On Labeling Noise and Outliers for Robust Concept Learning for Image Databases

Abstract

Recently mixture model has been used to model image databases. The retrieval experiences derived from multiple users' relevance feedbacks have been used to improve model fitting in a semi-supervised manner. However, the mixture model for image databases remains as a challenging task since the database may contain clutter and outliers, and labelling information derived from multiple users may be inconsistent. Thus, neither the mixture model nor the labelling information is as ideal as most of the researchers have previously assumed. In this paper, we (a) address the problems of the noise disturbances for both mixture model and users' labelling information, (b) propose to process retrieval experiences in an intelligent manner using Bayesian analysis, (c) present a robust mixture model fitting algorithm to achieve visual concept learning, and (d) construct a concept-based indexing structure for efficient search of the database. The experimental results on a Corel image set show the correctness of our retrieval experience analysis, the effectiveness of the proposed concept learning approach, and the improvement of retrieval performance based on the indexing structure.

1 Introduction

There are still many challenging areas (retrieval, learning, indexing, visualization, etc.) in content-based image retrieval (CBIR) after the extensive research efforts of the last 13 years. Mixture model [1] has been adopted by some researchers in this field since it provides well-established statistical techniques and more subtle learning approaches may be used based on the model assumption. In particular, Gaussian mixture model (GMM) has been used to model the image distribution in the feature space of image databases [2] [3] [4]. The task of concept learning is to explore the characteristics of the features that can represent a concept. Specifically, for Gaussian mixture model assumption, concept learning is for mixture model fitting, which includes estimating the number of components and the component parameters. The concept learning knowledge may provide good classifiers and thus, improve retrieval performance.

However, the research of mixture model fitting for image databases is still at the embryonic stage due to the challenges such as the existence of *outlier and clutter images*, *high dimensionality* of feature space, and the *concept drift* caused by the *dynamic mechanism* of databases. Thus, it is usually impossible to achieve satisfactory mixture model fitting for image databases in an unsupervised manner, and the correct model has to be learned with the help of some additional labeling knowledge, i.e., the learning should be achieved in a *semi-supervised* manner.

Recently, there have appeared some image retrieval systems which exploit meta knowledge derived from the previous *multiple users'* retrieval sessions to improve retrieval performance [5] [6] [7]. Specifically, the *relevance feedback* [8] mechanism of the system allows each of the users to label the presented images as positive or negative. The collection of such positive and negative labeling information derived from the multiple users may also help the mixture model fitting [4] [9]. In this scenario, a widely-used approach [6] [7] to improve retrieval precision is to adopt a new similarity (dissimilarity) measurement between images by combining a visual-feature-based similarity (dissimilarity) term and a user-labeling-based similarity (dissimilarity) term. Although this technique has be proved to be effective by the experiments in these papers, the task of determining the values of weights to make balance between these two terms is impossible or at least very difficult. Thus, this is only a heuristic technique.

Compared with the traditional manner of providing labeling information directly, obtaining labeling information from multiple users' retrieval sessions is a more natural and a practical way to help learning. However, this scenario brings a new problem that different users may provide inconsistent labeling information. We regard the labeling opinion supported by the majority of people as "correct" labeling, which should be used to help the concept learning, and the other inconsistent labeling (supported only by the minority of people) as *labeling noise*, which the system should avoid to use due to its misleading effect on concept learning. we propose to process and exploit labeling information derived from multiple users using Bayesian analysis.

We achieve concept learning via estimating mixture model by using *Expectation-Maximization* (EM) algorithm [1], which necessitates the number of components in advance. The task of determining this number for mixture model (also called *model selection*) is a challenging problem, although it has been extensively studied [1] over decades. Due to the unavoidable existence of outliers in image databases, the EM algorithm has to be robust to achieve satisfactory model fitting. The research on the robustness of clustering has experienced many researchers' efforts [10]. In our EM algorithm, outliers are detected based on their distances to clus-

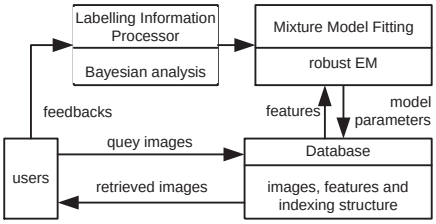


Figure 1: System diagram for robust concept learning and indexing.

ters, so that the clustering is insensitive to them. Such robustness method is more important for out system since it may avoid further misleading effect.

In this paper, we address the problems in image databases brought by (1) the outliers for mixture model assumption and (2) users' labeling noise, which have seldom been considered by CBIR researchers previously. Figure 1 illustrates the diagram for our system with concept learning and indexing. The contributions of this papers are: (a) processing retrieval experiences intelligently and correctly using Bayesian analysis, (b) a robust discriminant EM framework to avoid the misleading effect of outliers; (c) a concept-based indexing structure using the result of concept learning to help efficient search. We will presents the experimental results to demonstrate the effectiveness of the proposes approaches with over 6000 Corel images.

2 Technical approach

2.1 Image distribution in databases

In the feature space of an image database, some images belonging to the same concept may be close to each other in sense of some dimensions, and thus they form a cluster. The mixture model for image databases originates from this observation, as such clusters form a mixture of densities. A concept corresponding to such a cluster is called a *visual concept*, which differentiates from *semantic concept*. A semantic concept is at higher level compared with visual concept, and it may contain multiple visual concepts. A visual concept can be regarded as a class, which we call a *visual class*. In this paper, for the reason of simplicity, we use "class" to refer visual class, and "concept" to refer visual concept. The task of (visual) concept learning for an image database is to explore the mixture model hidden in the image distribution.

