover classes such as "Evergreen forest", "Clearcut", "Bare-soil cropland", "Clouds", "Perennial snow" ... thus achieving a direct mapping between the spectral signature and (basic) image semantics.

3.2.2　Textural information

An image texture is defined as a set of pixels spatially organized according to several spatial relationships and forming an homogeneous region inside an image. Texture features are used to capture granularity and to characterize repetitive patterns inside images. Their role is particularly useful in optical satellite imagery due to their direct relation to image semantics: classes such as forest, grassland and urban areas can be very well distinguished between each other using simple texture attributes. The figure 3.2 shows some typical texture images extracted from different QuickBird scenes.

Haralick et al. [Haralick et al., 1973] have proposed to use quantities computed from grey level co-occurrence matrices to characterize the image structure. Co-occurrence matrices correspond to second order statistics and represent the distribution of co-occurring pixel values at a given offset. The fourteen Haralick coefficients are statistical measures such as entropy, homogeneity, up to fifth order statistical moments ... computed on these matrices. More recent approaches rely on transforms to extract textural information, mainly transforms in the spatial-frequency domain since the purpose is to capture both the frequency and the orientation of repetitive patterns which constitute a textured area. The first to exploit this principle were Manjunath et al. [Manjunath and Ma, 1996] who introduced an extension of Gabor filters to two-dimensional signals. Gabor filters allow to capture the spectral energy distribution in different frequency domains and in different directions. The corresponding texture descriptors are generally obtained by applying a bank of filters working each in a particular direction and in a particular frequency domain and by computing first and second order statistics on their outputs. The same idea is applied in Randen et al. [Randen and Husoy, 1994] using this time quadrature mirror filters to perform a wavelet transform on the input image. First and second order statistics are then computed on each sub-band of the wavelet decomposition and are concatenated in a single feature vector which serves as the final texture descriptor. Instead of keeping only statistical moments computed from wavelet coefficients, Do et al. [Do and Vetterli, 2002] have proposed to model independently wavelet coefficient in each sub-band by using Generalized Gaussian Densities, arguing that it provides a natural and normalization-free framework for retrieval thanks to the existence of an analytical expression of the Kull-
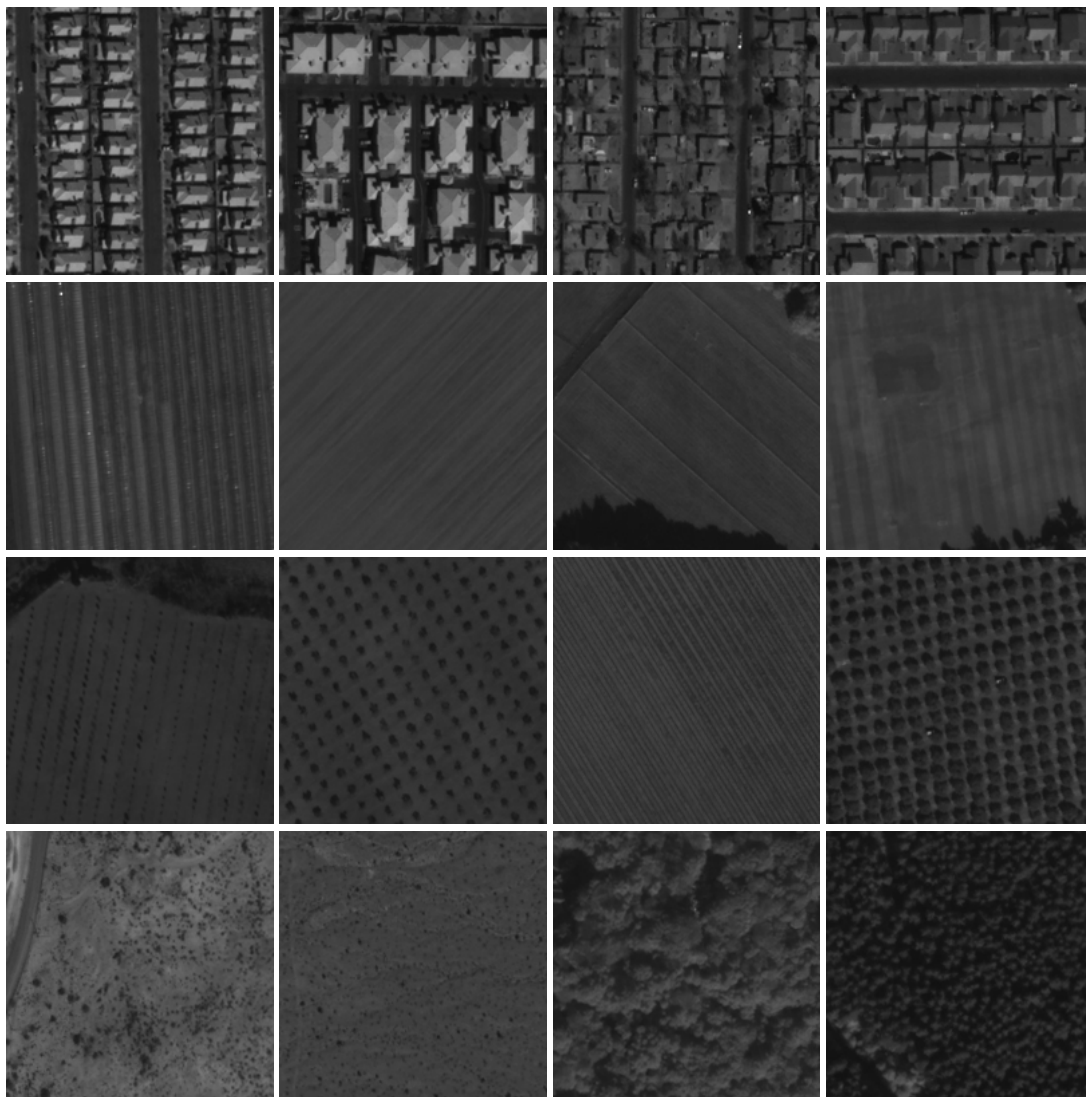
Figure 3.2: QuickBird texture images corresponding to different image classes: urban area (first row), fields (second and third row), desert and forest (fourth row) ©DigitalGlobe.

back Leibler divergence for such distributions.

Building texture features invariant to geometric and photometric transformations is desirable for many applications such as segmentation of natural scenes, recognition of materials ... In this prospect, Lazebnik et al. [Lazebnik et al., 2003] have proposed a representation of textures based on a sparse set of regions selected using a Laplacian blob detector and from which an affine-invariant descriptor is extracted. A distribution of textures inside each class is learned in a supervised way using a GMM in which each mixture component corresponds to a sub-class. This allows to define a generative model linking the sub-class labels to the extracted regions. This local representation of textures is then augmented with a description of the spatial relationships between neighboring regions through the use of a supervised sub-class co-occurring model. The obtained framework allows to perform retrieval and segmentation of multi-texture images through the combination of both the generative and the co-occurring model. The same idea of interest point detection for sparsity has been exploited in Mikolajczyk et al. [Mikolajczyk and Schmid, 2004] where a Harris detector is used to locate highly textured points and a method based on the derivatives of the Laplacian allows to compute the scale and the affine shape of the corresponding local structures. Affine- and scale-invariant texture information can then be extracted from these local structures/regions.

In parallel with the above mentioned approaches, models based on Gibbs-Markov Random Fields (GMRF) have begun to emerge. In this kind of models, an image is considered to be the realization of a stochastic process but with an added memoryless (Markov) property i.e. the conditional probability of a pixel given the entire image is equal to the conditional probability of the pixel given a local neighborhood, or in other words, the value of a pixel will entirely depend on the knowledge of the pixel values in a neighboring area. GMRF-based texture modeling consists generally in estimating the parameters of the Gibbs distribution associated with the Markov Random Field. Cross et al. [Cross and Jain, 1983] were the first to use GMRF to model spatial dependencies inside textures. GMRF have then been applied successfully to many remote sensing applications. For instance, Datcu et al. [Datcu et al., 1998] have presented a method to extract structural information from remote sensing images where GMRF are used as priors model in a Bayesian inference framework. Schröder et al. [Schröder et al., 2000b] have proposed to use auto-binomial models (which are a very simple form of GMRF for texture analysis involving only first and second order neighborhoods/cliques) in the domain of optical satellite imagery. Auto-binomial models have received particular attention due to the fact that they possess less parameters but still allow very good performance in texture classification.

Another unrelated but very popular texture descriptor is Tamura's texture feature introduced by Tamura et al. [Tamura et al., 1978] which contains six features selected via psychological experiments and which capture the high-level perceptual attributes of a texture (coarseness, contrast, directionality, linelikeness, regularity, roughness). These very simple features have been shown to perform very well on generalist databases. Deselaers et al. [Deselaers et al., 2008] have noticed that in some cases, they even outperform traditional texture features such as first and second order statistics on Gabor filter banks outputs. The QBIC system makes use of these features.

3.2.3 Geometrical information

During the past few years, shape has been considered as a key attribute in image retrieval because of the existence of robust and efficient representations of the corresponding information. Besides, shape of object is often closely related to their functionality and identity: humans can recognize objects solely from their shapes, which illustrates the fact that shape is strongly linked to image semantics, much more than color or texture attributes which, taken alone, generally do not allow to identify an object.

Shape descriptors have been employed in many CBIR systems, including earliest ones such as QBIC which makes use of simple geometrical features (shape area, circularity, eccentricity, major axis orientation and a set of algebraic moment invariants) to describe image content. In the QBIC system, the object shape delineation is done through a semi-automatic process where the user is asked to enter an approximation of the object outline which is then aligned with the nearby edges using this time an automatic active contours technique. The obvious drawback of such an approach is the high user involvement in the feature extraction process.

A lot of shape descriptors require the extraction of edges as a preprocessing step. This is the case of edge orientation histograms which characterize the spatial distribution of edges in an image. In the MPEG-7 standard, images are subdivided to allow the computation of local edge distributions. Local orientation histograms are then generated from each image subdivision by mapping their edges into five bins: horizontal, vertical, 45diagonal, 135diagonal and nondirectional edges. The resulting local histograms are concatenated to form the final edge orientation histogram. In the same vein, Histogram of Oriented Gradients (HOG) count occurrences of gradient orientations in an image. The technique differs from edge orientation histogram in that it consists in computing local histograms on a grid of uniformly spaced and possibly overlapping cells, using local contrast normalization over overlapping cell blocks for improved accuracy. This descriptor was first described in Dalal et al. [Dalal and Triggs, 2005] and was originally designed to perform human detection. It has been later proved to work successfully also on generic visual object recognition problems (Dalal et al.). Ferecatu et al. [Ferecatu, 2005] have proposed a modified edge orientation histogram based on the Hough transform: the gradient direction is first computed in each pixel of the image. Then, a joint histogram for the gradient direction and the distance of a reference point (for instance the upper left corner of the image) to the tangent line going through the current pixel is built.

Contour-based and region-based shape descriptors intended to be used in a complementary way are described in the MPEG-7 standard. Region-based descriptors express the pixels distribution within an object or a region and can describe complex objects composed of multiple disconnected regions as well as holes. The descriptor works by decomposing a shape into a number of 2D orthogonal complex-valued basis function using a complex 2D Angular Radial Transform introduced by .... The normalized coefficients assigned to each basis function are then used to define the shape. As for the the contour shape descriptor, it is based on the Curvature Scale-Space (CSS) representation of the contours. This representation was introduced by ... and starts from the observation that, when comparing shapes, humans tend to decompose contours into convex and concave sections. The CSS representation thus decomposes objects into

convex and concave parts by determining the inflection points. The multi-scale aspect comes from the fact that contour representations are computed at different scales. A CSS image is finally obtained representing the position of the inflection points on the contour as a function of the scale. The extracted descriptor consists of the number of peaks on the CSS image, the magnitude (height) of the largest peak and the *x, y* positions of the remaining peaks.

In a completely different prospect, object-oriented shape descriptors using mathematical morphology have been proposed in Pesaresi et al. [Pesaresi and Benediktsson, 2001]. Their approach relies on Differential Morphological Profiles (DMP) which allow to perform a multi-scale analysis of an image and to extract an histogram of object sizes from it. The multi-scale analysis consists in performing successive erosions followed each time by a morphological reconstruction. The staged erosion process eliminates objects whose size is inferior to the current value of the scale. Other objects are kept unchanged due to the reconstruction part. By subtracting images from consecutive scales, we can then determine at which scale an object disappears (the difference being null for objects still present in the image because of the reconstruction which leaves them unimpaired despite the successive erosions) and subsequently the scale of the object.

3.2.4   Building image signatures

Building image signatures from descriptors is not so trivial as one might think. There exists of course the simple approach which consists in using directly the extracted descriptor vector as the image signature but this is not always the smartest way to do things. As mentioned in section ..., image signatures can roughly be divided into two categories: vectors and discrete distributions. The latter are generally obtained by computing histograms or a set of weighted vectors from images. As mentioned by Rubner et al., histograms can be thought as discrete distributions by considering that each bin is a couple containing the bin location and the associated frequency. Distances between distributions such as the Earth Mover Distance (EMD [Rubner et al., 1998]) can then be used on histograms as well.

Region-based signatures also belong to the class of discrete distributions but proceed from a different extraction process. Generally, a clustering is performed on all the database image pixels set together, leading to the creation of a codebook containing a certain number of visual words (codewords). The signature associated with an image is then an histogram of visual words, each bin of the histogram counting the number of pixels in the image which are the closest to the corresponding codeword. This approach can be further refined by exploiting sparse-coding like approaches: instead of letting a pixel vote for only one histogram bin, we can make it vote for all the bins by considering the distances of the pixel to all the bins. Weights proportional to the inverse squared pixel/codewords distances can be used as well as more elaborated weighting schemes using sparse-coding (Liu et al. [Liu et al., 2009]). They are besides plenty ways of building the codebook, taking the pixel values being the simplest one. We could think for instance of extracting color, texture, shape ... descriptors on small neighborhoods around each pixel and then of performing a clustering on the set of obtained descriptors to extract the codebook. In the RETIN system, the visual feature

extracted from a pixel (x, y) is its value in the $L^\star a^\star b^\star$ color space concatenated to the output at the same (x, y) coordinates of 12 complex Gabor filters in three scales and four orientations. The codebook is then computed on these visual features using an ELBG algorithm.

3.2.5   Combining information of different nature

In this section, we focus on unsupervised ways to fuse image descriptors of different natures. The most naive way of proceeding would consist in simply concatenating after a normalization and weighting step the extracted descriptors of color, texture, shape ... to form a signature vector and then use a metric provided by the L1 or L2 distance (Barnard et al.). The problem in doing so is that we do not keep the properties of descriptors such as histograms on which much more effective distances can be computed. A widely spread technique for incorporating information from different nature without sacrificing the histogram arrangement is to use joint (multidimensional) histograms. This issue has been extensively studied in Pass et al. [Pass and Zabih, 1999] where combinations of different features (color, edge density, texturedness, gradient magnitude, rank (Zabih et al.)) into joint histograms are tested on a generalist database of color images. Unsurprisingly, the more features are incorporated, the highest the performance is, but often at the expense of extremely large-size and sparse histograms, which in turn brings the problem of efficient histogram matching. Some works address specifically this problem: in Jou et al. [Jou et al., 2004], an histogram matching algorithm unrelated to the histogram sizes is proposed which allows the use of common matching schemes such as histogram intersection, χ^2 distance ... without causing an increase in computational complexity as the size/number of dimensions of the histograms augments.

Rather than fusing image descriptors at feature level right after the feature generation process, one could think of combining descriptors in an unsupervised learning framework. A state of the art approach using this idea is that of Quelhas et al. [Quelhas and Odobez, 2006] who have proposed a fusion strategy to combine color and texture information in a bag-of-visual-words (BOV) representation. In this approach, texture and color descriptors are first computed on small regions around interest points detected using a Difference of Gaussians local interest point detector. Two separate codebooks/vocabularies are then extracted from color and texture descriptors respectively, allowing to compute BOV representations of images under the form of histograms of visual words. The two obtained BOV models are then fused by concatenating the corresponding histograms h_C and h_T with the use of a mixing coefficient α, yielding a final signature under the form $(\alpha h_V, (1 - \alpha) h_T)$. More recently, Hörster et al. [Hörster and Lienhart, 2007] have described three unsupervised fusion techniques based on a latent topic image modeling approach (i.e. images are considered to be random mixtures over latent topics). Latent Dirichlet Allocation (LDA) is used to compute the hidden/latent topic model in the three scenarios. The fusion process yields an expression of the likelihood of every possible visual words combination given an image model. The first scenario consists in extracting one LDA model per type of feature. Given an image, the fusion is then carried out at decision level by multiplying the likelihood distributions of each visual word composing the image given the obtained image model

(there exists on likelihood model per feature type). The second scenario fuses the feature types at the visual words level by assuming a joint observation of each kind of visual word. A joint distribution over all kinds of visual word for each topic is thus employed in the LDA model to relate images to latent topics. Given an image, the fusion is performed by multiplying the likelihood distributions of all "joint visual words" composing the image given the obtained image model. In the third scenario, fusion is carried out during topic generation. In this model, topics represent visual words extracted from only one feature type, yielding a model containing one hidden variable per feature type, but contrary to the fist scenario, the topic model remains the same for all the hidden variables. Fusion is performed as in the first scenario by multiplying the likelihood distributions of each visual word composing the image given the obtained image model, the difference being this time that there exists only one single likelihood model for all the feature types.

Many approaches in CBIR systems makes use of a feature combination process but most of the time through the use of a supervised learning approach. A typical way to do so is to use boosting approaches. In Yin et al. [Yin et al., 2005], for instance, a boosting algorithm is proposed where one weak classifier per feature set is trained. The outputs of the obtained classifiers are then combined through weighted voting inside the boosting framework. But we will not develop further the progresses made in the field of supervised feature combination since our purpose in this section was to give an overview of unsupervised methods to extract and deal with features.

3.3   Attribute selection vs. dimensionality reduction

The features extracted from images are vectors with possibly a very high number of dimensions. This may cause several problems. The first one and the easiest to understand is the extra-computational burden generated by the adding of new dimensions to the description space. Since the image retrieval process is based on the computation of similarity measures between descriptors extracted from images, it is suitable at least for real-time applications to restrain the size of the feature vector to as few dimensions as possible.

Another fundamental motivation for feature selection is the curse of dimensionality, that is, the exponential increase of the volume of the description space (that we can roughly model as a hyper-rectangle) with the adding of extra dimensions. This is problematic in many machine learning problems where we try to learn the underlying distribution of data from a finite number of data samples. Basic retrieval techniques such as nearest neighbors search may also fail due to the fact that for equal sample sizes, the probability of two samples to be close goes down very quickly as the dimension of space increases.

The other problem with maintaining a large number of dimensions is the volume of the hypothesis space, i.e., the space which contains all the combinations of hypotheses that can be learned from the data. This is even worse than the curse of dimensionality: for N binary features, the hypothesis space will be of size 2^{2^N} ! Selecting a subset of feature makes it easier to find a correct hypothesis.

The process of feature selection consists in removing the irrelevant and redundant features. A feature is considered irrelevant when it can be removed without impact-

ing the learning. Redundant features can also be considered as a kind of irrelevant features: the only difference is that two redundant features are relevant taken separately but one of them can be removed without affecting the learning performance. Most feature selection techniques in the literature are directly inspired from the idea of maximizing the relevance while at the same time minimizing the redundancy. They are mainly variations in the way of formalizing the ideas of relevance and redundancy: ... We observe two main strategies regarding the feature selection process: The first one consists in ranking the features according to some criterion and then select the k top-ranked features.

The second one consists in selecting a subset of k features where k is the minimum number for which no performance deterioration is observed. The advantage of this second class of methods is that they allow to determine the number of features to select.

We can further distinguish between techniques using filter and wrapper approaches. Wrapper techniques consist in embedding a learning algorithm in the feature selection process and to use its performance on different feature subsets to determine which group of attributes to select. Most supervised selction algorithms are based on the idea of class separation: the interesting features are those which contribute the most to separating data from different classes, the other can be removed. The most well-known wrapper approach and which is a direct application of the class-based separation principle mentioned previously is the SVM-RFE (for Recursive Feature Elimination) method proposed by Guyon et al. [Guyon et al., 2002]. This algorithm achieves a ranking of the features using a criterion which is the margin size of a linear SVM classifier trained on a set of labeled data. The SVM-RFE algorithm performs sequential backward selection by starting with all the features and sequentially removing the one which yields the smallest value of the criterion. The features are then ranked according to their order of deletion in the sequential process. Peng et al. [Peng et al., 2005] have proposed a method which relies on a maximal statistical dependency criterion based on mutual information. The idea is to find the set of features S which has the largest dependency on the target class c, the dependency being defined here as the mutual information $I(S, c)$ between S and c. The max-dependency problem being not directly tractable, an approximation of this idea is described under the form of max-relevance-min-redundancy (mRMR) scheme. The mRMR criterion is then embedded in a wrapper which allows to select the right number of attribute using sequential forward selection (i.e. new features are added sequentially using the mRMR criterion until the classification error begins to decrease).

Dimensionality reduction refers mainly to unsupervised techniques, that is, techniques able to deal with unlabeled data. Contrary to feature selection methods, they do not necessarily keep the initial attributes as such but can also perform transformations of the initial description space. Principal component analysis is perhaps the most illustrative space transforming technique which is also used for dimensionality reduction purposes. It amounts to finding the principal directions defined as the directions of space along which the variance of the data is maximum. The underlying transformation is simply a rotation of the original orthogonal system of axis. Dimensionality reduction properly speaking is then achieved by selecting the first k principal components ordered by the percentage of the total variance they "explain". Indepen-

dent Component Analysis (ICA) tries to find a transformation of the space such that the obtained transformed variables are statistically independent. It is closely related to the problem of source separation. One way to achieve such a goal is to maximize the "non-gaussianity" of the linear mixing of random variables corresponding to each estimated source. By doing so, we ensure that the observed signal corresponding to the mixed sources will be much closer to the Gaussian density than each source taken separately. We are thus in the "frame" of the central limit theorem which allows to say that the gaussianity of a sum/mixing of variables/sources is a measure of the statistical independence between these variables/sources. Such an approach is employed for instance in FastICA (Hyvärinen et al. [Hyvarinen, 1999]).

We may wish to use techniques which perform "true" feature selection rather than space transformations to keep the original meaning of features. Many unsupervised feature selection methods which try to do so are closely related to unsupervised learning techniques. Supervised methods generally maximize a function of prediction accuracy, naturally selecting the features which lead to the predefined classes in the training set. The problem is much more difficult in the unsupervised paradigm where we are not given any class labels. Many unsupervised feature selection techniques rely on the quality of an unsupervised clustering of the data, trying to find "the smallest subset of features that best uncovers interesting natural groupings/clusters from data" (Dy et al.). The adjective "interesting" is generally formalized into a criterion or an objective function such as the sum of intraclass variances which is the functional optimized by the K-means clustering algorithm. We briefly review some generic clustering algorithms in section 3.4.1. Unsupervised feature selection consists of two main steps: (1) identifying a subset of relevant features and (2) assessing a measure of relevance of the selected subset. Like in the supervised case, we can make a distinction between wrapper and filter approaches. Wrapper methods consists of a (1) search component, (2) a clustering part and (3) an evaluation criterion. So a wide variety of unsupervised feature selection algorithms can be built by simply mixing these components, regardless of the techniques which are used for each one. As far as the search component is concerned, Narendra et al. [Narendra and Fukunaga, 1977] have proposed a branch and bound algorithm which allows to search for the best subspace among 2^N possible subspaces (where N is the initial dimension of space). But this approach becomes quickly impracticable as the number of features grows. To cope with this, greedy approaches have been introduced. Marill et al. [Marill and Green, 1963] have proposed a sequential backward selection algorithm which begins with all the features and iteratively discards the feature among the remaining ones which contributes the less to the criterion. Whitney et al. [Whitney, 1971] have introduced a sequential forward selection algorithm which starts with an empty set of feature and iteratively add one at each iteration of the algorithm. The problem with the two above-mentioned approaches is that once a feature has been added or deleted, one cannot delete or reselect it. To remove this drawback, Stearns et al. [Stearns, 1976] have proposed a method called Plus-l-Minus-r which involves at each iteration the augmentation of the current subset of features by l new features followed by a depletion of r features, the number of selected features at each iteration being thus $l - r$. Pudil et al. [Pudil et al., 1994] have introduced a modified version of this algorithm by proposing to adjust automatically the values of l and r at each iteration. Their algorithm is called sequential forward

floating selection. In an other prospect, random search methods based for instance on genetic and random mutation hill climbing algorithms have also been proposed in the literature.

Filter methods refer to methods which are independent of any learning algorithm. Most of filter techniques are space transforming methods such as the two cited previously. We can still mention the approach of Dash et al. [Dash et al., 2002] which is based on the entropy of distances between data points. They relied on the observation that when the data is clustered, the entropy of distances is lower. He et al. [He et al., 2006] have proposed a method based on the Laplacian Score which is used to evaluate the locality preserving power of the selected features. The underlying idea behind this approach is that two data points belonging to the same class should be close from each each other. Starting from this idea, a "good" feature is then a feature which preserves this locality property.

Other simple unsupervised feature selection schemes can be envisaged: in Campedel et al. [Campedel et al., 2008], a K-means algorithm is applied in the space of attributes. The feature selection method then boils down to keeping the attributes which are the closest to the obtained cluster centroids and to discarding the others. The same strategy can be applied using a kernel K-means clustering or a Support Vector Clustering (Ben-Hur et al. [Ben-Hur et al., 2002]) algorithm.

Another approach in feature selection which belongs neither to the class of supervised nor to the one of unsupervised methods is to embed the feature selection process into an active learning scheme.

3.4 Learning paradigms for content-based image retrieval systems

It is not our purpose to be exhaustive regarding learning paradigms. A complete review of learning methods has been done for instance in Hastie et al. [Hastie et al., 2005]. We content ourselves with giving an overview of some techniques which have proved useful in the following of this work. The presented methods fall each under one of the three possible scenarios, that is, the unsupervised, supervised and semi-supervised learning scenarios.

3.4.1 Unsupervised, Supervised and Semi-supervised learning models

Unsupervised learning models Unsupervised learning techniques generally refer to clustering algorithms. They are useful when it comes to handling large amounts of unlabeled data. They can roughly be divided into three types: optimization of an overall objective function or quality measure, pairwise-distance-based and statistical modeling. The first category covers fundamental techniques which have been explored since the early days of machine learning. The most famous one is the K-means algorithm ([MacQueen et al., 1967]) which measures the quality of a clustering as the sum of intra-cluster distortions, i.e. the sum of the distances of each data point to

the centroid of the cluster it belongs to. The drawback of this approach is that it requires to specify the number of clusters in advance. A simple approach to assess the number of clusters is to gradually increase it until the average distance between a data point and its cluster centroid is below a certain threshold. Patané et al. [Patané and Russo, 2001] have proposed a modified version of the standard K-means algorithm called the Enhanced LBG (ELBG). This algorithm minimizes the same functional as the standard K-means algorithm but is endowed with an heuristic to avoid convergence towards poor local minima. The underlying heuristic consists, roughly speaking, in equalizing the intra-cluster distortions, the aim being to obtain a final codebook in which each codeword/cluster contributes equally to the total distortion (i.e. to the sum of the intra-cluster distortions). The main operation to achieve such a goal is the merging of pairs of neighboring low-distortion clusters in parallel with the splitting of high-distortion clusters to preserve the total number of codewords. The ELBG algorithm is besides much faster (around 4 times) than the K-means algorithm since it requires in average much less iterations to converge for the same computational complexity at each iteration. A "fuzzified" version of the preceding algorithm has been proposed in Bezdek et al. [Bezdek et al., 1984] under the name of Fuzzy C-means (FCM). The objective function is the sum of the weighted square distances of each data point x_i to the cluster centroids β_j: $F_{obj}(K) = \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ij}^2 d^2(x_i, \beta_j)$. The weights u_{ij} are generally computed as the inverse distances between data points and cluster centroids normalized over all clusters, i.e., $u_{ij}^2 = \frac{1/d(x_i, \beta_j)}{\sum_{l=1}^{K} 1/d(x_i, \beta_l)}$. A competitive agglomeration algorithm has been presented in Saux et al. which does not require to specify the number of clusters before running the FCM algorithm. A term is added to the objective function through the use of a Lagrangian multiplier α to penalize complex models with a large number of clusters, leading to the following objective function: $F_{obj2}(K, \alpha) = F_{obj}(K) - \alpha \sum_{i=1}^{N} \sum_{j=1}^{K} u_{ij}^2$. In an information theory perspective, Tishby et al. [Tishby et al., 2000] have proposed a clustering algorithm based on the Information Bottleneck (IB) principle. The information bottleneck principle amounts to modeling processes such as clustering as the transmission of a coded signal over a communication channel: when transmitting a signal, we want to achieve the maximum compression ratio of the original signal X while at the same time minimizing the distortion of the retrieved signal Y. Applied to a clustering problem and using the preceding notations, it boils down to solving the following problem: $\arg\min_{p(\beta_j|x_i)} I(X, \beta) - \gamma I(Y, \beta)$, that is minimizing the mutual information between the original and the coded signal (in order to achieve maximum compression) and at the same time maximizing the mutual information between the coded and the retrieved signal (in order to retain as much as possible of the initial information). The term γ betrays the fact that the IB principle tries to achieve a compromise between compression and accuracy. In a supervised setting, the retrieved relevant information Y is generally the class labels found in the training database. An adaptation of the IB principle to unsupervised clustering has been proposed in Goldberger et al. [Goldberger et al., 2002] who describe an algorithm to cluster images represented by their extracted feature vector (a color histogram in their case). The original signal X corresponds to the raw images whereas the retrieved relevant information Y is considered to be the extracted feature vectors. The intermediary variables β are the underlying clusters that we seek to discover. Kernel methods have also made their apparition in the domain of unsupervised clustering. The ker-

nelized version of the K-means algorithm consists in simply replacing the standard euclidean dot product by a kernel dot product. The objective function is the same as in standard K-means. Dhillon et al. [Dhillon et al., 2004] have showed that the weighted kernel K-means algorithm is tightly related to the spectral clustering and normalized cuts algorithms. In another prospect, the Support Vector Clustering algorithm introduced by Ben-Hur et al. [Ben-Hur et al., 2002] reformulates the problem of clustering as a constrained quadratic convex optimization problem. This algorithm performs a mapping of the data into a high-dimensional feature space as in commonly done in kernel methods through the use of a kernel function. It looks then for the minimum enclosing sphere in the high-dimensional space. By projecting the obtained sphere back to the original data space, we obtain a set of contours delineating groups of data points. Clusters are then formed by considering that elements inside a contour belong to the same cluster. The idea behind the SVC algorithm is that the obtained support vectors yield a surface delineating the underlying data distribution. The algorithm can thus be seen as identifying valleys in the underlying data probability distribution.

The frontier between methods performing statistical modeling and those optimizing an objective function is not always very clear. The information bottleneck is an example of the lack of separation between the two categories. Let's say that statistical-modeling-based clustering methods generally treat every cluster as a pattern characterized by a simple distribution (for example the Gaussian distribution). One obvious advantage of the statistical modeling approach is that it not only provides a partition of the data but also an estimated density, which can be desired for some applications. The baseline in this area is represented by Gaussian mixture modeling which amounts to modeling the data as a mixture of Gaussians. The parameters of the statistical model, that is, the means and covariance matrices of the Gaussian components, which correspond respectively to the locations and shapes of clusters, are then obtained via an Expectation-Maximization (EM) algorithm. To prevent the EM algorithm from falling into a local optimum, Ueda et al. have introduced a variant using deterministic annealing. Their DAEM algorithm estimates the parameters of the Gaussian mixture through the optimization of modified likelihood objective function. The proposed objective consists of an energy term E modeling the distortion and an entropy term H modeling the rate. A DAEM algorithm replaced in the framework of the rate-distortion theory has been described in Rose et al. [Rose, 1998]. A more specific description of these algorithms is done in the following chapter.

Pairwise-distance-based methods require the computation of distances between every pair of data points and thus have a complexity of at least $O(N^2)$. There is again not a clear distinction between this category of methods and methods optimizing an objective function. Spectral clustering is perhaps the most representative example. Linkage clustering methods can also be placed under this category.

Supervised learning models   When labeled data are available for training, supervised learning paradigms have proved to be much more efficient than unsupervised ones. Supervised classification methods allow the partition of image databases into semantic categories in the purpose of retrieval or can be used to perform automatic annotation as well. We focus here on the first type of methods. They can be divided into two very broad though distinct categories: generative and discriminative approaches.

There also exists hybrid generative-discriminative methods. As said in section 3.1, generative approaches model the density of data within each class and then, the Bayes formula is applied to compute the posterior. The most representative example of a generative model is perhaps the Latent Dirichlet Allocation (LDA) model which we have already shortly described in section 3.1. Prior to the apparition of the LDA model, we can mention Probabilistic Latent Semantic Analysis proposed by Hofmann et al. [Hofmann, 1999]. PLSA is a latent variable model associating unobserved class variables z with each observation. The probabilistic model consists in expressing a joint probability of words w and documents d by considering that each co-occurrence (w,d) is a mixture of conditionally independent multinomial distributions, yielding the following model of the joint probability: $p(w,d) = \sum_z p(z)p(d|z)p(w|z)$. In this symmetric formulation, w and d are generated in similar ways from the latent class z by using the conditional probabilities $p(w|z)$ and $p(d|z)$. The values of $p(z)$, $p(w|z)$ and $p(d|z)$ in the model are then estimated using Expectation-Maximization. Among other state of the art generative models, we can mention the naive Bayes classifier. In this model, we make strong independence assumptions between the features characterizing a class, that is, given a class C, we consider each feature F_i to be conditionally independent on the other features F_j ($j \neq i$), i.e. $p(F_i, F_j|C) = p(F_i|C) \cdot p(F_j|C)$. By deriving the model using Bayes rule, we finally obtain: $p(C, F_1, \ldots, F_n) = p(C) \prod_{i=1}^n p(F_i|C)$. The parameters of the model are generally obtained using a maximum likelihood estimator which amounts to counting occurrences of features in the training dataset and then extracting corresponding relative frequencies. The Averaged One-Dependence Estimators (AODE) have been introduced by Webb et al. [Webb et al., 2005] to address the feature-independence issue in the naive Bayes classifier. These estimators try to weaken the independence assumptions between attributes by selecting a subset of 1-dependence classifiers, $p(F_j|C, F_i)$, and by combining the predictions of "qualified" classifiers among this subset. The "qualified" classifiers are selected according to the number n_i of elements in the training data containing the attribute F_i, and by retaining only the 1-dependence classifiers for which this number is superior to a threshold m. The final model can be expressed under the following form: $p(C, F_1, \ldots, F_n) = \frac{\sum_{i|n_i \geq m} p(C, F_i) \prod_{j=1}^N p(F_j|C, F_i)}{|\{l|n_l \geq m\}|}$.

Another very broad category of generative models are hidden Markov models (HMM).

More recently, Jacobs et al. [Jacobs et al., 1991] have introduced the Mixtures of Experts (ME). ME can be considered as mixture models in which the mixture components are conditional probability distributions, $p(y|x, c_i)$, where y is the output associated to an input data x and c_i refers to the i-th mixture component. In an ME framework, we assume that the data can be explained by a collection of functions which are defined each on a local region of the input data space. The regions possess soft boundaries which are generally built using simple parameterized surfaces estimated from the data. Thus, the input space is softly divided into overlapping regions that can be assigned to one or more expert networks. A general ME model can be formulated using the following equations: $p(y|x) = \sum_i p(y, c_i|x) = \sum_i p(c_i|x)p(y|x, c_i) = \sum_i g_i(x)p(y|x, c_i)$. The functions $g_i(x)$ are called "gating functions" because they are in charge of "assigning" a region of space to a given classifier/expert. The power of such a formulation is that any classifier can be employed to model the posterior probabilities $p(y|x, c_i)$: Lima et al., for instance, have proposed a model called Mixture of Support Vector Machine Experts

using SVM classifiers for the experts and soft-max activation functions for the gating modules. The whole model is trained using an EM algorithm optimizing a global likelihood function. Bishop et al. have proposed an enhancement of the basic mixture of experts model under the form of hierarchical mixtures of experts (HME).

Discriminative approaches try to estimate directly the posterior probabilities of classes. Among the most famous discriminative learning methods, we can mention the Support Vector Machine (SVM) classifier[Schölkopf and Smola, 2002] which, in its simplest form, is a binary linear classifier. It belongs to the class of maximum-margin classifier, that is, it tries to find the separating hyperplane which is the farthest from both the positive and negative training data samples (under the constraint of course that the separating hyperplane correctly classifies the training data, that is, respects the class labels assigned to the training samples). Non-linear SVMs classifiers are obtained by applying the kernel trick to maximum-margin hyperplanes. The idea is to replace the standard euclidean dot product by a non-linear kernel function. It has been shown to be equivalent to mapping the data from the original input space into a high-dimensional feature space. A maximum-margin classifier is then fit to the data in this transformed space. Many variants of the initial SVM algorithm have been proposed, including fuzzy SVMs which allows to train the SVM model with fuzzy class memberships, Relevance Vector Machine (RVM [Tipping, 2001]) which tries to further sparsify the set of support vectors obtained by the standard SVM algorithm. The latter approach exploits a Bayesian learning framework which brings the additional benefits of yielding probabilistic predictions in the output as well as the possibility to use non-Mercer kernel functions. The functional form of the model remains the same, that is, we still seek to find the parameters (w, b) of a separating hyperplane y in the high-dimensional feature space: $y(x) = \sum_i \alpha_i k(x, x_i) + b$ where $k(., .)$ is the kernel function (the hyperplane parameter w is defined by the equality $< w, \phi(x) > = \sum_i \alpha_i k(x, x_i)$ where $< ., . >$ is the dot product defined by $k(., .)$ in the high-dimensional feature space and ϕ the mapping function from the original input space to the feature space). The RVM approach considers that the targets y_n are sampled from the model with additive Gaussian noise, yielding: $p(y_n|x_n) = \mathcal{N}(x_n; y_n, \sigma^2)$. By considering a function $y(x_n)$ as a weighted sum of kernel functions, we obtain with an extra assumption of independence between the targets y_n: $p(y|w, \sigma^2) = (2\pi\sigma^2)^{-N/2} \exp(-\frac{1}{2\pi\sigma^2}||y - \gamma w||^2)$ where γ is an $N \times (N+1)$ matrix corresponding to the Gram matrix $\{k(x_i, x_j)\}_{1 \le i, j \le N}$ with an added column of 1 before and $w = [b, w_1, \ldots, w_N]^T$ is the parameter defining the separating hyperplane. Here, $y = [y_1, \ldots, y_N]^T$. To complete the Bayesian formulation, priors are added on the parameters w and σ^2. The choice of a Gaussian prior on the hyperplane parameter w (i.e. $p(w|\alpha) = \prod_{i=0}^{N} \mathcal{N}(w_i; 0, \alpha_i^{-1})$) encodes a preference for less complex models. Having defined the priors, we continue with the Bayesian inference process by trying to estimate the posterior over all unknown parameters given the data: $p(w, \alpha, \sigma^2|y) = p(w|y, \alpha, \sigma^2)p(\alpha, \sigma^2|y)$. Given a new test point x^*, predictions are made for the corresponding target y^* in the following way: $p(y^*|y) = \int p(w|y, \alpha, \sigma^2)p(\alpha, \sigma^2|y) \, dw \, d\alpha \, d\sigma^2$. By deriving the model, it can be shown that $p(w|y, \alpha, \sigma^2)$ and $p(\alpha, \sigma^2|y)$ are Gaussian distributions. Thus, $p(y^*|y) = \mathcal{N}(y^*; \mu^*, \sigma^{*2})$. The general RVM framework described above can be applied to regression as well as classification problems. The form of $p(y|w)$ in the classification case will only be different from the one described above since we have to account for the distribution of

the target values which do not follow a Gaussian distribution but a Bernouilli distribution, yielding: $p(y|w) = \prod_{n=1}^{N} \sigma \left(y(x_n; w) \right)^{y_n} \left[1 - \sigma \left(y(x_n; w) \right) \right]^{1-y_n}$ where σ is the sigmoid function.

RVMs are a special case of a larger category of models called Gaussian Processes (GP). In fact, GP models are derived exactly in the same way except that we replace the distribution $p(y|w)$ which relies on a predefined form of the prediction function $y(x)$ by a generic distribution $p(y)$ following a multivariate Gaussian distribution. $p(y)$ is called a Gaussian process and it is a distribution over functions. According to the above considerations, defining a Gaussian process amounts then to choosing a form for the mean and covariance function defining the Gaussian distribution $p(y)$.

There exists many other discriminative models for classification, like linear discriminant analysis, neural networks, decision trees, boosting techniques ... We only give here an insight into kernel-based classification techniques since these are the techniques which have mainly been exploited throughout this work.

Discriminative methods possess the advantage of addressing the classification problem in a straightforward way by directly computing the probabilities $p(y|x)$ of class labels y given the input data x. In this sense, they can succeed where generative methods fail (Long et al. [Long and Servedio, 2006]). Generative approaches, indeed, seek to solve a more general problem through the modeling of the conditional distributions of data given class labels $p(x|y)$ (which generally leads us to make very strong assumptions regarding the form of these distributions) instead of dealing directly with the classification problem. But, discriminative methods will be impaired by the necessity of having representative datasets to train the models. To cope with this drawback, semi-supervised discriminative approaches have been proposed, which make use of both labeled and unlabeled training data. In the following paragraph, we try to give an overview of semi-supervised methods both from the discriminative and the generative point of view. A complete survey of the field can be found in Chapelle et al. [Chapelle et al., 2006a].

Semi-supervised learning models   Gathering enough labeled data for a particular learning problem often requires a lot of efforts from an outside agent (generally human) to manually classify training examples. The cost of constituting a representative labeled training set may thus be prohibitive in many cases. In this context, it can be interesting to try to make the most out of unlabeled data which are relatively inexpensive to acquire. It has indeed been proved through years of research on semi-supervised techniques that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in the learning accuracy. Three of the main methods that seek to exploit the intrinsic information contained in the unlabeled data are (a) statistical generative models, (b) kernel methods and (c) graph-based methods.

Among the statistical methods which can be ranked among semi-supervised methods, we can mention the associative models and the latent variable models (whether this last category of models belongs to semi-supervised methods is questionable). The idea of associative models is to determine probabilistic associations between semantic concepts and visual words (which most often result from a vector quantization or a clustering of the database). In these models, the unlabeled data are used indirectly in

the vector quantization step. More theoretically sound approaches consider an unsupervised model of the data under the form of a mixture model. Computing the parameters of the whole associative model is done through the maximization of a global likelihood function, often using an EM algorithm. In this case, the parameters of the mixture components and the probabilistic associations between components and semantic concepts are determined simultaneously. Latent variable models are also based on the idea of visual words, but seek to discover latent "topics" in the data which are intermediate unobserved variables allowing to explain the semantic categories in terms of visual words. The goal of the learning process is thus to compute models which best represent the distribution of codewords inside each category. Both associative models and latent variable models can be considered as generative models.

Semi-supervised kernel methods exploit two guiding principles. The first is to modify a standard kernel (e.g. a Gaussian kernel), with the help for instance of an additive or multiplicative term determined using the unlabeled data, hoping that the result will better capture the geometric structure of the data. This idea is generally referred to in the literature as the kernel deformation principle. Among the techniques based on this principle, we can mention the approach of Chapelle et al. [Chapelle et al., 2006b] who have proposed "cluster kernels" based on a modification of the metric of space: at equal distances, two points are considered closer if they belong to the same cluster. This is a direct translation of one of the fundamental underlying assumptions of semi-supervised methods, the "cluster assumption" saying that two data points are likely to have the same labels if and only if there is a path connecting them passing through areas of high density only. This assumption can be formulated in an equivalent way using the idea of "low density separation" saying that a decision boundary between two different classes has to pass through areas of low density only. Although the equivalence with the "cluster assumption" is easy to see, this second formulation has nevertheless inspired quite different algorithms from the first one. In the "cluster kernels" approach, the "cluster assumption" is encoded using a modification of the kernel matrix eigenspectrum. The proposed method possesses a strong connection to spectral clustering methods. It consists first in extracting the Laplacian matrix L of the data and then to compute the eigenvectors of L corresponding to the first k eigenvalues $(v_1, \ldots, v_k)$. The data points are then replaced by their spectral representation (the data point x_i for instance is replaced by $(v_1^i, \ldots, v_k^i)$ with an added normalization step to have length one). The advantage of the spectral representation is that points are naturally clustered along (quasi-)orthogonal axis, i.e., if there exists k distinct groupings/clusters inside the data, there will be in the new spectral representation k vectors orthogonal to each other such that each data point is mapped to one of these k vectors depending on the cluster is belongs to. It is then easy to see that the Gram matrix obtained from the new point representation will have a general term equal to one for data points belonging to the same cluster (since it will be the dot product of almost colinear vectors of norm 1) and zero for data points belonging to different clusters (since it will be the dot product of almost orthogonal vectors). The obtained kernel will thus reflect in a straight way the natural clusters inside the data.

There exists other formulations of the "cluster assumption" whose purpose is also to define a kernel which directly encodes the structure of the data. We can mention the mutual information and the marginalized kernels which rely both on a statistical mod-

eling of the data as a mixture of Gaussians. As in the preceding case, these kernels are designed in such a way that two points belonging to two different clusters/components will have a low dot product.

The second guiding principle in kernel-based semi-supervised methods is the one exploited by transductive SVMs [Joachims, 1999]. The idea which is formalized here is rather that of low density separation: transductive SVMs indeed look explicitly for a separation surface passing through low-density regions. The basic problem is a combinatorial one: in addition to the parameters (w, b) of the separating hyperplane, we also look for the labeling of the unlabeled data such that if a classical SVM was trained with this labeling, we would obtain a classifier that has the largest possible margin (and consequently that passes through the regions of lowest density). It amounts thus to maximizing over the 2^N possible labelings (where N is the number of unlabeled data). Several solutions have been proposed in the literature to avoid solving the combinatorial problem as such. The historical approach is that of Joachims et al. who have described an iterative method which converges towards a local optimum of the objective function. The proposed method consists in initializing the problem with the labeling given by an inductive (standard) SVM trained over the labeled data. Then the algorithm improves the solution by switching the current labels of test samples based on the values of the associated slack variables. The influence of the test samples is gradually increased through the iterations by adjusting the SVM cost-factors "C^*" associated with the unlabeled/test samples. It has been shown that this scheme converges towards a stable solution.

More direct approaches have been proposed to address the underlying combinatorial problem. Chapelle et al. have introduced a branch and bound algorithm which is an alternative to exhaustive search but which still finds the optimal solution of the problem [Chapelle et al., 2007]. This method considerably reduces the number of possible solutions to train and evaluate but it remains impracticable when the number of unlabeled data is very large. Xu et al. [Xu et al., 2005] have proposed an approximate reformulation of the S3VM problem under the form of a semi-definite program, which has interesting properties such as convexity but which is very expensive both in terms of computations and memory requirements. In general, we may notice that the solutions proposed to solve the basic transductive SVM problem are not designed to handle large volumes of data or suffer from optimization issues such as convergence towards local minima of the objective function.

Graph-based methods are the third category of semi-supervised methods we want to briefly describe in this section. They are generally based on the construction of the graph Laplacian of the data, on which is then applied a principle of label propagation (which, as its name indicates, "propagates" the labels from the labeled data to the unlabeled data). Zhu et al. [Zhu and Ghahramani, 2002] have proposed a slightly different approach based on the notion of transition matrix but the derivation of the model follows essentially the same steps: given a transition matrix T whose general term T_{ij} is the probability to jump from node j to node i and a label matrix Y of size $l + u$ where l is the number of labeled data and u the number of unlabeled data, we want to find the final label vector Y_n such that $Y_n = T Y_n$ (which is equal to $\lim_{n \to \infty} T^n Y_0$). It can be shown that the solution to this problem can be obtained by solving a sparse system of linear equations or by repeating an iterative scheme of the form $Y^{n+1} \leftarrow T Y^n$ until

convergence. The main problem is the resources needed to store in memory a connected graph with a number of nodes equal to the number of data points (typically, we store the corresponding Laplacian or transition matrix depending on the method). It is also quite computationally demanding since it amounts to solving a linear system of size equal to the number of unlabeled data points. One can nevertheless mention the approach of Zhu and Lafferty [Zhu and Lafferty, 2005] who use a structuring of the input space by performing a clustering on the data to alleviate the problems due to the size of the graph.

3.4.2   Learning models for auto-annotation

Semantic image annotation can be thought of as a supervised learning problem where the parameters of a model are learned using a training dataset. Figure 3.3 illustrates a typical scenario for analyzing and annotating image databases. The first step consists of extracting feature vectors (descriptors) from images. A training database consisting of annotated samples is then used to compute the model parameters. The computed model is re-used in the last step to extend the annotations to the whole database. We can roughly distinguish between three categories of image annotating systems: (1) those which rely on a probability distribution of image data (most of the time, a Gaussian Mixture Model (GMM)) to model the per-concept distribution of image data [Fan et al., 2008], [Carson et al., 2002]; (2) those which use one versus all strategies where a classifier is learned from a binary training set (concept we want to learn versus the rest). The learned classifier is then used to annotate the rest of the database images. Most of one versus all strategies rely on a Support Vector Machine (SVM) classifier which maximizes the margin between the positive and negative images [Tong and Chang, 2001]; (3) The last category consists of latent variable models [Loehlin, 1987] whose basic idea is to use a set of latent variables which represent hidden (not directly observable) "behaviors" in the data [Fei-Fei and Perona, 2005], [Lienou et al., 2010]. These hidden "behaviors" are generally called "topics" or "themes". As a general rule, models falling into the categories (1) and (3) can be considered as generative models whereas models falling into the category (2) rather belong to the class of discriminative models.

3.4.3   Learning models for interactive image search

Interactive systems involving the user in the learning process have received a growing interest these past few years, especially in the CBIR community. New learning paradigms have emerged centered on the notion of human-computer interaction, marking a difference with learning paradigms in earlier systems which were focused on fully automatic strategies. More and more recent CBIR systems indeed make use of a "relevance feedback" feature (which allows the users to mark retrieved images as relevant or non relevant) to iteratively learn a model of the query target. In this context, active learning has become a state of the art tool. At each retrieval step, it consists in presenting to the user the most informative items while trying to achieve two goals: first, to learn with accuracy the targeted concept and, second, to do so as quickly as possible with minimal effort from the user. In the following paragraph, we present a brief review of active learning methods proposed in the literature.
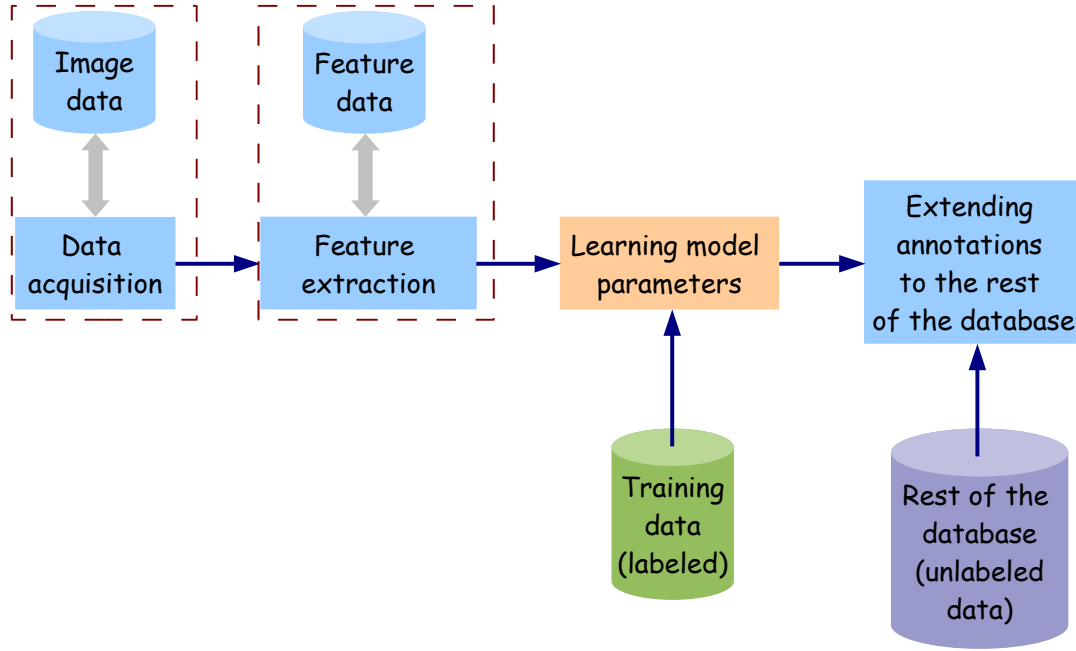
Figure 3.3: Analyzing and annotating image databases. We start by extracting descriptors from images. A training database consisting of annotated samples is then used to compute the model parameters. The computed model is re-used in the last step to extend the annotations to the unannotated samples.

Relevance feedback and active learning: selecting the most informative samples in the feedback process   In interactive search systems, the user is allowed to initiate the search using a single query image as an example of the category/target he is interested in. A selection of images is then presented to the user by the system. It can be the first top-ranked similar images or any other set of images depending on the feedback strategy implemented by the system. The interactive process then allows the user to refine his request as much as necessary using a relevance feedback loop. Many interaction schemes have been proposed in the literature a review of which is presented in Chang et al. but the most common one is a binary scheme in which the user is simply asked to assess as relevant or non-relevant the images fed back by the system.

We can distinguish between two distinct approaches in the use of the relevance feedback component. The first one consists in using the feedback loop to refine the user query. These methods are generally referred to in the literature as *query modification* or *query reweighing*, depending on the use they make of the user labelings. The most simple query modification scheme consists in computing at each iteration of the feedback loop a new query as the average of the descriptors corresponding to the images which have been tagged as relevant by the user (Rui et al. [Rui et al., 1998]). Rocchio et al. [Rocchio, 1971] have described a query point movement algorithm trying to improve the estimate of the "ideal" query at each iteration, based on weighted means of relevant and irrelevant queries respectively. At iteration k, the new query estimate Q_k is obtained as: $Q_k = \alpha Q_{k-1} + \beta \left(\frac{1}{|D_R|} \sum_{i \in D_R} v_i \right) - \gamma \left(\frac{1}{|D_N|} \sum_{i \in D_N} v_i \right)$ where D_R and D_N are respectively the sets of relevant and irrelevant descriptors v_i. The constants α, β, γ

have to be determined using a cross-validation process for instance. The idea behind the above update formula is to move away from irrelevant images while getting closer to relevant images.

Query reweighing is a more sophisticated scheme whose purpose is to tune the distance/
similarity function between the query and any image in the database. Aksoy et al. [Aksoy et al., 2000] have proposed an approach consisting in weighting each axis in the description space using the weighted $L1$ distance. The idea they exploit is based on the fact that, for a dimension in the description space to be good/informative regarding the targeted class, the variance of relevant elements along this dimension has to be small compared to the total variance along the corresponding axis (i.e elements of the same class have to be grouped along this dimension). The weight along the i-th dimension is finally set as $w_i = \frac{\sigma_i^{tot}}{\sigma_i^{Rel(k)}}$ where σ_i^{tot} is the standard deviation of elements in the database along the i-th axis and $\sigma_i^{Rel(k)}$ is the standard deviation of relevant elements at the k-th iteration of the feedback loop along the i-th axis. There exists many other works presenting re-weighting relevance feedback algorithms. The idea is always to give more importance to the features which are helpful for retrieving relevant information and to reduce the importance of the features which do not contribute in a significant way to the retrieval process. We can nevertheless mention the approach of Peng et al. [Peng et al., 1999] who have introduced a probabilistic method which uses the Bayes prediction error as a criterion to automatically assess the relevance of a certain feature. The weight for the i-th dimension is computed as: $w_i = r_i(z)^t / \sum_{j=1}^{q} r_j(z)^t$ where $r_i(z)$ is the reduction in the prediction error achieved by using a least square estimate for the prediction function f_i instead of the null predictor (i.e. the predictor which predicts the class of z to be 0 while z belongs to class 1 with probability 1). Thus: $r_i(z) = (f_i(z) - 0) - (f_i(z) - E[f|x_i = z]) = E[f|x_i = z]$ (z represents the query). The weights w_i are then used to compute a weighted Euclidean distance.

More efficient active learning schemes consider the relevance feedback component inside a binary classification problem. In this case, finding the optimal query is reformulated as a binary classification task between the relevant and irrelevant image classes. Many classification techniques have been employed inside relevance feedback schemes including decision trees, neural networks, kernel methods, Bayesian methods ...

The reference among Bayesian methods is perhaps the PicHunter system proposed by Cox et al. [Cox et al., 2000]. This system implements the user feedback in terms of "relative similarity judgments" among images (this imposes less constraints on the user than a categorical feedback where one has to select only the images that are in the same category as the target). The PicHunter system relies on a learned model of human behavior to better exploit the user feedback at each learning iteration. The retrieval model can be summarized by the following update equation:

$$p(I_i|H_k) = \frac{p(A_k|I_i, D_k, H_{k-1}) p(I_i|H_{k-1})}{\sum_{j=1}^{N} p(A_k|I_j, D_k, H_{k-1}) p(I_j|H_{k-1})}$$

where I_i is the i-th image, D_k the set of images displayed at iteration k and A_k the action taken by the user at this iteration (i.e., the images he intends to select on the

display as pertaining to the targeted concept). The history of previous iterations is denoted as $H_k = \{D_1, A_1, \ldots, D_k, A_k\}$. The key element of this equation is the term $p(A_k|I_i, D_k, H_{k-1})$ which represents the user model. A_k is a discrete variable which can take any value between 1 and the size of the display. In plain language, $p(A_k = l|I_i, D_k, H_{k-1})$ is the probability of the l-th image on the display to be more relevant than the other images which are displayed. In the PicHunter system, a feedback strategy is adopted which aims at minimizing the number of iterations left in the feedback process. The resulting scheme tries to maximize at each iteration the transfer of information from the user to the machine, which is globally equivalent to minimizing the number of "questions" the system has to ask the user to resolve the ambiguity (i.e. to find the target concept). Information theory suggests the entropy as an estimate of this number. The distribution $P(I)$ being the distribution which reflects the amount of knowledge the system has acquired about the target, the number $C(P(I))$ of remaining iterations/"questions to ask" is estimated as: $-\alpha \sum_{i=1}^N P(I = I_i) \log P(I = I_i)$. This quantity is then re-used to compute the expected number of remaining iterations $C\left(I_{d_1}, \ldots, I_{d_{N_d}}\right)$ when the display is composed of the images $\left\{I_{d_1}, \ldots, I_{d_{N_d}}\right\}$. In order to find which images to display in the end, an algorithm is proposed which minimizes $C\left(I_{d_1}, \ldots, I_{d_{N_d}}\right)$ over all possible N_d tuples $\left\{I_{d_1}, \ldots, I_{d_{N_d}}\right\}$.

The Knowledge-driven Information Mining (KIM) system proposed by Datcu et al. [Datcu et al., 2002] is another relevance-feedback-based system using a Bayesian paradigm. This system consists of three distinct parts: a primitive feature extraction component, a Bayesian network as the classification component used to generate interactively image classifications and a database management system for the image content information catalogue. The user knowledge is transferred to the system by means of a simple relevance feedback algorithm which consists in presenting to the user the posterior maps of the targeted concepts. The user can then indicate positive and negative examples directly on the posterior maps, which allows to update the posterior according to the following equation: $p(L|w_i, D) = p(L)\frac{\sum_i p(w_i|L)p(w_i|D)}{p(w_i)}$ where L is the target concept, w_i are signal categories obtained by performing a clustering on the database and D represents the data. The user feedback is used to compute $p(w_i|L)$ according to the formula: $p(w_i|L) = \frac{1+N_i}{\sum_j 1+N_j}$ where N_i is the number of times we observe the signal category w_i among the positive feedback examples.

Among more recent Bayesian systems making use of interactive learning techniques, we can mention the system proposed by Li et al. [Li and Bretschneider, 2006], which is a relevance-feedback-based CBIR system using a context-sensitive Bayesian network to link low level image features to semantic concepts. Among other characteristics, the context-sensitive Bayesian network allows to take into account the context information by considering pairs of adjacent regions. It consists of four interconnected layers containing respectively an image I_i, the region pairs $(R_i^{j_1}, R_i^{j_2})$ extracted from this image, the code pairs $\left(C(R_i^{j_1}), C(R_i^{j_2})\right)$ associated with the region pairs and the semantic concepts $SC_1, \ldots, SC_L$, the aim being to compute the posterior probabilities $p(SC_k|I_i)$ of each semantic concept SC_k given an image I_i. These posterior probabilities depend on the probabilities $p(C(R_i^{j_1}), C(R_i^{j_2})|SC_k)$ of the code pairs given the semantic concepts which are learned using an interactive learning process that simply consists in

counting the relative code pairs frequencies in a user-supplied training set. A query-point-movement-like scheme is further added to better capture the user's query intentions. More precisely, the authors propose to combine all regions from the initial query and the positive examples into a single pseudo query image which is used as the optimal query for the next retrieval iteration. They make use of a popular Region-Based Relevance Feedback (RBRF) scheme proposed by Jing et al. [Jing et al., 2004] which consists in reducing the size of the pseudo query image using a K-means algorithm. The query image results indeed from the accumulation of positively-tagged examples during the relevance feedback process, so, its size increases at each iteration, which, in turn, causes the retrieval speed to slow down gradually. The only purpose of the k-mean algorithm is to synthesize a pseudo query image of constant size.

In recent systems, more attention has been paid to techniques using kernel learning inside a relevance feedback scheme. The historical approach is perhaps the one of Tong et al. [Tong and Chang, 2001] who have introduced a support vector machine active learning algorithm for image retrieval. At each iteration of the feedback loop, the $\text{SVM}_{\text{active}}$ algorithm selects the most informative samples to display to the user and adjusts the SVM boundary delineating the target class in accordance with the user feedback on each of these samples. More theoretical details about this algorithm will be given in the next chapter, so we will content ourselves with mentioning that this technique achieves significant improvements over preceding state of the art active learning techniques in terms of both learning speed (number of iterations in the active learning loop) and learning accuracy. Costache et al. [Costache et al., 2006] have presented a slightly different version of the $\text{SVM}_{\text{active}}$ algorithm and describe its integration into a search engine to perform category search in high-volume EO image repositories.

Among more sophisticated approaches using an SVM classifier in an active learning context, we can mention the approach of Tao et al. [Tao et al., 2006] who have introduced an algorithm to handle the class imbalance between positive and negative samples. This naturally occurs when using relevance feedback to build the training set, due to the fact that the number of positive feedback samples is often far less important than the number of negative feedback samples. To alleviate this problem, they propose an asymmetric bagging-based SVM (AB-SVM) to address both the class imbalance problem and the instability of SVM classifiers on small training sets. Bagging (also called bootstrap aggregating) consists in training T classifiers on T different subsets $D_1, \ldots, D_T$ obtained by sampling uniformly and with replacement from the original training set. The outputs of the obtained classifiers are then combined using a Majority Voting Rule, or, if the classifiers yield probabilistic outputs, by using a Bayes Sum Rule: $C(x) = \arg\max_k \left[\sum_{t=1}^{T} p(y_k | f_t(x)) \right]$ where f_t is the decision function associated with the t-th classifier. Tao et al. propose to use only the negative samples in the bootstrap process since there are far more negative samples than positive ones. The bagging process allows thus to train a classifier with a balanced number of positive and negative training samples.

