EXPLOITING GROUP STRUCTURE TO IMPROVE RETRIEVAL ACCURACY AND SPEED IN IMAGE DATABASES

Nuno Vasconcelos

HP Cambridge Research Laboratory (nuno.vasconcelos@hp.com)

ABSTRACT

Most image retrieval systems perform a linear search over the database to find the closest match to a query. However, databases usually exhibit a natural grouping structure into content classes that can be exploited to improve retrieval precision and speed. We investigate methods that enable search at both the class and image level. It is shown that, through the combination of Bayesian averaging and hierarchical density estimation, it is possible to achieve significant gains in retrieval accuracy and speed, at the cost of a marginal increase in training complexity. The technique is also shown to enable the efficient design of semantic classifiers.

1. INTRODUCTION

The problem of content-based retrieval from image or video databases is naturally formulated as one of statistical classification. Given a query feature space $\mathcal{X}$, the design of a retrieval system consists of finding a map

$$\begin{array}{rccc} g: & \mathcal{X} & \rightarrow & \mathcal{M} = \{1, \ldots, M\} \\ & \mathbf{x} & \rightarrow & y \end{array}$$

from $\mathcal{X}$ to the set $\mathcal{M}$ of image classes in the database. When the goal is to minimize the probability of error it is well known that the optimal solution is given by the Bayes classifier [1]

$$g^*(\mathbf{x}) = \arg\max_i P(y = i|\mathbf{x}). \quad (1)$$

The simplest way to define the image classes in the database is to assume that each image is a class on its own. This definition is implicitly adopted by any retrieval system that compares the query against all images in the database, and is therefore prevalent today.

However, it is usually the case that databases exhibit a natural grouping structure which can be exploited to improve retrieval precision and speed. For example, an object database may contain images of each object at different poses or scales, a database of scenic imagery may contain replicas of the same location shot at different times of the year, a database of industrial parts may contain images of several variations on each canonical part, and a video database typically contains many images for each shot. In all these examples, the database can be thought of as composed by a number of canonical image classes (objects, locations, canonical parts, and shots), each imaged under variable conditions (pose and scale, time of the year, part revision, and location within the shot).

Given a query image, searching for the closest canonical class is faster than searching for the closest image (because there are significantly less classes than images) and can also be more accurate (by reducing the probability of false positives). If the classification structure can be extended to multiple levels, i.e. classes grouped into meta-classes and so forth, the gains in retrieval speed can be quite significant [2]. On the other hand, there are applications where retrieving one representative image from the same class as the query is not satisfactory, e.g. an object recognition task where, in addition to the object, it is important to recognize its pose. Hence, retrieval systems should be able to support both types of search and, ideally, image-level searches should be an extension of class-level ones (by finding the best image match within the best class match).

The straightforward manner to enable searches at the class level is to rely on Bayes rule

$$g^*(\mathbf{x}) = \arg\max_i P(\mathbf{x}|y = i)P(y = i) \quad (2)$$

and *direct estimation* of the canonical class densities $P(\mathbf{x}|y = i)$ (using for each class a training set consisting of all the feature vectors extracted from all the images in that class). This approach has two limitations. First, assuming that queries at the image level also have to be supported, the retrieval system needs to estimate both the class density and the density of each of its images. This means that there will be a significant increase in training cost over that required by image-level search only. Second, the resulting class densities cannot be updated incrementally, whenever images are added to or deleted from the database. While these are not overwhelming problems for histogram-based estimates, they can be a major hurdle when more sophisticated density models are used. We have previously shown that the use of these more sophisticated models, namely Gauss mixtures, can lead to significant improvements in retrieval accuracy over that achieved with histograms [3].

In this paper, we investigate more efficient solutions to enable searches at both the class and image level. One possibility is to introduce a hidden variable h that accounts for each of the possible *imaging hypotheses* within a class. Bayesian principles can then be used to fuse information from the various hypotheses, when searching at the class level. In this way, densities only need to be estimated once, at the image level, and there is no training cost associated with class-level search. We will see, however, that this solution leads to an increase in the evaluation of (2) that cancels the computational savings of class-level search. Or, in other words, improving the training efficiency destroys that of retrieval.

To achieve simultaneous training and retrieval efficiency, the density model must reflect the hierarchical structure inherent to the fact that images are feature subsets of image classes. In any sensible feature space, this implies that image densities from the same class will have shared components, and the overall class density will be the mixture of all such components. Such structure can be represented formally as a mixture hierarchy [2]. We show that combining Bayesian fusion with mixture hierarchies restores the computational savings of class-level search, with a minimal increase in training complexity over that of simple Bayesian fusion.