For an image database in real applications, we assume that the image distribution is a c-component Gaussian mixture $\mathcal{C} = \{C_1, \ldots, C_c\}$, whose probability density function (PDF) is

$$f(x; \Psi_c) = \sum_{i=1}^{c} \pi_i f_i(x; \mu_i, \Sigma_i) + \pi_0 f_0(x) \qquad (1)$$

The first term on the right side of (1) represents the standard GMM [1], which has been widely adopted in the field of multimedia retrieval, where x is d-dimensional feature, $f_i(x)$ are component densities and π_i $(i = 1, 2, \ldots, c)$ are component proportions. Each component is corresponding to a visual concept. The component densities are specified by means μ_i and covariances Σ_i. Ψ_c is the vector containing all the unknown parameters i.e., $\Psi_c = \bigcup_{i=1}^{c} \{\pi_i, \mu_i, \Sigma_i\}$. However, the image distribution in a real image databases is usually not so ideal: there are some noise images which do not belong to any existing component since the visual concepts these images are belonging to have too few images to form their own components in the feature space. These images are called *clutter* images, whose distribution is represented by the second term on the right side of (1), where π_0 is their proportion and $f_0(x)$ is their PDF, which may be in any form due to the unpredictability of clutter image distribution. The component proportions satisfy $0 \leq \pi_i \leq 1$ $(i = 0, 1, 2, \ldots, c)$ and $\sum_{i=0}^{c} \pi_i = 1$.

Besides the clutter images, the rest of the database images are called *component images* since each of them belongs to one and only component. One may argue that an image may belong to different visual concepts. We admit this possibility, and assign the image to the component that the majority of people support this assignment. Some component images may be *outliers*, which are far away from their corresponding components. The outlier images and clutter images have the same misleading effect on clustering, and make the correct mixture model fitting difficult. For convenience, we call both of such two kinds of images as outliers.

2.2 Labeling information analysis

We define the labeling information derived from the relevance feedback of a user as a *retrieval experience* $\mathcal{E} = \{\mathcal{X}^+, \mathcal{X}^-\}$, where $\mathcal{X}^+ = \{x_1^+, x_2^+, \ldots, x_{N^+}^+\}$ are labeled as belonging to (positive for) a certain but unknown class while another portion of samples $\mathcal{X}^- = \{x_1^-, x_2^-, \ldots, x_{N^-}^-\}$ are labeled as NOT belonging to (negative for) that unknown class. Note that x_i^+ $(i = 1, 2, \ldots, N^+)$ and x_j^- $(j = 1, 2, \ldots, N^-)$ are image visual feature vectors.

We have to be careful when using labeling information from retrieval experiences. The pieces of labeling information provided by different users may be inconsistent due to the reasons such as: (1) Different people perceive visual content differently. (2) Some users may not be cooperative during the labeling process in relevance feedback. (3) Some images may belong to the same semantic concept but different visual concepts, and a user may label all of them as positive, or only label the images belonging to a single concept as positive. We assume that each database image belongs to one and only class (supported by the majority of people). We regard this assignment as "correct" labeling information, and all the other contradicting labeling information on this image as *labeling noise*.

Since the amount of retrieval experiences directly provided by users may be huge, and the system cannot store all of them during the database lifetime, an efficient way to accumulate and analyze the multiple retrieval experiences is necessary. In the case with no labeling noise, if a pair of retrieval experiences $\mathcal{E}_1$

and $\mathcal{E}_1$ contain one or more common positive images, we deduce that all the positive images in $\mathcal{E}_1$ and $\mathcal{E}_1$ belong to a common concept C, and all their negative images do not belong to C. For example, for $\mathcal{E}_1 = \{3^+, 4^+, 5^-\}$ and $\mathcal{E}_2 = \{2^+, 3^+, 1^-\}$, we can merge them as $\{2^+, 3^+, 4^+, 1^-, 5^-\}$. We call such a piece of labeling information as a *concept experience*, since merge decision is based the assumption that different retrieval experiences to be merged are for the same concept. An extreme case is that a retrieval experience can also be regarded as a concept experience if it can not be merged to any other retrieval experiences.

With a collection of retrieval experiences obtained, the system derives a concept retrieval collection Φ by merging, which is as informative as and more succinct than the original retrieval experience collection. Let Φ denote the concept experience collection, and ϕ_i $(i = 1, 2, \ldots, |\Phi|)$ be the concept experiences in Φ. Thus, the system does not have to memorize the retrieval experience collection during the lifetime of the database; instead, the system keeps Φ and updates it whenever a new retrieval experience $\mathcal{E}_{new}$ is obtained by either merging $\mathcal{E}_{new}$ into a concept experience in Φ, or directly inserting $\mathcal{E}_{new}$ into Φ as a new concept experience.

A concept retrieval can be expressed in a probabilistic form, in which each image is represented by its index and the probability that this image belongs to this concept. For example, the concept experience $\{2^+, 3^+, 4^+, 1^-, 5^-\}$ is expressed as $\{(2,1), (3,1), (4,1), (1,0), (5,0)\}$. Such probabilistic expression of concept experience is necessary in the Bayesian analysis for the existence of labeling noise, which we will present in the following.

If labeling noise exists, i.e., a user may mislabel the images given to him/her, and the direct merging method would be disastrous for labeling information collection. For example, $\mathcal{E}_1 = \{\cdots, 3^+, 4^-, 5^-, \cdots\}$ and $\mathcal{E}_2 = \{\cdots, 2^+, 3^+, 1^-, \cdots, \}$ would be merged as $\{\cdots, 2^+, 3^+, 1^-, 4^-, 5^- \cdots\}$. However, if Image 3 is mislabelled in $\mathcal{E}_2$ (i.e., Image 3 is actually not belonging to C_2), and $C_1 \neq C_2$, the labeling information expressed in the deducted concept experience would contain many errors, which may mislead the subsequent mixture model fitting for concept learning.