Ferecatu et al. [Ferecatu and Boujemaa, 2007] have proposed a strategy to enhance the sparsity of the feedback examples in the $\text{SVM}_{\text{active}}$ algorithm. This ensures among other a better exploration of the SVM separating frontier by eliminating redundancy

among displayed feedback samples. Their approach called "Most Ambiguous and Orthogonal" (MAO) consists in selecting the samples which are the closest to the SVM separating surface (most ambiguous "component") and which possess the smallest pairwise scalar products (most orthogonal "component"), i.e. the samples $x_1, \ldots, x_d$ such that $x_j = \arg\min_{x \in S} \max_{i \in 1, \ldots, n} k(x, x_i)$ where S is the set of images not yet included in the preceding MAO steps and x_i, $i = 1, \ldots, n$ are the already chosen candidates. In plain language, we seek each time to minimize the highest of the values taken by $k(x_j, x_i)$ where $k(.,.)$ is the kernel function. The "max" part ensures that the new candidate is close to the separating surface (in fact close to the already chosen candidates which we assume close to the separating surface) and the "min" part guarantees that the new candidate is as much orthogonal as possible to the already chosen ones.

The RETIN system [Gosselin et al., 2008] also uses an SVM classifier with a strategy to prevent the selection of feedback samples close to each other. Considering the same notations as before, a new candidate x_j is selected according to:

$$x_j = \arg\min_{x \in S} \left(g(x) + \max_{i \in 1, \ldots, n} k(x, x_i) \right)$$

where $g(.)$ is the decision function associated with the current SVM classifier. Moreover, a boundary correction feature is added to better position the separating surface in the early steps of the relevance feedback process. This is indeed always problematic since there are very few training samples available at this stage. It is also sometimes hard to find initial examples of the target class without some third-party knowledge (such as keywords, for instance). And, mostly, the most ambiguous strategy yield good results on condition that the class boundary is already quite accurately positioned. The correction scheme consists in shifting the SVM surface by a constant b_t recomputed at each iteration t of the feedback loop: $g_t^*(x) = g_t(x) - b_t$. The idea is to move the boundary towards the most uncertain area of the database by considering the proportion of positive feedback samples against the proportion of negative ones among the samples which are displayed to the user based on the current estimate of the boundary: the surface is well positioned when the set of selected images is well balanced between positive and negative samples. The proposed scheme relies entirely on the ranking obtained using the current SVM decision function g_t: $x_{i_1}, x_{i_2}, \ldots, x_{i_r}, x_{i_{r+1}}, \ldots, x_{i_{r+m}}, \ldots, x_{i_{n-1}}, x_{i_n}$. The samples are ranked in decreasing order of relevance. The area $x_{i_r}, x_{i_{r+1}}, \ldots, x_{i_{r+m}}$ represents the zone of highest uncertainty on which the user is asked to give his feedback: if the user labels more relevant samples than irrelevant ones, it means that the zone of uncertainty can be shifted towards samples of lower rank to get more irrelevant samples. On the contrary, if the user labels more irrelevant samples than relevant ones, the threshold i_r is shifted towards samples of higher rank to get more relevant samples. The shift b_t is finally evaluated as $g_t(x_{i_{r+\lfloor \frac{m}{2} \rfloor}})$.

3.5 Proposed concepts for searching huge databases using small training sets

Among the difficulties encountered when searching image databases, we may think on one hand to the problems arising when the training sets used to represent the targeted
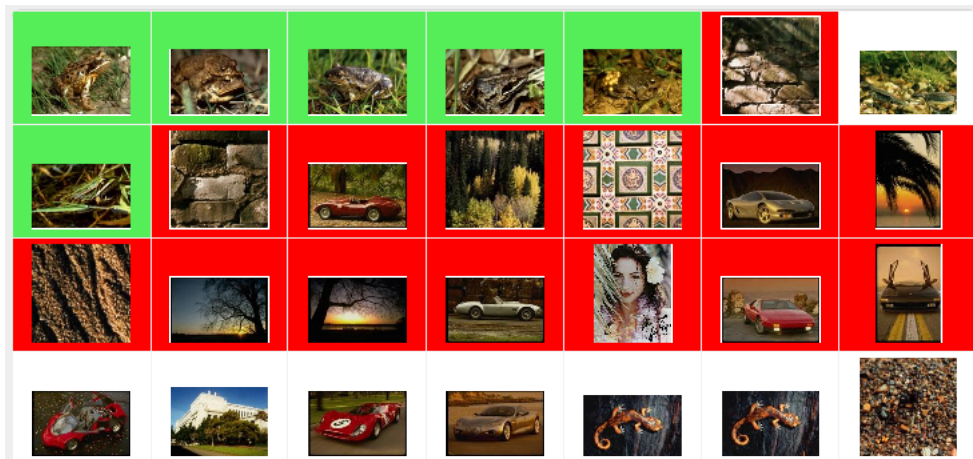
Figure 3.4: Graphical user interface of the RETIN system (http://retin.ensea.fr/). The above image corresponds to the current display after ten iterations of feedback. The target class is the one represented by the images surrounded in green (here, we look for batracians).

image classes contains very few examples, and, on the other hand to the problems inherent to high-volume image databases.

Developing retrieval and indexing algorithms which scale with databases containing hundreds of terabytes of images is an emerging problematic which poses many questions regarding the validity of state of the art approaches on such high-volume databases. This problematic is also linked to the first one since large volume of data often means high semantic diversity i.e. that the number of semantic classes contained in the database is very high. It is thus altogether impossible to constitute exhaustive training datasets with representative examples of each semantic class present in the database. High-volume datasets also render the task of exploring the database very demanding on a computational point of view. This is a problem which has raised but little concern in recent and less recent CBIR systems. Most active learning schemes implemented in these systems nonetheless require the evaluation of the current decision function on the whole dataset as an essential step of the feedback process. Choosing the most informative examples to feed back to the user is indeed based on the response of each element of the database to the decision function associated with the current classifier. This is notwithstanding the fact that highly performant classifiers yield complex decision functions involving a non-negligible amount of computation to train and evaluate.

To handle this problem some systems use a metric structuring of the database to quickly identify informative samples. This requires the use of space-partitioning methods and index data structures, such as kd-trees or quadtrees, which address the problem of how to organize the data and how to perform queries without accessing all available data. Among the few practical implementations of such systems, we can mention the KIM system Datcu et al. [2002] as well as GeoIRIS Shyu et al. [2007] which are both dedicated to the indexation of very high-volume (hundreds of terabytes) remote-sensing image databases. However, these systems can be used only for direct queries and ignore the intrinsic dependencies in the data which are needed to model more

complex classes.

Another way of coping with very high-volume databases is to use machine learning techniques such as for instance "smart" semi-supervised active learning schemes in the case of semantic category search and coarse-to-fine methods in the case of object detection. We will focus on this second way of envisaging the problem to develop our own solutions.

The ideal approach would be of course to combine the advances made by both the database and the object retrieval communities. Databases moreover address frontally the problem of large memory requirements such as may be encountered when dealing with huge amounts of data. This problem is not tackled directly in this work, but we refer to it frequently in the following since it is an underlying and recurring problem in the deployment of semi-supervised methods.

The key contributions of our work are summarized below:

- We propose a method to handle the non-exhaustiveness of training databases in the case of auto-annotation systems and we also describe how this approach can be used to help the user in his database exploration task by providing him with relevant examples of unseen categories [Blanchart and Datcu, 2009, 2010; Blanchart et al., 2011d], that is, categories which do not possess any representative in the training set used to build the auto-annotation model.

- We describe a method which aims at increasing the learning speed in the case of interactive image retrieval systems, that is, systems involving the user during the learning through the use of a relevance feedback loop [Blanchart et al., 2010, 2011c,d]. Minimizing the number of iterations in this loop is a critical issue. We propose a semi-supervised algorithm which exploits the intrinsic structure of the data to speed up the interactive learning process.

- We introduce a cascaded active learning method to detect objects in large satellite image repositories [Blanchart et al., 2011a]. Our method performs coarse-to-fine testing using a multiscale patch-based representation of satellite image scenes. Unlike most object detection methods which require large training sets and a costly offline training step, our approach makes use of a "cascaded" active learning strategy to build a classifier at each level of the coarse-to-fine hierarchy (which can also be envisaged under the form of a cascade of classifiers).

The three different systems and the related methods mentioned above can be seen as the core components of a more general system for auto-annotating image databases starting from a small training dataset containing only a few semantic categories with a few training examples per category [Blanchart et al., 2011b]. The auto-annotation and the interactive image search components are indeed to be envisaged in a complementary way, the interactive image search engine being used to build the training dataset which in turn is used to build the auto-annotation model. The whole process can thus be seen as a loop in which, after each use of the interactive image search engine, the newly-learned categories are incorporated inside the training database of the auto-annotation system. The latter is then used to extend the annotation to the whole database based on the auto-annotating model trained over all the previously learned categories. The unknown structure discovery feature of the auto-annotation

system allows the identification of new unseen classes to be searched for with the category search or the object detection tool ("unseen classes" are classes that are not represented in the current training set of the auto-annotation component). The whole concept is summarized in the diagram of the figure 3.5.
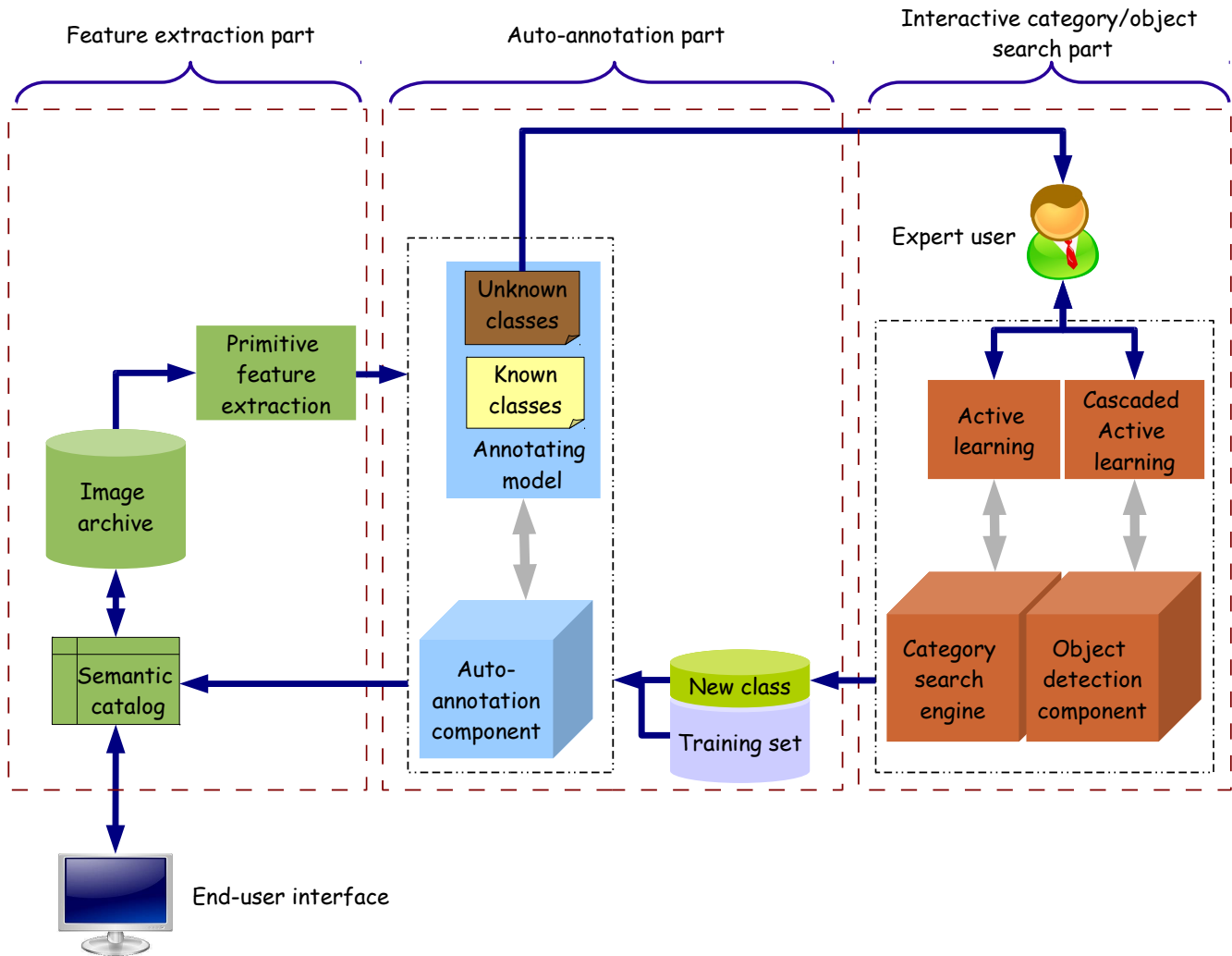
Figure 3.5:   Concept for an hybrid system combining the advantages of auto-annotation and CBIR systems. The whole information mining process can be seen as a loop in which the interactive image search engine is used to build the training dataset, which, in turn, is used to train the auto-annotation model. The latter allows then to extend the annotations to the whole database. The unknown structure discovery feature of the auto-annotation system can also be exploited to identify new/unseen classes to be searched for with the category search or the object detection components of the interactive image search engine.

Chapter 4

Basic algorithms and frameworks for classification and learning

In this chapter, we introduce the theoretical foundations which are useful to understand our contributions to the field. After exposing some basics, we start with an insight into unsupervised classification methods, then we present supervised and semi-supervised classification techniques, trying to identify common ideas which are used by the three paradigms. We also point out how learning methods can be employed to perform related tasks such as feature selection or feature extraction. In the end, we describe some well-known learning strategies, that is, some general frameworks in which classification methods are deployed.

4.1   Statistical point of view of classification

In this section, we present some basic notions of probability and Bayesian inference from the point of view of classification problems.

The most intuitive approach of probability is perhaps the frequentist one which views probabilities as frequencies of occurrence of random repeatable events. It is often opposed to the Bayesian point of view in which probabilities are considered to provide a quantification of uncertainty.

In the following, we denote by X and Y two random variables. The corresponding probability distributions are referred to as $p(X)$ and $p(Y)$. The probability that X takes the value x is denoted $p(X = x)$. To simplify the notations, we will simply write $p(x)$ to refer to the value taken by $p(X)$ in x. With these notations, the two fundamental rules of probability can be written as:

$$\textbf{sum rule :}\ p(X) = \sum_Y p(X, Y) \tag{4.1}$$

$$\textbf{product rule :}\ p(X, Y) = p(Y|X)p(X) \tag{4.2}$$

By using the product rule and the symmetry property (i.e. $p(X, Y) = p(Y, X)$), we obtain the following relationship between conditional probabilities:

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)} = \frac{p(X|Y)p(Y)}{\sum_Y p(X|Y)p(Y)} \tag{4.3}$$

which is called the Bayes' theorem. It plays a central role in pattern recognition and machine learning where it is often formulated as:

$$p(w|D, M) = \frac{p(D|w, M)p(w|M)}{p(D|M)} \tag{4.4}$$

In this formulation, $D = \{x_1, \ldots, x_n\}$ refers to the observed data, M to the model which best fits the data and w to the model parameters. In this case, M is considered to be known in advance and the problem is thus to find the best set of parameters. Inferring the model parameters w from the data implies making assumptions on:

- the likelihood function $p(D|w, M)$: this function expresses how the data are generated from the assumed model.

- the prior function $p(w|M)$: this function represents the prior belief, that is, the assumptions about w before observing the data.

Fitting the model M to the available data D is done by estimating the model parameters w. In this work, we consider only the two most well-known estimators:

$$\textbf{Maximum likelihood estimator :}\ w_{ML} = \arg\max_{w} p(D|w) \tag{4.5}$$

$$\textbf{Maximum a posteriori estimator :}\ w_{MAP} = \arg\max_{w} p(w|D)$$
$$= \arg\max_{w} p(D|w)p(w) \tag{4.6}$$

$$\textbf{Least squares estimator :}\ w_{LSQ} = \arg\min_{w} \sum_{i=1}^{N} ||y_i - f(x_i, w)||^2 \tag{4.7}$$

In the least squares case, w is the parameter which controls the shape of a function f we want to fit to the training data $(x_i, y_i)_{i=1,\ldots,N}$.

The choice of one estimator or the other is dependent on the problem at hand.

4.2   Unsupervised classification

In this part, we try to provide an insight into unsupervised classification techniques starting from the statistical point of view exposed above. Then, we progressively move towards more heuristic approaches less theoretically founded but which still provide very good results in unsupervised classification/clustering.

Probabilistic models explain the dependencies within a set of observed data through the use of hidden variables which are the parameters of an underlying probability density function (pdf). The most widely used probabilistic model in clustering is without any doubt the mixture model and more particularly the finite Gaussian mixture model which considers the data distribution to be a linear superposition of Gaussian components:

$$p\left(v; \{\pi_l, \mu_l, \Sigma_l\}_{l=1,\ldots,L}\right) = \sum_{l=1}^{L} \pi_l \cdot \mathcal{N}(v; \mu_l, \Sigma_l) \tag{4.8}$$

where the prior probabilities of each component of the mixture, π_l, are normalized such that $\sum_{l=1}^{L} \pi_l = 1$, and μ_l and Σ_l respectively denote the mean and covariance matrices associated with each Gaussian component:

$$\mathcal{N}(v; \mu_l, \Sigma_l) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_l|^{\frac{1}{2}}} \exp(-\frac{1}{2}(v - \mu_l)^T \Sigma_l^{-1}(v - \mu_l))$$

(in the above formula, d is the dimension of the space). The parameters of such a mixture model can be learned using an Expectation-Maximization (EM) algorithm. We derive this algorithm below for the particular Gaussian mixture component case. In the following, we denote by $\theta = \{\pi_1, ..., \pi_L, \theta_1, ..., \theta_L\}$ the mixture model parameters where θ_l refers to l-th mixture component parameters: $\theta_l = \{\mu_l, \Sigma_l\}$.

A common tool to learn the distribution of observed data conditioned on model parameters is the Expectation Maximization algorithm. The EM algorithm [Dempster et al., 1977] gives a way of finding maximum likelihood estimates of model parameters and thus provides a convenient basis to perform Bayesian inference which is the most common way of testing between several hypotheses when we have a probabilistic model of the data. For Gaussian mixture models, the maximum likelihood problem cannot be solved analytically because of the nonlinearity of likelihood equations. The EM algorithm provides an iterative procedure for approximating these estimates by iteratively updating the GMM parameters in such a way that the likelihood function remains non decreasing. We derive a special case of this algorithm dedicated to GMM parameters estimation in the forthcoming paragraph.

EM algorithm   We denote $\mathcal{L}(D, Z, \theta)$ the likelihood of the complete data. $D = \{d_1, ..., d_N\}$ refers to the observed data ($d_i \in R^d$), Z denotes the unobserved data (missing values) and θ is the vector of model parameters as introduced above. In the case of finite mixture models, we often take: $Z = \{z_1, ..., z_N\}$ where z_i is a vector of length L whose l-th component z_i^l characterizes the membership of the observation d_i to the component c_l. The maximum likelihood estimate (MLE) consists in maximizing the likelihood function $\mathcal{L}(D, \theta)$ with respect to θ, which is equivalent to maximizing the marginal likelihood of the complete data $E_Z[\mathcal{L}(D, Z, \theta)]$ with respect to θ: $\theta_{MLE} = \arg\max_{\theta} E_Z[\mathcal{L}(D, Z, \theta)]$. In the finite Gaussian mixture case, we obtain:

$$\theta_{MLE} = \arg\max_{\theta} \sum_{i=1}^{N} \sum_{l=1}^{L} p(z_i^l = 1 | d_i, \theta) \log\left[\pi_l p(d_i | \theta_l)\right] \tag{4.9}$$

(we consider the expected log-likelihood $E_Z\left[\log \mathcal{L}(D, Z, \theta)\right]$ instead of the expected likelihood $E_Z[\mathcal{L}(D, Z, \theta)]$). The EM algorithm provides a two-step procedure to try to find the MLE. The first step (E-step) consists in computing the expected value of the log-likelihood with respect to the conditional density of Z given D under the current estimation of the parameter vector $\theta^{(t)}$:

$$Q(\theta | \theta^{(t)}) = E_{Z|D, \theta^{(t)}}\left[\log \mathcal{L}(D, Z, \theta)\right] = \sum_{i=1}^{N} \sum_{l=1}^{L} p(z_i^l = 1 | d_i, \theta^{(t)}) \log\left[\pi_l p(d_i | \theta_l)\right] \tag{4.10}$$

with $p(z_i^l = 1|d_i, \theta^{(t)}) = \frac{p(z_i^l = 1, d_i|\theta^{(t)})}{p(d_i|\theta^{(t)})} = \frac{p(z_i^l = 1|\theta^{(t)})p(d_i|z_i^l = 1, \theta^{(t)})}{\sum_{k=1}^{L} p(z_i^k = 1|\theta^{(t)})p(d_i|z_i^k = 1, \theta^{(t)})} = \frac{\pi_l^{(t)} p(d_i|\theta_l^{(t)})}{\sum_{l=1}^{L} \pi_l^{(t)} p(d_i|\theta_l^{(t)})}$. The second step (M-step) consists in maximizing $Q(\theta|\theta^{(t)})$ with respect to θ, i.e., we obtain $\theta^{(t+1)}$ as:

$$\theta^{(t+1)} = \arg\max_{\theta} Q(\theta|\theta^{(t)})$$

The above two steps are repeated until a convergence criterion is met.

To obtain the update equations in the M-step, we use the Lagrangian L of $Q(\theta|\theta^{(t)})$ with a normalization constraint on the prior probabilities π_l of each mixture component: $L(\theta) = Q(\theta|\theta^{(t)}) - \lambda \sum_{l=1}^{L} (\pi_l - 1)$. By expanding the Lagrangian, we obtain:

$$L(\theta) = \left(\sum_{i=1}^{N} \sum_{l=1}^{L} p(z_i^l = 1|d_i, \theta^{(t)}) \left(-\frac{d}{2} \ln(2\pi) - \frac{1}{2} \ln(|\Sigma_l|) \right. \right.$$
$$\left. \left. -\frac{1}{2}(d_i - \mu_l)^T \Sigma_l^{-1}(d_i - \mu_l) + \ln \pi_l \right) \right) - \lambda \left(\sum_{l=1}^{L} (\pi_l - 1) \right) \tag{4.11}$$

To find the new estimate $\theta^{(t+1)}$ of the model parameters, we look for the point θ where $\frac{\partial L(\theta)}{\partial \theta} = 0$. Thus for the mean, we have:

$$\frac{\partial L(\theta)}{\partial \mu_l} = \sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \left(-\frac{\partial}{\partial \mu_l} \frac{1}{2}(d_i - \mu_l)^T \Sigma_l^{-1}(d_i - \mu_l) \right)$$
$$= \sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \left(\Sigma_l^{-1}(d_i - \mu_l) \right) = 0 \tag{4.12}$$

$4.12 \Longrightarrow \sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \Sigma_l^{-1} d_i = \sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \Sigma_l^{-1} \mu_l$

$$\Longrightarrow \mu_l = \frac{\sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) d_i}{\sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)})} \tag{4.13}$$

Taking the derivative with respect to the covariance matrices yields:

$$\frac{\partial L(\theta)}{\partial \Sigma_l} = \sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \left(-\frac{\partial}{\partial \Sigma_l} \frac{1}{2} \ln(|\Sigma_l|) - \frac{\partial}{\partial \Sigma_l} \frac{1}{2}(d_i - \mu_l)^T \Sigma_l^{-1}(d_i - \mu_l) \right)$$
$$= \sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \left(-\frac{1}{2} \Sigma_l^{-T} + \frac{1}{2} \Sigma_l^{-T}(d_i - \mu_l)(d_i - \mu_l)^T \Sigma_l^{-T} \right) = 0 \tag{4.14}$$

$4.14 \Longrightarrow \sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \Sigma_l^{-1} = \sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \Sigma_l^{-1}(d_i - \mu_l)(d_i - \mu_l)^T \Sigma_l^{-1}$

$$\Longrightarrow \Sigma_l = \frac{\sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)})(d_i - \mu_l)(d_i - \mu_l)^T}{\sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)})} \tag{4.15}$$

And last:

$$\frac{\partial L(\theta)}{\partial \pi_l} = \left(\sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \frac{\partial \ln \pi_l}{\partial \pi_l} \right) - \lambda \left(\frac{\partial \pi_l}{\partial \pi_l} \right)$$
$$= \left(\sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)}) \left(\frac{1}{\pi_l} \right) \right) - \lambda = 0 \tag{4.16}$$

$4.16 \Longrightarrow \pi_l = \frac{1}{\lambda}\sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)})$ By inserting this expression into the normalization constraint, we obtain:

$$\sum_{k=1}^{L} \pi_k = \sum_{k=1}^{L} \frac{1}{\lambda}\sum_{i=1}^{N} p(z_i^k = 1|d_i, \theta^{(t)}) = 1 \Longrightarrow \lambda = \sum_{k=1}^{L}\sum_{i=1}^{N} p(z_i^k = 1|d_i, \theta^{(t)}) \quad (4.17)$$

$$4.17 \Longrightarrow \pi_l = \frac{\sum_{i=1}^{N} p(z_i^l = 1|d_i, \theta^{(t)})}{\sum_{k=1}^{L}\sum_{i=1}^{N} p(z_i^k = 1|d_i, \theta^{(t)})} \quad (4.18)$$

The new values of μ_l, Σ_l and π_l computed using the update equations derived above become then the new estimate $\theta^{(t+1)}$ to be used in the next estimation step.

We can see that the final values of probabilities $p(z_i^l = 1|x_i, \theta)$ which characterizes the obtained probabilistic clustering depend on the model parameters θ. This suggests that there is an almost deterministic dependency of the algorithm on the initialization parameter $\theta^{(0)}$. This explains why EM is very sensitive to initial parameter values and will thus get easily trapped in local maxima. To remove this dependence on initial parameters and make the convergence towards the global maximum more likely, a variant of EM called deterministic annealing EM (DAEM) has been proposed by Ueda et al. [Ueda and Nakano, 1998].

Maximum likelihood estimation with Deterministic Annealing EM (DAEM)   To introduce the DAEM algorithm, we can start from the following observation: the maximization of $\mathscr{L}(D, \theta)$ with respect to θ is equivalent to the maximization of the following function:

$$F(Z, \theta) = E_Z\left[\log \mathscr{L}(D, Z, \theta)\right] + H(Z) \quad (4.19)$$

which we refer to in the following as the free energy. $H(Z)$ is the entropy of the variable Z:

$$H(Z) = -E_Z\left[\log f_Z(Z)\right] = -\sum_{i=1}^{N}\sum_{l=1}^{L} p(z_i^l = 1|d_i, \theta)\log p(z_i^l = 1|d_i, \theta) \quad (4.20)$$

where f_Z is an arbitrary pdf over the variable Z. It has been shown in Neal et al. [Neal and Hinton, 1998] that if $F(Z, \theta)$ has a local maximum at f_Z^* and θ^*, then the likelihood function $\mathscr{L}(D, \theta)$ has a local maximum at θ^* as well. The same result can be proved for global maxima as well. By introducing $d(d_i, c_l) = -\log\left[\pi_l p(d_i|\theta_l)\right]$ the distance between the vector d_i and the Gaussian component c_l, we can rewrite $E_Z\left[\log \mathscr{L}(D, Z, \theta)\right]$ as a distortion measure:

$$E_Z\left[\log \mathscr{L}(D, Z, \theta)\right] = -Dist = \sum_{i=1}^{N}\sum_{l=1}^{L} p(c_l|d_i, \theta)d(d_i, c_l)$$

$$= \sum_{i=1}^{N}\sum_{l=1}^{L} p(z_i^l = 1|d_i, \theta)d(d_i, c_l) \quad (4.21)$$

The problem is now to maximize $F = -Dist + H$ or to minimize $F = Dist - H$. By introducing a temperature parameter T, we can give more or less importance to the entropy

parameter H. We obtain then the usual statistical mechanic formulation of the free energy: $F = Dist - TH$ which is also the basic formulation of DA. At high temperature, the term H is predominant which implies that we perform almost random moves on the likelihood surface. As we lower the temperature, the term $Dist$ gains more importance and we tend towards the initial EM approach. We recognize here the generic idea behind deterministic annealing: at the beginning of the procedure, we do not have any a priori knowledge about the ML clustering. Then, to obtain f_Z, we adopt a maximum entropy approach which is the only viable principle in that case (it says that when no a priori information is available, the best probability assignment is given by the distribution that contains the less information, that is, the distribution with the highest entropy). As we gain confidence in the initialization of the ML clustering, we can give more importance to the first term $Dist$ and tend towards a minimum distortion approach which performs local optimization and makes the algorithm converge towards the closest local minimum. The relative importance of both principles (maximum entropy and minimum distortion) is adjusted through the temperature parameter T.

We have seen above that finding the MLE is equivalent to minimizing $Dist - H$ with respect to f_Z and θ successively. We can extend this idea to the minimization of the free energy: at $T = 1$, we will obtain exactly the MLE. By differentiating $F = Dist - TH$ with respect to f_Z and solving $\frac{\partial F}{\partial f_Z} = 0$, we obtain the Gibbs distribution, $p(z_i^l = 1|d_i, \theta) = \frac{1}{Z_{d_i}} \exp(-\frac{d(d_i, c_l)}{T})$, which is the only distribution that maximizes the entropy for a given expected distortion. $Z_{d_i} = \sum_{l=1}^{L} \exp(-\frac{d(d_i, c_l)}{T})$ is a normalization factor. By plugging the expression of $p(z_i^l = 1|d_i, \theta)$ back into the expression of F, we get:

$$F^*(\theta) = \min_{f_Z} F(Z, \theta) = -T \sum_{i=1}^{N} \log \sum_{l=1}^{L} \exp(-\frac{d(d_i, c_l)}{T}) \tag{4.22}$$

To maximize F^* with respect to θ, we proceed the same way as in the case of the EM algorithm by computing the Lagrangian with normalization constraints. We find (Ueda et al.) $p(z_i^l = 1|d_i, \theta) = \frac{(\pi_l p(d_i|\theta_l))^\beta}{\sum_{l=1}^{L} (\pi_l p(d_i|\theta_l))^\beta}$ where β is the inverse of the temperature: $\beta = \frac{1}{T}$. We can then express the expected log-likelihood of complete data as:

$$Q(\theta|\theta^{(t)}; \beta) = \sum_{i=1}^{N} \sum_{l=1}^{L} \left\{ \frac{(\pi_l^{(t)} p(d_i|\theta_l^{(t)}))^\beta}{\sum_{k=1}^{L} (\pi_k^{(t)} p(d_i|\theta_k^{(t)}))^\beta} \log \left[\pi_l p(d_i|\theta_l) \right] \right\} \tag{4.23}$$

The DAEM algorithm as described by Ueda and Nakano [Ueda and Nakano, 1998] is:

1. Set min and max inverse temperature parameter values: β_{min} and β_{max}

2. Set $\beta = \beta_{min}$ and choose a random initial parameter vector value $\theta^{(0)}$

3. Iterate the following two steps until convergence:

    - **E-step:** Compute $Q(\theta|\theta^{(t)}; \beta)$
    - **M-step:** Find $\theta^{(t+1)} = \arg\max_{\theta} Q(\theta|\theta^{(t)}; \beta)$

4. Increase β

5. If $\beta < \beta_{max}$, $t \leftarrow t + 1$, go back to step 3.

In the GMM case, the M-step consists in updating mixing proportions π_l, mean vectors μ_l and covariance matrices Σ_l of the Gaussian components according to the recursive formulas 4.18, 4.13, and 4.15. The DAEM algorithm has been shown in Geman et al. [Geman and Geman, 1993] to converge to a global maximum of the likelihood function provided that the cooling scheme is sufficiently slow (but in practice, the "sufficiently slow" is often unacceptable for computational reasons). Because of this, the DAEM algorithm will still have a strong dependency on the initial parameter values $\theta^{(0)}$. The mass-constrained algorithm introduced by Rose in Rose et al. [Rose et al., 1993] starts from a very simple observation: at very high temperature, the distribution f_Z becomes uniform, that is, $\forall i$, $p(z_i^1 = 1|d_i, \theta) = \cdots = p(z_i^L = 1|d_i, \theta) = \frac{1}{L}$. By looking at the recursive update formulas 4.18, 4.13, and 4.15, it is easy to see that we will then have one single effective cluster which contains all d_i. On the contrary, when the temperature is close to zero, the system tends towards a nearest neighbor approach, that is, each data point d_i is assigned to the nearest cluster (the distance considered here is the one introduced above: $d(d_i, c_l)$) with probability one ($\max_l p(z_i^l = 1|d_i, \theta) = 1$). We can then fix an arbitrary number L of clusters ($L < N$) and still obtain the same number L of effective (distinct) clusters. Between these two extreme cases, it has been shown in Rose et al. that the system is subject to a sequence of phase transitions which correspond to natural splitting of clusters. That is, as we lower the temperature, the number of effective clusters grows via splitting of clusters obtained during previous steps of the procedure. This phenomenon can be explained from a heuristic point of view: the temperature T can be seen as a scale parameter in the probabilities $p(z_i^l = 1|d_i, \theta) = \frac{1}{Z_{d_i}} \exp(-\frac{d(d_i, c_l)}{T})$ which determines the scope of the influence of cluster c_l over neighboring data points. As we lower T, we reduce this scope (i.e the influence of c_l becomes more localized) so we need new clusters around c_l to explain the data points which are no longer explained by c_l. A natural solution to do so is to split c_l (in practice, we create an other cluster by introducing a small perturbation to the initial cluster c_l). The interest of what precedes lies in the fact that critical temperatures which corresponds to phase transitions (cluster splitting) can be identified and even computed analytically in certain cases. This allows us to considerably accelerate the annealing process: we can indeed adopt a faster cooling schedule between phase transitions without affecting the performance. We only need to be very careful around the critical temperatures by lowering the temperature very progressively.

The mass-constrained algorithm relies on these observations to progressively increase the number of clusters and accelerate the cooling in the temperature intervals where nothing happens, that is, between phase transitions. We avoid thus the "sufficiently slow" cooling schedule imposed by the DAEM algorithm to ensure convergence towards a global maximum. In the following, we briefly evoke the mass-constrained algorithm introduced by Rose et al.. In the contribution part, we propose a modified version of this algorithm which allows to update both cluster means and covariance matrices.

Mass-constrained clustering algorithm The mass-constrained algorithm as described in [Rose et al., 1993] is a clustering algorithm that finds cluster centroids without as-

suming a particular underlying statistical model of the data (there is still a Gaussian hypothesis regarding the shape of clusters). The final clustering is given by the association probabilities $p(d_i|c_l) = \frac{p(c_l)p(c_l|d_i)}{p(d_i)} = \frac{\pi_l \exp(-\frac{d(d_i,c_l)}{T})}{N \cdot Z_{d_i}}$. The chosen distance measure d is often the Euclidean distance which makes the mass-constrained algorithm similar to a deterministic annealing K-means algorithm. The formulation of the algorithm proposed by Rose is the following:

1. Fix the maximum number of clusters L_{max} and the minimum temperature T_{min}.

2. Choose $T > 2\lambda_{max}$, $L = 1$, $\mu_1 = \frac{1}{N}\sum_{i=1}^{N} d_i$ and $\alpha_1 = p(c_1) = 1$.

3. Iterate the following two steps until a convergence criterion is met:

   - **E-step:** Compute the association probabilities according to $p(c_l|d_i) = \frac{\exp(-\frac{d(d_i,c_l)}{T})}{Z_{d_i}}$

   - **M-step:** Update the mixing proportions and the cluster centers according to: $\pi_l = \frac{\sum_{i=1}^{N} p(c_l|d_i)}{N}$, and $\mu_l = \frac{\sum_{i=1}^{N} p(c_l|d_i)d_i}{\sum_{i=1}^{N} p(c_l|d_i)}$

4. If $T < T_{min}$, stop.

5. Cooling step: decrease T.

6. If $L < L_{max}$, check the condition for phase transition (i.e cluster splitting) for $l = 1, \cdots, L$. If the condition is met for cluster j, add a new cluster c_{L+1}. Set $\mu_{L+1} = \mu_j + \delta$, $\pi_{L+1} = \pi_j/2$ and $\alpha_j = \alpha_j/2$. Increment L. Go to step 3.

The problem of such an algorithm is that it becomes quite ineffective when the volume of data is high due to the exponential term that we have to assess for each data point in the E-step. Several heuristics have been proposed – mainly derived from the K-means algorithm – which try to avoid convergence towards local optima of the objective function. In the following paragraphs, we leave aside the statistical viewpoint and ML estimators to focus on least squares approaches which differ from the previous methods in the sense that they make no clear statistical assumptions on the data. The simplest one is the K-means algorithm [MacQueen et al., 1967] which we expose next.

K-means   The purpose of this algorithm is to minimize the sum of intraclass distortions, i.e., using the same notations as above, to minimize the objective:

$$F_{\text{obj}}(D) = \sum_{l=1}^{L} \sum_{i=1}^{N} z_i^l d(d_i, c_l) \tag{4.24}$$

where z_i^l is a point-cluster assignment variable which equals 1 if $\text{argmax}_k d(d_i, c_k) = l$ and 0 otherwise (in other words, each data point is assigned to the closest cluster according to the metric d). A two-step iterative procedure is used to compute the cluster centers:

   - Iterate the following two steps until a convergence criterion is met:

- (Re-)compute cluster centers according to $\mu_l = \frac{\sum_{i=1}^{N} z_i^l d_i}{\sum_{i=1}^{N} z_i^l}$.

- (Re-)compute point-cluster assignments: set $z_i^l = 1$ if $\arg\max_k d(d_i, c_k) = l$ and $z_i^l = 0$ otherwise.

The problem of the iterative procedure is that it converges towards a local minima of the objective function F_{obj}. The Enhanced LBG (ELBG) algorithm is a very nice heuristic which has been proposed by Patane et al. [Patané and Russo, 2001] to try to make the iterative standard K-means procedure converge towards a better optimum. The procedure is detailed in the next paragraph.

ELBG   As said in section 3.4.1, the ELBG optimizes the same functional as the K-means algorithm using roughly the same two-step iterative procedure but with an extra-strategy so that each cluster makes an equal contribution to the total amount of distortion. This strategy is inspired from the Gersho's theorem ([Gersho, 1979]) which states that: "*Each cell makes an equal contribution to the total distortion in optimal vector quantization with high resolution*". This theorem is valid only for quantizers whose number of codewords tends to infinite but it has been shown experimentally (Chinrungrueng et al. [Chinrungrueng and Sequin, 1995]) that it also retained some validity when the codebook has a finite number of elements.

To assess the contribution of each cluster to the total distortion, Patane et al. have introduced a measure U of the "utility" of a cluster:

$$U_l = \frac{D_l}{D_{\text{mean}}} \text{ with } D_l = \sum_{i=1}^{N} z_i^l d(d_i, c_l) \text{ and } D_{\text{mean}} = \frac{1}{L} \sum_{l=1}^{L} D_l \tag{4.25}$$

The idea of the ELBG is to obtain the equalization of the cluster utilities by performing simultaneously two kinds of operations: (1) merging a low-utility (lower than 1) cluster with a cluster adjacent to it in order to obtain a cluster whose utility is closer to 1 than before and (2) splitting a high-utility (higher than 1) cluster into two smaller ones whose utilities are also closer to 1 than the original high-utility cluster. One iteration of the ELBG algorithm, besides the two standard operations performed in the K-means procedure, consists in doing several shifts of clusters such as described above and checking whether the performed shifts result in a lower mean square error (total distortion) or not. In the following, the operations (1) and (2) are referred to as Shift of Cluster Attempts (SoCA). A SoCA is performed on two clusters, a cluster c_{l_1} such as $U_{l_1} < 1$ (operation (1)) and a cluster c_{l_2} such as $U_{l_2} > 1$ (operation (2)).

The low-utility clusters c_{l_1} are taken in a sequential way whereas the high-utility clusters c_{l_2} are drawn randomly according to their weights w_{l_2} computed as the normalized utilities over all the current low-utility clusters: $w_{l_2} = \frac{U_{l_2}}{\sum_{l_k:U_{l_k}>1} U_{l_k}}$. A SoCA operation involves three clusters: a low-utility cluster c_{l_1}, the adjacent cluster c_{l_3} it is to be merged with, and a high-utility cluster c_{l_2} which is to be split in two. Concretely speaking, the SoCA operation consists in merging c_{l_1} and c_{l_3} and to recompute the new cluster centroid (recomputing the whole partition is too costly so the new centroid is determined using only the elements of c_{l_1} and c_{l_3} before the merging operation). Then, we perform the splitting of c_{l_2} using a very simple strategy: the longest diagonal of the

cluster bounding box is divided into three parts, the central part being twice as long as the two other ones. The new cluster centers (i.e. the centers of the clusters resulting from the splitting operation) are then placed at the two ends of the central part. At last, we recompute the centroids of the two obtained cluster using only the elements from the original high-utility cluster. The SoCA is confirmed iff $D_{\text{new}} < D_{\text{old}}$ where D_{new} is the new total amount of distortion. One iteration of the ELBG algorithm consists in performing SoCAs until there is no low-utility cluster left.

The procedure is summarized by the Algorithm 1.

Algorithm 1 Algorithmic description of the ELBG procedure

- Initialize the cluster centroids

While the convergence criterion is not met **do**

- Compute the new partition according to the current estimates of the cluster centroids

While there is at least one cluster whose utility is lower than 1 **do**

- Select two clusters c_{l_1} and c_{l_2} such as $U_{l_1} < 1$ and $U_{l_2} > 1$
- Perform a SoCA operation on the clusters c_{l_1} and c_{l_2}
- Re-compute the centroids of the clusters resulting from the merging and splitting operations in the SoCA

If $D_{\text{new}} < D_{\text{old}}$ **then**

- Confirm the SoCA operation

Else

- Cancel the SoCA operation

End If

End While

- Compute the new cluster centroids

End While

Clustering and feature extraction   Clustering operations are often used to generate signatures under the form of visual words histograms. To obtain such representations from images, a codebook is first generated using a vector quantization/clustering over the whole database. The elements which are quantized are directly the pixel values plus possibly extra information at the pixel level computed using a filtering process (such as Gabor filtering if we want for instance textural information in addition to the color information provided by the original pixel values). After the extraction of the codebook, an histogram is computed from each image by mapping each pixel to the closest codebook element, i.e., if v_{xy} is the computed value of the pixel of coordinates (x, y) and $\{c_1, \dots, c_K\}$ the obtained codebook, we compute the associated bag-of-words representation of an image as the vector $\{h_1, \dots, h_K\}$ where $h_k = \frac{1}{N_x \cdot N_y} \sum_x \sum_y \sum_{k=1}^{K} \delta_{v_{xy}}^{c_k}$. The function $\delta_{v_{xy}}^{c_k}$ equals 1 if c_k is the codebook element which is the closest to v_{xy} and 0 otherwise. The term $\frac{1}{N_x \cdot N_y}$ is a normalization term to ensure that the histogram bins sum up to one.

4.3   Supervised classification

Given a training set of the form $\{(x_1, y_1), \dots, (x_N, y_N)\}$, a supervised learning algorithm seeks a function $g : X \to Y$, where X is the input space and Y the output space. The function g belongs to a space G of possible functions called the hypothesis space. In many probabilistic models such as logistic regression for instance, g takes the form of a conditional probability model, i.e., $g(x) = P(y|x)$. There are two basic approaches to choosing the function g: Empirical Risk Minimization (ERM) and Structural Risk Minimization (SRM). The ERM principle consists in finding the function that best fits the training data. SRM uses the same principle as ERM but adds a penalty function which controls the tradeoff between bias and variance (which in classification is often referred to as the tradeoff between goodness-of-fit and generalization capabilities).

Support Vector Machine (SVM) from the SRM point of view   SVMs are directly inspired from the SRM principle and their formulation allows to control directly the bias/variance tradeoff. In this paragraph, we will only envisage the SVM formalism used in classification though there exists an extension to the regression case as well.

SVMs look for a function g of the form $g(x) = \text{sign}(f(x))$ where f is called the decision function and can be expressed as a linearly-weighted sum of possibly non-linear basis functions $\phi(x) = \{\phi(x_1), \dots, \phi(x_M)\}^T$, i.e., $f(x; w, b) = \sum_{i=1}^{M} w_i \phi(x_i) + b = w^T \cdot \phi(x) + b$. From the geometrical point of view, it amounts to computing a separating hyperplane in a transformed space defined by the mapping function ϕ. Solving the SVM problem consists then in finding "good values" for the parameters $w = \{w_1, \dots, w_M\}$ and b which are the hyperplane parameters, w being the normal to the hyperplane defining its direction and b being the offset to the origin.

The SVM optimization problem can then be derived directly from the SRM principle: we want to obtain a classifier that correctly fits/classifies the training data but which is not too complex to avoid overfitting. This tradeoff is the main point of the statistical learning theory (also known as Vapnik-Chervonenkis theory [Vapnik, 1998]) inside which the SRM principle is formulated. The goodness-of-fit is measured via the empirical risk which is simply taken as the mean error rate of the classifier on the training dataset: $R_{emp}(w, b) = \frac{1}{2N} \sum_{i=1}^{N} |y_i - g(x_i; w, b)|$. To assess the generalization capabilities of the classifier, one has to compute the generalization error on unseen data, that is, the expectation of the test error: $R(w, b) = \int \frac{1}{2} |y - g(x; w, b)| p(x, y) dx dy$. This quantity, which is the one we are ultimately interested in is of course not directly tractable and we have to resort to a result obtained by Vapnik et al. which establishes a very nice connection between the empirical risk and the expected risk. It states that the following upper bound holds with probability $(1 - \eta)$ where $0 \le \eta \le 1$:

$$R(w, b) \le R_{emp}(w, b) + \sqrt{\left(\frac{h(\log(2N/h) + 1) - \log(\eta/4)}{N} \right)} \tag{4.26}$$

In this equation, h is a non-negative integer called the Vapnik-Chervonenkis (VC) dimension. It is a measure of the "capacity" of the classifier, that is, its ability to learn any training set without error. More concretely speaking, given a certain class of classifiers (such as linear classifiers for instance), the VC dimension is the maximum number
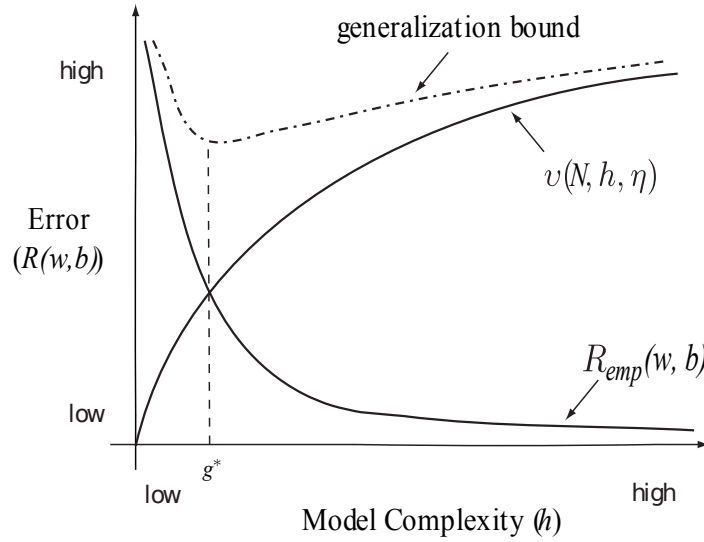
Figure 4.1: Relationship between the empirical risk $R_{emp}(w, b)$ and the VC-confidence $v(N, h, \eta)$. As the complexity of the model increases, the empirical risk decreases: it means that complex models will yield a better modeling of the data (with a risk of overfitting). On the contrary, as the complexity of the model increases, so does the VC-confidence: it means that complex models will commit more errors on data not contained in the training dataset. The generalization bound is the sum of the two other curves. Minimizing the generalization bound/expected risk is equivalent to making the right tradeoff between the goodness-of-fit/empirical risk and the complexity/generalization capabilities of the classifier.

of points that can be correctly classified by a member of this class, whatever their labeling is. In other words, we'll say that the VC dimension associated with a class of classifier C is h iff we can find h data points (and no more than h) for which there exists at least one member of C which correctly classifies these points (i.e. respects their labels) for each of the 2^h possible labelings over these points (it does not have to be of course the same member of the class which correctly classifies the data points for all the 2^h possible labelings).

The SRM principle consists then in minimizing the upper bound on the expected risk to minimize the expected risk itself. The upper bound is the sum of the empirical risk and of a second term called the VC confidence (see the right hand side of equation 4.26) which is a monotonic increasing function of h. Thus for a selection of classifiers whose empirical risk is zero, one wants to select the classifier which belongs to the class having the lowest VC dimension since this classifier will yield a better upper bound on the expected risk. This of course does not guarantee the chosen classifier to have better performance than an another classifier belonging to a class of higher VC dimension. Eq. 4.26 is only used as a guide to determine a classifier whose expected risk does not exceed the upper bound (with some chosen probability $(1 - \eta)$).

In the following, we look closer to how the SRM principle is concretely implemented in the SVM formalism. We can start from the simple observation that classifiers with small margins and large VC-dimensions will yield accurate but complex decision sur-

faces whereas classifiers with large margins and small VC-dimensions will induce less accurate but at the same time less complex decision surfaces.

The relation between SRM and SVM is established in the separable case by a theorem from Vapnik [Vapnik, 1998] which states that minimizing the norm of w in the SVM problem is equivalent to minimizing the VC dimension h and consequently the "capacity" term which is a growing function of h. The exact formulation of this theorem is the following:

Let D be the diameter of the smallest ball around the training data points $x_1,\ldots,x_N$. Considering the class of separating hyperplanes described by the equation $w^T x + b = 0$, the upper bound to the VC dimension is

$$VC \leq \min\left(\lceil \frac{D^2}{m^2} \rceil, M\right) + 1$$

In the above inequality, M corresponds to the dimension of the input space and m is the margin of the separating hyperplane that has the smallest margin within the class of separating hyperplanes of the form $w^T x + b = 0$. Thus, minimizing the expected risk $R(w, b)$ is equivalent to maximizing m while keeping the empirical risk $R_{emp}(w, b)$ low or zero. By requiring the support vectors to lie on the hyperplanes $w^T x + b = -1$ and $w^T x + b = +1$, the margin of the separating hyperplane can be shown to be $\frac{2}{\|w\|}$ (see next paragraph). Thus, we can see that the SVM plainly exploits the SRM principle when it minimizes $\frac{1}{2}\|w\|^2 = \frac{2}{m^2}$ under the constraints:

$$(w^T x_i + b) \geq +1 \text{ for } y_i = +1 \tag{4.27}$$

$$(w^T x_i + b) \leq -1 \text{ for } y_i = -1 \tag{4.28}$$

These two sets of constraints ensure that the empirical risk is kept low or zero. The SRM principles applies in the same way when using a mapping function ϕ to transpose the SVM problem into a high-dimensional feature space.

Support Vector Machine from the geometrical point of view Support Vector Machines can also be formalized from a mere geometrical point a view. We consider first the linear case, i.e. we look for a separating hyperplane of the form $w^T x + b = 0$. The SVM algorithm simply looks for the parameters w and b which maximize the margin.

We first envisage the separable case, that is the case where the training data can be separated by a linear classifier without classification errors. In this simple setting, the maximum margin is defined as the distance between the parallel hyperplanes that are as far apart from each other as possible while still correctly separating the training data. This imposes the contraints:

$$(w^T x_i + b) \geq +1 \text{ for } y_i = +1 \tag{4.29}$$

$$(w^T x_i + b) \leq -1 \text{ for } y_i = -1 \tag{4.30}$$

which can be rewritten: $y_i(w^T x_i + b) \geq 1, \forall i = 1,\ldots,N$. Let's consider first the points for which the inequality 4.29 holds (requiring that there exists such a point is just a matter of correctly scaling w and b). These points lie on the hyperplane H_1 : $w^T x + b = 1$.

Similarly, the points for which the inequality 4.30 holds lie on the hyperplane H_2 : $w^T x + b = -1$. The perpendicular distances from H_1 and H_2 to the origin are then respectively $|1 - b|/||w||$ and $|-1 - b|/||w||$. Hence, the distance between H_1 and H_2 is $(|1 - b|/||w||) - (|-1 - b|/||w||) = 2/||w||$. Maximizing the margin thus amounts to maximizing the quantity $2/||w||$ which is equivalent to minimizing the quantity $\frac{1}{2}||w||^2$. The SVM problem in the linear separable case can then be formulated as:

$$\min_{w,b} \frac{1}{2}||w||^2 \tag{4.31}$$

$$\text{s.t. } y_i(w^T x_i + b) \geq 1, \forall i = 1, \ldots, N \tag{4.32}$$

The above optimization problem when applied to non-separable data possesses no feasible solution. The idea is to relax the constraints 4.29 and 4.30 to allow the misclassification of some points when necessary. To translate this idea, an extra cost is added to the objective function of the SVM problem 4.31 and *slack variables* $\xi_1, \ldots, \xi_N$ are introduced into the constraints 4.29 and 4.30 to account for eventual misclassifications. The new constraints can then be formulated :

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \forall i = 1, \ldots, N$$
$$\xi_i \geq 0, \forall i = 1, \ldots, N$$

For a misclassification to occur, the corresponding slack variable must exceed 1. Thus, the quantity $\sum_{i=1}^{N} \xi_i$ is a lower bound on the number of training errors. A natural way to integrate the misclassification so that they penalize the SVM objective function is then simply to add an extra term to the objective 4.31, yielding a new objective function of the form: $\frac{1}{2}||w||^2 + C\sum_{i=1}^{N} \xi_i$. The SVM problem in the linear non-separable case can then be written in the following way:

$$\min_{w,b,\xi_i} \frac{1}{2}||w||^2 + C\sum_{i=1}^{N} \xi_i$$
$$\text{s.t. } \begin{cases} y_i(w^T x_i + b) \geq 1 - \xi_i, \forall i = 1, \ldots, N \\ \xi_i \geq 0, \forall i = 1, \ldots, N \end{cases} \tag{4.33}$$

The parameter C is fixed by the user and allows to control the amount of classification errors by penalizing models with a lot of such errors. A high value of C amounts to assigning a higher penalty to errors, leading to a model which better fit the training data but with a risk of overfitting. On the contrary, a small value of C might lead to a model which underfits the training data. Finding the right value of C is thus equivalent to finding the right tradeoff between goodness-of-fit and generalization capabilities, which is a very interesting feature of SVM classifiers.

In the following, we derive the dual formulation of the problem 4.33. The advantage of such a formulation is that the training data will only appear in the form of dot products between training vectors. This property allows the generalization to the non-linear case.

The dual formulation is obtained by writing the Lagrangian $L(w, b, \xi; \alpha, \nu)$ of the problem 4.33 and by using the Karush-Kuhn-Tucker (KKT) conditions on the Lagrangian.

The KKT conditions are necessary conditions for the solution of a constraint optimization problem to be optimal. In the case of a convex optimization problem with linear constraints, they can also be shown to be sufficient conditions for optimality. The problem 4.33 is a quadratic optimization problem with linear contraints and as such is convex. Thus, finding the optimum is equivalent to finding the point which verifies the KKT conditions. This property can be used to derive another formulation of the optimization problem 4.33 called the Wolfe dual and which is equivalent to the primal problem.

To form the Lagrangian, we introduce $2N$ Lagrange multipliers $\alpha_1, \ldots, \alpha_N, v_1, \ldots, v_N$ for the $2N$ constraints $y_i(w^T x_i + b) \geq 1 - \xi_i, \forall i = 1, ..., N$ and $\xi_i \geq 0, \forall i = 1, ..., N$, yielding:

$$L(w, b, \xi; \alpha, v) = \frac{1}{2} w^T w + C \sum_{i=1,\ldots,N} \xi_i - \sum_{i=1}^{N} \alpha_i (y_i(w^T x_i + b) - 1 + \xi_i) - \sum_{i=1}^{N} v_i \xi_i \quad (4.34)$$

For the primal problem 4.33, the KKT conditions can be written as [Fletcher, 1981]:

$$\nabla L(w, b, \xi; \alpha, v) = 0 \quad (4.35)$$

$$\alpha_i y_i(w^T x_i + b) = 0, \ \forall i = 1, \ldots, N \quad (4.36)$$

$$v_i \xi_i \geq 0, \ \forall i = 1, \ldots, N \quad (4.37)$$

$$\alpha_i \geq 0, \ \forall i = 1, \ldots, N \quad (4.38)$$

$$v_i \geq 0, \ \forall i = 1, \ldots, N \quad (4.39)$$

Using the condition 4.35, we obtain (4.40, 4.41 and 4.42):

$$\frac{\partial L(w, b, \xi; \alpha, v)}{\partial w} = w - \sum_{i=1}^{N} \alpha_i y_i x_i = 0 \quad \Rightarrow \quad w = \sum_{i=1}^{N} \alpha_i y_i x_i \quad (4.40)$$

$$\frac{\partial L(w, b, \xi; \alpha, v)}{\partial b} = \sum_{i=1}^{N} \alpha_i y_i = 0 \quad (4.41)$$

$$\left(\frac{\partial L(w, b, \xi; \alpha, v)}{\partial \xi} \right)_i = C - \alpha_i - v_i = 0 \quad \Rightarrow \quad v_i = C - \alpha_i \quad (4.42)$$

Because of (4.42), (4.34) is equivalent to:

$$L(w, b, \xi; \alpha, v) = \frac{1}{2} w^T w - \sum_{i=1}^{N} \alpha_i (y_i(w^T x_i + b) - 1) \quad (4.43)$$

By substituting 4.40 into equation 4.43 and by denoting $\langle u, v \rangle$ the scalar product $u^T v$, we obtain:

$$L(w,b,\xi;\alpha,v) = \frac{1}{2}\langle \sum_{i=1}^{N}\alpha_i y_i x_i, \sum_{i=1}^{N}\alpha_i y_i x_i \rangle - \sum_{i=1}^{N}\alpha_i(y_i(\langle \sum_{j=1}^{N}\alpha_j y_j x_j, x_i \rangle + b) - 1)$$

$$= \frac{1}{2}\sum_{i,j=1}^{N}y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle - \sum_{i=1}^{N}\alpha_i(y_i(\sum_{j=1}^{N}\alpha_j y_j \langle x_j, x_i \rangle + b) - 1)$$

$$= \frac{1}{2}\sum_{i,j=1}^{N}y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle - \sum_{i=1}^{N}\alpha_i(\sum_{j=1}^{N}\alpha_j y_i y_j \langle x_i, x_j \rangle + y_i b) + \sum_{i=1}^{N}\alpha_i$$

$$= \frac{1}{2}\sum_{i,j=1}^{N}y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle - \sum_{i,j=1}^{N}y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle - \left(\sum_{i=1}^{N}\alpha_i y_i\right)b + \sum_{i=1}^{N}\alpha_i$$

Using the equation 4.41, we finally obtain:

$$L(w,b,\xi;\alpha,v) = \sum_{i=1}^{N}\alpha_i - \sum_{i,j=1}^{N}y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle \qquad (4.44)$$

The KKT conditions 4.38 and 4.39 impose that $\alpha_i \geq 0, \forall i = 1,\ldots,N$ and $v_i \geq 0, \forall i = 1,\ldots,N$. Since minimizing the primal is equivalent to maximizing the dual in the case of a convex optimization problem, we finally obtain the following quadratic optimization problem:

$$\max_{\alpha} L(\alpha,v) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N}y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle$$

$$\text{s.t. } \begin{cases} \alpha_i \geq 0, \forall i = 1,\ldots,N \\ v_i = C - \alpha_i \geq 0, \text{ i.e. } \alpha_i \leq C, \forall i = 1,\ldots,N \end{cases} \qquad (4.45)$$

This problem can then be solved by any standard Quadratic Programming (QP) solver. To compute b, we use the KKT condition 4.36, which yields:

$$\alpha_i(y_i(\sum_{j=1}^{N}\alpha_j y_j \langle x_j, x_i \rangle + b) - 1) =$$

$$\sum_{j=1}^{N}\alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + b\alpha_i y_i - \alpha_i = 0, \ \forall i = 1,\ldots,N$$

$$\Rightarrow \quad b = -y_i \sum_{j=1}^{N}\alpha_j y_i y_j \langle x_i, x_j \rangle + y_i, \ \forall i \text{ such as } 0 < \alpha_i \leq C \qquad (4.46)$$

In the following, we show how the above problem can be generalized to the case where the decision function is not a linear function of the training data. Boser et al. [Boser et al., 1992] have proposed to use a "trick" based on the observation that the training data only appear under the form of dot products $\langle x_i, x_j \rangle$ in the dual formulation. The "kernel trick" as it is referred to in the literature consists in considering a

mapping of the data in another Euclidean (possibly infinite dimensional) space $\mathcal{H}$, using a mapping function which we denote by ϕ in the following: $\phi : \mathcal{R}^d \rightarrow \mathcal{H}$. The training algorithm will then only depend on the dot products $\langle \phi(x_i), \phi(x_j) \rangle$ in that space. We thus never need to know the function ϕ explicitly, the mapping being defined through the choice of a kernel function $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$. One example of commonly used kernel is the Gaussian kernel: $k(x_i, x_j) = \exp(-d^2(x_i, x_j)/2\sigma^2)$. Using the kernel trick, the optimization problem in 4.45 is transformed into:

$$\max_{\alpha} L(\alpha, \nu) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j k(x_i, x_j)$$

$$\text{s.t.} \begin{cases} \alpha_i \geq 0, \forall i = 1, ..., N \\ \nu_i = C - \alpha_i \geq 0, \text{ i.e. } \alpha_i \leq C, \forall i = 1, ..., N \end{cases} \tag{4.47}$$

yielding the following decision function: $f(x) = \sum_{i \in SV} \alpha_i y_i k(x, x_i) + b$ where SV is the set of support vectors, i.e. $SV = \{i | 0 < \alpha_i <= C\}$.

The figures 4.3 represent the SVM decision function values in the four different cases envisaged above.