Furthermore, due to a data-driven form of regularization inherent to hierarchical density estimation, hierarchical estimates have better generalization and lead to higher retrieval accuracy than simple fusion. Finally, because hierarchical estimation is very efficient, it provides a practical solution for the design of semantic classifiers in the context of large image databases.

2. BAYESIAN HYPOTHESIS FUSION

We denote by h the hidden variable whose states are associated with different hypothesis for image interpretation. For example, in the case of an object database where each object appears in one of P poses, $h \in \{1, \ldots, P\}$. The posterior probabilities of (1) can then be written as

$$\begin{aligned}
P(y=i|\mathbf{x}) = & \\
&= \sum_j P(y=i, h=j|\mathbf{x}) \\
&= \sum_j P(y=i|h=j,\mathbf{x}) P(h=j|\mathbf{x}) \\
&= \sum_j \frac{P(\mathbf{x}|y=i,h=j)P(y=i|h=j)}{P(\mathbf{x}|h=j)} \times \\
&\quad \times \frac{P(\mathbf{x}|h=j)P(h=j)}{P(\mathbf{x})} \\
&= \sum_j \frac{P(\mathbf{x}|y=i,h=j)P(y=i|h=j)P(h=j)}{P(\mathbf{x})} \\
&= \frac{P(y=i)}{P(\mathbf{x})} \sum_j P(\mathbf{x}|y=i,h=j)P(h=j|y=i) \\
&\propto P(y=i) \sum_j P(\mathbf{x}|y=i,h=j)P(h=j|y=i) \\
&= P(y=i) \sum_j P(\mathbf{x}|y=i,h=j)\alpha_{j|i} \qquad (3)
\end{aligned}$$

where $\alpha_{j|i} = P(h=j|y=i)$ and the normalization constant $P(\mathbf{x})$ is irrelevant for all purposes related to (1), since it does not depend on i. It is, therefore, clear that the optimal way to integrate the information from the different hypothesis is to take a weighted average of the likelihood of the query $\mathbf{x}$ under each of them. This is to be contrasted to the standard *linear search* for the closest image in the database, that can be expressed as

$$g(\mathbf{x}) = \arg\max_{i,j} P(\mathbf{x}|y=i,h=j). \qquad (4)$$

In terms of practical implementation, (3) can be computed in two ways. The first is to compute the likelihood of the query under each of the individual image models $P(\mathbf{x}|y=i,h=j)$ and then average across hipothesis. Like (4), this has computational cost $O(MP)$, i.e. linear in the size of the database. The second is to compute, off-line, the average model for each class

$$P(\mathbf{x}|y=i) = \sum_j P(\mathbf{x}|y=i,h=j)\alpha_{j|i} \qquad (5)$$

and then apply (2) directly. Under this implementation the search complexity is $O(M)$ and can be significantly smaller, depending on the number of hipotheses P, than (4). Hence, model averaging is the most appealing solution from a computational standpoint.

3. MODEL AVERAGING

Model averaging according to (5) can be performed in several ways. The direct application of (5) is feasible when the densities $P(\mathbf{x}|y=i,h=j)$ are defined over a (common) partition of the space. For example, if all densities are histograms defined on a partition of $\mathcal{X}$ into Q cells $\{\mathcal{X}_q\}, q=1,\ldots,Q$, $h^q_{i,j}$ the number of feature vectors from class i that land on cell $\mathcal{X}_q$ under hypothesis j, then the average class histogram is simply

$$\hat{h}^q_i = \sum_j h^q_{i,j}\alpha_{j|i}.$$

However, when 1) the underlying partition is not the same for all histograms or 2) more sophisticated models (e.g. mixture or kernel density estimates) are used model averaging is not as simple.

3.1. Naive averaging

Consider, for example, the Gauss mixture model

$$P(\mathbf{x}|y=i,h=j) = \sum_k \pi^k_{i,j}\mathcal{G}(\mathbf{x},\mu^k_{i,j},\Sigma^k_{i,j}) \qquad (6)$$

where $\mathcal{G}(\mathbf{x},\mu,\Sigma)$ is a Gaussian probability density function with mean μ and covariance $\mathbf{\Sigma}$, and $\pi^k_{i,j}$ a probability mass function such that $\sum_k \pi^k_{i,j} = 1$. Direct application of (5) leads to

$$P(\mathbf{x}|y=i) = \sum_{j,k} \alpha_{j|i}\pi^k_{i,j}\mathcal{G}(\mathbf{x},\mu^k_{i,j},\Sigma^k_{i,j}), \qquad (7)$$

i.e. a P-fold increase in the number of Gaussian components per mixture that cancels the computational reduction from $O(MP)$ to $O(M)$ achieved by model averaging. We denote this solution by *naive averaging*, since it has no real advantage over evaluating the query likelihood for each individual image model and then averaging the resulting likelihoods.