To strictly analyze the labeling information, we define the *labeling noise rates* as

$\alpha(I, \mathcal{E}) = prob(I \notin \mathcal{E} | I \in C; \mathcal{E} \subset C)$, and

$\beta(I, \mathcal{E}) = prob(I \in \mathcal{E} | I \notin C; \mathcal{E} \subset C)$,

where "$\mathcal{E} \subset C$" means that the user who has provided retrieval experience $\mathcal{E}$ seeks the images of Concept C, "$I \in (\notin)\ C$" represents that Image I is (not) belonging to Concept C, and "$I \in (\notin)\ \mathcal{E}$" denotes that Image I is labelled as positive (negative) in retrieval experience $\mathcal{E}$. Basically, $\alpha(I, \mathcal{E})$ denotes the probability that a user labels an image as negative when this user is seeking a concept and this image belongs to this concept, and $\beta(I, \mathcal{E})$ denotes the probability that a user labels an image as positive when this user is seeking a concept and this image does NOT belong to this concept.

For a pair of experiences (either retrieval experience or concept experience), to avoid incorrect merge, we will compute the probability that they are for the same concept, and the experience merge will happen only if such probability is very high. First, when a retrieval experience $\mathcal{E}$ is derived directly from a user whose is seeking concept C, if the labeling noise rates $\alpha(I, \mathcal{E})$ and $\beta(I, \mathcal{E})$ are known, the probability that an positive image I^+ in $\mathcal{E}$ belongs to Concept C is

$prob(I^+ \in C | I^+ \in \mathcal{E})$

$= \frac{prob(I^+ \in \mathcal{E} | I^+ \in C) prob(I^+ \in C)}{prob(I^+ \in \mathcal{E} | I^+ \in C) prob(I^+ \in C) + prob(I^+ \in \mathcal{E} | I^+ \notin C) prob(I^+ \notin C)}$

$= \frac{1 - \alpha(I^+, \mathcal{E})}{(1 - \alpha(I^+, \mathcal{E})) + \beta(I^+, \mathcal{E})(c_{local} - 1)}$,

where c_{local} is the number of classes that the image may possibly belong to. Note that c_{local} is not the total number of the concepts in the database, but the number of concepts that may overlap around an image, and its values is much smaller than the total number of concepts. The prior probability $prob(I^+ \in C)$ that an image I^+ belongs to a concept C is $1/c_{local}$, which is used in the deduction. The exact values of $\alpha(I^+, \mathcal{E})$, $\beta(I^+, \mathcal{E})$ and c_{local} are unknown, but we can predetermine them as constants α, β and $\hat{c}_{local}$ respectively, such that the predetermined values are higher than their real values. In this way, the labeling information precession is a little conservative but avoids mistakes. Similarly, the probability that an negative image I^- in $\mathcal{E}$ belongs to Concept C is

$$prob(I^- \in C | I^- \notin \mathcal{E}) = \frac{\alpha(I^-, \mathcal{E})}{\alpha(I^-, \mathcal{E}) + (1 - \beta(I^-, \mathcal{E}))(c_{local} - 1)}. \quad (2)$$

Now we study the probability that two experiences are for the same concept, since this probability provides the criterion whether these two experiences should be merged or not. Formally, if two experiences ϕ_1 and ϕ_2 (ϕ_1 is for Concepts C_1 and ϕ_2 is for Concepts C_2) contain the common images $\{I_1, I_2, \ldots, I_l\}$ whose probabilities belonging to C_1 and C_2 are $prob(I_i \in C_1) = p_{i1}$ and $prob(I_i \in C_2) = p_{i2}$ respectively $(i = 1, 2, \ldots, l)$, how to calculate $prob(C_1 = C_2 | \phi_1, \phi_2)$?

For an image I_i, there are four possibilities for its relationships with C_1 and C_2: $\{I_i \in C_1, I_i \in C_2\}$, $\{I_i \in C_1, I_i \notin C_2\}$, $\{I_i \notin C_1, I_i \in C_2\}$ and $\{I_i \notin C_1, I_i \notin C_2\}$. However, such relationships between different images are not independent. For example, $\{I_1 \in C_1, I_1 \in C_2\}$ prohibits the possibility of $\{I_2 \in C_1, I_2 \notin C_2\}$, since the first case implies $C_1 = C2$ (remember our assumption that each image is belonging to one and only visual concept) and the last one implies $C_1 \neq C2$, thus the two cases are contradicting. Generally, there are two possible cases:

(1) $\bigcap_i \{(I_i \in C_1) \cap (I_i \in C_2) \bigcup (I_i \notin C_1) \cap (I_i \notin C_2)\}$,

(2) $\bigcap_i \{(I_i \in C_1) \cap (I_i \notin C_2) \bigcup (I_i \notin C_1) \cap (I_i \in C_2)\}$.

In the first case, all the possibilities implies $C_1 = C_2$, except the possibility that no image is belonging to either C_1 or C_2, whose probability is $\prod_i p_{i1} p_{i2}$. Although it is still possible for $C_1 = C_2$ in this case, its probability is slim and we can ignore it. Thus,

$$prob(\phi_1, \phi_2 | C_1 = C_2)$$

Algorithm 1 Algorithm for initializing and updating the concept experience collection Φ.