"Soft" variants of SVMs   In this paragraph, we present two soft variants of the standard SVM problem allowing the use of soft-labeled training examples. We introduce first the well-known fuzzy-SVM and then we present an other formulation of a soft SVM

Fuzzy SVMs have been introduced by Lin et al. [Lin and Wang, 2002] to account for the fact that training data points in a two-class problem are not necessarily fully assigned to one class or the other but instead may be given fuzzy membership degrees. Thus, each training data point makes a different contribution to the learning of the decision surface. In the following, we denote by μ_i ($0 \leq \mu_i \leq 1$) the membership degree assigned to the training data point x_i. The value μ_i represents the degree of belief we have that the data point x_i is a representative of the class y_i. The idea is to modify the slack variables by multiplying them by the membership degrees. The slack variable ξ_i reflects indeed the amount of misclassification we authorize on the training data x_i. A high value of μ_i when multiplied with ξ_i will constrain the latter to remain low and thus will limit the classification error made in this point. On the contrary, a low value of μ_i will allow the value of ξ_i to grow higher and there will be thus less contraints on the corresponding data point x_i to be correctly classified. This amounts to giving more importance in the learning to the training data points with high membership degrees and to reduce the influence of those with low membership degrees. The formulation of the fuzzy-SVM problem is the following:

$$\min_{w, b, \xi_i} \frac{1}{2} ||w||^2 + C \sum_{i=1}^{N} \mu_i \xi_i$$

$$\text{s.t.} \begin{cases} y_i(w^T x_i + b) \geq 1 - \xi_i, \forall i = 1, ..., N \\ \xi_i \geq 0, \forall i = 1, ..., N \end{cases} \tag{4.48}$$

(a)   (b)

(c)   (d)

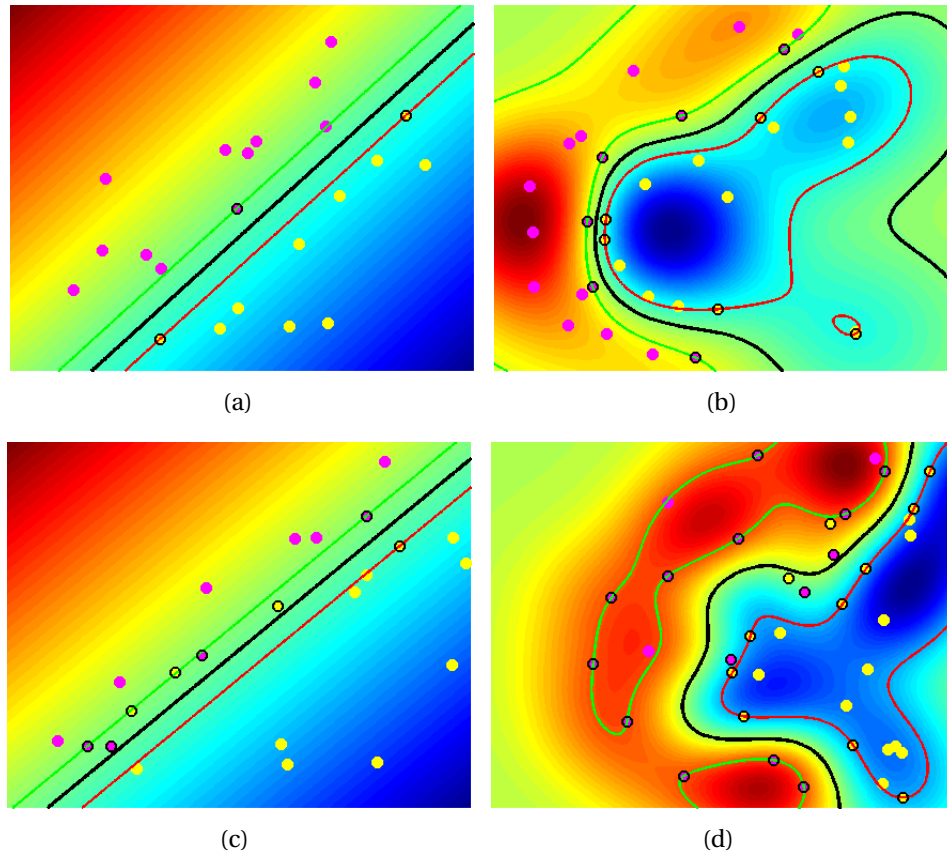Figure 4.2: Fig. 4.2(a): linear SVM in the separable case. Fig. 4.2(b): linear SVM in the non-separable case. Fig. 4.2(c): non-linear SVM in the separable case. Fig. 4.2(d): non-linear SVM in the non-separable case. In all the figures above, the circled dots represent the support vectors and the black line, the SVM separation surface. The green and red lines correspond respectively to the $+1$ and -1 margins.

which yields the following dual formulation:

$$\max_{\alpha} L(\alpha, v) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle$$

$$\text{s.t. } 0 \leq \alpha_i \leq \mu_i C, \ \forall i = 1, \ldots, N \qquad (4.49)$$

The principal limitation of the fuzzy-SVM formulation is due to the fact that the membership degree of a training sample is taken into account only when a misclassification occurs (in the contrary case, the slack variable associated with the training sample is 0, thus, the associated membership has no influence in the training). The soft-SVM formulation which we derive below [Liu, 2006] allows to take account of the membership degree no matter whether the training sample is correctly classified or not. When misclassification occurs on a given training sample, the corresponding slack variable is penalized in proportion to the membership degree associated with the sample, i.e., the less certain the membership degree is, the less "important" the misclassification will be. On the contrary, correctly classified samples are allowed to influence the separating surface by pulling it close or pushing it away depending on the value of the corresponding memberships. Thus, both correctly and non-correctly classified samples will be allowed to have an influence proportioned to their respective membership values during the training. To reflect the degree of membership of a training sample x_i, we relax the constraint $y_i(wx_i + b) \geq 1$ into $y_i(wx_i + b) \geq \mu_i$, which amounts to allow the optimal hyperplane to move closer to the samples with less certain membership degrees. In the non-separable case, we obtain the following constraint: $y_i(wx_i + b) \geq \mu_i - \xi_i$. To further penalize the misclassification of samples with high μ_i values, we replace the error term ξ_i by $\mu_i \xi_i$ as in the fuzzy-SVM problem. We finally obtain the following primal formulation of the soft-SVM problem:

$$\min_{w,b,\xi_i} \frac{1}{2} ||w||^2 + C \sum_{i=1}^{N} \mu_i \xi_i$$

$$\text{s.t. } \begin{cases} y_i(w^T x_i + b) \geq \mu_i - \xi_i, \forall i = 1, \ldots, N \\ \xi_i \geq 0, \forall i = 1, \ldots, N \end{cases} \qquad (4.50)$$

which yields the dual problem:

$$\max_{\alpha} L(\alpha, v) = \sum_{i=1}^{N} \mu_i \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle$$

$$\text{s.t. } 0 \leq \alpha_i \leq \mu_i C, \ \forall i = 1, \ldots, N \qquad (4.51)$$

An important question associated with the use of a soft-SVM classifier is how we can obtain the membership degrees associated with the training samples. The most natural way of proceeding is to envisage the problem from the probability point of view and to use the conditional probabilities $p(y_i = +1|x_i)$ and $p(y_i = -1|x_i)$. The maximal certainty about the class label is produced when $p(y_i = +1|x_i) = 1$ or $p(y_i = -1|x_i) = 1$.
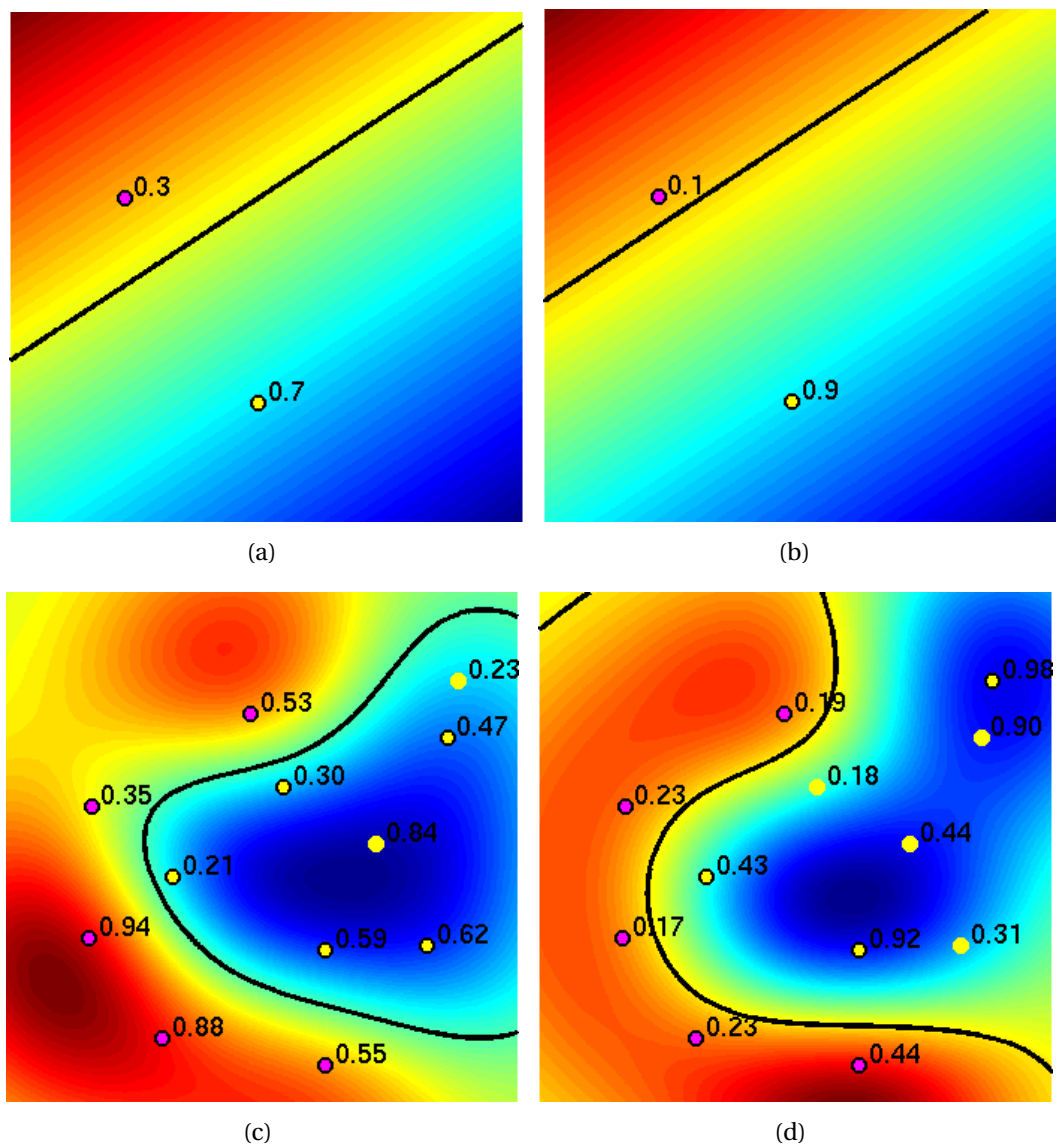
Figure 4.3: Effect of varying the membership degrees of the training data points in the case of a linear (fig. 4.3(a) and fig. 4.3(b)) and a non-linear (fig. 4.3(c) and fig. 4.3(d)) soft-SVM classifier.

In this case, $\mu_i = 1$. The maximum uncertainty corresponds to $p(y_i = +1|x_i) = p(y_i = -1|x_i) = 0.5$. In this case, $\mu_i = 0$. In the following, we set $p_i^+ = p(y_i = +1|x_i)$ and $p_i^- = p(y_i = -1|x_i)$. One can observe that the quantity $p_i^+ - 0.5$ plays the same role in the range $[-0.5, 0.5]$ that $\mu_i y_i$ in the range $[-1, +1]$. A mapping between probabilistic and fuzzy membership degrees can thus be established as: $\mu_i y_i = 2(p_i^+ - 0.5) = 2p_i^+ - 1 = p_i^+ - p_i^-$. The membership degree μ_i can thus be obtained as: $\mu_i = y_i(p_i^+ - p_i^-) = |p_i^+ - p_i^-|$ since $y_i = \text{sign}(p_i^+ - p_i^-)$.

Probabilistic outputs for SVMs   A classifier than can output a posterior probability of the form $p(\text{class}|\text{input})$ is very useful in many practical situations. The problem of SVMs is that they produce in the output an uncalibrated value which is not a probability. A common method to transform the value of the SVM decision function f into a posterior probability consists in using a sigmoid, i.e. we compute $p(y = 1|x)$ as: $p(y = 1|x) = \frac{1}{1+\exp(Af(x)+B)}$. The problem is then to adjust the parameters A and B of the sigmoid. Platt et al. [Platt, 2000] propose to perform maximum likelihood estimation from the SVM training set $\{(x_i, y_i)_{i=1,...,N}\}$. In the following, we denote by p_i the sigmoid response to x_i (i.e. $p_i = \frac{1}{1+\exp(Af(x_i)+B)}$). Considering that the variable Y_i (i.e. the random variable associated with the label of the training point x_i) follows a Bernoulli law of parameter p_i (the attribution of the label y_i is similar in nature to the result of a *yes/no* experiment with a probability of success equal to p_i and a probability of failure equal to $1 - p_i$), we have: $p(Y_i = y_i|x_i) = p_i^{t_i}(1 - p_i)^{(1-t_i)}$ with $t_i = \frac{y_i+1}{2}$, i.e. $t_i = 1$ if $y_i = 1$ and $t_i = 0$ if $y_i = -1$. Thus, the global likelihood can be written as (by assuming independence between the training set instances): $\mathcal{L}(A, B) = \sum_{i=1}^{N} p_i^{t_i}(1 - p_i)^{(1-t_i)}$. In the following, we use the negative log-likelihood:

$$L(A, B) = -\sum_{i=1}^{N} t_i \log(p_i) + (1 - t_i) \log(1 - p_i) \tag{4.52}$$

The parameters A and B are found by minimizing $L(A, B)$, i.e.: $(A, B) = \arg\min_{(A,B)} L(A, B)$. This problem is a two-parameter non-linear estimation problem which can be solved using any optimization algorithm (Levenberg-Marquardt ...). Platt et al. have proposed their own algorithm which is described in [Platt, 2000].

The negative log-likelihood 4.52 can also be seen as the Kullback-Leibler divergence between p_i and t_i. Thus, we can imagine that the variable t_i does not only take binary values. This is useful when we want to use a model of "out-of-sample" data (we can for instance impose the distribution of the t_i's).

Supervised methods and learning strategies   In this paragraph, we describe three different settings for supervised learning. We start with Multiple Instance Learning (MIL). We then give an insight into active learning and cascaded learning strategies. We focus each time on the use of SVMs within these three particular learning frameworks.

Multiple Instance Learning (MIL)   Multiple Instance Learning refers to a learning framework where there is an uncertainty on the labels of training instances: in the MIL setting, training data is available under the form of bags of instances with labels for the bags. Thus, the problem is to learn the target concept given positive and negative bags

of instances. Each bag may contain one or several instances. A bag is labeled as positive if and only if there is at least one positive instance within the bag (i.e. one instance which falls within the targeted concept). A bag is labeled as negative if and only if all the instances it contains are negative (i.e. do not fall within the targeted concept). The goal of the learning is to induce a classifier able to label individual instances correctly. In the following, we introduce a common reformulation of Multiple Instance Learning as a maximum margin problem using SVMs. We refer to this approach as MIL-SVM [Andrews et al., 2003]. We will use the following notations: $x_1, \ldots, x_N$ refer to the training instances which are grouped into M bags $B_1, \ldots, B_M$ where $B_m = \{x_i | i \in I_m\}$ and I_m is a subset of $\{1, \ldots, N\}$. We assume that the sets $I_1, I_2, \ldots, I_M$ form a partition of the set $\{1, \ldots, N\}$ and are non-overlapping. We denote by Y_m the label associated with the m-th bag. The maximum margin formulation of Multiple Instance Learning can be written as the following mixed integer programming problem:

$$
\min_{\{y_i\}_{i=1,\ldots,N}} \min_{w,b,\xi} \frac{1}{2}||w||^2 + C \sum_{i=1}^{N} \xi_i
$$

$$
\text{s.t.} \begin{cases}
y_i(\langle w, \phi(x_i) \rangle + b) \geq 1 - \xi_i, \ \forall i = 1, \ldots, N \\
\xi_i \geq 0, \ \forall i = 1, \ldots, N \\
\sum_{j \in I_m} \frac{y_j + 1}{2} \geq 1 \ \forall m \text{ s.t. } Y_m = 1 \\
y_j = -1 \ \forall j \in I_m \ \forall m \text{ s.t. } Y_m = -1
\end{cases} \tag{4.53}
$$

The problem is thus to discover the labels $y_1, \ldots, y_N$ of single instances. The third constraint implies that there is at least one positive instance per positive bag and the fourth constraint implies that all the instances inside a negative bag have a negative label. The MIL problem taken as such is thus a combinatorial problem which can be solved by a brute force approach which consists in testing all the possible labelings inside the positive bags with the constraint that there is at least one positive element per positive bag. By denoting N_m the cardinal of the m-th bag, we have thus

$$
\binom{N_m}{1} + \binom{N_m}{2} + \cdots + \binom{N_m}{N_m} = 2^{N_m} - 1
$$

possible labelings to test per positive bag. In the end, the MIL-SVM problem 4.53 amounts to training $\prod_{m=1}^{M} \left[2^{N_m} - 1 \right]$ SVM problems and to select the one which gives the largest margin. There exists of course approximate solutions in the literature which do not require to solve the combinatorial problem as such but they all yield suboptimal solutions.

Active learning   In section 3.4.3, we mentioned that active learning could be envisaged from a machine learning point of view, the goal of the learner being to learn the user's query concept. We remain close to this idea in the following by considering active learning as a way to build the training set of a two-class classifier using an interactive query process involving the user.

Using this definition, the main issue of active learning becomes to select the most informative images to present to the user from the current pool of unlabeled images.

The term "active" refers indeed to a strategy of "feedback images" selection which aims at minimizing the number of iterations in the active learning loop. Active learners differ thus from "passive" learners which use random selection for the choice of feedback examples. In an active learning approach, this choice is performed according to the user response on the set of images selected during the previous active learning iterations. As mentioned before, an active learning learner must ideally fulfill two goals, which are: (1) to learn the user's query concept as quickly as possible and (2) as accurately as possible. In the following, we describe SVM$_{\text{active}}$ which is a state-of-the-art tool to perform active learning using an SVM classifier. In this algorithm, the most informative samples are computed as the samples among the current pool of unlabeled samples which are the closest to the current SVM separating surface. These samples are termed Most Ambiguous (MA) in the following. For a feedback scheme involving one feedback sample at a time, this strategy has been proved to be optimal in terms of speed (the notion of "speed" being defined as the number of iterations needed in the active learning loop to arrive at a proper definition of the targeted concept using an SVM classifier). A theoretical justification is provided in [Tong and Chang, 2001]. It relies on the notion of version space which is defined as the set of hyperplanes that correctly separate the training data in the feature space induced by the kernel function. By denoting $\mathcal{H}$ the set of possible hyperplanes (i.e. $\mathcal{H} = \left\{ f \mid f(x) = \frac{\langle w, \phi(x) \rangle}{||w||}, \ w \in \mathcal{F} \right\}$ where $\mathcal{F}$ is the feature space), we can define the version space $\mathcal{V}$ as: $\mathcal{V} = \left\{ f \in \mathcal{H} \mid y_i f(x_i) > 0 \ \forall i = 1, \dots, N \right\}$. There exists a bijection between the set of possible hypotheses in $\mathcal{H}$ and the set of unit vectors w in $\mathcal{H}$. Thus, the version space can be rewritten as: $\mathcal{V} = \left\{ w \in \mathcal{H} \mid ||w|| = 1 \text{ and } y_i f(x_i) > 0 \ \forall i = 1, \dots, N \right\}$. We have to notice that the notion of version space exists only if the training data is linearly separable in the feature space. In the following, we start from the idea that observing a training point x_i restrains $\mathcal{V}$ to the set of hyperplanes that correctly classifies x_i, i.e. to the set of hyperplanes that satisfy $y_i \langle w, \phi(x_i) \rangle > 0$. By rewriting the new constraint brought by x_i under the form $\langle w, y_i \phi(x_i) \rangle > 0$, we can view $y_i \phi(x_i)$ as the normal to an hyperplane in the "normalized" feature space. Thus, the constraint $y_i \langle w, \phi(x_i) \rangle > 0$ defines a half-space in the "normalized" feature space and $y_i \langle w, \phi(x_i) \rangle = 0$ corresponds to an hyperplane which acts as a boundary of the version space. The latter can be seen at the beginning of the active learning process (when there exists no constraint) as a unit hypersphere. According to what precedes, adding a new training point divides the hypersphere into two parts. With these considerations in mind, we can go back to the problem at hand. Intuitively, we can easily figure that the winning search strategy will be the one which halves the current version space at each active learning iteration, thus reducing by two the uncertainty about the "target hypothesis" (i.e. the separating surface delineating the user's query concept). Thus, at each active learning iteration, one has to choose the sample in the pool of unlabeled data that is the closest to the center of the current version space. To do this, one can start from the observation that the solution of the SVM problem trained with the current training dataset will be the point of the current version space such that the hypersphere centered at this point and which does not intersect with the hyperplanes corresponding to the current training instances has the largest radius possible. The resulting hypersphere roughly approximates the version space, and thus, has its center close to the center of the current version space. Choosing the next feedback sample is then somewhat equivalent to choos-
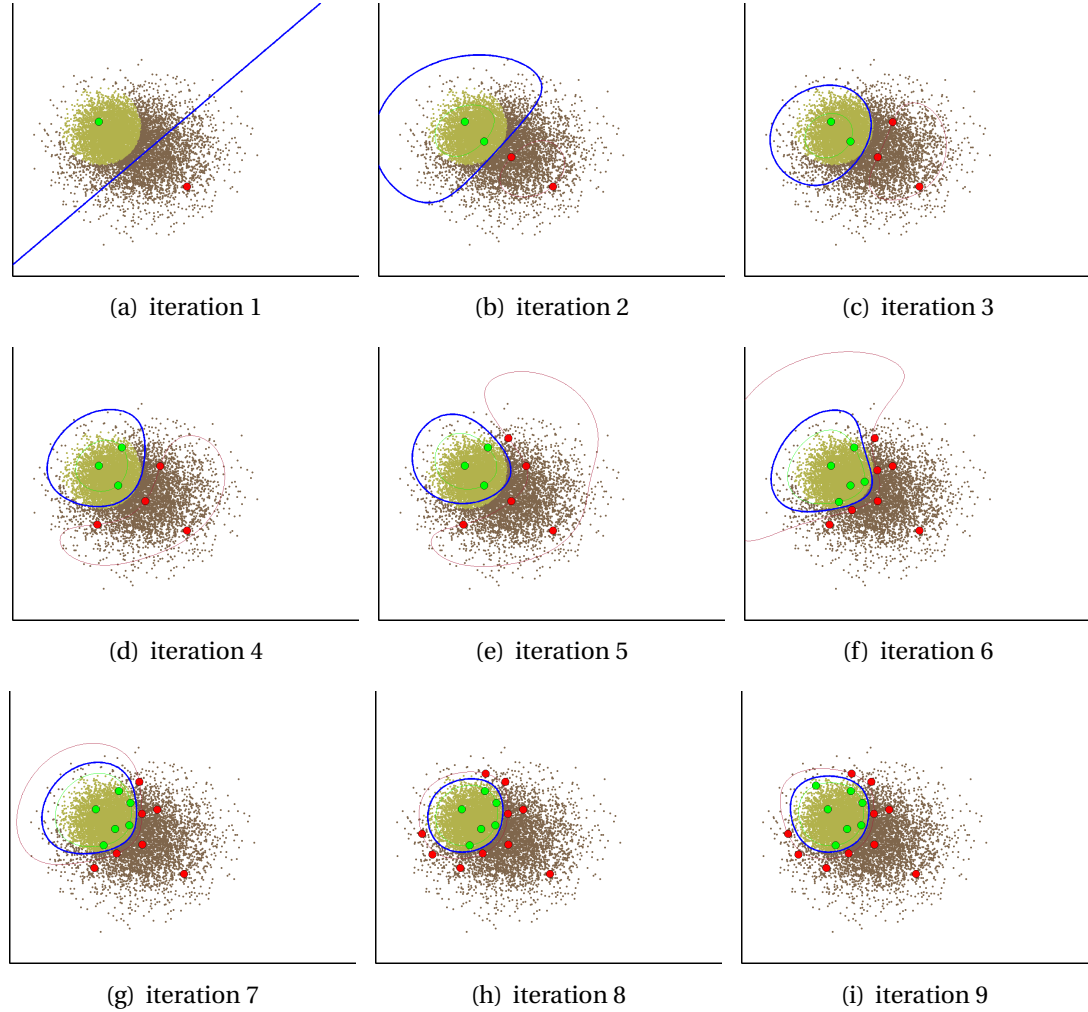
Figure 4.4: Iterations of the SVM$_{\text{active}}$ algorithm. The green and red dots refer respectively to positive and negative feedback examples and the blue line to the SVM separating surface.

ing the sample whose corresponding hyperplanes comes closest to the center w_t of this hypersphere, i.e. we might want to choose the sample $x_t = \arg\min_{x_t \in U} \left| w_t \cdot \phi(x_t) \right|$ where U is the pool of unlabeled samples. The sample x_t will also be the point whose image in the feature space will be the closest from the current SVM separating surface defined by w_t. Thus, at each active learning iteration, the strategy will be to return the point belonging to the pool of unlabeled data whose image in the feature space is the closest to the current SVM separating surface.

Cascaded learning   In this paragraph, we shortly describe the Viola and Jones [Viola and Jones, 2004] real-time object detection framework. The proposed scheme relies on a coarse-to-fine strategy implemented as a cascade of detectors of increasing complexity/discriminating power as we go down the cascade. The purpose is to

Figure 4.5: Synopsis of a cascaded active learning scheme: sub-windows which are "accepted" at one stage of the cascade are propagated to the next stage for further processing, the other sub-windows are simply discarded. This diagram is directly inspired from the one in [Viola and Jones, 2004].
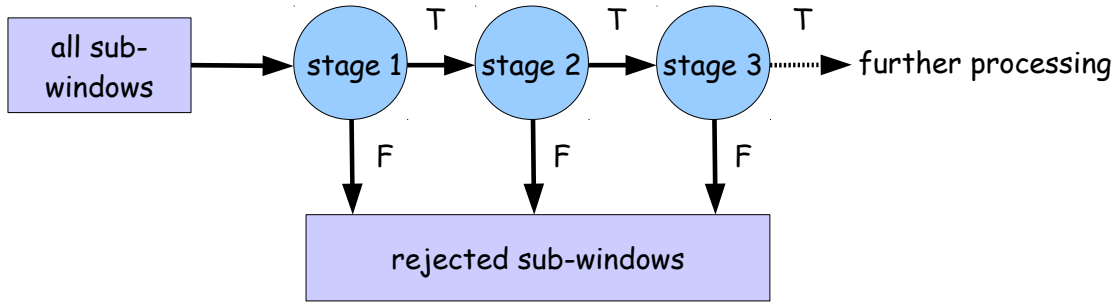
eliminate a large number of sub-windows very unlikely to contain the targeted object in the higher stages of the cascade with the help of very cheap detectors. We can thus focus on a reduced number of sub-windows in the lower stages, allowing the use of more complex/costly detectors on these sub-windows without "infringing" the real-time constraint. The processing of a sub-window thus goes on as soon as it it is rejected at some stage of the cascade, in which case, the sub-window will not be "propagated" to the lower stages (see figure 4.5). To be more specific about the Viola et Jones object detection framework, a boosted version of an SVM classifier is trained by progressively introducing new features at each stage of the cascade. In the face detection application described in [Viola and Jones, 2004], the first classifier in the cascade uses only two features but still allows to eliminate around half the database with an almost null false negative rate and a false positive rate of 40%.

Generally speaking, due to the cascaded structure, the detection rate of an L-stage cascade is $\prod_{l=1}^{L} d_l$ where d_l is the detection rate at stage l and similarly, the false positive rate is $\prod_{l=1}^{L} f_l$ where f_l is the false positive rate at stage l. These formulas imply that we do not need to set the focus on the discriminating power (false positive rate) of the classifiers at each stage of the cascade since even a cascade of classifiers having each quite a high false positive rate will still result in the whole cascade having a low false positive rate (due to the global false positive rate being the product of the false positive rates of all the classifiers). The counterpart is that we need each classifier to have a very high detection rate to avoid missing too much objects.

4.4 Semi-supervised classification

Semi-supervised learning covers a class of learning methods that exploit both labeled and unlabeled data during the training (typically a small amount of labeled data and a large amount of unlabeled data). Thus, it falls between unsupervised and supervised learning and as such uses methods from both "domains". The interest of semi-supervised techniques has already been demonstrated in section 3.4.1, so, in this section we will only quickly remind the fundamental assumptions on which semi -supervised methods lean and detail some of the semi-supervised techniques that are the

closest to the algorithms we develop in the following parts of this work.

The two main assumptions of semi-supervised methods are on one hand the ***cluster assumption*** and on the other hand the ***low density separation assumption***. They amount, in essence, to the same underlying idea but they have nevertheless inspired quite different techniques in practice by suggesting two rather different ways of tackling the problem of exploiting the unlabeled data. We can formulate these two assumptions in the following way:

- *Cluster assumption:* If two points are in the same cluster, they are likely to be in the same class.

- *Low density separation:* The decision boundary between two classes should lie in a region of low density.

Before using any semi-supervised method, one should systematically ask himself the question: do the unlabeled data help or not ? From a very simple point of view, many standard semi-supervised techniques make a very strong hypothesis on the distribution of unlabeled data compared to that of labeled data: these two distributions are supposed to be the same in many approaches and by-passing this hypothesis may impact the learning in a negative way, the "wrongly distributed" unlabeled data acting as noise during the learning process. To put it in a more formal way, the knowledge of $p(x)$ that one gains through the use of the unlabeled data should bring some information useful in the inference of $p(y|x)$. If this is not the case, semi-supervised learning will not bring any improvement over supervised learning and might even result in a worse performance than using the labeled data alone. In the following, we illustrate this fundamental idea within the particular case of transductive SVMs such as introduced in [Joachims, 1999]. The underlying idea behind the method has already been described in section 3.4.1, so we will just content ourselves with introducing the mathematical formalism. We wish to optimize over the 2^{N_u} possible labelings of the N_u unlabeled data points, the purpose of the optimization problem being to find the labeling which yields the SVM classifier with the largest margin (low density separation assumption). The associated mathematical formalism is the following:

$$\min_{y_1^*,...,y_{N_u}^*} \quad \min_{w,b,\xi,\xi^*} \quad \frac{1}{2}\|w\|^2 + C_l \sum_{i=1}^{N_l} \xi_i + C_u \sum_{i=1}^{N_u} \xi_i^*$$

$$\text{s.t.} \begin{cases} \text{sign}(y_i \cdot (\langle w, \phi(x_i)\rangle + b)) \geq 1 - \xi_i, \forall i = 1,...,N_l \\ \text{sign}(y_i^* \cdot (\langle w, \phi(x_i^*)\rangle + b)) \geq 1 - \xi_i^*, \forall i = 1,...,N_u \\ \xi_i \geq 0, \forall i = 1,...,N_l \\ \xi_i^* \geq 0, \forall i = 1,...,N_u \end{cases} \tag{4.54}$$

The variables related to the unlabeled data are denoted with a "*". In the two graphics of the figure 4.6, we represent both the case where the unlabeled data help obtain a better classifier and the case where they impact the performance of the learning in a negative way. The brute force and most simple approach to solving the problem 4.54 consists in using exhaustive search over all the possible labelings. This isn't of course

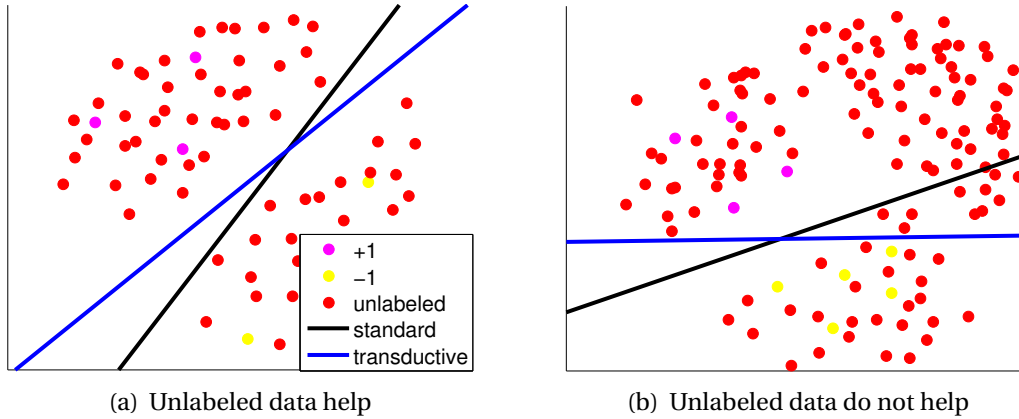(a) Unlabeled data help   (b) Unlabeled data do not help

Figure 4.6: Example of using a transductive SVM in the case where the distribution of unlabeled data fits that of labeled data 4.6(a) and in the case where the distribution of unlabeled data possesses an extra mode compared to that of labeled data 4.6(b). In the second, case, the unlabeled data acts as noise during the training and the performance is worse than using the labeled data alone.

a practicable solution in the case where N_u is large but one has to keep in mind that all the proposed practical solutions [Joachims, 1999; Xu and Schuurmans, 2005] in the literature are sub-optimal and subject to convergence towards local optima of the objective function.

4.5 Summary

In this chapter, we described some classification techniques and learning frameworks which somewhat inspired our contributions to the field. In brief, we provided an insight into the following methods:

- unsupervised classification methods: modeling the data as a mixture of Gaussians and estimating the paramaters of that mixture with an EM algorithm is a very common way to perform unsupervised classification. We inspire ourselves from this approach in the chapter 5 to estimate the parameters of a three-level hierarchical model which rely on a mixture-of-Gaussian assumption regarding the intrinsic distribution of the data. We also use the mixture-of-Gaussian modeling approach in the chapter 6, but, in this case, it is rather a weak assumption regarding the "form" of the data than a real modeling used in the classification process. We make use of unsupervised clustering algorithms like K-means or ELBG in both cases to "seed" the EM algorithm. The ELBG algorithm is also used in the chapter 7 to "summarize" the negative feedback examples we extract in the automatic label propagation part of our cascaded active learning scheme.

- supervised classification methods: we mainly described the SVM formalism and we introduced soft variants of this algorithm which we use in the chapter 6 and more especially in the section of this chapter dealing with the unlearning of er-

roneously introduced mixture components. We also use a soft-SVM as the base learner in the cascaded scheme described in chapter 7.

- semi-supervised classification methods: the semi-supervised SVM described in the section 4.4 is the closest in essence from the composent-based SVM we introduce in the chapter 6. In both cases, we constrain the SVM surface to pass through low-density areas but the two methods rely on completely different approaches to implement this idea.

- learning frameworks/strategies: we exploit the SVM_{active} strategy described in 4.3 to decide which samples to feed back to the user after each active learning iteration in both the chapter 6 and the chapter 7. As for the MIL framework and the coarse-to-fine strategy described respectively in 4.3 and 4.3, they are implemented in our object retrieval scheme (see chapter 7).

Part II

Fast learning methods and concepts for earth observation image information mining: theory and results

Chapter 5

Semi-supervised annotation and unknown semantic structures discovery in satellite image repositories

In this chapter, we envisage the problem of mining satellite image repositories from the point of view of automatic annotation systems such as represented in figure5.1. These systems are designed to achieve two goals which are: (1) the automatic annotation of previously unseen images and (2) the retrieval of database images using keyword-based queries. Through these two goals, annotation and semantic retrieval are envisaged in a complementary way: once the database has been annotated with predefined keywords, it is indeed quite straightforward to formulate semantic queries using the same keywords.

The system we introduce belongs to the category of latent variable models (we provide a general description of this category of models in section 3.1). A recurring problem with most state-of-the-art auto-annotation methods is that they rely on a fully-supervised training step to build the annotation model, and, consequently, they need a sufficiently large number of training samples to accurately learn the mapping model between image descriptors and semantic concepts. They also make the assumption that all the target classes possess representative training samples in the training database, which is far from being a realistic scenario. In satellite image scenes, we can indeed observe a quasi-infinite variety of structures because of, among others, spatial and temporal variations in soil. Therefore, it is often not a reasonable hypothesis to consider that there exists a training database summarizing all these structures. Instead, we might want a system able to learn the image classes present in the training database and at the same time able to detect and retrieve unknown image classes by analyzing the unlabeled data, the final goal being to feed back relevant examples of these unknown classes to the user so that he/she gives them an annotation. Among semi-supervised techniques dedicated to classifying/annotating remote sensing data, we can mention the approach of Bruzzone et al. [Bruzzone et al., 2006] who have proposed an adaptation of the S3VM described in Joachims [Joachims, 1999] to the multi-class problem. Though the proposed method allows both the exploitation of labeled and unlabeled samples, an extra hypothesis is made regarding the distribution of unlabeled samples which is supposed to be the same as that of the labeled ones. In the con-
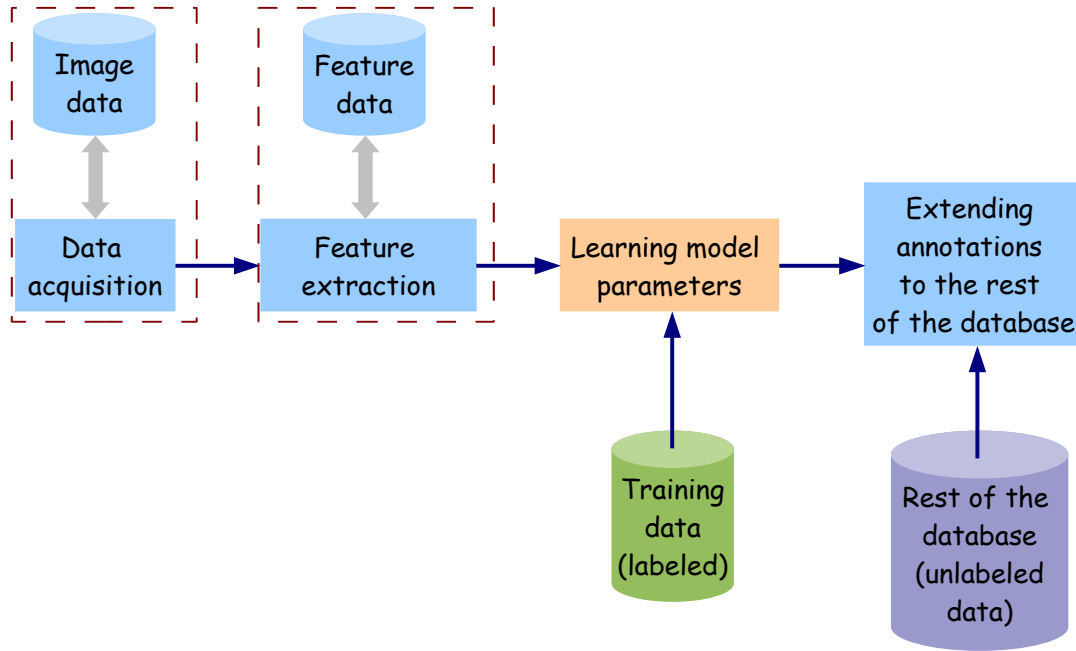
Figure 5.1: Generic architecture of a system to perform analysis and auto-annotation of remote sensing data. We start by extracting descriptors from images. A training database consisting of annotated samples is then used to compute the model parameters. The computed model is re-used in the last step to extend the annotations to the unannotated samples.

trary case (for instance if the distribution of unlabeled samples is modeled as a GMM and possesses modes which do not figure in the distribution of labeled samples), unlabeled samples will just act as noise in the model and the result can be a classifier which performs worse than the classifier trained with only the labeled samples.

5.1 Latent variable models for semi-supervised knowledge discovery

In the following, we introduce a semi-supervised approach based on latent variable models to perform auto-annotation of image databases and discover unknown image classes. We describe a semi-supervised three-level hierarchical model which allows us to exploit the huge amount of unlabeled data. The advantages of such a model are: (1) because of the presence of unlabeled data, the training algorithm used to learn the model parameters is not affected by the small sample size problem [Shahshahani and Landgrebe, 1994], thus leading to more reliable estimates of the model parameters; (2) such a training algorithm can be used to infer unknown image classes among the data.

The flowchart of the whole system is given in Figure 5.2. We start by extracting the model M corresponding to the labeled part of the training database. We use only labeled samples to do so, which is sufficient to obtain a first approximation of this model since by definition, the labeled data contains all known image classes (i.e. the classes

"urban area", "desert", "forest", "fields" in the example of Figure 5.2). The model M is then re-used to compute the general model of the data. Since the unlabeled part of the training database contains samples from both known and unknown image classes, we use it to learn the model $\bar{M}$ corresponding to the unknown image classes. We exploit the knowledge of the already learned model M to infer which of the unlabeled data samples are likely to belong to an unknown image class. These unlabeled data are then used to learn the model $\bar{M}$ and the remaining unlabeled data are used to refine the model M. Thus, the distribution of unlabeled data needs not necessarily fit the distribution of the labeled data. There can be extra modes in the unlabeled data distribution which will be interpreted as corresponding to unknown image classes and thus will not weaken the performance of the classifier on known image classes.
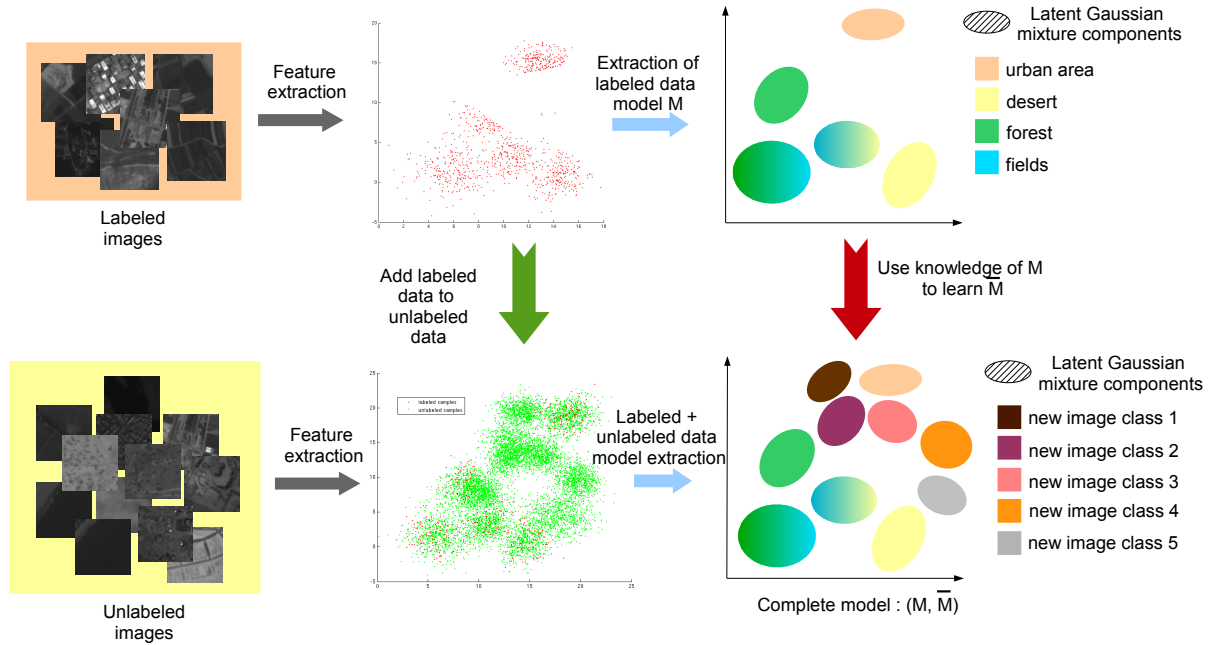
Figure 5.2: Flowchart of the whole system: the first step consists in extracting the descriptors from the two parts (labeled and unlabeled) of the training database. We then extract the model M of the labeled training data. The probabilistic associations between semantic concepts and clusters are represented through the use of a gradient of color. For ease of representation, we have represented well-separated ellipsoidal clusters and we have chosen to assimilate the latent Gaussian mixture components with the Gaussian mixture components which would have been obtained in a purely data-driven mode. In reality, Gaussian mixture components in our model are "discovered" by analyzing the associations between semantic concepts and descriptors in the training database, so they may not match exactly those computed directly from data. The last step is the semi-supervised part of the model and consists in incorporating the unlabeled samples so as to make the unknown image classes appear. The model M is re-used to identify these new classes and its parameters are also re-estimated. We obtain in the output an estimation of both the known and the unknown part of the model $(M, \bar{M})$.

Our model is similar in nature to LDA-based models. We start from low-level visual words (image descriptors in our case) that we relate to high-level semantic concepts through the use of intermediary latent variables which are, in our case, latent Gaussian mixture components. As in latent variable models, the latent Gaussian mixture components are not directly inferred from the data as would be done for instance in a classical GMM-based approach [Carson et al., 2002; Fan et al., 2008], but from the associations between semantic concepts and image descriptors in the training database. Each semantic concept can be viewed as a "mixture" of a small number of latent mixture components; more precisely we can define each semantic concept through the use of a probabilistic vector describing the probability of the concept to appear given each latent mixture component. The creation of each feature vector (which correspond to visual words in our case) is also conditioned on the latent mixture components. The main conceptual difference between our model and LDA-based models lies in the fact that our model does not define a true top-down generative process, that is, starting from a category of document, LDA defines a complete generative process of documents falling into this category. Our approach is rather a bottom-up approach, that is, starting from a feature vector, we define an inference process which allows to compute the most probable semantic concepts given that feature vector. In this sense, our approach is more similar to that of Fan et al. [Fan et al., 2008].

The use of GMMs as a basic modeling of data is questionable since it often leads to difficult estimation problems especially when using high-dimensional feature sets. However several approaches have been tested successfully in the literature [Fan et al., 2008; Shahshahani and Landgrebe, 1994; Sheikholeslami et al., 2002; Barnard et al., 2003; Carson et al., 2002]. In all these approaches, both the dimensionality of the feature space and the number of mixture components must be controlled but the obtained results remain still very satisfying even for complex annotation problems. The semantic classes are indeed formed by grouping together several mixture components. Thus, even with a few mixture components, there is a huge number of possible combinations, making GMM-based systems suitable for complex annotations problems. In table 5.1, we have represented the dimensionality of feature spaces along with the number of mixture components and the sizes of training sets in several systems using GMMs. The last row represents the number of semantic classes which are learned using these mixtures. We observe that the number of semantic classes is often comparable with the number of mixture components in the model. For complex semantic classes, this means that the model is using combinations of mixture components to define these classes.

The chapter is organized as follows: we introduce first two versions of a fully-supervised algorithm using only the annotated samples in the training database. This algorithm is used to compute the part M of the model (cf. Fig. 5.2). Next, we present a semi-supervised procedure which allows us to incorporate unannotated samples and to infer the existence of unknown image classes. In the last sections, we present experimental results on a synthetic dataset, making a comparison of our algorithm with a semi-supervised SVM as introduced in [Bruzzone et al., 2006]. We also demonstrate the discovery of unknown image classes on a database of SPOT5 satellite images.

system	[Fan et al., 2008]	[Shahshahani and Landgrebe, 1994]	[Sheikholeslami et al., 2002]	[Barnard et al., 2003]	[Carson et al., 2002]
D	35	10	20	30	24
L	25	4	22	500	15
S	4500	1000	2940	8000	5000
C	27	8	15	155	42

Table 5.1: Characteristics of different GMM-based systems. D refers to the dimension of the feature space, L to the number of mixture components in the GMM-based model, S to the size of the training set used to train the model and C to the number of semantic classes which are learned.

5.2 Computing the auto-annotation model via a fully supervised approach

In this section, we introduce two fully-supervised algorithms using only the labeled part of the training database to compute the model parameters. More precisely, we define two procedures that allow us to predict "atomic" image concepts with high posterior probability given an image. In our approach, "atomic" image concepts refer to the keywords used to annotate images. We call them "atomic" since we usually choose to annotate images low-level concepts with small intra-concept variations on visual properties. For example, on SPOT5 images, we will use concepts such as "urban area", "fields", "clouds", "ports", "forest", "sea" (cf. section 5.4.2)... Some examples of images, which can be annotated with these concepts, are given in Figure 5.4. We will avoid using more generic concepts, such as "agriculture" or "nature", which are in general much more user-specific, that is, which may correspond to different visual realities depending on the user. We could of course imagine a system which also allows the use of higher level concepts, but this generally implies creating another level in the hierarchy to map the high-level and often subjective semantic image concepts into more "atomic" ones (cf. Fig. 5.3). Besides, hierarchical models with many levels are generally prone to inter-level error transmission, that is, an error which is committed in the learning at one level of the hierarchy is propagated to the other levels since the learning of the model parameters is done all at once for all the levels and not in several independent steps. Instead, we might think of keeping things simple and consider that the user is directly using "atomic" concepts to annotate the images. In the case of SPOT5 images, the "atomic" concepts are predefined in function of the different structures we can observe in the images (some examples of "atomic concepts" have been given above). The "atomic" concepts are sufficient to describe semantically these structures. The idea behind using such "atomic" concepts is simply to remove all subjectivity in the use of semantic concepts (subjectivity often means multiple and/or conflicting annotations of images in the training database, so it is likely to impact the learning process in a bad way).

The use of "atomic" concepts instead of more complex annotations will not impact the generalizability of the approach since images are generally decomposable in terms of
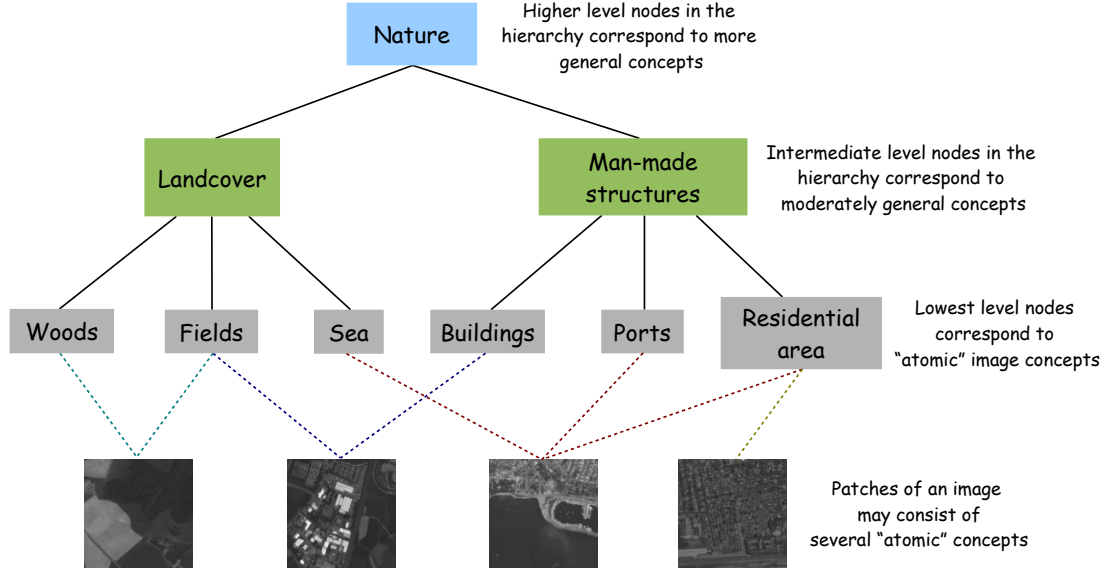
Figure 5.3: Hierarchy of concepts: the "atomic" image concepts are the concepts at the lowest level of the hierarchy. Images may consist of several "atomic" concepts due to their compositional nature. As we progress upwards the hierarchy, concepts are becoming more and more general. The highest level concepts are besides often subjective, that is, user-dependent.

low-level concepts. This is what is referred to in the literature as the compositional nature of images [Barnard et al., 2003]. An example is given in figure 5.4 where we have represented in the last two rows more complex images which cannot be explained with only one "atomic" concept. We need instead two or three such concepts to summarize the content of the image but the complex structure it contains can still be explained using a combination of "atomic" concepts. For instance, in figure 5.4, the seaside areas will be decomposed into "sea", "ports", "residential area" with corresponding high probabilities of these concepts in the unigram models describing the image.

The idea of the following two algorithms is to "explain" each atomic concept with the help of one or several latent data structures (cf. Fig. 5.5). The two statistical models which link "atomic" image concepts to low-level visual features are both computed on a small training database using an Expectation-Maximization (EM) algorithm [Dempster et al., 1977]. As mentioned in the introduction, the models presented here have some similarities with latent variable models [Loehlin, 1987]. We assume that our data, that is, feature vectors v extracted from images (color, texture, and shape descriptors for instance), possess an underlying structure which can be modeled by a Gaussian mixture density (in the formula below, we use a star to avoid confusion between the parameters of the Gaussian mixture model used to represent the a priori distribution of data and the parameters of the latent Gaussian mixture components in our model):

$$p(v|\{\pi_l^*, \mu_l^*, \Sigma_l^*\}_{l=1,\dots,L}) = \sum_{l=1}^{L} \pi_l^* \cdot \mathcal{N}(v; \mu_l^*, \Sigma_l^*) \qquad (5.1)$$
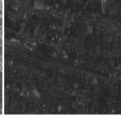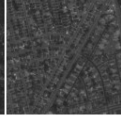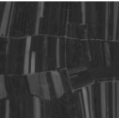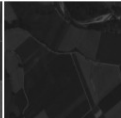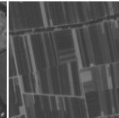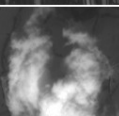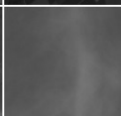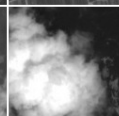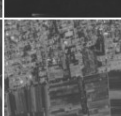
Patches of an image	Number of mixture components
	4
	3
	2
	6
	8

Figure 5.4: Examples of images which can be annotated with the five "atomic" concepts "urban area", "fields", "clouds", "ports" and "sea". Images in the first three rows can be annotated using only one "atomic" concept whereas images in the last two rows are composed of several "atomic" concepts. The figures in the last column represent the number of latent mixture components which are used to explain the concepts present in the images.

where the prior probabilities of mixture components, π_l^*, are such that $\sum_{l=1}^{L} \pi_l^* = 1$. But rather than learning directly each mixture component as would be done in a classical Expectation Maximization approach, we infer those components from the associations between "atomic" image concepts and feature vectors. In this sense, the underlying Gaussian mixture components can be considered as latent variables since they correspond to a latent (not directly observable) behavior in the data. The following notations will be used: we denote $a_1, a_2, ..., a_I$ "atomic" image concepts used to annotate images. Each "atomic" concept is "explained" by probabilistic associations with latent mixture components (cf. Fig. 5.5). We denote $v_1, v_2, ..., v_N$ feature vectors extracted from each annotated image of the training database (for further details, see section 7.7). We call an annotated image a "document". We can extract one or several feature vectors from each image (on condition that they all belong to the same feature space) and we can associate one or several annotations ("atomic" concepts) to an image. A document d_k will be thus represented by a couple of sets $\{A^k, V^k\}$ with $A^k = \{a_1^k, ..., a_{|A^k|}^k\}$ and $V^k = \{v_1^k, ..., v_{|V^k|}^k\}$ where $|\Omega|$ refers to the cardinality of set Ω. In case we have explicit associations between "atomic" concepts and feature vectors inside d_k, we can also write: $d_k = \{(a_i^k, v_j^k), ...\}$ where each couple (a_i^k, v_j^k) points out a specific association between the i-th "atomic" concept and the j-th feature vector of d_k. In the following we present successively two algorithms for handling the two
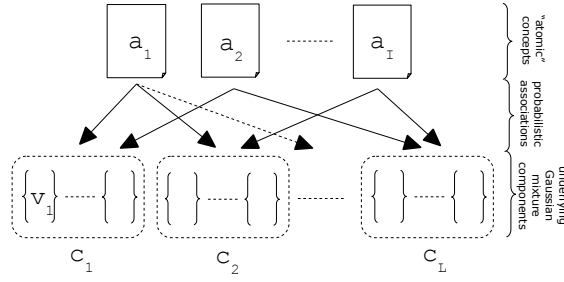
Figure 5.5: Latent variable model. Each manifest variable, i.e. each "atomic" concept a_i, is explained by one or several latent Gaussian mixture components c_l.

Figure 5.6: Graphical model representing the generative process associated with documents in the case of no explicit associations between image concepts and feature vectors. Rounded boxes represent replicates. Shaded circles represent hidden random variables (not directly observed) and ordinary circles represent observed random variables. Rectangles contain model parameters.

situations, i.e., with and without explicit associations. In both cases we want to learn the latent Gaussian mixture component parameters (mean μ_l, covariance matrix Σ_l, and prior probability of mixture component π_l) as well as the associations between "atomic" concepts and latent mixture components, that is, the probabilities of each "atomic" concept given the latent mixture components, $p(a_i|c_l)$.

5.2.1   Case 1: learning with no explicit associations between "atomic" concepts and feature vectors

In this first case, we represent training documents as unordered collections of feature vectors and "atomic" concepts. This is what is called the "bag of words" assumption in latent variable models. Under this assumption, feature vectors within a document are considered conditionally independent given a latent mixture component. The same assumption holds for "atomic" concepts. More formally speaking, the "bag of words"

assumption implies that documents inside a latent class are independent realizations of a random variable whose distribution is that of the underlying generative model associated with the latent mixture component. This allows us to interpret similarity within a latent class as a statistical notion which implies that the underlying generative models of documents in that class are identical. Under the above-mentioned two-fold "bag of words" assumption, the conditional probabilities of feature vectors and "atomic" concepts can be written respectively as: $p(V^k|c_l) = p(v_1^k, ..., v_{|V^k|}^k|c_l) = \prod_{j=1}^{|V^k|} p(v_j^k|c_l)$ where $p(v_j^k|c_l)$ is Gaussian with mean μ_l and covariance matrix Σ_l and $p(A^k|c_l) = p(a_1^k, ..., a_{|A^k|}^k|c_l) = \prod_{i=1}^{|A^k|} p(a_i^k|c_l)$. Assuming that feature vectors and "atomic" concepts inside a document are conditionally independent given a mixture component c_l, the probability of a document given a mixture component is expressed by:

$$p(d_k|c_l) = \left[\prod_{j=1}^{|V^k|} p(v_j^k|c_l) \right] \cdot \left[\prod_{i=1}^{|A^k|} p(a_i^k|c_l) \right] \tag{5.2}$$

We can then explain the generative process of a document by: $p(d_k|\theta) = \sum_{l=1}^{L} \pi_l \cdot p(d_k|c_l, \theta)$ where θ refers to model parameters (cf. 5.6). The corresponding graphical model is represented in Figure 5.6. To train the model, we use an Expectation Maximization (EM) algorithm which is an iterative procedure to try to find the maximum likelihood estimate (MLE) of the model parameters. Using the EM algorithm involves introducing a hidden variable, H_l^k, characterizing the membership of document d_k into the latent class c_l. In the EM procedure, the problem of finding the MLE ($\theta_{MLE} = \arg\max_\theta \mathcal{L}(D; \theta)$) is reformulated in the following way: $\theta_{MLE} = \arg\max_\theta E_H[\mathcal{L}(D, H; \theta)]$ where $H = \{H_l^k\}_{l,k}$ refers to the hidden variable and D to the training documents (observed variable). The EM procedure consists of two steps. The first (expectation step) consists in computing the expected log-likelihood of complete data with respect to the conditional distribution of H given D under the current estimate of the parameters θ_t:

$$\begin{aligned} Q_D(\theta|\theta_t) &= E_{H|D,\theta_t} \left[\log \mathcal{L}(D, H; \theta) \right] \\ &= E_H \left[\log \mathcal{L}(D, H; \theta) | D, \theta_t \right] \end{aligned} \tag{5.3}$$

The second step (maximization step) consists in maximizing the quantity $Q_D(\theta|\theta_t)$ with respect to θ:

$$\theta_{t+1} = \arg\max_\theta Q_D(\theta|\theta_t) \tag{5.4}$$

In the following, we refer to $Q_D(\theta|\theta_t)$ as the conditional expected log-likelihood of complete (hidden and observed) data. The conditional expected log-likelihood $Q_D(\theta|\theta_t) =$

$E_H\left[\log\mathcal{L}(D,H;\theta)|D,\theta_t\right]$ of complete data is given by:

$$Q_D(\theta|\theta_t) = E_H\left[\log\prod_{k=1}^{K}p(d_k,H_l^k|\theta)|D,\theta_t\right] \tag{5.5}$$

$$= E_H\left[\sum_{k=1}^{K}\log p(d_k,H_l^k|\theta)|\{d_k\}_{k=1,\dots,K},\theta_t\right]$$

$$= \sum_{k=1}^{K}E_{H_l^k}\left[\log p(d_k,H_l^k|\theta)|d_k,\theta_t\right]$$

$$= \sum_{k=1}^{K}\sum_{l=1}^{L}p(H_l^k=1|d_k,\theta_t)\log p(d_k,H_l^k|\theta)$$

$$= \sum_{k=1}^{K}\sum_{l=1}^{L}p(H_l^k=1|d_k,\theta_t)\times$$

$$\log\left[p(d_k|H_l^k=1,\theta)\cdot p(H_l^k=1|\theta)\right]$$

$$= \sum_{k=1}^{K}\sum_{l=1}^{L}p(H_l^k=1|d_k,\theta_t)\cdot\log\left[p(d_k|c_l,\theta)\cdot p(c_l)\right]$$

where θ refers to the model parameters:

$$\theta = \left\{\pi_l,\mu_l,\Sigma_l,\{p(a_i|c_l)\}_{i=1,\dots,I}\right\}_{l=1,\dots,L} \tag{5.6}$$

We have: $p(c_l) = \pi_l$. To compute the EM iterative formulas, we use the Lagrangian of the conditional expected log-likelihood with normalization constraints. The details of the computations are provided in appendix A. We obtain the following coupled re-estimation equations:

$$\gamma_l^k = p(H_l^k=1|d_k,\theta) = \frac{\pi_l\cdot p(d_k|c_l,\theta)}{\sum_{l=1}^{L}\pi_l\cdot p(d_k|c_l,\theta)},$$

$$\pi_l = \frac{\sum_{k=1}^{K}\gamma_l^k}{K}, \ \mu_l = \frac{\sum_{k=1}^{K}\left[\gamma_l^k\cdot\sum_j v_j^k\right]}{\sum_{k=1}^{K}\gamma_l^k\cdot|V^k|},$$

$$\Sigma_l = \frac{\sum_{k=1}^{K}\left[\gamma_l^k\cdot\sum_j(v_j^k-\mu_l)\cdot(v_j^k-\mu_l)^T\right]}{\sum_{k=1}^{K}\gamma_l^k\cdot|V^k|},$$

$$p(a_i|c_l) = \frac{\sum_{k=1}^{K}\gamma_l^k\cdot N_{a_i}^k}{\sum_{q=1}^{I}\sum_{k=1}^{K}\gamma_l^k\cdot N_{a_q}^k}$$

where $N_{a_i}^k$ is the number of times a_i is in d_k (a concept can be used several times to annotate the same image).

5.2.2 Case 2: learning with explicit associations between "atomic" concepts and feature vectors

In some cases, we possess explicit associations between "atomic" concepts and feature vectors in the training database. These associations are taken into account in the
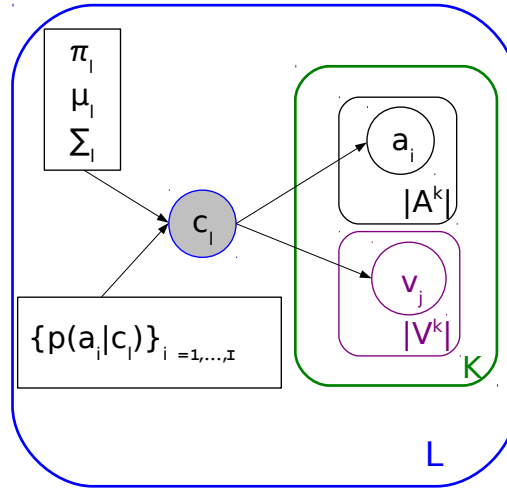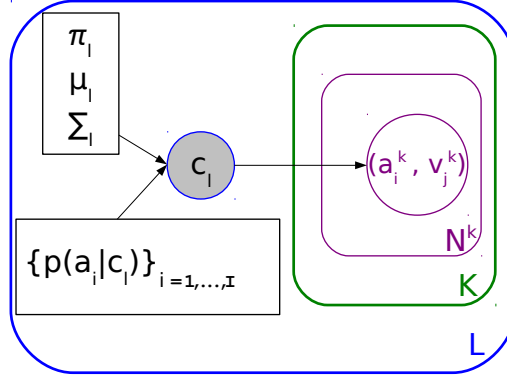
Figure 5.7: Graphical model representing the generative process associated with documents in the case of explicit associations between image concepts and feature vectors. We use the same graphical conventions as in figure 5.6

learning procedure by modifying the hidden variable in the EM algorithm, i.e., we take a variable H_{ijl}^k which points out explicitly the association between the i-th "atomic" concept and the j-th feature vector in the document d_k. Thus, H_{ijl}^k characterizes the membership of the couple (a_i^k, v_j^k) to the latent class c_l. This allows us to relax the "bag of words" assumption; we are not considering a generative model of documents anymore but a generative model of couples {"atomic" concept, feature vector}. The modified graphical model associated with this generative process is represented in Figure 5.7. We keep however the document-based representation because it is the way the database has been designed. The only assumption we are making is that of conditional independence between "atomic" concepts and feature vectors inside a couple. That is, given a latent class c_l, an "atomic" concept a_i^k is considered independent of the feature vector v_j^k it is associated with: $p(a_i^k, v_j^k|c_l) = p(a_i^k|c_l) \cdot p(v_j^k|c_l)$. Following a similar approach as in 5.2.1, we obtain the following expression for the conditional expected log-likelihood of complete data:

$$
\begin{aligned}
Q_D(\theta|\theta_t) &= \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} E_{H_{ijl}^k} \left[\log p(a_i^k, v_j^k, H_{ijl}^k|\theta)|(a_i^k, v_j^k), \theta_t \right] \\
&= \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \sum_{l=1}^{L} \gamma_{ijl}^k(t) \cdot \log \left[p(a_i^k, v_j^k|c_l, \theta) \cdot \pi_k \right]
\end{aligned}
\tag{5.7}
$$

where $\gamma_{ijl}^k(t) = p(H_{ijl}^k = 1|a_i^k, v_j^k, \theta_t)$.