3.2. Mixture hierarchies

The problem of naive model averaging is that there is no explicit modeling of the relationships between image densities in the same class. Consider the trivial example of Figure 1, where we present the densities associated with three images of an object with four faces, each textured in a way that gives rise to a Gaussian in feature space. Due to the geometry of the object, only three of the four faces are visible in any image. The effective number of components at the class level is therefore 4 (the total number of faces in the object) and significantly smaller than the total number of components in all images, which is 12. By failing to take this into account, (7) requires the evaluation of 12 components, most of which are replicas of each other.

We will refer to a model that captures the relationships exemplified in Figure 1 as a mixture hierarchy. Roughly speaking, this is a collection of mixtures, organized hierarchically, where children densities consist of different combinations of subsets of the parents' components. A formal definition is also possible [4], but we omit the details for brevity. The important point is that, when the densities conform to the mixture hierarchy model, it is possible to estimate the parameters of the class mixture directly from those available for the individual image mixtures, using a two-stage procedure. The first stage, is the naive averaging of (7). Assuming

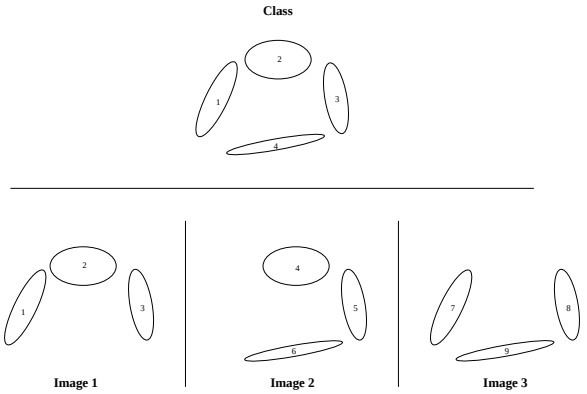

Fig. 1. A simple mixture hierarchy with three images. The class model is shown at the top, image models are shown at the bottom.

that each image mixture has K components, this leads to an overall mixture with PK components and parameters[1]

$$\{\pi_j^k, \mu_j^k, \Sigma_j^k\}, j = 1, \ldots, P, k = 1, \ldots, K. \quad (8)$$

The second is an extension of the EM algorithm, which clusters the Gaussian components into a T-component mixture [5, 2], where T is the number of components at the class level. Denoting by $\{\pi_c^t, \mu_c^t, \Sigma_c^t\}, t = 1, \ldots, T$ the parameters of the class mixture, this algorithm iterates between the following steps.
E-step: compute

$$h_{jk}^t = \frac{\left[\mathcal{G}(\mu_j^k, \mu_c^t, \mathbf{\Sigma}_c^t) e^{-\frac{1}{2} trace\{(\mathbf{\Sigma}_c^t)^{-1}\mathbf{\Sigma}_j^k\}}\right]^{\pi_j^k N} \pi_c^t}{\sum_l \left[\mathcal{G}(\mu_j^k, \mu_c^l, \mathbf{\Sigma}_c^l) e^{-\frac{1}{2} trace\{(\mathbf{\Sigma}_c^l)^{-1}\mathbf{\Sigma}_j^k\}}\right]^{\pi_j^k N} \pi_c^l} \quad (9)$$

where N is a user-defined parameter (see [5] for details).
M-step: set

$$(\pi_c^t)^{new} = \frac{\sum_{jk} h_{jk}^t}{PK} \quad (10)$$

$$(\mu_c^t)^{new} = \sum_{jk} w_{jk}^t \mu_j^k, \text{ where } w_{jk}^t = \frac{h_{jk}^t \pi_j^k}{\sum_{jk} h_{jk}^t \pi_j^k} \quad (11)$$

$$(\mathbf{\Sigma}_c^t)^{new} = \sum_{jk} w_{jk}^t \left[\mathbf{\Sigma}_j^k + (\mu_j^k - \mu_c^t)(\mu_j^k - \mu_c^t)^T\right] \quad (12)$$

Notice that the number of parameters in each image mixture is orders of magnitude smaller than the number of feature vectors in the image itself. Hence the complexity of estimating the class mixture parameters is negligible when compared to that of estimating the individual mixture parameters for all images in the class. It follows that the overall training complexity is dominated by the latter task, i.e. only marginally superior to that of naive averaging and significantly smaller than that associated with direct estimation of class densities. On the other hand, the retrieval complexity is exactly the same as that achievable with direct estimation, and significantly smaller than that of naive averaging. This is quantified on Table 1.