$t = 0$, $\Phi \leftarrow \emptyset$.
repeat {the system gets a new retrieval experience $\mathcal{E}_t$ ($\mathcal{E}_t \subset C_t$), $t = t + 1$}
  1. For the images in $\mathcal{E}_t$, compute $prob(I^+ \in C_t | I^+ \in \mathcal{E}_t)$ and $prob(I^- \in C_t | I^- \in \mathcal{E}_t)$, respectively.
  2. If $\Phi = \emptyset$, **insert** $\mathcal{E}_t$ into Φ as a new concept experience, and go back to Repeat;
  3. $\phi_s \leftarrow \arg\max_{\phi_i \in \Phi} prob(C_i = C_t | \phi_i, \phi_t)$;
  if $prob(C_s = C_t | \phi_s, \phi_t) < T_{merge}$ **then**
    Insert $\mathcal{E}_t$ into Φ directly as a new concept experience;
  else
    Merge $\mathcal{E}_t$ into ϕ_s; flag $\leftarrow 1$;
    while flag = 1 **do**
      $\phi_{s'} \leftarrow \arg\max_{\phi_i \in \Phi, i \neq s} prob(C_i = C_s | \phi_i, \phi_s)$;
      if $prob(C_{s'} = C_s | \phi_{s'}, \phi_s) > T_{merge}$ **then**
        Merge ϕ_s into $\phi_{s'}$; **Remove** ϕ_s from Φ;
        Call $\phi_{s'}$ as ϕ_s;
      else
        flag $\leftarrow 0$
      end if
    end while
  end if
until the database finishes its lifetime

$$= \prod_i \{p_{i1}p_{i2} + (1-p_{i1})(1-p_{i2})\} - \prod_i p_{i1}p_{i2}. \quad (3)$$

For the second case,

$$prob(\phi_1, \phi_2 | C_1 \neq C_2) = \prod_i \{p_{i1}(1-p_{i2}) + (1-p_{i1})p_{i2}\}. \quad (4)$$

Thus, the probability that C_1 and C_2 are the same concept based on the observation of ϕ_1 and ϕ_2 is

$prob(C_1 = C2 | \phi_1, \phi_2) =$
$\frac{prob(\phi_1,\phi_2|C_1=C_2)prob(C_1=C_2)}{prob(\phi_1,\phi_2|C_1=C_2)prob(C_1=C_2)+prob(\phi_1,\phi_2|C_1\neq C_2)prob(C_1\neq C_2)}$,
where $prob(C_1 = C_2) = 1/c_{max}$ and $prob(C_1 \neq C_2) = 1 - 1/c_{max}$ (c_{max} is the highest possible number of classes known by the system).

To merge an experience ϕ_2 to another experience ϕ_1, for an image I_i which is contained in both ϕ_1 and ϕ_2, the probability that it belongs to C_1 is updated as

$prob(I_i \in C_1 | C_1 = C_2; \phi_1, \phi_2)$
$= \frac{prob(C_1=C_2|I_i\in C_1;\phi_1,\phi_2)prob(I_i\in C_1|\phi_1,\phi_2)}{prob(C_1=C_2|\phi_1,\phi_2)}$,
where $prob(C_1 = C_2 | I_i \in C_1; \phi_1, \phi_2)$ is
$prob(C_1 = C_2 | I_i \in C_1, I_i \in C_2; \phi_1, \phi_2) prob(I_i \in C2 | \phi_1, \phi_2) = p_{i2}$,
by Bayesian inference with $prob(C_1 = C_2 | I_i \in C_1, I_i \notin C2) = 0$. Thus, it is easy to get

$$prob(I_i \in C_1 | C_1 = C_2; \phi_1, \phi_2) = \frac{p_{i1}p_{i2}}{prob(C_1 = C_2 | \phi_1, \phi_2)}. \quad (5)$$

For an image I, if $I \in \phi_1$ and $I \notin \phi_2$, we keep the probability $prob(I \in C_1)$ unchanged in ϕ_1 when merging ϕ_2 into ϕ_1. If $I \notin \phi_1$ and $I \in \phi_2$, to relate I to ϕ_1 during merging, we assign

$$prob(I \in C_1) = prob(I \in C_2) prob(C_1 = C_2; \phi_1, \phi_2). \quad (6)$$

When we mention that ϕ_2 is merged to ϕ_1, it implies that the probabilities of all the images related with C_1 or C_2 are updated or assigned regarding ϕ_1 as we have introduced above, and ϕ_2 is deleted.

Algorithm 1 shows our algorithm for the computation of the concept experience collection Φ. At the beginning stage of the database without retrieval experience, Φ is initialized to be a null set. Each time a new retrieval experience is provided by the a user, we use a threshold parameters T_{merge} ($T_{merge} = 0.95$ in this work) to decide whether the new retrieval experience should be added directly to Φ or be merged to an existing concept experience in Φ. In the second case, the update of ϕ_s may make it eligible to be merged into another existing concept retrieval $\phi_{s'}$ in Φ, and so on. The "while" loop describes the details of the merging series.