This leads to the following EM iterative formulas:

$$\gamma_{ijl}^k = p(H_{ijl}^k = 1 | a_i^k, v_j^k, \theta) = \frac{\pi_l \cdot p(a_i^k, v_j^k | c_l, \theta)}{\sum_{l=1}^L p(a_i^k, v_j^k | c_l, \theta)}$$

$$\pi_l = \frac{\sum_{k=1}^K \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k}{N_C}, \mu_l = \frac{\sum_{k=1}^K \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k \cdot v_j^k}{\sum_{k=1}^K \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k},$$

$$\Sigma_l = \frac{\sum_{k=1}^K \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k \cdot (v_j^k - \mu_l) \cdot (v_j^k - \mu_l)^T}{\sum_{k=1}^K \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k},$$

$$p(a_{i_0} | c_l) = \frac{\sum_{k=1}^K \sum_{(a_i^k, v_j^k)} \delta_{a_i^k}^{a_{i_0}} \gamma_{ijl}^k}{\sum_{q=1}^I \sum_{k=1}^K \sum_{(a_i^k, v_j^k)} \delta_{a_i^k}^{a_q} \gamma_{ijl}^k}$$

where $\delta_{a_i^k}^{a_q}$ is such that: $\delta_{a_i^k}^{a_q} = 1$ if $a_i^k = a_q$ and 0 otherwise.

Remarks and notations: "$\sum_{(a_i^k, v_j^k)}$" means a summation over all couples contained in d_k (the same "atomic" concept may appear in several couples). However, for probabilities $p(a_{i_0} | c_l)$, we are considering only distinct atomic concepts. The summation "$\sum_{q=1}^I$" is thus over distinct concepts. N_C refers to the total number of couples in the training database.

5.2.3   Unigram models

Starting from the above statistical model, we can compute unigram models for new unannotated images by using Bayesian inference. Given an unannotated image Img, which is modeled by a set of feature vectors $\{v_j\}$, we obtain:

$$p(a_i | Img) = \sum_{l=1}^L p(a_i, c_l | v_j \in Img) =$$

$$\sum_{l=1}^L p(a_i | c_l, v_j \in Img) \cdot p(c_l) \cdot p(v_j \in Img | c_l) \qquad (5.8)$$

According to the "bag of words" assumption, the feature vectors are conditionally independent given a mixture component c_l, thus: $p(v_j \in Img | c_l) = \prod_{v_j \in Img} p(v_j | c_l)$. Using the hypothesis of conditional independence between feature vectors and "atomic" concepts given a mixture component c_l, we obtain: $p(a_i, v_j \in Img | c_l) = p(a_i | c_l) \cdot p(v_j \in Img | c_l)$. We have also, using Bayes law: $p(a_i, v_j \in Img | c_l) = p(a_i | v_j \in Img, c_l) \cdot p(v_j \in Img | c_l)$. By comparing the two last expressions, we obtain: $p(a_i | v_j \in Img, c_l) = p(a_i | c_l)$. Thus,

$$p(a_i | Img) = \sum_{l=1}^L \left[\pi_l \cdot p(a_i | c_l) \cdot \prod_{v_j \in Img} p(v_j | c_l) \right] \qquad (5.9)$$

$\{p(a_i | Img)\}_{i=1,\cdots,I}$ is the unigram model associated with the image Img. The interest of such an approach is that we can set an arbitrary "resolution" for the annotation

process. For a given image, we can choose to annotate this image by conditioning the unigram model computation on the entire image. We can also decide to divide it into smaller patches and annotate each patch separately by making one unigram model computation per patch (this is what is done in the example of the next section). This is particularly useful when we have to model very large images such as satellite images, for which a global feature vector will not be able to individualize the model of particular regions or structures contained in the image. Subdividing the image has the advantage of extracting the significant feature vectors of smaller image patches which will be then modeled by a non-singleton set $\{v_j\}$. Thus the unigram model in 5.9 will refer to the whole image, but will still be relevant with the localized image content.

5.3 Computing the auto-annotation model via a semi-supervised approach

The approaches presented above, however, have several drawbacks. For instance, they do not allow us to exploit unannotated samples during the training step. This can be critical in several cases where the ratio of the number of training samples to the number of feature measurements (unannotated samples) is small. It is well-known that, in such cases, we are confronted with the "small sample size problem" [Fukunaga, 1990]; in particular, the estimates of the discriminant functions (in our case likelihood functions) are very inaccurate since they rely on the maximum-likelihood estimates of the covariance matrices, which are often singular for small sample sizes. A solution to this problem might be to use a plug-in estimate for covariance matrices like the sample group covariance estimate [Thomaz et al.]. A more satisfactory solution can be found in [Shahshahani and Landgrebe, 1994] where it is shown that by using additional unannotated samples, more representative estimates can be obtained. In particular, it is shown that the variance of estimators obtained by integrating unlabeled samples is lower. This is not surprising since this quantity is lower-bounded by the Cramer-Rao bound, which, in the Gaussian case, for the mean and covariance component estimates, is inversely proportional to the number of samples used in the estimation. The authors of [Shahshahani and Landgrebe, 1994] propose a semi-supervised algorithm relying on Expectation Maximization to perform maximum-likelihood estimation. However, this approach does not take into account the fact that unknown image classes might exist among unannotated data as it is the case, for instance, in Earth Observation imagery. The solution we propose is based on integrating an unknown class inference process within the EM scheme. The model we develop in the following is based on the approach presented in 5.2.1. The same line of reasoning would apply to 5.2.2.

5.3.1 Integrating unlabeled samples into the learning process

We divide the set of samples D_{tot} into two disjoint subsets $\{D_a, D_u\}$, which refer respectively to annotated and unannotated samples. Thus, D_a corresponds to the set of annotated documents, which is referred to as D above. We wish to maximize the joint likelihood of annotated and unannotated data. The conditional expected joint

log-likelihood of the complete data is given by:

$$Q_{D_{tot}}(\theta,\phi|\theta_t,\phi_t) = Q_{D_a \cup D_u}(\theta|\theta_t) =$$
$$\sum_{d_k \in D_a} E_{H_l^k}\left[\log p_a(d_k, H_l^k|\theta)|d_k,\theta_t\right]$$
$$+ \sum_{d_k \in D_u} E_{H_l^k}\left[\log p_u(d_k, H_l^k|\phi)|d_k,\phi_t\right] \tag{5.10}$$

where $\phi = \{\pi_l, \mu_l, \Sigma_l\}_{l=1,...,L}$. In this equation, the first term represents the conditional expected likelihood of the complete (hidden and observed) annotated data and the second the conditional expected likelihood of the complete unannotated data. We use two different likelihood functions, p_a and p_u, since in the first term we want to compute the likelihood with respect to the latent variable model parameters θ and in the second term the likelihood with respect to the latent Gaussian mixture parameters ϕ. We denote by d_k both the annotated and the unannotated documents but in the latter case, d_k is just a singleton containing one feature vector (unannotated documents containing more than one feature vector are split into several singletons: this avoids considering unannotated documents containing both known and unknown structures, which might be problematic).

5.3.2   Inferring the existence of unknown semantic structures

"Known" structures are defined as the latent Gaussian mixture components which correspond to "known" image classes. "Unknown" structures are defined as the latent Gaussian mixture components which correspond to "unknown" image classes, that is, image classes not represented in the training database. The previous approach only considers "known" mixture components among the data: the expectation is done over the same mixture components both for annotated and unannotated samples. Unknown structures however correspond to the underlying mixture components that cannot be inferred by using annotated samples only. Starting from this observation, we introduce additional mixture components to account for unknown structures. The set of latent mixture components thus becomes: $C = \{C_d, C_{nd}\}$ where C_d refers to already-defined (known) mixture components and C_{nd} to unknown mixture components. The corresponding conditional expected joint log-likelihood of the complete (hidden and observed) data is given by:

$$Q_{D_{tot}}(\theta,\phi|\theta_t,\phi_t) = \sum_{d_k \in D_a} E_{H_l^k}^{l \in C_d}\left[\log p_a(d_k, H_l^k|\theta)|d_k,\theta_t\right]$$
$$+ \sum_{d_k \in D_u^{known}} E_{H_l^k}^{l \in C_d}\left[\log p_u(d_k, H_l^k|\phi^d)|d_k,\phi_t^d\right]$$
$$+ \sum_{d_k \in D_u^{unknown}} E_{H_l^k}^{l \in C_{nd}}\left[\log p_u(d_k, H_l^k|\phi^{nd})|d_k,\phi_t^{nd}\right] \tag{5.11}$$

where $\phi_d = \{\pi_l, \mu_l, \Sigma_l\}_{l \in C_d}$ and $\phi_{nd} = \{\pi_l, \mu_l, \Sigma_l\}_{l \in C_{nd}}$. The first term of (5.11) deals with training database documents so the expectation is performed only over the known mixture components C_d. The last two terms are explained by the fact that, in unannotated documents, we have both known and unknown mixture components, so we

need to compute the expectation over both subsets C_d and C_{nd}. Like for mixture components for which we consider two disjoint sets, we force a "hard" separation between documents "explained" by known mixture components and those "explained" by unknown mixture components (in the following, we call these documents "unknown elements" and we write $D_u = \{D_u^{known}, D_u^{unknown}\}$). This works much better in practice and can be explained from a heuristic point of view: at the beginning of the EM procedure, we don't know anything about unknown mixture components (we perform a random initialization of these components). So, instead of learning unknown mixture components, the initial tendency of the algorithm might be as well to "unlearn" known components. This might happen if the likelihood of "unknown elements" among unannotated data is higher with respect to already-defined components than with respect to randomly-initialized unknown components. This is quite a normal behavior considering the fact that the EM algorithm converges towards a local maximum which can be reached directly from its initialization point (that is, without jumping over states scoring a higher likelihood). To avoid that, we might want to enforce already-defined components to remain almost unchanged during the first iterations of the procedure while waiting for "unknown elements" among unannotated data to become more likely given the unknown mixture components. This is an heuristic way to force the EM scheme to converge towards a more optimal solution (the ideal being a procedure which preserves known components but still refines them using unannotated samples and at the same time allows to learn unknown components from unannotated samples). The solution we have retained to split the set of unannotated documents is to put a dynamic threshold on the likelihood of a given sample given the part of the model constituted by already-defined mixture components (ϕ^d). We start with quite a high threshold to be sure that samples in D_u^{known} are not in fact "unknown elements". As the unknown part of the model (ϕ^{nd}) becomes better defined (that is as "unknown elements" become more likely given that part of the model), we decrease the threshold to "equalize" the chances of a sample to be explained either by known mixture components or by unknown mixture components (by doing so, at each iteration, we get closer to the standard EM procedure). When β reaches the value 0.5, we can replace the arbitrary thresholding by a comparison of the respective likelihoods of both parts of the model, ϕ^d and ϕ^{nd}. That is, we decide that an unannotated sample d_k belongs to D_u^{known} when $\mathcal{L}_{\phi^d}(d_k) > \mathcal{L}_{\phi^{nd}}(d_k)$ where:

$$\mathcal{L}_{\phi^d}(d_k) = \sum_{l \in C_d} \pi_l \cdot p(d_k | c_l, \phi^d) \text{ and}$$

$$\mathcal{L}_{\phi^{nd}}(d_k) = \sum_{l \in C_{nd}} \pi_l \cdot p(d_k | c_l, \phi^{nd}) \tag{5.12}$$

$\mathcal{L}_{\phi^d}(d_k)$ represents the likelihood of the unannotated sample d_k regarding the known part of the model ϕ^d and $\mathcal{L}_{\phi^{nd}}(d_k)$ the likelihood of the unannotated sample d_k regarding the unknown part of the model ϕ^{nd}.

The quantity we want to maximize at each iteration t is:

$$Q(\theta, \phi; \theta_t, \phi_t) =$$

$$\sum_{d_k \in D_a} \sum_{c_l \in C_d} \gamma_l^k(\theta_t) \cdot \log \left[p(V^k|c_l, \theta) \cdot p(A^k|c_l, \theta) \cdot \pi_l \right]$$

$$+ \sum_{d_k \in D_u^{known}} \sum_{c_l \in C_d} \gamma_l^k(\phi_t^d) \cdot \log \left[p(V^k|c_l, \phi^d) \cdot \pi_l \right]$$

$$+ \sum_{d_k \in D_u^{unknown}} \sum_{c_l \in C_{nd}} \gamma_l^k(\phi_t^{nd}) \cdot \log \left[p(V^k|c_l, \phi^{nd}) \cdot \pi_l \right] \qquad (5.13)$$

where $\gamma_l^k(\theta_t) = p(H_l^k = 1|d_k, \theta_t)$, $\gamma_l^k(\phi_t^d) = p(H_l^k = 1|d_k, \phi_t^d)$ and $\gamma_l^k(\phi_t^{nd}) = p(H_l^k = 1|d_k, \phi_t^{nd})$.

The procedure described above is summarized in the algorithm 2.

Algorithm 2 Semi-supervised training algorithm

Step 1: training step using only annotated documents $\rightarrow \theta_0$

Step 2: EM algorithm using the whole set $D = \{D_a, D_u\}$

  Set $t = 0$, $\beta = \beta_0$, $D_u^{known} = \{\emptyset\}$, $D_u^{unknown} = D_u$ and $\theta = \theta_0$

  Iterate the following four steps until convergence:

    E-step: Compute $Q(\theta, \phi; \theta_t, \phi_t)$

    Decision-step:

    If $\beta > 0.5$ **then**

      Set $D_u^{known} = \left\{ d_k \in D_u \middle| \mathcal{L}_{\phi^d}(d_k) > \beta \right\}$

      Decrease threshold β, $t \rightarrow t+1$

    Else

      Set $D_u^{known} = \left\{ d_k \in D_u \middle| \mathcal{L}_{\phi^d}(d_k) > \mathcal{L}_{\phi^{nd}}(d_k) \right\}$

    End If

    Set $D_u^{unknown} = D_u \backslash D_u^{known}$

    M-step: Set $\{\theta_{t+1}, \phi_{t+1}\} = \arg\max_{(\theta, \phi)} Q(\theta, \phi; \theta_t, \phi_t)$

  End For

The EM iterative formulas we obtain are very similar to the ones obtained in section 5.2.1 and are displayed below. For the hidden variable, we have:

$$p(H_l^k = 1|d_k \in D_a, \theta) = \begin{cases} \frac{\pi_l \cdot p(d_k|c_l, \theta)}{\sum_{l \in C_d} \pi_l \cdot p(d_k|c_l, \theta)} & \text{if } c_l \in C_d \\ 0 & \text{if } c_l \in C_{nd} \end{cases}, \qquad (5.14)$$

$$p(H_l^k = 1|d_k \in D_u, \phi) =$$

$$\begin{cases} \frac{\pi_l \cdot p(d_k|c_l, \phi^d)}{\sum_{l \in C_d} \pi_l \cdot p(d_k|c_l, \phi^d) + \sum_{l \in C_{nd}} \pi_l \cdot p(d_k|c_l, \phi^{nd})} & \text{if } c_l \in C_d \\ \frac{\pi_l \cdot p(d_k|c_l, \phi^{nd})}{\sum_{l \in C_d} \pi_l \cdot p(d_k|c_l, \phi^d) + \sum_{l \in C_{nd}} \pi_l \cdot p(d_k|c_l, \phi^{nd})} & \text{if } c_l \in C_{nd} \end{cases} \qquad (5.15)$$

and for the latent mixture components parameters:

$$\pi_l =$$

$$\begin{cases} \dfrac{\sum_{d_k \in D_a} \gamma_l^k(\theta) + \sum_{d_k \in D_u^{known}} \gamma_l^k(\phi^d)}{N_{\text{samples}}} & \text{if } c_l \in C_d \\[3mm] \dfrac{\sum_{d_k \in D_u^{unknown}} \gamma_l^k(\phi^{nd})}{N_{\text{samples}}} & \text{if } c_l \in C_{nd} \end{cases} \tag{5.16}$$

where N_{samples} is the total number of samples (annotated plus unannotated): $N_{\text{samples}} = |D_a| + |D_u|$.

$$\mu_l = \begin{cases} \dfrac{\sum_{d_k \in D_a} \gamma_l^k(\theta) \sum_j v_j^k + \sum_{d_k \in D_u^{known}} \gamma_l^k(\phi^d) \sum_j v_j^k}{\sum_{d_k \in D_a} \gamma_l^k(\theta) \cdot |V^k| + \sum_{d_k \in D_u^{known}} \gamma_l^k(\phi^d) \cdot |V^k|} & \text{if } c_l \in C_d \\[3mm] \dfrac{\sum_{d_k \in D_u^{unknown}} \gamma_l^k(\phi^{nd}) \sum_j v_j^k}{\sum_{d_k \in D_u^{unknown}} \gamma_l^k(\phi^{nd})} & \text{if } c_l \in C_{nd} \end{cases} \tag{5.17}$$

$$\Sigma_l = \begin{cases} \dfrac{\begin{array}{l} \sum_{d_k \in D_a} \gamma_l^k(\theta) \sum_j (v_j^k - \mu_l) \cdot (v_j^k - \mu_l)^T + \cdots \\ \cdots + \sum_{d_k \in D_u^{known}} \gamma_l^k(\phi^d) \sum_j (v_j^k - \mu_l) \cdot (v_j^k - \mu_l)^T \end{array}}{\begin{array}{l} \sum_{d_k \in D_a} \gamma_l^k(\theta) \cdot |V^k| + \cdots \\ \cdots + \sum_{d_k \in D_u^{known}} \gamma_l^k(\phi^d) \cdot |V^k| \end{array}} & \text{if } c_l \in C_d \\[5mm] \dfrac{\sum_{d_k \in D_u^{unknown}} \gamma_l^k(\phi^{nd}) \sum_j (v_j^k - \mu_l) \cdot (v_j^k - \mu_l)^T}{\sum_{d_k \in D_u^{unknown}} \gamma_l^k(\phi^{nd})} & \text{if } c_l \in C_{nd} \end{cases} \tag{5.18}$$

Remark: the probabilities $p(a_i|c_l)$ are computed during step 1 of the algorithm and must also be updated during step 2 since the parameters associated with known mixture components are modified during this step. This is done using the same formula as in section 5.2.1. To compute unigram models which take into account the unknown image classes, we need to introduce annotations for "unknown elements"; this can be done in a very simple way by extending $p(a_i|c_l)$ to unknown mixture components. In other words, we just add concepts *"unknown1", "unknown2"* ... that we associate respectively with first, second, ... unknown mixture component with probability 1.

The Figure 5.8 shows the workflow of the steps employed in the algorithm 2.

5.4 Experimental results

In this section, we evaluate the procedure described previously on both synthetic and real data sets. A testing protocol on synthetic data is first presented in order to illustrate the capabilities of the model. We use synthetic data to test several configurations: we vary for instance the numbers of annotated and unannotated data introduced. We also employ synthetic data to compare our method with a semi-supervised SVM (SS-SVM) as described in [Bruzzone et al., 2006] (it should be noted though that the two methods are not fully comparable since SS-SVM does not allow to perform unknown structure detection). In the second part, we present some results on a database of satellite images. We do not possess a complete ground-truth on this database but it is well-suited for our purpose because of the huge amount of unannotated data and the potentially high number of unknown structures it contains.
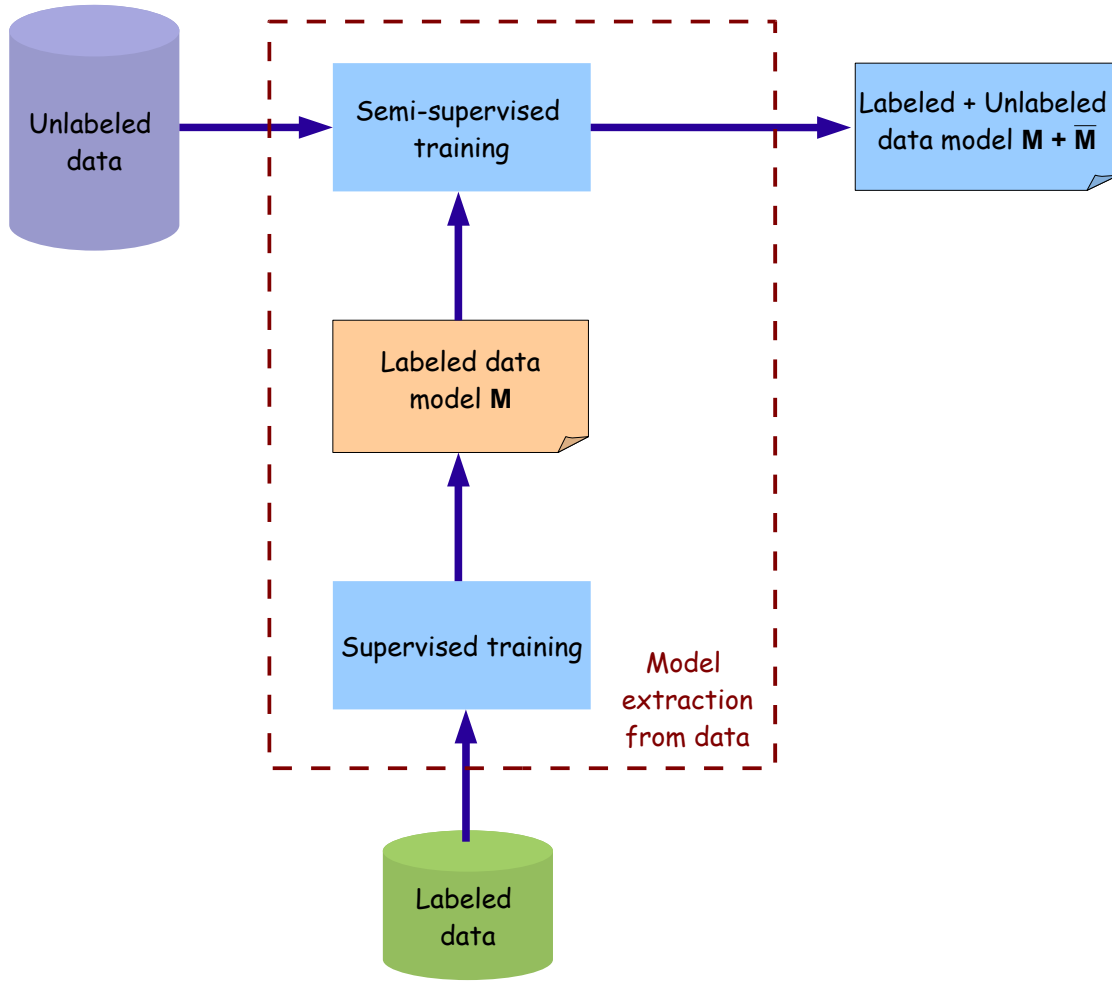
Figure 5.8: Workflow of the steps employed in the algorithm 2. M refers to the model of annotated (labeled) data which has been enhanced through the use of the unannotated data coming from known structures. $\bar{M}$ refers to the model of unannotated (unlabeled) data coming from unknown structures.

5.4.1 Evaluation on synthetic data

The choice of a synthetic dataset rather than a standard image database comes from the fact that standard databases are not well-suited to test semi-supervised methods. Most standard testing databases such as Caltech 101, Corel, Wang database [Wang et al., 2006] ... possess indeed an important number of different classes but a rather small number of images per class, which does not allow to place ourselves in a semi-supervised setting. A realistic scenario such as the one presented in the second part implies indeed having ten or even a hundred times as much unannotated samples as annotated ones, which is impracticable on the previously mentioned databases.

concept	a_1	a_2	a_3	a_4	a_5
mixture component	c_1	c_2	c_3, c_4	c_5, c_6	c_7, c_8

Table 5.2: Associations between image concepts and mixture components in the first testing scenario

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
a_1	1	1	0	0	0	0	0	0
a_2	0	1	1	0	0	0	0	0
a_3	0	0	0	1	1	0	0	0
a_4	1	0	0	0	0	0	0	1
a_5	0	0	0	0	0	1	0	0
a_6	0	0	0	0	0	1	1	0

Table 5.3: Associations between image concepts and mixture components in the second testing scenario

5.4.1.1 Description of the test scenarios

We start by generating the data according to a two-dimensional Gaussian mixture distribution. The samples thus generated are supposed to represent feature vectors extracted from images. For a more realistic simulation, we change the distribution corresponding to each mixture component; instead of using one single Gaussian component, we represent each mixture component as a sum of two Gaussians, one rather peaked and the other with larger spread. By moving the two centers slightly apart from each other, we get closer to real clusters which are often not strictly Gaussian and possess heavier asymmetric tails. The aim is not of course to simulate real features but to test our algorithm in the case where the data do not exactly fit the model; on condition that each cluster remains monomodal, the largest of the two Gaussians in each cluster will not be detected and the samples corresponding to that Gaussian will act as outliers. The advantage is that the overall distribution remains a Gaussian mixture (drawing from such a distribution is very easy) but still differs from the assumed distribution in term of the number of mixture components.

Assuming the existence of correlations between features, we use full covariance matrices. According to what precedes, a cluster c_l will be represented by the following distribution:

$$f(v|c_l) = \gamma \cdot \mathcal{N}(v; \mu_l, \Sigma_l) + (1 - \gamma) \cdot \mathcal{N}(v; \mu_l + \delta_{\mu_l}, \Omega_l) \tag{5.19}$$

where $0 < \gamma < 1$ and $\Omega_l = P_{\phi_l + \delta_{\phi_l}} \eta D_l P_{\phi_l + \delta_{\phi_l}}^T$.

P_{ϕ_l} is a rotation matrix whose angle ϕ_l is such that $\Sigma_l = P_{\phi_l} D_l P_{\phi_l}^T$. D_l is a diagonal matrix whose diagonal elements represent the variance along the two axes of the ϕ_l-rotated referential. δ_{μ_l} and δ_{ϕ_l} are slight "perturbations" that we add respectively to the mean and the orientation of cluster c_l. The overall distribution of data will thus be:

$$p(v|\theta) = \sum_{l=1}^{L} \pi_l \cdot f(v|c_l) \tag{5.20}$$

For our experiments, we take $L = 15$, and we choose arbitrarily the first eight mixture components to represent already-defined mixture components, the remaining seven representing "unknown" mixture components. Thus, $C_d = \{c_1, ..., c_8\}$ and $C_{nd} = \{c_9, ..., c_{15}\}$. We then define two possible scenarios for the annotations. In the first one, we suppose that one underlying mixture component can be associated with at most one "atomic" image concept. This corresponds to the case where an image in the training database is annotated with one single "atomic" concept. In the second one, we allow the underlying mixture components to be associated with several "atomic" image concepts. This corresponds to the case where an image in the training database may be annotated with several "atomic" concepts. In the first case, we speak of "grouping image concepts" and in the second of "overlapping image concepts". It should be noticed that the second scenario does not exclude the possibility of having also "grouping image concepts".

   - *First scenario:* We arbitrarily fix the number of "atomic" image concepts to five. The associations between concepts and mixture components are represented in Table 5.2. Samples are generated according to the distribution (5.19). Our procedure to generate annotated data follows the classical method to sample from a mixture distribution:

1. An integer $l \in \{1, ..., 8\}$ is drawn with probability $\pi'_l = \frac{\pi_l}{\sum_{j=1}^{8} \pi_j}$.

2. A sample v^k is drawn according to the distribution associated with the mixture component c_l: $f(v|c_l)$.

3. We obtain a new annotated sample, $d_k = \{v^k, a_i\}$, where a_i is the annotation associated with the mixture component c_l (cf. table 5.2).

The unannotated samples are drawn according to the whole mixture distribution (including both already-defined and "unknown" mixture components).

   - *Second scenario:* The only thing we change here in comparison with the first scenario is the association table between the mixture components and "atomic" image concepts. We use this time 6 image concepts that we associate with the mixture components according to Table 5.3. $table(l, i) = 1$ indicates that mixture component c_l is associated with image concept a_i. The third step of the training data generation procedure is changed into:

3. We obtain a new annotated sample, $d_k = \{v^k, A^k\}$ where $A^k = \{a_i | table(a_i, c_l) = 1\}$.

   It should be noted that the training documents d_k produced by the two procedures described previously contain only one feature vector, that is $|V^k| = 1$ (In the previous sections, we have described a method capable of handling several feature vectors per training document). This restriction is necessary to compare our method with the semi-supervised SVM described in [Bruzzone et al., 2006]. We shall see later the effect of using non-singleton ensembles V^k when we compare the two cases described in sections 2.1 and 2.2.

5.4.1.2   Evaluation of the performance and comparison with other models

In this section, we compare our method with a semi-supervised SVM and a traditional SVM. We have re-implemented the SS-SVM algorithm proposed in [Bruzzone et al., 2006] using `CVX`, a matlab package for solving convex programs [Grant and Boyd, 2008]. In the following, we present some results using the two testing scenarios described in
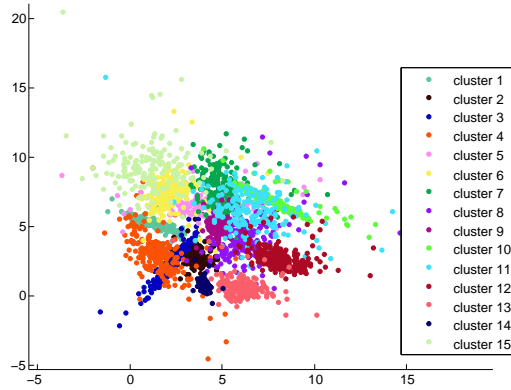
Figure 5.9: Underlying Gaussian mixture distribution used for testing

the previous section. All the following experiments are done using the same underlying Gaussian mixture density (cf. Figure 5.9). For the need of the experiments, we generate 10000 samples (5313 coming from already-defined mixture components and 4687 coming from "unknown" mixture components). Depending on the experiment, we use a part or the integrity of this dataset. For testing, we use an other group of 10000 samples generated independently from the first group. Among the testing samples, 5322 are coming from already-defined mixture components and the other 4678 are coming from "unknown" mixture components (since the sample generation process is based on randomness, we cannot ensure that the repartition of testing samples between known and unknown mixture components is exactly the same). Likewise, depending on the experiment, we use only the test samples coming from already-defined mixture components or the integrity of the test samples.

- *First scenario:* In this scenario, the annotations can be considered as class labels since we have exactly one annotation per training sample (case of "grouping image concepts"). We start by using only the part of the dataset which corresponds to already-defined mixture components (that is, the 5313 samples which can be annotated using "known" image concepts). Figure 5.10 represents the average overall accuracy when we vary the percentage of the total number of annotated data introduced (that is, the number of annotated data introduced over the total number of annotated data) (left) and when we vary the percentage of the total number of unannotated samples introduced, keeping unchanged the number (350) of annotated training samples (right). The traditional SVM has been trained using only the annotated part of the training dataset. In the left figure, we are using only annotated samples so we just make a comparison of our algorithm with the traditional SVM. We observe in the left figure that SVM-based methods are much more efficient when the number of annotated data is very low. This can be explained by the fact that the association probabilities $p(a_i|c_l)$ cannot be reliably estimated on a small training database. There are indeed $5 \cdot |C_d| = 40$ such probabilities to estimate. We observe on the right figure that adding unannotated samples leads to an increase in the overall classification accuracy for both our method and the SS-SVM. This comes from the fact that, here, unannotated samples have also been sampled using the already-defined part of the model. To prove that this accuracy improvement is not due to random deviation, we compute the standard errors
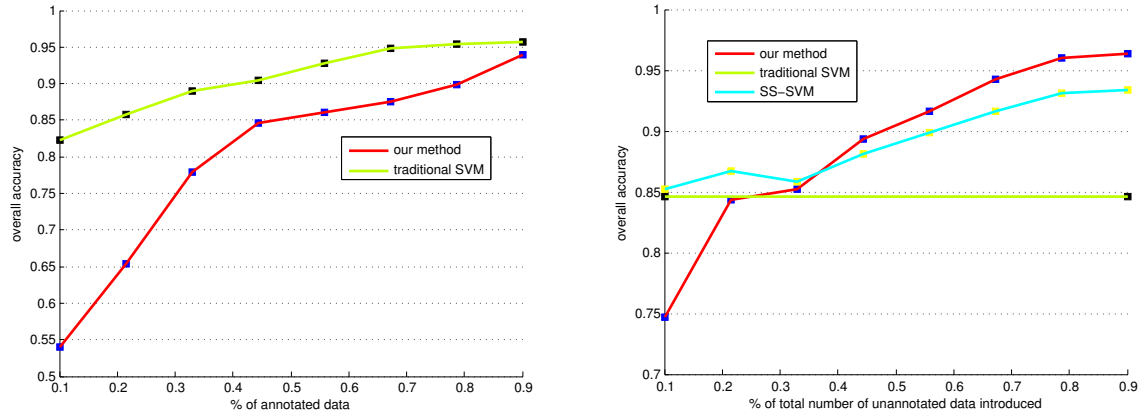
Figure 5.10: First scenario: Average overall accuracy when varying the percentage of the total number of annotated samples introduced (left) and when varying the percentage of the total number of unannotated samples introduced, keeping unchanged the number of annotated training samples (right). In this case, only unannotated samples coming from "known" mixture components are considered, which explains why SVM-based methods are competitive here

associated with the misclassification probabilities for each image concept, that is, the probabilities $p(a_i|\neg a_i)$, $i = 1, ..., 5$. We use a bootstrap estimator [Hollander and Wolfe, 1999] which relies on simple Monte Carlo simulation to compute the standard deviation of the sampling distribution associated with each misclassification probability. Figure 5.11 shows the results obtained using a fixed number of training samples and increasing each time the number of unannotated samples introduced in the learning process. We can see that as this number augments, the standard errors are decreas-
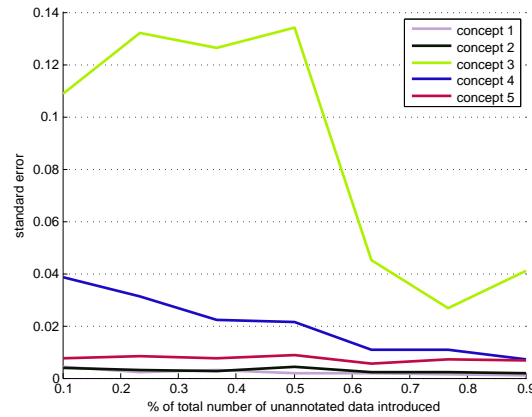
Figure 5.11: Standard errors associated with misclassification probabilities

ing, which means that the estimates of the model parameters are becoming more and more reliable. We observe though that standard errors are not strictly decreasing. This comes from the bootstrap estimator used to estimate these quantities. Like all methods
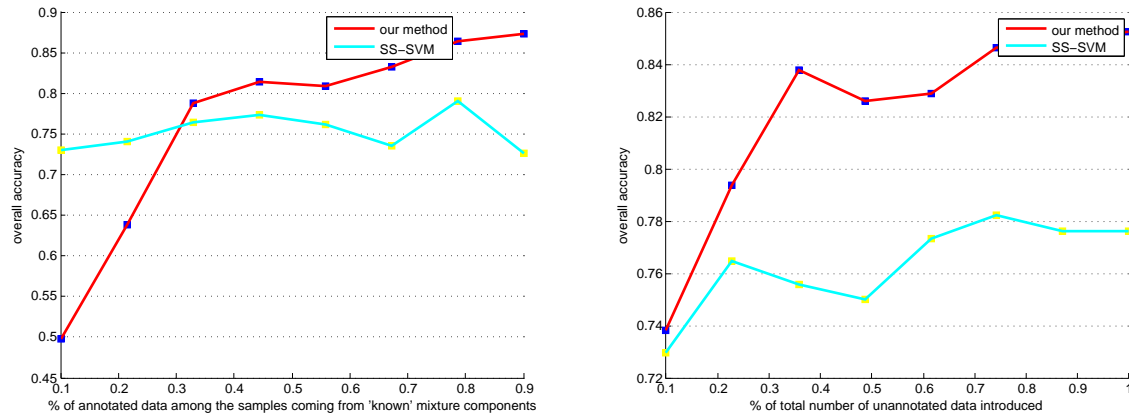
Figure 5.12: First scenario: Effect of introducing samples coming from unknown structures. Average overall accuracy when varying the proportion of annotated samples among the samples coming from known mixture components (we fix in advance the number of samples coming from known mixture components and from unknown mixture components) (left) and when varying the percentage of the total number of unannotated samples introduced, keeping unchanged the number of annotated training samples (right). In the two figures above, unannotated samples are coming from both "known" and "unknown" mixture components

relying on Monte Carlo simulation, the bootstrap estimator is only proved to converge asymptotically. We have repeated quite a high number of Monte Carlo experiments but we can still have random fluctuations in the estimates which cause the standard errors not to be strictly decreasing.

The experiments presented above are done using only samples coming from already-defined mixture components. This is the main reason why the SS-SVM is performing somewhat better than our method in these experiments (at least in the fully-supervised case). Our algorithm is indeed not competitive with highly discriminative methods like SVMs on a dataset containing only "known" structures, which are all represented in the annotated part of the training database. To see the effect of integrating the "unknown" part of the model, we repeat these experiments and we allow this time unannotated samples to be sampled also from the "unknown" mixture components (cf. Figure 5.12). We observe in the left figure that, in spite of the increase in the proportion of annotated training samples, the performance of the SS-SVM remains almost constant. We observe a similar behavior when we increase the number of unannotated samples, keeping unchanged the number of annotated training samples (right figure). Our method, on the contrary, is able to benefit both from the increase in the training sample size and the increase in the number of unannotated samples introduced. To see more directly the effect of the unannotated samples coming from the "unknown" structures, we vary the number of these samples, keeping fixed the number of annotated training samples and the number of unannotated samples coming from "known" structures. We can thus see (Figure 5.13) the influence of the "unknown" structures ("unknown" mixture components) over the training of the already-defined part of the model. When we decrease the number of unannotated samples coming from "unknown" structures, the accuracy increases (which is logical since the samples corresponding to "unknown"
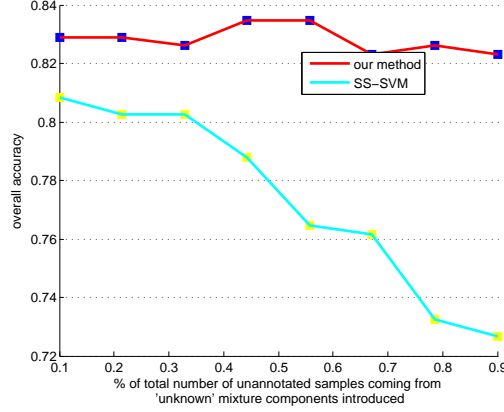
Figure 5.13: First scenario: Effect of varying directly the number of unannotated samples coming from "unknown" mixture components (the number of annotated training samples and the number of unannotated samples coming from "known" structures is fixed)

structures have a negative impact on the training of the already-defined part of the model). We also see that our method is much less influenced by these "wrong" samples than the SS-SVM. This is due to the fact that the algorithm we propose is able to categorize these samples into new categories (which we refer to as "unknown" mixture components) so that they are not taken into account in the training of the already-defined part of the model. The SS-SVM, on the contrary, will give good results only if the distributions of labeled samples and unlabeled samples used in the training process are very similar, which is not the case here by definition of "unknown" structures. In other words, the SS-SVM assumes labeled samples and unlabeled samples used in the training to represent a similar problem.

    - *Second scenario:* In this scenario, we can no longer consider annotations as class labels since several image concepts can be used to annotate the same sample. We need consequently another type of measure to assess the accuracy of the predicted unigram models for each sample. We use here a simple Euclidean distance, that is, given an unannotated sample (which is represented here by a feature vector v^k) and its predicted unigram model $\tilde{u}^k = \{p(a_i|v^k)\}_{i=1,...,6}$, we define a quantity $acc^k = \|\tilde{u}^k - u^k\|$ where u^k is the true unigram model (ground truth). We average then over the whole testing dataset to obtain an overall measure of accuracy like in the first testing scenario. The results are presented in Figure 5.14 (see Figure 5.10 for an explanation of what is represented). We can see in this figure (figure 5.14), when comparing with performances obtained in Figure 5.10, that our method is less affected by multi-labeling of training documents than the traditional and the semi-supervised SVM (as in Figure 5.10, we are only considering here samples coming from already-defined mixture components).

Remark: We have slightly modified the training database in the case of SS-SVMs since SS-SVMs cannot be trained on a testing database with multi-labeled samples. A training document d_k annotated with n concepts $a_1^k, ..., a_n^k$ is thus split into n single-labeled documents $\{v^k, a_1^k\}, ..., \{v^k, a_n^k\}$, which can be handled by a SS-SVM. To obtain unigram models also in the case of SVMs, we use the probabilistic output of standard SVMs.
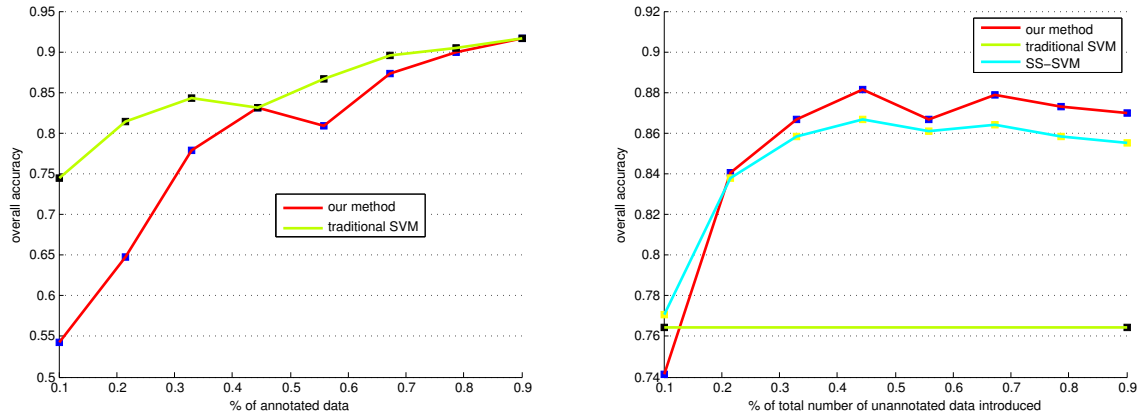
Figure 5.14: Second scenario: Effect of multi-labeling. Average overall accuracy when varying the percentage of the total number of annotated samples introduced (left) and when varying the percentage of the total number unannotated samples introduced, keeping unchanged the number of annotated training samples (right). In this case, only unannotated samples coming from "known" mixture components are considered, which explains why SVM-based methods still perform very well here

Probabilistic outputs can also be computed in the case of SS-SVMs (in both cases, probabilities of belonging to a class are computed as functions of the distance to the separation surface defining this class).

5.4.1.3 Explicit associations and no explicit associations: comparison

In this section, we compare the two cases when we know the explicit associations between feature vectors and image concepts inside a document and when we don't. We only show results in the fully supervised case. To generate the training documents, we re-use the generation procedure used in the first testing scenario. We slightly modify steps 2 and 3 to generate several feature vectors per document: instead of drawing only one feature vector in step 2 of the procedure, we draw N such vectors. We then proceed as before, using Table 5.2, to generate the annotations. Figure 5.15 represents the accuracy on the training database for different numbers of feature vectors per document. We see that when we don't know the explicit associations between image concepts and feature vectors, the accuracy is decreasing very quickly whereas it remains almost constant when we know these associations. In a real case, we would not have such a decrease since, most of the time, (and it is especially true in satellite imagery), the images are spatially coherent; it means that, inside documents which are usually small patches of an image, there isn't such a great variability between the concepts represented.

5.4.1.4 Effect of increasing the dimensionality

In this section, we study the effect of increasing the dimensionality of the feature space to assess the difficulty of the estimation problem. We generate a 15-dimensional data set consisting of 10000 samples according to a Gaussian mixture distribution with 15
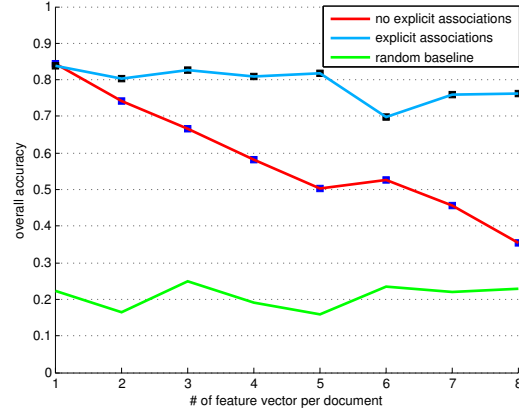
Figure 5.15: Explicit associations and no explicit associations: comparison of the two cases

mixture components. The means of mixture components are placed randomly in a fifteen-dimensional hyper-rectangle. The 15-dimensional covariance matrices are generated randomly using the positive semi-definite property of covariance matrices, i.e. given a covariance matrix Σ, there exists a normal matrix P and a diagonal matrix D such that $\Sigma = PDP^T$. The diagonal matrix D contains the variances in each direction. In our case, we generate the diagonal values according to a uniform distribution. The normal matrix P is more difficult to build. We use a Gram-Schmidt orthonormalization procedure, that is, we start by generating a random set of linearly-independent 15-dimensional vectors. We then apply the Gram-Schmidt orthogonalization procedure to this set to obtain an orthogonal basis $\{r_1, \cdots, r_{10}\}$. We then set $P = \{r_1|\cdots|r_{10}\}$.

We use the same number of "atomic" concepts as in the first testing scenario and the same associations between "atomic" concepts and mixture components (cf. Table 5.2). We do not use the second testing scenario since it does not allow an easy comparison of our method with the SS-SVM due to the multi-labeling of training instances. The figure 5.16 represents the average overall accuracy when we vary the dimensionality of the space. We add one dimension at a time starting from the two-dimensional case. We use a fixed number (350) of annotated training samples which are chosen randomly among the samples coming from known mixture components (but we keep the same annotated training samples for the whole testing process: we just make an update of these each time we add a new dimension). As in the first scenario, the tests are made on a different data set of 10000 samples drawn from the same Gaussian mixture.

Both our method and the SS-SVM are affected by the increase of the space dimensionality. The fact that our method still performs well in a high-dimensional feature space comes from the fact that we are using unlabeled samples to learn the model parameters. So, we do not encounter problems of sparsity which generally affect statistical methods when the training set size is very small. A similar study has been made by Langrebe et Al. in [Shahshahani and Landgrebe, 1994] where a theoretical insight is provided regarding the effect of introducing unlabeled samples to avoid sparsity issues as well as to increase the accuracy of estimators in a small sample size setting.
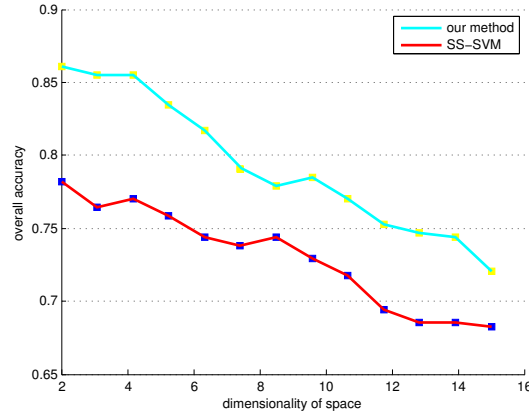
Figure 5.16: Effect of varying the dimensionality of the feature space, keeping unchanged the number of annotated training samples.

5.4.2 Demonstration on Earth Observation images

We use a testing database of sixty-four 3000×3000 pixels SPOT5 panchromatic images provided by CNES [1]. The ground resolution of these images is 2.5 meters. The advantage of satellite images is that they offer a huge diversity of structures provided we select images from various geographical locations. Moreover, the variations in soil, plantation type, etc cause conditional densities, that is, densities of the data given "atomic" concepts, to be multimodal, which is the case where latent variable models are expected to outperform more traditional methods. We cut each 3000×3000 image into smaller patches of size 64×64, which still represent quite a large ground area given the resolution. From each patch, we extract four feature vectors containing texture and shape parameters (we subdivide each patch into four regions to obtain four feature vectors). For texture, we use Haralick descriptors as well as Gabor and Quadrature Mirror filters [Campedel et al., 2004]. For shape descriptors, we use statistical moments extracted from image edges. To eliminate redundant information, we perform dimensionality reduction using Principal Component Analysis on texture and shape parameters separately. The final feature vector possesses 12 dimensions.

The training database we use consists of three hundred 64×64 images from the same sensor, which have been picked outside the testing database. In comparison, we can extract about 600000 such images from the testing database. Three "atomic" concepts are used to annotate these images: fields, urban area, and clouds. Examples of annotated samples are given in Figure 5.17. The images represented in this figure are quite "pure", that is, one of the three concepts "fields", "urban area", and "clouds" is generally sufficient to annotate these images but we may have in the training database training samples which possess more than one annotation, for instance, an image of a field shaded by a cloud or an image which is partly occupied by an urban area and partly by fields. In such cases, the images will appear in the training database with more than one annotating concepts.

To run our algorithm, we need an estimation of the number of mixture compo-

---

[1]French Space Agency (Centre National d'Etudes Spatiales), http://www.cnes.fr
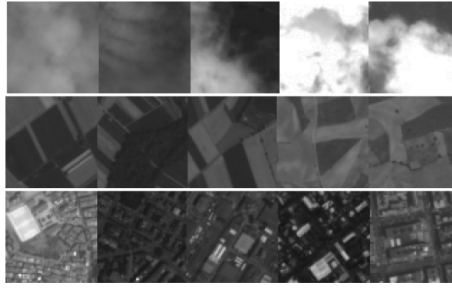
Figure 5.17: Examples of training images annotated with the three concepts "fields", "urban area", and "clouds".

nents among annotated and unannotated data respectively. We use the Minimum Description Length criterion (MDL) as described in [Kyrgyzov et al., 2007]. Other criteria like the Bayesian Information Criterion (BIC) [McQuarrie and Tsai, 1998] could be used as well or we could set a large number of mixture components at the beginning and eliminate insignificant ones during EM iterations by looking for instance at the singular values of the covariance matrices. Figure 5.19 represents posterior maps of "atomic" concepts. A zoom is shown in Figure 5.18. Tables in Figure 5.4 represent confusion matrices for the most probable concepts which are considered here as class labels. Ground truth has been assessed visually on images containing at most three known classes (fields, urban area, and clouds) plus several unknown classes. *Five unknown structures have been identified (see Figure 5.19) which can be matched with the following real categories (or at least part of them): mountain-like areas, wooded areas, desert, sea and urban structures such as airports or harbors.* It should be noted here that the annotation of unknown structures with meaningful concepts such as those enumerated above is done a posteriori by the user. Our algorithm is just able to identify these structures and provide them with annotations such as *unknown1, unknown2, ...* (cf. section 5.3.2). It belongs to the user to associate each unknown structure with a meaningful annotation.

From the confusion matrices, we compute the average probability of a known element to be correctly annotated $p(CA|known)$ and we see that it is higher using the semi-supervised approach (second table) than the fully-supervised one (first table). To prove this is not due to random deviation, we also compute the standard error

We have retained the two most represented unknown structures (desert and sea) and we have included these two structures inside the third confusion matrix which illustrates the performances of our semi-supervised algorithm. We observe that urban areas are often annotated as fields; this happens mostly around city boundaries where urban areas are less dense. Clouds are also often annotated as fields (this is often the case when we have transparent clouds with fields below). We can also remark that desert is often confused with fields: this comes from the fact that the two classes are very close (or at least possess modes which are overlapping at signal level since in most cases classes are multimodal). We see that the performance of our semi-supervised approach is better for the classes "urban areas" and "fields" but nor for the class "cloud". This can be explained by the fact the class "cloud" is almost monomodal at signal level. So, the representation of this class in the training dataset is quite complete, which explains why the SS-SVM performs somewhat better for this class.

From the confusion matrices, we compute the average probability of a known element to be correctly annotated $p(CA|known)$ and we see that it is higher using the semi-supervised approach (second table) than the fully-supervised one (first table). To prove this is not due to random deviation, we also compute the standard error

"stderr" on this statistic via simple Monte Carlo simulation and see that it is smaller in the second case. This second metric allows us to confirm the idea exposed in section 5.3 that we get more reliable estimates by integrating unannotated samples. We also compare our algorithm with a semi-supervised SVM (second confusion matrix). Since the SS-SVM does not allow to perform unknown structure detection, we represent only three concepts in the second confusion matrix but the training is done with the whole dataset (annotated plus unannotated samples). We observe that our method gives slightly better results than the SS-SVM in addition to the fact that it provides some valuable extra information by identifying unknown structures.

Figure 5.18: Zoomed areas of posterior maps obtained by applying our algorithm to panchromatic SPOT5 images. We can see from these images that our method allows a relatively accurate delineation of classes.

(a)

truth	**annotated**		
	urban	fields	clouds
urban	**0.69**	0.21	0.1
fields	0.22	**0.68**	0.1
clouds	0.04	0.16	**0.8**

(b)

truth	**annotated**		
	urban	fields	clouds
urban	**0.74**	0.17	0.09
fields	0.15	**0.77**	0.08
clouds	0.03	0.11	**0.86**

(c)

truth	**annotated**				
	urban	fields	clouds	desert	sea
urban	**0.79**	0.07	0.05	0.06	0.03
fields	0.05	**0.81**	0.02	0.09	0.03
clouds	0.01	0.12	**0.8**	0.07	0
desert	0.06	0.13	0	**0.78**	0.03
sea	0.02	0.02	0.03	0.01	**0.92**

Table 5.4: Confusion matrices. Table IV(a): fully supervised case using the algorithm presented in section 5.2.1: $p(CA|known) = 0.71$ and *stderr* = 0.01 ; Table IV(b): semi-supervised SVM: $p(CA|known) = 0.76$; Table IV(c): semi-supervised case with unknown structures discovery: $p(CA|known) = 0.8$ and *stderr* = 0.006.
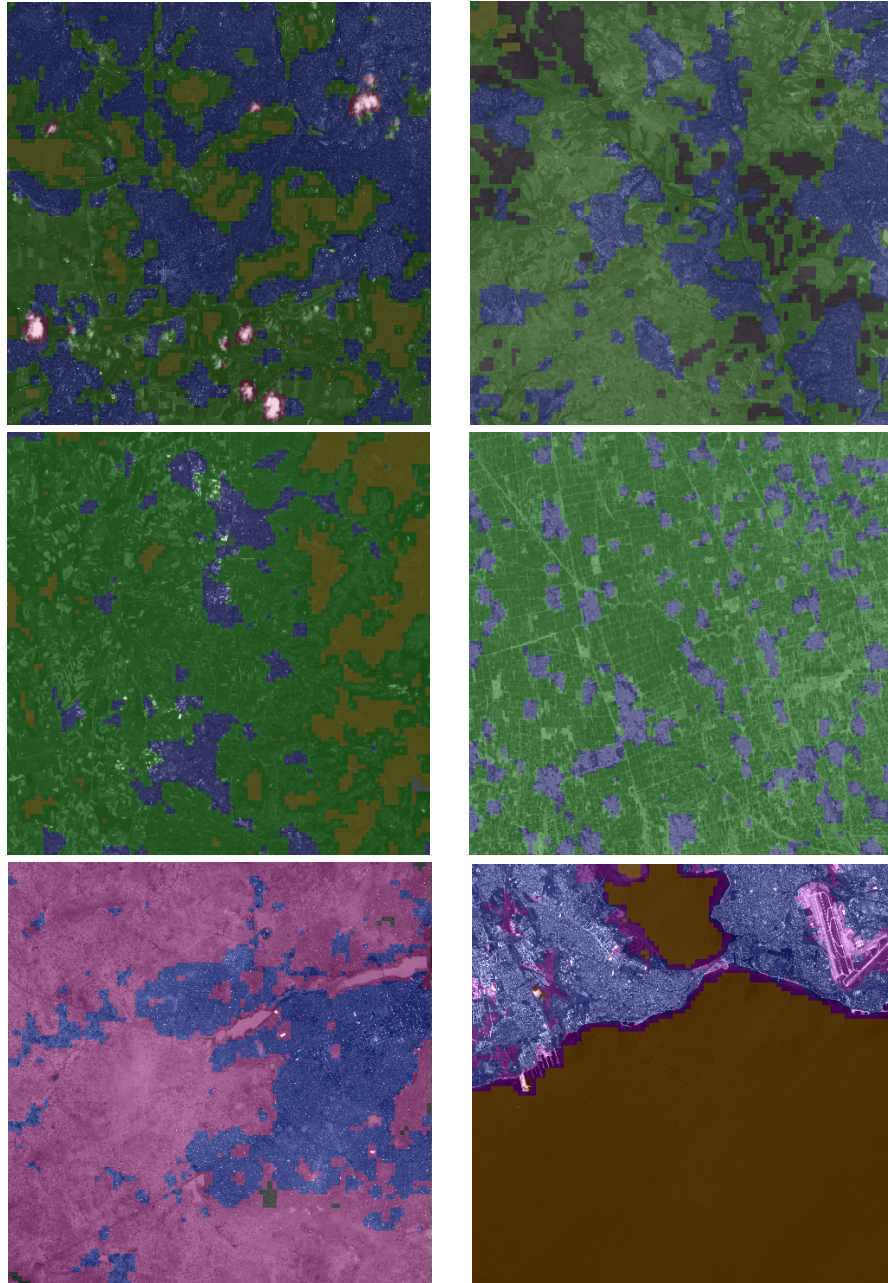
Figure 5.19: Posterior maps obtained by applying our algorithm to two panchromatic SPOT5 images: the different colors correspond to "atomic" concepts with highest posterior probability. Concepts represented here are "urban areas" (blue), "fields" (green), "clouds" (garnet-red, upper left image). Five additional unknown structures have been detected corresponding to mountain-like areas (dark grey, upper right image), wooded areas (khaki, upper left, upper right and middle left images), desert (pink, bottom left image), sea (orange-brown, bottom right image) and urban structures (purple, bottom right image).

5.4.3   A Deterministic Annealing Approach for Learning Finite Mixture Model Parameters: results

The main drawback of the presented semi-supervised approach concerns the difficulty to obtain accurate estimates of the model parameters. As in most techniques relying on Expectation Maximization to find the MLE of the model parameters, there is a risk of convergence of the proposed algorithm to a local minimum. A classical solution to remedy this problem is to use an annealing-like process. The confusion matrices in figure 5.20 are obtained using a modified version of the mass-constrained algorithm ([Rose et al., 1993]) to perform maximum likelihood estimation of the parameters of finite mixture models. This section present the results obtained using this algorithm, a complete description of which is given in appendix B. We consider only the fully-supervised case using a training database consisting of 5 classes. We compare our modified mass-constrained algorithm to a classical Deterministic-Annealing algorithm to perform MLE such as described in [Ueda and Nakano, 1998]. We see that using our algorithm, we obtain a perceptible amelioration on the diagonal of the confusion matrix of around 5.5% in average.

truth	annotated				
	class 1	class 2	class 3	class 4	class 5
class 1	0.8747	0.0349	0.0117	0.0592	0.0195
class 2	0.03	0.8026	0.0373	0.0786	0.0515
class 3	0.066	0.0554	0.7902	0.0431	0.0452
class 4	0.0168	0.0692	0.0387	0.8028	0.0725
class 5	0.0573	0.0068	0.0373	0.0663	0.8324

truth	annotated				
	class 1	class 2	class 3	class 4	class 5
class 1	0.8801	0.0198	0.0072	0.0644	0.0286
class 2	0.0164	0.8911	0.0051	0.0574	0.0299
class 3	0.0563	0.0192	0.8748	0.0415	0.0082
class 4	0	0.0716	0.0524	0.8307	0.0452
class 5	0.0205	0.0095	0.0239	0.0462	0.8999

Figure 5.20:   Confusion matrices. (top) training with the DAEM algorithm; (bottom) training with our modified mass-constrained algorithm. We obtain an amelioration on the diagonal of the confusion matrix of around 5.5% in average.

Further tests should of course be conducted by deriving a semi-supervised version of the algorithm described in B.

5.5   Conclusion

In this chapter, we have presented a semi-supervised procedure to auto-annotate images and discover unknown structures inside them. Performance appears to be superior to that of semi-supervised SVM when the majority of unannotated samples in the

training data reflect unknown semantic structures. Our method offers some advantages over a semi-supervised SVM like the possibility of using multi-labeled samples in the training database or the capability of discovering unknown structures among data. We have demonstrated that our algorithm can perform well on satellite imagery. The unknown structures we obtain are visually coherent and can be matched with real categories or at least part of real categories.

Chapter 6

Active learning using the data distribution for interactive image classification and retrieval

In this chapter, we describe a system that allows the users to explore large image data repositories and to build semantic models for their search targets. Since methods based on unsupervised learning are still far from providing acceptable results for open systems (where the user can search *anything*), we focus on iterative supervised methods by asking the user to asses iteratively the results suggested by the current model of the target class (relevance feedback). We take advantage of the intrinsic distribution of the data and we propose a new semi-supervised SVM algorithm based on Gaussian components that allows to process large data sets, while at the same time providing good generalization properties.

Our main application is to geospatial images, but the methods we develop are not specific to these and can be applied to less specialized images, such as those in multimedia databases. For this reason, we asses the performance of our method both for satellite images and for generic image databases.

Positioning, context and motivation   Traditional spatial image analysis techniques use local pixel characteristics to create a link between the low level features of the images and the high level descriptions needed by the users. The first generation of search engines proposed for satellite imagery worked essentially as filters, trying to combine knowledge of experts to low level image features to extract semantics relevant to the query (see, for example, KIM [Daschiel and Datcu, 2005], SIMR [Samal et al., 2009], RISE-SIMR [Bhatia et al., 2007]). Other successful methods include Bayesian classifiers used to represent land cover labels for pixels [Schröder et al., 2000a], hierarchical image segmentations for similarity retrieval [Tusk et al., 2003], visual grammars created by the automatic fusion of spectral, textural and other attributes [Aksoy et al., 2005], multiresolution hidden Markov models [Parulekar et al., 2005] and interest points [Newsam and Yang, 2007].

Another problem that often renders difficult the task of the retrieval systems is the "semantic gap" which separates the high level user-defined semantic concepts from the low level descriptors extracted from images. To alleviate this problem, recent Con-

tent Based Image Retrieval (CBIR) systems make use of a "relevance feedback" feature (which allows the users to mark retrieved images as relevant or non relevant [Zhou and Huang, 2003]) to iteratively learn a model of the query target. In this context, active learning has become a state of the art tool. At each retrieval step, it presents to the user the most informative items while trying to achieve two goals: first, to learn with accuracy the target concept and, second, to do so as quickly as possible with minimal effort from the user [Tong and Chang, 2001; Ferecatu and Boujemaa, 2007; Costache et al., 2006].

In this chapter, we focus on the second goal. The idea is to use a structuring of the database, obtained in our case with the help of an unsupervised generative model, to accelerate its exploration. The word "structuring" refers here to an intermediate representation situated between low-level descriptors and high level image semantic. In our case, we learn the parameters of a Gaussian mixture model (which can be seen as a probabilistic clustering) on the data. We then introduce a new semi-supervised C-SVM algorithm which works directly on the convex hulls of the mixture components and we describe an active learning strategy which consists first in readjusting at each iteration of the feedback loop the convex hulls of the mixture components and second in retraining our component-based SVM on these new convex hulls.

To illustrate the idea behind our approach, we compare the database to a book and the structuring to an index of this book. The underlying idea is that it is much faster when looking for something specific to navigate through the index of the book than to browse all its pages. The index provides us with a rough idea of the range of pages where the information we are looking for is situated and we can then proceed with a more local search by refining the range of pages provided by the index.