[1]Note that this notation is the same as that of (6) with the class index i omitted.

Table 1. Time complexity of class-level parameter estimation vs. retrieval complexity for the methods discussed in the paper. All times are per class, i.e. must be multiplied by the number of classes M to obtain total costs. P is the number of image mixtures per class, K the number of components in each of these mixtures, S the number of feature vectors per image, and T the number of class mixture components. Typically, $T \approx K \ll S$.

	Training time	Retrieval time
Linear search	-	O(PK)
Naive averaging	-	O(PK)
Direct estimation	O(TPS)	O(T)
Mixture hierarchy	O(TPK)	O(T)

One final interesting property of the EM steps above is that they enforce a data-driven form of regularization which improves generalization. This regularization is visible in (12) where the variances on the left hand-size can never be smaller than those on the right-hand side. We will see in the next section that, due to this property, hierarchical class density estimates are much more reliable than those obtained with direct learning.

4. EXPERIMENTAL EVALUATION

In this section we present results of an experimental comparison of the four retrieval methods discussed above: linear search, class search with naive averaging, class search with direct density estimation, and hierarchical class search. The evaluation was performed on the the Columbia object database (COIL-100), which is a common benchmark for object recognition and image retrieval. Our version of the database is a set of images from 100 objects each shot in 9 different views obtained by rotating the object in $3D$ in steps of 40^o. This subset was split into two subgroups, a *query* database containing the first image of each object and a *retrieval* database containing the remaining 8. In all cases, the YBR color space was used, and the image features were the coefficients of the 8×8 discrete cosine transform. Standard EM was then used to estimate a Gaussian mixture for each image. For the class-based searches, each object was considered as a different class.

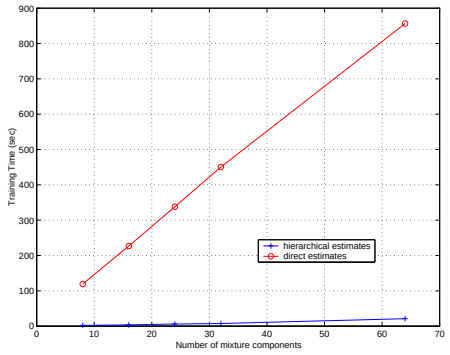

Fig. 2. Training complexity as a function of the number of class-level mixture components.

Figure 2 presents the times required to learn the class models with direct and hierarchical density estimation. While the compu-

Table 2. Confusion matrix for Corel. Entry i, j gives the fraction of the queries in which a query from class i was assigned to class j.

	AH	AR	O	R	S	SG	SS	C	D	E	EG	Fw	GM	MR	OP
AH	**100**	0	0	0	0	0	0	0	0	0	0	0	0	0	0
AR	0	**96**	0	0	0	0	0	0	0	0	**4**	0	0	0	0
O	0	0	**88**	0	0	0	0	0	0	0	0	0	0	**12**	0
R	0	0	0	**100**	0	0	0	0	0	0	0	0	0	0	0
S	0	0	0	0	**91**	0	**3**	0	0	0	**3**	0	0	**3**	0
SG	0	0	0	0	0	**72**	**6**	0	**6**	**5**	0	**1 1**	0	0	0
SS	0	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0
C	0	0	0	0	0	0	0	**90**	0	0	0	0	**10**	0	0
D	0	0	0	0	0	0	0	0	**100**	0	0	0	0	0	0
E	**4**	0	0	0	0	0	**4**	**4**	0	**64**	0	0	£4	**17**	**3**
EG	0	0	0	0	0	0	0	0	0	0	**100**	0	0	0	0
Fw	0	0	**7**	0	0	0	0	0	0	**7**	0	**80**	0	0	**6**
GM	0	0	0	0	0	0	0	**13**	0	0	0	0	**47**	**40**	0
MR	0	0	0	0	0	0	0	0	0	**5**	0	0	0	**95**	0
OP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**100**

tational savings depend on factors such as the number of images per class and the number of mixture components (see table 1) it is clear that the gains of the hierarchical approach can be quite significant even when these numbers are small. For example, its cost is about 40 times smaller than that of direct estimation (21.1 seconds per model for the former vs. 14.2 minutes for the latter) when the class model has 64 components and there are only 8 images per class.