2.3 Semi-supervised EM framework

2.3.1 Over-splitting initialization: To exploit the collected concept experiences to help the mixture model fitting, we defuzzy each of the concept experience in Φ. At any time t, for the concept experience ϕ_j whose corresponding concept is $\mathcal{C}_j$, we defuzzy the images' probabilities for $\mathcal{C}_j$ in ϕ_j as

$$\bar{p}_{ij} = \begin{cases} 1 & \text{if } prob(I_i \in C_j) > 1 - \epsilon \\ 0 & \text{if } prob(I_i \in C_j) < \epsilon \\ \text{unkown} & \text{otherwise} \end{cases} \quad (7)$$

where ϵ is a small value ($\epsilon = 0.1$ in this paper) and I_i is one of the images contained in ϕ_j. If ϕ_j contains enough number of images whose defuzzied probabilities in (7) are "1", it can be used in the EM algorithm for the mixture model fitting. We set the threshold value for such number of images in a concept experience to be 5. For such defuzzied concept experiences, we can estimate their centers by averaging the features of positive images (with $\bar{p}_{ij} = 1$). Let us denote c_Φ as the number of concept experiences that contained enough images with defuzzied probabilities of "1".

To estimate the mixture model using EM, the number of components for the algorithm has to be given, although the system usually does not know the real number of classes c_{real}. With the knowledge of highest possible number of classes c_{max} and size of the defuzzied concept experience collection c_Φ, we can determine the initial number of components c for the EM algorithm as $c = max(c_{max}, c_\Phi)$. We set c_Φ of the c initial component centers to be the c_Φ concept experience centers, and randomly select $c - c_\Phi$ data as the remaining $c - c_\Phi$ component centers. We also initialize the c component covariances matrices to be identity matrices.

To evaluate the over-splitting clustering result $\mathcal{R} = \{\mathcal{R}_1, \ldots, \mathcal{R}_c\}$, we compare it with the groundtruth mixture model $\mathcal{C} = \{C_1, \ldots, C_{c_{real}}\}$ ($c > c_{real}$) by using a statistical index. A pair of vectors $\{x_i, x_j\}$ are referred to as (I) BS if both vectors belong to the same component in $\mathcal{C}$ and to the same cluster in $\mathcal{R}$, (II) BB if both vectors belong to the same component in $\mathcal{C}$. Let ξ_1 and ξ_2 be the number of BS and BB respectively, we use $SI = \frac{\xi_1}{\xi_2}$ to evaluate an over-splitting clustering result. Obviously, for two different clustering results with

the same number of clusters c, the one with higher SI value is a better clustering since it is more consistent with groundtruth model.

We argue that if the results of an over-splitting clustering is relatively good, i.e., the value of SI is high, the system may also yield good retrieval performance by using this clustering result for classification. When a query image is given to the system, the system may limit the search within the cluster region where the query feature vector is located. If the value of SI is high, i.e., most of the images within this cluster belong to the class which is corresponding to the class the query image belongs to, the retrieved images within this cluster can satisfy the user, although their average visual similarity with the query image around the boundary of the cluster may be a little lower.

2.3.2 Outlier detection and rejection: For each component at the EM iteration, we select a small proportion of the images with the lowest PDF values, and reject them in the estimation for component means, covariances and proportions in the subsequent M-step. The number of such images to be selected in Component j ($j = 1, 2, \ldots, c$) is $\lfloor \varrho \pi_j N \rfloor$, where N is the number of images in the database, π_j is the proportion of Component j, and the value of ϱ can be given based on the knowledge on the proportion of outliers. This strategy is similar to the work in [11], which also detect outliers based on the distances between a point and the clusters. In this way, we can prevent the misleading of the outliers. Even some images may be misclassified as outliers, this does not influence the model parameter estimation since these images are also relatively far away from the components.

For outliers, both labeling information and the outlier detection mechanism help to avoid them, and the scenario of their combination has never been addressed in either clustering or CBIR research.

2.3.3 EM framework: To explore discriminating features in a self-supervised fashion, the integration of multiple discriminant analysis (MDA) with EM framework is proposed [12]. This is called D-EM, whose probabilistic extension is given in [9]. We implement the probabilistic D-EM in our algorithm.

Since over-split clustering may lead to very small components, which cause the singularity problem, we remove those small components right after the component proportion estimation at each iteration. The removal criterion is: if $\pi_j < \delta(\frac{1}{c})$, $j = 1, 2, \ldots, c$, the jth component is removed. This is called *component annihilation* [13]. Although component annihilation is usually based on the relationship between sample sizes and dimensionality, we simply use a constant parameter δ ($\delta = 0.1$ in this work) since the purpose of our component removal is only to avoid singularity.

Algorithm 2 presents the EM algorithm for concept learning with our proposed components, including over-splitting initialization by exploiting the labeling information, outlier detection and rejection, the integration with MDA and component annihilation.

Algorithm 2 Probabilistic D-EM algorithm for concept learning.

Given the data $\mathcal{X}$, the concept experience collection Φ, and c_{max}.
Initialization (see Section 2.3.1).
Estimate component proportions $\{\pi_1, \pi_2, \ldots, \pi_c\}$.
repeat
1. Remove the jth component if $\pi_j < \delta\frac{1}{c}$, $j = 1, 2, \ldots, c$. Normalize the proportions for the remaining c' components, $c \leftarrow c'$.
2. Detect outliers, which will not be used for the subsequent steps.
3. E-step: Estimate component-indicators $\mathcal{Z}$.
4. Modify $\mathcal{Z}$ by concept collection Φ.
5. $\mathcal{X} \leftarrow$ probabilistic MDA($\mathcal{X}, \mathcal{Z}, \Phi$) [9].
6. M-step: Compute component proportions, means and covariances respectively ([1]).
until termination criterion is met