In our context, active learning is seen as a form of semi-supervised learning, where one has a relatively small number of labeled data (obtained from successive user feedbacks) and a huge amount of unlabeled data. As a quick reminder, the three main categories of methods that seek to exploit the intrinsic information contained in the unlabeled data are (a) statistical methods [Shahshahani and Landgrebe, 1994; Nigam et al., 2000], (b) kernel methods [Zhu, 2006] and (c) graph-based methods [Zhu, 2006; Zhou et al., 2005]. We provide an overview of these three categories of methods in section 3.4.1.

Among the state-of-the-art methods which get closest to the one we present in this chapter, several SVM-based approaches exist in the literature which make use of a structuring of the feature space in the form of a clustering. Although they do not envisage the problem of semi-supervised learning as such, they are similar in essence to what we propose, namely, they seek "to simplify" the learning process by positioning it at a lower granularity in the descriptor space, i.e. by considering bigger entities such as clusters in place of data points. Yu et Al. [Yu et al., 2003] have proposed a clustering-based SVM which exploits a hierarchical clustering to select better training samples for the SVM algorithm and thus reduce its complexity. Their aim is to design a SVM which scales with large datasets, but there is no suggestions regarding the use of unlabeled data. The clustering is performed on a labeled dataset and is not usable in our scenario, where we possess very few labeled training instances and a very large amount of unlabeled samples. In the same spirit, Boley et al. [Boley and Cao, 2004] have proposed an SVM algorithm which uses a clustering of the training dataset to

achieve a quicker identification of the support vectors and non-support vectors. Tsang et al. [Tsang et al., 2006] have proposed an algorithm which exploits the similarity between the formulation of the Minimum Enclosing Ball problem and the formulation of the SVM problem. As before, these methods do not apply to our learning scenario since they rely on a clustering of a labeled dataset and their aim is thus only to reduce the complexity of the standard SVM training procedure. The link with our work may be that we use a mixture of Gaussian components to model the intrinsic data distribution. However, a Gaussian component is not a cluster, but a statistical assumption about the data. We *do* produce clusters in our approach in the sense that we compute a convex hull for each mixture component, and we use active learning to adjust interactively the convex hulls (see Sec. 6.3.)

6.1   Mining image databases with adaptive convex hulls

In the following, we introduce a semi-supervised active learning method which incorporates the intrinsic data distribution in order to refine an SVM model of the target class using interactive relevance feedback and which is also able to manage large datasets. First, each satellite image is split into a collection of small superposing patches of 200 by 200 pixels and from each patch a set of image descriptors is extracted. A detailed description of this procedure is given in Sec. 7.7 (Experimental Results.) The database is thus seen as a set of images, each image being represented by a point in the description space. We use the Expectation Maximization algorithm to compute the intrinsic data distribution as a mixture of Gaussian components. Each mixture component generates a cluster as the set of data points situated inside its convex hull, which is the surface of equi-probability situated around the mean value (see Fig. 6.1 for some examples). In the following, whenever we use the word "cluster" it is understood that we refer to the cluster associated to a mixture component. The descriptors' space is thus seen as a topological structure where the size of the clusters are controlled by the convex hull of each Gaussian mixture component; however, we compute explicitly the clusters only for very few mixture components: those annotated by the users.

To start a search loop, the user is presented with the prototypes of all the mixture components and asked to mark as positive or negative those that are close to his query target. We thus make the implicit supposition that the target class can be seen as a union of existing clusters. While this is definitely limiting its generality, as long as the clusters are consistent, the method manages to get very good results on our test databases (see Sec. 7.7 for cluster consistency tests). Once this initialization step is over, the system enters an active learning loop the aim of which is to build iteratively an SVM classifier which approximates the concept targeted by the user. To do so, we define a modified SVM which works directly on the convex envelopes of the mixture components. The repositioning of the SVM surface is performed at each iteration of the active learning loop by first adjusting iteratively the convex hulls of the mixture components using the user feedback and second by re-training the modified SVM on the new convex hulls. The user provides feedback on the *critical points*, defined as the points of the convex hulls which are the closest to the current SVM decision surface, and which are re-computed at each iteration of the active learning loop.

Of course, the consistency of a cluster depends on its size, smaller clusters being
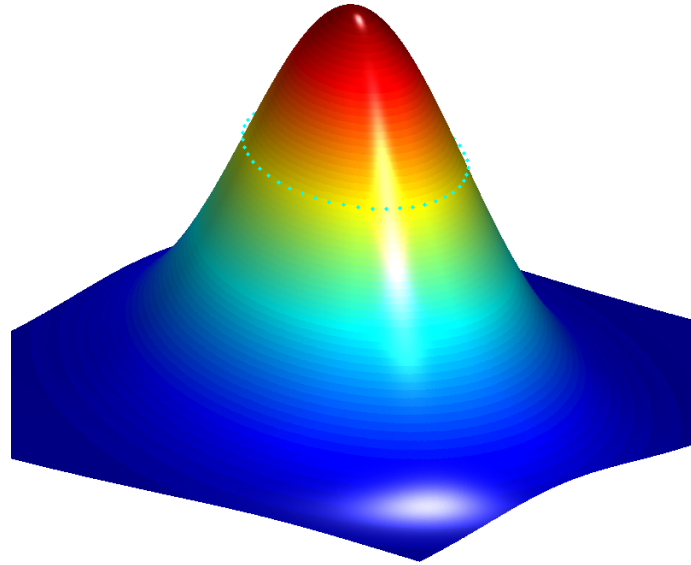
(a)
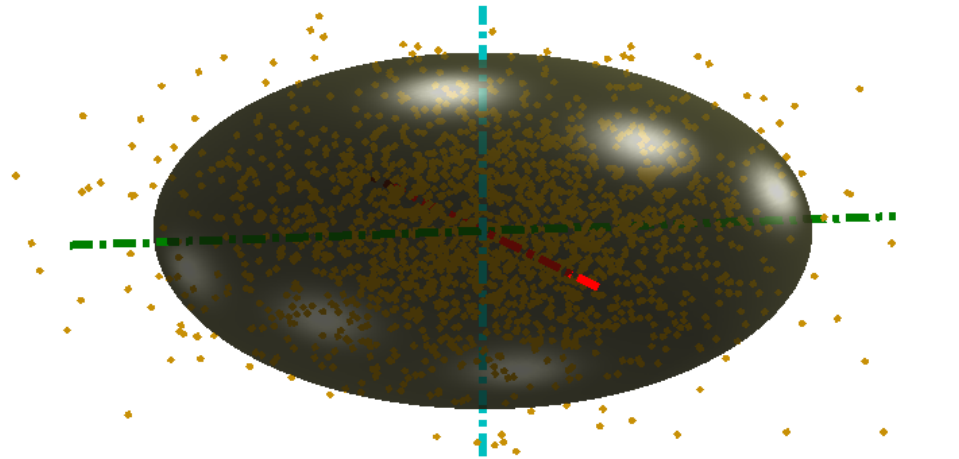
(b)

Figure 6.1: **(a)** Two-dimensional Gaussian mixture component with *an* equiprobable envelope (convex hull) which is in this case a two-dimensional ellipse (cyan blue dotted line). **(b)** Three-dimensional mixture component under its sampled form with *an* equiprobable envelope (convex hull) which is in this case a three-dimensional ellipsoid.

more likely to be consistent and larger ones being more loose. By dynamically adjusting the convex envelope during the active learning process, we use the user's supervision to find the right balance between accuracy (small clusters) and speed (larger clusters). If the clusters were consistent enough, to define the target class we would only need to learn the envelopes of the mixture components and the binary associative model between the mixture components and the target concept. The SVM classifier is in fact introduced to compensate for the lack of precision of the associative model which is very rough even with carefully adjusted envelopes of mixture components. It is indeed too strong an hypothesis to consider that the target class can be defined by a union of Gaussian clusters even if this is a good approximation in many cases (this is actually the assumption our method is exploiting to quickly position an SVM surface which is then used to refine the model of the target class). In the case of very complex target classes, the discriminative method we present will succeed whereas the probabilistic model associating the targeted concept to clusters will yield poor results because it seeks to solve a more general problem, that is, modeling $p(y|x)$, the posterior distribution of labels y given data points x, as a continuous density (which leads us to generally make very strong and restrictive assumptions on the form of the distribution $p(y|x)$) instead of directly solving the classification problem by simply estimating $p(y|x)$ on each data point x. This is one of the main issue pointed by Long et Al. [Long and Servedio, 2006] to explain why discriminative classifiers are still likely to yield acceptable results on problems where generative classifiers do not perform well. Even though the simple associative model we present does not belong to the class of generative models, it shows the same limitations as a generative approach which seeks to solve the very general problem of finding the distribution of $p(x|y)$. As a consequence, we do not expect our method to bring an improvement for all target classes: it all depends on the goodness of the adopted model of class labels given data points $p(y|x)$ (in our case, a union of Gaussian clusters). On classes that can be well approximated using a union of Gaussian clusters, our method converges much faster in terms of precision and recall than the baseline. On more complex classes, we have showed experimentally that our method does not perform worse and might even bring a small improvement since clusters always retain some consistency near their center (see Sec. 7.7).

More specifically, our key contributions are the following:

- We introduce a new semi-supervised C-SVM algorithm, that works directly on the convex hulls of the mixture components and we investigate its stability and convergence properties. This algorithm allows us to manage much larger volumes of data, without sacrificing too much the quality of the resulting learning model. In the following, we refer to this new form of semi-supervised C-SVM as the "component-based SVM".

- We propose an active learning scheme which relies on the user feedback to iteratively re-adjust the convex hulls of the mixture components involved in the training of the component-based SVM. To refine an SVM model of the targeted category, it is indeed faster (in terms of learning speed) to directly re-adjust the convex hulls of the mixture components than to work at the level of the data points.

- At last, we introduce an enhanced version of the preceding active learning algorithm which allows to interactively adjust the weights of the mixture components in relation to their relevance to the user's target concept. This component re-weighting approach allows both the unlearning of mixture components the user might have tagged erroneously during the learning process and the introduction of new mixture components during the feedback loop. Also, this allows the learning of the associative model between the mixture components and the targeted concept directly in the active learning loop. We call this approach *online learning*, as opposed to the initial strategy, called *batch learning*, which requires the user to annotate all mixture component prototypes before starting the active learning loop (such a scenario is not very realistic for the simple reason that the number of mixture components can be very high).

We use as a baseline a recent state-of-the-art active-learning-with-relevance-feedback method which have been reported to offer good results both on satellite images [Ferecatu and Boujemaa, 2007] and generic multimodal text and image databases [Ferecatu et al., 2008]. Experimental results obtained on a large dataset of QuickBird high resolution satellite images and on the well known Corel image database are very encouraging: our system scales much better to large databases compared to the baseline and in some cases offer more adequate results.

The chapter is structured as follows. In Sec. 6.2.1 we present the structuring of the description space. The component-based SVM is described in Sec. 6.2.2 and in Sec. 6.2.3 we discuss the convergence of the algorithm and its implementation. In Sec. 6.3 we introduce our active learning scheme. We consider here two scenarios: first, the case where the tagging of relevant components is done all at once at the beginning of the active learning loop (batch learning approach Sec. 6.3.1) and second, the case where components can be introduced progressively at each iteration of the active learning process (online learning approach Sec. 6.3.2). We continue with the experimental validation in Sec. 7.7 and we provide a quick conclusion in Sec. 6.5.

6.2 SVM and exploitation of the descriptor space structuring

6.2.1 Notations and structuring of the descriptor space

Our system is restricted by three important constraints: (a) Generality: the target class is not known *a priori*, thus the system must be able to function across a large range of query subjects; (b) Scalability: the system functions on large databases—here we work with hundreds of thousands of image patches; and (c) Real time: constraint specific to search systems; they must return some answers in a relatively short time (typically no more than a few seconds). These constraints guided many of our choices, including, as we shall see next, the choice of image descriptors.

Finding image descriptors that accurately describe the visual content of many different classes of images is a difficult task [Deselaers et al., 2008]. Such descriptors are easier to compute some specialized databases (for example medical images, fingerprints, or face images), where prior knowledge can be used to devise dedicated mathe-

1: Initialization step

The user is asked to mark the
relevant component prototypes ▲

2: User/system interaction

● ○ ● Critical points

4: Training of the new SVM model

● Critical points of S
● Critical points of S̄
⌒ New SVM surface

3: Readjustment of convex hulls

⬭ New (readjusted) convex hulls
⬭ Previous convex hulls
⌒ Current SVM surface
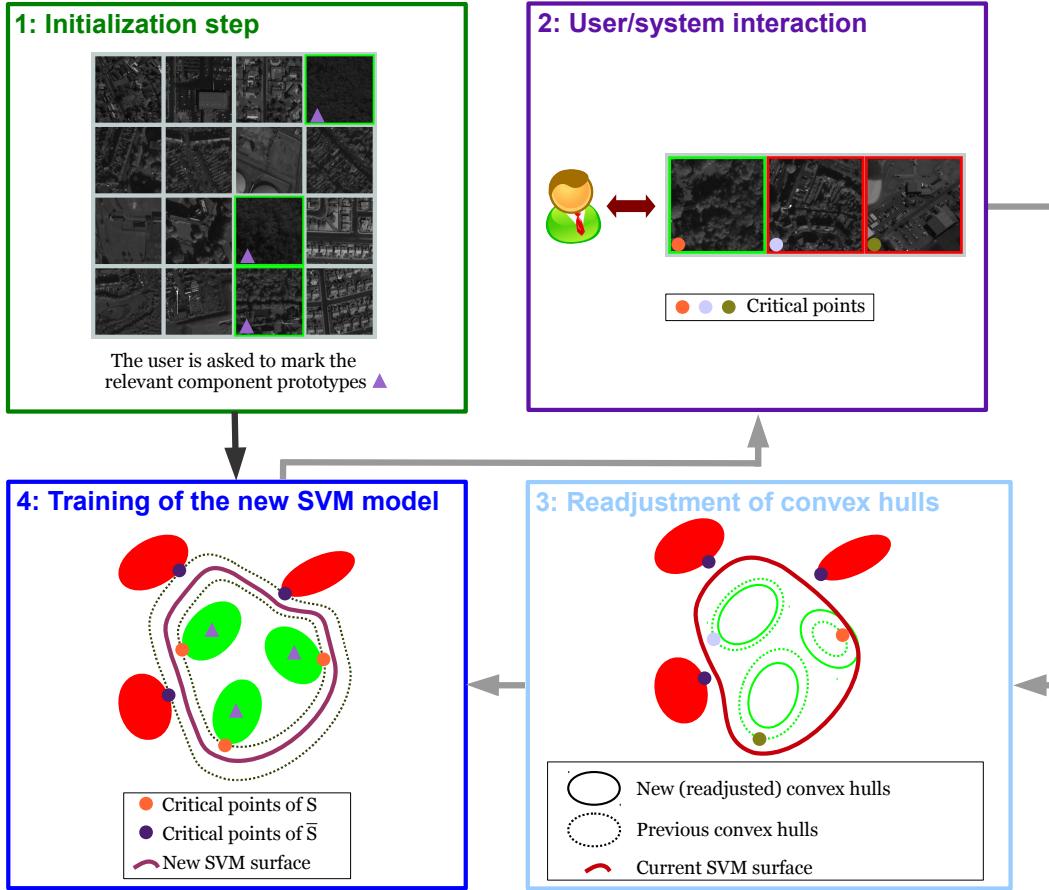
Figure 6.2: Schematic view of our approach. *Box 1:* The user is asked to tag all the images corresponding to the relevant Gaussian component prototypes. This is not a part of the active learning loop, it is an initialization step and is performed once at the beginning. *Box 2:* The user is asked to mark as relevant or non-relevant the images associated with the critical points (see Sec. 6.2.2) of components tagged by the user in the initialization step (Box 1). In the online-learning approach, the display is split between the critical points of components already annotated by the user and the prototypes of new components which are likely to match the user request. *Box 3:* The convex hulls of the mixture components are re-adjusted according to the user feedback in each of the critical points. *Box 4:* Re-training of the component-based SVM with the readjusted convex hulls of the components.

matical models of the image content. Local descriptors, e.g., points of interest or image regions, have been successfully used in several object detection tasks [Liu, 2006]. They may be fine-tuned to perform well for a some specific classes of objects, but they need large training sets, not available in a relevance feedback context. Moreover, they are resource intensive in terms of storage and computation, and consequently not well adapted to large-scale image retrieval systems that require answers in real time. Instead, we use a combination of different components of the image content, such as color, texture and shape [Datta et al., 2008; Lew et al., 2006; Deselaers et al., 2008], which is better adapted to generic images and, as in our case, to high resolution satellite images. They have small memory requirements, do not necessitate complex data structures or special handling and have been shown to perform well in our context, for example with SVM-based relevance feedback [Zhou and Huang, 2003] (see Sec. 7.7 for further details.)
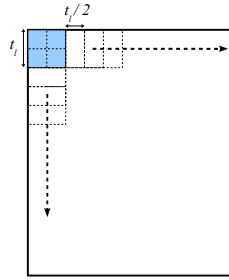
Figure 6.3: Sliding window used to divide a large satellite image into smaller patches.

In the following, we denote by v the feature vectors extracted from images as a concatenation of texture, color and shape descriptors. Since satellite images are very large, we use a sliding window to divide each image into several smaller patches (see figure 6.3). From each patch, we extract a single feature vector. The structuring of space is done by learning the parameters of a mixture model in the feature space $\mathcal{V}$. To do this, we assume that the feature vectors have a density that can be modeled by a mixture of L Gaussians:

$$p\left(v; \{\pi_l, \mu_l, \Sigma_l\}_{l=1,\dots,L}\right) = \sum_{l=1}^{L} \pi_l \cdot \mathcal{N}(v; \mu_l, \Sigma_l) \tag{6.1}$$

where the prior probabilities of each component of the mixture, π_l, are normalized such that $\sum_{l=1}^{L} \pi_l = 1$, and μ_l and Σ_l respectively denote the mean and covariance matrices associated with each Gaussian component. The parameters of the mixture model are learned using an Expectation Maximization algorithm [Hastie et al., 2005] initialized with the Enhanced LBG clustering algorithm [Patané and Russo, 2001]. As explained in paragraph 4.2, this algorithm minimizes the same functional as the standard K-means algorithm but is endowed with an heuristic to avoid convergence towards poor local minima. In Sec. 6.4.2 we discuss the enabling assumption (the mixture components are Gaussian) and we propose an heuristic method to determine the number of components in the mixture model.
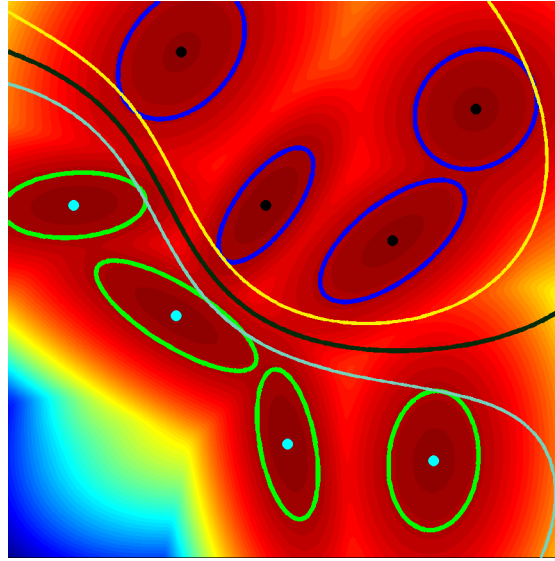
Figure 6.4: Idea of "low density separation": the surface separating the target S from the background $\bar{S}$ is constrained to pass through low density areas, represented here as color saturated areas.

6.2.2 Low density separation and component-based SVM

The idea we try to exploit, which is one of the key assumptions underlying the semi-supervised methods, is that of intra-cluster coherence: two elements belonging to a same region of high density are very likely to belong to the same class ("cluster assumption" [Chapelle et al., 2006a]). To re-use this idea and apply it to the SVM problem, we consider the equivalent formulation: the decision boundaries between classes are located in areas of low density, i.e. at the periphery of clusters (*low density separation*, see Fig. 6.4).

In the following, we use interchangeably the words "cluster" and "mixture component", the mixture model applied to the data being comparable to a probabilistic clustering, each cluster being delimited, for each given probability value, by the surface of equi-probability (see Sec. 6.1). The problem we are interested in is a two-class problem: the query target, S, and its complement in the database, $\bar{S}$. We also assume, similar to the case of hierarchical models discussed in 3.4.1, that the query target (known to the user) is modeled by an association of clusters in the description space. The difference is that, instead of probabilistic associations between concepts and mixture components, we consider binary associations of type 0 (the components is not relevant to the user query) or 1 (the component is relevant). Note that this assumption is not restrictive regarding the final shape of the classifier; rather, it is used as a starting heuristics to guide the construction of the SVM surface which will be refined later using the active learning process (see Sec. 6.3). This is justified by the fact that semantic concepts in satellite imagery often result in multimodal densities at descriptors level. For instance, consider the semantic class "agricultural land". This class corresponds to regions in a satellite image having each very particular texture characteristics, fact reflected by the appearance of several modes in the space of texture descriptors. We first deal with the

simpler case when the associations between the mixture components and the semantic class S defined by the user query are known. For the general case, we describe in Sec. 6.3.2 an online extension to learn these associations.

The first difficulty is to translate the idea of "low density separation" in our case. Because semantic classes can be multimodal, the density is no longer a sufficient criterion for characterizing the areas through which the separating surface has to pass (there may indeed exist areas of low density within the same class between modes). One way to constrain the problem is to force the separating surface to pass through areas of density as low as possible while remaining relevant to the associative model between the mixture components and the target semantic class. In the following, we note $\mathscr{L}_p(v) = p(S|v,\theta) = \frac{1}{p(v;\theta)} \sum_{l=1}^{L} \delta_l^S \cdot \pi_l \cdot \mathscr{N}(v; \mu_l, \Sigma_l)$ and $\mathscr{L}_{np}(v) = p(\bar{S}|v,\theta) = \frac{1}{p(v;\theta)} \sum_{l=1}^{L} (1 - \delta_l^S) \cdot \pi_l \cdot \mathscr{N}(v; \mu_l, \Sigma_l)$, where θ denotes the set of parameters of the associative model: $\theta = \{\delta_L^S, \pi_l, \mu_l, \Sigma_l\}_{l=1,...,L}$ with $\delta_L^S = 1$ if the l-th component is relevant to the query S and $\delta_L^S = 0$ otherwise. $\mathscr{L}_p(v)$ is the likelihood of the concept S at point v and $\mathscr{L}_{np}(v)$ is the likelihood of the concept $\bar{S}$ (see 6.5 for a complete derivation of the associative model). A point v of the separation surface must necessarily respect the equality $\mathscr{L}_p(v) = \mathscr{L}_{np}(v)$ so that the surface remains relevant regarding the associative model. This condition is sufficient to define a hypersurface $A = \{v \, ; \, \mathscr{L}_p(v) = \mathscr{L}_{np}(v)\}$. The idea of "low density separation" appears here through the fact that the surface A is necessarily located the farthest away from component centers because the two quantities $\mathscr{L}_p(v)$ and $\mathscr{L}_{np}(v)$ are very different in these places. The simplest idea to construct the SVM surface is then to seek to approximate the surface A by noting that the membership to S or $\bar{S}$ is given by the sign of $\mathscr{L}_p(v) - \mathscr{L}_{np}(v)$. Denoting by N the total number of vectors in the database, we try thus to build the following surface:

$$\min_{w,b,\xi} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \xi_i$$
$$\text{s.t.} \quad \begin{cases} \text{sign}(\mathscr{L}_p(v_i) - \mathscr{L}_{np}(v_i)) \cdot (w \cdot \phi(v_i) + b) \geq 1 - \xi_i \\ \xi_i \geq 0, \forall i = 1, ..., N \end{cases} \quad (6.2)$$

The problem in the above formulation is that all the points of the database will be involved in the learning process, which is not desirable in a context of active learning. Instead, we seek to identify M points $v_1^*, ..., v_M^*$ (not necessarily belonging to the database) which are sufficient to obtain a reasonable approximation of the surface. We call these *critical points*. For a problem with two components belonging respectively to S and $\bar{S}$, the two critical points will be the points of each corresponding cluster that are closest to the other cluster. The idea is to place oneself systematically in the worst case, i.e. to identify the points of each cluster such that the SVM classifier trained with this set of points has the smallest margin possible (see Fig. 6.5(a)). Using this criterion ensures that the obtained classifier retains a maximum of consistency with the associative model. Indeed, critical points obtained with this criterion are necessarily closer to the separating surface in the separable case. We will thus have (still in the separable case): $\mathscr{L}_p(v) \geq \mathscr{L}_{np}(v), \forall v \in D_S$ and $\mathscr{L}_p(v) \leq \mathscr{L}_{np}(v), \forall v \in D_{\bar{S}}$ (see Fig. 6.5(b)) where D_S (resp. $D_{\bar{S}}$) refers to the set of points where the obtained classifier decides S (resp. $\bar{S}$).

The above reasoning holds of course only if we are able to define a convex hull for
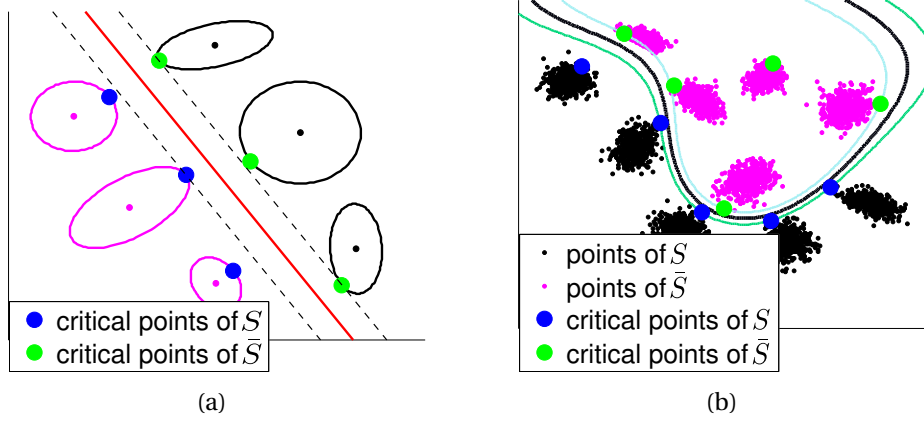
Figure 6.5: **(a)** Critical points in a linearly separable case. The magenta and black colors refer respectively to the sets S and $\bar{S}$. **(b)** Surface obtained by solving the problem (6.2) in the separable case. In black, points such as $\mathcal{L}_p(v) > \mathcal{L}_{np}(v)$ and in magenta, points such as $\mathcal{L}_p(v) < \mathcal{L}_{np}(v)$. The critical points are represented in blue and green.

each mixture component. We can solve the problem by noting that the equiprobable envelope of a Gaussian i.e. the set $\{v \, ; \, \mathcal{N}(v; \mu, \Sigma) = \rho_1\}$ is a multidimensional ellipse of equation $(v - \mu)^T \Sigma^{-1} (v - \mu) = \rho_2$ with $\rho_2 = -2 \cdot \log\big((2\pi)^{d/2} \det(\Sigma)^{1/2} \cdot \rho_1\big)$. The problem is then to determine ρ_1 in order to have $\mathcal{L}_p(v) = \mathcal{L}_{np}(v)$ at critical points. This issue will be examined in Sec. 6.3. For now, we consider that the constants ρ_1 are known for each component.

6.2.3   Algorithmic description of the component-based SVM

We have seen in the previous section that the problem is to identify L critical points $v_1^*, ..., v_L^*$ such that the SVM classifier trained with this set of points has the smallest possible margin. This problem can be reformulated as follows:

$$\max_{v_1^*, ..., v_L^*} \min_{w, b, \xi} \quad \frac{1}{2} \|w\|^2 + C \sum_{l=1}^{L} \xi_l$$

$$\text{s.t.} \begin{cases} (2\delta_l^S - 1) \cdot (w \cdot \phi(v_l^*) + b) \geq 1 - \xi_l \\ \xi_l \geq 0, \forall l = 1, ..., L \\ (v_l^* - \mu_l)^T \Sigma_l^{-1} (v_l^* - \mu_l) \leq \rho_2^l, \forall l = 1, ..., L \end{cases} \tag{6.3}$$

To solve this problem, it is possible to use an alternative max min scheme that converges under certain assumptions. We notice that the knowledge of critical points fully determines the solution of problem (6.3) and vice versa, knowing the parameters of the separating hyperplane completely determines the critical points. One can imagine a two-stage iterative scheme: we first determine the parameters w and b of the separating hyperplane to compute the critical points associated with this hyperplane. These points are determined as the points of the convex hulls of the mixture components which are closest to the hyperplane. We then compute the parameters of a new hyperplane using the critical points we just determined as the new training set. To ensure
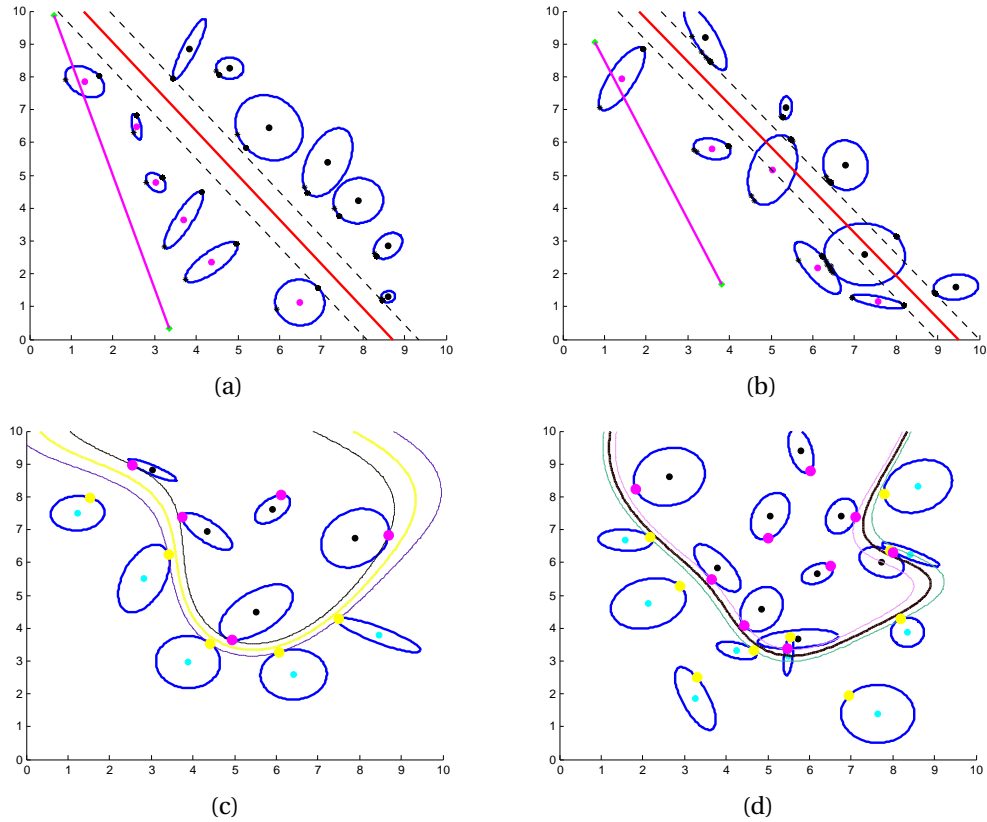
Figure 6.6: **(a) and (b) :** Application of the algorithm 1 in the (a) linearly separable case and (b) non-linearly separable case. The black dots on the ellipses represent critical points at different iterations of the algorithm. The red classifier is the classifier obtained after convergence of the algorithm. The magenta classifier is the one used to initialize the algorithm (a smarter way of proceeding and which often accelerates the convergence is to initialize the algorithm with a classifier trained with the cluster centers); **(c) and (d):** nonlinear classifier (Gaussian kernel) obtained with Algorithm 1 in the (c) separable and (d) non-separable case. The critical points are represented in yellow and magenta.

convergence towards a stable solution, we keep the critical points from previous iterations in the training set. With this condition, it is possible to prove the convergence of the alternating optimization scheme in the linearly separable case (see Appendix C.1), i.e. the convergence of (w, b) to a value (w^*, b^*).

As long as the clusters are linearly separable, it is clear that the proposed optimization scheme provides an alternative solution to the problem (6.2) — this solution of course does not match exactly the solution we would have obtained if we had solved the problem (6.2) as such since the points of the database are only an approximation of clusters. The algorithm 3 summarizes the alternative scheme proposed to solve the component-based SVM problem (6.3). In practice, convergence is very fast (4 or 5 iterations) and the size of the training set A_t remains very small even if completed with the new critical points at each iteration.

Note that in the non-linear separation case a cluster may yield support vectors in

---

Algorithm 3 Component-based SVM

   Initialization

   $A_0 = \left\{ (\mu_1, 2\delta_1^S - 1), ..., (\mu_L, 2\delta_L^S - 1) \right\}$

   Train C_0 with A_0

   While $A_{t+1} \neq A_t$ **do**

      Determine the critical points $v_1^*, ..., v_L^*$ associated with the current classifier C_t

      Train C_{t+1} with A_{t+1} where A_{t+1} is built from A_t by adding the pairs $(v_i^*, 2\delta_i^S - 1)$

      such that $\min\limits_{v_j \in A_t} \| v_i^* - v_j \| > \epsilon$

   End

---

several directions from the center. This is due to the fact that at each step of the algorithm 3, we retain the critical points from the preceding iterations.

6.3 Integration into an active learning scheme

In this section we propose a way to use the component-based SVM presented previously in an active learning context. We first consider the case where the mixture components relevant to the user request are identified all at once during the first iteration of feedback. We term this approach "batch learning of relevant components" (Sec. 6.3.1). We then extend this method to a more realistic scenario where the user is allowed to identify progressively the components that are relevant to the search target and we present an algorithm which assists him in doing so. We call this approach "online learning of relevant components" (Sec. 6.3.2).

6.3.1 Batch learning of relevant components

We have seen in Sec. 6.2.2 that building a SVM classifier approximating the surface of equal likelihood between the two parts of the model (S and $\bar{S}$) requires to know the constants ρ_1^l that define the convex hull associated with each mixture component. Since our model uses Gaussian mixture components, the convex hull of constant probability is a multidimensional ellipse with the directions of main axes given by the eigenvectors of the covariance matrix. The convex hull associates to each mixture component a cluster containing all elements of the database located inside it, and with its size controlled by ρ_1^l. The ideal situation would be to have clusters as large as possible, but that implies small values of the probability ρ_1^l, and thus there will be many data points that do not fit well with the semantics of the cluster. Using the "right" values for the constants $\rho_1^l, l = 1 \dots L$, is essential, but computing them explicitly is impossible because consistency of semantics with respect to the descriptors depends on the dataset. Instead, we determine them using the feedback from the user. In the following, we propose an heuristic that, on our test data, has proved to work quite well. The idea is to readjust the convex hulls of the mixture components based on the user feedback in each critical point (that is, for each of the images that are the closest to the critical points). In the following we denote by $P_n^* = \left\{ v_1^*(n), ..., v_L^*(n) \right\}$ the critical points obtained using Algorithm 1 at the n-th iteration of feedback.

(a)   (b)
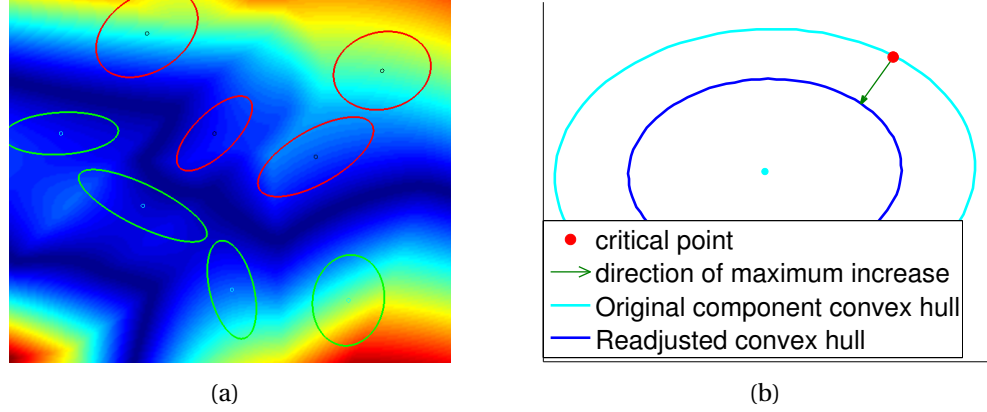
Figure 6.7: **(a)** The sum of likelihood ratios S_R: we see that the value of S_R on both sides of the zone of equi-probability (dark blue zone) increases very quickly. The green and blue ellipses represent the clusters associated respectively with the components of S and $\bar{S}$. **(b)** Adjusting the convex hull of a mixture component by progressing in the direction of maximum increase of S_R.

At each round of feedback, we display the images closest to the critical points, which requires the user to evaluate L images where L is the number of mixture components. It is better to identify the clusters that are very far from the separating surface, which, in practice allows to reduce considerably the number of images that the user must assess at each iteration. It is indeed unnecessary to modify the convex hulls of the mixture components that are far from the surface and thus not involved in the learning process. In the following, we propose a method to adjust the constants ρ_1^l based on the likelihood ratio at critical points. The idea is to look for the direction of maximum increase (steepest ascent) of this ratio and redefine the convex hulls of the mixture components based on the variation of the likelihood ratio when moving in this direction (see Fig. 6.7(b)). We do not use directly the likelihood ratio but rather the sum of two ratios:

$$S_R = \frac{\mathscr{L}_p}{\mathscr{L}_{np}} + \frac{\mathscr{L}_{np}}{\mathscr{L}_p} = \frac{\mathscr{L}_p{}^2 + \mathscr{L}_{np}{}^2}{\mathscr{L}_p \mathscr{L}_{np}}$$

which possesses better properties in the area of equi-probability. In particular, it presents a sharp increase on both sides of this area (see Fig. 6.7(a)). The use of S_R instead of the likelihood ratio can be justified in the following way: since the functions $\mathscr{L}_p$ and $\mathscr{L}_{np}$ are defined as linear combinations of Gaussians, they decrease very quickly outside their respective supports $Sup_{\mathscr{L}_p} = \{v; \mathscr{L}_p(v) \geq \mathscr{L}_{np}(v)\}$ and $Sup_{\mathscr{L}_{np}} = \{v; \mathscr{L}_{np}(v) \geq \mathscr{L}_p(v)\}$. It is thus a reasonable approximation to consider that $S_R \simeq \frac{\mathscr{L}_p}{\mathscr{L}_{np}}$ outside $Sup_{\mathscr{L}_{np}}$ and $S_R \simeq \frac{\mathscr{L}_{np}}{\mathscr{L}_p}$ outside $Sup_{\mathscr{L}_p}$. As a consequence, the gradient of S_R possesses the same orientation as the gradient of the likelihood ratios $\frac{\mathscr{L}_p}{\mathscr{L}_{np}}$ or $\frac{\mathscr{L}_{np}}{\mathscr{L}_p}$ respectively on $Sup_{\mathscr{L}_p}$ and $Sup_{\mathscr{L}_{np}}$.

The direction of maximum increase is determined using the gradient:

$$\nabla S_R = \frac{\mathscr{L}_p{}^2 - \mathscr{L}_{np}{}^2}{\mathscr{L}_p{}^2 \mathscr{L}_{np}{}^2} \left(\mathscr{L}_p' \mathscr{L}_{np} - \mathscr{L}_p \mathscr{L}_{np}' \right) \tag{6.4}$$

where:

$$\mathscr{L}_p'(v) = \nabla \mathscr{L}_p(v) = - \sum_{l=1}^{L} \delta_l^S \cdot \pi_l \cdot \mathscr{N}(v; \mu_l, \Sigma_l) \cdot \Sigma_l^{-1}(v - \mu_l)$$

The constant ρ_1^l is then determined by ensuring that the move d_l starting from the critical point v_l^* in the direction $-\nabla S_R(v_l^*)$ generates a variation $|\mathscr{L}_p(v_l^* + d_l) - \mathscr{L}_{np}(v_l^*)|$ which doesn't exceed a threshold α, i.e. $d_l = r_l \cdot n_l$ with $r_l = \arg_r \{ |\mathscr{L}_p(v_l^* + r \cdot n_l) - \mathscr{L}_{np}(v_l^*)| = \alpha \}$ and $n_l = \frac{-\nabla S_R(v_l^*)}{\|\nabla S_R(v_l^*)\|}$. To determine r_l, we use a linear dichotomic search. The value ρ_1^l is then determined by computing $\rho_1^l = \mathscr{N}(v_l^* + d_l; \mu_l, \Sigma_l)$. In the following, we denote by f_l^n the user feedback associated with the image closest to the l-th critical point at the n-th iteration of feedback. The value of f_l^n is 0 or 1 depending on whether the user finds the image relevant to his query or not. We decide that the constant ρ_1^l is sufficiently well adjusted when f_l^n changes value, i.e. when $|f_l^{n+1} - f_l^n| = 1$. The approach described above is summarized in the algorithm 4.

Algorithm 4 Adjustment of the convex hulls of the mixture components via an active learning process

---

  Initialization
  We start with a set $K_0 = \{\rho_1^1(0), ..., \rho_1^L(0)\}$ and we determine a first classifier C^0 using Algorithm 1.
  We set $F_0 = \{1, ..., L\}$ and $n = 1$
  While $F_n \neq \emptyset$ **do**
    For $l \in F_n$ **do**
      If $|f_l^{n+1} - f_l^n| = 0$ **then**
        compute $\rho_1^l(n)$ with the help of the critical points belonging to C_{n-1}
      Else
        $\rho_1^l(n) = \rho_1^l(n-1)$
        $F_n = F_{n-1} - \{l\}$
      End If
    End For
    Compute the classifier C^n using the algorithm 1 with the new set $K_n = \{\rho_1^1(n), ..., \rho_1^L(n)\}$
  End While

The algorithm we described considers only the case where the relevant components are marked all at once by the user during the first iteration of feedback. It does not include the possibility to add or remove components during the learning process, which might be problematic when the number of components is very high (we cannot reasonably ask the user to annotate a thousand components during the first iteration). We thus have to find a way to guide the user and help him explore the database

while continuing to improve the classifier on the already-annotated components. This two-sided problem is often referred to in the literature as the compromise between database exploration and exploitation. The exploration aims at roughly building a prototype in the feature space of the category the user is looking for. The exploitation is in some sort the complementary operation since it aims at improving the prototype built during the exploitation step. Thus, the exploration feature will improve the recall while the exploitation feature will improve the precision. In our case, the exploration will consist in identifying (as exhaustively as possible) the components which are relevant to the user request while the exploitation will consist in adjusting the convex hulls of the mixture components. This motivates the introduction of the "online learning" approach in the next section.

6.3.2   Online learning of the relevant mixture components

The method introduced in the previous section has two drawbacks: (1) it asks the user to annotate the prototypes of all mixture components, which can be a lot a images for large databases and (2) lack of flexibility: it does not allow unlearning of components introduced by mistake, or adding new components to the target model. In this section we remedy this situation and we describe a strategy to help the user in his database exploration task. The operational flow of the system is described in Fig. 6.8. The idea is to adjust interactively the weights of the components in the associative model at each round of feedback. For this, we no longer consider a binary associative model between mixture components and semantic concepts but a probabilistic model. Using Bayes rule, we obtain:

$$p(S|v;\theta) = \sum_{l=1}^{L} p(S,c_l|v;\theta) = \frac{1}{p(v;\theta)} \sum_{l=1}^{L} p(S|c_l;v;\theta) \cdot p(c_l;\theta) \cdot p(v|c_l;\theta)$$

$$= \frac{1}{p(v;\theta)} \sum_{l=1}^{L} p(S|c_l) \cdot \pi_l \cdot \mathcal{N}(v;\mu_l,\Sigma_l) \tag{6.5}$$

where $\theta = \{\pi_l,\mu_l,\Sigma_l\}_{l=1,...,L}$ and $p(v;\theta)$ is the value of the underlying Gaussian mixture density at point v that is given by equation 6.1. In practice, we do not need to compute this quantity since it disappears when computing the ratio ∇S_R 6.4.

By setting $p(S|c_l) = \delta_l^S$, we are in the particular case of the binary associative model such as described in section 6.3.1. By using the probabilistic associations between mixture components and semantic concepts, it is possible to introduce an unlearning component by adjusting the a posteriori probabilities, $p(S|c_l)$, of each component $c_{l=1,...,L}$ to belong to the semantic class S. The unlearning feature proved to be useful in the case where the user slightly changes his request during the learning process. This might happen when the number of components is very high, in which case, a real user is often tempted to mark as relevant the components which partially match what he is looking for and not necessarily the components which exactly correspond to his request (to do so, he would have to browse all the component prototypes which can be quite tedious). The aim of the system is then to guide the user towards the relevant components while forgetting progressively the previously annotated components which might not be pertinent regarding the user's request. The unlearning feature can
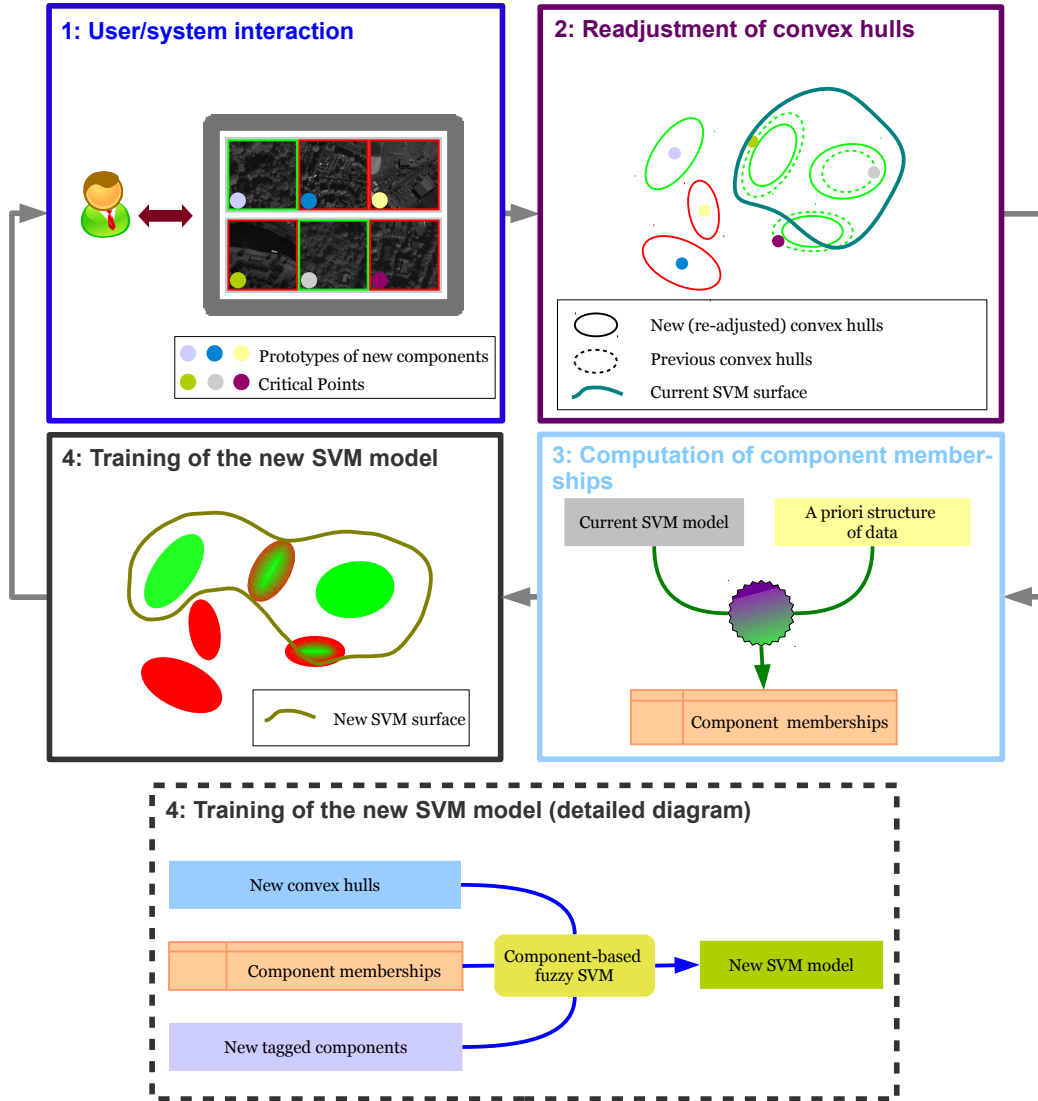
Figure 6.8: Proposed concept for the online learning of relevant components. *Box 1:* the user is asked to mark as relevant or non-relevant the images associated with the critical points of clusters corresponding to components tagged during previous iterations of the active learning loop as well as some images associated with prototypes of new components. *Box 2:* the convex hulls of the corresponding mixture components are re-adjusted according to the user feedback in each of the critical points. The new components whose corresponding prototypes have been tagged by the user are integrated with arbitrarily defined sizes of convex hulls. *Box 3:* component membership degrees (weights) inside the associative model components/user concept are computed according to the current classifier and the intrinsic model of data. *Box 4:* a fuzzified version of our component-based SVM algorithm is then trained with the re-adjusted convex hulls of the mixture components and the new components tagged by the user. The gradients of color inside the ellipses represent the weights of the mixture components in the associative model. These weights are used to train our component-based SVM with fuzzy membership degrees of components.

also be useful to compensate for the lack of coherence of certain clusters by reducing the weights of the corresponding mixture components in the associative model. The probabilities $p(S|c_l)$ are thus re-computed at each iteration of feedback. We have:
$p(S|c_l) = \sum_{v\in\mathcal{V}} p(y_v = 1, v|c_l) = \sum_{v\in\mathcal{V}} p(y_v = 1|v, c_l) \cdot p(v|c_l) \simeq \sum_{v\in\mathcal{V}} p(y_v = 1|v) \cdot p(v|c_l)$
where y_v is the label of the feature point v ($y_v \in \{1, -1\}$). The probabilities $p(v|c_l)$ are computed using the underlying probabilistic model, giving $p(v|c_l) = \mathcal{N}(v; \mu_l, \Sigma_l)$. As for the probabilities $p(y_v = 1|v)$, they are calculated using the output of the current SVM classifier at the n-th iteration of the active learning loop: denoting by f_n the decision function associated with this classifier, we use a sigmoid function to compute a probabilistic output of the following form:

$$p_v = p(y_v = 1|v) = \frac{1}{1 + \exp(-a_n \cdot f_n(v) + b_n)}$$

Considering that the variable y_v follows a Bernoulli law of parameter p_v (the attribution of the label y_v is similar in nature to a *yes/no* experiment with a probability of success being equal to p_v and a probability of failure equal to $1 - p_v$ and, thus, can be modeled as a Bernoulli trial), we have: $p(y_v|v) = p_v^{\gamma_v} \cdot (1 - p_v)^{1-\gamma_v}$ where $\gamma_v = 1$ if $y_v = 1$ and $\gamma_v = 0$ if $y_v = -1$. The coefficients a_n and b_n of the sigmoid are adjusted using a maximum likelihood estimator upon the training data i.e. $(a_n, b_n) = \arg\max_{a_n, b_n} \prod_{v\in T_{f_n}} p_v^{\gamma_v} \cdot (1 - p_v)^{1-\gamma_v}$ where T_{f_n} is the training set used to train the classifier f_n. To perform the optimization, we use Platt's algorithm such as described in [Platt, 2000]. To take into account the probabilities associated with each component, we use a fuzzified version of the algorithm 3, that is, we solve the following problem:

$$\max_{v_1^*,...,v_L^*} \min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{l=1}^{L} \tau_l \cdot \xi_l$$

$$\text{s.t.} \begin{cases} (2\delta_l^S - 1) \cdot (w \cdot \phi(v_l^*) + b) \geq \tau_l - \xi_l \\ \xi_l \geq 0, \forall l = 1, ..., L \\ (v_l^* - \mu_l)^T \Sigma_l^{-1} (v_l^* - \mu_l) \leq \rho_2^l, \forall l = 1, ..., L \end{cases} \quad (6.6)$$

The algorithm we use to solve the optimization problem above is similar to the one used to solve the problem (6.3), with the difference that we replace the standard SVM with a probabilistic SVM inspired from [Liu, 2006] in the second step of the `while` loop of the algorithm 3. In the above probabilistic SVM formulation, each training element must be given a membership $0 \leq \tau_l \leq 1$ which is the belief that the training sample v_l belongs to the class $y_l = 2\delta_l^S - 1$. We propose to compute τ_l for each component as $\tau_l = p(S|c_l)/\text{ND}_l$ if $c_l \in S$ and $\tau_l = 1$ otherwise. The normalization coefficient ND_l is computed as $\text{ND}_l = \sum_{v\in\mathcal{V}} p(v|c_l)$. It ensures that $0 \leq \tau_l \leq 1$. Thus, we have $\tau_l = \delta_l^S \cdot p(S|c_l)/\text{ND}_l + (1 - \delta_l^S)$.

The constraints $(2\delta_l^S - 1) \cdot (w \cdot \phi(v_l^*) + b) \geq \tau_l - \xi_l$ in the problem 6.6 allow to account for unbalanced memberships by shifting the SVM boundary towards the most uncertain training samples. As in the standard SVM, ξ_l are slack variables which penalize the objective function in the case of misclassification. The problem 6.6 is solved under its dual form yielding the final objective function: $f(v) = \sum_{l\in SV} (2\delta_l^S - 1) \cdot \alpha_l \cdot k(v, v_l) + b$ where the α_l are the coefficients associated with the support vectors SV and k the kernel function.

In the algorithm 8, the weights τ_l are only computed for the labeled components. The weights of the unlabeled components are all set to 1: the unlearning component concerns indeed only those components which have been tagged as relevant by the user in a former iteration of the active learning loop. Since the coefficients τ_l are degrees of membership and not probabilities, we do not need any normalization.

In the following, we briefly describe the strategy we have retained for exploring the database. The goal is to point to the user the components which are more likely to correspond to his request. A quite straightforward strategy is to present to the user the components which are close to the ones he has already marked as relevant: it ensures a certain consistency in the database exploration while exploring uncharted territories near the already tagged components. In our experiments, it proved to work much better than picking components at random or based on their proximity to the SVM separating surface. The main reason is that the concepts we are looking for are represented in the feature space by several "superclusters" i.e. by several groups of clusters (cf. Fig. 6.9(a)). In other terms, the semantic concepts consist of several homogeneous sub-classes. So, it makes more sense to proceed with a local search near the already tagged components than to look for new components far from the preceding ones. The data exploration feature is preserved by the fact that we work directly at cluster levels, which ensures a certain sparseness of the training data set. Considering a Gaussian kernel for the SVM and a kernel bandwidth sufficiently small, the components close to the already tagged ones will be those situated in a stripe $[T, +\infty]$ in the high dimensional SVM feature space (with T sufficiently high). The notion of stripe is defined here using the distance $d(., f)$ to the separating surface f (the distance d is defined as the value given by the decision function associated with the SVM model and it is positive since we only look on the positive side of the SVM decision surface f). Thus, a point v will belong to the stripe $[T, +\infty]$ iff $T \leq d(v, f) < +\infty$. The advantage of such an approach is that we can adjust the range in which we want to perform the search for new relevant components around the already marked ones (see Fig. 6.9). Such an approach works well when the SVM surface is sufficiently well-positioned in the feature space, which is not the case at the beginning of the learning process. So, we have first to obtain an approximate positioning of the "superclusters" which define the class we are looking for. The idea is to start with a broad search over all components and then to restrain progressively the search to the components closest to the already tagged ones. An example scenario is given in Fig. 6.9.

We use an annealing-like process on the "temperature" parameter T which tunes the distance from the separating surface beyond which the system is allowed to search for relevant components. The temperature parameter is decreased at each iteration of the feedback loop to limit step by step the exploration of the database to the components farthest to the SVM surface. To formalize this approach, we use a Gibbs distribution taking as the energy function the components distance to the separating surface f:

$$p(c_l \in S) = \frac{1}{Z_C} \cdot \exp(-\frac{d_{max} - d(c_l, f)}{T})$$

where $d_{max} = \max_{l \in S} d(c_l, f)$, $Z_C = \sum_{l \in S, d(c_l, f) > T} d(c_l, f)$. In these formulas, the notion of distance of a mixture component to the separating surface f is understood as the distance of the critical point associated with the mixture component to the SVM sepa-
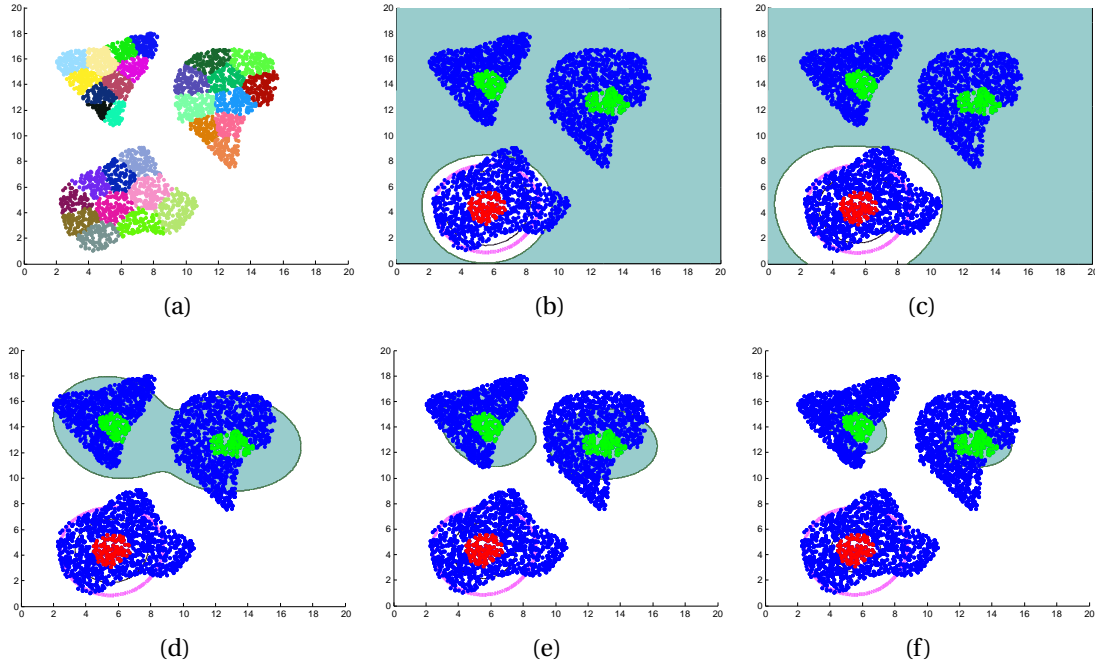
Figure 6.9: **(a)** Superclusters; **(b), (c), (d), (e)** and **(f)** Illustration of the cooling scheme. In each figure, the light blue area represents the zone inside which we perform the search for new target components. We start with a high value of T, which amounts to perform a broad search over almost all components (figure (b)) except the one tagged as non-relevant (in red) and we progressively decrease the value of the temperature parameter (the figures are ordered alphabetically ((b), (c), (d), (e) and (f)) in function of their decreasing associated T value). We then see that the search area becomes more and more concentrated around components tagged as relevant (green ones).

rating surface f. Also, it should be noted that the Gibbs distribution here is used more as a formalism than a real modeling.

We use a display which is split into three parts to account for the two different strategies used in the database exploration feature on one hand and the database exploitation feature on the other hand (see Fig. 6.22). The respective sizes of the displays corresponding to the three parts of the graphical interface are referred to as K_1, K_2 and K_3 in the following. At each iteration of the active learning loop, K_1 components are sampled according to the Gibbs distribution. When the parameter T is high, the influence of the distance to the separating surface is less important in the exponential term of the distribution, so components close to the surface can be sampled as well. On the contrary, when we decrease T, the influence of the distance to the separating surface increases, making components far from the surface more probable. To ensure that the exploration is performed on both sides of the separating surface, we still maintain a most ambiguous (MA) feature, that is, we return the K_2-th components closest to the separating surface. The exploitation is done in a very simple way, by returning on the display the K_3 images corresponding to the critical points associated with the K_3 labeled components which are closest to the separating surface. The problem is to optimize the display, that is, values $K_{12} = K_1 + K_2$ and K_3 so as to find the correct

compromise between the exploration of new regions and the refinement of the class current estimate.

The algorithm 5 summarizes the procedure described above (the sub-programs Sample$_{\text{Gibbs}}$ (algorithm 7), ComputeComponentMembership (algorithm 8) and ComponentBasedFuzzySVM (algorithm 9) are presented in the appendix C.3). The following notations are used: $K = K_1 + K_2 + K_3$ is the size of the display, n is the current iteration of the active learning loop, L^n_{comp} and U^n_{comp} are respectively the set of labeled and unlabeled components at iteration n, $Closest^n(D, k)$ refers to the k closest components to the current decision surface at iteration n among the set of components D, the distance of a component to the current decision surface being defined as in 6.3.2 as the distance of the current critical point associated with this component to the current decision surface. $CP^n(D)$ are the critical points associated with the set of components D at the n-th iteration. $d^n_{comp}(D)$ are the distances of a set of components D to the current decision surface at iteration n. L^n_{pts} and U^n_{pts} refer respectively to the set of labeled and unlabeled data points at iteration n. $Comp$ is the set of components in the mixture model. Prot(D) are the corresponding component prototypes associated with the set of components D. f^n is the current decision surface at iteration n and it also refers in our formalism to the decision function associated with the classifier.

6.4   Experimental results and comparison

To assess the performance of our approach, we conducted tests on two datasets. The first is a set of high-resolution QuickBird satellite images and the second consists of images 60,000 images from the Corel Stock Photo Library [1], which is a widely used image database in the image retrieval community. For each dataset, we evaluate the retrieval capabilities of our system both in terms of precision and recall and we assess the generalization capabilities of the obtained classifier by looking at the number of support vectors defining the SVM surface. We also discuss the choice of the number of mixture components, choice which influence the intra-cluster consistency (an important enabling assumption for our system). Other features like the ability to forget irrelevant clusters are also evaluated. We conclude this section by discussing which strategy performs best with respect to the choice of the new component prototypes to be presented to the user at each iteration of the active learning loop. We compare our strategy which consists in sampling new component prototypes according to a Gibbs distribution with the classical most ambiguous and most relevant strategies used by our baseline [Tong and Chang, 2001; Ferecatu et al., 2008]. According to the discussion in 6.3.2, we also perform tests using a mixed strategy which consists in using our Gibbs sampling based strategy along with a most ambiguous strategy, as proposed in the Algorithm 3. We incidentally evoke graphical interface issues: our interface is indeed a bit involved for a novice user — we propose and test an alternative to the three-part interface needed by the Algorithm 3.

---

[1]http://www.corel.com

Algorithm 5 active learning loop with online learning of relevant mixture components

---

Initialization

We set $n \leftarrow 0$, $L_{comp}^0 \leftarrow \emptyset$, $U_{comp}^0 \leftarrow \{1, \ldots, L\}$, $Mb^0 \leftarrow \{0, \ldots, 0\}$, $T \leftarrow T_0$.