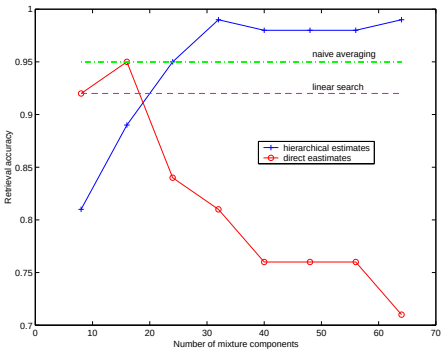

Fig. 3. Retrieval accuracy as a function of the number of class-level mixture components.

On the other hand, Figure 3 shows that, given enough mixture components, the retrieval performance achieved with mixture hierarchies is clearly superior to that of direct estimation. The improvements can be quite significant: while the latter achieves 95% accuracy but is very sensitive to variations in the number of mixture components, the former achieves 99% accuracy and is very robust. The figure also shows that queries using mixture hierarchies perform significantly better than the significantly more complex linear searches or class searches based on naive averaging. The high accuracy and robustness of searches based on mixture hierarchies are a consequence of more accurate density estimates, due to the data-driven regularization discussed in section 3.2.

5. SEMANTIC CLASSIFIERS

We finalize, by discussing the use of mixture hierarchies as a means to design semantic classifiers [6, 7]. This design can be carried out in two steps. The first is to estimate a mixture density for each image in the database. Since such estimates are required for elementary retrieval operations it does not add any complexity to a retrieval system. Given a labeled collection of images (e.g. outdoors vs. indoors), the second step is then to estimate class models directly from the associated image models. Due to the efficiency of hierarchical estimation this step can be performed very quickly. Hence, once the individual image densities are available, it is almost trivial to add higher-level classifiers to the retrieval system[2].

To evaluate the feasibility of this idea, we tried it on 15 classes[3] from the Corel image database, with 100 images each. Once again we created a query and retrieval database, this time by assigning each image to the query set with a probability 0.2. All other parameters were the same as in the experiments of the previous section. Table 2 presents the resulting confusion matrix. For 11 of the 15 classes[4], the recognition accuracy was 88% or better and 8 of these had accuracy of 95% or better. Errors in high error-rate classes usually coincided with queries that, from the standpoint of visual appearance, were similar to two or more classes. Typically, higher-level forms of scene interpretation would be required to correct these errors. For example, pictures of Egyptian monuments were frequently confused for pictures of Mayan monuments, pictures of religious stained glass were sometimes confused for pictures of fireworks, and pictures of glaciers and mountains were commonly mistaken for scenes of coasts or landscapes containing Mayan monuments.

6. REFERENCES

[1] L. Devroye, L. Gyorfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer-Verlag, 1996.

[2] N. Vasconcelos, "Image Indexing with Mixture Hierarchies," in *Proc. IEEE Computer Vision and Pattern Recognition Conf.*, Kawai, Hawaii, 2001.

[3] N. Vasconcelos and A. Lippman, "A Probabilistic Architecture for Content-based Image Retrieval," in *Proc. IEEE Computer Vision and Pattern Recognition Conf.*, Hilton Head, North Carolina, 2000.

[4] N. Vasconcelos, ""Learning Models from Models" for Generic Visual Recognition," Submitted for publication.

[5] N. Vasconcelos and A. Lippman, "Learning Mixture Hierarchies," in *Neural Information Processing Systems, Denver, Colorado*, 1998.

[6] Martin Szummer and Rosalind Picard, "Indoor-Outdoor Image Classification," in *Workshop in Content-based Access to Image and Video Databases*, 1998, Bombay, India.

[7] A. Vailaya, M. Figueiredo, A. Jain, and H. Zhang, "Image Classification for Content-Based Indexing," *IEEE Trans. on Image Processing*, vol. 10, no. 1, pp. 117–130, January 2001.

[2] Assuming, of course, that image labels are available.

[3] See Table 2 for a class list.

[4] Class labels are AH for Arabian horses, AR for Auto racing, O for Owls, R for Roses, S for Ski scenes, SG for religious stained glass, SS for sunsets and sunrises, C for coasts, D for Divers and diving, E for Land of the pyramids (pictures of Egypt), EG for English country gardens, Fw for fireworks, GM for Glaciers and mountains, MR for Mayan and Aztec ruins, and OP for Oil Paintings.