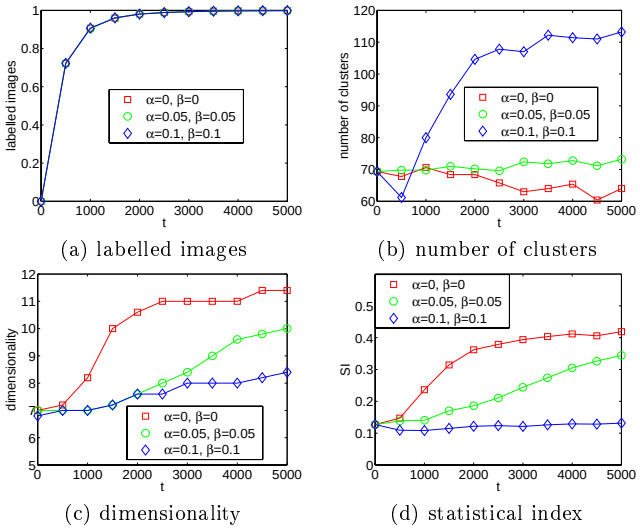
2.4 Concept-based indexing and search

The huge amount of data necessitates an indexing structure for the database to achieve efficient search and update. There have appeared many indexing structures for multimedia databases with multidimensional feature space, such as X-tree, M-tree, TV-tree, etc [14]. However, the effectiveness of many of these indexing approaches is data dependent and difficult to predict.

Our concept-based indexing strategy is based on the intuition that the images regarded by the majority of people as belonging to the same class should be arranged on hard drive as consecutively as possible, so that the page access on hard drive can be saved while the retrieval precision is good. We directly use the clustering result of our EM algorithm for indexing, by which the images belonging to the same cluster are stored consecutively on hard drive. When a query images comes, the system chooses the cluster with the highest probability that the query belongs to this cluster based on the mixture model parameters. Thus, the search is limited to the images belonging to this cluster so that the search time is saved compared with the global search. Although this concept-based indexing strategy has been proposed in [15], it cannot achieve good performance for real image databases without robust concept learning since the outliers may mislead the clustering.

3 Experiments

We implement our concept learning and retrieval approach on an image database which is not matching Gaussian mixture model assumption well and thus more challenging for concept learning task. There are 6155 images with 49 classes, which are corresponding to the CDs in Corel stock photo library. In each of the photo library CDs which are not included, the images in the CD do not have common visual similarity. These excluded CDs are for semantic concepts, whose level are so high that they are totally beyond the scope of CBIR. For example, the CD of *Australia* (69000) contains a variety of images on different visual concepts such as coast, bird, building, desert, mountain, tree, lake, etc,



(a) labelled images (b) number of clusters

(c) dimensionality (d) statistical index

Figure 2: Performance for concept learning with various labeling noise rates.

and the images on each of these concepts are too fewer to form a statistically significant cluster. For each CD included in our database, we select all of its 100 images except we remove 39 images from the CD of *glaciers & mountains* (114000) and remove 6 images from the CD of *polar bears* (183000) since these removed images are obviously outliers. In most of these classes (concepts), there are also many outliers, whose visual features are far away the features of the typical images contained in their classes. Images are represented by texture features and color features. The texture features are derived from 16 *Gabor* filters [16]. We also extract means and standard deviations from the three channels in HSV color space. Thus, each image is represented by 22 features.

We set the system running time as $t = 0, 1, 2, \ldots$; at each t, a user makes his/her queries and provides a retrieval experience by executing relevance feedback. At each relevance feedback iteration, the system presents 20 images for users to label. We set different sets of values for the labeling noise rates α and β, which are known by the system in advance (see Section 2.2). We assume that the system only knows that $c_{max} = \lfloor 1.5 c_{real} \rfloor = 73$ in advance. To detect outliers, we set the outlier detection parameter $\varrho = 0.1$.

To evaluate our concept learning approach, we will adopt quite a few measurements to show the details during the learning process, including the number of clusters after EM algorithm, the reduced feature dimensionality, the statistical index to evaluate the over-splitting clustering, etc. The advantages of our indexing and image search approach on image retrieval performance will be demonstrated by retrieval precision, precision/recall curves, and search time.

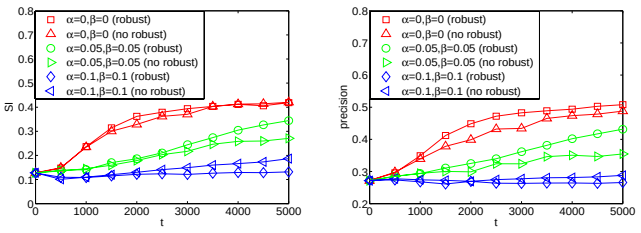
Figure 2 provides various performances for concept learning with different labeling noise rates. All the curves are the average results by repeating the system random process 10 times. Figure 2(a) shows that the percentage of the images ever being labelled (no matter positive or negative) increases with retrieval experiences increased. We observe that the three curves for different labeling noise rates in Figure 2(a) are so close that they can be regarded as the same curve; thus, it will be fair to compare the learning improvements with regard to different labeling noise rates since the amounts of labeling information given the system at any moment are almost equal.

From Figure 2(b), we observe the changes of the number of clusters (after EM algorithm) over time. The number of resulting clusters increases over time, since more concept experiences are available to be fed into EM algorithm.