While 1 **do** %active learning loop

  If $n = 0$ **then**

    $Disp \leftarrow \text{Prot}(Comp(1:K))$ %we select the prototypes of the first K mixture components for the display

    $L_{comp}^n \leftarrow Comp(1:K)$

    $L_{pts}^n \leftarrow Disp$

    $Feed \leftarrow \text{GetFeedback}(Disp)$

    $\text{Update}(Mb^n \, ; Feed)$

  Else

    $Set_1 \leftarrow \text{Sample}_{\text{Gibbs}}\left(U_{comp}^{n-1} \, ; d_{comp}^n(U_{comp}^{n-1}) \, ; K_1 \, ; T\right)$

    $Disp_1 \leftarrow \text{Prot}(Set_1)$ %components drawn according to a Gibbs distribution in the range $[T; +\infty]$

    $Set_2 \leftarrow Closest^n(D, K_2)$

    $Disp_2 \leftarrow \text{Prot}(Set_2)$ %most ambiguous components

    $Set_3 \leftarrow Closest^n\left(L_{comp}^n, K_3\right)$

    $Disp_3 \leftarrow PC^{n-1}(Set_3)$ %already marked components (convex hulls re-adjustment)

    $Disp \leftarrow Disp_1 \cup Disp_2 \cup Disp_3$

    $L_{comp}^n \leftarrow L_{comp}^{n-1} \cup Set_1$

    $U_{comp}^n \leftarrow Comp \setminus L_{comp}^n$

    $L_{pts}^n \leftarrow L_{pts}^{n-1} \cup Disp_1 \cup Disp_2 \cup Disp_3$

    %user feedback: we suppose here for clarity reasons that the user annotates all the elements of the sets $Disp_1, Disp_2, Disp_3$

    $Feed_1 \leftarrow \text{GetFeedback}(Disp1)$

    $Feed_2 \leftarrow \text{GetFeedback}(Disp2)$

    $Feed_3 \leftarrow \text{GetFeedback}(Disp3)$

    $\text{AdjustConvexHulls}(Set_3 \, ; Feed_3)$

    $\text{Update}(Mb^n \, ; Feed_1 \cup Feed_2)$

  End If

  $\left(f_n \, ; CP^n\right) \leftarrow \text{ComponentBasedFuzzySVM}(L_{comp}^n \, ; Mb^{n-1} \, ; L_{pts}^n)$

  For $i = 1 : \text{Card}(L_{comp}^n)$ **do** %Re-compute membership degrees of tagged components

    $Mb^n(i) \leftarrow \text{ComputeComponentMembership}(L_{comp}^n(i) \, ; L_{pts}^n \, ; f_n)$

  End For

  $n \leftarrow n + 1$

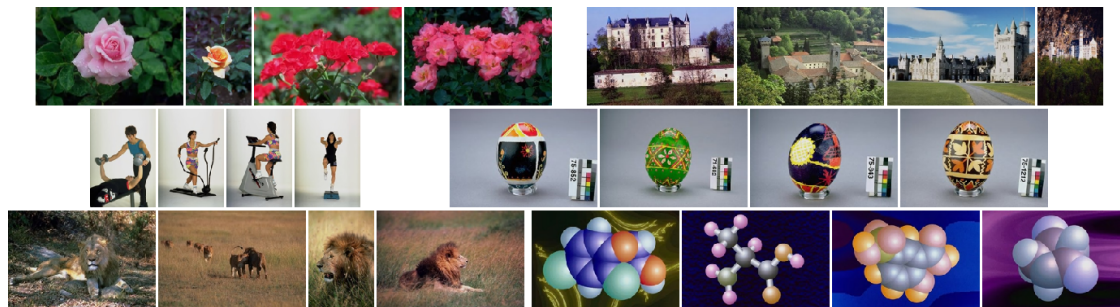 $T \leftarrow \alpha \cdot T$

End While

Figure 6.10: Example of images from six ground truth classes. From left to right in row-wise order: "flowers", "castles", "sportsmen", "painted eggs", "lions", "molecules".

6.4.1 Description of the test databases and of the image descriptors

Quickbird dataset: This dataset is built from 10 panchromatic scenes with a ground resolution of 61 cm representing overall views of Acapulco, Las Vegas, Los Angeles, London and Ouagadougou. The whole scenes are of approximate size 30000 by 30000 pixels. We extract descriptors within a sliding window of size 200 by 200. To build the database, we go through each scene by shifting the window in both directions, horizontal and vertical, with a step equal to half the window size each time such as to cover the whole scene. Thus, for a 30000×30000 scene, we obtain about 900000 feature vectors. The descriptors we use with this database are texture descriptors (statistical moments computed on gray-level co-occurrence matrices [Haralick et al., 1973] as well as statistical moments computed on several sub-band decompositions obtained using a bank of quadrature mirror filters), shape descriptors (Gradient Orientation Histograms) and gray-level histograms [Deselaers et al., 2008]. We perform a principal component analysis (PCA) on each descriptor separately to eliminate linear dependencies. We retain each time the first d principal axis where d is such that their cumulated inertia does not exceed 95%. We then concatenate the results to obtain the final feature vector. The data are then centered and normalized to unit variance.

Corel dataset: This is a well known database in image retrieval community. It consists of 60,000 natural images covering a broad range of human activities. We selected by hand a ground truth of about 5000 images organized in 46 different semantic classes, which ensures a certain diversity and renders difficult the task of retrieving a given category because of the presence of many other categories which can be very close or even overlapping with the targeted one at signal level. A sample from several ground truth classes is shown in Fig. 6.10. The descriptors here are the same as those used for the Quickbird dataset, with the exception that we added color histograms in RGB space, because they have been reported to improve the quality of results for many databases of natural images [Deselaers et al., 2008].

6.4.2 Verifying the enabling assumption and tuning the number of Gaussian components in the mixture model

As described in Sec. 6.2.1, we determine the intrinsic distribution of the data as a mixture of Gaussians by using an EM procedure seeded by an ELBG algorithm to initialize

a probabilistic clustering. The number of clusters in the ELBG algorithm is adjusted so as to ensure intra-cluster consistency, which, as we have seen before is one of the key assumptions on which rests the proposed method. We start by assessing how far the cluster assumption holds for the test datasets and we present an heuristic way of adjusting the number of Gaussian components of the mixture model.

6.4.2.1   Evaluating clusters consistency

Quickbird dataset   To perform the evaluation, we use a ground truth with six semantic classes (residential area, forest, road structures, parking lots, desert and agricultural fields), each class containing approximately 5000 image patches. The ground truth was built with the help of a CBIR system, and the content of each class was checked manually to eliminate misclassified patches. A sample of each class is shown in Fig. 6.11. Patches inside a class are coming from different scenes, which explains the observed variability.

Some classes like agricultural fields, forest or desert are visually very consistent whereas other like road structure or parking lots are much more mixed up. In fact, the latter classes will be the most difficult to learn since on a signal point of view, they strongly overlap with other classes. For instance, the class "road structures" is mixed with the class "residential area", which is also true on a mere visual point of view since on each side of a road we often observe residential areas. Some illustrations of what happens at signal level are given in Fig. 6.12. While reading this figure, keep in mind that the overlap between some classes at signal level in is due partially to the projection on the first two principal components. There is probably a subspace where these classes retain some separability otherwise it would be impossible to discriminate between them during the learning, which is not the case.

To assess the clusters consistency, we learn a Gaussian mixture model on the ground truth. The purpose here is to assess the validity of the "Gaussian mixture assumption" which in our case is closely related to the "cluster assumption" since clusters are obtained at each iteration of the active learning loop as the sets delineated by the current convex hulls of the mixture components. To be able to build quantitative measures we perform the tests on the ground truth classes. Of course, the number of mixture components is smaller compared to the mixture model on the whole database, but the classes in the ground truth are representative for the type of images a user is likely to search.

We study both the influence of the number of components in the mixture model and the influence of the tightness of the convex hulls. We start by representing the mean purity and the mean entropy of the clusters as a function of the number of components (see Fig. 6.13(a) and 6.13(b)) for fixed convex hulls. For simplicity reasons, we use c_l to denote both the cluster associated with the l-th mixture component and the mixture component itself. The mean purity is defined as $Purity = \sum_{l=1}^{L} \frac{n_l}{N_{tot}} p_l$ with $p_l = \max_c p_{cl}$, $n_l = \text{Card}(c_l)$, $N_{tot} = \sum_{l=1}^{L} n_l$ and $p_{cl} = \frac{n_c^l}{n_l}$ where n_c^l is the number of elements inside the cluster c_l belonging to the class c ($c \in$ {residential area, forest, road structures, parking lots, desert, agricultural fields }). The mean entropy is defined as $Entropy = -\sum_{l=1}^{L} \sum_c \frac{n_l}{N_{tot}} p_{cl} \log_2 p_{cl}$, i.e, $n_c^l = \text{Card}(\{v; v \in c_l \text{ and } v \in c\})$. The analytical expressions of n_l and n_c^l are given in the next paragraph. Both measures, purity
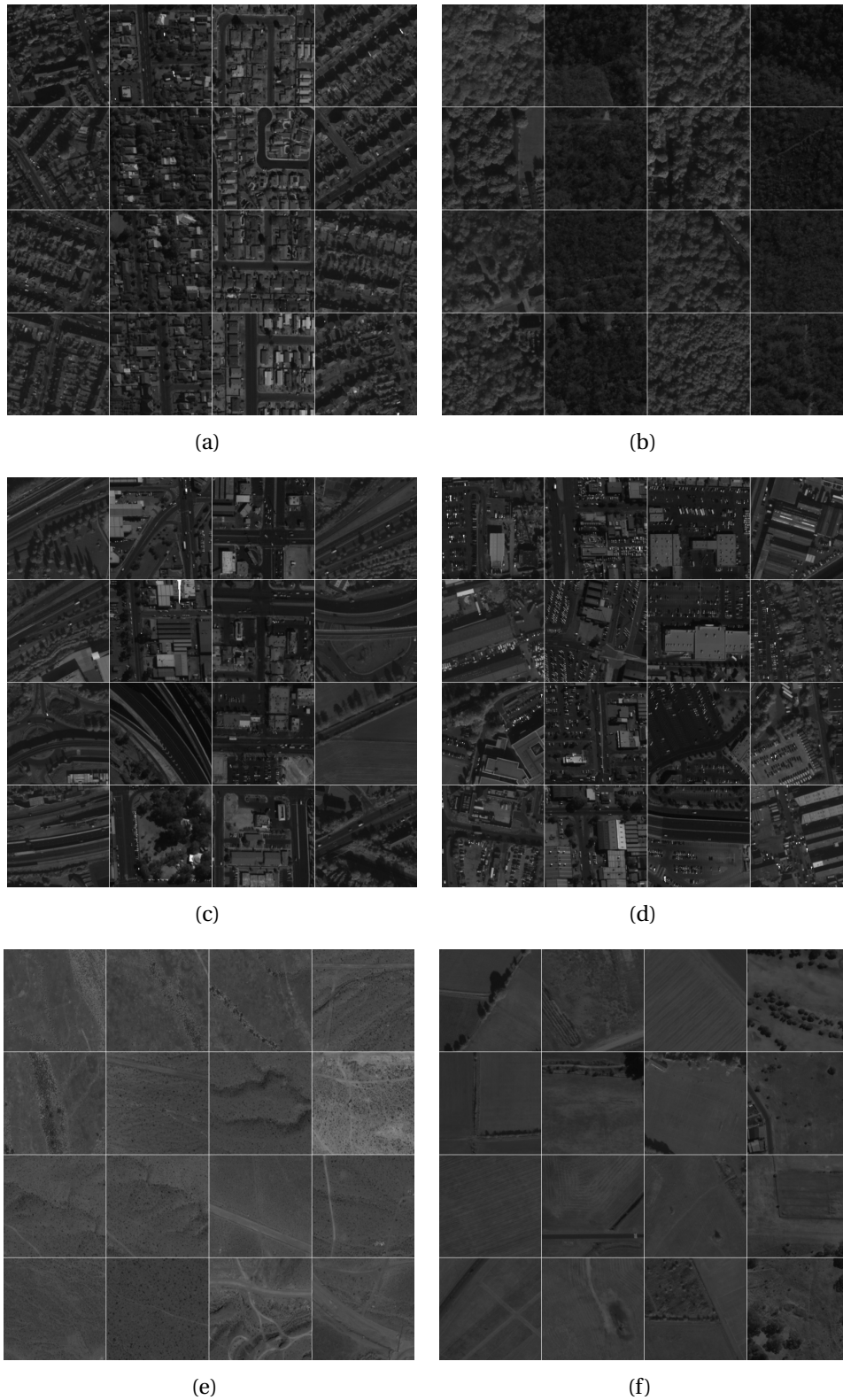
Figure 6.11: Samples taken from each of the six ground truth classes: **(a)** residential area; **(b)** forest; **(c)** road structures; **(d)** parking lots; **(e)** desert; **(f)** agricultural fields. Images in a column come all from the same scene whereas images in a row come each from a different scene.
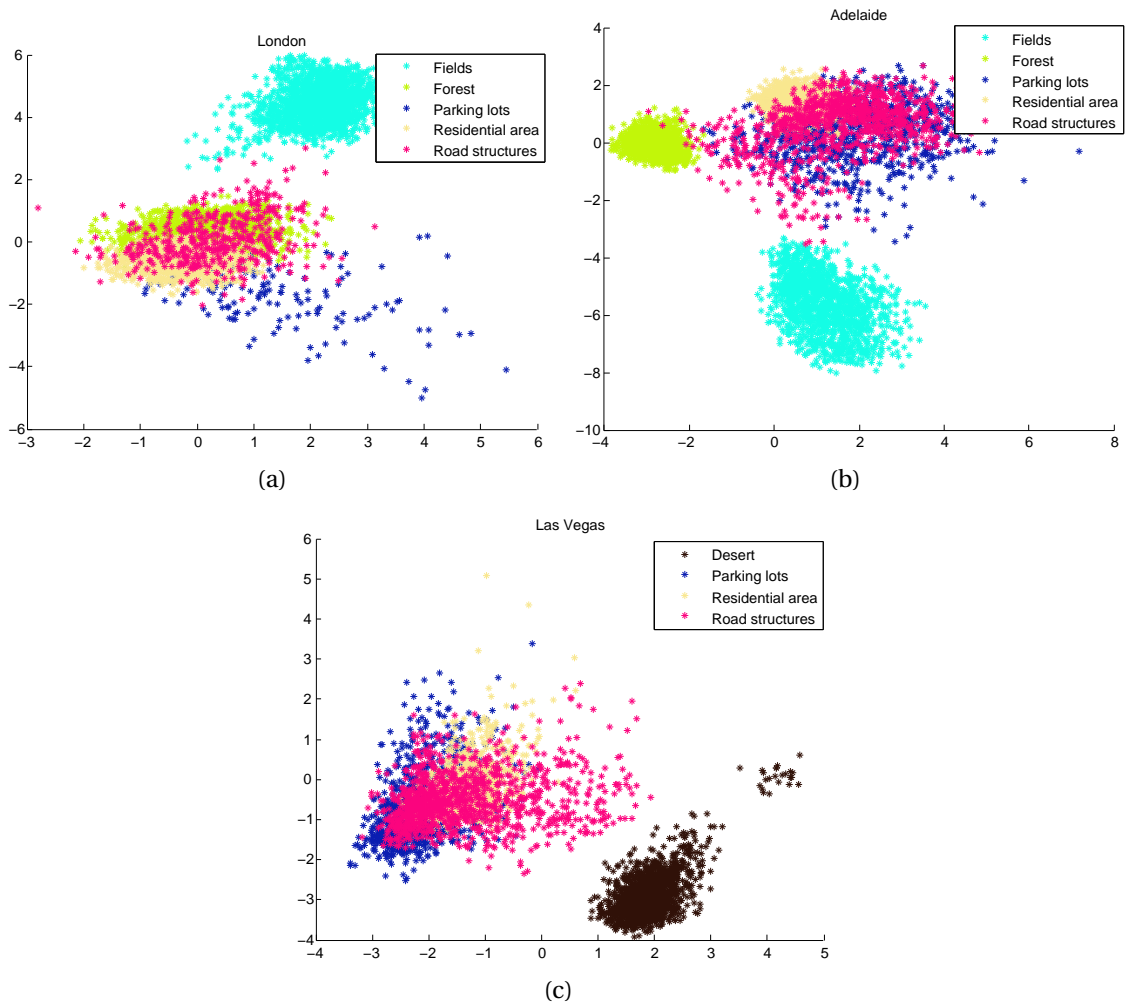
Figure 6.12: Representation of the descriptors in the system of axes formed by the first two principal components for three different scenes: **(a)** London; **(b)** Adelaide; **(c)** Las Vegas. Visually consistent classes can be easily distinguished one from another: "forest" in cyan, "desert" in black and "fields" in green. On the contrary, less visually consistent classes strongly overlap with each other. For instance, "road structures" in garnet red is mixed with "residential area" in yellow.
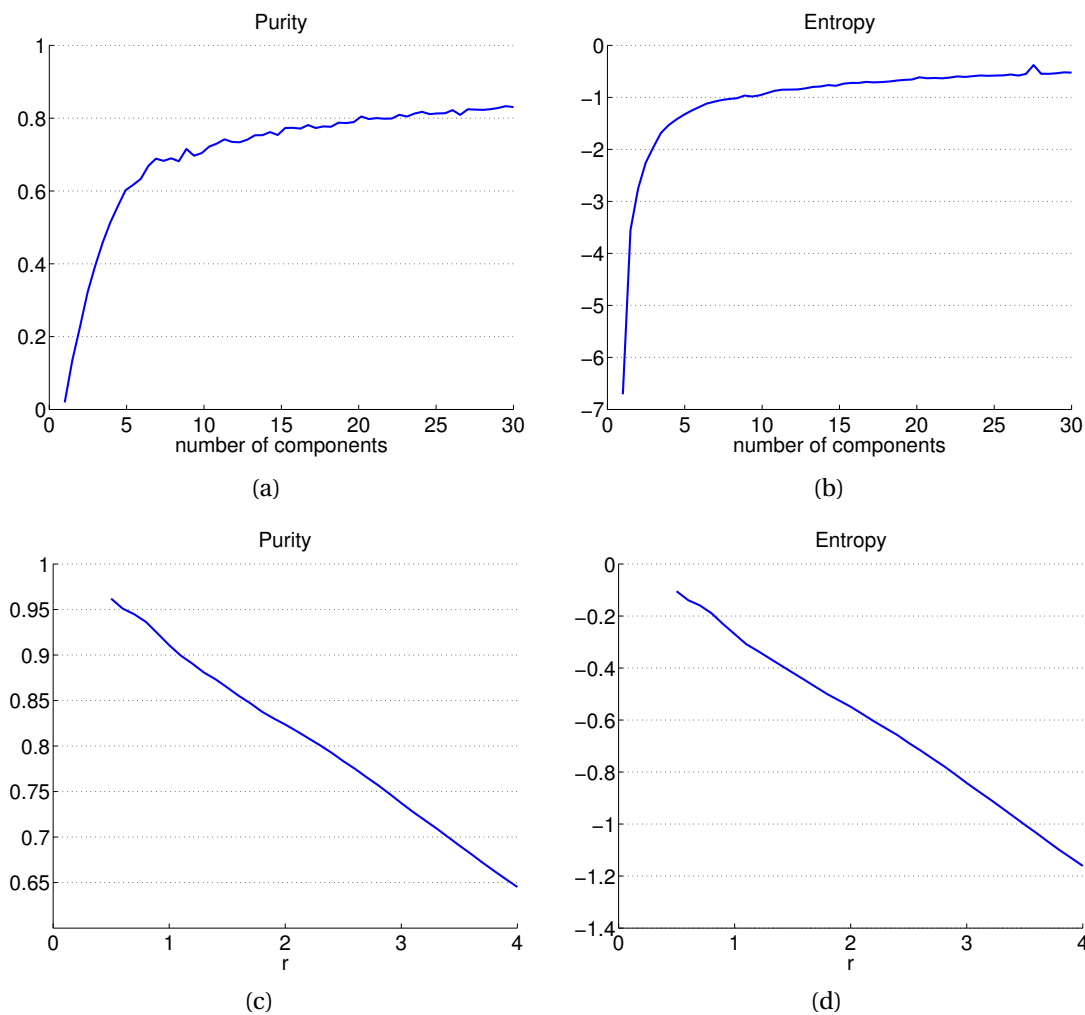
Figure 6.13: **QuickBird dataset: (a)** Clusters mean purity and **(b)** mean entropy as a function of the number of components (*r* is fixed to 1.5). **(c)** Clusters mean purity and **(d)** mean entropy as a function of the "tightness" of the convex hulls (the number of components is fixed to 20).
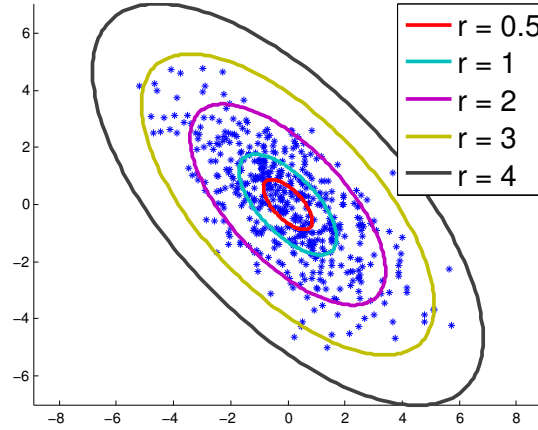
Figure 6.14:   Convex hull of a mixture component as a function of the number r of "sigmas" we keep in each principal direction.

and entropy, represent the extent to which a cluster consists of objects belonging to a single class. As we increase their number, clusters become more and more coherent, that is the purity increases (tends towards one) and the entropy decreases (tends towards zero).

In the Fig. 6.13(c) and Fig. 6.13(d), we have represented the mean purity and the mean entropy of the clusters as a function of the size of the convex hulls for a fixed number of components. That is, for a given component c_l whose covariance matrix is given by $\Sigma_l = {}^t P_l D_l P_l$, we make the values of ρ_1^l vary in the interval $\left[\mathcal{N}(\gamma_l\left(\frac{1}{2}\right); \mu_l, \Sigma_l) ; \mathcal{N}(\gamma_l(3); \mu_l, \Sigma_l) \right]$ with $\gamma_l(r) = \mu_l + r \times \sqrt{D_l(1,1)} P_l(:,1)$. The matrix D_l is the diagonal matrix obtained by diagonalizing the symmetric positive definite matrix Σ_l in an orthogonal basis of eigen vectors formed by the lines of the matrix P_l. The notation $P_l(:,1)$ is used to designate the first column of P_l. A simple calculation shows that: $\mathcal{N}(\gamma_l(r); \mu_l, \Sigma_l) = \frac{\exp(-\frac{r^2}{2})}{(2\pi)^{d/2} \det(\Sigma_l)^{1/2}}$. Because the probabilities ρ_1^l do not reflect directly the notion of tightness, we introduce the notation r to stand for the "number of sigmas" we retain in each principal direction to define the equiprobable envelopes (see Fig. 6.14). Using the definition of clusters and for a given r, we obtain:

$$n_l = \mathrm{Card}\left(\{v; \mathcal{N}(v; \mu_l, \Sigma_l) < \mathcal{N}(\gamma_l(r); \mu_l, \Sigma_l)\}\right)$$

and

$$n_c^l = \mathrm{Card}\left(\{v; \mathcal{N}(v; \mu_l, \Sigma_l) < \mathcal{N}(\gamma_l(r); \mu_l, \Sigma_l) \text{ and } v \in c\}\right)$$

The x-axis of the Fig. 6.13(c) and Fig. 6.13(d) represents the parameter r which varies between $\frac{1}{2}$ and 4. We see thus that low values of r ("tight" convex hulls) cause clusters to be more coherent.

Corel dataset   We repeat the same experiments for the Corel dataset. To be consistent with what precedes, we only evaluate these two quantities on the same number of classes (the six classes presented in Fig. 6.10), which we also use to assess the retrieval performance of the system. The results are shown in Fig. 6.15.
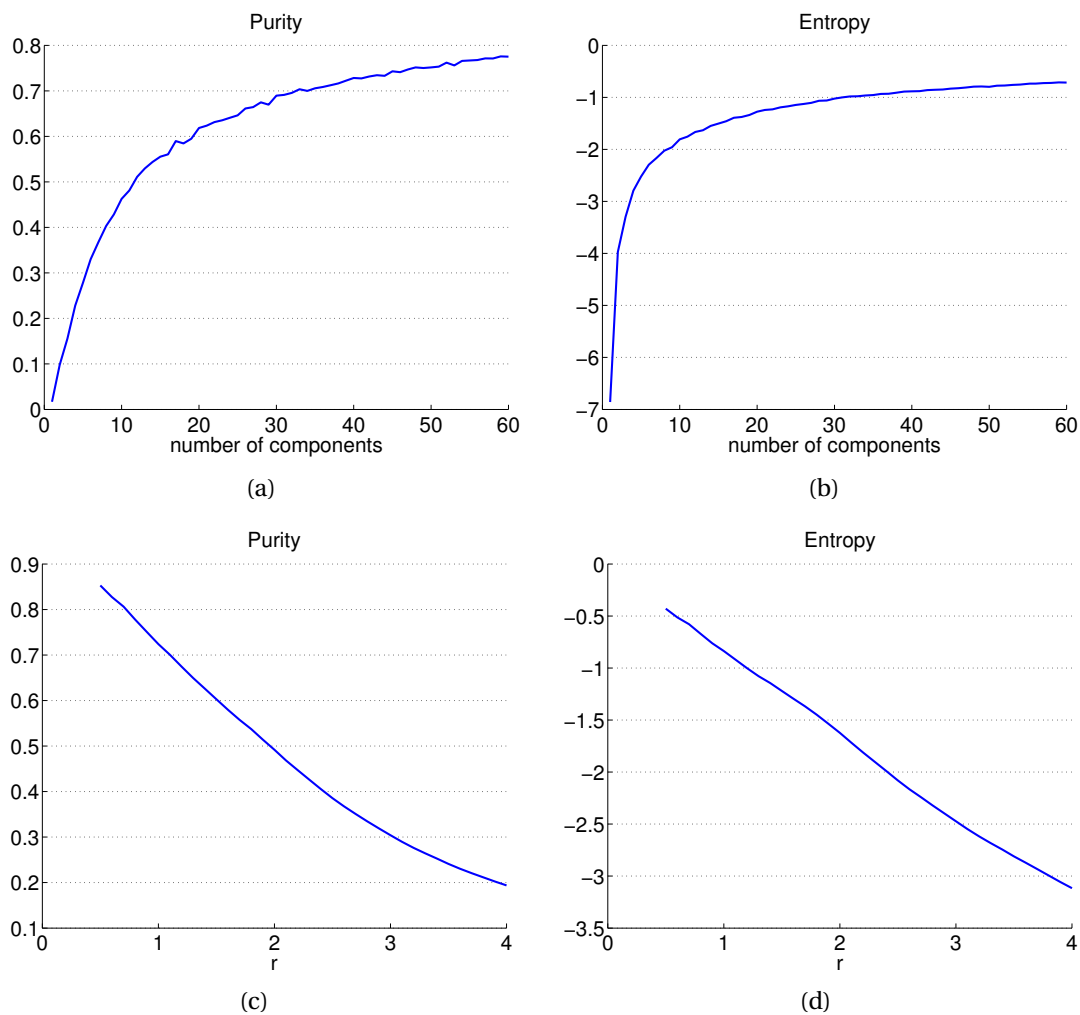
Figure 6.15: **Corel dataset: (a)** Clusters mean purity and **(b)** mean entropy as a function of the number of components (r is fixed to 1.5). **(c)** Clusters mean purity and **(d)** mean entropy as a function of the "tightness" of the convex hulls (the number of components is fixed to 20).

For both datasets, we see that the assumption of intra-cluster coherence is well justified provided we choose a high enough number of components in the mixture model. For the Quickbird dataset, the value of purity is situated around 0.75 for a mixture model consisting of only 12 components and a value of r equal to 1.5. In comparison, for the partial Corel dataset, we attain the same "level of purity" by setting to 60 the number of components in the mixture model. Thus, to ensure that the enabling assumption is verified, we have to tune the number of mixture components according to the database. Generally speaking, a small number of mixture components will ensure a small number of iterations in the first part of the active learning loop (the one dedicated to the database exploration and the adjustment of convex hulls). But the obtained SVM model will be much less precise since the clusters are not consistent enough. Thus, we will have, in a second part of the active learning loop, to refine the SVM model using a classical active learning scheme working at point level such as our baseline. On the contrary, a large number of mixture components will ensure that the obtained clusters are consistent. Thus, the first part of the active learning loop will yield a very precise SVM model of the targetted category but at the expense of many iterations in this part of the active learning loop. On the other hand, we might not need to add a model refinement strategy in a second part, as it would be necessary in the case of a small number of mixture components. The better is to find a compromise by choosing a number of components which is not to small, to ensure that the result of the first part of the active learning loop will be correct enough to minimize the number of iterations in an eventual refinement step in a second part. And at the same time, the number of components must not be too high, to arrive at a quick definition of the target class in the first part. We still have to keep in mind that there is a compensation mechanism for the lack of coherence of clusters which consists in re-adjusting the convex hulls of the corresponding mixture components at each iteration of the active learning loop. For both datasets, we have chosen to set to 0.7 the initial level of purity of clusters, which, according to our tests, leads to a good compromise in the above-mentioned acceptation: for the QuickBird dataset, the average precision of the obtained classifier at the end of the first part of the active learning loop is ≈ 0.86 (for an average recall of ≈ 0.75) and 20 iterations are necessary to arrive at such a level of precision (see Fig. 6.18). We then need only 10 iterations in the second part of the active learning loop to obtain a "saturated" classifier, that is, a classifier which do not improve on precision and recall anymore even when performing "extra iterations". The same considerations apply for the Corel dataset. In the following paragraph, we compute bidimensional models representing the purity as a function of both the number of components in the mixture model and the value of clusters "tightness" r. We then determine the isolines for a purity value of 0.7 and we look for the best compromise between r and the number of components.

6.4.2.2   Tuning the number of components in the Gaussian mixture model

To tune the number of components in the Gaussian mixture model, we repeat the last experiments, using this time all the feature vectors even those not present in the ground truth. In the case of the Corel dataset, we "simulate" a partial ground truth by considering that we only know the ground truth for the six classes we want to learn (it is indeed not a realistic setting to consider that we possess a complete ground truth

over the whole database). We still use a majority voting rule to determine the class of each mixture component and we discard the components which do not contain any element from the ground truth. The measures of purity and entropy we obtain are, of course, only approximations since we evaluate them only on the ground truth classes and on the images of these classes which effectively belong to the ground truth (there might be images which can be classified according to a ground truth class but which are not in the ground truth). To compute the mean purity and the mean entropy of clusters, we use slightly different formulas:

$$\text{Purity} = \sum_{l=1}^{L} \frac{n_l^{'}}{N_{tot}^{'}} p_l^{'} \text{ and Entropy} = -\sum_{l=1}^{L} \sum_{c} \frac{n_l^{'}}{N_{tot}^{'}} p_{cl}^{'} \log_2 p_{cl}^{'}$$

In the above formulas, $n_l^{'}$ refers to the number of elements from the ground truth belonging to the convex envelope of the mixture component c_l i.e.:

$$n_l^{'} = \text{Card}(\{v; \mathcal{N}(v; \mu_l, \Sigma_l) < \mathcal{N}(\gamma_l(r); \mu_l, \Sigma_l\} \cap \{\text{ground truth}\})$$

$N_{tot}^{'}$ refers to the total number of elements in the ground truth inside the components convex hulls i.e. $N_{tot}^{'} = \sum_{l=1}^{L} n_l^{'}$. We then have $p_l^{'} = \max_c p_{cl}^{'}$ and $p_{cl}^{'} = \frac{ngt_c^l}{n_l^{'}}$ where ngt_c^l is the number of elements from the ground truth inside the convex envelope of the mixture component c_l belonging to the class c.

We observe that the isolines of the Fig. 6.16(b) and Fig. 6.17(b) can roughly be decomposed into two straight lines with different slopes. In the first part of the isolines, the slope is quite small, meaning that an important increase in r will result in a relatively small increase in the number of components necessary to ensure the same level of purity. On the contrary, in the second part, a small increase in r will cause a relatively large increase in the number of components to keep the purity constant. We have chosen to place ourselves at the junction of the two parts, since it provides a natural compromise between r and the number of components, which is what we are looking since we want to retain as much information as possible from each component and at the same time to minimize the number of components (keeping the purity constant). For a purity level of 0.7, we obtain according to the above considerations: $r = 2$ and $L = 200$ in the case of the QuickBird dataset and $r = 2$ and $L = 300$ in the case of the Corel dataset. The differences in the found number of components between the two datasets reflect the fact that semantic consistency at signal level, i.e. semantic consistency inside a cluster is much more difficult to obtain on photographic images than on remote sensing images. Similar results are obtained considering the entropy instead of the purity as a measure of clusters consistency.

6.4.3   Evaluation of the system performance

In this section, we present a method of evaluation of our active learning scheme. We perform six distinct experiments on each of the six classes of the ground truth. Each time we compute the average precision and the average recall at each step of the active learning loop. We compare ourselves to the art active learning method proposed in [Ferecatu and Boujemaa, 2007], which has been reported to yield good results both for

(a)

(b)

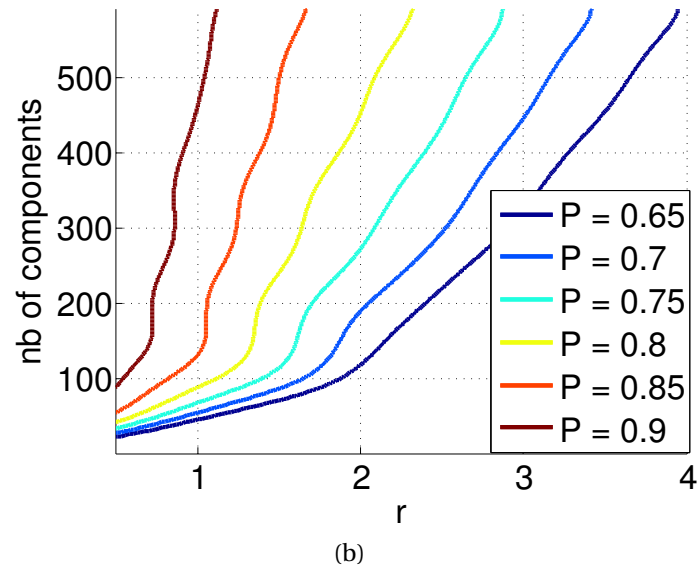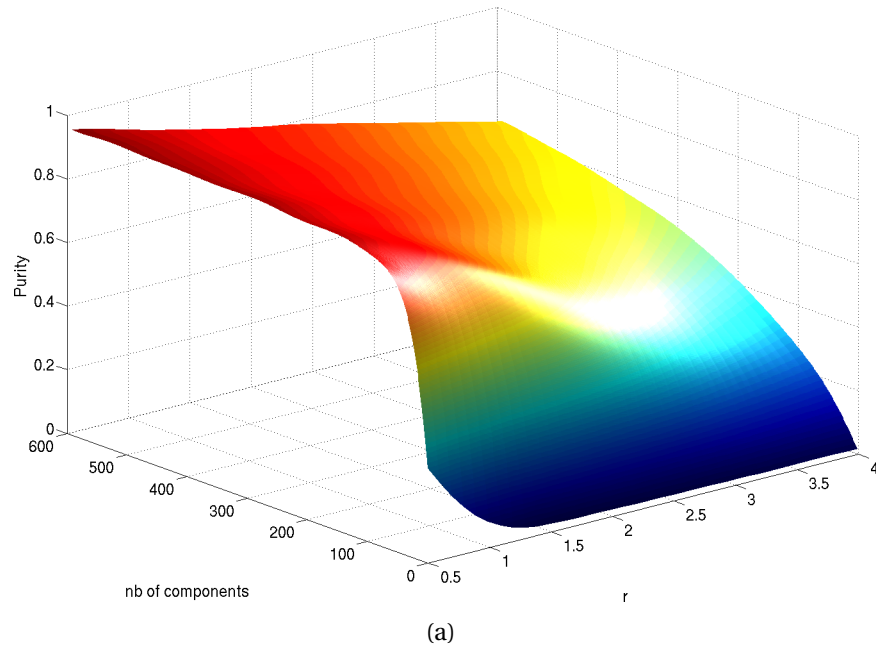Figure 6.16: **QuickBird dataset. Left:** clusters mean purity as a function of the number of components and the "tightness" of convex hulls. The whole dataset is used to estimate the Gaussian mixture parameters but the purity is assessed only on the images belonging to the ground truth. **Right:** Isolines for different purity values.
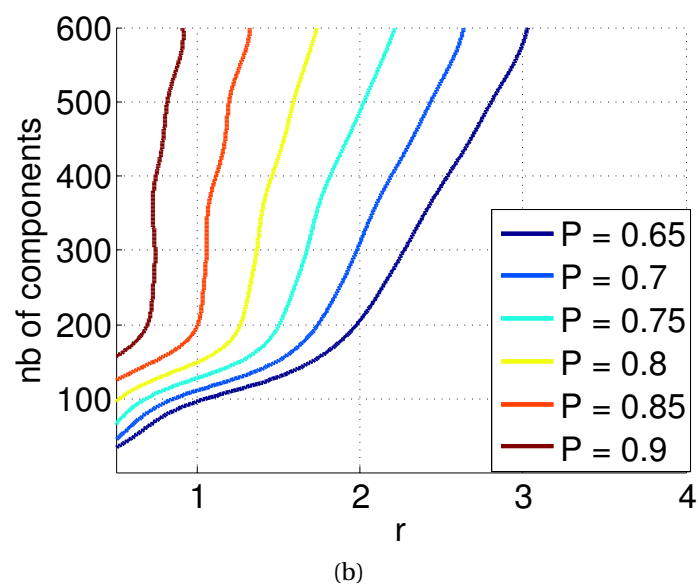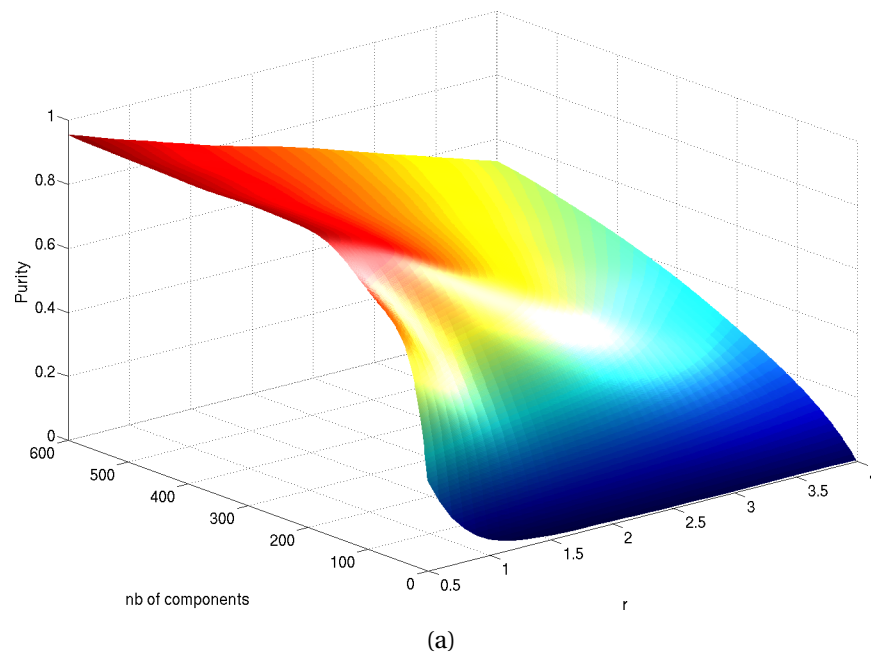
(a)

(b)

Figure 6.17: **Corel dataset.** See caption of figure 6.16.

satellite images and for generic images [Ferecatu et al., 2008] (called in the following "the baseline method"). We consider here only the *online learning* of relevant clusters. The *batch learning* procedure, presented in Sec. 6.3.1, is only used as an intermediary step to facilitate the presentation of the whole online procedure. The batch learning of relevant components does not correspond indeed to a realistic scenario since we cannot reasonably ask the user to exhaustively mark all the components he thinks relevant among hundreds of other components within one single iteration of the learning process.

A key aspect in our evaluation method is the initialization of the two methods (baseline and ours). To start a feedback loop, the baseline method needs both positive and negative examples from the target class, while our method needs a few positive and negative prototypes of components relevant to the target class. We chose to initialize both methods with two positive and two negative component prototypes. The first iteration of the active learning loop in the figures below corresponds thus to the first iteration after this initialization step. To give the two methods equal chances, we use displays of equal sizes for both systems. That is, at each iteration of the active learning loop, the user is asked to tag exactly the same number of images in both cases. The precision is computed as the number of relevant images over the total number of retrieved images (we consider of course only images from the ground truth). The recall is computed as the number of relevant images over the total number of retrieved images from the target class.

$$\text{Precision} = \frac{\text{Card}\left(\{\text{relevant retrieved images}\} \cap \{\text{ground truth}\}\right)}{\text{Card}\left(\{\text{retrieved images}\} \cap \{\text{ground truth}\}\right)}$$
$$\text{Recall} = \frac{\text{Card}\left(\{\text{relevant images}\} \cap \{\text{ground truth}\}\right)}{\text{Card}\left(\{\text{retrieved images}\} \cap \{\text{ground truth}\}\right)} \tag{6.7}$$

We make a comparison of the two methods over 30 iterations of the active learning process. For all the ground truth classes, we restrict to 20 the number of active learning iterations within the Algorithm 3, because we observed experimentally that the accuracy did not improve any more after that. Indeed, because the Algorithm 3 works at the cluster level, there is a saturation of the learning process induced by the stabilization of the convex hulls of the clusters and also by the fact that all the relevant clusters have been retrieved by the online learning process. At this point, further progress can be made by focusing the learning process at a lower granularity, i.e. at the level of individual points. Thus, from iteration 20 we switch our active learning process with that of the baseline, which is better adapted to work with individual points. Practically, depending on the target class this produces an improvement in precision of 5–10% during iterations 20 to 30 in our experiments (see Fig. 6.18(a–f) and Fig. 6.19(a–f)). It has almost no effect on the recall.

In the Fig. 6.18 and Fig. 6.19, we have represented for the two datasets the precision, the recall and the number of support vectors of the current SVM classifier as functions of the number of iterations in the active learning process. At each iteration, the values of precision, recall and the number of support vectors are averaged over 30 retrieval sessions.

We use a Gaussian kernel for both methods. In the first five cases (desert, fields, forest, parking lots, residential area), our method converges much faster towards a sta-
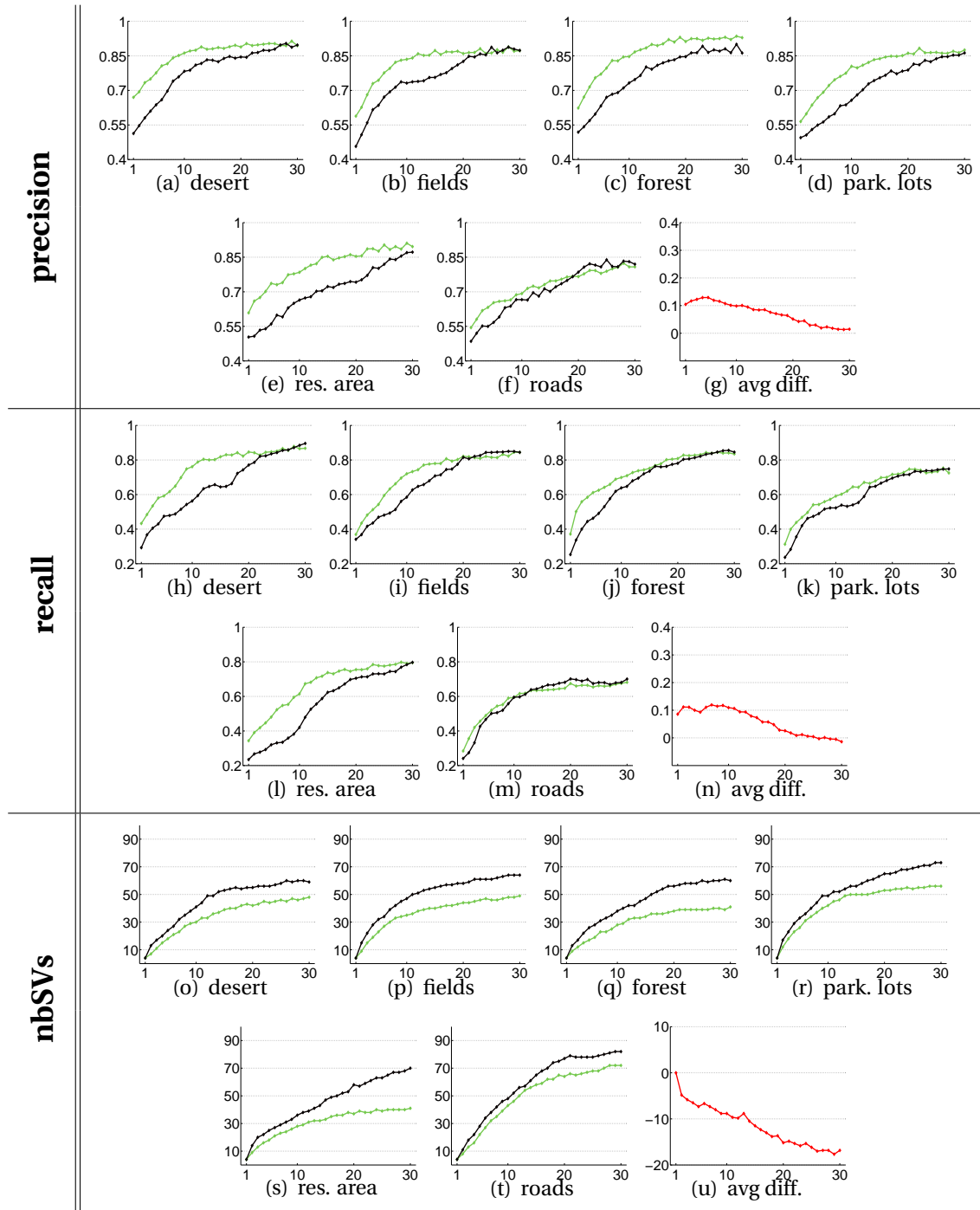
Figure 6.18: **QuickBird dataset:** comparison in terms of **precision** (Fig. (a), (b), (c), (d), (e), (f), (g)), **recall** (Fig. (h), (i), (j), (k), (l), (m), (n)) and **number of support vectors** (Fig. (o), (p), (q), (r), (s), (t), (u)) of our method (green curve) with the baseline (black curve) for six semantic classes. The x axis represents the number of iterations in the active learning process. The graphics with the red curves represents the average difference between the two methods over the six classes.
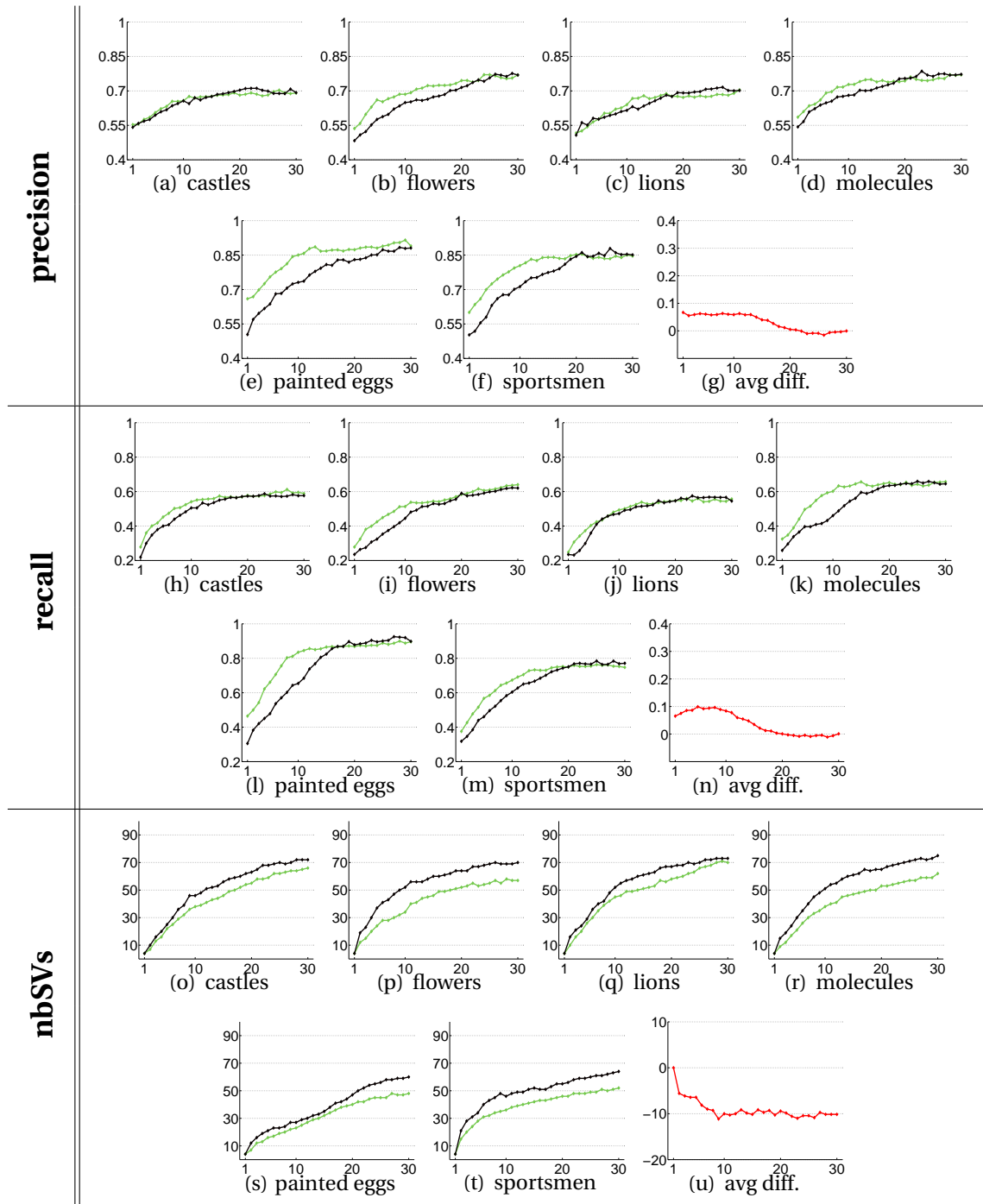
Figure 6.19: **Corel dataset:** cf. legend of Figure 6.18

ble classifier, which does not improve the accuracy any more even by increasing the number of active learning iterations. The result is less pronounced for the class "road structures" (Fig. 6.18(f), 6.18(m) and 6.18(t)). This class possesses indeed a much more complex representation at feature level and doesn't easily lend itself to a standard clustering, unlike the two previous classes for which the clusters obtained are very consistent. Our method being highly dependent on the intra-cluster consistency ("cluster assumption"), this explains why it does not work as well as in the other cases.

Similar tests are performed on the Corel dataset (see Fig. 6.19). The results are satisfactory for the classes "sportsmen", "painted eggs", "flowers" and "molecules", while the gain of our method is less clear for the classes "castles" and "lions". This comes from the fact that these two classes are not well modeled by the Gaussian mixture model and their elements are scattered over several mixture components and are mixed up with elements from other classes.

6.4.4   Evaluation of the unlearning behavior

In this section, we evaluate the ability of the system to unlearn the mixture components which have been misclassified as relevant by the user at the beginning of the learning process. We perform tests on both our QuickBird image database and the Corel dataset. For the first database, we use the class "urban area" as the target concept and for the second database, the class "lions". We introduce in both cases three errors in the initial annotations of components, that is we start the learning process with three false positives (component wrongly annotated as relevant). In the graphics of the Fig. 6.20, we have represented the membership degrees of the three misclassified components (Fig. 6.20(a) and 6.20(b)) and the membership degrees of three arbitrarily chosen relevant components (Fig. 6.20(c) and 6.20(d)) as a function of the number of active learning iterations. We observe that as the number of iterations augments, the memberships of misclassified components decrease and the memberships of relevant components remain almost constant, which is the expected behavior. We observe that at least 20 iterations are needed in the active learning process to completely forget a component, but, as illustrated by the 2D example in Fig. 6.21, it is not mandatory for a component to have a quasi null membership degree to be forgotten. In fact, after only ten iterations a component wrongly annotated as relevant by the user has a much smaller impact, even if its membership degree is still high. This behavior is clearly seen in our 2D example: even with a membership degree of 0.4, the major part of the red cluster is situated outside the zone delineated by the SVM surface. The inconvenient is that components wrongly annotated as relevant at the end of the learning process might not be forgotten due to the low reactivity of the system. A solution to this problem could be to emphasize the decrease of component membership degrees, but it might wrongly affect the membership degrees of relevant components as well and cause relevant components to be forgotten during the learning process, which is, of course, an undesirable effect.

(a) Quickbird

(b) Corel

(c) Quickbird

(d) Corel
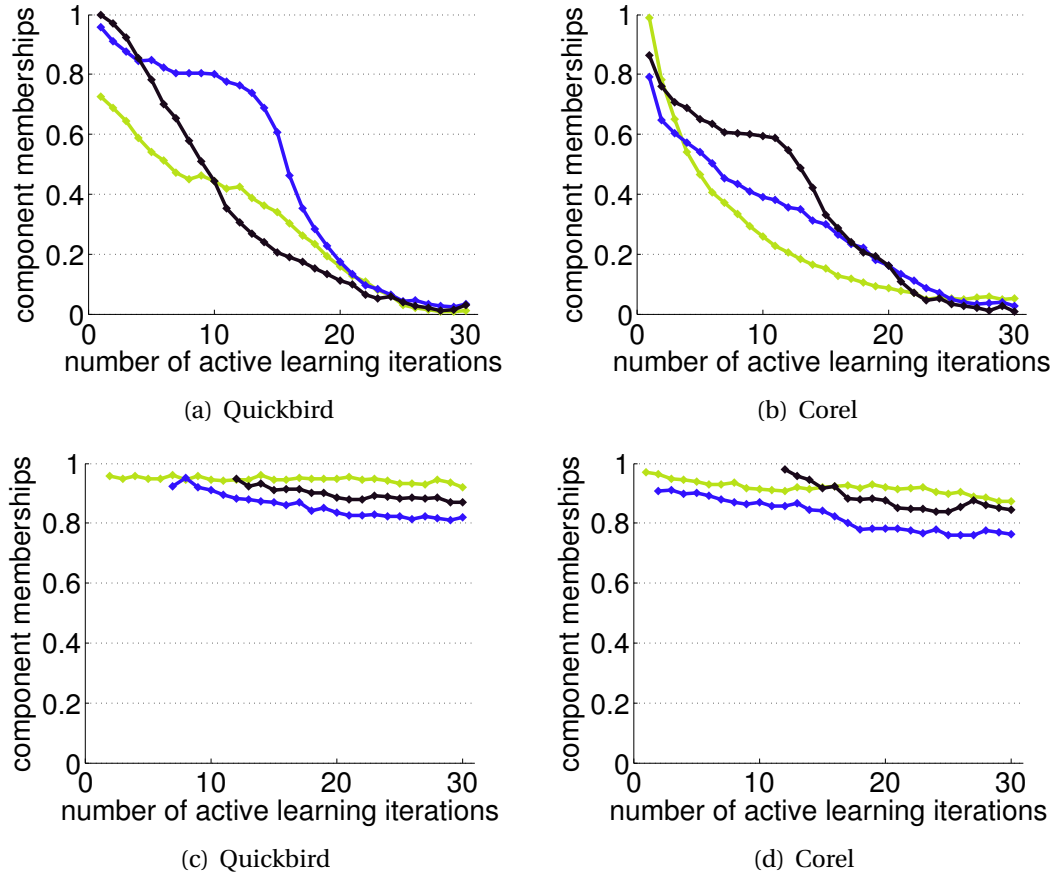
Figure 6.20: Membership degrees of three misclassified components ((**a**) and (**b**)) and membership degrees of three arbitrarily chosen relevant components ((**c**) and (**d**)) as a function of the number of active learning iterations. In (**a**) and (**b**), the results are averaged over ten runs of the whole learning process whereas in (**c**) and (**d**), the results are computed over just one run. We cannot ensure in fact that relevant components are added at the same iteration of the active learning loop each time so it just makes no point here to average the results over several runs of the learning process.

(a) m = 1   (b) m = 0.8   (c) m = 0.6

(d) m = 0.4   (e) m = 0.2   (f) m = 0.05
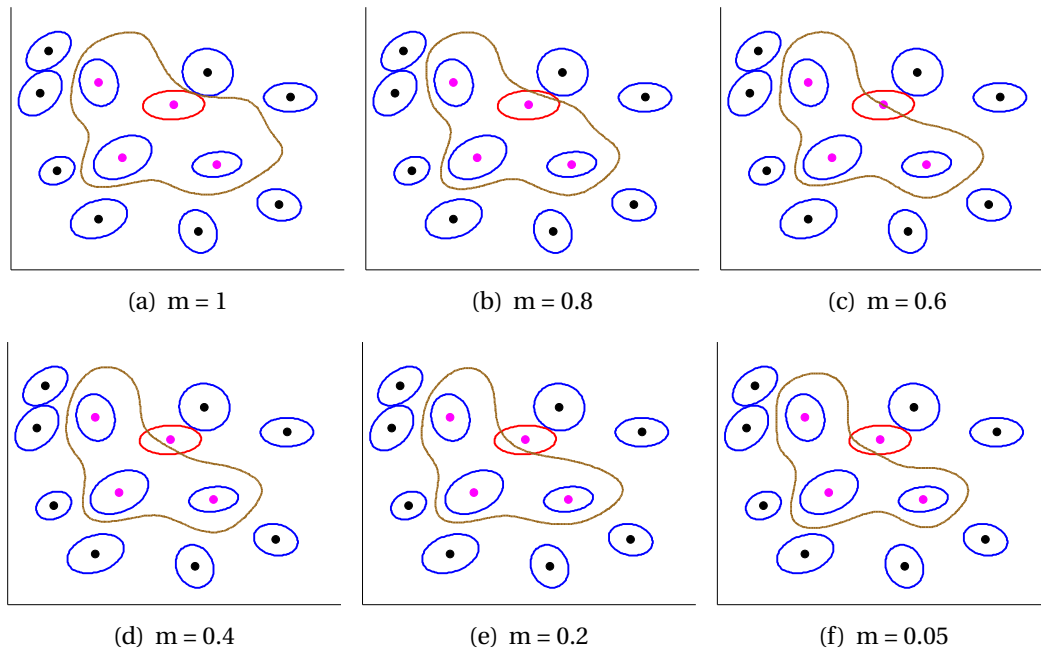
Figure 6.21: Example of application of our component-based fuzzy SVM on a 2D example. The relevant components are those marked with a magenta dot in the center. The others are considered to be irrelevant components. We vary the membership degree of the mixture component corresponding to the red cluster in the associative model and we train our component-based fuzzy SVM each time. As the membership degree decreases, the part of the cluster contained in the zone delineated by the SVM surface also decreases.
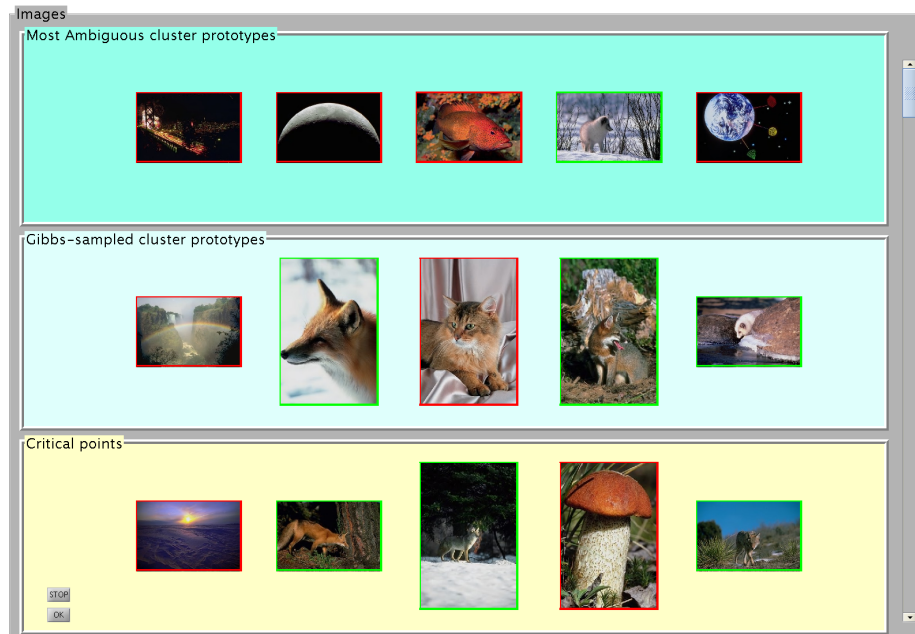
Figure 6.22:   Screenshot of the graphical interface of the system. The upper box contains the prototypes of the most ambiguous components, the middle box the prototypes of components sampled according to a Gibbs distribution and the bottom box the critical points associated with the already marked components

6.4.5   Evaluation of the proposed strategy for the choice of new components: comparison with a "Most Ambiguous" strategy

In this section, we compare our strategy which consists in sampling new components according to a Gibbs distribution with the standard Most Ambiguous (MA) strategy described in [Ferecatu and Boujemaa, 2007]. This selection strategy chooses as the next batch of images to be sent to the user for feedback, the elements of the database that are closest to the SVM decision function. The idea is to maximize the transfer of information between the system and the user by avoiding regions of the description space where the current SVM classifier is confident enough. We also evaluate the effect of maintaining a MA component in the display throughout the learning process. Fig. 6.23 presents the three scenarios. We see that the "mixed strategy" which consists in displaying to the user prototypes of components corresponding to both MA and Gibbs-sampled components performs slightly better than the two others: there is almost no increase in precision but the recall is better. For both datasets, we present the gain in precision and recall over the baseline method averaged over the six classes.

The curves corresponding to the mixed strategy are the same as the average precision and recall curves in the figures 6.18 and 6.19. The strategy for the choice of components used in Sec. 6.4.3 is indeed the mixed strategy described above and the graphical interface is the three-part interface shown in Fig. 6.22.

We have also made some tests regarding the appearance of the graphical interface presented to the user. Our standard interface, which consists of three distinct parts (one for the most ambiguous component prototypes, one for the Gibbs-sampled com-

(a) precision (Quickbird)

(b) recall (Quickbird)

(c) precision (Corel)

(d) recall (Corel)
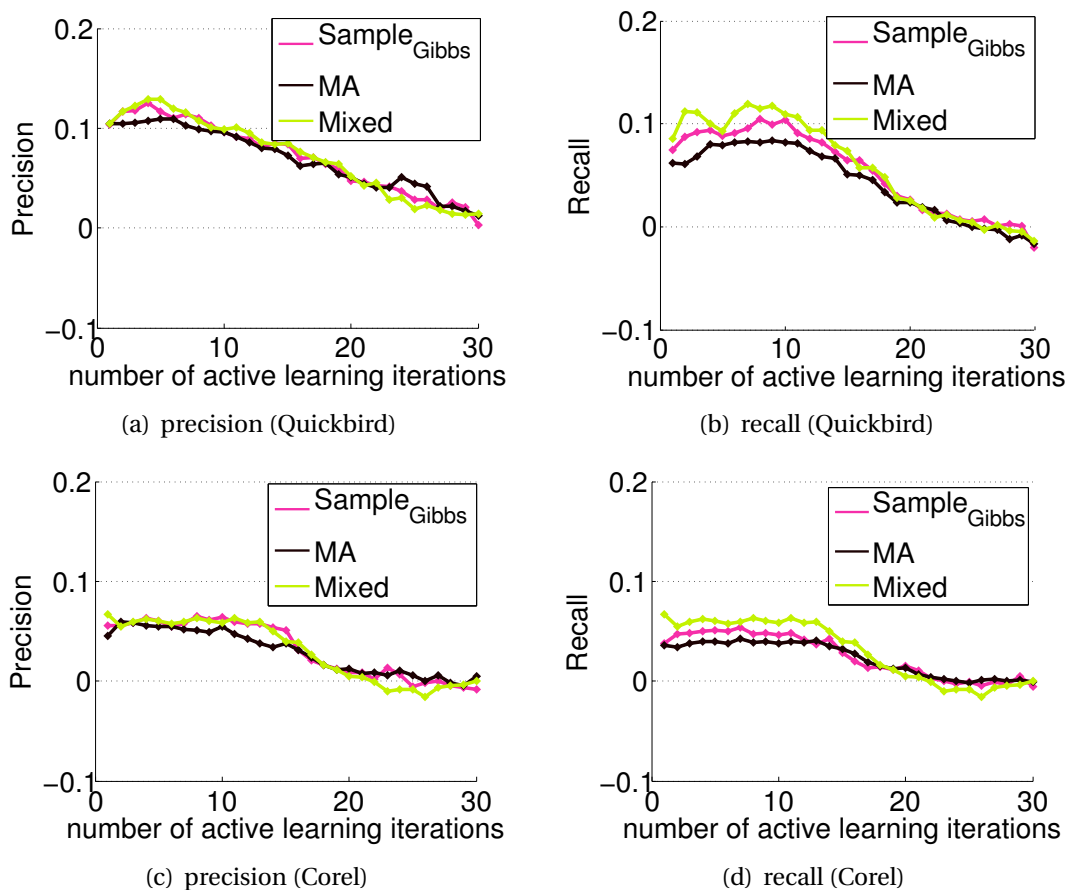
Figure 6.23: Comparison of the three strategies for the choice of new components. The magenta curve refers to the strategy which consists in sampling new components according to a Gibbs distribution, the black curve refers to the most ambiguous strategy and the green curve to our "mixed strategy". The curves represent the average difference in terms of precision and recall on the six classes of each dataset between our method and the baseline used for comparison, a positive difference meaning that our method performs better than the baseline.