Figure 2(c) shows that integrating probabilistic MDA with EM effectively reduces the dimensionality of feature space (original dimensionality is 22). Since the main computational load of EM is the computation of the inverse of each component's covariance, whose complexity is $\mathcal{O}(cd^3)$ (c is the number of components and d is feature dimensionality). For the probabilistic MDA, let $\lambda_1, \lambda_2, \ldots, \lambda_d$ denote the eigenvalues for $S_W^{-1} S_B''$ (see Section 2.3.2). These eigenvalues are non-negative and arranged in non-increasing order. We select d' such that $\sum_{i=1}^{d'} \lambda_i / \sum_{i=1}^{d} \lambda_i \leq 95\%$ and $\sum_{i=1}^{d'+1} \lambda_i / \sum_{i=1}^{d} \lambda_i > 95\%$, i.e., we set the energy remained after MDA is 95%. In this experiment, the dimensionality is reduced from $d = 22$ to $d' = 7 \sim 12$. Thus, the computational load for the EM is alleviated significantly. Since the number of clusters decreases over time, the dimensionality of $S_W^{-1} S_B''$ is lower, i.e., this matrix has fewer eigenvalues. As we we have fixed the value of the remained energy (= 95%), there are fewer projected features represented by the small eigenvalues to be thrown away. Thus, the reduced dimensionality increases over time in the long term, as we observe in Figure 2(c).

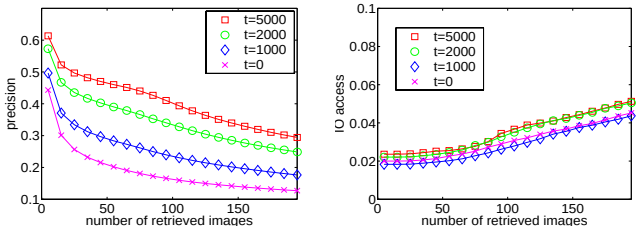
Figure 2(d) validates that the clustering (evaluated by the statistical index SI) is improved with more labeling information provided by retrieval experiences. This demonstrates that the model fitting is in the trend of improvement over time.

Figure 3 compares the proposed robust clustering approach (outlier detection parameter $\varrho = 0.1$) with the clustering approach without outlier detection (i.e., $\varrho = 0$). For the cases with no labeling noise or labeling noise rates being 0.05, the clustering result (represented by SI) and the retrieval precision using robust approach are superior to those without outlier detection: with limited amount of retrieval experiences (t is small), the robust approach is no better since the clustering is too bad to support the outlier detection strategy which is based on the distances between data and cluster centers (but the cluster centers are too far away from the true centers). With more retrieval experiences obtained, the clustering is better and, thus, support the outlier detection strategy better. In this way, both the exploitation of retrieval experiences and the outlier detection strategy are to improve the clus-



(a) statistical index (b) precision

Figure 3: Comparison of robust clustering and non-robust clustering.



(a) recall (b) IO access

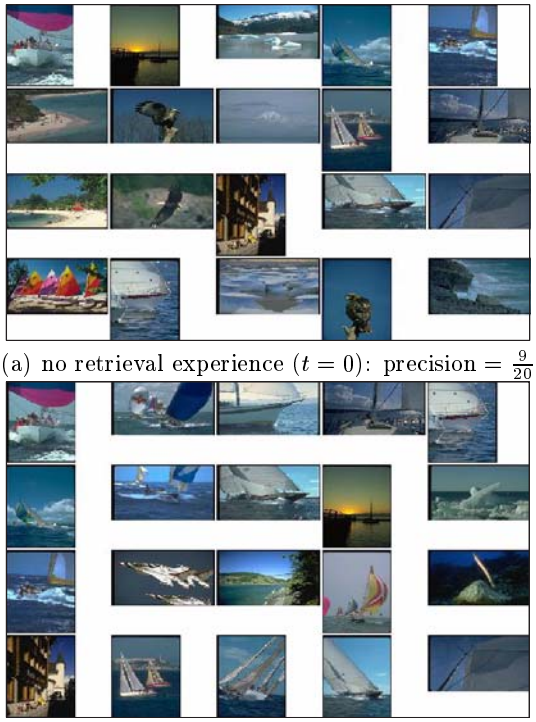
Figure 4: Noise rate = 0.0.

tering; furthermore, they boost each other to achieve better clustering which cannot be achieved by any of them solely. For the case that the labeling noise rates are 0.1, robust approach has no advantage since the clustering is too bad.

Figure 4 presents more results on retrieval performance for the case without labeling noise. Figure 4(a) shows the retrieval precision versus the number of retrieved images, which is similar to the precision/recall curve being widely used in CBIR. The curves in the left region of Figure 4(a) (from 0 to 100) are approximately equal to their corresponding precision/recall curves since the average size of each class is 100, and the curves in the right region (from 100 to 200) give information on the retrieval precision when the number of images to be retrieved is more than the number of relevant images in the database. With more retrieval experiences over time, the precision is better.

To count the image search time, we do not use the actual running time for the search experiment, since it is inaccurate and cannot reflect the search time in the real system. Since the database images are stored on hard drive, most of the time consumed on search is spent on reading database images from hard drive, i.e., IO access. Due to our indexing structure, the images within the same cluster are arranged consecutively on hard drive, which consists of equal-sized pages. The number of pages to be accessed on hard drive represent the search time for a real image retrieval system. Since the number of pages to be accessed is approximately proportional to the number of images to be read from hard drive, we use the number of images to be read to represent IO access. From Figure 4(b), we observe that the IO access is below 6% of the global search time. Thus, we conclude that the indexing structure requires much less IO access compared with the global search.

Figure 5 gives a retrieval example whose precision increases when the concept learning improves with more retrieval experiences.



(a) no retrieval experience ($t = 0$): precision = $\frac{9}{20}$

(b) $t = 500$: precision = $\frac{15}{20}$

Figure 5: Retrieval precision is improved as retrieval experiences are increased. The user is looking for *sailing & sailboats*, and the query is the first image in each image group. (a) When there is no retrieval experience, the search yields only 9 *sailing & sailboats* images as shown in (a) (row 1: image 1, 2, 4, 5; row 2: image 4, 5; row 3: image 4, 5; row 4: image 2). (b) After 500 retrieval experiences, the system gives 15 *sailing & sailboats* images (except row 2: image 5; row 3: image 2, 3, 5; row 4: image 1).

Besides color and texture features, we have also extract other features such as structural features to represent images. In this way, the feature dimensionality is higher (over 40). However, the concept learning improvement is very similar to the presented results since the structural features are not as discriminating as color and texture features for Corel classes, and they are discarded after MDA in EM. This shows that our system is flexibly effective for different image feature representations, as long as some of them are useful for discrimination.

We observe that even the highest retrieval precision is still below 50% in Figure 3(b). This is understandable since the image search is based on the concept learning result, which is represented by the mixture model parameters. Since there are many outlier images in the feature space, the K-nearest neighbor search (a pure content-based method) results may yield some unrelated images even the concept learning is very good. An alternative non-model method memorizing the re-

lationship between each pair of database images (e.g., [7]) may yield better retrieval precision, especially for fixed image database. However, such method is only the accumulation and exploitation of labeling information, instead of exploring visual features systematically. Furthermore, when the query is non-database image (which is a very common application scenario), or the database is dynamic that new images are added, the information of pairwise image relationship cannot be exploited to improve retrieval precision. Our mixture-model-based concept learning approach attempts to systematically and fully explore and exploit visual features, so that the model knowledge can help the retrieval even for the cases such as non-database query and dynamic database. More significantly, the learned concept knowledge from an image database may have more general application possibilities such that it can be used for another database.

The goal of our system is to learn concepts to the best extent given a certain amount retrieval experiences, instead of achieving best retrieval precision after obtaining enough amount of retrieval experience. Since users may not be so cooperative to provide labeling information by executing relevance feedbacks, it is difficult for the system to obtain many retrieval experiences during a certain time of the database. Thus, our learning scenario is more realistic compared with the system which expects to accumulate enough amount of retrieval experiences before good retrieval performance is obtained.

4 Discussions and conclusions

A semantic concept may contain multiple components, which are far away from each other in feature space. Our purpose is to estimate every single component for visual concept, which has no conflict with the multiple component assumption for a semantic concept.

An image may belong to multiple concepts. However, images can only be stored on hard drive in a single way, and the images belonging to the same classes should be stored consecutively to minimize the hard drive page accesses. Thus, an image should be assigned to the class if the majority of people support this assignment; thus, it will satisfy most of the users when they enter queries. For the users being not satisfied, they may implement relevant feedback to get more images they need.

In this paper, we explored users' labeling noise and outliers for concept learning in image database: (1) retrieval experiences derived from previous users' relevance feedbacks are efficiently and correctly processed using Bayesian analysis, (2) outliers can be detected and rejected. We integrated these topics into a novel semi-supervised EM algorithm to achieve satisfactory concept learning. A visual-concept-based indexing structure is proposed based on the concept learning result, and it helps to improve retrieval performance in two aspects: the retrieval precision is improved, and the search time is saved.

References

[1] G. McLachlan and D. Peel, *Finite Mixture Models*, John Wiley & Sons, 2000.

[2] N. Vasconcelos, *Bayesian Models for Visual Information Retrieval*, Ph.D thesis, MIT, 2000.

[3] K. Barnard and D. Forsyth, "Learning the semantics of words and pictures," *Proc. IEEE Int'l Conf. Computer Vision*, vol. 2, pp. 408–415, 2001.

[4] F. Qian, M. Li, L. Zhang, H.J. Zhang, and B. Zhang, "Gaussian mixture model for relevance feedback in image retrieval," *Proc. IEEE Int'l Conf. Multimedia & Expo*, vol. 1, pp. 229–232, 2002.

[5] D. Heisterkamp, "Building a latent semantic index of an image database from patterns of relevance feedback," *Proc. Int'l Conf. Pattern Recognition*, vol. 4, pp. 134–137, 2002.

[6] M. Li, Z. Chen, and H.J. Zhang, "Statistical correlation analysis in image retrieval," *Pattern Recognition*, vol. 35, no. 12, pp. 2687–2693, 2002.

[7] P. Yin, B. Bhanu, K. Chang, and A. Dong, "Improving retrieval performance by long-term relevance information," *Proc. Int'l Conf. on Pattern Recognition*, vol. III, pp. 533–536, August 2002.

[8] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra, "Relevance feedback: a power tool for interactive content-based image retrieval," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 8, no. 5, pp. 644–655, September 1998.

[9] Authors, "to be filled later," .

[10] R.N. Dave and R. Krishnapuram, "Robust clustering methods: a unified view," *IEEE Trans. on Fuzzy Systems*, vol. 5, no. 2, pp. 270–293, May 1997.

[11] R.N. Dave, "Characterization and detection of noise in clustering," *Pattern Recognition Lett.*, vol. 12, pp. 657–664, 1991.

[12] Y. Wu and T. S. Huang, "Towards self-exploring discriminating features for visual learning," *Engineering Applications of Artifical Intelligence*, vol. 15, pp. 139–150, April 2002.

[13] M. Figueiredo and A.K. Jain, "Unsupervised learning of finite mixture models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 381–396, March 2002.

[14] C. Bohm, S. Berchtold, and D. A. Keim, "Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases," *ACM Computing Surveys*, vol. 33, no. 3, pp. 322–373, September 2001.

[15] N. Vasconcelos, "Image indexing with mixture hierarchies," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 3–10, 2001.

[16] B. S. Manjunath and W. Y. Ma, "Texture feature for browsing and retrieval of image data," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, August 1996.