(a) precision (Quickbird)

(b) recall (Quickbird)

(c) precision (Corel)

(d) recall (Corel)
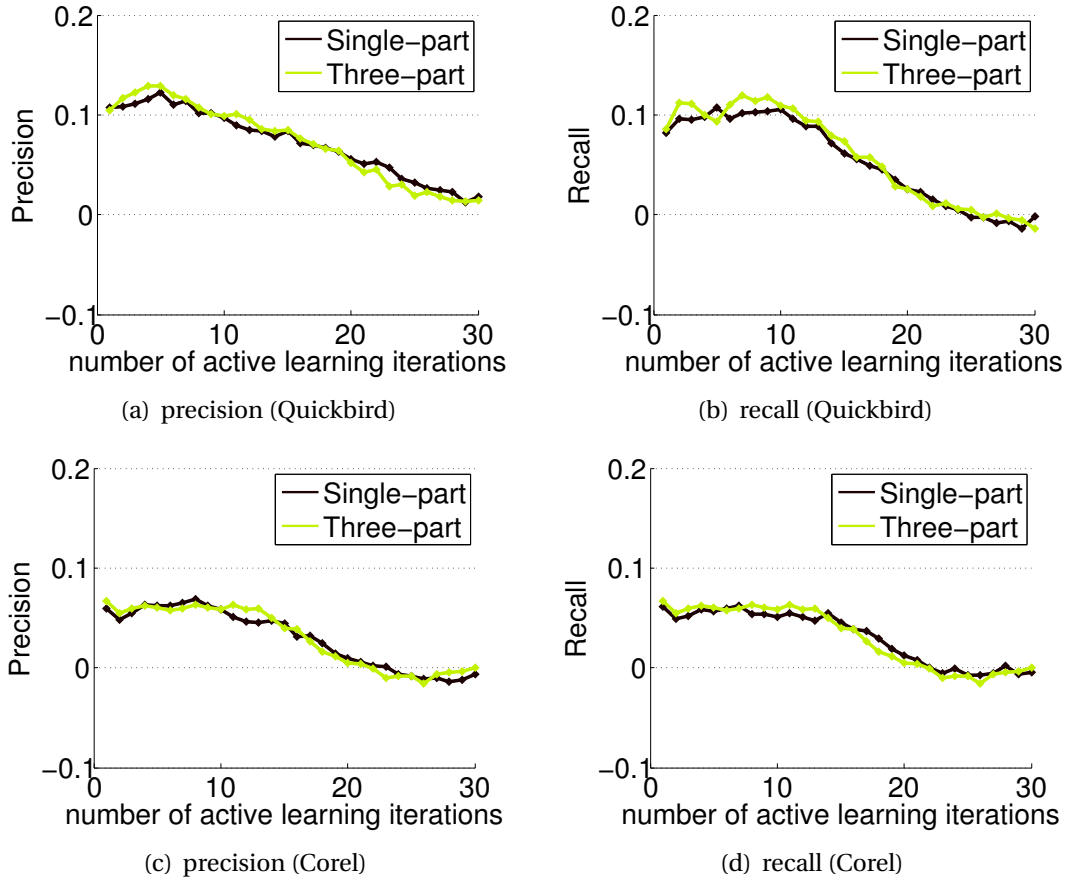
Figure 6.24:   Impact on the system performance of changing the user interface. The black curve refers to the single-part interface and the green curve to three-part interface. The curves represent the average difference in terms of precision and recall on the six classes of each dataset between our method and the baseline used for comparison, a positive difference meaning that our method performs better than the baseline.

ponent prototypes and one for the critical points), is indeed a bit complicated for users who are not familiar with the basic functioning of the system and have no idea of the underlying algorithmic part. Consequently, we have tested a simplified version of the system using a single-part graphical interface. In this simplified version, the images which were spread between the three parts in the preceding case are simply mixed up randomly and presented to the user on a single display. The results are still quite satisfying compared to those obtained with the three-part interface (see Fig. 6.24). The strategy used for the choice of new components is the mixed strategy presented before (this is of course not visible when using the system with the single-part simplified interface).

6.5 Conclusion

In this chapter, we have presented an active learning method that exploits a structuring of the database in the form of a mixture of Gaussians to speed up an active learning process involving the user. The approach is very promising in view of the results obtained on a large database of satellite images and on a generalist database of color images. The tests on the Corel dataset let us think that our method still yields gains in performance, even in cases where the intra-cluster coherence assumption is not strictly verified. The advantages of the proposed approach are directly quantifiable in terms of learning speed (much fewer feedback iterations are needed to arrive at a classifier that satisfies the user) and complexity of the obtained classifier (number of support vectors). The decrease in complexity lets us think that the obtained classifier has better generalization capabilities. It has also the advantage of reducing the computational cost when evaluating the SVM decision function. Also, our method retains some advantages of semi-supervised methods such as the ability to exploit unlabeled data without the drawbacks, such as the constraints in terms of memory occupation and computation time for large volumes of data.

Chapter 7

Cascaded active learning for object retrieval using multiscale coarse-to-fine analysis

High-volume datasets render the task of exploring the database very demanding from a computational point of view. This problem has raised a lot of concern in recent interactive image search engines. The active learning strategies implemented in these systems require the evaluation of the current decision function on the whole dataset as an essential step of the relevance feedback process Tong and Chang [2001]; Gosselin et al. [2008]; Ferecatu and Boujemaa [2007]. Choosing the most informative examples to feed back to the user is indeed based on the response of each element of the database to the decision function associated with the current classifier. However, it should be noted that high-performance classifiers yield complex decision functions involving a non-negligible amount of computations when it comes to the evaluation part.

A way of proceeding is to use machine learning techniques, such as coarse-to-fine strategies, to reduce the amount of data to process. Regarding hierarchical coarse-to-fine approaches, the literature is very abundant in the face detection community. The fact that faces are very rare events in the images and also require complex detectors costly to train and evaluate has naturally led to methods which allow to quickly discard large parts of images, reserving the evaluation of the complex detectors on more promising (as well as more spatially limited) areas (see for example Sahbi and Geman [2006]; Viola and Jones [2004]; Fleuret and Geman [2001]). There are other approaches which still rely on a coarse-to-fine strategy but at the same time try to address specifically the issue of minimizing the cost incurred by the computation of the decision function. In Sahbi et al. [2002], a strategy is developed to reduce the number of support vectors defining the SVM model of the target. In Romdhani et al. [2001], the SVM decision function is not evaluated entirely, the idea being that we do not need all the support vectors in the decision function to determine whether we are in the presence of a target or not.

A second aspect related to object detection is the need for (preferably large) training databases. Most state of the art object recognition methods, whether it be coarse-to-fine methods Sahbi and Geman [2006]; Viola and Jones [2004]; Fleuret and Geman [2001] or more classical learning approaches Perrotton et al. [2010], rely indeed on the

presence of a training set which is used to build the retrieval model during an offline training phase. But this is problematic in the case of high-volume databases where the semantic diversity is very large and there is no *a priori* on the type of object the user may be interested in. The problem of retrieving objects in an active learning context has been so far little investigated in the literature. We can nevertheless mention the work of Abramson et al. Abramson and Freund [2006] in which an extension of the face detection framework of Viola et al. to an active learning setting is proposed. However, this method is not oriented towards the exploration of large databases and, most importantly, the active learning scheme does not operate in unsupervised settings but is used to select informative samples from a pre-constituted training set. In Lechervy et al. [2010], an active learning strategy coupled with the RankBoost algorithm is proposed. A weak classifier is trained on each feature, the combination of the weak classifiers being realized through the intermediary of the boosting framework. The main drawback of this approach is that the number of iterations in the active learning loop is correlated to the dimension of the input space, which is problematic in an active learning context where the involvement from the user must be minimum. Moreover, the proposed framework is not adapted to large databases and the complexity of the obtained classifier is strongly related to the dimension of the input space, which is prohibitive when the dimension is high.

In the following of this chapter, we describe a Cascaded Active Learning method for Object Retrieval, CALOR, which relies on a coarse-to-fine strategy to retrieve any object of interest in large satellite image databases. In Sec. 7.1, we give an overall description of the proposed approach. The next section is devoted to the description of the cascaded active learning process. We justify the use of a certain type of features and classifier and we describe the employed active learning strategy. In Sec. 7.3, we describe a method to propagate the training examples pointed by the user at one level of the hierarchy to the level below. We conclude this part with experimental results on a large database of QuickBird high-resolution optical images.

7.1 Overview of the CALOR (Cascaded Active Learning for Object Retrieval) process

The method we propose is inspired by the needs of satellite imagery users: our aim is to build a system which allows to search for *any* object of interest in large satellite image repositories. We focus this time on iterative *supervised* methods, meaning that we still need the user's assessing iteratively the results suggested by the current model of the target class (relevance feedback). Each large satellite image scene is partitioned into a set of overlapping patches (see Fig. 7.1(b)) which are then organized in a hierarchy following the patch size. Thus, each level of the hierarchy corresponds to a particular patch size, the smallest patches being at the bottom of the hierarchy. The aim of the feedback process is then to learn a cascade of classifiers, each classifier being dedicated to a specific level of the hierarchy and thus to a specific patch size. The cascaded structure allows to quickly eliminate large parts of images in the highest/roughest levels of the hierarchy and to considerably reduce the number of patches to be processed at the lowest/finest levels. Our approach is thus similar in nature to that of Viola et al.

Viola and Jones [2004] who have proposed a cascaded detector for face recognition in video sequences. The principle is to quickly discard large parts of images with a very cheap detector, reserving a more complex processing for more promising areas. The key assumption is that objects being very rare events in the images, we can easily build cheap detectors with very low false negative rates, and then reduce false positive rates by using more and more powerful detectors in lower levels of the cascade Fleuret and Geman [2001].

There are two main contributions of our object detection framework: First, we propose a multiscale coarse-to-fine approach in an active learning context with a strategy of patches size refinement: instead of increasing the discriminating power of the classifier itself as in Viola et al. Viola and Jones [2004], we reduce progressively the patch size (analysis window), relying on the fact that a smaller window will better capture the properties of the object and thus yield a classifier with a much lower false positive rate. The increase in computational complexity comes from the fact that with a smaller analysis window, we will have much more patches to process, justifying the use of the above-mentioned cascaded structure. Second, we describe a Multiple Instance Learning (MIL) Yang [2008] algorithm to propagate *automatically* the training examples from one level of the hierarchy to the other.

The experiments show that our method achieves an important gain in speed but still compares favorably in terms of precision and recall with other recent state of the art active learning approaches for satellite images, such as the one presented in Ferecatu and Boujemaa [2007], working directly at the finest level of the hierarchy. Besides, no further iterations are needed in the active learning loop to arrive at a suitable definition of the targeted object class.

7.2   Cascade of classifiers and learning

We use the following notations: we denote by E_l and C_l, $l = 1, \ldots, L$, respectively the set of patches and the classifier at the l−th level of the L−level hierarchy. T_l refers to the subset of E_l which is to be further processed i.e. $T_l = \{n \in E_l \,|\, \exists x \in T_{l+1}$ such as $n \cap x \neq \emptyset$ and $C_{l+1}(x) = \text{true}\}$. We set $T_L = E_L$. The aim of the feedback loop at the l−th level will thus be to build a classifier which discards as much elements as possible among the set T_l containing the patches which remain to be classified at this level. A synopsis of the whole approach is given in Fig. 7.1(a).

7.2.1   Features and classifiers

Feature extraction   Descriptors are extracted within a sliding window whose size t_l depends on the level l of the hierarchy. We go through each image of the database by shifting the window horizontally and vertically with a step equal to half the window size so as to cover the whole image (see Fig. 7.1(b)). The feature vectors $\left\{v_l^i\right\}_{i=1,\ldots,|E_l|}$ at level l are Weber Local Descriptors (WLD) histograms Chen et al. [2009] computed over the $t_l \times t_l$ analysis windows.

(a)   (b)
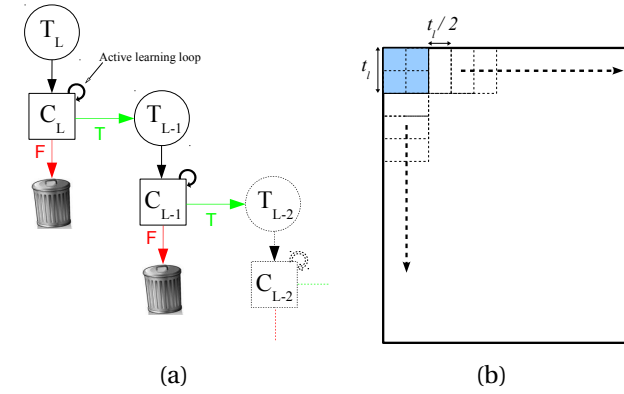
Figure 7.1: **(a)** Synopsis of the CALOR procedure (see Sec. 7.2). The feedback process is done in a top-down manner: we start by building the classifier C_L at the L-th level of the hierarchy. To define the set T_{L-1}, we determine the patches of E_{L-1} which intersect with positively classified patches of the set T_L. We repeat this procedure until we reach the lowest level of the hierarchy. The classifiers C_l are built using an interactive relevance feedback process. **(b)** Extraction of patches from images using a sliding window.

Description of the classifiers   At each level of the hierarchy, we train a probabilistic C-SVM classifier using an active learning strategy involving the user. The use of a probabilistic SVM is justified here by the fact that the examples tagged by the user are propagated in an unsupervised mode between the different levels of the hierarchy and thus, with a certain degree of confidence (see Sec. 7.3). To take this into account, we use a probabilistic SVM with soft-labeled training examples, except those directly tagged by the user which are attributed a label -1 and 1. We use a soft-SVM formulation derived from the theoretical framework proposed in Liu [2006].

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_i \mu_i \xi_i \text{ , s.t. } \begin{cases} y_i(w\cdot\phi(v_i)+b) \geq \mu_i - \xi_i \\ \xi_i \geq 0, \forall i \end{cases} \tag{7.1}$$

The value μ_i is the belief that the training sample v_i belongs to the class y_i. In the following, μ_i is to be interpreted as a membership degree, so it is considered to lie in the interval $[0,1]$ although the problem 7.1 could be solved as well with negative values of μ_i. The constraints $y_i(w\cdot\phi(v_i)+b) \geq \mu_i - \xi_i$ allow to account for unbalanced memberships by shifting the SVM boundary towards the most uncertain training samples. As in the standard SVM, ξ_i are slack variables which penalize the objective function in the case of misclassification. The problem (7.1) is solved under its dual form:

$$\max_{\alpha_i} \sum_{i=1} \mu_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\text{s.t. } \sum_{i=1} y_i \alpha_i = 0 \text{ and } 0 \leq \alpha_i \leq \mu_i C, \forall i \tag{7.2}$$

yielding the final decision function:

$$f(v) = \sum_{i \in SV} y_i \alpha_i k(v, v_i) + b$$

where the α_i are the coefficients associated with the support vectors SV and k the kernel function. Since we are working on (WLD) histograms, an interesting kernel function is the Gaussian kernel $k(x_i, x_j) = \exp\left(\frac{-d(x_i,x_j)^2}{2\lambda^2}\right)$ since it allows to chose any distance function d and, in particular, histogram distances. In our case, we made our tests with a χ^2 distance $d(x_i, x_j)^2 = \sum_{r=1}^{R} \frac{(x_{ri}-x_{rj})^2}{x_{ri}+x_{rj}}$ where R is the dimension of the WLD histograms. We could use more robust distances such as the Earth Mover Distance but this leads to extra-computational costs and might prove problematic to process huge databases since the distance function is directly involved in the computation of the decision function associated with the current classifier. In the following, we denote by a triplet $\left\{\{v_i\}_i, \{\mu_i\}_i, \{y_i\}_i\right\}$ the training set used to compute the classifier (7.1).

7.2.2 Active learning strategy

Choice of feedback examples   Choosing which examples to feed back to the user is a key issue of active learning. The most common approach used with SVM-like classifiers is to select the most ambiguous elements, that is, those which are the closest to the SVM separating surface. However, it is better to add a criterion to ensure sparsity of the selected samples. We propose a strategy based on the solution of the following problem (see Ferecatu and Boujemaa [2007]):

$$\arg\max_{i_1,\dots,i_D\in S} \min_{\substack{(j_1,j_2)\in\{i_1,\dots,i_D\} \\ \text{with } j_1<j_2}} d(v_{j_1}, v_{j_2}) \tag{7.3}$$

In the above equation, D is the size of the display and $i_1,\dots,i_D$ are the indexes of the D selected elements among a set S of n elements with $n > D$. The set S is built by considering the elements which are close to the SVM margin i.e. $S = \left\{v \,\middle|\, \left|f(v)\right| < \epsilon\right\}$. To express the problem 7.3 in plain language, we look for the d elements of the set S whose minimum pairwise distance is maximum. A sub-optimal solution to this combinatorial problem is obtained by setting arbitrarily the first element i_1 and considering the iterative rule: $i_t^* = \arg\max_{i\in S\setminus\{i_1^*,\dots,i_{t-1}^*\}} \min_{j\in\{i_1^*,\dots,i_{t-1}^*\}} d(v_i, v_j)$.

Cascaded active learning strategy   The feedback is done in a top-down manner, that is, we start from the highest level of the hierarchy which corresponds to the largest size of the analysis window and we go down progressively. However, since it is difficult to assess the number of iterations necessary at each level, we use a heuristic criterion based on experimental results such as the one shown in Fig. 7.7(a). Our objective is to ensure a low False Negative Rate (FNR) at each level. Fig. 7.7(a) presents the average recall (which is equal to $1 - \text{FNR}$) over ten image classes, as a function of the percentage η of top-ranked elements that we retain for the recall computation. The ranking function is the decision function $f_l(v)$ of a SVM classifier working at level l. We thus make the recall computation over the q_l top-ranked elements $v_l^{i_1}, v_l^{i_2}, \dots, v_l^{i_{q_l}} \in E_l$ where $q_l = \lfloor \eta |E_l| \rfloor$. For instance, we see on the curves of the figure 7.7(a) that with a classifier learned over ten iterations of feedback, we can safely discard around 70% of the database for a patch size of 100×100 pixels, still ensuring a recall of 90%, i.e. a False Negative Rate of 10%. The values of precision and recall in Fig. 7.7 are computed on the percentage of the database we keep.
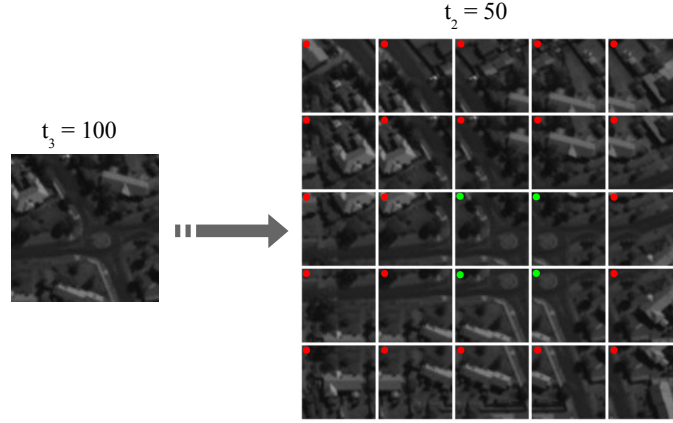
Figure 7.2: Patches of size 50 pixels (right) intersecting with a patch of size 100 pixels (left). This example illustrates the difficulty of propagating training samples from one level of the hierarchy to the level below.
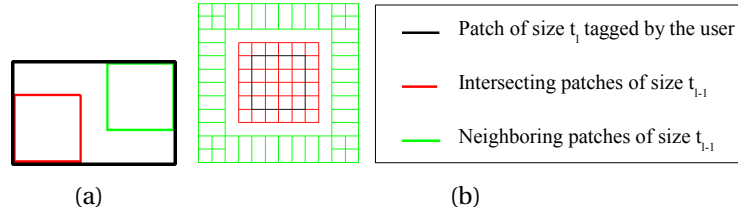
Figure 7.3: **(a)** Minimum enclosing rectangle. **(b)** Extracting the neighborhood of size t_{l-1} from a user-tagged patch of size t_l.

7.3   Inter-level propagation of training examples

7.3.1   Positioning of the problem

One key issue of our approach is to propagate the examples tagged by the user during the feedback process at level l to the level below $l-1$. The case of a negative example v_l^i is straightforward: we determine all the negative patches of size t_{l-1} intersecting with v_l^i (i.e. the elements of the set $b_i^- = \left\{ v_{l-1}^j \in E_{l-1} | v_{l-1}^j \cap v_l^i \neq \emptyset \right\}$) and give them a negative label with an associated confidence level of 1. The case of positive examples is more difficult, because we do not know which ones of the intersecting elements of size t_{l-1} will effectively contain the targeted object (see Fig 7.2). The only information we possess is that each set b_i^+ contains one or several positive elements (those marked with a green dot in Fig. 7.2), the other being negative elements (those marked with a red dot). Our problem is to identify in each positive bag which is the most representative element, that is, the one which is more likely to contain the object. This is typically a MIL framework though our problem verifies only one of the two fundamental underlying hypothesis of MIL problems. A key assumption of MIL (besides the fact that the positive elements of positive bags must belong to the same class) is that the negative elements of the positive bags must have some similarities with the elements of

the negative bags so that we can differentiate the positive elements from the others. The idea behind most MIL algorithms is indeed to identify the "outstanding" elements that the positive bags have in common. If the negative elements of the positive bags are different from the elements of the negative bags, they might be considered as "outstanding" as well, which is, of course, not the expected behavior. In our case, we are in the wrong settings since negative bags are taken as the negative examples given by the user. They are thus very unlikely to share a similar visual context with the negative elements of the positive bags. To cope with this, we discard the negative bags coming from the user feedback and we replace them with the neighbors of positive bags, relying on the assumption that the context of the neighbors will be the same as that of the negative elements of the positive bags, which is most often the case due to low spatial variability in satellite images (see Fig. 7.3(b)). To build the neighborhoods associated with the positive bags, we define a distance between square patches of equal sizes based on the length $s_{\max}$ of the longest side of the minimum enclosing rectangle (that is, the rectangle of smallest area containing the two patches (see figure 7.3(a))). Denoting by s_i and s_j two squares of side length t, we set $d_{sq}(s_i, s_j) = (s_{\max} - t)/t$. We then define the negative neighborhood of a positive bag b_i^+ at level l as the set $N_i = \text{Neigh}\left(b_i^+\right) = \left\{n \in E_l | \exists n_b \in b_i^+ \text{ such as } 1/2 \leq d_{sq}(n, n_b) \leq 3/2\right\}$.

To solve this problem, we use a common SVM-based MIL formulation Andrews et al. [2003], looking for the K elements of the K positive bags such that the (possibly probabilistic) SVM problem trained with this K elements as the positive part of the training set and the elements of negative bags as the negative part of the training set yields the largest possible margin. This is a combinatorial problem which becomes quickly intractable as K augments. We propose in Sec. 7.3.2 an alternative solution to the brute force approach. The latter consists indeed in evaluating all the possible combinations of K elements, which is impossible to do in practice: let's say the user-tagged patches are of size 100×100 pixels and the patches of the level below are of size 50×50. This gives us bags of cardinality 25. Thus, for 10 feedback rounds and 3 positively-tagged patches per round in average, we obtain in the end 30 positive bags and $25^{30} \approx 10^{42}$ different problems to train and evaluate ! In the following, we denote by $N = \bigcup_i N_i$ the set of elements contained in the negative bags and σ the selection function whose value is 1 on the selected element of each positive bag and 0 on the other elements of the positive bags. We denote by S the set of positive elements which do not belong to a bag, but which can be incorporated in the MIL problem as well. The probabilistic SVM-based MIL problem can be formulated in the following way:

$$\min_{\sigma, w, b, \xi} \frac{1}{2}\|w\|^2 + C \sum_{i \,|\, v_i \in N \cup P_\sigma \cup S} \mu_i \xi_i$$

$$\text{s.t.} \begin{cases} (w \cdot \phi(v_i) + b) \geq \mu_i - \xi_i, \, \forall i \,|\, v_i \in P_\sigma \cup S \\ -(w \cdot \phi(v_i) + b) \geq \mu_i - \xi_i, \, \forall i \,|\, v_i \in N \\ \xi_i \geq 0, \, \forall i \,|\, v_i \in N \cup P_\sigma \cup S \end{cases} \tag{7.4}$$

where $P_\sigma = \{v_{i_1} \in b_1^+, \ldots, v_{i_K} \in b_K^+ \text{ such as } \sigma(v_{i_j}) = 1 \text{ for } j = 1, \ldots, K\}$. In the problem (7.4), the non-selected elements of the positive bags are discarded. To denote the train-

ing set of the MIL classifier, we use a couple of ensembles of the following form:

$$\left\{ \begin{array}{l} \left\{ b_1 \bullet \ldots \bullet b_K , \mu_1 \bullet \ldots \bullet \mu_K \right\} ; \\[2mm] \left\{ S \cup N , \mu_S \cup \mu_N , \mathbb{1}_{|S|} \cup -\mathbb{1}_{|N|} \right\} \end{array} \right\}$$

The first ensemble consists only of positive bags b_i with their associated confidence level μ_i. The second ensemble consists of single positive and negative elements, S and N referring respectively to the set of positive elements and to the set of negative elements. In our notations, $|S|$ is the cardinal of the set S and $\mathbb{1}_{|S|}$ is the multiset of ones of size $|S|$. The sets μ_S and μ_N contain the confidence levels associated respectively with the elements of S and N.

7.3.2   Solving the MIL problem

In this section, we propose an iterative solution to the problem (7.4) that we refer to as Incremental MIL-SVM (IMSVM) in the following. To eliminate redundant information and reduce the cardinality of the negative set N, we perform a clustering on this set using the Enhanced LBG (ELBG) algorithm Patané and Russo [2001] described in section 4.2. The ELBG algorithm is much faster (around 4 times) than the K-means algorithm since it requires in average much less iterations to converge for the same computational complexity at each iteration, and thus proves to be well suited for our application where the clustering operations are to be performed in an active learning context. We furthermore improve the quickness of convergence by choosing the initial codeword. This allows to achieve a reduction in the number of iterations by a factor 3 in average. We use an heuristic which consists in sampling the initial seeds according to the underlying data density. Among the ways of sampling through an unknown discrete density of points, we propose an approximation which boils down to sampling according to the kd-tree built from the data points on which we intend to perform the clustering. The sampling operation on a kd-tree amounts to performing a random walk from the root to a leaf. In other words, while descending from the root, a branch is selected randomly at each level with a probability of 0.5 (kd-trees are binary trees). Even if this delivers an extremely-biased sampling due to the fact that, at a given level, the nodes of the tree possess a different number of branches, we achieve a significant reduction of the number of iterations in the following clustering step using an almost costless operation (sampling from a binary tree yields an algorithmic complexity of $O(h)$ where h is the height of the tree). We fix arbitrarily the number of clusters to $4 \times K$, where K is the number positive bags. This can be explained by the fact that we consider square neighborhoods around the positive bags, which leads us to look in four different directions to extract the negative elements. Common sense suggests then that we will observe at most four different contexts among the negative elements, necessitating a codeword of 4 elements. We have performed tests (not reproduced here) using a higher number of codewords but we did not observe significant improvements over the solution of our MIL-problem. The clustering process allows also to manage the presence of target objects in the negative neighborhoods of positive bags (this is rare but still likely to occur). By setting the number of codewords to only 4 elements per positive bags, the

clustering process will indeed only model the context of objects and not the objects themselves and thus naturally discard the target objects from the negative elements.

Description of the proposed MIL procedure   We start by replacing the set N of negative elements by a set P of much lower cardinality containing the cluster prototypes obtained using the ELBG algorithm with the improvements described in the preceding paragraph. Our solution consists in solving separately the K SVM-based MIL problems associated with each of the K positive bags, i.e. the problems defined by the MIL training sets:

$$\left\{\{b_k^+, \gamma_k^{t-1}\} ; \{P, \mathbb{1}_{|P|}, -\mathbb{1}_{|P|}\}\right\}_{k=1,\dots,K}$$

The idea is then to exploit the information that the positive bags have in common (they each possess at least one representative of the target class) by incorporating at each iteration t the most trustable solution not yet incorporated among the K solutions $\{s_k^t\}_{k=1,\dots,K}$ of the K MIL problems. The iterative process stops when one element per positive bag has been incorporated into the training set. The problem is to find a way to compute at each iteration a confidence level γ_k^t for each of the K solutions in a completely unsupervised way. To do this, we train a probabilistic SVM 7.1 on the triplet $\left\{\{s_k^t\}_{k=1,\dots,K} \cup P, \{\gamma_k^{t-1}\}_{k=1,\dots,K} \cup \mathbb{1}_{|P|}, \mathbb{1}_K \cup -\mathbb{1}_{|P|}\right\}$ and then we determine the γ_k^t as the probabilistic outputs obtained by fitting a sigmoid over the values given by the decision function of the probabilistic SVM, as explained in Platt Platt [2000]. In our formulation, we optimize a slightly different log-likelihood functional. We indeed replace the probabilities of correct label that are computed in Platt using an out-of-sample model by the confidence levels from the preceding iteration, yielding the following expression of the negative log-likelihood of the training data:

$$\mathcal{L}_t(a, b) = -\sum_{i \,|\, v_i \in \{s_k^t\}_{k=1,\dots,K} \cup P} \left[t_i \log(p_i) + (1 - t_i) \log(1 - p_i) \right]$$

where $p_i = \frac{1}{1+\exp(ag_t(v_i)+b)}$, $t_i = \gamma_k^t$ if $v_i \in \{s_k^t\}_{k=1,\dots,K}$ and $t_i = 0$ if $v_i \in P$. The parameters (a_t, b_t) of the sigmoid are then obtained as $(a_t, b_t) = \arg\min_{a,b} \mathcal{L}_t(a, b)$. A summary of the whole procedure is given in the Algorithm 1. This algorithm is similar in nature to a backtracking algorithm: to complete the set of already incorporated elements I_{t-1} at iteration t, we choose the positive bag $k \in \{1, \dots, K\} \setminus I_{t-1}$ with the highest associated confidence level γ_k^t and we update the already constituted training set, that is, the set $\{s_k^t\}_{k \in I_{t-1}}$.

In the above algorithm, the function ELBG(S) performs a clustering of the set S and returns the obtained cluster prototypes. The function PROB_SVM(T_1) solves the probabilistic SVM problem 7.1 on the training set T_1 and returns the decision function associated with the computed classifier. As for the function MIL_SVM(T_2), it solves the SVM-based MIL problem 7.4 on the training set T_2 and returns the positive element(s) of the positive bag(s). A complete overview of the cascaded active learning scheme with automatic inter-level propagation of training samples is given in the appendix D.1.

Algorithm 6 IMSVM algorithm for the *automatic* propagation of training samples between consecutive levels of the hierarchy.

Input
- Input a set of positive bags $\{b_k^+\}_{k=1,\dots,K}$ with associated initial membership degrees $\{\gamma_k^0\}_{k=1,\dots,K}$

Initialization
- Set $I_0 = \emptyset$ and $P = \text{ELBG}\left(\text{Neigh}\left(\{b_k^+\}_{k=1,\dots,K}\right)\right)$

For $t = 1,\dots,K$ **do**

- $\{s_k^t\} \leftarrow \text{MIL\_SVM}(M_k)$, $k = 1,\dots,K$ where $M_k = \left\{\{b_k^+, \gamma_k^{t-1}\}; \left\{\left\{s_j^{t-1}\right\}_{J_{t-1}} \cup P, \left\{\gamma_j^{t-1}\right\}_{j \in J_{t-1}} \cup \mathbb{1}_{|P|}, \quad \mathbb{1}_{|J_{t-1}|} \cup -\mathbb{1}_{|P|}\right\}\right\}$ and $J_{t-1} = \{j \in I_{t-1}, j \neq k\}$
- $g_t \leftarrow \text{PROB\_SVM}\left(\{\{s_k^t\}_{k=1,\dots,K} \cup P, \{\gamma_k^{t-1}\}_{k=1,\dots,K} \cup \mathbb{1}_{|P|}, \quad \mathbb{1}_K \cup -\mathbb{1}_{|P|}\}\right)$
- Compute sigmoid parameters (a_t, b_t)
- Compute confidence levels $\gamma_k^t = \frac{1}{1+\exp(a_t g_t(s_k^t)+b_t)}$
- Set $I_t = I_{t-1} \cup k^*$ where $k^* = \arg\max_{k \in \{1,\dots,K\} \setminus I_{t-1}} \gamma_k^t$

End For
- Return the set $\{s_k^K\}_{k=1,\dots,K}$ and the set of associated membership degrees $\{\gamma_k^K\}_{k=1,\dots,K}$

7.4   Experiments and results

We conducted our tests on a database of 10 panchromatic QuickBird scenes with a ground resolution of 61cm. The scenes represent overall views of Acapulco, Las Vegas, Los Angeles, London and Ouagadougou and are of approximate size 30000 by 30000 pixels. We used a 4-level hierarchy with patches of size 200, 100, 50 and 25 pixels for the finest level. We assessed our approach on ten object classes: roundabouts, storehouses, tall buildings, marina, moving boats, gas holders, swimming pools, crossroads, planes and baseball grounds. Some representative examples of these classes are given in Fig. 7.4. The results are given in terms of the amount of calculation at each iteration of the feedback loop, that is, the number of evaluations of the current SVM decision function multiplied by the number of support vectors defining the current SVM model. We compare our method to the recent state of the art relevance feedback SVM approach described in Ferecatu and Boujemaa [2007], working directly at the finest level of the hierarchy. This method has been reported to produce good results for satellite images and we refer to it as SVM$_{baseline}$ in the following. We did not choose for comparison the method proposed in Lechervy et al. [2010] since, in our case, the dimension of the input space is very high (240), which would require a considerable number of iterations in the active learning loop to build the boosting classifier.

The two methods are given the same starting point at different scales. For instance, for the class "roundabout", we initialize our method with a patch of size 200×200 centered on a roundabout and we initialize the baseline with a patch of size 25×25 centered on the same roundabout. To ensure equality of chances between the two methods, the user is forced at each active learning iteration to give a feedback on all images which are presented to him on the display.
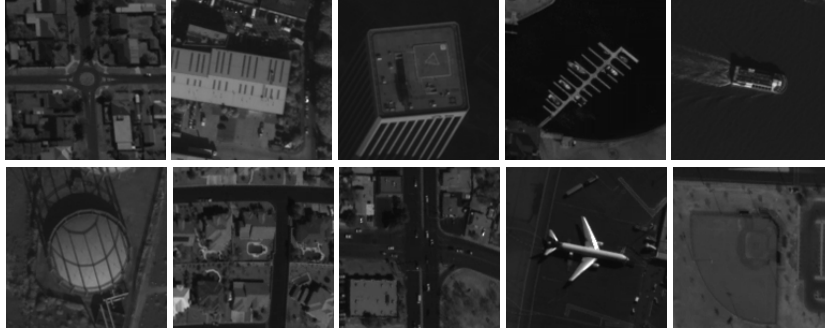
Figure 7.4: Examples of elements of the ten target classes used for the experiments. First row, from left to right: roundabouts, storehouses, tall buildings, marina and moving boats. Second row, from left to right: gas holders, swimming pools, crossroads, planes and baseball grounds.

We see that using our method (blue curves), much less computations are needed than using SVM$_{baseline}$ (green curves): we observe on the figure 7.7(b) that there is a difference of more than two orders of magnitude in average. The curves of the figures 7.7(c) and 7.7(d) show respectively the precision and the recall averaged over the ten classes at each iteration of the feedback loop. For the precision and recall computation, we consider that a sample x at level l is positively classified iff $x \in T_1$ and $f_l^{q_1}(x) \geq 0$. We notice that, in average, our method performs a little better in terms of precision at the end of the retrieval process while the recall is approximately the same. We observe that after ten iterations of active learning, the recall does not improve anymore or in a very insignificant way during the remaining iterations. This saturation can be explained by the fact that we are using an active learning strategy which is oriented towards the exploration of the database by trying to maximize the sparsity of the samples which are presented to the user at each iteration of the active learning loop.

The curves of the figures 7.5 and 7.6 show the detail for the precision and the recall for each of the ten classes. The results are averaged each time over 20 runs of the active learning process, each run being performed by the same real user. The "jumps" we observe every ten iterations on the blue curves are due to the changes in scale and the fact that we keep only a small percentage of the top-ranked elements when switching to the level below in the hierarchy: at level 200, we keep 35% of the top-ranked elements, which yields an estimated recall of 90%. At level 100, 50 and 25, we retain respectively 30%, 20% and 5% of the top-ranked elements to maintain an estimated recall of 90% each time as well (see figure 7.7(a)).

The results obtained on the classes "roundabouts", "marina", "moving boats", "gas holders" and "crossroads" are quite good in terms of precision and recall for both our method and the method we are comparing ourselves to. This can be explained by the fact that these classes are visually consistent and that the objects they contain appear in similar contexts (for instance, crossroads and roundabouts are situated most of the time in a urban context, while marina and boats are generally located in water areas). The underlying learning problem is thus much easier: the similarity of the contexts in which the objects from a same class can be found as well as the visual coherence of the objects within a class will indeed be repercuted at signal level, rendering the task

of exploring the database and of delineating the targeted object class with an SVM sep-arating surface much easier. The lack of visual consistence will explain the fact that classes like "buildings" and "storehouses" will yield very poor results. There is indeed a huge visual variability inside such classes. It shows some limitation of our method to effectively bridge the semantic gap in such cases. Another problem comes from the size of the patches in the top-down processing cascade. There is indeed also some variability between the different classes and our method as well as the baseline would benefit from an adjustment of the analysis window depending on the scale of the tar-geted object. We see for instance that for the class "tall buildings", a window of size 25×25 pixels which is the patch size at the finest level of the used hierarchy is a lit-tle under-dimensioned compared to the size of the object. The consequences can be directly observed on the precision curve 7.5(c): during the iterations $20 \rightarrow 40$ which corresponds to the two finest levels of the hierarchy and thus to an unadapted size of the analysis window, the precision remains the same and even decreases at the end the learning process. On the contrary, during the iteration $1 \rightarrow 19$, the precision increases in a much more significant way. The opposite problem can be observed as well: for the class "swimming pool", the analysis window at the highest levels of the hierarchy will be a little over-dimensioned and thus will capture to much of the englobing context, yielding a descriptor which is not sufficiently discriminant for the characterization of the class. It has a direct influence in terms of precision as well: on the curve 7.5(g), we observe that the most effective part of the active learning process is situated be-tween the iterations $30 \rightarrow 40$, which corresponds to the finest level of the hierarchy and the precision does not improve much over the first 29 iterations which corresponds to "coarser" levels. The same remark can be done about the class "moving boats" (see 7.5(e)). A suggestion for improving our method would thus be to design an adaptive hierarchy in which the size of the analysis window at each stage is adjusted relatively to the scale of the targeted object.

The detail of the amount of computations per class is not represented because there isn't enough visual difference between the ten classes due to the employed semi-logarithmic scale. The difference of two orders in magnitude remains approximately the same through all the classes, yielding the expected conclusion that our system allows to considerably reduce the amount of computations regarding the evaluation of the decision function, while maintaining approximately the same performance in terms of precision and recall at the end of the retrieval process.

7.5 Conclusion

In this chapter, we have presented a multiscale "coarse-to-fine" cascaded approach to perform object retrieval in large satellite images. Unlike most techniques in the liter-ature which rely on large training sets to build the retrieval model, ours is based on an active learning process which allows the user to search for *any* object he might be interested in without necessitating the building of an exhaustive training database of objects (which might prove impossible due to the diversity of objects encountered in satellite images) and a costly offline training phase. In our case, the "coarse-to-fine" exploration of the database was performed by building SVM models working on de-creasing size of patches, yielding a more and more spatially restrained definition of

the target object class. We have proposed a multiple instance learning algorithm to propagate the user-tagged examples from one level of the hierarchy to the other. The presented MIL algorithm addresses a more general problem, which is to retrieve exact locations of objects in training samples and thus could be used separately to solve various problems related to object recognition such as reducing training sets error rates, which is quite a recurring problem. An active learning strategy was also introduced, which ensures sparsity among the selected feedback examples without resorting to costly clustering techniques.

Figure 7.5:   Precision as a function of the number of active learning iterations. The blue curves represent the results given by our method and the green curves the results given by $SVM_{baseline}$.
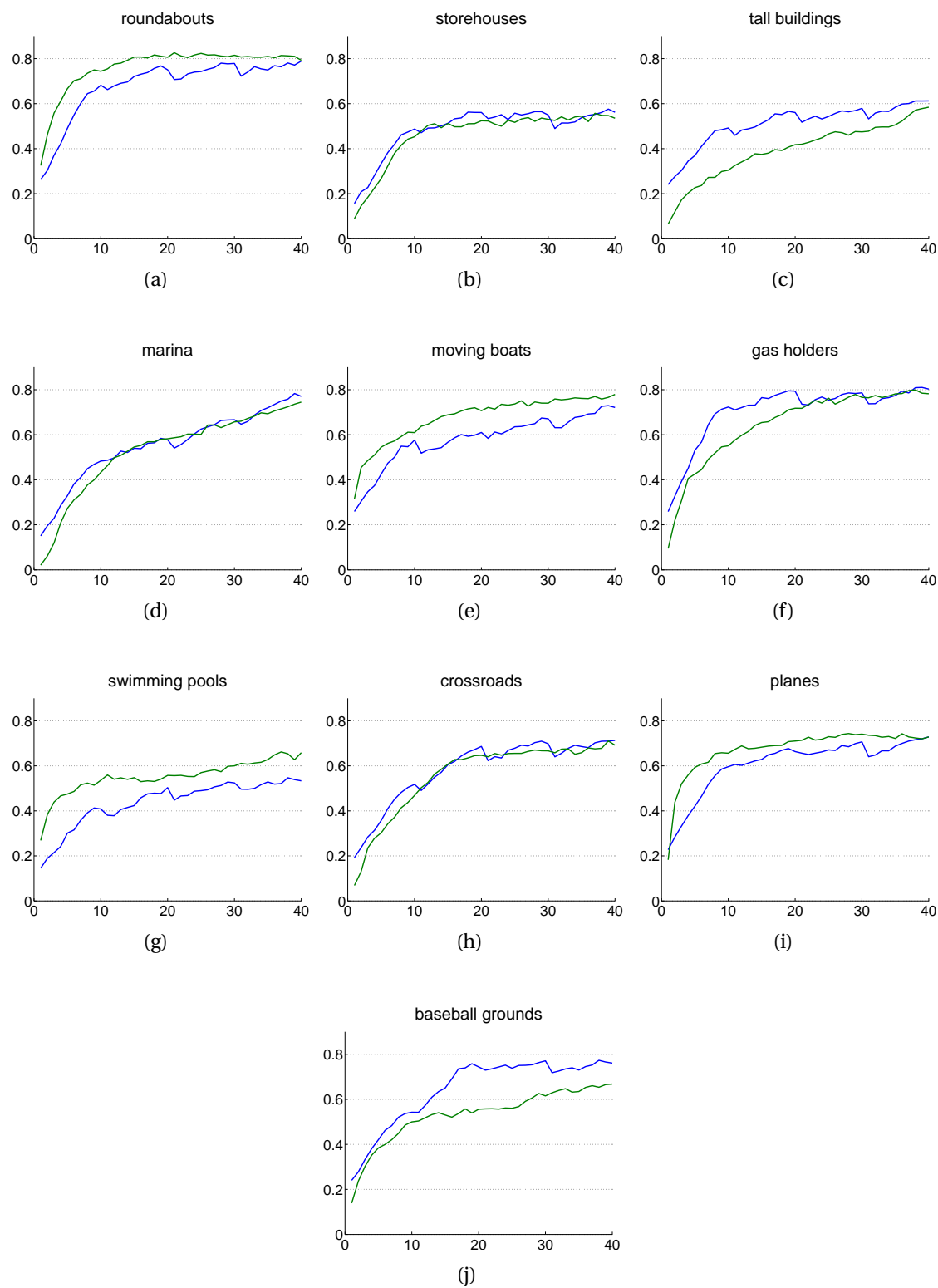
Figure 7.6: Recall as a function of the number of active learning iterations. The blue curves represent the results given by our method and the green curves the results given by $\text{SVM}_{baseline}$.

Figure 7.7: **(a):** Average recall as a function of the percentage of top-ranked elements we keep in the whole database for four different sizes of patches. The ranking function is the decision function of $\text{SVM}_{baseline}$ after ten iterations of feedback. **(b)** average amount of calculation at each 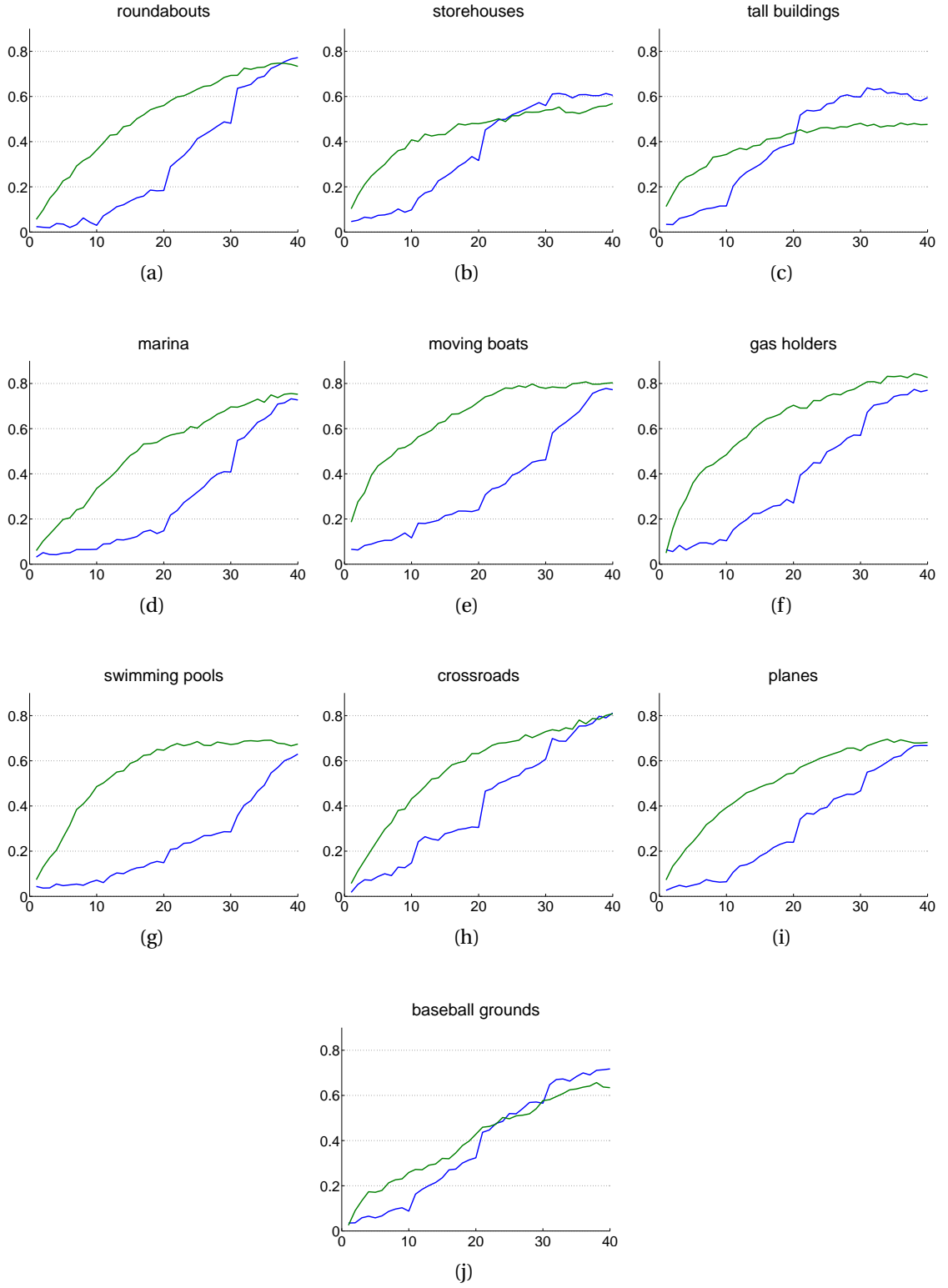iteration of the feedback loop. **(c)** and **(d):** average precision and recall as a function of the number of active learning iterations. The blue curves represent the results given by our method and the green curves the results given by $\text{SVM}_{baseline}$.
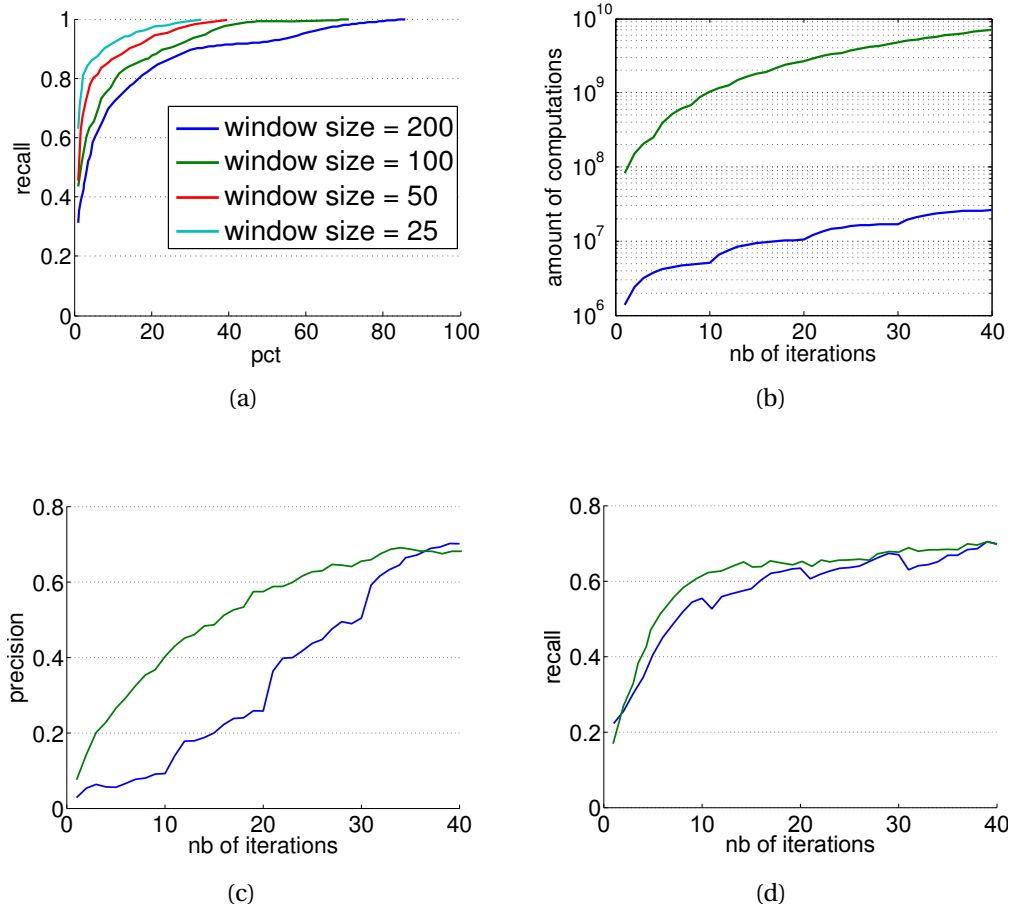
Chapter 8

Conclusions and perspectives

In this work, we proposed some solutions to the complex problem of mining high-volume satellite image repositories using a few training examples. We envisaged this problem both from the point of view of auto-annotations systems and of interactive image/object category search engines. We also proposed a "unifying" concept showing the complementarity of our three contributions when it comes to addressing the aforementioned problem.

8.1 Conclusions

In the chapter 5, we proposed a semi-supervised algorithm to perform auto-annotation of satellite image repositories and to discover "unknown" semantic structures among these repositories. In a real case of study, the non-exhaustiveness of training databases is indeed the rule rather than the exception so it is important to design methods that take this fact into account, especially when using semi-supervised methods which very often do the hypothesis that the distribution of unlabeled data follows that of unlabeled data. By incorporating an unknown classes discovery feature, we avoid making this assumption and we can plainly exploit the information contained inside the unlabeled data without running the risk of perturbing the learning with "noisy" classes in the unlabeled data not represented in the training database. We demonstrate successfully our approach on a database of SPOT5 panchromatic satellite imagery, making obvious the deficiency of state-of-the-art semi-supervised methods when it comes to learning from non-exhaustive training datasets while trying to make the most out of the unlabeled data. We can nevertheless point out some weaknesses of our method regarding the learning of the auto-annotating model which is done through a Maximum Likelihood estimator obtained using an EM algorithm. This algorithm is indeed subject to convergence towards local minima, but we try to cope with this drawback in appendix B by proposing an "annealed" version of the preceding auto-annotation algorithm. The statistical modeling isn't of course discriminative enough to allow the learning of complex image classes but the results on moderately complex image classes are very satisfying regarding the original goal which was to address the semi-supervised learning problem in the presence of an incomplete training dataset and to point out existing structures in the data that aren't represented in it.

In the chapter 6, we proposed a semi-supervised active learning algorithm in the

context of interactive image search engines whose goal is to speed up the learning of the target category by exploiting the intrinsic structure of the data under the form of an unsupervised mixture of Gaussians. As in the preceding case, the algorithm is designed so as to avoid making the standard assumption of semi-supervised methods regarding the distribution of labeled and unlabeled data. The proposed algorithm allows to work at a lower granularity of space and consequently to position much quickly an approximation of the separating surface delineating the target category. The obtained approximation can then be refined by bringing back the learning problem at the finest level of granularity, that is, the level corresponding to the original data points in the input space. The compromise between the accuracy and the quickness of the learning has to be finely tuned through the adjustment of the number of mixture components in the Gaussian mixture model. Too few mixture components will lead to a very quick learning phase (i.e., the number of iterations in the active learning loop will be small) since the learning will amount in this case to position a surface separating a small number mixture components while adjusting their convex hulls. The accuracy on the contrary will be impaired because of the coarseness of the original model and the lack of consistency of the mixture components which we consider as the training entities used to build the classifier. On the opposite, too large a number of mixture components will slow down the learning though increasing the accuracy. A compromise has thus to be found which is not quite easy to do on an unknown dataset. Our method does nevertheless bring noticeable improvements in terms of learning speed while achieving sensibly the same performance in terms of precision and recall (the test were performed both on a database of high-resolution panchromatic optical QuickBird images and on a generalist database of color images built from the Corel dataset which is one of the most challenging datasets in machine learning as far as intraclass and interclass variability are concerned).

Our last chapter of contributions is dedicated to the description of an object retrieval algorithm encapsulated within an active learning scheme. A major drawback of standard active learning scheme is their time of response between two iterations of the learning process in a context where fast exchanges between the system and the user have to be favored. This is mainly due to the fact that most of these scheme rely on complex classifiers to model complex classes (of objects) and thus lead to multiple evaluations of complex decisions functions at each active learning iterations. This becomes quickly impracticable as the database grows up since it drastically slows down the response of the system. We consequently propose a coarse-to-fine strategy still allowing the use of complex classifier functions while avoiding the explosion of the amount of computations which characterizes most state-of-the-art approaches when confronted to the problem of learning complex image classes inside high-volume repositories. Our strategy consists in considering several levels of patch sizes and of eliminating most patches in the highest levels of the corresponding hierarchy (the higher levels contain much less patches since we consider bigger patch sizes in the higher levels so it is still envisageable to process all those patches or at least a great part of them). Our results on a database of high-resolution panchromatic optical QuickBird images are quite convincing since our method achieves a reduction of the number of computations by two orders of magnitude in average while attaining quite the same performance in terms of precision and recall. The number of active learning itera-

tions at each stage of the hierarchy has still to be well adjusted to avoid "eliminating too much" in the highest levels as well as keeping too many irrelevant elements in the levels below.

8.2   Perspectives

Several interesting axes of study arise from this work some of them are exposed in the following.

8.2.1   Semi-Supervised Annotation and Unknown Semantic Structures Discovery in Satellite Image Repositories

An important issue related to this part is the use of "atomic" concepts to annotate images, that is, according to our definition of those, the use of concepts which lead to the smallest possible intraclass variability on visual properties. To effectively bridge the semantic gap, we may need in some situations to introduce higher-level semantic concepts such as user-defined ones. Starting from this observation, an interesting perspective would be to extend our system so it can learn user-specific taxonomies (that is hierarchical representations of concepts) [Talavera and Béjar, 2001]. In this scenario, we would trace back each high-level concept to its unigram representation in terms of "atomic" concepts which in our definition would be the concepts situated at the lowest level of the taxonomy. Since our model can produce unigram models for each image, a similarity measure between unigrams such as the Earth Mover Distance [Rubner et al., 1998] could be employed to retrieve similar images.

8.2.2   Active Learning Using the Data Distribution for Interactive Image Classification and Retrieval

A possible extension of this part of our work would be to investigate how the notion of distance could be modified in order to better suit the user request. The interactions with the user could indeed be used as well to learn which descriptor is the most appropriate given the user request. We observe indeed some limitations due to some descriptor masking an other: for instance, when looking for small objects like power pylons in very textured images such as desert, the texture descriptor corresponding to the background tends to mask an other descriptor which might better characterize the object we are looking for. The result is that the system does not understand the user's request and thus will only retrieve desert images.

Another very important aspect in satellite imagery is the notion of low spatial variability. That is, the label attributed to a patch will be strongly conditioned on the labels of the neighboring patches. Such an information could be exploited to reduce misclassification, especially when combined with a priori knowledge on the possible neighborings (for instance, patches containing beach areas are likely to be close to patches "containing" sea).

8.2.3   Cascaded Active Learning for Object Retrieval using Multiscale Coarse-to-fine Analysis

We are currently investigating a way of estimating the false negative rate at each iteration of the active learning strategy. This is a recurring problem of active learning methods where it is always problematic to estimate the percentage of "top-classified" elements to keep to ensure a good tradeoff between precision and recall. This problem is directly linked to that of tuning the number of active learning iterations necessary at each stage of the cascaded active learning process to achieve a good identification of the targeted object class. We are also planning to extend our framework to more specific object recognition problems like face detection.

8.2.4   Other perspectives

The use of database technologies as well as that of space-partitioning data structures would provide very interesting perspectives in terms of optimization and concrete implementation of the proposed procedures. Even if our methods are oriented towards mining from large image repositories, we do no provide any hint on a concrete implementations of these methods inside a geographical information system operating on thousands of terabytes of data. This problem was of course far beyond the scope of this thesis and besides more an engineering problem than a problem of machine learning research, that is why, we did not address it as such but it would certainly open very interesting perspectives in the direct continuation of this study. To take an example, we could think in our second contribution of a space-partitioning data structure implementing the structuring of the input space under the form of a mixture of Gaussians. The idea of convex hulls could be transcribed as well directly inside a data structure. Regarding our coarse-to-fine scheme for object retrieval, there exists structures such as quadtrees which concretely implement the idea of hierarchy based on the patch size, and, thus, which could greatly improve the efficiency of our algorithms.

Another perspective concerns the use of visual data mining techniques to speed up the learning of the target class in interactive image search engines. Dimensionality reduction techniques such as manifold learning provides very interesting representations of image databases allowing the user to visualize all at once the content of the databases. We could for instance imagine a querying process where the user selects coherent image groups in the low-dimensional representation obtained using manifold learning algorithms. This might bring significant improvements to the standard scheme consisting in performing a query with a single image. We could imagine an active learning algorithm based on a query-point-movement-like strategy, the modification of the query being directly performed in the spectral space obtained through the help of manifold learning techniques such as Laplacian Eigenmaps.

Appendix A

Estimation of the auto-annotation model parameters (5.2): details of the computations

In this appendix, we provide the details of the computations to obtain the update formulas in the M steps of the EM algorithms used to estimate the parameters of the auto-annotation models presented in the figures 5.6 and 5.7. We start with the case of no explicit associations between image concepts and feature vectors (5.2.1) and then, we derive the formulas in the case where the associations in the training datasets are explicit (5.2.2). The notations are the same than those used in the sections 5.2.1 and 5.2.2.

A.1 Case 1: no explicit associations between feature vectors and "atomic" concepts 5.2.1)

In this case, the maximization step of the EM algorithm consists in maximizing the quantity $Q_D(\theta|\theta_t)$ with respect to θ:

$$\theta_{t+1} = \arg\max_{\theta} Q_D(\theta|\theta_t) \tag{A.1}$$

where

$$Q_D(\theta|\theta_t) = \sum_{k=1}^{K} \sum_{l=1}^{L} p(H_l^k = 1|d_k, \theta_t) \cdot \log\left[p(d_k|c_l, \theta) \cdot p(c_l)\right]$$

and

$$p(H_l^k = 1|d_k, \theta_t) = \frac{p(H_l^k = 1, d_k|\theta_t)}{p(d_k|\theta_t)} = \frac{p(H_l^k = 1|\theta_t)p(d_k|H_l^k = 1, \theta_t)}{\sum_{i=1}^{L} p(H_i^k = 1|\theta_t)p(d_k|H_i^k = 1, \theta_t)} = \frac{p(c_l)p(d_k|c_l, \theta_t)}{\sum_{i=1}^{L} p(c_i)p(d_k|c_i, \theta_t)}$$

In the following, we use the notations: $p(c_l) = \pi_l$, $\gamma_l^k = p(H_l^k = 1|d_k, \theta_t)$ and $p_i^l = p(a_i|c_l)$.

The Lagrangian of the objective $Q_D(\theta|\theta_t)$ with normalization constraints can be written as:

$$L(\theta) = \sum_{k=1}^{K} \sum_{l=1}^{L} \gamma_l^k \cdot \log\left[p(d_k|c_l,\theta) \cdot \pi_l\right] - \lambda_0\left(\sum_{l=1}^{L}(\pi_l - 1)\right)$$

$$-\lambda_1\left(\sum_{i=1}^{I}(p_i^1 - 1)\right) - \ldots - \lambda_L\left(\sum_{i=1}^{I}(p_i^L - 1)\right) \tag{A.2}$$

We have (see formula 5.2): $p(d_k|c_l,\theta_t) = \left[\prod_{j=1}^{|V^k|} p(v_j^k|c_l)\right] \cdot \left[\prod_{i=1}^{|A^k|} p(a_i^k|c_l)\right]$ and $p(v_j^k|c_l) = \mathcal{N}(v_j^k; \mu_l, \Sigma_l) = \frac{1}{(2\pi)^{d/2}|\Sigma_l|^{1/2}} \exp\left(-\frac{1}{2}(v_j^k - \mu_l)^T \Sigma_l^{-1}(v_j^k - \mu_l)\right)$. Thus:

$$L(\theta) = \left(\sum_{k=1}^{K} \sum_{l=1}^{L} \gamma_l^k \sum_{j=1}^{|V^k|}\left[-\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln(|\Sigma_l|) - \frac{1}{2}(v_j^k - \mu_l)^T \Sigma_l^{-1}(v_j^k - \mu_l)\right.\right.$$

$$\left.\left.-\ln(p_{ik}^l) + \ln(\pi_l)\right]\right) - \lambda_0\left(\sum_{l=1}^{L}(\pi_l - 1)\right) - \lambda_1\left(\sum_{i=1}^{I}(p_i^1 - 1)\right) - \ldots - \lambda_L\left(\sum_{i=1}^{I}(p_i^L - 1)\right) \tag{A.3}$$

To obtain the new estimate θ_{t+1} of the model parameters, we look for the point θ such as $\frac{L(\theta)}{\partial\theta} = 0$. Thus, for the mean, we have:

$$\frac{\partial L(\theta)}{\partial\mu_l} = \sum_{k=1}^{K} \gamma_l^k \sum_{j=1}^{|V^k|} -\frac{\partial}{\partial\mu_l}\left(\frac{1}{2}(v_j^k - \mu_l)^T \Sigma_l^{-1}(v_j^k - \mu_l)\right)$$

$$= \sum_{k=1}^{K} \sum_{j=1}^{|V^k|} \gamma_l^k\left(\Sigma_l^{-1}(v_j^k - \mu_l)\right) = 0 \tag{A.4}$$

A.4 $\Longrightarrow \sum_{k=1}^{K} \sum_{j=1}^{|V^k|} \gamma_l^k \Sigma_l^{-1} v_j^k = \sum_{k=1}^{K} \sum_{j=1}^{|V^k|} \gamma_l^k \Sigma_l^{-1} \mu_l$

$$\Longrightarrow \mu_l = \frac{\sum_{k=1}^{K} \gamma_l^k \sum_{j=1}^{|V^k|} v_j^k}{\sum_{k=1}^{K} \gamma_l^k |V^k|} \tag{A.5}$$

Taking the derivative with respect to the covariance matrices yields:

$$\frac{\partial L(\theta)}{\partial\Sigma_l} = \sum_{k=1}^{K} \gamma_l^k \sum_{j=1}^{|V^k|} -\frac{\partial}{\partial\Sigma_l}\left(\frac{1}{2}\ln(|\Sigma_l|) - \frac{1}{2}(v_j^k - \mu_l)^T \Sigma_l^{-1}(v_j^k - \mu_l)\right)$$

$$= \sum_{k=1}^{K} \sum_{j=1}^{|V^k|} \gamma_l^k\left(-\frac{1}{2}\Sigma_l^{-T} + \frac{1}{2}\Sigma_l^{-T}(v_j^k - \mu_l)(v_j^k - \mu_l)^T \Sigma_l^{-T}\right) = 0 \tag{A.6}$$

A.6 $\Longrightarrow \sum_{k=1}^{K} \sum_{j=1}^{|V^k|} \gamma_l^k \Sigma_l^{-1} = \sum_{k=1}^{K} \sum_{j=1}^{|V^k|} \gamma_l^k \Sigma_l^{-1}(v_j^k - \mu_l)(v_j^k - \mu_l)^T \Sigma_l^{-1}$

$$\Longrightarrow \Sigma_l = \frac{\sum_{k=1}^{K} \gamma_l^k \sum_{j=1}^{|V^k|}(v_j^k - \mu_l)(v_j^k - \mu_l)^T}{\sum_{k=1}^{K} \gamma_l^k |V^k|} \tag{A.7}$$

For π_l, we obtain:

$$\frac{\partial L(\theta)}{\partial \pi_l} = \left(\sum_{k=1}^{K} \gamma_l^k \sum_{j=1}^{|V^k|} \frac{\partial \ln \pi_l}{\partial \pi_l} \right) - \lambda_0 \frac{\partial \pi_l}{\partial \pi_l}$$

$$= \sum_{k=1}^{K} \sum_{j=1}^{|V^k|} \gamma_l^k \left(\frac{1}{\pi_l} \right) - \lambda_0 = 0 \qquad (A.8)$$

A.8 $\implies \pi_l = \frac{1}{\lambda_0} \sum_{k=1}^{K} \sum_{j=1}^{|V^k|} \gamma_l^k = \frac{1}{\lambda_0} \sum_{k=1}^{K} \gamma_l^k |V^k|$. Using the normalization constraint $\sum_{i=1}^{L} \pi_i = 1$, we obtain: $\lambda_0 = \sum_{i=1}^{L} \sum_{k=1}^{K} \gamma_i^k |V^k|$ and finally:

$$\pi_l = \frac{\sum_{k=1}^{K} \gamma_l^k |V^k|}{\sum_{i=1}^{L} \sum_{k=1}^{K} \gamma_i^k |V^k|} \qquad (A.9)$$

And last, for p_i^l:

$$\frac{\partial L(\theta)}{\partial p_i^l} = \sum_{k=1}^{K} \gamma_l^k \sum_{j=1}^{|V^k|} \frac{\partial \ln p_{ik}^l}{\partial p_i^l} - \lambda_l \left(\sum_{i=1}^{I} \frac{\partial p_i^l}{\partial p_i^l} \right)$$

$$= \left(\sum_{k=1}^{K} \gamma_l^k \sum_{j=1}^{|V^k|} \delta_{a_i}^{a_j^k} \frac{1}{p_i^l} \right) - \lambda_l = 0 \qquad (A.10)$$

A.10 $\implies p_i^l = \frac{1}{\lambda_l} \sum_{k=1}^{K} \left[\gamma_l^k \sum_{j=1}^{|V^k|} \delta_{a_i}^{a_j^k} \right] = \frac{1}{\lambda_l} \sum_{k=1}^{K} \gamma_l^k N_{a_i}^k$. Using the normalization constraint $\sum_{j=1}^{I} p_j^l = 1$, we obtain: $\lambda_l = \sum_{j=1}^{I} \sum_{k=1}^{K} \gamma_l^k N_{a_j}^k$ and finally:

$$p_i^l = \frac{\sum_{k=1}^{K} \gamma_l^k N_{a_i}^k}{\sum_{j=1}^{I} \sum_{k=1}^{K} \gamma_l^k N_{a_j}^k} \qquad (A.11)$$

A.2   Case 2: explicit associations between feature vectors and "atomic" concepts 5.2.2)

In this case, the conditional expected log-likelihood writes itself as:

$$Q_D(\theta|\theta_t) = \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \sum_{l=1}^{L} \gamma_{ijl}^k(t) \cdot \log \left[p(a_i^k, v_j^k | c_l, \theta) \cdot \pi_k \right] \qquad (A.12)$$

where $\gamma_{ijl}^k(t) = p(H_{ijl}^k = 1 | a_i^k, v_j^k, \theta_t)$. We use a Lagrangian function of the same form as the one in A.14:

$$L(\theta) = \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \sum_{l=1}^{L} \gamma_{ijl}^k(t) \cdot \log\left[p(a_i^k, v_j^k|c_l, \theta) \cdot \pi_k\right]$$

$$-\lambda_0\left(\sum_{l=1}^{L}(\pi_l - 1)\right) - \lambda_1\left(\sum_{i=1}^{I}(p_i^1 - 1)\right) - \ldots - \lambda_L\left(\sum_{i=1}^{I}(p_i^L - 1)\right) \tag{A.13}$$

By expanding the expression above, we obtain:

$$L(\theta) = \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \sum_{l=1}^{L} \gamma_{ijl}^k(t) \cdot \log\left[p(a_i^k|c_l, \theta) \cdot p(v_j^k|c_l, \theta) \cdot \pi_k\right]$$

$$= \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \sum_{l=1}^{L} \gamma_{ijl}^k(t) \cdot \left[\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln(|\Sigma_l|) - \frac{1}{2}(v_j^k - \mu_l)^T\Sigma_l^{-1}(v_j^k - \mu_l)\right.$$

$$\left. - \ln(p_{ik}^l) + \ln(\pi_l)\right] \tag{A.14}$$

Thus, we obtain:

$$\frac{\partial L(\theta)}{\partial \mu_l} = \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k(t) \frac{\partial}{\partial \mu_l}\left(-\frac{1}{2}(v_j^k - \mu_l)^T\Sigma_l^{-1}(v_j^k - \mu_l)\right)$$

$$= \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k(t)\left(\Sigma_l^{-1}(v_j^k - \mu_l)\right) = 0 \tag{A.15}$$

$$\text{A.15} \Longrightarrow \sum_{k=1}^{K}\sum_{(a_i^k,v_j^k)} \gamma_{ijl}^k(t)\Sigma_l^{-1}v_j^k = \sum_{k=1}^{K}\sum_{(a_i^k,v_j^k)} \gamma_{ijl}^k(t)\Sigma_l^{-1}\mu_l$$

$$\Longrightarrow \mu_l = \frac{\sum_{k=1}^{K}\sum_{(a_i^k,v_j^k)} \gamma_{ijl}^k(t)v_j^k}{\sum_{k=1}^{K}\sum_{(a_i^k,v_j^k)} \gamma_{ijl}^k(t)} \tag{A.16}$$

$$\frac{\partial L(\theta)}{\partial \Sigma_l} = \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k(t) \frac{\partial}{\partial \Sigma_l}\left(-\frac{1}{2}\ln(|\Sigma_l|) - \frac{1}{2}(v_j^k - \mu_l)^T\Sigma_l^{-1}(v_j^k - \mu_l)\right)$$

$$= \sum_{k=1}^{K} \sum_{(a_i^k, v_j^k)} \gamma_{ijl}^k(t)\left(-\frac{1}{2}\Sigma_l^{-T} + \frac{1}{2}\Sigma_l^{-T}(v_j^k - \mu_l)(v_j^k - \mu_l)^T\Sigma_l^{-T}\right) = 0 \tag{A.17}$$

$$\text{A.17} \Longrightarrow \sum_{k=1}^{K}\sum_{(a_i^k,v_j^k)} \gamma_{ijl}^k(t)\Sigma_l^{-1} = \sum_{k=1}^{K}\sum_{(a_i^k,v_j^k)} \gamma_{ijl}^k(t)\Sigma_l^{-1}(v_j^k - \mu_l)(v_j^k - \mu_l)^T\Sigma_l^{-1}$$

$$\Longrightarrow \Sigma_l = \frac{\sum_{k=1}^{K}\sum_{(a_i^k,v_j^k)} \gamma_{ijl}^k(t)(v_j^k - \mu_l)(v_j^k - \mu_l)^T}{\sum_{k=1}^{K}\sum_{(a_i^k,v_j^k)} \gamma_{ijl}^k(t)} \tag{A.18}$$

Appendix B

Appendices of chapter 5: A Deterministic Annealing Approach for Learning Finite Mixture Model Parameters

In this appendix, we discuss an extension of the mass-constrained clustering algorithm ([Rose et al., 1993]) to perform maximum likelihood estimation of the parameters of finite mixture models. As argued in section 4.2, the DAEM algorithm [Ueda and Nakano, 1998] is indeed not the better way to apply annealing in the case of finite mixture models.

Besides the description of the algorithm itself, we also point out that it provides a natural framework for estimating the number of mixture components which is one of the most challenging task in unsupervised/supervised mixture modeling. We conclude this appendix by showing how the proposed algorithm can be reformulated to estimate the parameters of the hierarchical Bayesian model presented in section 5.2.1.

B.1 Modification of the mass-constrained algorithm to perform ML-estimation

In this section, we consider the problem of estimating the parameters of a mixture of Gaussians: $p(v) = \sum_{k=1}^{K} \alpha_k \cdot p(v|c_k) = \sum_{k=1}^{K} \alpha_k \cdot \mathcal{N}(v; \mu_k, \Sigma_k)$. We propose in the following a modification of the mass-constrained algorithm to perform maximum likelihood estimation of the model parameters. It has been mentioned in the paragraph 4.2 that the DAEM and the mass-constrained algorithm seek to minimize objective functions of the same form: $F = Dist - TH$ where $Dist$ is the total distortion, T a temperature parameter and H the entropy (see paragraph 4.2 for a definition of these terms in the case of the DAEM algorithm). All we have to do to perform ML-estimation is to modify the distortion measure used in the mass-constrained algorithm and add an update step for covariance matrices in the M-step of the inner loop (step 3) of this algorithm. The conditions for cluster splitting will also be different: for the squared error distance and in the case where just the cluster means are updated, it can be shown that the critical

temperatures are equal to twice the largest eigenvalue of the cluster covariance matrices. It means that a cluster c_j will be split when T becomes inferior to $2\lambda_{max}^j$ where λ_{max}^j is the largest eigenvalue of the covariance matrix $\Sigma_j = \frac{\sum_{i=1}^N p(c_k|v_i)(v_i-\mu_k)(v_i-\mu_k)^T}{N \cdot \alpha_k}$. This result can be obtained by looking at the hessian of $F^*(\theta)$: when the hessian is no longer positive definite, it means that the current estimate of model parameters θ is no longer a global minimum of the free energy for the new temperature. To find critical temperatures, we thus have to find the upper limit T_c where the hessian is still non negative. The condition we obtain for the squared error distance is the one mentioned above. In the following, we continue to note d_{ML} the distance we use for ML-estimation: $d_{ML}(v_i, c_k) = -\log(\alpha_k p(v_i|\theta_k))$. If we use this distance in the mass-constrained algorithm along with an update step for covariance matrices, it becomes much more difficult to derive critical temperatures for cluster splitting: the hessian will indeed not only depend on the cluster means but also on the covariance matrices, which makes it very difficult to compute (we would have to compute the following second derivatives: $\frac{\partial^2 F^*}{\partial \mu_i \partial \mu_j}$, $\frac{\partial^2 F^*}{\partial \mu_i \partial \Sigma_j}$ and $\frac{\partial^2 F^*}{\partial \Sigma_i \partial \Sigma_j}$). Even if we succeed in computing the hessian, there is still the problem of extracting a splitting condition per cluster: this implies writing H as a sum of two symmetric matrices, $H_1 + H_2$, where H_1 is a block diagonal matrix with each block corresponding to the sub-hessian associated with one cluster, and proving that the positive definiteness of H is equivalent to the positive definiteness of H_1. Instead of trying to extract an analytical condition by looking at the hessian, we can think of a very simple heuristic criterion which has proved to work very well in practice: we maintain permanently two centroids per cluster. When we decrease the temperature, we add a small perturbation to the centroid of one of the two clusters. When the critical temperature is reached, that is, when cluster splits occur, the two centroids will simply drift apart. On the contrary, between phase transitions, the two centroids will be stuck together again after the convergence of the inner loop. In the following, a cluster c_k is a couple $\{c_k^1, c_k^2\}$ where c_k^1 is the true cluster and c_k^2 its perturbed version. We denote by $p(z_i^{k,l} = 1|v_i, \theta)$ the probability that the element v_i originates from the mixture component c_k^l ($l \in \{1,2\}$). We obtain the following algorithm:

1. Set maximum number of clusters K_{max} and minimum temperature T_{min}.

2. Choose $T > 2\lambda_{max}$, $K = 1$, $\mu_1^1 = \mu_1^2 = \frac{1}{N}\sum_{i=1}^N v_i$, $\alpha_1^1 = \alpha_1^2 = 1/2$, $\Sigma_1^1 = \Sigma_1^2 = \Sigma_{init} = \frac{1}{N}\sum_{i=1}^N (v_i - \mu_1^1)(v_i - \mu_1^1)^T$.

3. For $k = 1, \cdots, K$, perturb cluster c_k^2: $\{\alpha_k^2, \mu_k^2, \Sigma_k^2\} = \{\alpha_k^1, \mu_k^1 + \delta_k, \Sigma_k^1\}$.

4. Iterate the following two steps until convergence criterion is met:

   - **E-step:** Compute association probabilities
     $$p(z_i^{k,l} = 1|v_i, \theta) = \frac{(\alpha_k^l p(v_i|\theta_k^l))^\beta}{\sum_{l=1}^K \left[(\alpha_l^1 p(v_i|\theta_i^1))^\beta + (\alpha_l^2 p(v_i|\theta_i^2))^\beta\right]}$$

   - **M-step:** Update mixing proportions, means and covariance matrices according to: $\alpha_k^l = \frac{\sum_{i=1}^N p(z_i^{k,l}=1|v_i,\theta)}{N}$, $\mu_k^l = \frac{\sum_{i=1}^N p(z_i^{k,l}=1|v_i,\theta)v_i}{\sum_{i=1}^N p(z_i^{k,l}=1|v_i,\theta)}$

$$\text{and } \Sigma_k^l = \frac{\sum_{i=1}^N p(z_i^{k,l}=1|v_i,\theta)(v_i-\mu_k^l)(v_i-\mu_k^l)^T}{\sum_{i=1}^N p(z_i^{k,l}=1|v_i,\theta)}$$

5. If $T < T_{min}$, stop.

6. Cooling step: decrease T.

7. If $K < K_{max}$, check condition for phase transition (i.e cluster splitting) for $k = 1, \cdots, K$. If condition is met for cluster j, add a new cluster $c_{K+1} = \{c_{K+1}^1, c_{K+1}^2\}$. Set $\mu_{K+1}^1 = \mu_{K+1}^2 = \mu_j^2$, $\Sigma_{K+1}^1 = \Sigma_{K+1}^2 = \Sigma_j^2$, $\alpha_{K+1}^1 = \alpha_{K+1}^2 = \alpha_j^2/2$ and $\alpha_j^1 = \alpha_j^1/2$. Increment K. Go to step 3.

K refers to the number of effective clusters. For each of the K effective clusters, we denote $\{\alpha_k^1, \mu_k^1, \Sigma_k^1\}$ the parameters of c_k^1 and $\{\alpha_k^2, \mu_k^2, \Sigma_k^2\}$ the parameters of the perturbed version c_k^2. Checking condition for phase transition consists here in looking if a cluster and its perturbed version have moved apart or are still stuck together after convergence of the inner loop. We could simply use an Euclidean distance between cluster centers but the change in scale during the annealing process makes it difficult to keep a static threshold on such a measure: very large clusters such as can be seen at the beginning of the annealing process can have centers quite far away from each other but can nevertheless have a strong overlapping. On the contrary, clusters with very localized influence such as can be seen at the end of the annealing process can have their centers very close from each other but still be well separated. We need consequently a notion of distance which includes the idea of separation (distinct clusters). In the case of GMMs, we use a Mahalanobis like distance between clusters: $d(c_i, c_j) = \frac{\alpha_i \alpha_j}{(\alpha_i+\alpha_j)^2}(\mu_i-\mu_j)^T P_{ij}^{-1}(\mu_i-\mu_j)$ with $P_{ij} = \frac{\alpha_i}{\alpha_i+\alpha_j}\Sigma_i + \frac{\alpha_j}{\alpha_i+\alpha_j}\Sigma_j + \frac{\alpha_i \alpha_j}{(\alpha_i+\alpha_j)^2}(\mu_i-\mu_j)(\mu_i-\mu_j)^T$. We can show easily that $d(c_i, c_j) = tr(P_{ij}^{-1}W_{ij})$ where $W_{ij} = \frac{\alpha_i \alpha_j}{(\alpha_i+\alpha_j)^2}(\mu_i-\mu_j)(\mu_i-\mu_j)^T$. W_{ij} corresponds to the "loss" in within-components covariance which results when we merge the two components of a two-component mixture. The multiplication by P_{ij}^{-1} acts like a normalization factor (we have thus $0 \le d(c_i, c_j) \le 1$) and makes the dissimilarity measure invariant under linear transformations. To decide wether two clusters have drifted apart or are still stuck together, we set a static threshold Ω on this measure. The splitting condition for cluster c_k can thus be written: $d(c_k^1, c_k^2) > \Omega$.

In the next section, we show that the modified version of the mass-constrained algorithm described above provides a natural framework for estimating the number of Gaussian mixture components, which is one of the most challenging task in clustering.

B.2   Estimation of the number of mixture components

Deterministic annealing has a strong connection to Rate-Distortion (RD) theory [Rose, 1994]. The aim of RD theory is to give theoretical bounds for how much compression can be achieved without exceeding a given distortion between the input and the output of a compression system. RD is usually used to characterize the performances of lossy source compression methods, the best method being the one which for a given distortion achieves the highest compression rate. It is quite common to introduce RD-related notions in clustering problems since clustering can be easily reinterpreted as

a lossy source compression problem. The mathematical formulation for the function which relates the rate to distortion is the following:

$$R(D_0) = \min_{p(y|x),\ D \leq D_0} I(X, Y) \tag{B.1}$$

where $D = \sum_x \sum_y p(x, y) d(x, y)$ is the achieved distortion, $I(X, Y)$ is the mutual information between input variable X and output variable Y. d is an arbitrary distortion measure (we can take for instance the squared error distance). Since $I(X, Y) = H(X) - H(X|Y)$ and $H(X)$ is constant, the above minimization problem can be rewritten as: $R(D_0) = \max_{p(y|x),\ D \leq D_0} H(X|Y)$. Introducing a Lagrange multiplier β, we can show that there exists a value of β, β_0, for which: $R(D_0) = \max_{p(y|x)} \left[H(X|Y) - \beta_0 \cdot (D - D_0) \right] = \max_{p(y|x)} \left[H(X|Y) - \beta_0 \cdot D \right]$. By noting $T_0 = \frac{1}{\beta_0}$, we have:

$$R(D_0) = \min_{p(y|x)} [D - T_0 \cdot H(X|Y)] \tag{B.2}$$

which is exactly the objective function we seek to minimize in the mass-constrained algorithm. It can be shown [Cover et al., 1991] that the parameter β is related to the slope of the rate-distortion curve $R(D)$ by the following expression: $\beta = \frac{1}{T} = -\frac{\partial R}{\partial D}$. So, when we perform annealing, we simply climb up the rate-distortion curve. The position on the curve is determined by the temperature parameter T which is the inverse of the slope β. This equivalence between the annealing process and the computation of the rate-distortion curve proves that the number of clusters which is found at each temperature is optimal on the point of view of rate-distortion theory: the increase in the number of effective clusters (during phase transitions) can indeed be seen as a way to maintain an optimal trade-off between rate and distortion. To stay on the rate-distortion curve as we increase the temperature, we have to compensate for the increase in rate by a decrease in distortion (the rate-distortion curve is monotonically decreasing), which is achieved by updating the mapping probabilities $p(y|x)$ that condition the effective number of clusters. Therefore, on condition that the maximum number of clusters K_{max} is not to low, the number of effective clusters found at the end of the annealing process is optimal.

B.3 ML-estimation of the parameters of a hierarchical Bayesian model

In this section, we test the performance of the algorithm described in section B.1 on learning the parameters of the auto-annotation model introduced in 5.2.1. This algorithm is indeed not limited to estimating the parameters of a finite Gaussian mixture: we can apply it to all kinds of finite mixture models. The only changes will be in the M-step which is the only part of the algorithm to be model-specific: the M-step is indeed where model parameters are estimated, so it is logical that the update equations we obtain in this part depend specifically on the parametric family associated with the finite mixture (there is still an exception regarding mixing proportions which remain model-independent [Ueda and Nakano, 1998]). As mentioned above, the E-step remains unchanged: the expression of the association probabilities $p(z_i^k = 1|v_i, \theta)$ can

indeed be derived without assuming a particular pdf (parametric family) over the mixture.

The DA approach is justified here by the fact that the pdf of documents given model parameters is that of a finite mixture:

$$p(d_k|\theta) = \sum_{l=1}^{L} p(d_k, c_l|\theta) = \sum_{l=1}^{L} p(c_l|\theta)p(d_k|c_l,\theta) = \sum_{l=1}^{L} \pi_l \cdot p(d_k|\theta_l) \tag{B.3}$$

So, all the results mentioned above can be applied to our problem. In particular, the modified DA mass-constrained algorithm we have presented in section 3 can be reused to perform maximum likelihood estimation of model parameters. We have seen in the introduction of this section that the only model-specific part of the algorithm is the M-step. An other slight modification has still to be done: documents d_k live indeed in a space which is partly continuous (feature space) and partly discrete (semantic concept space). So, in the perturbation step of our modified mass-constrained algorithm (step 3), we cannot define a perturbation on the centroid of a "cluster" of documents since here the notion of "cluster" is not well-defined (because of the existence of two-spaces of very different nature). Instead, we can think of exploiting the probabilistic clustering induced by the underlying pdf of data in the feature space, which is a finite Gaussian mixture. Thus, cluster splits will be provoked by perturbations in the feature space and the same condition as the one used for GMMs can be used to determine wether two clusters have drifted apart or are still stuck together. In particular, we can reuse the distance measure introduced in B.1. The algorithm we obtain is not very different from the one exposed in section B.1:

1. Set maximum number of clusters L_{max} and minimum temperature T_{min}.

2. Choose $T > 2\lambda_{max}$, $L = 1$, $\mu_1^1 = \mu_1^2 = \frac{\sum_{k=1}^{K}\sum_j v_j^k}{\sum_{k=1}^{K}|V_k|}$, $\pi_1^1 = \pi_1^2 = 1/2$, $\Sigma_1^1 = \Sigma_1^2 = \Sigma_{init} = \frac{\sum_{k=1}^{K}\sum_j (v_j^k-\mu_1^1)(v_j^k-\mu_1^1)^T}{\sum_{k=1}^{K}|V_k|}$.

3. For $l = 1, \cdots, L$, perturb cluster c_l^2:

$$\{\pi_l^2, \mu_l^2, \Sigma_l^2, \{p(a_i|c_l^2)\}_{i=1,\cdots,I}\} = \{\pi_l^1, \mu_l^1 + \delta_l, \Sigma_l^1, \{p(a_i|c_l^1)\}_{i=1,\cdots,I}\}.$$

4. Iterate the following two steps until convergence criterion is met:

   • **E-step:** Compute association probabilities

   $$p(h_k^{l,m} = 1|d_k,\theta) = \frac{(\pi_l^m p(d_k|\theta_l^m))^\beta}{\sum_{l=1}^{L}\left[(\pi_l^1 p(d_k|\theta_l^1))^\beta + (\pi_l^2 p(d_k|\theta_l^2))^\beta\right]}$$

   • **M-step:** Update mixing proportions, means and covariance matrices according to: $\pi_l^m = \frac{\sum_{k=1}^{K} p(h_k^{l,m}=1|d_k,\theta)}{K}$, $\mu_l^m = \frac{\sum_{k=1}^{K}\left[p(h_k^{l,m}=1|d_k,\theta)\sum_j v_j^k\right]}{\sum_{k=1}^{K} p(h_k^{l,m}=1|d_k,\theta)\cdot|V_k|}$,

   $\Sigma_l^m = \frac{\sum_{k=1}^{K}\left[p(h_k^{l,m}=1|d_k,\theta)\sum_j (v_j^k-\mu_l^m)(v_j^k-\mu_l^m)^T\right]}{\sum_{k=1}^{K} p(h_k^{l,m}=1|d_k,\theta)\cdot|V_k|}$

   and $p(a_i|c_l^m) = \frac{\sum_{k=1}^{K} p(h_k^{l,m}=1|d_k,\theta)\cdot N_{a_i}^k}{\sum_{q=1}^{I}\sum_{k=1}^{K} p(h_k^{l,m}=1|d_k,\theta)\cdot N_{a_q}^k}$

5. If $T < T_{min}$, stop.

6. Cooling step: decrease T.

7. If $L < L_{max}$, check condition for phase transition (i.e cluster splitting) for $l = 1, \cdots, L$. If condition is met for cluster j, add a new cluster $c_{L+1} = \{c_{L+1}^1, c_{L+1}^2\}$. Set $\mu_{L+1}^1 = \mu_{L+1}^2 = \mu_j^2$, $\Sigma_{L+1}^1 = \Sigma_{L+1}^2 = \Sigma_j^2$, $\pi_{L+1}^1 = \pi_{L+1}^2 = \pi_j^2/2$, $\pi_j^1 = \pi_j^1/2$ and $\{p(a_i|c_{L+1}^1)\}_{i=1,\cdots,I} = \{p(a_i|c_{L+1}^2)\}_{i=1,\cdots,I} = \{p(a_i|c_j^2)\}_{i=1,\cdots,I}$. Increment L. Go to step 3.

Appendix C

Appendices of chapter 6

C.1   Proof of convergence of the linear component-based SVM

Proof of convergence in norm of w**:** We show that the margin (i.e. the ratio $1/\|w\|$) decreases at each iteration of the alternating optimization scheme. For this, it suffices to note that the training set at iteration t, A_t is included in the training set at iteration $t+1$, A_{t+1}. The parameter C of SVM remaining unchanged from one iteration to another, the margin at iteration $t+1$ is necessarily smaller (or equal) than the margin at iteration t. The sequence $1/\|w_t\|$ being decreasing and lower-bounded (linearly separable case), it converges to a value $1/\|w\|^*$. One can also prove that this value is reached (and therefore that it is a minimum) but we will not do it here for lack of space. Heuristically, the choice of critical points to increase the training set can be explained by the fact that before convergence, critical points are systematically located within the margin of the current classifier. The margin of the classifier obtained at the next iteration is therefore smaller since we add the new critical points to the training set that is used to learn this classifier.

   Proof of convergence of $C_t = (w_t, b_t)$**:** We denote by C_t the classifier obtained with the training set A_t and P_{C_t} the set of critical points computed from C_t. We have of course: $A_{t+1} = A_t \cup P_{C_t}$. We can start by observing that if there is a classifier C_{t_f} such that the points $P_{C_{t_f}}$ are all outside (in a non-strict sense) the margin of C_{t_f}, then $C_{t_f+1} = C_{t_f}$, which ensures the convergence of the alternating scheme. It remains to show that such a classifier exists. Let's suppose this is not the case. Let t_f be the iteration for which $1/\|w_{t_f}\| = 1/\|w\|^*$. Because of the hypothesis we made above, at least one of the points of the set $P_{C_{t_f}}$ is necessarily within (in a strict sense) the margin of the classifier C_{t_f}. The classifier C_{t_f+1} trained with the set $A_{t_f+1} = A_{t_f} \cup P_{C_{t_f}}$ then necessarily possesses a margin which is strictly smaller than $1/\|w\|^*$ [1] which contradicts the fact that the sequence $1/\|w_t\|$ has converged and reached its minimum at iteration t_f. This completes the proof that the sequence $C_t = (w_t, b_t)$ is convergent.

---

[1]If it had the same margin, it would mean that there is a classifier other than C_{t_f} which also separates the points of A_{t_f} with a margin $1/\|w\|^*$, which contradicts the fact that the SVM problem has a unique global minimum

We can also prove that the obtained optimum (w^*, b^*) is global and unique. Indeed, given two classifiers C^1 and C^2 such that the points of P_{C^1} and P_{C^2} are all located outside the respective margins of C^1 and C^2 (see definition of convergence above), then the classifier trained with the set $B_{12} = A^1 \cup A^2$ is equal to C^1. Indeed, by definition, the points of P_{C^1} being the closest to C^1 and being still outside the margin of C^1, the points of A^2 a fortiori are also located outside the margin of C^1 and therefore will not have any influence during the learning if we add them to the set A^1. Using a similar reasoning, we can show that the classifier trained with the set $B_{21} = A^2 \cup A^1$ is equal to C^2. Since $B_{12} = B_{21}$, we then have $C^1 = C^2$, which proves that the obtained optimum is global and unique.

C.2   Determination of the critical points

The critical point associated with a Gaussian mixture component is the point of the associated convex hull which is the closest to the separating hyperplane (w, b). In the case where the separating hyperplane intersects the convex hull, the critical point is the point of the convex hull which is the farthest from the separating hyperplane but on the other side of it (see Fig. C.1). For a component c whose convex hull is given by the equation $(v - \mu)^T \Sigma^{-1} (v - \mu) = $ constant and whose label is y_c, the critical point associated with the component is the solution of the problem: $\min_v y_c \cdot (w \cdot \phi(v) + b)$ under the constraint $(v - \mu)^T \Sigma^{-1} (v - \mu) = $ constant.
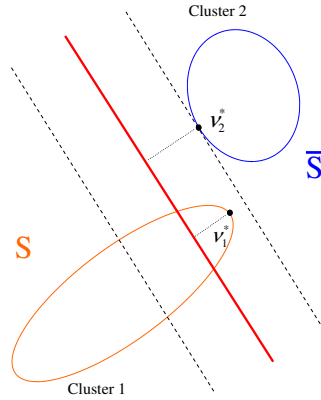
Figure C.1: Determination of the critical points.

linear separator: we show that the critical point v_l^* associated with the l-th mixture component is given by:

$$v_l^* = \mu_l - (2\delta_l^S - 1)\sqrt{\rho_1^l} \cdot \frac{\Sigma_l \cdot w}{\sqrt{w^T \Sigma_l w}} \tag{C.1}$$

nonlinear separator (using a Gaussian kernel): given an approximation of the kernel with a Taylor expansion of order 1, we show that the critical point v_l^* is given by the formula:

$$v_l^* = \mu_l - (2\delta_l^S - 1)\sqrt{\rho_1^l} \cdot \frac{\Sigma_l \cdot u}{\sqrt{u^T \Sigma_l u}} \tag{C.2}$$

with $u = \sum_{j=1}^{L} \alpha_j y_{\gamma_j} \nabla_v k(v, \gamma_j)|_{v=\mu_l}$ where the α_j are the Lagrange multipliers in the dual problem whose solution yields the classifier (w, b). The γ_j are the points that were used to train the classifier. $k(.,.)$ refers to the Gaussian kernel.

C.3 Sub-programs used by the Algorithm 5

Algorithm 7 Sample$_{\text{Gibbs}}$

%Sample K components according to a Gibbs distribution with temperature parameter T

%D is an array containing the distances of non-tagged components to the current SVM separating surface

Input D, K, T

$W = \{w_1, \ldots, w_{\text{Card}(D)}\}$ with $w_i = \frac{1}{Z_C} \cdot \exp(-\frac{d_{max} - D(i)}{T})$ where $d_{max} = \max_i D(i)$ and $Z_C = \sum_i D(i)$

$\{i_1, \ldots, i_K\} \sim \mathcal{U}(w'_1, \ldots, w'_{\text{Card}(D)})$ %sample K indices according to the normalized importance weights $\left\{w'_1, \ldots, w'_{\text{Card}(D)}\right\}$

Algorithm 8 ComputeComponentMembership

%Compute the membership degree τ_l of a component c_l depending on the current SVM model f. TS is the training set used to train f

Input c_l, TS, f

If $c_l \in \bar{S}$ **then**

   $\tau_l = 1$

Else

   $(a, b) = \text{Platt}(TS, f)$ %compute sigmoid parameters using Platt's algorithm

   $\tau_l = \frac{1}{\sum_{v \in \mathcal{V}} \mathcal{N}(v; \mu_l, \Sigma_l)} \sum_{v \in \mathcal{V}} \frac{1}{1 + \exp(-a \cdot f(v) + b)} \cdot \mathcal{N}(v; \mu_l, \Sigma_l)$

End If

In a real implementation of the algorithm 8, we do not perform a summation over all the elements $v \in \mathcal{V}$ but on the elements v such that $\mathcal{N}(v; \mu_l, \Sigma_l) > \mathcal{N}(\gamma(3); \mu_l, \Sigma_l)$ (see Sec. 6.4.2.1 for an explanation about the function γ).

We have to notice here that the positively-labeled points in the set A_0 are all situated within the current convex hulls of positively-labeled mixture components. So it is possible to assign to them the membership degrees associated with the component corresponding to the cluster they belong to.

Algorithm 9 ComponentBasedFuzzySVM

%Solve the fuzzy component-based SVM problem on the set of labeled clusters $Clust$ using component membership degrees Mb and initializing the algorithm with the set of labeled points A_0

Input $Clust, Mb, A_0$

Using a probabilistic SVM, train C_0 with A_0 and memberships contained in Mb

While $A_{t+1} \neq A_t$ **do**

Determine the critical points $v_1^*, \ldots, v_{\mathrm{Card}(Clust)}^*$ associated with the current classifier C_t

Using a probabilistic SVM, train C_{t+1} with the training set A_{t+1} and the memberships Mb, where A_{t+1} is built from A_t by adding the points v_i^* such that $\min_{v_j \in A_t} \|v_i^* - v_j\| > \epsilon$

End While

Appendix D

Appendices of chapter 7

D.1 Algorithmic description of the overall CALOR process

In the algorithm 10, P_f^l and N_f^l refer respectively to the set of positive and negative examples which are used to train the current classifier at the level l of the hierarchy. The user feedback at the i-th iteration of the active learning loop at level l is denoted by $\{pf_i^l, nf_i^l\}$, pf_i^l being the set of elements on which the user feedback is positive and nf_i^l the set of elements on which the user feedback is negative. The membership degrees associated with the elements of the set P_f^l are referred to as μ_P^l. The values $\{q_l\}_{l=1,\dots,L}$ are the number of iterations in the active learning loop at each stage of the hierarchy. The function $\text{Inter}(n, x)$ computes the degree of intersection between the patches n and x as the percentage of the area of the patch n which is covered by the patch x. $thresh$ is a threshold which is fixed to 0.5 (50 %) for all stages of the hierarchy. The thresholds ζ_l are calculated so that the ratio $\frac{\left|\left\{v_l^{i1}, v_l^{i2}, \dots \in T_l \,\middle|\, f(v_l^{i1}) \geq f(v_l^{i2}) \geq \dots \geq \zeta_l\right\}\right|}{|T_l|}$ respects the proportions fixed above for the percentage of top-ranked elements we keep when switching to the level below in the hierarchy. The other notations are introduced in the chapter 7.

Algorithm 10 Algorithmic description of the CALOR procedure

Input The user is asked to point a patch $v_L^{i_0}$ of size t_L centered on an example of the targeted object

Initialization

- Set $T_L = E_L$, $P_f^L = \left\{ v_L^{i_0} \right\}$ and $\mu_P^L = \{1\}$
- Select D random patches of size t_L inside the set T_L to display to the user

%Coarse-to-fine loop

For $l = L, L-1, \dots, 1$ **do**

  %Active learning loop to build the classifier C_l

  For $i = 1, \dots, q_l$ **do**

    • Update P_f^l and N_f^l with the user feedback $\{pf_i^l, nf_i^l\}$:

    $P_f^l \leftarrow P_f^l \cup pf_i^l$ and $N_f^l \leftarrow N_f^l \cup nf_i^l$

    • Update μ_P^l: $\mu_P^l \leftarrow \mu_P^l \cup \mathbb{1}_{|pf_i^l|}$

    • $f_l^i \leftarrow \text{PROB\_SVM}\left(\left\{ P_f^L \cup N_f^l , \mu_P^l \cup \mathbb{1}_{|N_f^l|} , \mathbb{1}_{|P_f^l|} \cup -\mathbb{1}_{|N_f^l|} \right\} \right)$

    • Display D examples $\{i_1, \dots, i_D\}$ to the user using the strategy proposed in 7.2.2: $\{i_1, \dots, i_D\} = \text{argmax}_{i_1, \dots, i_D \in S_l} \min_{\substack{(j_1, j_2) \in \{i_1, \dots, i_D\} \\ \text{with } j_1 < j_2}} d(v_{j_1}, v_{j_2})$ and $S_l =$

  $\left\{ v_l^i \in T_l \backslash \left\{ P_f^l \cup N_f^l \right\} \,\middle|\, f_l^i(v_l^i) < \epsilon \right\}$

  End For

  • Set $T_{l-1} = \{ n \in E_l \,|\, \exists x \in T_l \text{ such as } n \cap x \neq \emptyset \text{ and } f_l^{q_l}(x) > \zeta_l \}$

  • Set $\left\{ P_f^{l-1}, \mu_P^{l-1} \right\} = \text{IMSVM}(P_f^l, \mu_P^l)$ and $N_f^{l-1} = \left\{ n \in E_{l-1} | \exists x \in N_f^l \text{ such as } \text{Inter}(n, x) \geq thresh \right\}$

End For

• Return T_1 and $f_1^{q_1}$

Bibliography

Y. Abramson and Y. Freund. Active learning for visual object detection. *Univ. California San Diego, San Diego, CA, CS2006-0871 Tech. Rep., Nov*, 19, 2006.

S. Aksoy, R.M. Haralick, F.A. Cheikh, and M. Gabbouj. A weighted distance approach to relevance feedback. In *icpr*, page 4812. Published by the IEEE Computer Society, 2000.

S. Aksoy, K. Koperski, C. Tusk, G. Marchisio, and J.C. Tilton. Learning Bayesian classifiers for scene classification with a visual grammar. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):581, 2005.

S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. *Advances in neural information processing systems*, pages 577–584, 2003. ISSN 1049-5258.

A. Baraldi, V. Puzzolo, P. Blonda, L. Bruzzone, and C. Tarantino. Automatic spectral rule-based preliminary mapping of calibrated Landsat TM and ETM+ images. *Geoscience and Remote Sensing, IEEE Transactions on*, 44(9):2563–2586, 2006. ISSN 0196-2892.

K. Barnard, P. Duygulu, D. Forsyth, N. De Freitas, D.M. Blei, and M.I. Jordan. Matching words and pictures. *The Journal of Machine Learning Research*, 3:1107–1135, 2003. ISSN 1532-4435.

A. Ben-Hur, D. Horn, H.T. Siegelmann, and V. Vapnik. Support vector clustering. *The Journal of Machine Learning Research*, 2:125–137, 2002. ISSN 1532-4435.

J.C. Bezdek, R. Ehrlich, et al. FCM: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984. ISSN 0098-3004.

S. Bhatia, A. Samal, and P. Vadlamani. RISE-SIMR: a robust image search engine for satellite image matching and retrieval. *Advances in Visual Computing*, pages 245–254, 2007.

P. Blanchart and M. Datcu. A Semi-Supervised Algorithm for Auto-Annotation and Unknown Structures Discovery in Satellite Image Databases. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 3(4):698–717, 2010. ISSN 1939-1404.

P. Blanchart and M. Datcu. Semi-supervised learning and discovery of unkown structures among data: Application to satellite image annotation. In *Geoscience and Remote Sensing Symposium, 2009 IEEE International, IGARSS 2009*, volume 3, pages III–777. IEEE, 2009.

P. Blanchart, M. Ferecatu, and M. Datcu. Apprentissage actif et utilisation de la structure a priori des données : application à une base d'images satellites haute résolution. In *Proceedings of RFIA*, 2010.

P. Blanchart, M. Ferecatu, and M. Datcu. Cascaded Active Learning for Object Retrieval using Multiscale Coarse-to-fine Analysis. In *Image Processing, 2011. ICIP 2011. IEEE Computer Society Conference on*. IEEE, 2011a.

P. Blanchart, M. Ferecatu, and M. Datcu. Mining Large Satellite Image Repositories using Semi-supervised Methods. In *Geoscience and Remote Sensing Symposium, 2011 IEEE International, IGARSS 2011*. IEEE, 2011b.

P. Blanchart, M. Ferecatu, and M. Datcu. Active Learning Using the Data Distribution for Interactive Image Classification and Retrieval. In *2011 IEEE Symposium on Computer Intelligence and Data Mining*. IEEE, 2011c.

P. Blanchart, M. Ferecatu, and M. Datcu. Indexation of Large Satellite Image Repositories Using Small Training Sets. In *ESA-EUSC-JRC Seventh Conference on Image Information Mining: Geospatial Intelligence from Earth Observation*. JRC, 2011d.

D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003. ISSN 1532-4435.

D. Boley and D. Cao. Training support vector machine using adaptive clustering. In *Proceedings of the Fourth SIAM International Conference on Data Mining, MW Berry, U. Dayal, C. Kamath, and D. Skillicorn, eds., SIAM Press, Philadelpha*, pages 126–137. Citeseer, 2004.

B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

N. Boujemaa, J. Fauqueur, M. Ferecatu, F. Fleuret, V. Gouet, B. LeSaux, and H. Sahbi. Ikona for interactive specific and generic image retrieval. In *Proceedings of International workshop on Multimedia Content-Based Indexing and Retrieval (MM-CBIR'2001)*. Citeseer, 2001.

L. Bruzzone, M. Chi, and M. Marconcini. A novel transductive SVM for semisupervised classification of remote-sensing images. *Geoscience and Remote Sensing, IEEE Transactions on*, 44(11):3363–3373, 2006. ISSN 0196-2892.

M. Campedel, B. Luo, M. Roux, and I. Kyrgyzov. Indexation of satellite Images. Technical report, Telecom Paristech, 2004.

M. Campedel, I. Kyrgyzov, and H. Maître. Unsupervised feature selection applied to spot5 satellite images indexing. *FSDM, Anvers (Belgique)*, 2008.

C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1026–1038, 2002. ISSN 0162-8828.

O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised learning*. Citeseer, 2006a.

O. Chapelle, J. Weston, B. Schölkopf, A. Elisseeff, V. Vapnik, C. Leslie, E. Eskin, et al. Cluster kernels for semi-supervised learning. In *7th Göttingen Meeting of the German Neuroscience Society*, volume 7, page 1. ISCB Student Council, 2006b.

O. Chapelle, V. Sindhwani, and S.S. Keerthi. Branch and bound for semi-supervised support vector machines. *Advances in neural information processing systems*, 19: 217, 2007.

J. Chen, S. Shan, C. He, G. Zhao, M. Pietikäinen, X. Chen, and W. Gao. WLD: A robust local image descriptor. *IEEE transactions on pattern analysis and machine intelligence*, pages 1705–1720, 2009. ISSN 0162-8828.

C. Chinrungrueng and C.H. Sequin. Optimal adaptive k-means algorithm with dynamic adjustment of learning rate. *Neural Networks, IEEE Transactions on*, 6(1): 157–169, 1995.

M. Costache, M. Datcu, and H. Maître. Categorization based relevance feedback search engine for earth observation images repositories. In *IEEE International Geoscience and Remote Sensing Symposium*, 2006.

T.M. Cover, J.A. Thomas, J. Wiley, et al. *Elements of information theory*, volume 6. Wiley Online Library, 1991.

I.J. Cox, M.L. Miller, T.P. Minka, T.V. Papathomas, and P.N. Yianilos. The Bayesian image retrieval system, PicHunter: theory, implementation, and psychophysical experiments. *Image Processing, IEEE Transactions on*, 9(1):20–37, 2000. ISSN 1057-7149.

G.R. Cross and A.K. Jain. Markov random field texture models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (1):25–39, 1983. ISSN 0162-8828.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893, 2005.

H. Daschiel and M. Datcu. Design and Evaluation of Human–Machine Communication for Image Information Mining. *IEEE Transactions on Multimedia*, 7(6), 2005.

M. Dash, K. Choi, P. Scheuermann, and H. Liu. Feature selection for clustering-a filter solution. In *Proceedings of the 2002 IEEE International Conference on Data Mining*. Published by the IEEE Computer Society, 2002.

M. Datcu, K. Seidel, and M. Walessa. Spatial information retrieval from remote-sensing images. I. Information theoretical perspective. *Geoscience and Remote Sensing, IEEE Transactions on*, 36(5):1431–1445, 1998. ISSN 0196-2892.

M. Datcu, K. Seidel, S. D'Elia, and PG Marchetti. Knowledge-driven information mining in remote-sensing image archives. *E. S. A. Bulletin*, pages 26–33, 2002. ISSN 0376-4265.

R. Datta, D. Joshi, J. Li, and J.Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (CSUR)*, 40(2):5, 2008.

A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977. ISSN 0035-9246.

Thomas Deselaers, Daniel Keysers, and Hermann Ney. Features for image retrieval: An experimental comparison. *Information Retrieval*, 11(2):77–107, 2008.

I.S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: spectral clustering and normalized cuts. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–556. ACM, 2004.

M.N. Do and M. Vetterli. Wavelet-based texture retrieval using generalized Gaussian density and Kullback-Leibler distance. *Image Processing, IEEE Transactions on*, 11 (2):146–158, 2002. ISSN 1057-7149.

J. Fan, Y. Gao, H. Luo, and R. Jain. Mining multilevel image semantics via hierarchical classification. *Multimedia, IEEE Transactions on*, 10(2):167–187, 2008. ISSN 1520-9210.

L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 524–531. Ieee, 2005. ISBN 0769523722.

M. Ferecatu. *Image retrieval with active relevance feedback using both visual and keyword-based descriptors*. PhD thesis, University of Versailles, Saint-Quentin-en-Yvelines (UVSQ), 2005.

M. Ferecatu and N. Boujemaa. Interactive remote-sensing image retrieval using active relevance feedback. *IEEE Transactions on Geoscience and Remote Sensing*, 45(4), 2007.

M. Ferecatu, N. Boujemaa, and M Crucianu. Semantic interactive image retrieval combining visual and conceptual content description. *ACM Multimedia Systems Journal*, 13(5-6):309–322, 2008.

R. Fletcher. Practical methods of optimization: Vol. 2: Constrained optimization. *JOHN WILEY & SONS, INC., ONE WILEY DR., SOMERSET, N. J. 08873, 1981, 224*, 1981.

F. Fleuret and D. Geman. Coarse-to-fine face detection. *International Journal of Computer Vision*, 41(1):85–107, 2001. ISSN 0920-5691.

M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, et al. Query by image and video content: The QBIC system. *Computer*, 28(9):23–32, 1995. ISSN 0018-9162.

Keinosuke Fukunaga. *Introduction to statistical pattern recognition*. Academic Pr, 1990. ISBN 0122698517.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images*. *Journal of Applied Statistics*, 20(5):25–62, 1993. ISSN 0266-4763.

A. Gersho. Asymptotically optimal block quantization. *Information Theory, IEEE Transactions on*, 25(4):373–380, 1979.

T. Gevers and A.W.M. Smeulders. Pictoseek: Combining color and shape invariant features for image retrieval. *Image Processing, IEEE Transactions on*, 9(1):102–119, 2000. ISSN 1057-7149.

J. Goldberger, H. Greenspan, and S. Gordon. Unsupervised image clustering using the information bottleneck method. *Pattern Recognition*, pages 158–165, 2002.

P.H. Gosselin, M. Cord, and S. Philipp-Foliguet. Combining visual dictionary, kernel-based similarity and learning strategy for image category retrieval. *Computer vision and image understanding*, 110(3):403–417, 2008.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming. *Available httpstanford edu boydcvx*, 2008.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1):389–422, 2002. ISSN 0885-6125.

R.M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *Systems, Man and Cybernetics, IEEE Transactions on*, 3(6):610–621, 1973. ISSN 0018-9472.

T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2): 83–85, 2005.

X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. *Advances in Neural Information Processing Systems*, 18:507, 2006. ISSN 1049-5258.

T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999. ISBN 1581130961.

M. Hollander and D.A. Wolfe. Nonparametric statistical methods. 1999.

E. Hörster and R. Lienhart. Fusing Local Image Descriptors for Large-Scale Image Retrieval. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

J. Huang, S.R. Kumar, M. Mitra, W.J. Zhu, and R. Zabih. Image indexing using color correlograms. In *cvpr*, page 762. Published by the IEEE Computer Society, 1997.

A. Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *Neural Networks, IEEE Transactions on*, 10(3):626–634, 1999. ISSN 1045-9227.

R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991. ISSN 0899-7667.

F. Jing, M. Li, H.J. Zhang, and B. Zhang. Relevance feedback in region-based image retrieval. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(5):672–681, 2004. ISSN 1051-8215.

T. Joachims. Transductive inference for text classification using support vector machines. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 200–209. MORGAN KAUFMANN PUBLISHERS, INC., 1999.

F.D. Jou, K.C. Fan, and Y.L. Chang. Efficient matching of large-size histograms. *Pattern Recognition Letters*, 25(3):277–286, 2004. ISSN 0167-8655.

FJ Kriegler, WA Malila, RF Nalepka, and W. Richardson. Preprocessing transformations and their effects on multispectral recognition. In *Remote Sensing of Environment, VI*, volume 1, page 97, 1969.

I. Kyrgyzov, O. Kyrgyzov, H. Maître, and M. Campedel. Kernel MDL to determine the number of clusters. *Machine Learning and Data Mining in Pattern Recognition*, pages 203–217, 2007.

S. Lazebnik, C. Schmid, and J. Ponce. Affine-invariant local descriptors and neighborhood statistics for texture recognition. In *Computer Vision. ICCV 2003. Proceedings. 2003 International Conference on*, pages 649–655. Published by the IEEE Computer Society, 2003.

A. Lechervy, P.H. Gosselin, and F. Precioso. Active Boosting for interactive object retrieval. In *2010 International Conference on Pattern Recognition*, pages 3268–3271. IEEE, 2010.

M.S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 2(1):19, 2006.

J. Li and J.Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1075–1088, 2003. ISSN 0162-8828.

Y. Li and T. Bretschneider. Remote Sensing Image Retrieval Using a Context-Sensitive Bayesian Network with Relevance Feedback. In *Geoscience and Remote Sensing Symposium, 2006. IGARSS 2006. IEEE International Conference on*, pages 2461–2464. IEEE, 2006. ISBN 0780395107.

M. Lienou, H. Maître, and M. Datcu. Semantic annotation of satellite images using latent dirichlet allocation. *Geoscience and Remote Sensing Letters, IEEE*, 7(1):28–32, 2010. ISSN 1545-598X.

C.F. Lin and S.D. Wang. Fuzzy support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):464–471, 2002. ISSN 1045-9227.

C. Liu, Y. Yang, and Y. Chen. Constructing visual vocabularies using sparse coding for action recognition. In *Information Engineering and Computer Science, ICIECS. International Conference on*, pages 1–4. IEEE, 2009.

Y. Liu. *Studies on support vector machines and applications to video object extraction*. PhD thesis, The Ohio State University, 2006.

J.C. Loehlin. *Latent variable models*. Erlbaum, 1987. ISBN 0898599636.

P. Long and R. Servedio. Discriminative learning can succeed where generative learning fails. *Learning Theory*, pages 319–334, 2006.

J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14, 1967.

C. L. Mallows. A note on asymptotic joint normality. *Annals of Mathematical Statistics*, 43(2):508–515, 1972.

B.S. Manjunath and W.Y. Ma. Texture features for browsing and retrieval of image data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(8):837–842, 1996. ISSN 0162-8828.

BS Manjunath, P. Salembier, and T. Sikora. *Introduction to MPEG-7: multimedia content description interface*. John Wiley & Sons Inc, 2002. ISBN 0471486787.

T. Marill and D. Green. On the effectiveness of receptors in recognition systems. *Information Theory, IEEE Transactions on*, 9(1):11–17, 1963. ISSN 0018-9448.

A.D.R. McQuarrie and C.L. Tsai. *Regression and time series model selection*. World Scientific Pub Co Inc, 1998. ISBN 981023242X.

K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International journal of computer vision*, 60(1):63–86, 2004. ISSN 0920-5691.

P.M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *Computers, IEEE Transactions on*, 100(9):917–922, 1977. ISSN 0018-9340.

R.M. Neal and G.E. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. *Learning in graphical models*, 89:355–368, 1998.

S. Newsam and Y. Yang. Comparing global and interest point descriptors for similarity retrieval in remote sensed imagery. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*, page 9. ACM, 2007.

K. Nigam, A.K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine learning*, 39(2):103–134, 2000.

A. Parulekar, R. Datta, J. Li, and J.Z. Wang. Large-scale satellite image browsing using automatic semantic categorization and contentbased retrieval. In *Proceedings of the IEEE International Workshop on Semantic Knowledge in Computer Vision, in conjunction with IEEE International Conference on Computer Vision*, 2005.

G. Pass and R. Zabih. Comparing images using joint histograms. *Multimedia systems*, 7(3):234–240, 1999. ISSN 0942-4962.

G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *Proceedings of the fourth ACM international conference on Multimedia*, pages 65–73. ACM, 1997. ISBN 0897918711.

G. Patané and M. Russo. The enhanced LBG algorithm. *Neural Networks*, 14(9):1219–1237, 2001. ISSN 0893-6080.

H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, pages 1226–1238, 2005. ISSN 0162-8828.

J. Peng, B. Bhanu, and S. Qing. Probabilistic feature relevance learning for content-based image retrieval. *Computer vision and image understanding*, 75(1-2):150–164, 1999. ISSN 1077-3142.

A. Pentland, R.W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254, 1996. ISSN 0920-5691.

X. Perrotton, M. Sturzel, and M. Roux. Implicit hierarchical boosting for multi-view object detection. pages 958–965, 2010.

M. Pesaresi and J.A. Benediktsson. A new approach for the morphological segmentation of high-resolution satellite imagery. *Geoscience and Remote Sensing, IEEE Transactions on*, 39(2):309–320, 2001.

J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 2000.

P. Pudil, J. Novovicová, and J. Kittler. Floating search methods in feature selection. *Pattern recognition letters*, 15(11):1119–1125, 1994. ISSN 0167-8655.

P. Quelhas and J.M. Odobez. Natural scene image modeling using color and texture visterms. *Image and Video Retrieval*, pages 411–421, 2006.

T. Randen and J.H. Husoy. Multichannel filtering for image texture segmentation. *OPTICAL ENGINEERING-BELLINGHAM-INTERNATIONAL SOCIETY FOR OPTICAL ENGINEERING-*, 33:2617–2617, 1994. ISSN 0091-3286.

J.J. Rocchio. Relevance feedback in information retrieval. 1971.

S. Romdhani, P. Torr, et al. Computationally efficient face detection. pages 695–700, 2001.

K. Rose. A mapping approach to rate-distortion computation and analysis. *Information Theory, IEEE Transactions on*, 40(6):1939–1952, 1994.

K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of the IEEE*, 86(11):2210–2239, 1998. ISSN 0018-9219.

K. Rose, E. Gurewitz, and G.C. Fox. Constrained clustering as an optimization method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(8):785–794, 1993.

Y. Rubner, C. Tomasi, and L.J. Guibas. A metric for distributions with applications to image databases. 1998.

Y. Rui, T.S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool for interactive content-based image retrieval. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(5):644–655, 1998. ISSN 1051-8215.

H. Sahbi and D. Geman. A hierarchy of support vector machines for pattern detection. *The Journal of Machine Learning Research*, 7:2087–2123, 2006. ISSN 1532-4435.

H. Sahbi, D. Geman, and N. Boujemaa. Face detection using coarse-to-fine support vector classifiers. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 3, pages 925–928. IEEE, 2002. ISBN 0780376226.

A. Samal, S. Bhatia, P. Vadlamani, and D. Marx. Searching satellite imagery with integrated measures. *Pattern Recognition*, 42(11):2502–2513, 2009.

B. Schölkopf and A.J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. the MIT Press, 2002.

M. Schröder, H. Rehrauer, K. Seidel, and M. Datcu. Interactive learning and probabilistic retrieval in remote sensing image archives. *IEEE Transactions on Geoscience and Remote Sensing*, 38(5), 2000a.

M. Schröder, M. Walessa, H. Rehrauer, K. Seidel, and M. Datcu. Gibbs random field models: a toolbox for spatial information extraction. *Computers & Geosciences*, 26 (4):423–432, 2000b. ISSN 0098-3004.

B.M. Shahshahani and D.A. Landgrebe. The effect of unlabeled samples in reducing the small sample size problem and mitigating the Hughes phenomenon. *Geoscience and Remote Sensing, IEEE Transactions on*, 32(5):1087–1095, 1994. ISSN 0196-2892.

G. Sheikholeslami, W. Chang, and A. Zhang. Semquery: Semantic clustering and querying on heterogeneous features for visual data. *IEEE Transactions on Knowledge and Data Engineering*, pages 988–1002, 2002. ISSN 1041-4347.

C.R. Shyu, M. Klaric, G.J. Scott, A.S. Barb, C.H. Davis, and K. Palaniappan. GeoIRIS: Geospatial information retrieval and indexing system-content mining, semantics modeling, and complex queries. *Geoscience and Remote Sensing, IEEE Transactions on*, 45(4):839–852, 2007. ISSN 0196-2892.

J.R. Smith and S.F. Chang. VisualSEEk: a fully automated content-based image query system. In *ACM multimedia*, volume 96, pages 87–98. Boton MA: MCM Press, 1996.

S.D. Stearns. On selecting features for pattern classifiers. In *Proceedings of the 3rd International Conference on Pattern Recognition*, pages 71–75, 1976.

M. Stricker and M. Orengo. Similarity of color images. In *Proc. SPIE Storage and Retrieval for Image and Video Databases*, volume 2420, pages 381–392. Citeseer, 1995.

M.J. Swain and D.H. Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991. ISSN 0920-5691.

L. Talavera and J. Béjar. Generality-based conceptual clustering with probabilistic concepts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(2):196–206, 2001. ISSN 0162-8828.

H. Tamura, S. Mori, and T. Yamawaki. Textural features corresponding to visual perception. *Systems, Man and Cybernetics, IEEE Transactions on*, 8(6):460–473, 1978. ISSN 0018-9472.

D. Tao, X. Tang, X. Li, and X. Wu. Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1088–1099, 2006. ISSN 0162-8828.

C.E. Thomaz, D.F. Gillies, and R.Q. Feitosa. Small sample problem in bayes plug-in classifier for image recognition. *Small*, 510:2.

M.E. Tipping. Sparse Bayesian learning and the relevance vector machine. *The Journal of Machine Learning Research*, 1:211–244, 2001. ISSN 1532-4435.

N. Tishby, F.C. Pereira, and W. Bialek. The information bottleneck method. *Arxiv preprint physics/0004057*, 2000.

S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 107–118. ACM, 2001. ISBN 1581133944.

I.W. Tsang, J.T. Kwok, and P.M. Cheung. Core vector machines: Fast SVM training on very large data sets. *Journal of Machine Learning Research*, 6(1):363, 2006.

C. Tusk, K. Koperski, S. Aksoy, and G. Marchisio. Automated feature selection through relevance feedback. In *2003 IEEE International Geoscience and Remote Sensing Symposium, 2003. IGARSS'03. Proceedings*, pages 3691–3693, 2003.

N. Ueda and R. Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11 (2):271–282, 1998. ISSN 0893-6080.

J. Van De Weijer, T. Gevers, and A.W.M. Smeulders. Robust photometric invariant features from the color tensor. *Image Processing, IEEE Transactions on*, 15(1):118–127, 2006. ISSN 1057-7149.

V.N. Vapnik. Statistical learning theory. 1998.

C. Vertan and N. Boujemaa. Upgrading Color Distributions for Image Retrieval Can We Do Better? *Advances in Visual Information Systems*, pages 597–606, 2000.

P. Viola and M.J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004. ISSN 0920-5691.

J. Wang, J. Li, and G. Wiederholdy. Simplicity: Semantics-sensitive integrated matching for picture libraries. *Advances in Visual Information Systems*, pages 171–193, 2000.

J.Z. Wang, N. Boujemaa, A. Del Bimbo, D. Geman, A.G. Hauptmann, and J. Tesić. Diversity in multimedia information retrieval research. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 5–12. ACM, 2006. ISBN 1595934952.

G.I. Webb, J.R. Boughton, and Z. Wang. Not so naive bayes: Aggregating one-dependence estimators. *Machine Learning*, 58(1):5–24, 2005. ISSN 0885-6125.

A.W. Whitney. A direct method of nonparametric measurement selection. *Computers, IEEE Transactions on*, 100(9):1100–1103, 1971. ISSN 0018-9340.

L. Xu and D. Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In *Proceedings of the National Conference on Artificial Intelligence*, number 2(20), page 904. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. *Advances in neural information processing systems*, 17:1537–1544, 2005.

J. Yang. Review of Multi-Instance Learning and Its applications. 2008.

X.C. Yin, C.P. Liu, and Z. Han. Feature combination using boosting. *Pattern Recognition Letters*, 26(14):2195–2205, 2005. ISSN 0167-8655.

H. Yu, J. Yang, and J. Han. Classifying large data sets using SVMs with hierarchical clusters. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, page 315. ACM, 2003.

R. Zhang and Z.M. Zhang. Hidden semantic concept discovery in region based image retrieval. 2004. ISSN 1063-6919.

D. Zhou, B. Schölkopf, and T. Hofmann. Semi-supervised learning on directed graphs. *Advances in neural information processing systems*, 17:1633–1640, 2005.

X.S. Zhou and T.S. Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia systems*, 8(6):536–544, 2003.

X. Zhu. Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison*, 2006.

X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. *School Comput. Sci., Carnegie Mellon Univ., Tech. Rep. CMUCALD-02-107*, 2002.

X. Zhu and J. Lafferty. Harmonic mixtures: combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, page 1059. ACM, 2005.